SIEMENS

SIMATIC

S7
S7-1200 Programmable controller

System Manual

Preface

Product overview 1

STEP 7 programming software 2

Installation 3

PLC concepts 4

Device configuration 5

Programming concepts 6

Basic instructions 7

Extended instructions 8

Technology instructions 9

Communication 10

Web server 11

Communication processor 12

Teleservice communication (SMTP email) 13

Online and diagnostic tools 14

Technical specifications A

Calculating a power budget B

Order numbers C

04/2012

A5E02486680-06

Legal information

Warning notice system

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

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

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

⚠ CAUTION
with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken.

CAUTION
without a safety alert symbol, indicates that property damage can result if proper precautions are not taken.

NOTICE
indicates that an unintended result or situation can occur if the relevant information is not taken into account.

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

Qualified Personnel

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

Proper use of Siemens products

Note the following:

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

Trademarks

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

Disclaimer of Liability

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

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
GERMANY

Order number: 6ES7298-8FA30-8BH0
Ⓟ 05/2012 Technical data subject to change

Copyright © Siemens AG 2012.
All rights reserved

Preface

Purpose of the manual

The S7-1200 series is a line of programmable logic controllers (PLCs) that can control a variety of automation applications. Compact design, low cost, and a powerful instruction set make the S7-1200 a perfect solution for controlling a wide variety of applications. The S7-1200 models and the Windows-based programming tool give you the flexibility you need to solve your automation problems.

This manual provides information about installing and programming the S7-1200 PLCs and is designed for engineers, programmers, installers, and electricians who have a general knowledge of programmable logic controllers.

Required basic knowledge

To understand this manual, it is necessary to have a general knowledge of automation and programmable logic controllers.

Scope of the manual

This manual describes the following products:

- STEP 7 V11 Basic and Professional
- S7-1200 CPU firmware release V3.0

For a complete list of the S7-1200 products described in this manual, refer to the technical specifications (Page 699).

Certification, CE label, C-Tick, and other standards

Refer to the technical specifications (Page 699) for more information.

Service and support

In addition to our documentation, we offer our technical expertise on the Internet on the customer support web site (http://www.siemens.com/automation/).

Contact your Siemens distributor or sales office for assistance in answering any technical questions, for training, or for ordering S7 products. Because your sales representatives are technically trained and have the most specific knowledge about your operations, process and industry, as well as about the individual Siemens products that you are using, they can provide the fastest and most efficient answers to any problems you might encounter.

Documentation and information

S7-1200 and STEP 7 provide a variety of documentation and other resources for finding the technical information that you require.

- The S7-1200 system manual provides specific information about the operation, programming and the specifications for the complete S7-1200 product family. In addition to the system manual, the S7-1200 Easy Book provides a more general overview to the capabilities of the S7-1200 family.

  Both the system manual and the Easy Book are available as electronic (PDF) and printed manuals. The electronic manuals can be downloaded from the customer support web site and can also be found on the companion disk that ships with every S7-1200 CPU.

- The online information system of STEP 7 provides immediate access to the conceptual information and specific instructions that describe the operation and functionality of the programming package and basic operation of SIMATIC CPUs.

- My Documentation Manager accesses the electronic (PDF) versions of the SIMATIC documentation set, including the system manual, the Easy Book and the information system of STEP 7. With My Documentation Manager, you can drag and drop topics from various documents to create your own custom manual.

  The customer support entry portal (http://support.automation.siemens.com) provides a link to My Documentation Manager under mySupport.

- The customer support web site also provides podcasts, FAQs, and other helpful documents for S7-1200 and STEP 7. The podcasts utilize short educational video presentations that focus on specific features or scenarios in order to demonstrate the interactions, convenience and efficiency provided by STEP 7. Visit the following web sites to access the collection of podcasts:

  – STEP 7 Basic web page (http://www.automation.siemens.com/mcms/simatic-controller-software/en/step7/step7-basic/Pages/Default.aspx)

  – STEP 7 Professional web page (http://www.automation.siemens.com/mcms/simatic-controller-software/en/step7/step7-professional/Pages/Default.aspx)

- You can also follow or join product discussions on the Service & Support technical forum (https://www.automation.siemens.com/WW/forum/guests/Conferences.aspx?Language=en&siteid=csius&treeLang=en&groupid=4000002&extranet=standard&viewreg=WW&nodeid0=34612486). These forums allow you to interact with various product experts.

  – Forum for S7-1200 (https://www.automation.siemens.com/WW/forum/guests/Conference.aspx?SortField=LastPostDate&SortOrder=Descending&ForumID=258&Language=en&onlyInternet=False)

  – Forum for STEP 7 Basic (https://www.automation.siemens.com/WW/forum/guests/Conference.aspx?SortField=LastPostDate&SortOrder=Descending&ForumID=265&Language=en&onlyInternet=False)

Table of contents

Preface 3

1 Product overview 19

1.1 Introducing the S7-1200 PLC 19

1.2 Expansion capability of the CPU 22

1.3 S7-1200 modules 24

1.4 New features 25

1.5 Basic HMI panels 26

2 STEP 7 programming software 29

2.1 System requirements 29

2.2 Different views to make the work easier 30

2.3 Easy-to-use tools 31
2.3.1 Inserting instructions into your user program 31
2.3.2 Accessing instructions from the "Favorites" toolbar 32
2.3.3 Creating a complex equation with a simple instruction 33
2.3.4 Adding inputs or outputs to a LAD or FBD instruction 35
2.3.5 Expandable instructions 35
2.3.6 Selecting a version for an instruction 36
2.3.7 Modifying the appearance and configuration of STEP 7 36
2.3.8 Dragging and dropping between editors 37
2.3.9 Changing the operating mode of the CPU 37
2.3.10 Changing the call type for a DB 39
2.3.11 Temporarily disconnecting devices from a network 40
2.3.12 Virtual unplugging of devices from the configuration 41

3 Installation 43

3.1 Guidelines for installing S7-1200 devices 43

3.2 Power budget 44

3.3 Installation and removal procedures 46
3.3.1 Mounting dimensions for the S7-1200 devices 46
3.3.2 Installing and removing the CPU 49
3.3.3 Installing and removing an SB, CB or BB 51
3.3.4 Installing and removing an SM 52
3.3.5 Installing and removing a CM or CP 53
3.3.6 Removing and reinstalling the S7-1200 terminal block connector 55
3.3.7 Installing and removing the expansion cable 56
3.3.8 TS (teleservice) adapter 57
3.3.8.1 Connecting the TeleService Adapter 57
3.3.8.2 Installing the SIM card 58
3.3.8.3 Installing the TS adapter unit 59
3.3.8.4 Installing the TS adapter on a wall 60

3.4 Wiring guidelines ... 61

4 PLC concepts ... 67

4.1 Execution of the user program ... 67
4.1.1 Operating modes of the CPU ... 69
4.1.2 Processing the scan cycle in RUN mode ... 72
4.1.3 Organization blocks (OBs) ... 73
4.1.4 Event execution priorities and queuing ... 75
4.1.5 Monitoring the cycle time ... 80
4.1.6 CPU memory ... 82
4.1.6.1 System and clock memory ... 84
4.1.7 Diagnostics buffer ... 86
4.1.8 Time of day clock ... 86
4.1.9 Configuring the outputs on a RUN-to-STOP transition ... 87

4.2 Data storage, memory areas, I/O and addressing ... 87
4.2.1 Accessing the data of the S7-1200 ... 87

4.3 Processing of analog values ... 92

4.4 Data types ... 93
4.4.1 Bool, Byte, Word, and DWord data types ... 94
4.4.2 Integer data types ... 95
4.4.3 Floating-point real data types ... 95
4.4.4 Time and Date data types ... 96
4.4.5 Character and String data types ... 97
4.4.6 Array data type ... 99
4.4.7 Data structure data type ... 100
4.4.8 PLC data type ... 100
4.4.9 Pointer data types ... 101
4.4.9.1 "Pointer" pointer data type ... 101
4.4.9.2 "Any" pointer data type ... 102
4.4.9.3 "Variant" pointer data type ... 103
4.4.10 Accessing a "slice" of a tagged data type ... 104
4.4.11 Accessing a tag with an AT overlay ... 105

4.5 Using a memory card ... 107
4.5.1 Inserting a memory card in the CPU ... 108
4.5.2 Configuring the startup parameter of the CPU before copying the project to the memory card ... 110
4.5.3 Transfer card ... 110
4.5.4 Program card ... 112
4.5.5 Firmware update ... 115

4.6 Recovery from a lost password ... 118

5 Device configuration ... 119

5.1 Inserting a CPU ... 120

5.2 Detecting the configuration for an unspecified CPU ... 121

5.3 Adding modules to the configuration ... 122

5.4 Configuring the operation of the CPU ... 123

5.5 Configuring the parameters of the modules ... 125

5.6 Configuring the CPU for communication 126
5.6.1 Creating a network connection 126
5.6.2 Configuring the Local/Partner connection path....127
5.6.3 Parameters for the PROFINET connection 129
5.6.4 Assigning Internet Protocol (IP) addresses 132
5.6.4.1 Assigning IP addresses to programming and network devices 132
5.6.4.2 Checking the IP address of your programming device 134
5.6.4.3 Assigning an IP address to a CPU online 134
5.6.4.4 Configuring an IP address for a CPU in your project....136
5.6.5 Testing the PROFINET network 139
5.6.6 Locating the Ethernet (MAC) address on the CPU....140
5.6.7 Configuring Network Time Protocol synchronization 141
5.6.8 PROFINET device start-up time, naming, and address assignment 142

6 Programming concepts 145

6.1 Guidelines for designing a PLC system 145

6.2 Structuring your user program 146

6.3 Using blocks to structure your program 148
6.3.1 Organization block (OB)....148
6.3.2 Function (FC) 150
6.3.3 Function block (FB)....150
6.3.4 Data block (DB)....151

6.4 Understanding data consistency....153

6.5 Programming language....154
6.5.1 Ladder logic (LAD) 155
6.5.2 Function Block Diagram (FBD) 156
6.5.3 SCL 156
6.5.4 EN and ENO for LAD, FBD and SCL....163

6.6 Protection 164
6.6.1 Access protection for the CPU....164
6.6.2 Know-how protection 165
6.6.3 Copy protection 166

6.7 Downloading the elements of your program 168

6.8 Uploading from the CPU 168
6.8.1 Copying elements of the project 168
6.8.2 Using the compare function 170

6.9 Debugging and testing the program 170
6.9.1 Monitor and modify data in the CPU....170
6.9.2 Watch tables and force tables....170
6.9.3 Cross reference to show usage 171
6.9.4 Call structure to examine the calling hierarchy 172

7 Basic instructions 175

7.1 Bit logic....175
7.1.1 Bit logic contacts and coils....175
7.1.2 Set and reset instructions 178
7.1.3 Positive and negative edge instructions 180

7.2 Timers 182

7.3 Counters ... 190

7.4 Compare ... 196
7.4.1 Compare ... 196
7.4.2 In-range and Out-of-range instructions ... 197
7.4.3 OK and Not OK instructions ... 197

7.5 Math ... 198
7.5.1 Calculate instruction ... 198
7.5.2 Add, subtract, multiply and divide instructions ... 199
7.5.3 Modulo instruction ... 200
7.5.4 Negation instruction ... 201
7.5.5 Increment and decrement instructions ... 202
7.5.6 Absolute value instruction ... 202
7.5.7 Minimum and Maximum instructions ... 203
7.5.8 Limit instruction ... 204
7.5.9 Floating-point math instructions ... 205

7.6 Move ... 207
7.6.1 Move and block move instructions ... 207
7.6.2 FieldRead and FieldWrite instructions ... 209
7.6.3 Fill instructions ... 211
7.6.4 Swap instruction ... 212

7.7 Convert ... 213
7.7.1 CONV instruction ... 213
7.7.2 Conversion instructions for SCL ... 214
7.7.3 Round and truncate instructions ... 217
7.7.4 Ceiling and floor instructions ... 218
7.7.5 Scale and normalize instructions ... 219

7.8 Program control ... 222
7.8.1 Overview of SCL program control statements ... 222
7.8.2 IF-THEN statement ... 223
7.8.3 CASE statement ... 224
7.8.4 FOR statement ... 225
7.8.5 WHILE-DO statement ... 226
7.8.6 REPEAT-UNTIL statement ... 227
7.8.7 CONTINUE statement ... 227
7.8.8 EXIT statement ... 228
7.8.9 GOTO statement ... 229
7.8.10 RETURN statement ... 229
7.8.11 Jump and label instructions ... 230
7.8.12 JMP_LIST instruction ... 230
7.8.13 SWITCH instruction ... 231
7.8.14 RET execution control instruction ... 233
7.8.15 Re-trigger scan cycle watchdog instruction ... 234
7.8.16 Stop scan cycle instruction ... 235
7.8.17 Get Error instructions ... 235

7.9 Word logic operations ... 239
7.9.1 AND, OR, and XOR instructions ... 239
7.9.2 Invert instruction ... 240
7.9.3 Encode and decode instructions ... 240
7.9.4 Select, Multiplex, and Demultiplex instructions ... 242

7.10 Shift and Rotate 244
7.10.1 Shift instructions 244
7.10.2 Rotate instructions 245

8 Extended instructions 247

8.1 Date and time-of-day 247
8.1.1 Date and time instructions 247
8.1.2 Set and read system clock 249
8.1.3 Run-time meter instruction 251
8.1.4 SET_TIMEZONE instruction 252

8.2 String and character 254
8.2.1 String data overview 254
8.2.2 S_MOVE instruction 254
8.2.3 String conversion instructions 255
8.2.3.1 String to value and value to string conversions 255
8.2.3.2 String-to-characters and characters-to-string conversions 263
8.2.3.3 ASCII to Hex and Hex to ASCII conversions 265
8.2.4 String operation instructions 267
8.2.4.1 LEN 267
8.2.4.2 CONCAT 268
8.2.4.3 LEFT, RIGHT, and MID 269
8.2.4.4 DELETE 270
8.2.4.5 INSERT 271
8.2.4.6 REPLACE 272
8.2.4.7 FIND 273

8.3 Distributed I/O (PROFINET, PROFIBUS, or AS-i) 274
8.3.1 Distributed I/O Instructions 274
8.3.2 RDREC and WRREC 275
8.3.3 RALRM 278
8.3.4 STATUS parameter for RDREC, WRREC, and RALRM 280
8.3.5 DPRD_DAT and DPWR_DAT 284
8.3.6 DPNRM_DG 286

8.4 Interrupts 288
8.4.1 Attach and detach instructions 288
8.4.2 Cyclic interrupts 291
8.4.2.1 SET_CINT (Set cyclic interrupt) 291
8.4.2.2 QRY_CINT (Query cyclic interrupt) 293
8.4.3 Time delay interrupts 294
8.4.4 Asynchronous event interrupts 296

8.5 Diagnostics (PROFINET or PROFIBUS) 297
8.5.1 Diagnostic instructions 297
8.5.2 Diagnostic events for distributed I/O 297
8.5.3 LED instruction 298
8.5.4 DeviceStates instruction 299
8.5.5 ModuleStates instruction 301
8.5.6 GET_DIAG instruction 302

8.6 Pulse 309
8.6.1 CTRL_PWM instruction 309
8.6.2 Operation of the pulse outputs 311
8.6.3 Configuring a pulse channel for PWM 312

8.7 Data logging ... 313
8.7.1 Data log record structure ... 314
8.7.2 Program instructions that control Data logs ... 315
8.7.2.1 DataLogCreate ... 315
8.7.2.2 DataLogOpen ... 318
8.7.2.3 DataLogClose ... 319
8.7.2.4 DataLogWrite ... 320
8.7.2.5 DataLogNewFile ... 322
8.7.3 Working with data logs ... 324
8.7.4 Limits to the size of data log files ... 325
8.7.5 Data log example program ... 327

8.8 Data block control ... 332
8.8.1 READ_DBL, WRIT_DBL (Read from or write to a DB in load memory) ... 332

8.9 Common error codes for the "Extended" instructions ... 335

9 Technology instructions ... 337

9.1 High-speed counter ... 337
9.1.1 Operation of the high-speed counter ... 339
9.1.2 Configuration of the HSC ... 345

9.2 PID control ... 346
9.2.1 Inserting the PID instruction and technological object ... 348
9.2.2 PID_Compact instruction ... 350
9.2.3 PID_Compact instruction ErrorBit parameters ... 354
9.2.4 PID_3STEP instruction ... 355
9.2.5 PID_3STEP instruction ErrorBit parameters ... 362
9.2.6 Configuring the PID controller ... 363
9.2.7 Commissioning the PID controller ... 365

9.3 Motion control ... 366
9.3.1 Configuring the axis ... 370
9.3.2 Configuring the TO_CommandTable_PTO ... 372
9.3.3 Motion control instructions ... 375
9.3.3.1 MC_Power instruction ... 375
9.3.3.2 MC_Reset instruction ... 378
9.3.3.3 MC_Home instruction ... 379
9.3.3.4 MC_Halt instruction ... 381
9.3.3.5 MC_MoveAbsolute instruction ... 383
9.3.3.6 MC_MoveRelative instruction ... 385
9.3.3.7 MC_MoveVelocity instruction ... 387
9.3.3.8 MC_MoveJog instruction ... 390
9.3.3.9 MC_CommandTable instruction ... 392
9.3.3.10 MC_ChangeDynamic ... 394
9.3.4 Operation of motion control for S7-1200 ... 396
9.3.4.1 CPU outputs used for motion control ... 396
9.3.4.2 Hardware and software limit switches for motion control ... 397
9.3.4.3 Homing ... 401
9.3.4.4 Jerk limit ... 406
9.3.5 Commissioning ... 407
9.3.6 Monitoring active commands ... 410
9.3.6.1 Monitoring MC instructions with a "Done" output parameter ... 410
9.3.6.2 Monitoring the MC_Velocity instruction ... 414

9.3.6.3 Monitoring the MC_MoveJog instruction 418

10 Communication 423

10.1 Number of asynchronous communication connections supported 424

10.2 PROFINET 425
10.2.1 Local/Partner connection 425
10.2.2 Open user communication 427
10.2.2.1 Connection IDs for the PROFINET instructions 427
10.2.2.2 Protocols 430
10.2.2.3 Ad hoc mode 431
10.2.2.4 TCP and ISO on TCP 431
10.2.2.5 UDP 446
10.2.2.6 T_CONFIG 451
10.2.2.7 Common parameters for instructions 458
10.2.3 Communication with a programming device 460
10.2.3.1 Establishing the hardware communications connection 460
10.2.3.2 Configuring the devices 461
10.2.3.3 Assigning Internet Protocol (IP) addresses 462
10.2.3.4 Testing your PROFINET network 462
10.2.4 HMI-to-PLC communication 462
10.2.4.1 Configuring logical network connections between two devices 463
10.2.5 PLC-to-PLC communication 464
10.2.5.1 Configuring logical network connections between two devices 465
10.2.5.2 Configuring the Local/Partner connection path between two devices 465
10.2.5.3 Configuring transmit (send) and receive parameters 465
10.2.6 Configuring a CPU and PROFINET IO device 468
10.2.7 Diagnostics 471
10.2.8 Distributed I/O Instructions 472
10.2.9 Diagnostic instructions 472
10.2.10 Diagnostic events for distributed I/O 472

10.3 PROFIBUS 472
10.3.1 Communications modules PROFIBUS 474
10.3.1.1 Connecting to PROFIBUS 474
10.3.1.2 Communications services of the PROFIBUS CMs 474
10.3.1.3 Other properties of the PROFIBUS CMs 476
10.3.1.4 Configuration examples for PROFIBUS 477
10.3.2 Configuring a DP master and slave device 478
10.3.2.1 Adding the CM 1243-5 (DP master) module and a DP slave 478
10.3.2.2 Configuring logical network connections between two PROFIBUS devices 478
10.3.2.3 Assigning PROFIBUS addresses to the CM 1243-5 module and DP slave 479
10.3.3 Distributed I/O Instructions 480
10.3.4 Diagnostic instructions 480
10.3.5 Diagnostic events for distributed 481

10.4 AS-i 481
10.4.1 Configuring an AS-i master and slave device 481
10.4.1.1 Adding the AS-i master CM 1243-2 and AS-i slave 482
10.4.1.2 Configuring logical network connections between two AS-i devices 482
10.4.1.3 Configuring the properties of the AS-i master CM1243-2 483
10.4.1.4 Assigning an AS-i address to an AS-i slave 483
10.4.2 Exchanging data between the user program and AS-i slaves 484
10.4.2.1 STEP 7 basic configuration 484

10.4.2.2 Configuring slaves with STEP 7 ... 486
10.4.3 Distributed I/O Instructions ... 488
10.4.4 Working with AS-i online tools ... 488

10.5 S7 communication ... 489
10.5.1 GET and PUT instructions ... 489
10.5.2 Creating an S7 connection ... 493
10.5.3 Configuring the Local/Partner connection path between two devices ... 493
10.5.4 GET/PUT connection parameter assignment ... 494
10.5.4.1 Connection parameters ... 494
10.5.4.2 Configuring a CPU-to-CPU S7 connection ... 497

11 Web server ... 503

11.1 Enabling the Web server ... 504

11.2 Standard web pages ... 505
11.2.1 Accessing the standard Web pages from the PC ... 505
11.2.2 Layout of the standard Web pages ... 506
11.2.3 Introduction ... 508
11.2.4 Start ... 509
11.2.5 Identification ... 510
11.2.6 Diagnostic Buffer ... 510
11.2.7 Module Information ... 511
11.2.8 Communication ... 513
11.2.9 Variable Status ... 515
11.2.10 Data Logs ... 516
11.2.11 Update Firmware ... 519

11.3 User-defined web pages ... 521
11.3.1 Creating HTML pages ... 521
11.3.2 AWP commands supported by the S7-1200 Web server ... 522
11.3.2.1 Reading variables ... 523
11.3.2.2 Writing variables ... 524
11.3.2.3 Reading special variables ... 526
11.3.2.4 Writing special variables ... 527
11.3.2.5 Using an alias for a variable reference ... 529
11.3.2.6 Defining enum types ... 529
11.3.2.7 Referencing CPU variables with an enum type ... 530
11.3.2.8 Creating fragments ... 532
11.3.2.9 Importing fragments ... 533
11.3.2.10 Combining definitions ... 533
11.3.2.11 Handling tag names that contain special characters ... 534
11.3.3 Configuring use of user-defined Web pages ... 535
11.3.4 Programming the WWW instruction for user-defined web pages ... 537
11.3.5 Downloading the program blocks to the CPU ... 538
11.3.6 Accessing the user-defined web pages from the PC ... 539
11.3.7 Constraints specific to user-defined Web pages ... 539
11.3.8 Example of a user-defined web page ... 540
11.3.8.1 Web page for monitoring and controlling a wind turbine ... 540
11.3.8.2 Reading and displaying controller data ... 542
11.3.8.3 Using an enum type ... 543
11.3.8.4 Writing user input to the controller ... 544
11.3.8.5 Writing a special variable ... 545
11.3.8.6 Reference: HTML listing of remote wind turbine monitor Web page ... 545

11.3.8.7 Configuration in STEP 7 of the example Web page 549
11.3.9 Setting up user-defined Web pages in multiple languages 551
11.3.9.1 Creating the folder structure 551
11.3.9.2 Programming the language switch.....552
11.3.9.3 Configuring STEP 7 to use a multi-language page structure 554
11.3.10 Advanced user-defined Web page control.....555

11.4 Constraints 558
11.4.1 Features restricted when JavaScript is disabled 558
11.4.2 Features restricted when cookies are not allowed.....560
11.4.3 Importing the Siemens security certificate 560
11.4.4 Importing CSV format data logs to non-USA/UK versions of Microsoft Excel 561

12 Communication processor 563

12.1 Using the serial communication interfaces 563

12.2 Biasing and terminating an RS485 network connector.....564

12.3 Point-to-Point (PtP) communication.....565
12.3.1 Point-to-Point instructions 566
12.3.1.1 Common parameters for Point-to-Point instructions.....566
12.3.1.2 PORT_CFG instruction 568
12.3.1.3 SEND_CFG instruction 569
12.3.1.4 RCV_CFG instruction.....571
12.3.1.5 SEND_PTP instruction.....575
12.3.1.6 RCV_PTP instruction 578
12.3.1.7 RCV_RST instruction 580
12.3.1.8 SGN_GET instruction.....581
12.3.1.9 SGN_SET instruction 582
12.3.2 Configuring the communication ports 583
12.3.2.1 Managing flow control 585
12.3.3 Configuring the transmit (send) and receive parameters 586
12.3.3.1 Configuring transmit (send) parameters 586
12.3.3.2 Configuring receive parameters.....587
12.3.4 Programming the PtP communications 594
12.3.4.1 Polling architecture 595
12.3.5 Example: Point-to-Point communication.....596
12.3.5.1 Configuring the communication module 597
12.3.5.2 Configuring the RS422 and RS485 599
12.3.5.3 Programming the STEP 7 program 602
12.3.5.4 Configuring the terminal emulator.....603
12.3.5.5 Running the example program.....604

12.4 Universal serial interface (USS) communication 604
12.4.1 Requirements for using the USS protocol 605
12.4.2 USS_DRV instruction.....608
12.4.3 USS_PORT instruction 610
12.4.4 USS_RPM instruction 611
12.4.5 USS_WPM instruction.....612
12.4.6 USS status codes 614
12.4.7 General drive setup information.....616

12.5 Modbus communication 619
12.5.1 Overview of Modbus RTU and TCP communication 619
12.5.2 Modbus TCP 622

12.5.2.1 MB_CLIENT (Modbus TCP) ... 622
12.5.2.2 MB_SERVER (Modbus TCP) ... 628
12.5.2.3 MB_SERVER example: Multiple TCP connections ... 633
12.5.2.4 MB_CLIENT example 1: Multiple requests with common TCP connection ... 634
12.5.2.5 MB_CLIENT example 2: Multiple requests with different TCP connections ... 635
12.5.2.6 MB_CLIENT example 3: Output image write request ... 636
12.5.2.7 MB_CLIENT example 4: Coordinating multiple requests ... 636
12.5.3 Modbus RTU ... 637
12.5.3.1 MB_COMM_LOAD ... 638
12.5.3.2 MB_MASTER ... 641
12.5.3.3 MB_SLAVE ... 647
12.5.3.4 Modbus RTU master example program ... 653
12.5.3.5 Modbus RTU slave example program ... 654

12.6 Telecontrol and TeleService with the CP 1242-7 ... 655
12.6.1 Connection to a GSM network ... 655
12.6.2 Applications of the CP 1242-7 ... 657
12.6.3 Other properties of the CP ... 658
12.6.4 Accessories ... 659
12.6.5 Configuration examples for telecontrol ... 660

13 Teleservice communication (SMTP email) ... 665

13.1 TM_Mail transfer email instruction ... 665

14 Online and diagnostic tools ... 673

14.1 Status LEDs ... 673

14.2 Going online and connecting to a CPU ... 675

14.3 Assigning a name to a PROFINET IO device online ... 676

14.4 Setting the IP address and time of day ... 678

14.5 Resetting to factory settings ... 678

14.6 CPU operator panel for the online CPU ... 679

14.7 Monitoring the cycle time and memory usage ... 680

14.8 Displaying diagnostic events in the CPU ... 680

14.9 Comparing offline and online CPUs ... 681

14.10 Monitoring and modifying values in the CPU ... 682
14.10.1 Going online to monitor the values in the CPU ... 683
14.10.2 Displaying status in the program editor ... 684
14.10.3 Capturing the online values of a DB to reset the start values ... 684
14.10.4 Using a watch table to monitor and modify values in the CPU ... 685
14.10.4.1 Using a trigger when monitoring or modifying PLC tags ... 686
14.10.4.2 Enabling outputs in STOP mode ... 687
14.10.5 Forcing values in the CPU ... 688
14.10.5.1 Using the force table ... 688
14.10.5.2 Operation of the Force function ... 689

14.11 Downloading in RUN mode ... 690
14.11.1 Prerequisites for "Download in RUN mode" ... 691
14.11.2 Changing your program in RUN mode ... 692
14.11.3 Downloading selected blocks ... 693

14.11.4 Downloading a single selected block with a compile error in another block 694
14.11.5 System reaction if the download process fails 695
14.11.6 Downloading the program in RUN mode 696

A Technical specifications 699

A.1 General Technical Specifications 699

A.2 CPU 1211C 705
A.2.1 General specifications and features 705
A.2.2 Timers, counters and code blocks supported by CPU 1211C 707
A.2.3 Digital inputs and outputs 709
A.2.4 Analog inputs 710
A.2.4.1 Step response of the built-in analog inputs of the CPU 711
A.2.4.2 Sample time for the built-in analog ports of the CPU 711
A.2.5 CPU 1211C Wiring diagrams 712

A.3 CPU 1212C 715
A.3.1 General specifications and features 715
A.3.2 Timers, counters and code blocks supported by CPU 1212C 716
A.3.3 Digital inputs and outputs 718
A.3.4 Analog inputs 720
A.3.4.1 Step response of the built-in analog inputs of the CPU 720
A.3.4.2 Sample time for the built-in analog ports of the CPU 721
A.3.5 CPU 1212C Wiring diagrams 721

A.4 CPU 1214C 724
A.4.1 General specifications and features 724
A.4.2 Timers, counters and code blocks supported by CPU 1214C 725
A.4.3 Digital inputs and outputs 727
A.4.4 Analog inputs 729
A.4.4.1 Step response of the built-in analog inputs of the CPU 730
A.4.4.2 Sample time for the built-in analog ports of the CPU 730
A.4.5 CPU 1214C Wiring Diagrams 731

A.5 CPU 1215C 734
A.5.1 General specifications and features 734
A.5.2 Timers, counters and code blocks supported by CPU 1215C 735
A.5.3 Digital inputs and outputs 737
A.5.4 Analog inputs and outputs 739
A.5.4.1 Analog input specifications 739
A.5.4.2 Step response of built-in analog inputs of the CPU 740
A.5.4.3 Sample time for the built-in analog ports of the CPU 740
A.5.4.4 Analog output specifications 740
A.5.5 CPU 1215C Wiring Diagrams 741

A.6 Digital signal modules (SMs) 745
A.6.1 SM 1221 digital input specifications 745
A.6.2 SM 1222 8-Point Digital Output Specifications 747
A.6.3 SM 1222 16-Point Digital Output Specifications 749
A.6.4 SM 1223 Digital Input/Output VDC Specifications 753
A.6.5 SM 1223 Digital Input/Output AC Specifications 757

A.7 Analog signal modules (SMs) 760
A.7.1 SM 1231 analog input module specifications 760
A.7.2 SM 1232 analog output module specifications 764

A.7.3 SM 1234 analog input/output module specifications ... 766
A.7.4 Step response of the analog inputs ... 769
A.7.5 Sample time and update times for the analog inputs ... 769
A.7.6 Measurement ranges of the analog inputs for voltage ... 770
A.7.7 Measurement ranges of the analog inputs for current ... 770
A.7.8 Output (AQ) measurement ranges for voltage and current (SB and SM) ... 771

A.8 Thermocouple and RTD signal modules (SMs) ... 772
A.8.1 SM 1231 Thermocouple ... 772
A.8.1.1 Basic operation for a thermocouple ... 775
A.8.1.2 Selection tables for the SM 1231 thermocouple ... 776
A.8.2 SM 1231 RTD ... 778
A.8.2.1 Selection tables for the SM 1231 RTD ... 781

A.9 Digital signal boards (SBs) ... 784
A.9.1 SB 1221 200 kHz digital input specifications ... 784
A.9.2 SB 1222 200 kHz digital output specifications ... 786
A.9.3 SB 1223 200 kHz digital input / output specifications ... 789
A.9.4 SB 1223 2 X 24 VDC input / 2 X 24 VDC output specifications ... 792

A.10 Analog signal boards (SBs) ... 794
A.10.1 SB 1231 1 analog input specifications ... 794
A.10.2 SB 1232 1 analog output specifications ... 796
A.10.3 Measurement ranges for analog inputs and outputs ... 798
A.10.3.1 Step response of the analog inputs ... 798
A.10.3.2 Sample time and update times for the analog inputs ... 799
A.10.3.3 Measurement ranges of the analog inputs for voltage ... 799
A.10.3.4 Measurement ranges of the analog inputs for current ... 800
A.10.3.5 Output (AQ) measurement ranges for voltage and current (SB and SM) ... 800
A.10.4 Thermocouple SBs ... 801
A.10.4.1 SB 1231 1 analog thermocouple input specifications ... 801
A.10.4.2 Basic operation for a thermocouple ... 803
A.10.5 RTD SBs ... 805
A.10.5.1 SB 1231 1 analog RTD input specifications ... 805
A.10.5.2 Selection tables for the SB 1231 RTD ... 808

A.11 BB 1297 Battery Board ... 810

A.12 Communication interfaces ... 812
A.12.1 PROFIBUS ... 812
A.12.1.1 CM 1242-5 ... 812
A.12.1.2 CM 1243-5 ... 813
A.12.2 GPRS ... 815
A.12.2.1 CP 1242-7 ... 815
A.12.3 CM 1243-2 AS-i Master ... 818
A.12.3.1 Technical data for the AS-i master CM 1243-2 ... 818
A.12.3.2 Electrical connections of the AS-i master CM 1243-2 ... 819
A.12.4 RS232, RS422, and RS485 ... 821
A.12.4.1 CB 1241 RS485 Specifications ... 821
A.12.4.2 CM 1241 RS232 Specifications ... 823
A.12.4.3 CM 1241 RS422/485 Specifications ... 824

A.13 TeleService (TS Adapter and TS Adapter modular) ... 826

A.14 SIMATIC memory cards ... 826

A.15 Input simulators 826
A.16 I/O expansion cable 827
A.17 Companion products 828
A.17.1 PM 1207 power module 828
A.17.2 CSM 1277 compact switch module 828
B Calculating a power budget 829
C Order numbers 833
C.1 CPU modules 833
C.2 Signal modules (SMs), signal boards (SBs) and battery boards (BB) 833
C.3 Communication 834
C.4 Other modules 835
C.5 Memory cards 836
C.6 Basic HMI devices 836
C.7 Spare parts and other hardware 836
C.8 Programming software 837
C.9 Documentation 837
Index 839

Product overview

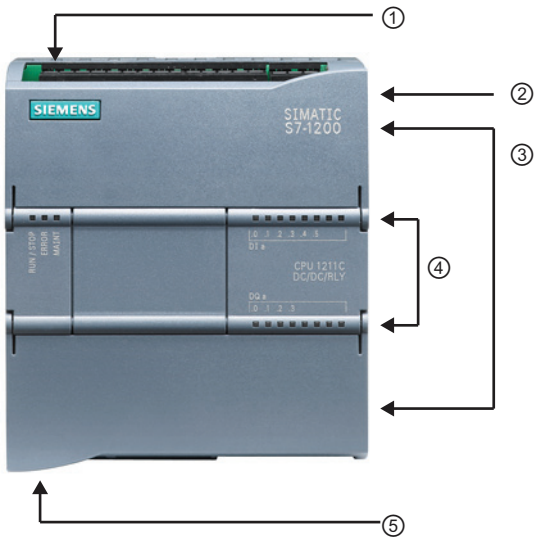
1

1.1 Introducing the S7-1200 PLC

The S7-1200 controller provides the flexibility and power to control a wide variety of devices in support of your automation needs. The compact design, flexible configuration, and powerful instruction set combine to make the S7-1200 a perfect solution for controlling a wide variety of applications.

The CPU combines a microprocessor, an integrated power supply, input and output circuits, built-in PROFINET, high-speed motion control I/O, and on-board analog inputs in a compact housing to create a powerful controller. After you download your program, the CPU contains the logic required to monitor and control the devices in your application. The CPU monitors the inputs and changes the outputs according to the logic of your user program, which can include Boolean logic, counting, timing, complex math operations, and communications with other intelligent devices.

The CPU provides a PROFINET port for communication over a PROFINET network. Additional modules are available for communicating over PROFIBUS, GPRS, RS485 or RS232 networks.



① Power connector

② Memory card slot under top door

③ Removable user wiring connectors (behind the doors)

④ Status LEDs for the on-board I/O

⑤ PROFINET connector (on the bottom of the CPU)

Several security features help protect access to both the CPU and the control program:

- Every CPU provides password protection (Page 164) that allows you to configure access to the CPU functions.
- You can use "know-how protection" (Page 165) to hide the code within a specific block.
- You can use copy protection (Page 166) to bind your program to a specific memory card or CPU.

Table 1- 1 Comparing the CPU models

Feature		CPU 1211C	CPU 1212C	CPU 1214C	CPU 1215C
Physical size (mm)		90 x 100 x 75	90 x 100 x 75	110 x 100 x 75	130 x 100 x 75
User memory	Work	30 Kbytes	50 Kbytes	75 Kbytes	100 Kbytes
	Load	1 Mbyte	1 Mbyte	4 Mbytes	4 Mbytes
	Retentive	10 Kbytes	10 Kbytes	10 Kbytes	10 Kbytes
Local on-board I/O	Digital	6 inputs/4 outputs	8 inputs/6 outputs	14 inputs/10 outputs	14 inputs/10 outputs
	Analog	2 inputs	2 inputs	2 inputs	2 inputs / 2 outputs
Process image size	Inputs (I)	1024 bytes	1024 bytes	1024 bytes	1024 bytes
	Outputs (Q)	1024 bytes	1024 bytes	1024 bytes	1024 bytes
Bit memory (M)		4096 bytes	4096 bytes	8192 bytes	8192 bytes
Signal module (SM) expansion		None	2	8	8
Signal board (SB), Battery board (BB), or communication board (CB)		1	1	1	1
Communication module (CM) (left-side expansion)		3	3	3	3
High-speed counters	Total	3 built-in I/O, 5 with SB	4 built-in I/O, 6 with SB	6	6
	Single phase	3 at 100 kHz SB: 2 at 30 kHz	3 at 100 kHz 1 at 30 kHz SB: 2 at 30 kHz	3 at 100 kHz 3 at 30 kHz	3 at 100 kHz 3 at 30 kHz
	Quadrature phase	3 at 80 kHz SB: 2 at 20 kHz	3 at 80 kHz 1 at 20 kHz SB: 2 at 20 kHz	3 at 80 kHz 3 at 20 kHz	3 at 80 kHz 3 at 20 kHz
Pulse outputs[1]		4	4	4	4
Memory card		SIMATIC Memory card (optional)			
Real time clock retention time		20 days, typ. / 12 day min. at 40 degrees C (maintenance-free Super Capicator)			
PROFINET		1 Ethernet communication port			2 Ethernet communication ports
Real math execution speed		2.3 µs/instruction			
Boolean execution speed		0.08 µs/instruction			

[1] For CPU models with relay outputs, you must install a digital signal board (SB) to use the pulse outputs.

Each CPU provides dedicated HMI connections to support up to 3 HMI devices. The total number of HMI is affected by the types of HMI panels in your configuration. For example, you could have up to three SIMATIC Basic panels connected to your CPU, or you could have up to two SIMATIC Comfort panels with one additional Basic panel.

The different CPU models provide a diversity of features and capabilities that help you create effective solutions for your varied applications. For detailed information about a specific CPU, see the technical specifications (Page 699).

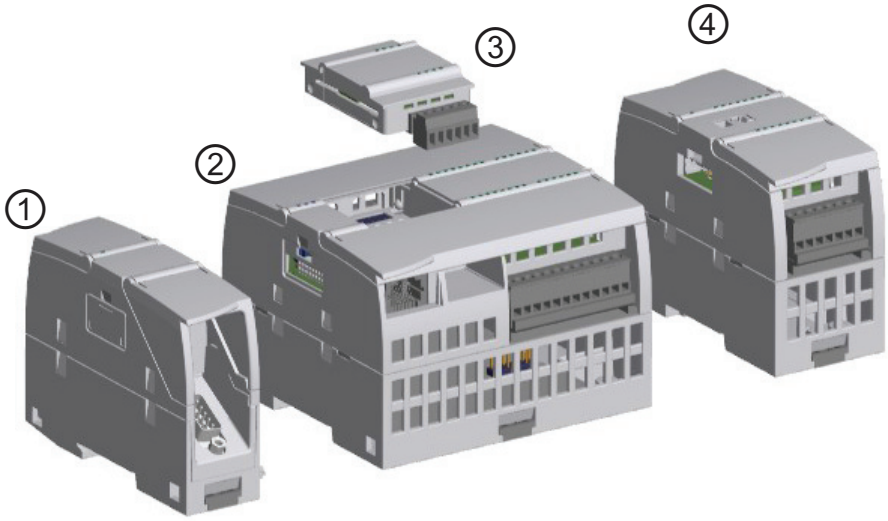
Table 1- 2 Blocks, timers and counters supported by S7-1200

Element		Description
Blocks	Type	OB, FB, FC, DB
	Size	30 Kbytes (CPU 1211C) 50 Kbytes (CPU 1212C) 64 Kbytes (CPU 1214C and CPU 1215C)
	Quantity	Up to 1024 blocks total (OBs + FBs + FCs + DBs)
	Address range for FBs, FCs, and DBs	1 to 65535 (such as FB 1 to FB 65535)
	Nesting depth	16 from the program cycle or start up OB; 4 from the time delay interrupt, time-of-day interrupt, cyclic interrupt, hardware interrupt, time error interrupt, or diagnostic error interrupt OB
	Monitoring	Status of 2 code blocks can be monitored simultaneously
OBs	Program cycle	Multiple: OB 1, OB 200 to OB 65535
	Startup	Multiple: OB 100, OB 200 to OB 65535
	Time-delay interrupts and cyclic interrupts	4[1] (1 per event): OB 200 to OB 65535
	Hardware interrupts (edges and HSC)	50 (1 per event): OB 200 to OB 65535
	Time error interrupts	1: OB 80
	Diagnostic error interrupts	1: OB 82
Timers	Type	IEC
	Quantity	Limited only by memory size
	Storage	Structure in DB, 16 bytes per timer
Counters	Type	IEC
	Quantity	Limited only by memory size
	Storage	Structure in DB, size dependent upon count type • SInt, USInt: 3 bytes • Int, UInt: 6 bytes • DInt, UDInt: 12 bytes

[1] Time-delay and cyclic interrupts use the same resources in the CPU. You can have only a total of 4 of these interrupts (time-delay plus cyclic interrupts). You cannot have 4 time-delay interrupts and 4 cyclic interrupts.

1.2 Expansion capability of the CPU

The S7-1200 family provides a variety of modules and plug-in boards for expanding the capabilities of the CPU with additional I/O or other communication protocols. For detailed information about a specific module, see the technical specifications (Page 699).



① Communication module (CM), communcation processor (CP), or TS Adapter
② CPU
③ Signal board (SB), communication board (CB), or Battery Board (BB)
④ Signal module (SM)

Table 1- 3 Digital signal modules and signal boards

Type	Input only	Output only	Combination In/Out
③ digital SB	• 4 x 24VDC In, 200 kHz • 4 x 5VDC In, 200 kHz	• 4 x 24VDC Out, 200 kHz • 4 x 5VDC Out, 200 kHz	• 2 x 24VDC In / 2 x 24VDC Out • 2 x 24VDC In / 2 x 24VDC Out, 200 kHz • 2 x 5VDC In / 2 x 5VDC Out, 200 kHz
④ digital SM	• 8 x 24VDC In	• 8 x 24VDC Out • 8 x Relay Out • 8 x Relay Out (Changeover)	• 8 x 24VDC In / 8 x 24VDC Out • 8 x 24VDC In / 8 x Relay Out • 8 x 120/230VAC In / 8 x Relay Out
	• 16 x 24VDC In	• 16 x 24VDC Out • 16 x Relay Out	• 16 x 24VDC In / 16 x 24VDC Out • 16 x 24VDC In / 16 x Relay Out

Table 1- 4 Analog signal modules and signal boards

Type	Input only	Output only	Combination In/Out
③ analog SB	• 1 x 12 bit Analog In • 1 x 16 bit RTD • 1 x 16 bit Thermocouple	• 1 x Analog Out	-
④ analog SM	• 4 x Analog In • 4 x Analog In x 16 bit • 8 x Analog In • Thermocouple: – 4 x 16 bit TC – 8 x 16 bit TC • RTD: – 4 x 16 bit RTD – 8 x 16 bit RTD	• 2 x Analog Out • 4 x Analog Out	• 4 x Analog In / 2 x Analog Out

Table 1- 5 Communication interfaces

Module	Type	Description
① Communication module (CM)	RS232	Full-duplex
	RS422/485	Full-duplex (RS422) Half-duplex (RS485)
	PROFIBUS Master	DPV1
	PROFIBUS Slave	DPV1
	AS-i Master (CM 1243-2)	AS-Interface
① Communication processor (CP)	Modem connectivity	GPRS
① Communication board (CB)	RS485	Half-duplex
① TeleService	TS Adapter IE Basic[1]	Connection to CPU
	TS Adapter GSM	GSM/GPRS
	TS Adapter Modem	Modem
	TS Adapter ISDN	ISDN
	TS Adapter RS232	RS232

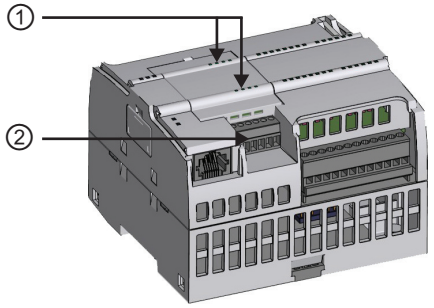
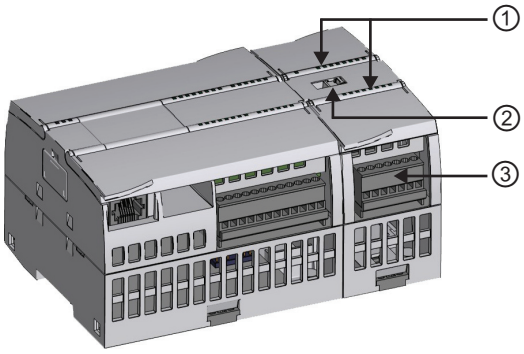
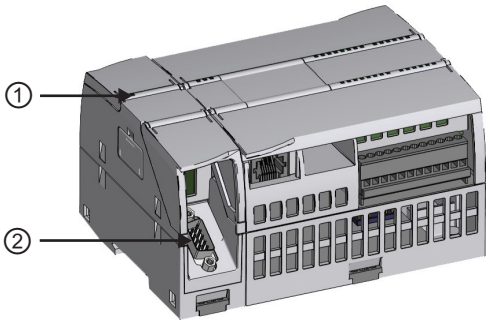
[1] The TS Adapter allows you to connect various communication interfaces to the PROFINET port of the CPU. You install the TS Adapter on the left side of the CPU and connect the TS Adapter modular (up to 3) onto the TS Adapter.

Table 1- 6 Other boards

Module	Description
③ Battery board	Plugs into expansion board interface on front of CPU. Provides long term backup of realtime clock

1.3 S7-1200 modules

Table 1- 7 S7-1200 expansion modules

Type of module	Description		
The CPU supports one plug-in expansion board: • A signal board (SB) provides additional I/O for your CPU. The SB connects on the front of the CPU. • A communication board (CB) allows you to add another communication port to your CPU. • A battery board (BB) allows you to provide long term backup of the realtime clock.		①	Status LEDs on the SB
		②	Removable user wiring connector
Signal modules (SMs) add additional functionality to the CPU. SMs connect to the right side of the CPU. • Digital I/O • Analog I/O • RTD and thermocouple		①	Status LEDs
		②	Bus connector
		③	Removable user wiring connector
Communication modules (CMs) and communications processors (CPs) add communication options to the CPU, such as for PROFIBUS or RS232 / RS485 connectivity (for PtP, Modbus or USS), or the AS-i master. A CP provides capabilities for other types of communication, such as to connect the CPU over a GPRS network. • The CPU supports up to 3 CMs or CPs • Each CM or CP connects to the left side of the CPU (or to the left side of another CM or CP)		①	Status LEDs
		②	Communication connector

1.4 New features

The following features are new in this release:

- A standard Web server page for performing a CPU firmware update (Page 519)
- The ability to use three PROFIBUS DP CM 1243-5 master modules or three AS-i CM 1243-2 master modules

Note

To use three AS-i modules as masters, you must update the firmware of the AS-i modules.

New modules for the S7-1200

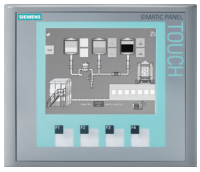
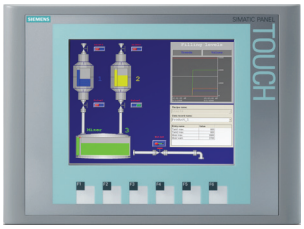
A variety of new modules expand the power of the S7-1200 CPU and provide the flexibility to meet your automation needs:

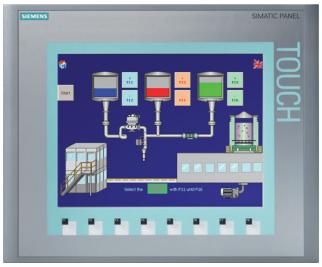
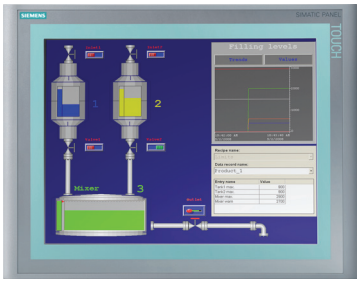
- New and improved CPUs:
  - New CPU 1215C DC/DC/DC, CPU 1215C DC/DC/Relay, and CPU 1215C AC/DC/Relay offer 100 Kbytes of work memory, dual Ethernet, and analog outputs.
  - New and improved CPU 1211Cs, CPU 1212Cs, and CPU 1214Cs have faster processing time, the possibility of 4 PTOs (the CPU 1211C requires a signal board), increased retentive memory (10 Kbytes), and increased time-of-day hold up time (20 days).
- New I/O signal module: SM 1231 AI 4 x 16 bit provides higher sample rate and increased number of bits.
- New battery board (BB 1297) offers long term backup of the realtime clock. The BB 1297 is pluggable in the signal board slot of the S7-1200 CPU (firmware 3.0 and later versions).

To use the new modules you must use STEP 7 V11 SP2 Update 3 or later (Basic or Professional) and you must download the hardware support package (HSP) for new modules from the STEP 7 **Options > Support Packages** menu command. Follow the instructions for adding modules to the hardware catalog for STEP 7 (TIA Portal) from the Siemens Service and Support Site (http://support.automation.siemens.com).

1.5 Basic HMI panels

Because visualization is becoming a standard component for most machine designs, the SIMATIC HMI Basic Panels provide touch-screen devices for basic operator control and monitoring tasks. All panels have a protection rating for IP65 and have CE, UL, cULus, and NEMA 4x certification.

Basic HMI Panel	Description	Technical data
KP 300 Basic PN	3.6" membrane keyboard with 10 freely configurable tactile keys • Mono (STN, black/white) • 87 mm x 31 mm (3.6") • Backlight color programmed (white, green, yellow or red) • Resolution: 240 x 80	• 250 tags • 50 process screens • 200 alarms • 25 curves • 40 KB recipe memory • 5 recipes, 20 data records, 20 entries
 KTP 400 Basic PN	4" touch screen with 4 tactile keys • Mono (STN, gray scale) • 76.79 mm x 57.59 mm (3.8") Portrait or landscape • Resolution: 320 x 240	• 250 tags • 50 process screens • 200 alarms • 25 curves • 40 KB recipe memory • 5 recipes, 20 data records, 20 entries
 KTP 600 Basic PN	6" touch screen with 6 tactile keys • Color (TFT, 256 colors) or Mono (STN, gray scales) • 115.2 mm x 86.4 mm (5.7") Portrait or landscape • Resolution: 320 x 240	• 500 tags • 50 process screens • 200 alarms • 25 curves • 40 KB recipe memory • 5 recipes, 20 data records, 20 entries

Basic HMI Panel	Description	Technical data
 KTP 1000 Basic PN	10" touch screen with 8 tactile keys • Color (TFT, 256 colors) • 211.2 mm x 158.4 mm (10.4") • Resolution: 640 x 480	• 500 tags • 50 process screens • 200 alarms • 25 curves • 40 KB recipe memory • 5 recipes, 20 data records, 20 entries
 TP 1500 Basic PN	15" touch screen • Color (TFT, 256 colors) • 304.1 mm x 228.1 mm (15.1") • Resolution: 1024 x 768	• 500 tags • 50 process screens • 200 alarms • 25 curves • 40 KB recipe memory (integrated flash) • 5 recipes, 20 data records, 20 entries

STEP 7 programming software 2

STEP 7 provides a user-friendly environment to develop, edit, and monitor the logic needed to control your application, including the tools for managing and configuring all of the devices in your project, such as controllers and HMI devices. To help you find the information you need, STEP 7 provides an extensive online help system.

STEP 7 provides standard programming languages for convenience and efficiency in developing the control program for your application.

- LAD (ladder logic) is a graphical programming language. The representation is based on circuit diagrams (Page 155).
- FBD (Function Block Diagram) is a programming language that is based on the graphical logic symbols used in Boolean algebra (Page 156).
- SCL (structured control language) is a text-based, high-level programming language.

When you create a code block, you select the programming language to be used by that block. Your user program can utilize code blocks created in any or all of the programming languages.

Note

STEP 7 is the programming and configuration software component of the TIA Portal. The TIA Portal, in addition to STEP 7, also includes WinCC for designing and executing runtime process visualization, and includes online help for WinCC as well as STEP 7.

2.1 System requirements

To install the STEP 7 software on a PC running Windows XP or Windows 7 operating system, you must log in with Administrator privileges.

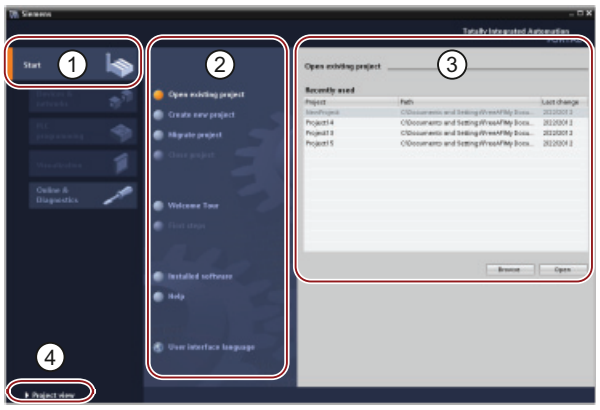
Table 2- 1 System requirements

Hardware/software	Requirements
Processor type	Pentium M, 1.6 GHz or similar
RAM	1 GB
Available hard disk space	2 GB on system drive C:\
Operating systems	• Windows XP Professional SP3 • Windows 2003 Server R2 StdE SP2 • Windows 7 Home Premium (STEP 7 Basic only, not supported for STEP 7 Professional) • Windows 7 (Professional, Enterprise, Ultimate) • Windows 2008 Server StdE R2

Hardware/software	Requirements
Graphics card	32 MB RAM 24-bit color depth
Screen resolution	1024 x 768
Network	20 Mbit/s Ethernet or faster
Optical drive	DVD-ROM

2.2 Different views to make the work easier

STEP 7 provides a user-friendly environment to develop controller logic, configure HMI visualization, and setup network communication. To help increase your productivity, STEP 7 provides two different views of the project: a task-oriented set of portals that are organized on the functionality of the tools (Portal view), or a project-oriented view of the elements within the project (Project view). Choose which view helps you work most efficiently. With a single click, you can toggle between the Portal view and the Project view.



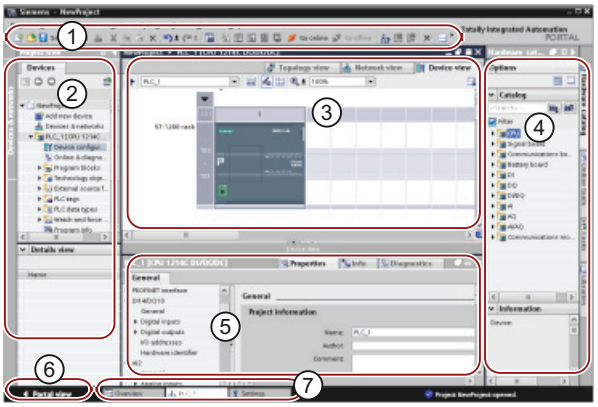
Portal view

① Portals for the different tasks

② Tasks for the selected portal

③ Selection panel for the selected action

④ Changes to the Project view



Project view

① Menus and toolbar

② Project navigator

③ Work area

④ Task cards

⑤ Inspector window

⑥ Changes to the Portal view

⑦ Editor bar

With all of these components in one place, you have easy access to every aspect of your project. For example, the inspector window shows the properties and information for the object that you have selected in the work area. As you select different objects, the inspector window displays the properties that you can configure. The inspector window includes tabs that allow you to see diagnostic information and other messages.

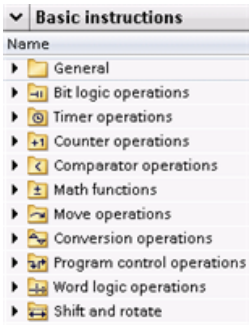
By showing all of the editors that are open, the editor bar helps you work more quickly and efficiently. To toggle between the open editors, simply click the different editor. You can also arrange two editors to appear together, arranged either vertically or horizontally. This feature allows you to drag and drop between editors.

2.3 Easy-to-use tools

2.3.1 Inserting instructions into your user program

STEP 7 provides task cards that contain the instructions for your program. The instructions are grouped according to function.

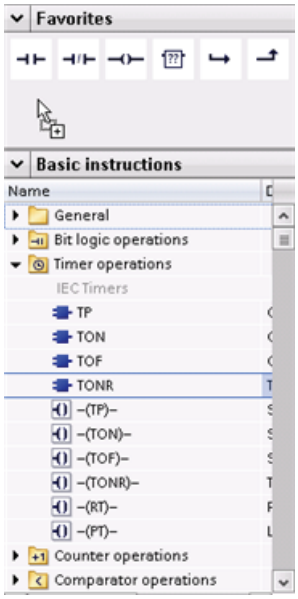
To create your program, you drag instructions from the task card onto a network.



2.3.2 Accessing instructions from the "Favorites" toolbar

STEP 7 provides a "Favorites" toolbar to give you quick access to the instructions that you frequently use. Simply click the icon for the instruction to insert it into your network!

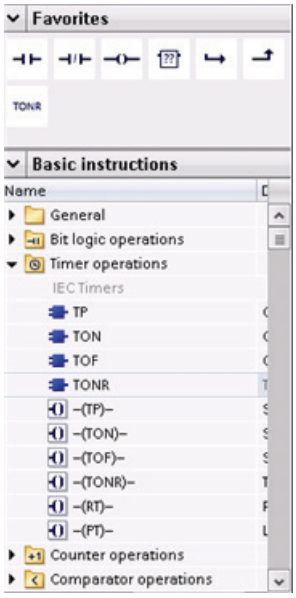
(For the "Favorites" in the instruction tree, double-click the icon.)



You can easily customize the "Favorites" by adding new instructions.

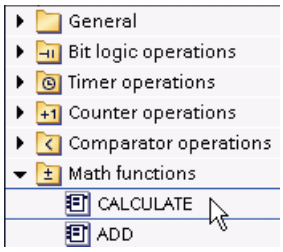
Simply drag and drop an instruction to the "Favorites".

The instruction is now just a click away!

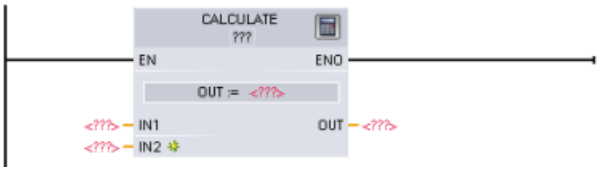


2.3.3 Creating a complex equation with a simple instruction

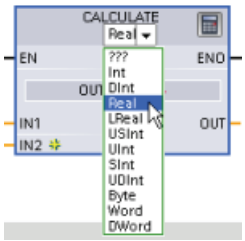
The Calculate instruction lets you create a math function that operates on multiple input parameters to produce the result, according to the equation that you define.



In the Basic instruction tree, expand the Math functions folder. Double-click the Calculate instruction to insert the instruction into your user program.



The unconfigured Calculate instruction provides two input parameters and an output parameter.



Click the "???" and select the data types for the input and output parameters. (The input and output parameters must all be the same data type.)

For this example, select the "Real" data type.

Click the "Edit equation" icon to enter the equation.

For this example, enter the following equation for scaling a raw analog value. (The "In" and "Out" designations correspond to the parameters of the Calculate instruction.)

$Out_{value} = ((Out_{high} - Out_{low}) / (In_{high} - In_{low})) * (In_{value} - In_{low}) + Out_{low}$

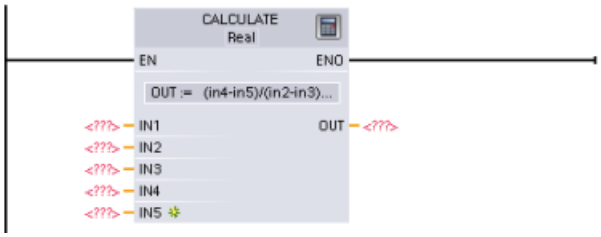
Out = ((in4 - in5) / (in2 - in3)) * (in1 - in3) + in5

Where:			
	Out_{value}	(Out)	Scaled output value
	In_{value}	(in1)	Analog input value
	In_{high}	(in2)	Upper limit for the scaled input value
	In_{low}	(in3)	Lower limit for the scaled input value
	Out_{high}	(in4)	Upper limit for the scaled output value
	Out_{low}	(in5)	Lower limit for the scaled ouput value

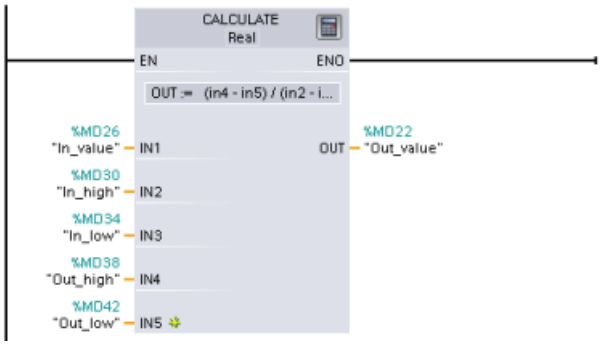
In the "Edit Calculate" box, enter the equation with the parameter names:

OUT = ((in4 - in5) / (in2 - in3)) * (in1 - in3) + in5

When you click "OK", the Calculate instruction creates the inputs required for the instruction.



Enter the tag names for the values that correspond to the parameters.



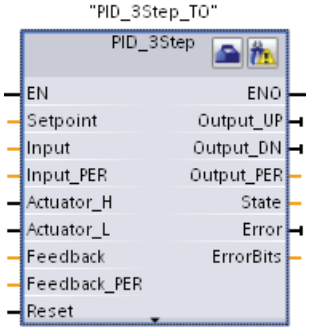
2.3.4 Adding inputs or outputs to a LAD or FBD instruction

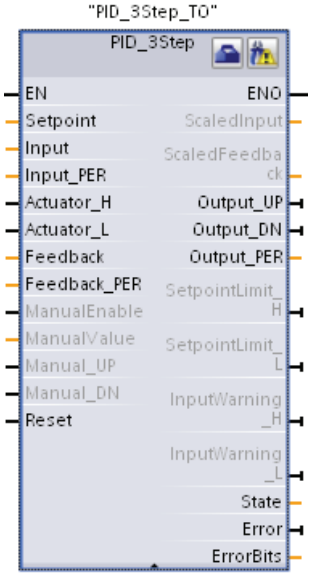
Some of the instructions allow you to create additional inputs or outputs.

- To add an input or output, click the "Create" icon or right-click on an input stub for one of the existing IN or OUT parameters and select the "Insert input" command.
- To remove an input or output, right-click on the stub for one of the existing IN or OUT parameters (when there are more than the original two inputs) and select the "Delete" command.

2.3.5 Expandable instructions

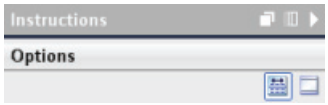
Some of the more complex instructions are expandable, displaying only the key inputs and outputs. To display the inputs and outputs, click the arrow at the bottom of the instruction.



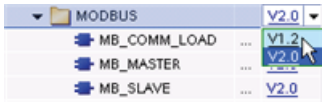


2.3.6 Selecting a version for an instruction

The development and release cycles for certain sets of instructions (such as Modbus, PID and motion) have created multiple released versions for these instructions. To help ensure compatibility and migration with older projects, STEP 7 allows you to choose which version of instruction to insert into your user program.

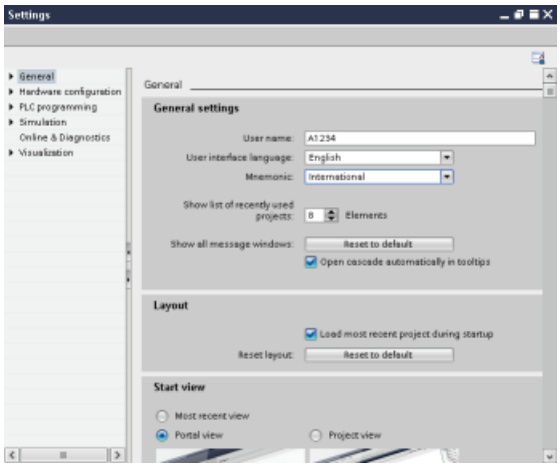


Click the icon on the instruction tree task card to enable the headers and columns of the instruction tree.



To change the version of the instruction, select the appropriate version from the drop-down list.

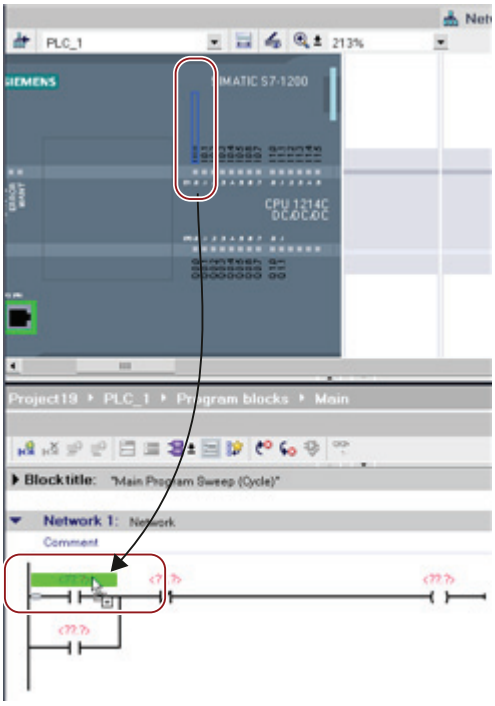
2.3.7 Modifying the appearance and configuration of STEP 7



You can select a variety of settings, such as the appearance of the interface, language, or the folder for saving your work.

Select the "Settings" command from the "Options" menu to change these settings.

2.3.8 Dragging and dropping between editors

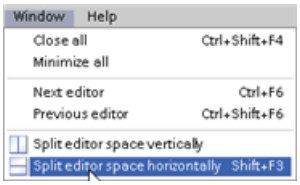


To help you perform tasks quickly and easily, STEP 7 allows you to drag and drop elements from one editor to another. For example, you can drag an input from the CPU to the address of an instruction in your user program.

You must zoom in at least 200% to select the inputs or outputs of the CPU.

Notice that the tag names are displayed not only in the PLC tag table, but also are displayed on the CPU.

To display two editors at one time, use the "Split editor" menu commands or buttons in the toolbar.



To toggle between the editors that have been opened, click the icons in the editor bar.

2.3.9 Changing the operating mode of the CPU

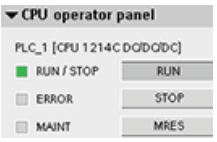
Refer to

The CPU does not have a physical switch for changing the operating mode (STOP or RUN).

Use the "Start CPU" and "Stop CPU" toolbar buttons to change the operating mode of the CPU.

When you configure the CPU in the device configuration, you configure the start-up behavior in the properties of the CPU.

The "Online and diagnostics" portal also provides an operator panel for changing the operating mode of the online CPU. To use the CPU operator panel, you must be connected online to the CPU. The "Online tools" task card displays an operator panel that shows the operating mode of the online CPU. The operator panel also allows you to change the operating mode of the online CPU.

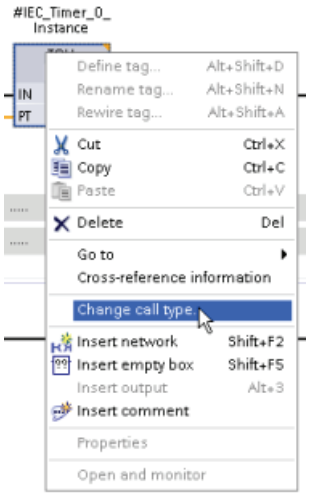


Use the button on the operator panel to change the operating mode (STOP or RUN). The operator panel also provides an MRES button for resetting the memory.

The color of the RUN/STOP indicator shows the current operating mode of the CPU. Yellow indicates STOP mode, and green indicates RUN mode.

Refer to Operating Modes of the CPU in the S7-1200 System Manual (Page 69) for configuring the default operating mode on power up.

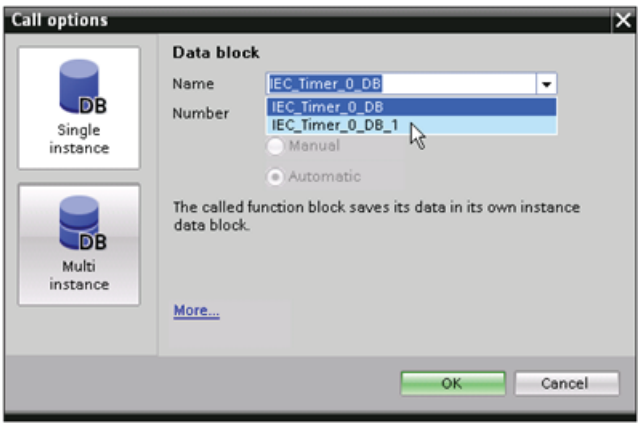
2.3.10 Changing the call type for a DB



STEP 7 allows you to easily create or change the association of a DB for an instruction or an FB that is in an FB.

- You can switch the association between different DBs.
- You can switch the association between a single-instance DB and a multi-instance DB.
- You can create an instance DB (if an instance DB is missing or not available).

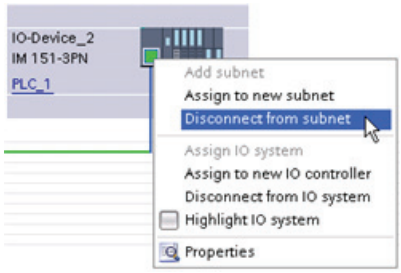
You can access the "Change call type" command either by right-clicking the instruction or FB in the program editor or by selecting the "Block call" command from the "Options" menu.



The "Call options" dialog allows you to select a single-instance or multi-instance DB. You can also select specific DBs from a drop-down list of available DBs.

2.3.11 Temporarily disconnecting devices from a network

You can disconnect individual network devices from the subnet. Because the configuration of the device is not removed from the project, you can easily restore the connection to the device.



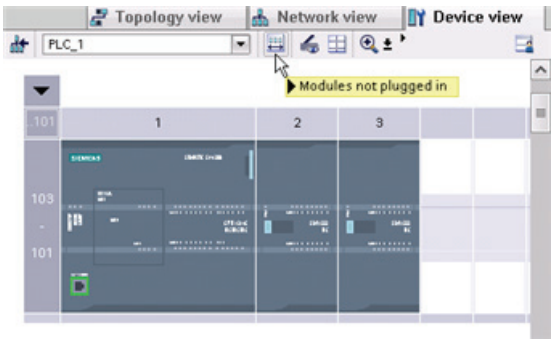
Right-click the interface port of the network device and select the "Disconnect from subnet" command from the context menu.

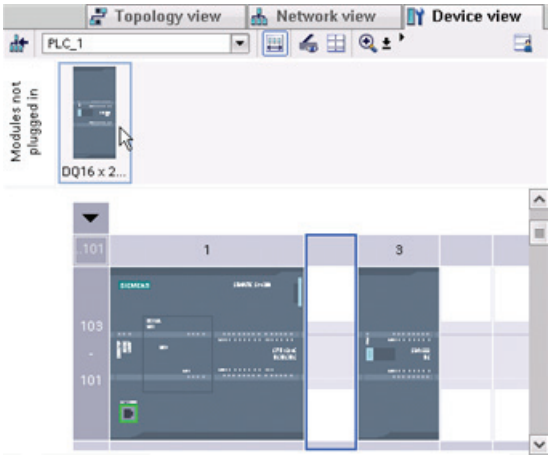
STEP 7 reconfigures the network connections, but does not remove the disconnected device from the project. While the network connection is deleted, the interface addresses are not changed.

When you download the new network connections, the CPU must be set to STOP mode.

To reconnect the device, simply create a new network connection to the port of the device.

2.3.12 Virtual unplugging of devices from the configuration





STEP 7 provides a storage area for "unplugged" modules. You can drag a module from the rack to save the configuration of that module. These unplugged modules are saved with your project, allowing you to reinsert the module in the future without having to reconfigure the parameters.

One use of this feature is for temporary maintenance. Consider a scenario where you might be waiting for a replacement module and plan to temporarily use a different module as a short-term replacement. You could drag the configured module from the rack to the "Unplugged modules" and then insert the temporary module.

Installation 3

3.1 Guidelines for installing S7-1200 devices

The S7-1200 equipment is designed to be easy to install. You can install an S7-1200 either on a panel or on a standard rail, and you can orient the S7-1200 either horizontally or vertically. The small size of the S7-1200 allows you to make efficient use of space.

> **WARNING**
>
> The SIMATIC S7-1200 PLCs are Open Type Controllers. It is required that you install the S7-1200 in a housing, cabinet, or electric control room. Entry to the housing, cabinet, or electric control room should be limited to authorized personnel.
>
> Failure to follow these installation requirements could result in death, severe personal injury and/or property damage.
>
> Always follow these requirements when installing S7-1200 PLCs.

Separate the S7-1200 devices from heat, high voltage, and electrical noise

As a general rule for laying out the devices of your system, always separate the devices that generate high voltage and high electrical noise from the low-voltage, logic-type devices such as the S7-1200.

When configuring the layout of the S7-1200 inside your panel, consider the heat-generating devices and locate the electronic-type devices in the cooler areas of your cabinet. Reducing the exposure to a high-temperature environment will extend the operating life of any electronic device.

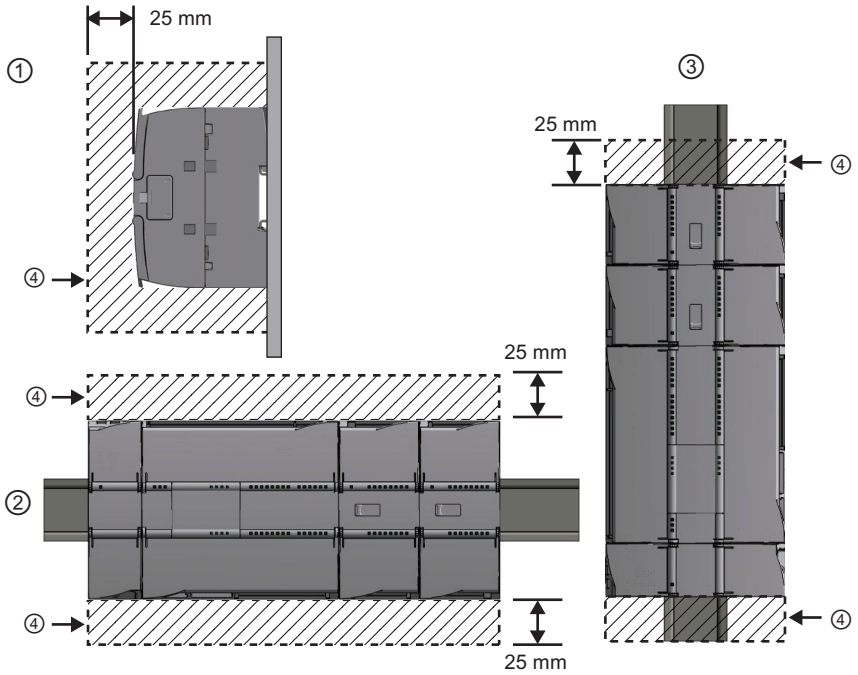
Consider also the routing of the wiring for the devices in the panel. Avoid placing low-voltage signal wires and communications cables in the same tray with AC power wiring and high-energy, rapidly-switched DC wiring.

Provide adequate clearance for cooling and wiring

S7-1200 devices are designed for natural convection cooling. For proper cooling, you must provide a clearance of at least 25 mm above and below the devices. Also, allow at least 25 mm of depth between the front of the modules and the inside of the enclosure.

> **CAUTION**
>
> For vertical mounting, the maximum allowable ambient temperature is reduced by 10 degrees C. Orient a vertically mounted S7-1200 system as shown in the following figure.

When planning your layout for the S7-1200 system, allow enough clearance for the wiring and communications cable connections.



①	Side view	③	Vertical installation
②	Horizontal installation	④	Clearance area

3.2 Power budget

Your CPU has an internal power supply that provides power for the CPU, the signal modules, signal board and communication modules and for other 24 VDC user power requirements.

Refer to the technical specifications (Page 699) for information about the 5 VDC logic budget supplied by your CPU and the 5 VDC power requirements of the signal modules, signal boards, and communication modules. Refer to the "Calculating a power budget" (Page 829) to determine how much power (or current) the CPU can provide for your configuration.

The CPU provides a 24 VDC sensor supply that can supply 24 VDC for input points, for relay coil power on the signal modules, or for other requirements. If your 24 VDC power requirements exceed the budget of the sensor supply, then you must add an external 24 VDC power supply to your system. Refer to the technical specifications (Page 699) for the 24 VDC sensor supply power budget for your particular CPU.

Note

The CM 1243-5 (PROFIBUS master module) requires power from the 24 VDC sensor supply of the CPU.

If you require an external 24 VDC power supply, ensure that the power supply is not connected in parallel with the sensor supply of the CPU. For improved electrical noise protection, it is recommended that the commons (M) of the different power supplies be connected.

WARNING

Connecting an external 24 VDC power supply in parallel with the 24 VDC sensor supply can result in a conflict between the two supplies as each seeks to establish its own preferred output voltage level.

The result of this conflict can be shortened lifetime or immediate failure of one or both power supplies, with consequent unpredictable operation of the PLC system. Unpredictable operation could result in death, severe personal injury and/or property damage.

The DC sensor supply and any external power supply should provide power to different points.

Some of the 24 VDC power input ports in the S7-1200 system are interconnected, with a common logic circuit connecting multiple M terminals. For example, the following circuits are interconnected when designated as "not isolated" in the data sheets: the 24 VDC power supply of the CPU, the power input for the relay coil of an SM, or the power supply for a non-isolated analog input. All non-isolated M terminals must connect to the same external reference potential.

WARNING

Connecting non-isolated M terminals to different reference potentials will cause unintended current flows that may cause damage or unpredictable operation in the PLC and any connected equipment.

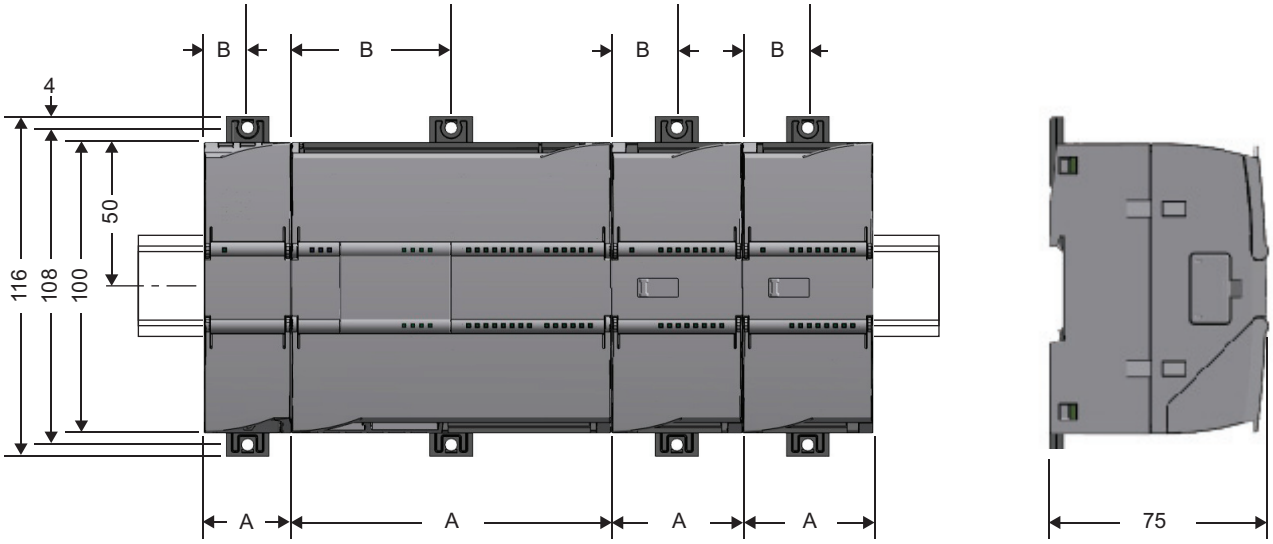
Failure to comply with these guidelines could cause damage or unpredictable operation which could result in death or severe personal injury and/or property damage.

Always ensure that all non-isolated M terminals in an S7-1200 system are connected to the same reference potential.

3.3 Installation and removal procedures

3.3.1 Mounting dimensions for the S7-1200 devices

CPU 1211C, CPU 1212C, CPU 1214C



CPU 1215C

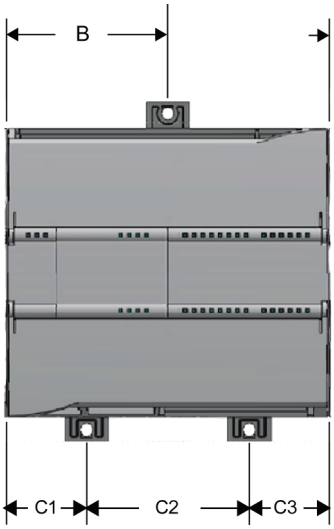


Table 3- 1 Mounting dimensions (mm)

S7-1200 Devices		Width A (mm)	Width B (mm)	Width C (mm)
CPU	CPU 1211C and CPU 1212C	90	45	--
	CPU 1214C	110	55	--

S7-1200 Devices		Width A (mm)	Width B (mm)	Width C (mm)
	CPU 1215C	130	65 (top)	Bottom: C1: 32.5 C2: 65 C3: 32.5
Signal modules	Digital 8 and 16 point Analog 2, 4, and 8 point Thermocouple 4 and 8 point RTD 4 point	45	22.5	--
	Digital DQ 8 x Relay (Changeover)	70	22.5	--
	Analog 16 point RTD 8 point	70	35	--
Communication interfaces	CM 1241 RS232, and CM 1241 RS422/485 CM 1243-5 PROFIBUS master and CM 1242-5 PROFIBUS slave CM 1242-2 AS-i Master CP 1242-7 GPRS	30	15	--
	TS AdapterIE Basic	60 [1]	15	--

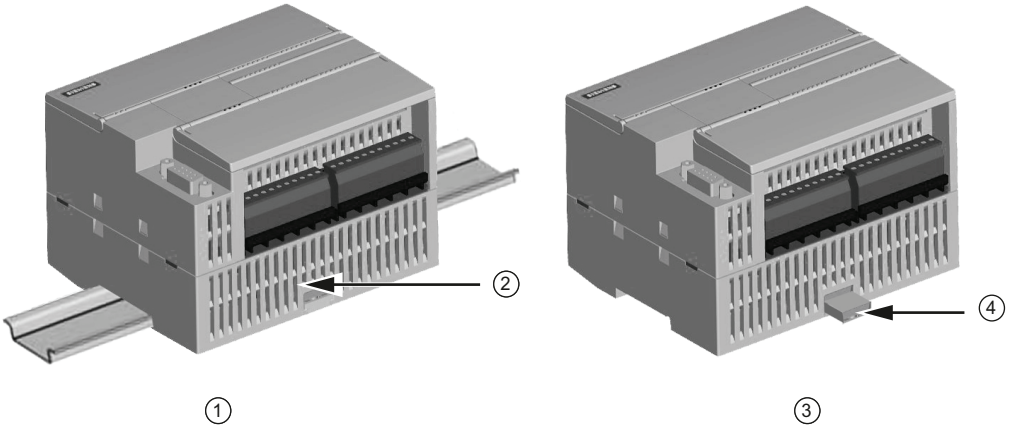
[1] Because you must install a TS Adapter modular with the TS Adapter, the total width ("width A") is 60 mm.

Each CPU, SM, CM, and CP supports mounting on either a DIN rail or on a panel. Use the DIN rail clips on the module to secure the device on the rail. These clips also snap into an extended position to provide screw mounting positions to mount the unit directly on a panel. The interior dimension of the hole for the DIN clips on the device is 4.3 mm.

A 25 mm thermal zone must be provided above and below the unit for free air circulation.

Installing and removing the S7-1200 devices

The CPU can be easily installed on a standard DIN rail or on a panel. DIN rail clips are provided to secure the device on the DIN rail. The clips also snap into an extended position to provide a screw mounting position for panel-mounting the unit.



① DIN rail installation
② DIN rail clip in latched position
③ Panel installation
④ Clip in extended position for panel mounting

Before you install or remove any electrical device, ensure that the power to that equipment has been turned off. Also, ensure that the power to any related equipment has been turned off.

WARNING

Installation or removal of S7-1200 or related equipment with the power applied could cause electric shock or unexpected operation of equipment.

Failure to disable all power to the S7-1200 and related equipment during installation or removal procedures could result in death, severe personal injury and/or property damage due to electric shock or unexpected equipment operation.

Always follow appropriate safety precautions and ensure that power to the S7-1200 is disabled before attempting to install or remove S7-1200 CPUs or related equipment.

Always ensure that whenever you replace or install an S7-1200 device you use the correct module or equivalent device.

WARNING

Incorrect installation of an S7-1200 module may cause the program in the S7-1200 to function unpredictably.

Failure to replace an S7-1200 device with the same model, orientation, or order could result in death, severe personal injury and/or property damage due to unexpected equipment operation.

Replace an S7-1200 device with the same model, and be sure to orient and position it correctly.

WARNING

Do not disconnect equipment when a flammable or combustible atmosphere is present.

Disconnection of equipment when a flammable or combustible atmosphere is present may cause a fire or explosion which could result in death, serious injury and/or property damage.

CAUTION

Electrostatic discharge can damage the device or the receptacle on the CPU.

Make contact with a grounded conductive pad and/or wear a grounded wrist strap whenever you handle the device.

3.3.2 Installing and removing the CPU

You can install the CPU on a panel or on a DIN rail.

Note

Attach any communication modules to the CPU and install the assembly as a unit. Install signal modules separately after the CPU has been installed.

Consider the following when installing the units on the DIN rail or on a panel:

- For DIN rail mounting, make sure the upper DIN rail clip is in the latched (inner) position and that the lower DIN rail clip is in the extended position for the CPU and attached CMs.
- After installing the devices on the DIN rail, move the lower DIN rail clips to the latched position to lock the devices on the DIN rail.
- For panel mounting, make sure the DIN rail clips are pushed to the extended position.

To install the CPU on a panel, follow these steps:

1. Locate, drill, and tap the mounting holes (M4), using the dimensions shown in table, Mounting dimensions (mm) (Page 46).
2. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power.
3. Extend the mounting clips from the module. Make sure the DIN rail clips on the top and bottom of the CPU are in the extended position.
4. Secure the module to the panel, using a Pan Head M4 screw with spring and flat washer. Do not use a flat head screw.

   Note

   The type of screw will be determined by the material upon which it is mounted. You should apply appropriate torque until the spring washer becomes flat. Avoid applying excessive torque to the mounting screws. Do not use a flat head screw.

   Note

   If your system is subject to a high vibration environment, or is mounted vertically, panel mounting the S7-1200 will provide a greater level of protection.

Table 3- 2 Installing the CPU on a DIN rail

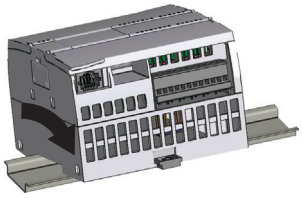
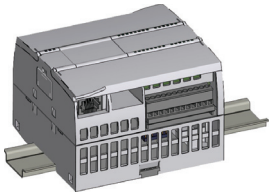
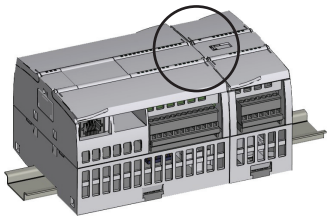
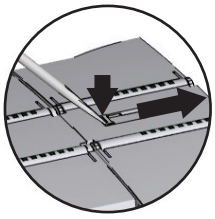
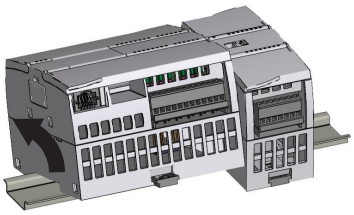
Task	Procedure
	1. Install the DIN rail. Secure the rail to the mounting panel every 75 mm. 2. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 3. Hook the CPU over the top of the DIN rail. 4. Pull out the DIN rail clip on the bottom of the CPU to allow the CPU to fit over the rail. 5. Rotate the CPU down into position on the rail. 6. Push in the clips to latch the CPU to the rail.
	

Table 3- 3 Removing the CPU from a DIN rail

Task	Procedure	
		1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Disconnect the I/O connectors, wiring, and cables from the CPU (Page 55). 3. Remove the CPU and any attached communication modules as a unit. All signal modules should remain installed. 4. If an SM is connected to the CPU, retract the bus connector: – Place a screwdriver beside the tab on the top of the signal module. – Press down to disengage the connector from the CPU. – Slide the tab fully to the right. 5. Remove the CPU: – Pull out the DIN rail clip to release the CPU from the rail. – Rotate the CPU up and off the rail, and remove the CPU from the system.
		

3.3.3 Installing and removing an SB, CB or BB

Table 3- 4 Installing an SB, CB, or BB 1297

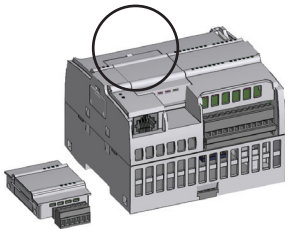
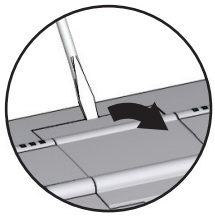
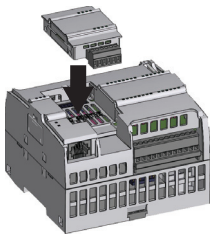
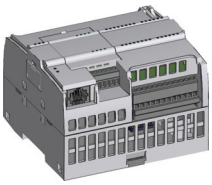
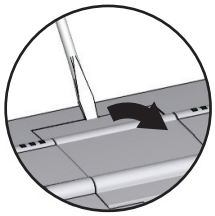
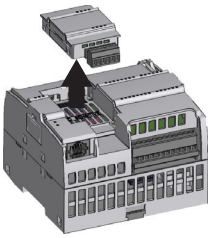
Task	Procedure
  	1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Remove the top and bottom terminal block covers from the CPU. 3. Place a screwdriver into the slot on top of the CPU at the rear of the cover. 4. Gently pry the cover up and remove it from the CPU. 5. Place the module straight down into its mounting position in the top of the CPU. 6. Firmly press the module into position until it snaps into place. 7. Replace the terminal block covers.

Table 3- 5 Removing an SB, CB or BB 1297

Task	Procedure
  	1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Remove the top and bottom terminal block covers from the CPU. 3. Place a screwdriver into the slot on top of the module. 4. Gently pry the module up to disengage it from the CPU. 5. Remove the module straight up from its mounting position in the top of the CPU. 6. Replace the cover onto the CPU. 7. Replace the terminal block covers.

Installing or replacing the battery in the BB 1297 battery board

The BB 1297 requires battery type CR1025. The battery is not included with the BB 1297 and must be purchased by the user. To install or replace the battery, follow these steps:

1. In the BB 1297, install a new battery with the positive side of the battery on top, and the negative side next to the printed wiring board.

2. The BB 1297 is ready to be installed in the CPU. Follow the installation directions above to install the BB 1297.

To replace the battery in the BB 1297:

1. Remove the BB 1297 from the CPU following the removal directions above.
2. Carefully remove the old battery using a small screwdriver. Push the battery out from under the clip.
3. Install a new CR1025 replacement battery with the positive side of the battery on top and the negative side next to the printed wiring board.
4. Re-install the BB 1297 battery board following the installation directions above.

3.3.4 Installing and removing an SM

Table 3- 6 Installing an SM

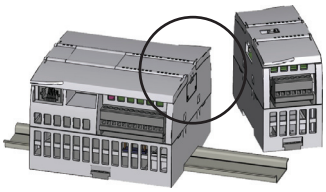
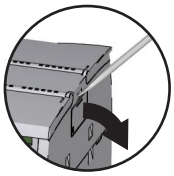
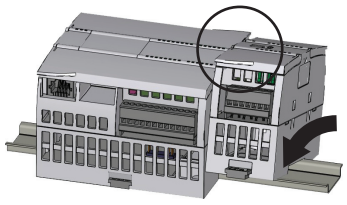
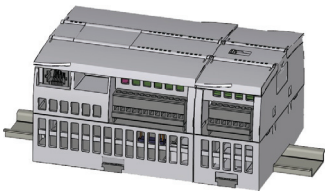
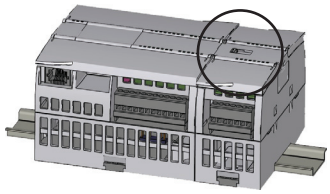
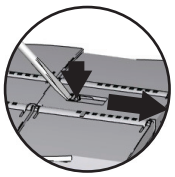
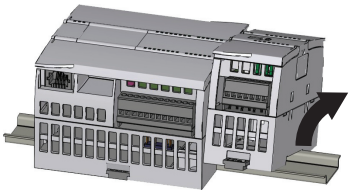
Task	Procedure	
		Install your SM after installing the CPU. 1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Remove the cover for the connector from the right side of the CPU. 3. Insert a screwdriver into the slot above the cover. 4. Gently pry the cover out at its top and remove the cover. Retain the cover for reuse.
		Connect the SM to the CPU: 1. Position the SM beside the CPU. 2. Hook the SM over the top of the DIN rail. 3. Pull out the bottom DIN rail clip to allow the SM to fit over the rail. 4. Rotate the SM down into position beside the CPU and push the bottom clip in to latch the SM onto the rail.
	Extending the bus connector makes both mechanical and electrical connections for the SM. 1. Place a screwdriver beside the tab on the top of the SM. 2. Slide the tab fully to the left to extend the bus connector into the CPU. Follow the same procedure to install a signal module to a signal module.	

Table 3- 7 Removing an SM

Task	Procedure
	You can remove any SM without removing the CPU or other SMs in place. 1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Remove the I/O connectors and wiring from the SM (Page 55). 3. Retract the bus connector. – Place a screwdriver beside the tab on the top of the SM. – Press down to disengage the connector from the CPU. – Slide the tab fully to the right. If there is another SM to the right, repeat this procedure for that SM.
	
	Remove the SM: 1. Pull out the bottom DIN rail clip to release the SM from the rail. 2. Rotate the SM up and off the rail. Remove the SM from the system. 3. If required, cover the bus connector on the CPU to avoid contamination. Follow the same procedure to remove a signal module from a signal module.

3.3.5 Installing and removing a CM or CP

Attach any communication modules to the CPU and install the assembly as a unit, as shown in Installing and removing the CPU (Page 49).

Table 3- 8 Installing a CM or CP

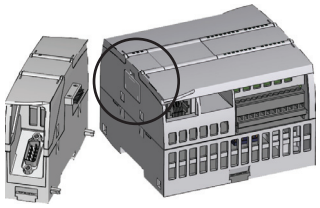
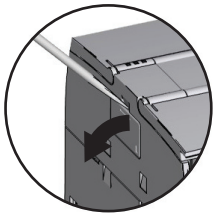
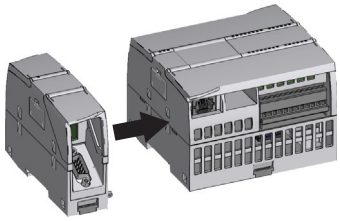
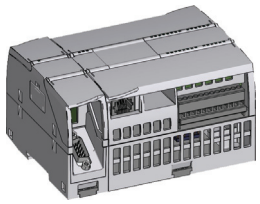
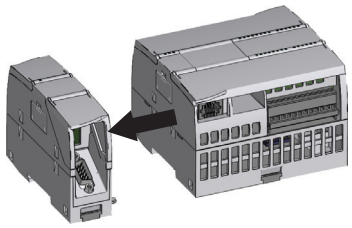
Task	Procedure
 	1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Attach the CM to the CPU before installing the assembly as a unit to the DIN rail or panel. 3. Remove the bus cover from the left side of the CPU: – Insert a screwdriver into the slot above the bus cover. – Gently pry out the cover at its top. 4. Remove the bus cover. Retain the cover for reuse. 5. Connect the CM or CP to the CPU: – Align the bus connector and the posts of the CM with the holes of the CPU – Firmly press the units together until the posts snap into place. 6. Install the CPU and CP on a DIN rail or panel.
	

Table 3- 9 Removing a CM or CP

Task	Procedure
	Remove the CPU and CM as a unit from the DIN rail or panel. 1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Remove the I/O connectors and all wiring and cables from the CPU and CMs. 3. For DIN rail mounting, move the lower DIN rail clips on the CPU and CMs to the extended position. 4. Remove the CPU and CMs from the DIN rail or panel. 5. Grasp the CPU and CMs firmly and pull apart.
	

CAUTION
Do not use a tool to separate the modules because this will damage the units.

3.3.6 Removing and reinstalling the S7-1200 terminal block connector

The CPU, SB and SM modules provide removable connectors to make connecting the wiring easy.

Table 3- 10 Removing the connector

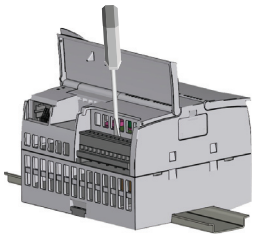
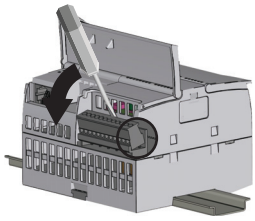
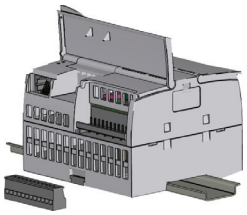
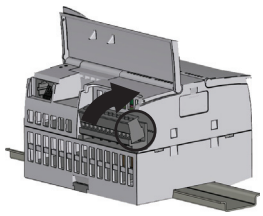
Task	Procedure
	Prepare the system for terminal block connector removal by removing the power from the CPU and opening the cover above the connector. 1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Inspect the top of the connector and locate the slot for the tip of the screwdriver. 3. Insert a screwdriver into the slot. 4. Gently pry the top of the connector away from the CPU. The connector will release with a snap. 5. Grasp the connector and remove it from the CPU.
	

Table 3- 11 Installing the connector

Task	Procedure
	Prepare the components for terminal block installation by removing power from the CPU and opening the cover for connector. 1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Align the connector with the pins on the unit. 3. Align the wiring edge of the connector inside the rim of the connector base. 4. Press firmly down and rotate the connector until it snaps into place. Check carefully to ensure that the connector is properly aligned and fully engaged.
	

3.3.7 Installing and removing the expansion cable

The S7-1200 expansion cable provides additional flexibility in configuring the layout of your S7-1200 system. Only one expansion cable is allowed per CPU system. You install the expansion cable either between the CPU and the first SM, or between any two SMs.

Table 3- 12 Installing and removing the male connector of the expansion cable

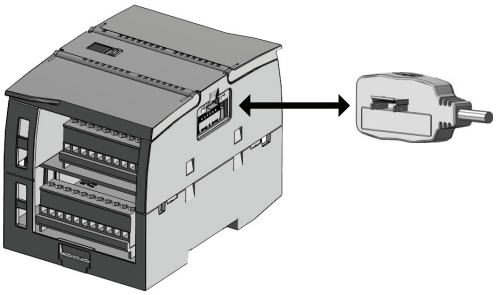
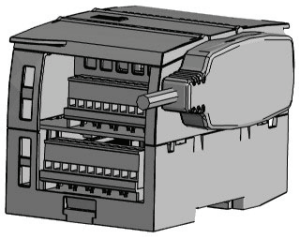
Task	Procedure
	To install the male connector: 1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Push the connector into the bus connector on the right side of the signal module or CPU. To remove the male connector: 1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Pull out the male connector to release it from the signal module or CPU.
	

Table 3- 13 Installing the female connector of the expansion cable

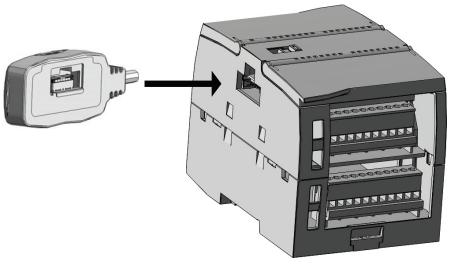
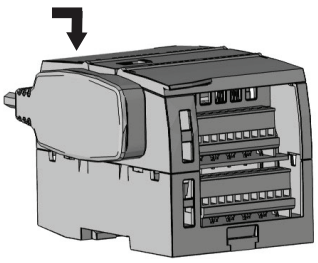
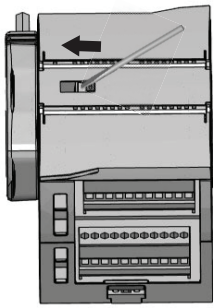
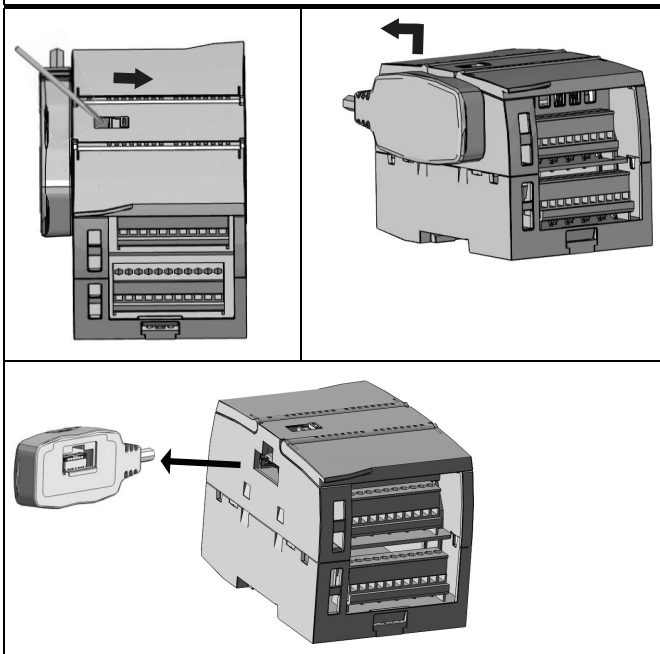
Task	Procedure
	1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Place the female connector to the bus connector on the left side of the signal module. 3. Slip the hook extension of the female connector into the housing at the bus connector and press down slightly to engage the hook. 4. Lock the connector into place: – Place a screwdriver beside the tab on the top of the signal module. – Slide the tab fully to the left. To engage the connector, you must slide the connector tab all the way to the left. The connector tab must be locked into place.
 	

Table 3- 14 Removing the female connector of the expansion cable

Task	Procedure
	1. Ensure that the CPU and all S7-1200 equipment are disconnected from electrical power. 2. Unlock the connector: – Place a screwdriver beside the tab on the top of the signal module. – Press down slightly and slide the tab fully to the right. 3. Lift the connector up slightly to disengage the hook extension. 4. Remove the female connector.

3.3.8 TS (teleservice) adapter

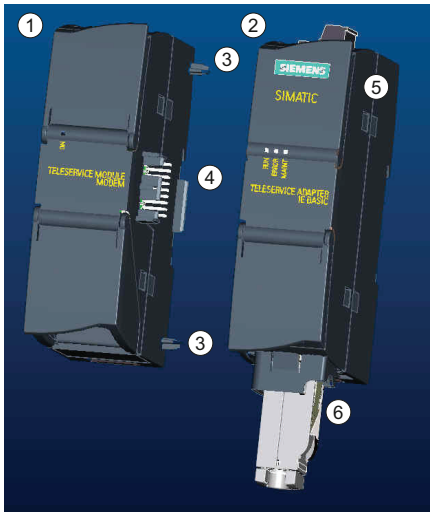
3.3.8.1 Connecting the TeleService Adapter

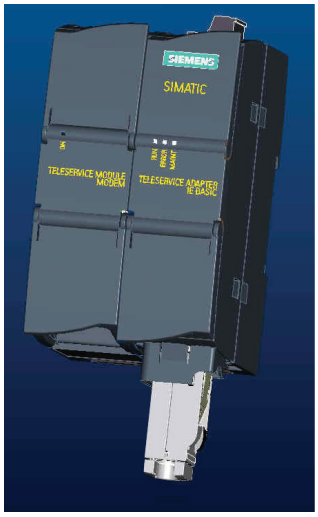
Before installing the TS (Teleservice) Adapter IE Basic, you must first connect the TS Adapter and a TS module.

Available TS modules:

- TS module RS232
- TS module Modem
- TS module GSM
- TS module ISDN

CAUTION
The TS module can be damaged if you touch the contacts of the plug connector ④of the TS module. Follow ESD guidelines in order to avoid damaging the TS module through electrostatic discharge. Before connecting a TS module and TS Adapter, make sure that both are in an idle state.





① TS module
② TS Adapter
③ Elements
④ Plug connector from the TS module
⑤ Cannot be opened
⑥ Ethernet port

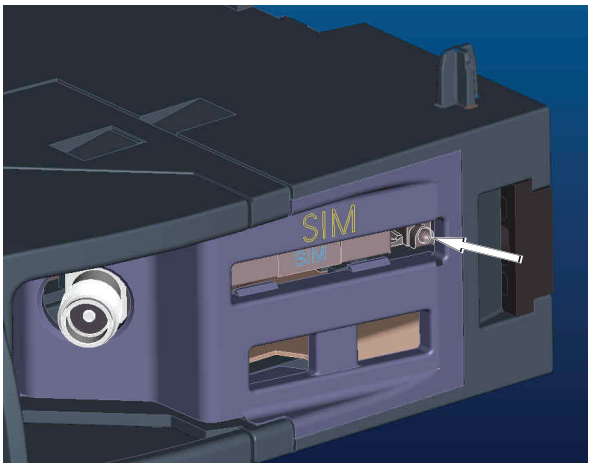
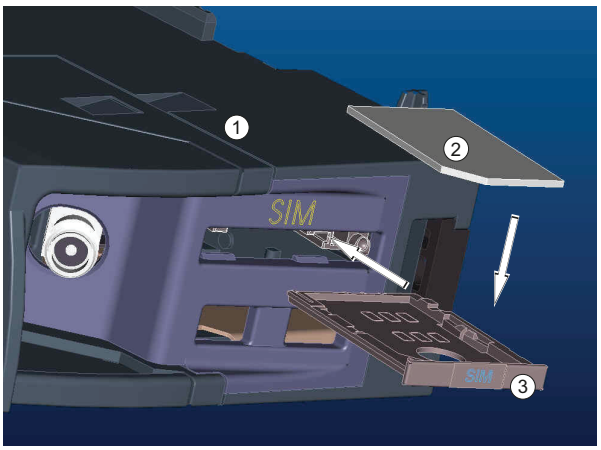
CAUTION
Before connecting a TS module and TS adapter basic unit, ensure that the contact pins ④ are not bent. When connecting, ensure that the male connector and guide elements are positioned correctly. Only connect a TS module into the TS adapter. Do not force a connection of the TS adapter to a different device, such as an S7-1200 CPU. Do not change the mechanical construction of the connector, and do not remove or damage the guide elements.

3.3.8.2 Installing the SIM card

Locate the SIM card slot on the underside of the TS module GSM.

NOTICE
The SIM card may only be removed or inserted if the TS module GSM is de-energized.

Table 3- 15 Installing the SIM card

Procedure	Task		
Use a sharp object to press the eject button of the SIM card tray (in the direction of the arrow) and remove the SIM card tray.			
Place the SIM card in the SIM card tray as shown and put the SIM card tray back into its slot.		①	TS Module GSM
		②	SIM card
		③	SIM card tray

Note

Ensure that the SIM card tray is correctly oriented in the card tray. Otherwise, the SIM card will not make connection with the module, and the eject button may not remove the card tray.

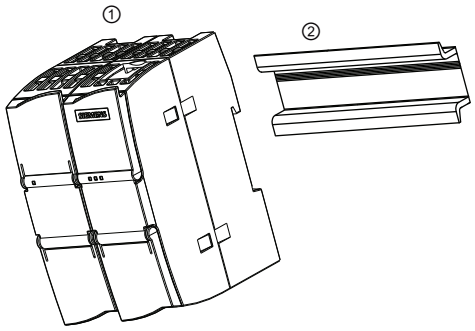
3.3.8.3 Installing the TS adapter unit

Prerequisites: You must have connected the TS Adapter and a TS module together, and the DIN rail must have been installed.

Note

If you install the TS unit vertically or in high-vibration environment, the TS module can become disconnected from the TS Adapter. Use an end bracket 8WA1 808 on the DIN rail to ensure that the modules remain connected.

Table 3- 16 Installing and removing the TS Adapter

Task	Procedure
	Installation: 1. Hook the TS Adapter with attached TS module ① on the DIN rail ②. 2. Rotate the unit back until it engages. 3. Push in the DIN rail clip on each module to attach the each module to the rail.
	Removal: 1. Remove the analog cable and Ethernet cable from the underside of the TS Adapter. 2. Remove power from the TS Adapter. 3. Use a screwdriver to disengage the rail clips on both modules. 4. Rotate the unit upwards to remove the unit from the DIN rail.

WARNING

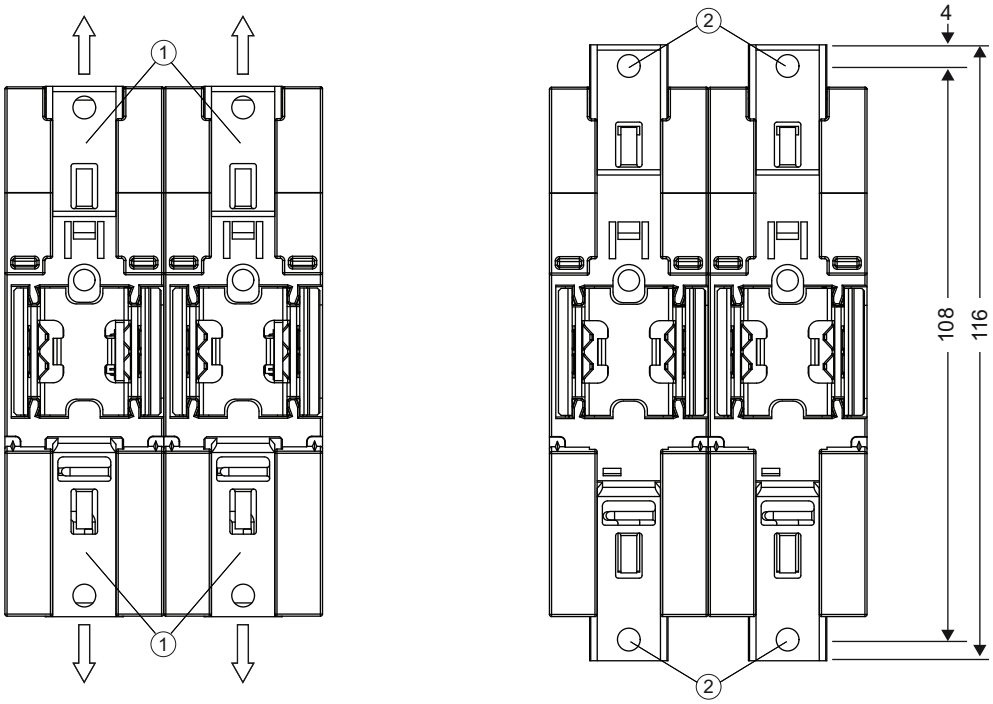
Before you remove power from the unit, disconnect the grounding of the TS Adapter by removing the analog cable and Ethernet cable.

3.3.8.4 Installing the TS adapter on a wall

Prerequisites: You must have connected the TS Adapter and TS module.

1. Move the attachment slider ① to the backside of the TS Adapter and TS module in the direction of the arrow until it engages.
2. Screw the TS Adapter and TS module to the position marked with ② to the designated assembly wall.

The following illustration shows the TS Adapter from behind, with the attachment sliders ① in both positions:



① Attachment slider
② Drill holes for wall mounting

3.4 Wiring guidelines

Proper grounding and wiring of all electrical equipment is important to help ensure the optimum operation of your system and to provide additional electrical noise protection for your application and the S7-1200. Refer to the technical specifications (Page 699) for the S7-1200 wiring diagrams.

Prerequisites

Before you ground or install wiring to any electrical device, ensure that the power to that equipment has been turned off. Also, ensure that the power to any related equipment has been turned off.

Ensure that you follow all applicable electrical codes when wiring the S7-1200 and related equipment. Install and operate all equipment according to all applicable national and local standards. Contact your local authorities to determine which codes and standards apply to your specific case.

WARNING

Installation or wiring the S7-1200 or related equipment with power applied could cause electric shock or unexpected operation of equipment. Failure to disable all power to the S7-1200 and related equipment during installation or removal procedures could result in death, severe personal injury, and/or damage due to electric shock or unexpected equipment operation.

Always follow appropriate safety precautions and ensure that power to the S7-1200 is disabled before attempting to install or remove the S7-1200 or related equipment.

Always take safety into consideration as you design the grounding and wiring of your S7-1200 system. Electronic control devices, such as the S7-1200, can fail and can cause unexpected operation of the equipment that is being controlled or monitored. For this reason, you should implement safeguards that are independent of the S7-1200 to protect against possible personal injury or equipment damage.

WARNING

Control devices can fail in an unsafe condition, resulting in unexpected operation of controlled equipment. Such unexpected operations could result in death, severe personal injury and/or property damage.

Use an emergency stop function, electromechanical overrides, or other redundant safeguards that are independent of the S7-1200.

Guidelines for isolation

S7-1200 AC power supply boundaries and I/O boundaries to AC circuits have been designed and approved to provide safe separation between AC line voltages and low voltage circuits. These boundaries include double or reinforced insulation, or basic plus supplementary insulation, according to various standards. Components which cross these boundaries such as optical couplers, capacitors, transformers, and relays have been approved as providing safe separation. Isolation boundaries which meet these requirements have been identified in S7-1200 product data sheets as having 1500 VAC or greater isolation. This designation is based on a routine factory test of (2Ue + 1000 VAC) or equivalent according to approved methods. S7-1200 safe separation boundaries have been type tested to 4242 VDC.

The sensor supply output, communications circuits, and internal logic circuits of an S7-1200 with included AC power supply are sourced as SELV (safety extra-low voltage) according to EN 61131-2.

To maintain the safe character of the S7-1200 low voltage circuits, external connections to communications ports, analog circuits, and all 24 V nominal power supply and I/O circuits must be powered from approved sources that meet the requirements of SELV, PELV, Class 2, Limited Voltage, or Limited Power according to various standards.

WARNING

Use of non-isolated or single insulation supplies to supply low voltage circuits from an AC line can result in hazardous voltages appearing on circuits that are expected to be touch safe, such as communications circuits and low voltage sensor wiring.

Such unexpected high voltages could cause electric shock resulting in death, severe personal injury and/or property damage.

Only use high voltage to low voltage power converters that are approved as sources of touch safe, limited voltage circuits.

Guidelines for grounding the S7-1200

The best way to ground your application is to ensure that all the common and ground connections of your S7-1200 and related equipment are grounded to a single point. This single point should be connected directly to the earth ground for your system.

All ground wires should be as short as possible and should use a large wire size, such as 2 mm^2 (14 AWG).

When locating grounds, consider safety-grounding requirements and the proper operation of protective interrupting devices.

Guidelines for wiring the S7-1200

When designing the wiring for your S7-1200, provide a single disconnect switch that simultaneously removes power from the S7-1200 CPU power supply, from all input circuits, and from all output circuits. Provide over-current protection, such as a fuse or circuit breaker, to limit fault currents on supply wiring. Consider providing additional protection by placing a fuse or other current limit in each output circuit.

Install appropriate surge suppression devices for any wiring that could be subject to lightning surges.

Avoid placing low-voltage signal wires and communications cables in the same wire tray with AC wires and high-energy, rapidly switched DC wires. Always route wires in pairs, with the neutral or common wire paired with the hot or signal-carrying wire.

Use the shortest wire possible and ensure that the wire is sized properly to carry the required current. The CPU and SM connector accepts wire sizes from 2 mm^2 to 0.3 mm^2 (14 AWG to 22 AWG). The SB connector accepts wire sizes from 1.3 mm^2 to 0.3 mm^2 (16 AWG to 22 AWG). Use shielded wires for optimum protection against electrical noise. Typically, grounding the shield at the S7-1200 gives the best results.

When wiring input circuits that are powered by an external power supply, include an overcurrent protection device in that circuit. External protection is not necessary for circuits that are powered by the 24 VDC sensor supply from the S7-1200 because the sensor supply is already current-limited.

All S7-1200 modules have removable connectors for user wiring. To prevent loose connections, ensure that the connector is seated securely and that the wire is installed securely into the connector. To avoid damaging the connector, be careful that you do not over-tighten the screws. The maximum torque for the CPU and SM connector screw is 0.56 N-m (5 inch-pounds). The maximum torque for the SB connector screw is 0.33 N-m (3 inch-pounds).

To help prevent unwanted current flows in your installation, the S7-1200 provides isolation boundaries at certain points. When you plan the wiring for your system, you should consider these isolation boundaries. Refer to the technical specifications for the amount of isolation provided and the location of the isolation boundaries. Do not depend on isolation boundaries rated less than 1500 VAC as safety boundaries.

Guidelines for lamp loads

Lamp loads are damaging to relay contacts because of the high turn-on surge current. This surge current will nominally be 10 to 15 times the steady state current for a Tungsten lamp. A replaceable interposing relay or surge limiter is recommended for lamp loads that will be switched a large number of times during the lifetime of the application.

Guidelines for inductive loads

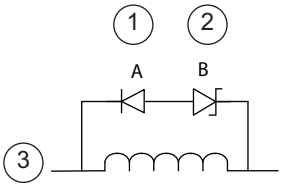
You should equip inductive loads with suppression circuits to limit voltage rise when the control output turns off. Suppression circuits protect your outputs from premature failure due to the high voltages associated with turning off inductive loads. In addition, suppression circuits limit the electrical noise generated when switching inductive loads. Placing an external suppression circuit so that it is electrically across the load, and physically located near the load is most effective in reducing electrical noise.

S7-1200 DC outputs include internal suppression circuits that are adequate for the inductive loads in most applications. Since S7-1200 relay output contacts can be used to switch either a DC or an AC load, internal protection is not provided.

Note

The effectiveness of a given suppression circuit depends on the application, and you must verify it for your particular use. Always ensure that all components used in your suppression circuit are rated for use in the application.

Typical suppressor circuit for DC or relay outputs that switch DC inductive loads



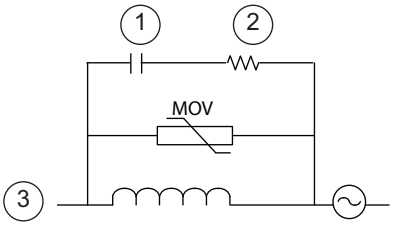
① 1N4001 diode or equivalent

② 8.2 V Zener (DC outputs),
36 V Zener (Relay outputs)

③ Output point

In most applications, the addition of a diode (A) across a DC inductive load is suitable, but if your application requires faster turn-off times, then the addition of a Zener diode (B) is recommended. Be sure to size your Zener diode properly for the amount of current in your output circuit.

Typical suppressor circuit for relay outputs that switch AC inductive loads



① 0.1 μ F

② 100 to 120 Ω

③ Output point

When you use a relay output to switch 115 V/230 VAC loads, place the appropriately rated resistor-capacitor-metal oxide varistor (MOV) circuit across the AC load. Ensure that the working voltage of the MOV is at least 20% greater than the nominal line voltage.

PLC concepts 4

4.1 Execution of the user program

The CPU supports the following types of code blocks that allow you to create an efficient structure for your user program:

- Organization blocks (OBs) define the structure of the program. Some OBs have predefined behavior and start events, but you can also create OBs with custom start events. Valid OB number ranges are shown in Event execution priorities and queuing (Page 75).
- Functions (FCs) and function blocks (FBs) contain the program code that corresponds to specific tasks or combinations of parameters. Each FC or FB provides a set of input and output parameters for sharing data with the calling block. An FB also uses an associated data block (called an instance DB) to maintain state of values between execution that can be used by other blocks in the program. Valid FC and FB numbers range from 1 to 65535.
- Data blocks (DBs) store data that can be used by the program blocks. Valid DB numbers range from 1 to 65535.

Execution of the user program begins with one or more optional start-up organization blocks (OBs) which are executed once upon entering RUN mode, followed by one or more program cycle OBs which are executed cyclically. An OB can also be associated with an interrupt event, which can be either a standard event or an error event, and executes whenever the corresponding standard or error event occurs.

A function (FC) or a function block (FB) is a block of program code that can be called from an OB or from another FC or FB, down to the following nesting depths:

- 16 from the program cycle or startup OB
- 4 from time delay interrupt, cyclic interrupt, time of day interrupt, hardware interrupt, time error interrupt, or diagnostic error interrupt OB

FCs are not associated with any particular data block (DB), while FBs are tied directly to a DB and use the DB for passing parameters and storing interim values and results.

The size of the user program, data, and configuration is limited by the available load memory and work memory in the CPU. There is no specific limit to the number of each individual OB, FC, FB and DB block. However, the total number of blocks is limited to 1024.

Each cycle includes writing the outputs, reading the inputs, executing the user program instructions, and performing background processing. The cycle is referred to as a scan cycle or scan.

The modules (SM, SB, BB, CB, CM or CP) are detected and logged in only upon power-up.

- Inserting or removing a module in the central rack under power (hot) is not supported. Never insert or remove a module from the central rack when the CPU has power..

> **WARNING**
>
> Insertion or removal of a module (SM, SB, BB, CD, CM or CP) from the central rack when the CPU has power could cause unpredictable behavior, resulting in damage to equipment and/or injury to personnel.
>
> Always ensure that power is removed from the CPU and central rack before inserting or removing a module from the central rack.

- You can insert or remove a SIMATIC memory card while the CPU is under power. However, inserting or removing a memory card when the CPU is in RUN mode causes the CPU to go to STOP mode.

> **CAUTION**
>
> Insertion or removal of a memory card when the CPU is in RUN mode causes the CPU to go to STOP, which might result in in damage to the equipment or the process being controlled.
>
> Whenever you insert or remove a memory card, the CPU immediately goes to STOP mode. Before inserting or removing a memory card, always ensure that the CPU is not actively controlling a machine or process. Always install an emergency stop circuit for your application or process.

- If you insert or remove a module in a distributed I/O rack (PROFINET or PROFIBUS) when the CPU is in RUN mode, the CPU generates an entry in the diagnostics buffer and stays in RUN mode.

Under the default configuration, all local digital and analog I/O points are updated synchronously with the scan cycle using an internal memory area called the process image. The process image contains a snapshot of the physical inputs and outputs (the physical I/O points on the CPU, signal board, and signal modules).

The CPU performs the following tasks:

- The CPU writes the outputs from the process image output area to the physical outputs.
- The CPU reads the physical inputs just prior to the execution of the user program and stores the input values in the process image input area. This ensures that these values remain consistent throughout the execution of the user instructions.
- The CPU executes the logic of the user instructions and updates the output values in the process image output area instead of writing to the actual physical outputs.

This process provides consistent logic through the execution of the user instructions for a given cycle and prevents the flickering of physical output points that might change state multiple times in the process image output area.

You can specify whether digital and analog I/O points are to be automatically updated and stored in the process image. If you insert a module in the device view, its data is located in the process image of the CPU (default). The CPU handles the data exchange between the module and the process image area automatically during the update of the process image. To remove digital or analog points from the process-image automatic update, select the appropriate device in Device configuration, view the Properties tab, expand if necessary to locate the desired I/O points, and then select "IO addresses/HW identifier". Then change the entry for "Process image:" from "Cyclic PI" to "---". To add the points back to the process-image automatic update, change this selection back to "Cyclic PI".

You can immediately read physical input values and immediately write physical output values when an instruction executes. An immediate read accesses the current state of the physical input and does not update the process image input area, regardless of whether the point is configured to be stored in the process image. An immediate write to the physical output updates both the process image output area (if the point is configured to be stored in the process image) and the physical output point. Append the suffix ":P" to the I/O address if you want the program to immediately access I/O data directly from the physical point instead of using the process image.

The CPU supports distributed I/O for both PROFINET and PROFIBUS networks (Page 423).

4.1.1 Operating modes of the CPU

The CPU has three modes of operation: STOP mode, STARTUP mode, and RUN mode. Status LEDs on the front of the CPU indicate the current mode of operation.

- In STOP mode, the CPU is not executing the program. You can download a project.
- In STARTUP mode, the startup OBs (if present) are executed once. Interrupt events are not processed during the startup mode.
- In RUN mode, the program cycle OBs are executed repeatedly. Interrupt events can occur and be processed at any point within the RUN mode. Some parts of a project can be downloaded in RUN mode (Page 690).

The CPU supports a warm restart for entering the RUN mode. Warm restart does not include a memory reset. All non-retentive system and user data are initialized at warm restart. Retentive user data is retained.

A memory reset clears all work memory, clears retentive and non-retentive memory areas, and copies load memory to work memory. A memory reset does not clear the diagnostics buffer or the permanently saved values of the IP address.

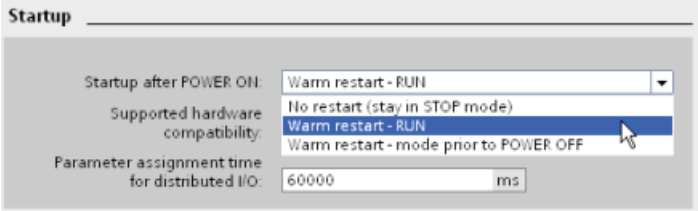
Note

When you download one or more DBs from STEP 7 V11 to an S7-1200 V2 CPU, the retentive and non-retentive values of those DBs are set to their start values. The next transition to RUN performs a warm restart, setting all non-retentive data to their start values and setting all retentive data to their retained values.

When you download project elements (such as device configuration, code blocks or DBs) from STEP 7 V10.5 to any S7-1200 CPU or from STEP 7 V11 to an S7-1200 V1 CPU (or a V2 CPU that has been configured as a V1 CPU), the next transition to RUN mode resets **all** of the DBs in the project to their start values.

You can configure the "startup after POWER ON" setting of the CPU. This configuration item appears under the "Device configuration" for the CPU under "Startup". When power is applied, the CPU performs a sequence of power-up diagnostic checks and system initialization. During system initialization the CPU deletes all non-retentive bit memory and resets all non-retentive DB contents to the initial values from load memory. The CPU retains retentive bit memory and retentive DB contents and then enters the appropriate operating mode. Certain detected errors prevent the CPU from entering the RUN mode. The CPU supports the following configuration choices:

- No restart (stay in STOP mode)
- Warm restart - RUN
- Warm restart - mode prior to POWER OFF



CAUTION

The CPU can enter STOP mode due to repairable faults, such as failure of a replaceable signal module, or temporary faults, such as power line disturbance or erratic power up event.

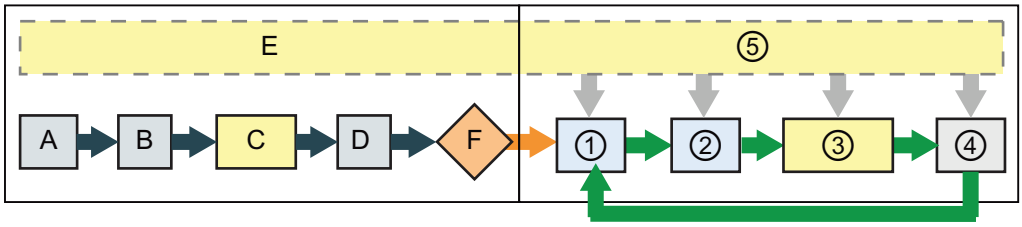
If the CPU has been configured to "Warm restart mode prior to POWER OFF", it will not return to RUN mode when the fault is repaired or removed until it receives a new command from STEP 7 to go to RUN. Without a new command, the STOP mode is retained as the mode prior to POWER OFF.

CPUs that are intended to operate independently of a STEP 7 connection should typically be configured to "Warm restart - RUN" so that the CPU can be returned to RUN mode by a power cycle following the removal of fault conditions.

You can change the current operating mode using the "STOP" or "RUN" commands from the online tools of the programming software. You can also include a STP instruction in your program to change the CPU to STOP mode. This allows you to stop the execution of your program based on the program logic.

- In STOP mode, the CPU handles any communication requests (as appropriate) and performs self-diagnostics. The CPU does not execute the user program, and the automatic updates of the process image do not occur.

  You can download your project only when the CPU is in STOP mode.
- In STARTUP and RUN modes, the CPU performs the tasks shown in the following figure.



STARTUP

A	Clears the I (image) memory area
B	Initializes the outputs with either the last value or the substitute value
C	Executes the startup OBs
D	Copies the state of the physical inputs to I memory
E	Stores any interrupt events into the queue to be processed after entering RUN mode
F	Enables the writing of Q memory to the physical outputs

RUN

①	Writes Q memory to the physical outputs
②	Copies the state of the physical inputs to I memory
③	Executes the program cycle OBs
④	Performs self-test diagnostics
⑤	Processes interrupts and communications during any part of the scan cycle

STARTUP processing

Whenever the operating mode changes from STOP to RUN, the CPU clears the process image inputs, initializes the process image outputs and processes the startup OBs. Any read accesses to the process-image inputs by instructions in the startup OBs read zero rather than the current physical input value. Therefore, to read the current state of a physical input during the startup mode, you must perform an immediate read. The startup OBs and any associated FCs and FBs are executed next. If more than one startup OB exists, each is executed in order according to the OB number, with the lowest OB number executing first.

Each startup OB includes startup information that helps you determine the validity of retentive data and the time-of-day clock. You can program instructions inside the startup OBs to examine these startup values and to take appropriate action. The following startup locations are supported by the Startup OBs:

Table 4- 1 Startup locations supported by the startup OB

Input	Data Type	Description
LostRetentive	Bool	This bit is true if the retentive data storage areas have been lost
LostRTC	Bool	This bit is true if the time-of-day clock (Real time Clock) has been lost

The CPU also performs the following tasks during the startup processing.

- Interrupts are queued but not processed during the startup phase
- No cycle time monitoring is performed during the startup phase
- Configuration changes to HSC (high-speed counter), PWM (pulse-width modulation), and PtP (point-to-point communication) modules can be made in startup
- Actual operation of HSC, PWM and point-to-point communication modules only occurs in RUN

After the execution of the startup OBs finishes, the CPU goes to RUN mode and processes the control tasks in a continuous scan cycle.

See also

Stop scan cycle instruction (Page 235)

CPU operator panel for the online CPU (Page 679)

4.1.2 Processing the scan cycle in RUN mode

For each scan cycle, the CPU writes the outputs, reads the inputs, executes the user program, updates communication modules, and responds to user interrupt events and communication requests. Communication requests are handled periodically throughout the scan.

These actions (except for user interrupt events) are serviced regularly and in sequential order. User interrupt events which are enabled are serviced according to priority in the order in which they occur.

The system guarantees that the scan cycle will be completed in a time period called the maximum cycle time; otherwise a time error event is generated.

- Each scan cycle begins by retrieving the current values of the digital and analog outputs from the process image and then writing them to the physical outputs of the CPU, SB, and SM modules configured for automatic I/O update (default configuration). When a physical output is accessed by an instruction, both the output process image and the physical output itself are updated.
- The scan cycle continues by reading the current values of the digital and analog inputs from the CPU, SB, and SMs configured for automatic I/O update (default configuration), and then writing these values to the process image. When a physical input is accessed by an instruction, the value of the physical input is accessed by the instruction, but the input process image is not updated.
- After reading the inputs, the user program is executed from the first instruction through the end instruction. This includes all the program cycle OBs plus all their associated FCs and FBs. The program cycle OBs are executed in order according to the OB number with the lowest OB number executing first.

Communications processing occurs periodically throughout the scan, possibly interrupting user program execution.

Self-diagnostic checks include periodic checks of the system and the I/O module status checks.

Interrupts can occur during any part of the scan cycle, and are event-driven. When an event occurs, the CPU interrupts the scan cycle and calls the OB that was configured to process that event. After the OB finishes processing the event, the CPU resumes execution of the user program at the point of interruption.

4.1.3 Organization blocks (OBs)

OBs control the execution of the user program. Each OB must have a unique OB number. The default OB numbers are reserved below 200. Other OBs must be numbered 200 or greater.

Specific events in the CPU trigger the execution of an organization block. OBs cannot call each other or be called from an FC or FB. Only a start event, such as a diagnostic interrupt or a time interval, can start the execution of an OB. The CPU handles OBs according to their respective priority classes, with higher priority OBs executed before lower priority OBs. The lowest priority class is 1 (for the main program cycle), and the highest priority class is 26 (for the time-error interrupts).

OBs control the following operations:

- Program cycle OBs execute cyclically while the CPU is in RUN mode. The main block of the program is a program cycle OB. This is where you place the instructions that control your program and where you call additional user blocks. Multiple program cycle OBs are allowed and are executed in numerical order. OB 1 is the default. Other program cycle OBs must be identified as OB 200 or greater.
- Startup OBs execute one time when the operating mode of the CPU changes from STOP to RUN, including powering up in the RUN mode and in commanded STOP-to-RUN transitions. After completion, the main "Program cycle" OB will begin executing. Multiple startup OBs are allowed. OB 100 is the default. Others must be OB 200 or greater.
- Cyclic interrupt OBs execute at a specified interval. A cyclic interrupt OB will interrupt cyclic program execution at user defined intervals, such as every 2 seconds. You can configure up to a total of 4 for both the time-delay and cyclic events at any given time, with one OB allowed for each configured time-delay and cyclic event. The OB must be OB 200 or greater.
- Hardware interrupt OBs execute when the relevant hardware event occurs, including rising and falling edges on built-in digital inputs and HSC events. A hardware interrupt OB will interrupt normal cyclic program execution in reaction to a signal from a hardware event. You define the events in the properties of the hardware configuration. One OB is allowed for each configured hardware event. The OB must be OB 200 or greater.
- A time error interrupt OB executes when either the maximum cycle time is exceeded or a time error event occurs. The OB for processing the time error interrupt is OB 80. If triggered, it executes, interrupting normal cyclic program execution or any other event OB. The events that trigger the time error interrupt and the reaction of the CPU to those events are described below:
  - Exceeding the maximum cycle time: You configure the maximum cycle time in the properties of the CPU. If OB 80 does not exist, the reaction of the CPU for exceeding the maximum time is to change to STOP.
  - Time errors: If OB 80 does not exist, the reaction of the CPU is to stay in RUN. Time errors occur if the time of day event is missed or repeated, the queue overflows, or an event OB (time delay event, time of day event, or cyclic interrupt) starts before the CPU finishes the execution of the first.

  The occurrence of either of these events generates a diagnostic buffer entry describing the event. The diagnostic buffer entry is generated regardless of the existence of OB 80.
- Diagnostic error interrupt OBs execute when a diagnostic error is detected and reported. A diagnostic OB interrupts the normal cyclic program execution if a diagnostics-capable module recognizes an error (if the diagnostic error interrupt has been enabled for the module). OB 82 is the only OB number supported for the diagnostic error event. You can include an STP instruction (put CPU in STOP mode) inside your OB 82 if you desire your CPU to enter STOP mode upon receiving this type of error. If there is no diagnostic OB in the program, the CPU ignores the error (stays in RUN).

4.1.4 Event execution priorities and queuing

The CPU processing is controlled by events. An event triggers an interrupt OB to be executed. You can specify the interrupt OB for an event during the creation of the block, during the device configuration, or with an ATTACH or DETACH instruction. Some events happen on a regular basis like the program cycle or cyclic events. Other events happen only a single time, like the startup event and time delay events. Some events happen when there is a change triggered by the hardware, such as an edge event on an input point or a high speed counter event. There are also events like the diagnostic error and time error event which only happen when there is an error. The event priorities and queues are used to determine the processing order for the event interrupt OBs.

The program cycle event happens once during each program cycle (or scan). During the program cycle, the CPU writes the outputs, reads the inputs and executes program cycle OBs. The program cycle event is required and is always enabled. You may have no program cycle OBs, or you may have multiple OBs selected for the program cycle event. After the program cycle event is triggered, the lowest numbered program cycle OB (usually OB 1) is executed. The other program cycle OBs are executed sequentially (in numerical order) within the program cycle.

The cyclic interrupt events allow you to configure the execution of an interrupt OB at a configured scan time. The initial scan time is configured when the OB is created and selected to be a cyclic interrupt OB. A cyclic event will interrupt the program cycle and execute the cyclic interrupt OB (the cyclic event is at a higher priority class than the program cycle event).

Only one cyclic interrupt OB can be attached to a cyclic event.

Each cyclic event can be assigned a phase shift so that the execution of cyclic interrupts with the same scan time can be offset from one another by the phase shift amount. The default phase shift is 0. To change the initial phase shift, or to change the initial scan time for a cyclic event, right click on the cyclic interrupt OB in the project tree, click "Properties", then click "Cyclic interrupt", and enter the new initial values. You can also query and change the scan time and the phase shift from your program using the Query cyclic interrupt (QRY_CINT) and Set cyclic interrupt (SET_CINT) instructions. Scan time and phase shift values set by the SET_CINT instruction do not persist through a power cycle or a transition to STOP mode; scan time and phase shift values will return to the initial values following a power cycle or a transition to STOP. The CPU supports a total of four cyclic and time-delay interrupt events.

The startup event happens one time on a STOP to RUN transition and causes the startup OBs to be executed. Multiple OBs can be selected for the startup event. The startup OBs are executed in numerical order.

The time delay interrupt events allow you to configure the execution of an interrupt OB after a specified delay time has expired. The delay time is specified with the SRT_DINT instruction. The time delay events will interrupt the program cycle to execute the time delay interrupt OB. Only one time delay interrupt OB can be attached to a time delay event. The CPU supports four time delay events.

The hardware interrupt events are triggered by a change in the hardware, such as a rising or falling edge on an input point, or a HSC (High Speed Counter) event. There can be one interrupt OB selected for each hardware interrupt event. The hardware events are enabled in Device configuration. The OBs are specified for the event in the Device configuration or with an ATTACH instruction in the user program. The CPU supports several hardware interrupt events. The exact events are based on the CPU model and the number of input points.

The time and diagnostic error interrupt events are triggered when the CPU detects an error. These events are at a higher priority class that the other interrupt events and can interrupt the execution of the time delay, cyclic and hardware interrupt events. One interrupt OB can be specified for each of the time error and diagnostic error interrupt events.

Understanding event execution priorities and queuing

The number of pending (queued) events from a single source is limited, using a different queue for each event type. Upon reaching the limit of pending events for a given event type, the next event is lost. Refer to the following section on "Understanding time error events" for more information regarding queue overflows.

Each CPU event has an associated priority. You cannot change the priority of an OB. In general, events are serviced in order of priority (highest priority first). Events of the same priority are serviced on a "first-come, first-served" basis.

Table 4- 2 OB events

Event	OB number	Quantity allowed	Start event	OB priority
Program cycle	OB 1, OB 200 to OB 65535	1 program cycle event Multiple OBs allowed	• Startup OB ends • Last program cycle OB ends	1
Startup	OB 100, OB 200 to OB 65535	1 startup event [1, 2] Multiple OBs allowed	STOP-to-RUN transition	1
Time	OB 200 to OB 65535	Up to 4 time events[3] 1 OB per event	Time-delay OB event is scheduled	3
			Cyclic OB event is scheduled	7
Process	OB 200 to OB 65535	Up to 50 process events[4] 1 OB per event	Edges: • Rising edge events: 16 max. • Falling edge events: 16 max.	5
			For HSC: • CV=PV: 6 max. • Direction changed: 6 max. • External reset: 6 max.	6

Event	OB number	Quantity allowed	Start event	OB priority
Diagnostic error	OB 82	1 event (only if OB 82 was loaded)	Module transmits an error	9
Time error	OB 80	1 event (only if OB 80 was loaded)[5]	• Maximum cycle time was exceeded • A second time interrupt (cyclic or time-delay) started before the CPU had finished execution of the first interrupt	26

1 The startup event and the program cycle event will never occur at the same time because the startup event will run to completion before the program cycle event will be started (controlled by the operating system).

2 Only the diagnostic error event (OB 82) interrupts the startup event. All other events are queued to be processed after the startup event has finished.

3 The CPU provides a total of 4 time events that are shared by the time-delay OBs and the cyclic OBs. The number of time-delay and cyclic OBs in your user program cannot exceed 4.

4 You can have more than 50 process events if you use the DETACH and ATTACH instructions.

5 You can configure the CPU to stay in RUN if the maximum scan cycle time was exceeded or you can use the RE_TRIGR instruction to reset the cycle time. However, the CPU goes to STOP mode the second time that the maximum scan cycle time was exceeded in one scan cycle.

After the execution of an OB with a priority of 2 to 25 has started, processing of that OB cannot be interrupted by the occurrence of another event, except for by OB 80 (time-error event, which has a priority of 26). All other events are queued for later processing, allowing the current OB to finish.

Interrupt latency

The interrupt event latency (the time from notification of the CPU that an event has occurred until the CPU begins execution of the first instruction in the OB that services the event) is approximately 175 µsec, provided that a program cycle OB is the only event service routine active at the time of the interrupt event.

Understanding time error events

The occurrence of any of several different time error conditions results in a time error event. The following time errors are supported:

- Maximum cycle time exceeded
- Requested OB cannot be started
- Queue overflow occurred

The maximum cycle time exceeded condition results if the program cycle does not complete within the specified maximum scan cycle time. See the section on "Monitoring the cycle time in the S7-1200 System Manual" (Page 80) for more information regarding the maximum cycle time condition, how to configure the maximum scan cycle time, and how to reset the cycle timer.

The requested OB cannot be started condition results if an OB is requested by a cyclic interrupt, a time-delay interrupt, or a time-of-day interrupt, but the requested OB is already being executed.

The queue overflow occurred condition results if the interrupts are occurring faster than they can be processed. The number of pending (queued) events is limited using a different queue for each event type. If an event occurs when the corresponding queue is full, a time error event is generated.

All time error events trigger the execution of OB 80 if it exists. If an OB 80 is not included in the user program, then the device configuration of the CPU determines the CPU reaction to the time error:

- The default configuration for time errors, such as starting a second cyclic interrupt before the CPU has finished the execution of the first, is for the CPU to stay in RUN.
- The default configuration for exceeding the maximum time is for the CPU to change to STOP.

You can use the RE_TRIGR instruction to reset the maximum cycle time. However, if two "maximum cycle time exceeded" conditions occur within the same program cycle without resetting the cycle timer, then the CPU transitions to STOP, regardless of whether OB 80 exists. See the section on "Monitoring the cycle time in the S7-1200 System Manual" (Page 80).

OB 80 includes startup information that helps you determine which event and OB generated the time error. You can program instructions inside OB 80 to examine these startup values and to take appropriate action.

Table 4- 3 Startup information for OB 80

Input	Data type	Description
fault_id	BYTE	16#01 - maximum cycle time exceeded 16#02 - requested OB cannot be started 16#07 and 16#09 - queue overflow occurred
csg_OBnr	OB_ANY	Number of the OB which was being executed when the error occurred
csg_prio	UINT	Priority of the OB causing the error

No time error interrupt OB 80 is present when you create a new project. If desired, you add a time error interrupt OB 80 to your project by double-clicking "Add new block" under "Program blocks" in the tree, then choose "Organization block", and then "Time error interrupt".

Understanding diagnostic error events

Analog (local), PROFINET, and PROFIBUS devices are capable of detecting and reporting diagnostic errors. The occurrence or removal of any of several different diagnostic error conditions results in a diagnostic error event. The following diagnostic errors are supported:

- No user power
- High limit exceeded
- Low limit exceeded

- Wire break
- Short circuit

Diagnostic error events trigger the execution of OB 82 if it exists. If OB 82 does not exist, then the CPU ignores the error. No diagnostic error interrupt OB 82 is present when you create a new project. If desired, you add a diagnostic error interrupt OB 82 to your project by double-clicking "Add new block" under "Program blocks" in the tree, then choose "Organization block", and then "Diagnostic error interrupt".

Note

Diagnostic errors for multi-channel local analog devices (I/O, RTD, and Thermocouple)

The OB 82 diagnostic error interrupt can report only one channel's diagnostic error at a time.

If two channels of a multi-channel device have an error, then the second error only triggers OB 82 under the following conditions: the first channel error clears, the execution of OB 82 triggered by the first error is complete, and the second error still exists.

OB 82 includes startup information that helps you determine whether the event is due to the occurrence or removal of an error, and the device and channel which reported the error. You can program instructions inside OB 82 to examine these startup values and to take appropriate action.

Table 4- 4 Startup information for OB 82

Input	Data type	Description
IOstate	WORD	IO state of the device: • Bit 0 = 1 if the configuration is correct, and = 0 if the configuration is no longer correct. • Bit 4 = 1 if an error is present (such as a wire break). (Bit 4 = 0 if there is no error.) • Bit 5 = 1 if the configuration is **not** correct, and = 0 if the configuration is correct again. • Bit 6 = 1 if an I/O access error has occurred. Refer to laddr for the hardware identifier of the I/O with the access error. (Bit 6 = 0 if there is no error.)
laddr	HW_ANY	Hardware identifier of the device or functional unit that reported the error[1]
channel	UINT	Channel number
multierror	BOOL	TRUE if more than one error is present

[1] The laddr input contains the hardware identifier of the device or functional unit which returned the error. The hardware identifier is assigned automatically when components are inserted in the device or network view and appears in the Constants tab of PLC tags. A name is also assigned automatically for the hardware identifier. These entries in the Constants tab of the PLC tags cannot be changed.

4.1.5 Monitoring the cycle time

The cycle time is the time that the CPU operating system requires to execute the cyclic phase of the RUN mode. The CPU provides two methods of monitoring the cycle time:

- Maximum scan cycle time
- Fixed minimum scan cycle time

Scan cycle monitoring begins after the startup event is complete. Configuration for this feature appears under the "Device Configuration" for the CPU under "Cycle time".

The CPU always monitors the scan cycle and reacts if the maximum scan cycle time is exceeded. If the configured maximum scan cycle time is exceeded, an error is generated and is handled one of two ways:

- If the user program does not include an OB 80, then the CPU generates an error and goes to STOP. (You can change the configuration of the CPU to ignore this time error and stay in RUN. The default configuration is for the CPU to go to STOP.)
- If the user program includes an OB 80, then the CPU executes OB 80

The RE_TRIGR instruction (Re-trigger cycle time monitoring) allows you to reset the timer that measures the cycle time. However, this instruction only functions if executed in a program cycle OB; the RE_TRIGR instruction is ignored if executed in OB 80. If the maximum scan cycle time is exceeded twice within the same program cycle with no RE_TRIGR instruction execution between the two, then the CPU transitions to STOP immediately. The use of repeated executions of the RE_TRIGR instruction can create an endless loop or a very long scan.

Typically, the scan cycle executes as fast as it can be executed and the next scan cycle begins as soon as the current one completes. Depending upon the user program and communication tasks, the time period for a scan cycle can vary from scan to scan. To eliminate this variation, the CPU supports an optional fixed minimum scan cycle time (also called fixed scan cycle). When this optional feature is enabled and a fixed minimum scan cycle time is provided in ms, the CPU will maintain the minimum cycle time within ±1 ms for the completion of each CPU scan.

In the event that the CPU completes the normal scan cycle in less time than the specified minimum cycle time, the CPU spends the additional time of the scan cycle performing runtime diagnostics and/or processing communication requests. In this way the CPU always takes a fixed amount of time to complete a scan cycle.

In the event that the CPU does not complete the scan cycle in the specified minimum cycle time, the CPU completes the scan normally (including communication processing) and does not create any system reaction as a result of exceeding the minimum scan time. The following table defines the ranges and defaults for the cycle time monitoring functions.

Table 4- 5 Range for the cycle time

Cycle time	Range (ms)	Default
Maximum scan cycle time[1]	1 to 6000	150 ms
Fixed minimum scan cycle time[2]	1 to maximum scan cycle time	Disabled

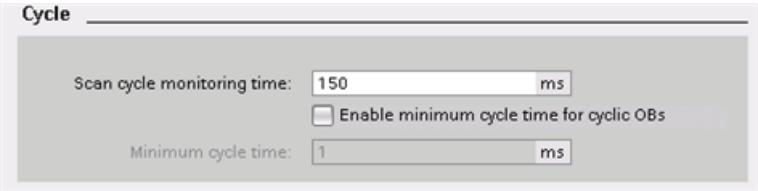
[1] The maximum scan cycle time is always enabled. Configure a cycle time between 1 ms to 6000 ms. The default is 150 ms.

[2] The fixed minimum scan cycle time is optional, and is disabled by default. If required, configure a cycle time between 1 ms and the maximum scan cycle time.

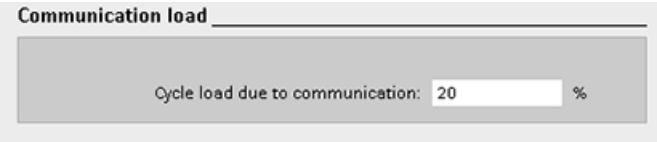
Configuring the cycle time and communication load

You use the CPU properties in the Device configuration to configure the following parameters:

- Cycle time: You can enter a maximum scan cycle time. You can also enter a fixed minimum scan cycle time.



- Communications load: You can configure a percentage of the time to be dedicated for communication tasks.



For more information about the scan cycle, see "Monitoring the cycle time". (Page 80)

4.1.6 CPU memory

Memory management

The CPU provides the following memory areas to store the user program, data, and configuration:

- Load memory is non-volatile storage for the user program, data and configuration. When a project is downloaded to the CPU, it is first stored in the Load memory area. This area is located either in a memory card (if present) or in the CPU. This non-volatile memory area is maintained through a power loss. The memory card supports a larger storage space than that built-in to the CPU.
- Work memory is volatile storage for some elements of the user project while executing the user program. The CPU copies some elements of the project from load memory into work memory. This volatile area is lost when power is removed, and is restored by the CPU when power is restored.
- Retentive memory is non-volatile storage for a limited quantity of work memory values. The retentive memory area is used to store the values of selected user memory locations during power loss. When a power down or power loss occurs, the CPU restores these retentive values upon power up.

To display the memory usage for the current project, right-click the CPU (or one of its blocks) and select "Resources" from the context. To display the memory usage for the current CPU, double-click "Online and diagnostics", expand "Diagnostics", and select "Memory".

Retentive memory

Data loss after power failure can be avoided by marking certain data as retentive. The following data can be configured to be retentive:

- Bit memory(M): You can define the precise width of the memory for bit memory in the PLC tag table or in the assignment list. Retentive bit memory always starts at MB0 and runs consecutively up through a specified number of bytes. Specify this value from the PLC tag table or in the assignment list by clicking the "Retain" toolbar icon. Enter the number of M bytes to retain starting at MB0.
- Tags of a function block (FB): If an FB was created with "Optimzed" selected, then the interface editor for this FB includes a "Retain" column. In this column, you can select either "Retentive", "Non-retentive", or "Set in IDB" individually for each tag. An instance DB that was created when this FB is placed in the program editor shows this retain column as well. You can only change the retentive state of a tag from within the instance DB interface editor if you selected "Set in IDB" (Set in instance data block) in the Retain selection for the tag in the optimized FB.

  If an FB was created with "Standard - compatible with S7-300/400" selected, then the interface editor for this FB does not include a "Retain" column. An instance DB created when this FB is inserted in the program editor shows a "Retain" column which is available for edit. In this case, selecting the "Retain" option for any tag results in **all** tags being selected. Similarly, deselecting the option for any tag results in **all** tags being deselected. For an FB that was configured to be "Standard - compatible with S7-300/400", you can change the retentive state from within the instance DB editor, but all tags are set to the same retentive state together.

  After you create the FB, you cannot change the option for "Standard - compatible with S7-300/400". You can only select this option when you create the FB. To determine whether an existing FB was configured for "Optimized" or "Standard - compatible with S7-300/400", right-click the FB in the Project tree, select "Properties", and then select "Attributes". The check box "Optimized block access" when selected shows you whether a block is optimized. Otherwise, it is standard and compatible with S7-300/400 CPUs.
- Tags of a global data block: The behavior of a global DB with regard to retentive state assignment is similar to that of an FB. Depending on the block access setting you can define the retentive state either for individual tags or for all tags of a global data block.
  - If you select "Optimized" when you create the DB, you can set the retentive state for each individual tag.
  - If you select "Standard - compatible with S7-300/400" when you create the DB, the retentive-state setting applies to all tags of the DB; either all tags are retentive or no tag is retentive.

A total of 10240 bytes of data can be retentive. To see how much is available, from the PLC tag table or the assignment list, click on the "Retain" toolbar icon. Although this is where the retentive range is specified for M memory, the second row indicates the total remaining memory available for M and DB combined. Note that for this value to be accurate, you must compile all data blocks with retentive tags.

4.1.6.1 System and clock memory

You use the CPU properties to enable bytes for "system memory" and "clock memory". Your program logic can reference the individual bits of these functions by their tag names.

- You can assign one byte in M memory for system memory. The byte of system memory provides the following four bits that can be referenced by your user program by the following tag names:
  - First cycle: (Tag name "FirstScan") bit is set to1 for the duration of the first scan after the startup OB finishes. (After the execution of the first scan, the "first scan" bit is set to 0.)
  - Diagnostics status changed (Tag name: "DiagStatusUpdate") is set to 1 for one scan after the CPU logs a diagnostic event. Because the CPU does not set the "diagnostic graph changed" bit until the end of the first execution of the program cycle OBs, your user program cannot detect if there has been a diagnostic change either during the execution of the startup OBs or the first execution of the program cycle OBs.
  - Always 1 (high): (Tag name "AlwaysTRUE") bit is always set to 1.
  - Always 0 (low): (Tag name "AlwaysFALSE") bit is always set to 0.
- You can assign one byte in M memory for clock memory. Each bit of the byte configured as clock memory generates a square wave pulse. The byte of clock memory provides 8 different frequencies, from 0.5 Hz (slow) to 10 Hz (fast). You can use these bits as control bits, especially when combined with edge instructions, to trigger actions in the user program on a cyclic basis.

The CPU initializes these bytes on the transition from STOP mode to STARTUP mode. The bits of the clock memory change synchronously to the CPU clock throughout the STARTUP and RUN modes.

CAUTION

Overwriting the system memory or clock memory bits can corrupt the data in these functions and cause your user program to operate incorrectly, which can cause damage to equipment and injury to personnel.

Because both the clock memory and system memory are unreserved in M memory, instructions or communications can write to these locations and corrupt the data.

Avoid writing data to these locations to ensure the proper operation of these functions, and always implement an emergency stop circuit for your process or machine.

System memory configures a byte with bits that turn on (value = 1) for a specific event.

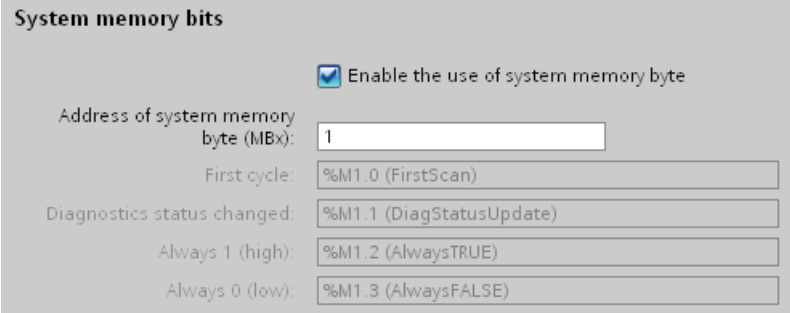


Table 4- 6 System memory

7	6	5	4	3	2	1	0
Reserved Value 0				Always off Value 0	Always on Value 1	Diagnostic status indicator • 1: Change • 0: No change	First scan indicator • 1: First scan after startup • 0: Not first scan

Clock memory configures a byte that cycles the individual bits on and off at fixed intervals. Each clock bit generates a square wave pulse on the corresponding M memory bit. These bits can be used as control bits, especially when combined with edge instructions, to trigger actions in the user code on a cyclic basis.

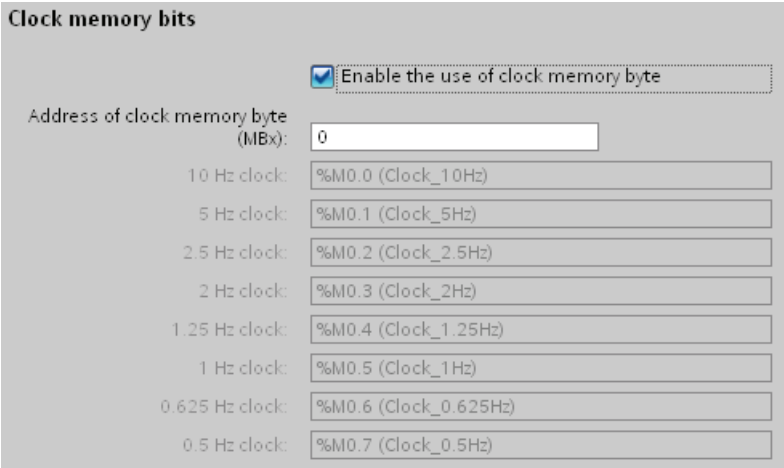


Table 4- 7 Clock memory

Bit number	7	6	5	4	3	2	1	0
Tag name								
Period (s)	2.0	1.6	1.0	0.8	0.5	0.4	0.2	0.1
Frequency (Hz)	0.5	0.625	1	1.25	2	2.5	5	10

Because clock memory runs asynchronously to the CPU cycle, the status of the clock memory can change several times during a long cycle.

4.1.7 Diagnostics buffer

The CPU supports a diagnostics buffer which contains an entry for each diagnostic event. Each entry includes a date and time the event occurred, an event category, and an event description. The entries are displayed in chronological order with the most recent event at the top. Up to 50 most recent events are available in this log. When the log is full, a new event replaces the oldest event in the log. When power is lost, the events are saved.

The following types of events are recorded in the diagnostics buffer:

- Each system diagnostic event; for example, CPU errors and module errors
- Each state change of the CPU (each power up, each transition to STOP, each transition to RUN)

To access the diagnostics buffer (Page 680), you must be online. From the "Online & diagnostics" view, locate the diagnostics buffer under "Diagnostics > Diagnostics buffer".

4.1.8 Time of day clock

The CPU supports a time-of-day clock. A super-capacitor supplies the energy required to keep the clock running during times when the CPU is powered down. The super-capacitor charges while the CPU has power. After the CPU has been powered up at least 24 hours, then the super-capacitor has sufficient charge to keep the clock running for typically 20 days.

STEP 7 sets the time-of-day clock to system time, which has a default value out of the box or following a factory rest. To utilize the time-of-day clock, you must set it. Timestamps such as those for diagnostic buffer entries, data log files, and data log entries are based on the system time. You set the time of day from the "Set time of day" function (Page 678) in the "Online & diagnostics" view of the online CPU. STEP 7 then calculates the system time from the time you set plus or minus the Windows operating system offset from UTC (Coordinated Universal Time). Setting the time of day to the current local time produces a system time of UTC if your Windows operating system settings for time zone and daylight savings time correspond to your locale.

STEP 7 includes instructions (Page 249) to read and write the system time (RD_SYS_T and WR_SYS_T), to read the local time (RD_LOC_T), and to set the time zone (SET_TIMEZONE). The RD_LOC_T instruction calculates local time using the time zone and daylight saving time offsets that you set in the "Time of day" configuration in the general properties of the CPU (Page 123). These settings enable you to set your time zone for local time, optionally enable daylight saving time, and specify the start and end dates and times for daylight saving time. You can also use the SET_TIMEZONE instructions to configure these settings.

4.1.9 Configuring the outputs on a RUN-to-STOP transition

You can configure the behavior of the digital and analog outputs when the CPU is in STOP mode. For any output of a CPU, SB or SM, you can set the outputs to either freeze the value or use a substitute value:

- Substituting a specified output value (default): You enter a substitute value for each output (channel) of that CPU, SB, or SM device.

  The default substitute value for digital output channels is OFF, and the default substitute value for analog output channels is 0.

- Freezing the outputs to remain in last state: The outputs retain their current value at the time of the transition from RUN to STOP. After power up, the outputs are set to the default substitute value.

You configure the behavior of the outputs in Device Configuration. Select the individual devices and use the "Properties" tab to configure the outputs for each device.

When the CPU changes from RUN to STOP, the CPU retains the process image and writes the appropriate values for both the digital and analog outputs, based upon the configuration.

4.2 Data storage, memory areas, I/O and addressing

4.2.1 Accessing the data of the S7-1200

STEP 7 facilitates symbolic programming. You create symbolic names or "tags" for the addresses of the data, whether as PLC tags relating to memory addresses and I/O points or as local variables used within a code block. To use these tags in your user program, simply enter the tag name for the instruction parameter.

For a better understanding of how the CPU structures and addresses the memory areas, the following paragraphs explain the "absolute" addressing that is referenced by the PLC tags. The CPU provides several options for storing data during the execution of the user program:

- Global memory: The CPU provides a variety of specialized memory areas, including inputs (I), outputs (Q) and bit memory (M). This memory is accessible by all code blocks without restriction
- PLC tag table: You can enter symbolic names in the STEP 7 PLC tag table for specific memory locations. These tags are global to the STEP 7 program and allow programming with names that are meaningful for your application.
- Data block (DB): You can include DBs in your user program to store data for the code blocks. The data stored persists when the execution of the associated code block comes to an end. A "global" DB stores data that can be used by all code blocks, while an instance DB stores data for a specific FB and is structured by the parameters for the FB.
- Temp memory: Whenever a code block is called, the operating system of the CPU allocates the temporary, or local, memory (L) to be used during the execution of the block. When the execution of the code block finishes, the CPU reallocates the local memory for the execution of other code blocks.

Each different memory location has a unique address. Your user program uses these addresses to access the information in the memory location. References to the input (I) or output (Q) memory areas, such as I0.3 or Q1.7, access the process image. To immediately access the physical input or output, append the reference with ":P" (such as I0.3:P, Q1.7:P, or "Stop:P").

Table 4- 8 Memory areas

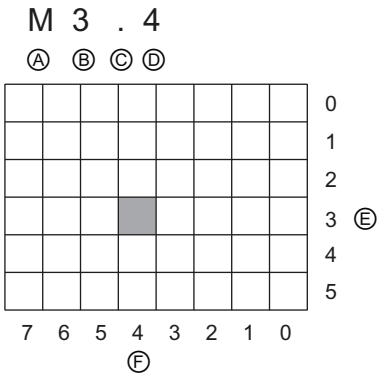
Memory area	Description	Force	Retentive
I Process image input	Copied from physical inputs at the beginning of the scan cycle	No	No
I_:P [1] (Physical input)	Immediate read of the physical input points on the CPU, SB, and SM	Yes	No
Q Process image output	Copied to physical outputs at the beginning of the scan cycle	No	No
Q_:P [1] (Physical output)	Immediate write to the physical output points on the CPU, SB, and SM	Yes	No
M Bit memory	Control and data memory	No	Yes (optional)
L Temp memory	Temporary data for a block local to that block	No	No
DB Data block	Data memory and also parameter memory for FBs	No	Yes (optional)

[1] To immediately access (read or write) the physical inputs and physical outputs, append a ":P" to the address or tag (such as I0.3:P, Q1.7:P, or "Stop:P").

Each different memory location has a unique address. Your user program uses these addresses to access the information in the memory location. The absolute address consists of the following elements:

- Memory area identifier (such as I, Q, or M)
- Size of the data to be accessed ("B' for Byte, "W" for Word, or "D" for DWord)
- Starting address of the data (such as byte 3 or word 3)

When accessing a bit in the address for a Boolean value, you do not enter a mnemonic for the size. You enter only the memory area, the byte location, and the bit location for the data (such as I0.0, Q0.1, or M3.4).



A	Memory area identifier	E	Bytes of the memory area
B	Byte address: byte 3	F	Bits of the selected byte
C	Separator ("byte.bit")		
D	Bit location of the byte (bit 4 of 8)		

In the example, the memory area and byte address (M = bit memory area, and 3 = Byte 3) are followed by a period (".") to separate the bit address (bit 4).

Accessing the data in the memory areas of the CPU

STEP 7 facilitates symbolic programming. Typically, tags are created either in PLC tags, a data block, or in the interface at the top of an OB, FC, or FB. These tags include a name, data type, offset, and comment. Additionally, in a data block, a start value can be specified. You can use these tags when programming by entering the tag name at the instruction parameter. Optionally you can enter the absolute operand (memory area, size and offset) at the instruction parameter. The examples in the following sections show how to enter absolute operands. The % character is inserted automatically in front of the absolute operand by the program editor. You can toggle the view in the program editor to one of these: symbolic, symbolic and absolute, or absolute.

I (process image input): The CPU samples the peripheral (physical) input points just prior to the cyclic OB execution of each scan cycle and writes these values to the input process image. You can access the input process image as bits, bytes, words, or double words. Both read and write access is permitted, but typically, process image inputs are only read.

Table 4- 9 Absolute addressing for I memory

Bit	I[byte address].[bit address]	I0.1
Byte, Word, or Double Word	I[size][starting byte address]	IB4, IW5, or ID12

By appending a ":P" to the address, you can immediately read the digital and analog inputs of the CPU, SB or SM. The difference between an access using I_:P instead of I is that the data comes directly from the points being accessed rather than from the input process image. This I_:P access is referred to as an "immediate read" access because the data is retrieved immediately from the source instead of from a copy that was made the last time the input process image was updated.

Because the physical input points receive their values directly from the field devices connected to these points, writing to these points is prohibited. That is, I_:P accesses are read-only, as opposed to I accesses which can be read or write.

I_:P accesses are also restricted to the size of inputs supported by a single CPU, SB, or SM, rounded up to the nearest byte. For example, if the inputs of a 2 DI / 2 DQ SB are configured to start at I4.0, then the input points can be accessed as I4.0:P and I4.1:P or as IB4:P. Accesses to I4.2:P through I4.7:P are not rejected, but make no sense since these points are not used. Accesses to IW4:P and ID4:P are prohibited since they exceed the byte offset associated with the SB.

Accesses using I_:P do not affect the corresponding value stored in the input process image.

Table 4- 10 Absolute addressing for I memory (immediate)

Bit	I[byte address].[bit address]:P	I0.1:P
Byte, Word, or Double word	I[size][starting byte address]:P	IB4:P, IW5:P, or ID12:P

Q (process image output): The CPU copies the values stored in the output process image to the physical output points. You can access the output process image in bits, bytes, words, or double words. Both read and write access is permitted for process image outputs.

Table 4- 11 Absolute addressing for Q memory

Bit	Q[byte address].[bit address]	Q1.1
Byte, Word, or Double word	Q[size][starting byte address]	QB5, QW10, QD40

By appending a ":P" to the address, you can immediately write to the physical digital and analog outputs of the CPU, SB or SM. The difference between an access using Q_:P instead of Q is that the data goes directly to the points being accessed in addition to the output process image (writes to both places). This Q_:P access is sometimes referred to as an "immediate write" access because the data is sent immediately to the target point; the target point does not have to wait for the next update from the output process image.

Because the physical output points directly control field devices that are connected to these points, reading from these points is prohibited. That is, Q_:P accesses are write-only, as opposed to Q accesses which can be read or write.

Q_:P accesses are also restricted to the size of outputs supported by a single CPU, SB, or SM, rounded up to the nearest byte. For example, if the outputs of a 2 DI / 2 DQ SB are configured to start at Q4.0, then the output points can be accessed as Q4.0:P and Q4.1:P or as QB4:P. Accesses to Q4.2:P through Q4.7:P are not rejected, but make no sense since these points are not used. Accesses to QW4:P and QD4:P are prohibited since they exceed the byte offset associated with the SB.

Accesses using Q_:P affect both the physical output as well as the corresponding value stored in the output process image.

Table 4- 12 Absolute addressing for Q memory (immediate)

Bit	Q[byte address].[bit address]:P	Q1.1:P
Byte, Word, or Double word	Q[size][starting byte address]:P	QB5:P, QW10:P or QD40:P

M (bit memory area): Use the bit memory area (M memory) for both control relays and data to store the intermediate status of an operation or other control information. You can access the bit memory area in bits, bytes, words, or double words. Both read and write access is permitted for M memory.

Table 4- 13 Absolute addressing for M memory

Bit	M[byte address].[bit address]	M26.7
Byte, Word, or Double Word	M[size][starting byte address]	MB20, MW30, MD50

Temp (temporary memory): The CPU allocates the temp memory on an as-needed basis. The CPU allocates the temp memory for the code block at the time when the code block is started (for an OB) or is called (for an FC or FB). The allocation of temp memory for a code block might reuse the same temp memory locations previously used by a different OB, FC or FB. The CPU does not initialize the temp memory at the time of allocation and therefore the temp memory might contain any value.

Temp memory is similar to M memory with one major exception: M memory has a "global" scope, and temp memory has a "local" scope:

- M memory: Any OB, FC, or FB can access the data in M memory, meaning that the data is available globally for all of the elements of the user program.
- Temp memory: Access to the data in temp memory is restricted to the OB, FC, or FB that created or declared the temp memory location. Temp memory locations remain local and are not shared by different code blocks, even when the code block calls another code block. For example: When an OB calls an FC, the FC cannot access the temp memory of the OB that called it.

The CPU provides temp (local) memory for each of the three OB priority groups:

- 16 Kbytes for startup and program cycle, including associated FBs and FCs
- 4 Kbytes for standard interrupt events including FBs and FCs
- 4 Kbytes for error interrupt events including FBs and FCs

You access temp memory by symbolic addressing only.

DB (data block): Use the DB memory for storing various types of data, including intermediate status of an operation or other control information parameters for FBs, and data structures required for many instructions such as timers and counters. You can access data block memory in bits, bytes, words, or double words. Both read and write access is permitted for read/write data blocks. Only read access is permitted for read-only data blocks.

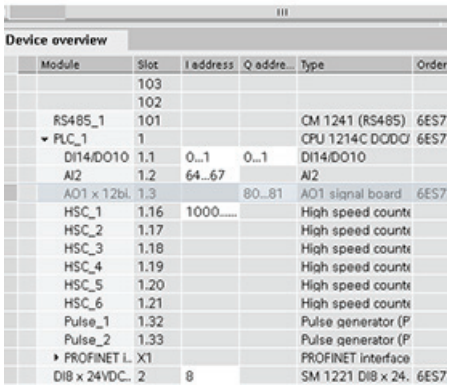
Table 4- 14 Absolute addressing for DB memory

Bit	DB[data block number].DBX[byte address].[bit address]	DB1.DBX2.3
Byte, Word, or Double Word	DB[data block number].DB [size][starting byte address]	DB1.DBB4, DB10.DBW2, DB20.DBD8

Note

When you specify an absolute address, STEP 7 precedes this address with a "%" character to indicate that it is an absolute address. While programming, you can enter an absolute address either with or without the "%" character (for example %I0.0 or I.0). If omitted, STEP 7 supplies the "%" character.

Configuring the I/O in the CPU and I/O modules



When you add a CPU and I/O modules to your configuration screen, I and Q addresses are automatically assigned. You can change the default addressing by selecting the address field in the configuration screen and typing new numbers.

- Digital inputs and outputs are assigned in groups of 8 points (1 byte), whether the module uses all the points or not.
- Analog inputs and outputs are assigned in groups of 2 points (4 bytes).

The figure shows an example of a CPU 1214C with two SMs and one SB. In this example, you could change the address of the DI8 module to 2 instead of 8. The tool will assist you by changing address ranges that are the wrong size or conflict with other addresses.

4.3 Processing of analog values

Analog signal modules provide input signals or expect output values that represent either a voltage range or a current range. These ranges are ±10V, ±5V, ±2.5V, or 0 - 20mA. The values returned by the modules are integer values where 0 to 27648 represents the rated range for current, and -27648 to 27648 for voltage. Anything outside the range represents either an overflow or underflow. See the tables for analog input representation (Page 770) and analog output representation (Page 771) for details.

In your control program, you probably need to use these values in engineering units, for example to represent a volume, temperature, weight or other quantitative value. To do this for an analog input, you must first normalize the analog value to a real (floating point) value from 0.0 to 1.0. Then you must scale it to the minimum and maximum values of the engineering units that it represents. For values that are in engineering units that you need to convert to an analog output value, you first normalize the value in engineering units to a value between 0.0 and 1.0, and then scale it between 0 and 27648 or -27648 to 27648, depending on the range of the analog module. STEP 7 provides the NORM_X and SCALE_X instructions (Page 219) for this purpose. You can also use the CALCULATE instruction (Page 198) to scale the analog values (Page 33).

4.4 Data types

Data types are used to specify both the size of a data element as well as how the data are to be interpreted. Each instruction parameter supports at least one data type, and some parameters support multiple data types. Hold the cursor over the parameter field of an instruction to see which data types are supported for a given parameter.

A formal parameter is the identifier on an instruction that marks the location of data to be used by that instruction (example: the IN1 input of an ADD instruction). An actual parameter is the memory location (preceded by a "%" character) or constant containing the data to be used by the instruction (example %MD400 "Number_of_Widgets"). The data type of the actual parameter specified by you must match one of the supported data types of the formal parameter specified by the instruction.

When specifying an actual parameter, you must specify either a tag (symbol) or an absolute (direct) memory address. Tags associate a symbolic name (tag name) with a data type, memory area, memory offset, and comment, and can be created either in the PLC tags editor or in the Interface editor for a block (OB, FC, FB and DB). If you enter an absolute address that has no associated tag, you must use an appropriate size that matches a supported data type, and a default tag will be created upon entry.

All data types except String are available in the PLC tags editor and the block Interface editors. String is available only in the block Interface editors. You can also enter a constant value for many of the input parameters.

- Bit and Bit sequences (Page 94): Bool (Boolean or bit value), Byte (8-bit byte value), Word (16-bit value), DWord (32-bit double-word value)
- Integer (Page 95)
  - USInt (unsigned 8-bit integer), SInt (signed 8-bit integer),
  - UInt (unsigned 16-bit integer), Int (signed 16-bit integer)
  - UDInt (unsigned 32-bit integer), DInt (signed 32-bit integer)
- Floating-point Real (Page 95): Real (32-bit Real or floating-point value), LReal (64-bit Real or floating-point value)
- Time and Date (Page 96): Time (32-bit IEC time value), Date (16-bit date value), TOD (32-bit time-off-day value), DT (64-bit date-and-time value)
- Character and String (Page 97): Char (8-bit single character), String (variable-length string of up to 254 characters)
- Array (Page 99)
- Data structure (Page 100): Struct
- PLC Data type (Page 100)
- Pointers (Page 101): Pointer, Any, Variant

Although not available as data types, the following BCD numeric format is supported by the conversion instructions.

Table 4- 15 Size and range of the BCD format

Format	Size (bits)	Numeric Range	Constant Entry Examples
BCD16	16	-999 to 999	123, -123
BCD32	32	-9999999 to 9999999	1234567, -1234567

4.4.1 Bool, Byte, Word, and DWord data types

Table 4- 16 Bit and bit sequence data types

Data type	Bit size	Number type	Number range	Constant examples	Address examples
Bool	1	Boolean	FALSE or TRUE	TRUE, 1,	I1.0 Q0.1 M50.7 DB1.DBX2.3 Tag_name
		Binary	0 or 1	0, 2#0	
		Octal	8#0 or 8#1	8#1	
		Hexadecimal	16#0 or 16#1	16#1	
Byte	8	Binary	2#0 to 2#11111111	2#00001111	IB2 MB10 DB1.DBB4 Tag_name
		Unsigned integer	0 to 255	15	
		Octal	8#0 to 8#377	8#17	
		Hexadecimal	B#16#0 to B#16#FF	B#16#F, 16#F	
Word	16	Binary	2#0 to 2#1111111111111111	2#1111000011110000	MW10 DB1.DBW2 Tag_name
		Unsigned integer	0 to 65535	61680	
		Octal	8#0 to 8#177777	8#170360	
		Hexadecimal	W#16#0 to W#16#FFFF, 16#0 to 16#FFFF	W#16#F0F0, 16#F0F0	
DWord	32	Binary	2#0 to 2#11111111111111111111111111111111	2#111100001111111100001111	MD10 DB1.DBD8 Tag_name
		Unsigned integer	0 to 4294967295	15793935	
		Octal	8#0 to 8#37777777777	8#74177417	
		Hexadecimal	DW#16#0000_0000 to DW#16#FFFF_FFFF, 16#0000_0000 to 16#FFFF_FFFF	DW#16#F0FF0F, 16#F0FF0F	

4.4.2 Integer data types

Table 4- 17 Integer data types (U = unsigned, S = short, D= double)

Data type	Bit size	Number Range	Constant examples	Address examples
USInt	8	0 to 255	78, 2#01001110	MB0, DB1.DBB4, Tag_name
SInt	8	-128 to 127	+50, 16#50	
UInt	16	0 to 65,535	65295, 0	MW2, DB1.DBW2, Tag_name
Int	16	-32,768 to 32,767	30000, +30000	
UDInt	32	0 to 4,294,967,295	4042322160	MD6, DB1.DBD8, Tag_name
DInt	32	-2,147,483,648 to 2,147,483,647	-2131754992	

4.4.3 Floating-point real data types

Real (or floating-point) numbers are represented as 32-bit single-precision numbers (Real), or 64-bit double-precision numbers (LReal) as described in the ANSI/IEEE 754-1985 standard. Single-precision floating-point numbers are accurate up to 6 significant digits and double-precision floating point numbers are accurate up to 15 significant digits. You can specify a maximum of 6 significant digits (Real) or 15 (LReal) when entering a floating-point constant to maintain precision.

Table 4- 18 Floating-point real data types (L=Long)

Data type	Bit size	Number range	Constant Examples	Address examples
Real	32	-3.402823e+38 to -1.175 495e-38, ±0, +1.175 495e-38 to +3.402823e+38	123.456, -3.4, 1.0e-5	MD100, DB1.DBD8, Tag_name
LReal	64	-1.7976931348623158e+308 to -2.2250738585072014e-308, ±0, +2.2250738585072014e-308 to +1.7976931348623158e+308	12345.123456789e40, 1.2E+40	DB_name.var_name Rules: • No direct addressing support • Can be assigned in an OB, FB, or FC block interface table

Calculations that involve a long series of values including very large and very small numbers can produce inaccurate results. This can occur if the numbers differ by 10 to the power of x, where x > 6 (Real), or 15 (LReal). For example (Real): 100 000 000 + 1 = 100 000 000.

4.4.4 Time and Date data types

Table 4- 19 Time and date data types

Data type	Size	Range	Constant Entry Examples
Time	32 bits	T#-24d_20h_31m_23s_648ms to T#24d_20h_31m_23s_647ms Stored as: -2,147,483,648 ms to +2,147,483,647 ms	T#5m_30s T#1d_2h_15m_30s_45ms TIME#10d20h30m20s630ms 500h10000ms 10d20h30m20s630ms
Date	16 bits	D#1990-1-1 to D#2168-12-31	D#2009-12-31 DATE#2009-12-31 2009-12-31
Time_of_Day	32 bits	TOD#0:0:0.0 to TOD#23:59:59.999	TOD#10:20:30.400 TIME_OF_DAY#10:20:30.400 23:10:1
DTL (Date and Time Long)	12 bytes	Min.: DTL#1970-01-01-00:00:00.0 Max.: DTL#2554-12-31-23:59:59.999 999 999	DTL#2008-12-16-20:30:20.250

Time

TIME data is stored as a signed double integer interpreted as milliseconds. The editor format can use information for day (d), hours (h), minutes (m), seconds (s) and milliseconds (ms).

It is not necessary to specify all units of time. For example T#5h10s and 500h are valid.

The combined value of all specified unit values cannot exceed the upper or lower limits in milliseconds for the Time data type (-2,147,483,648 ms to +2,147,483,647 ms).

Date

DATE data is stored as an unsigned integer value which is interpreted as the number of days added to the base date 01/01/1990, to obtain the specified date. The editor format must specify a year, month and day.

TOD

TOD (TIME_OF_DAY) data is stored as an unsigned double integer which is interpreted as the number of milliseconds since midnight for the specified time of day (Midnight = 0 ms). The hour (24hr/day), minute, and second must be specified. The fractional second specification is optional.

DTL

DTL (Date and Time Long) data type uses a12 byte structure that saves information on date and time. You can define DTL data in either the Temp memory of a block or in a DB. A value for all components must be entered in the "Start value" column of the DB editor.

Table 4- 20 Size and range for DTL

Length (bytes)	Format	Value range	Example of value input
12	Clock and calendar Year-Month-Day:Hour:Minute: Second.Nanoseconds	Min.: DTL#1970-01-01-00:00:00.0 Max.: DTL#2554-12-31-23:59:59.999 999 999	DTL#2008-12-16-20:30:20.250

Each component of the DTL contains a different data type and range of values. The data type of a specified value must match the data type of the corresponding components.

Table 4- 21 Elements of the DTL structure

Byte	Component	Data type	Value range
0	Year	UINT	1970 to 2554
1			
2	Month	USINT	1 to 12
3	Day	USINT	1 to 31
4	Weekday [1]	USINT	1(Sunday) to 7(Saturday) [1]
5	Hour	USINT	0 to 23
6	Minute	USINT	0 to 59
7	Second	USINT	0 to 59
8	Nanoseconds	UDINT	0 to 999 999 999
9			
10			
11			

[1] The weekday is not considered in the value entry.

4.4.5 Character and String data types

Table 4- 22 Character and String data types

Data type	Size	Range	Constant Entry Examples
Char	8 bits	ASCII character codes: 16#00 to 16#FF	'A', 't', '@'
String	n+ 2 bytes	n = (0 to 254 character bytes)	'ABC'

Char

Char data occupies one byte in memory and stores a single character coded in ASCII format. The editor syntax uses a single quote character before and after the ASCII character. Visible characters and control characters can be used. A table of valid control characters is shown in the description of the String data type.

String

The CPU supports the String data type for storing a sequence of single-byte characters. The String data type contains a total character count (number of characters in the string) and the current character count. The String type provides up to 256 bytes for storing the maximum total character count (1 byte), the current character count (1 byte), and up to 254 characters, with each character stored in 1 byte.

You can use literal strings (constants) for instruction parameters of type IN using single quotes. For example, ‘ABC’ is a three-character string that could be used as input for parameter IN of the S_CONV instruction. You can also create string variables by selecting data type "String" in the block interface editors for OB, FC, FB, and DB. You cannot create a string in the PLC tags editor.

You can specify the maximum string size in bytes by entering square brackets after the keyword "String" (once the data type "String" is selected from a data type drop-list). For example, "MyString String[10]" would specify a 10-byte maximum size for MyString. If you do not include the square brackets with a maximum size, then 254 is assumed.

The following example defines a String with maximum character count of 10 and current character count of 3. This means the String currently contains 3 one-byte characters, but could be expanded to contain up to 10 one-byte characters.

Table 4- 23 Example of a String data type

Total Character Count	Current Character Count	Character 1	Character 2	Character 3	...	Character 10
10	3	'C' (16#43)	'A' (16#41)	'T' (16#54)	...	-
Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	...	Byte 11

ASCII control characters can be used in Char and String data. The following table shows examples of control character syntax.

Table 4- 24 Valid ASCII control characters

Control characters	ASCII Hex value	Control function	Examples
$L or $l	0A	Line feed	'$LText', '$0AText'
$N or $n	0A and 0D	Line break The new line shows two characters in the string.	'$NText', '$0A$0DText'
$P or $p	0C	Form feed	'$PText', '$0CText'
$R or $r	0D	Carriage return (CR)	'$RText','$0DText'
$T or $t	09	Tab	'$TText', '$09Text'

Control characters	ASCII Hex value	Control function	Examples
$$	24	Dollar sign	'100$$', '100$24'
$'	27	Single quote	'$'Text$'','$27Text$27'

4.4.6 Array data type

Arrays

You can create an array that contains multiple elements of the same data type. Arrays can be created in the block interface editors for OB, FC, FB, and DB. You cannot create an array in the PLC tags editor.

To create an array from the block interface editor, name the array and choose data type "Array [lo .. hi] of type", then edit "lo", "hi", and "type" as follows:

- lo - the starting (lowest) index for your array
- hi - the ending (highest) index for your array
- type - one of the data types, such as BOOL, SINT, UDINT

Table 4- 25 ARRAY data type rules

<table>
<tr><th>Data Type</th><th colspan="3">Array syntax</th></tr>
<tr><td rowspan="3">ARRAY</td><td colspan="3">Name [index1_min..index1_max, index2_min..index2_max] of <data type>
• All array elements must be the same data type.
• The index can be negative, but the lower limit must be less than or equal to the upper limit.
• Arrays can have one to six dimensions.
• Multi-dimensional index min..max declarations are separated by comma characters.
• Nested arrays, or arrays of arrays, are not allowed.
• The memory size of an array = (size of one element * total number of elements in array)</td></tr>
<tr><td>Array index</td><td>Valid index data types</td><td>Array index rules</td></tr>
<tr><td>Constant or variable</td><td>USInt, SInt, UInt, Int, UDInt, DInt</td><td>• Value limits: -32768 to +32767
• Valid: Mixed constants and variables
• Valid: Constant expressions
• Not valid: Variable expressions</td></tr>
</table>

Example: array declarations	ARRAY[1..20] of REAL	One dimension, 20 elements
	ARRAY[-5..5] of INT	One dimension, 11 elements
	ARRAY[1..2, 3..4] of CHAR	Two dimensions, 4 elements
Example: array addresses	ARRAY1[0]	ARRAY1 element 0
	ARRAY2[1,2]	ARRAY2 element [1,2]
	ARRAY3[i,j]	If i =3 and j=4, then ARRAY3 element [3, 4] is addressed

4.4.7 Data structure data type

You can use the data type "Struct" to define a structure of data consisting of other data types. The struct data type can be used to handle a group of related process data as a single data unit. A Struct data type is named and the internal data structure declared in the data block editor or a block interface editor.

Arrays and structures can also be assembled into a larger structure. A structure can be nested up to eight levels deep. For example, you can create a structure of structures that contain arrays.

A Struct variable begins at an even-byte address and uses the memory to the next word boundary.

4.4.8 PLC data type

The PLC data type editor lets you define data structures that you can use multiple times in your program. You create a PLC data type by opening the "PLC data types" branch of the project tree and double-clicking the "Add new data type" item. On the newly created PLC data type item, use two single-clicks to rename the default name and double-click to open the PLC data type editor.

You create a custom PLC data type structure using the same editing methods that are used in the data block editor. Add new rows for any data types that are necessary to create the data structure that you want.

If a new PLC data type is created, then the new PLC type name will appear in the data type selector drop drop-lists in the DB editor and code block interface editor.

Potential uses of PLC data types:

- PLC data types can be used directly as a data type in a code block interface or in data blocks.
- PLC data types can be used as a template for the creation of multiple global data blocks that use the same data structure.

For example, a PLC data type could be a recipe for mixing colors. You can then assign this PLC data type to multiple data blocks. Each data block can then have the variables adjusted to create a specific color.