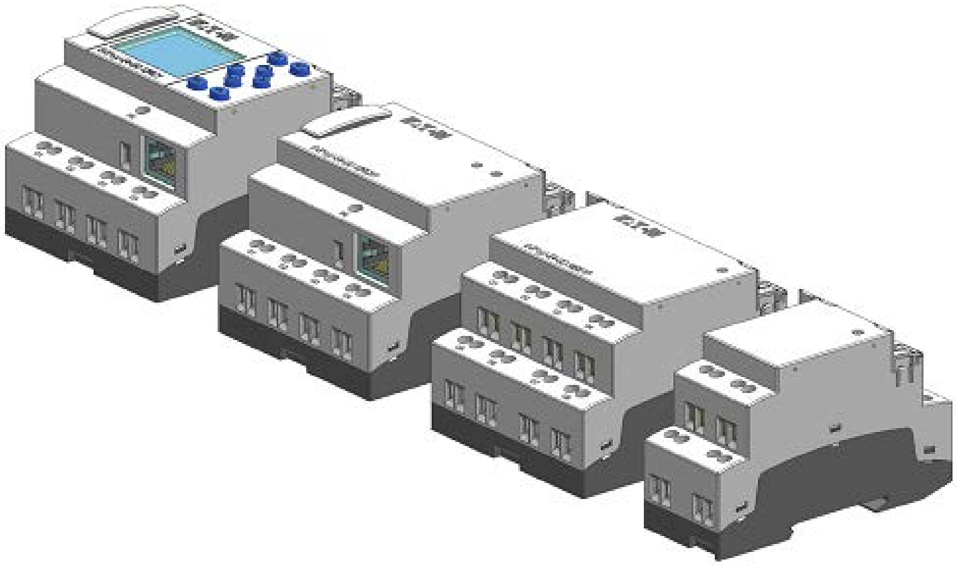
# Manual

10/19 MN050009 EN

## easyE4



Powering Business Worldwide

## Company information



**Break-Down Service**

Please contact your local office:
http://www.eaton.eu/aftersales
or the After Sales Service
Hotline After Sales Service:
+49 (0) 180 5 223822 (de,en)
AfterSalesEGBonn@eaton.com
automation@eaton.com

**Original Operating Instructions**

is the German-language edition of this document

Publication date
10/19 MN050009 EN 3.3 Edition, Build 011


Author: Electrical Sector, Business Unit MOC
Editor:
Eaton Industries GmbH, Hein-Moeller-Straße 7-11, D-53115 Bonn

**DANGER!**

Dangerous Electrical Voltage!

## Before starting with the installation

- Installation requires qualified electrician
- Disconnect the power supply of the device.
- Secure against retriggering
- Verify isolation from the supply
- Ground and short-circuit
- Cover or enclose any neighboring live parts.
- Follow the engineering instructions (IL) of the device concerned.
- Only suitably qualified personnel in accordance with EN 50110-1/-2 (VDE 0105 part 100) may work on this device/system.
- Before installation and before touching the device ensure that you are free of electrostatic charge.
- The functional earth (FE) must be connected to the protective earth (PE) or to the equipotential bonding. The system installer is responsible for implementing this connection.
- Connecting cables and signal lines should be installed so that inductive or capacitive interference does not impair the automation functions.
- Install automation devices and related operating elements in such a way that they are well protected against unintentional operation.
- Suitable safety hardware and software measures should be implemented for the I/O interface so that a line or wire breakage on the signal side does not result in undefined states in the automation devices.
- Deviations of the mains voltage from the nominal value must not exceed the tolerance limits given in the specifications, otherwise this may result in malfunction and hazardous states.
- Emergency-Stop devices complying with IEC/EN 60204-1 must be effective in all operating modes of the automation devices. Unlatching the emergency stop devices must not result in an automatic restart.
- Built-in devices for enclosures or cabinets must only be run and operated in an installed state; desktop devices and portable devices only when the housing is closed.
- Measures should be taken to ensure the proper restarting of programs interrupted after a voltage dip or outage. This should not result in dangerous operating states even for a short time. If necessary, emergency stop devices should be implemented.
- Wherever faults in the automation system may cause damage to persons or property, external measures must be implemented to ensure a safe operating state in the event of a fault or malfunction (for example, by means of separate limit switches, mechanical interlocks, etc.).

## Table of Contents


| | | |
|---|---|---|
| | **easyE4 Manual** | **1** |
| | Company information | 2 |
| | Before starting with the installation | 3 |
| | Table of Contents | 1 |
| 0.1 | About this manual | 13 |
| 0.1.1 | List of revisions | 13 |
| 0.1.2 | Target group | 14 |
| 0.1.3 | Legal disclaimer | 15 |
| 0.1.4 | Short designations | 16 |
| 0.1.5 | Writing conventions | 17 |
| 0.1.5.1 | Warning labels | 17 |
| 0.1.5.2 | Documents with additional information | 18 |
| **1.** | **easyE4 control relays description** | **19** |
| 1.1 | Use as intended | 19 |
| 1.2 | Function | 20 |
| 1.2.1 | Features | 20 |
| 1.3 | Device models - versions and part nos. | 22 |
| 1.3.1 | Base device versions | 22 |
| 1.3.2 | Expansion versions | 24 |
| 1.3.2.1 | Overview of available easyE4 devices | 26 |
| 1.4 | What the different parts of the part number mean | 28 |
| 1.5 | Accessory devices | 29 |
| 1.6 | Nameplate | 30 |
| 1.7 | Support | 31 |
| 1.8 | Programming software easySoft 7 | 32 |
| 1.9 | Safety regulations | 33 |
| 1.9.1 | Basics | 33 |
| 1.9.2 | Mandatory requirements, personnel requirements | 33 |
| 1.9.2.1 | Occupational safety | 33 |
| 1.9.2.2 | Personnel qualifications | 33 |
| 1.9.2.3 | Device documentation | 34 |

| | | |
|---|---|---|
| 1.9.2.4 | Installation, maintenance, and disposal | 34 |
| 1.9.2.5 | Prerequisites for proper operation | 35 |
| 1.9.3 | Device-specific hazards | 36 |
| 1.10 | Engineering | 39 |
| 1.10.1 | Length of input cables | 39 |
| 1.10.2 | Analog Signals | 40 |
| 1.10.3 | Notes on connecting EASY-E4-AC-... devices | 41 |
| **2.** | **Installation** | **43** |
| 2.1 | Prerequisites for the location of use | 44 |
| 2.1.1 | Installation position | 44 |
| 2.1.1.1 | Temperatures | 44 |
| 2.1.1.2 | Aeration and de-aeration | 45 |
| 2.2 | Unpacking and checking the equipment supplied | 46 |
| 2.3 | Mounting | 47 |
| 2.3.1 | Mounting easyE4 control relays | 47 |
| 2.3.1.1 | Installation on mounting rail | 50 |
| 2.3.1.2 | Screw mounting | 52 |
| 2.3.1.3 | Dismounting of a device | 53 |
| 2.4 | Connection terminals | 54 |
| 2.4.1 | Screw terminal connection | 54 |
| 2.4.2 | Push in connection | 55 |
| 2.4.3 | Connecting the power supply | 56 |
| 2.4.3.1 | Special notes on connecting EASY-E4-AC-... devices | 57 |
| 2.4.4 | Connect digital inputs | 58 |
| 2.4.4.1 | Connect digital counter inputs | 60 |
| 2.4.5 | Connecting analog inputs | 61 |
| 2.4.6 | Connecting relay outputs | 62 |
| 2.4.7 | Connecting transistor outputs | 63 |
| 2.4.7.1 | Transistor output behavior in the event of a short circuit/overload | 64 |
| 2.4.7.2 | Connecting outputs in parallel | 64 |
| 2.4.8 | Analog I/O expansion device | 65 |
| 2.4.9 | Analog input expansion with temperature measuring | 67 |
| 2.4.10 | Terminal configurations for individual devices | 70 |

| | | |
|---|---|---|
| 2.5 | External connections | 72 |
| 2.5.1 | External connection layouts | 72 |
| 2.5.2 | Memory card | 73 |
| 2.5.3 | Ethernet | 75 |
| 2.5.3.1 | Connecting the Ethernet cable | 76 |
| 2.6 | Programming software license | 78 |
| 2.6.1 | System requirements | 79 |
| 2.6.2 | Licensing | 79 |
| 2.6.2.1 | Getting a license key | 80 |
| 2.6.3 | Adding a license key later on | 81 |
| 2.6.4 | Software updates and hardware changes | 82 |
| 2.6.5 | Installation instructions | 83 |
| **3.** | **Commissioning** | **87** |
| 3.1 | Initial commissioning | 87 |
| 3.2 | Daily operation | 87 |
| 3.3 | Switch on | 88 |
| 3.3.1 | Startup behavior of easyE4 control relays with LED indicators | 88 |
| 3.3.2 | Startup behavior of easyE4 control relays with a display and keypad | 90 |
| 3.3.2.1 | Changing the menu language | 91 |
| 3.3.3 | Startup behavior of base devices with connected expansion devices | 93 |
| 3.3.4 | Status display on easyE4 control relays with display and keypad | 94 |
| 3.3.5 | Commissioning the Ethernet network | 96 |
| 3.3.5.1 | Network operation | 96 |
| 3.3.6 | Remote operation | 97 |
| 3.4 | Overview of switch-on behavior | 98 |
| 3.5 | Establishing an Ethernet connection and transferring a program | 100 |
| 3.6 | Automatic booting from the card | 106 |
| 3.6.1 | Preparing the card in the PC for booting with easySoft 7 | 106 |
| 3.6.2 | Preparing the card in the easyE4 device for booting with easySoft 7 | 110 |
| 3.6.3 | Preparing the card for booting on the easyE4 device itself | 114 |
| 3.7 | Reset – resetting the device to factory settings | 116 |

3.8 Updating firmware 117
3.9 microSDMemory card 124
3.9.1 microSD Ejecting the memory card 124
3.10 Setting a splash screen for the EASY-E4-...-12...C1(P) display 126

**4. Operation 127**
4.1 Base device with display and buttons 127
4.1.1 LCD Display 127
4.1.1.1 Display color backlight 128
4.1.2 Keyboard 128
4.1.3 Selecting menus and entering values 129
4.1.4 Cursor display 130
4.1.5 Entering of values 130
4.2 Operating modes of the easyE4 131
4.2.1 RUN mode 131
4.2.2 STOP mode 131
4.3 Operation of the menu selection and value entry 133
4.3.1 How to navigate the device menus 133
4.3.2 Operating principle in the circuit diagram and function block editor 133
4.3.3 Selecting a device menu 134
4.4 Overview of the menus on the device 135
4.4.1 Main menu 135
4.4.2 STOP RUN operating mode menu 135
4.4.3 Menu Parameter 136
4.4.4 Set clock menu 137
4.4.5 Menu Card 138
4.4.6 MenuInformation 139
4.4.7 System options menu 140
4.4.8 Program menu 142
4.5 Your first EDP program 144
4.5.1 Draw a wiring diagram 146
4.5.2 Testing the circuit diagram 149
4.5.3 Control options in RUN mode 150
4.5.4 Delete Program 152

4.6 Transfer program to the easyE4 device 153
4.6.1 Transfer with a microSD memory card 153
4.6.2 Establish Ethernet connection 158

**5. Programming on the device 161**
5.1 Program 161
5.2 Circuit diagram display 161
5.3 Circuit diagram elements 163
5.3.1 Function blocks 163
5.3.2 Relays 163
5.3.3 Contacts 164
5.3.4 Coils 165
5.4 Working with contacts and coils 170
5.4.1 Entering and modifying contacts 171
5.4.2 Changing an N/O contact to an N/C contact 172
5.4.3 Entering and modifying coils 173
5.4.4 Deleting contacts and coils 174
5.4.5 Creating and modifying connections 175
5.4.6 Deleting connections 176
5.4.7 Adding a rung 176
5.4.8 Deleting a rung 176
5.4.9 Got to a rung 177
5.4.10 Saving the circuit diagram 177
5.4.11 Exiting the circuit diagram without saving 178
5.4.12 Searching for contacts and coils 178
5.4.13 Switching with the Cursor Buttons 179
5.4.14 Checking the circuit diagram 180
5.4.15 Jumps 181
5.4.16 Wiring NET operands in the circuit diagram 183
5.5 Transferring programs from and to the microSD memory card 187
5.5.1 Configuration on base devices with a display 188
5.5.1.1 PROGRAM submenu 189
5.6 Working with function blocks 191
5.6.1 Adding function blocks to the circuit diagram for the first time 191

5.6.2 Function block list 193
5.6.3 Configuring parameters in the function block editor 194
5.6.4 PARAMETERS menu 197
5.6.5 Deleting function blocks 197
5.7 Using operands in a program 200

**6. Function blocks 216**
6.1 Manufacturer function blocks 218
6.1.1 Timer modules 218
6.1.1.1 HW - 7-day time switch (Hour Week) 218
6.1.1.2 HY - Year time switch (Hora Year) 228
6.1.1.3 OT - Operating hours counter 238
6.1.1.4 RC - Real-time clock 242
6.1.1.5 T - Timing relay 246
6.1.1.6 YT - Year time switch (Year Table) 261
6.1.1.7 WT - Weekly timer (WeekTable) 268
6.1.1.8 AC - Astronomic clock 272
6.1.2 Counter Function Blocks 281
6.1.2.1 C - Counter relay 281
6.1.2.2 CF - Frequency counter 287
6.1.2.3 CH - High-speed counter 293
6.1.2.4 CI - Incremental Counter 299
6.1.3 Arithmetic and analog function blocks 306
6.1.3.1 A - Analog value comparator 306
6.1.3.2 AR - Arithmetic 312
6.1.3.3 AV - Average 317
6.1.3.4 CP – Comparator 326
6.1.3.5 LS - Value scaling 330
6.1.3.6 MM - Min-/Max function 335
6.1.3.7 PM - Performance map 339
6.1.3.8 PW - Pulse width modulation 345
6.1.4 Open-loop and closed-loop function blocks 351
6.1.4.1 DC - PID controller 351
6.1.4.2 FT - PT1-Signal smoothing filter 358

| | | |
|---|---|---|
| 6.1.4.3 | PO - Pulse output | 364 |
| 6.1.4.4 | TC - Three step controller | 379 |
| 6.1.4.5 | VC - Value limitation | 384 |
| 6.1.5 | Data and register function blocks | 388 |
| 6.1.5.1 | BC - Block compare | 388 |
| 6.1.5.2 | BT - Block transfer | 395 |
| 6.1.5.3 | DB - Data function block | 401 |
| 6.1.5.4 | MX - Data multiplexer | 406 |
| 6.1.5.5 | RE - Recipe records | 410 |
| 6.1.5.6 | SR - Shift register | 415 |
| 6.1.5.7 | TB - Table function | 423 |
| 6.1.6 | NET Function Blocks | 428 |
| 6.1.6.1 | GT - Get values from NET | 428 |
| 6.1.6.2 | PT - Put values to NET | 432 |
| 6.1.6.3 | SC - Synchronizing clock via NET | 436 |
| 6.1.7 | Other function blocks | 440 |
| 6.1.7.1 | AL - Alarm function block | 440 |
| 6.1.7.2 | BV - Boolean operation | 444 |
| 6.1.7.3 | D - Text display | 448 |
| 6.1.7.4 | D - Text display editor | 458 |
| 6.1.7.5 | DL - Data logger | 475 |
| 6.1.7.6 | JC - Conditional jump | 486 |
| 6.1.7.7 | LB - Jump label | 491 |
| 6.1.7.8 | MR - MasterReset | 493 |
| 6.1.7.9 | NC - Numerical converter | 497 |
| 6.1.7.10 | ST - Set cycle time | 503 |
| 6.2 | interrupt function blocks | 506 |
| 6.2.1 | IC - Counter-controlled interrupt | 506 |
| 6.2.1.1 | General | 506 |
| 6.2.1.2 | Operating principle | 507 |
| 6.2.1.3 | The function block and its parameters | 508 |
| 6.2.1.4 | Other | 511 |
| 6.2.2 | IE - Edge-controlled interrupt | 519 |
| 6.2.2.1 | General | 519 |

| | | |
|---|---|---|
| 6.2.2.2 | Operating principle | 520 |
| 6.2.2.3 | The function block and its parameters | 521 |
| 6.2.2.4 | Other | 523 |
| 6.2.3 | IT - Time-controlled interrupt function block | 525 |
| 6.2.3.1 | General | 525 |
| 6.2.3.2 | Operating principle | 525 |
| 6.2.3.3 | The function block and its parameters | 527 |
| 6.2.3.4 | Other | 529 |
| 6.3 | UF - User function block | 533 |
| 6.3.1 | General | 533 |
| 6.3.1.1 | General information on user function blocks | 534 |
| 6.3.2 | Creating a user function block | 534 |
| 6.3.3 | Configuring a user function block | 536 |
| 6.3.4 | Programming a user function block | 541 |
| 6.3.4.1 | Programming view tabs | 542 |
| 6.3.5 | Calling a user function block in the main program | 543 |
| 6.3.5.1 | User function blocks in an ST main program | 547 |
| 6.3.6 | Saving a user function block | 548 |
| 6.3.7 | Exporting a user function block | 550 |
| 6.3.7.1 | Plausibility check | 551 |
| 6.3.8 | Importing a user function block | 552 |
| 6.3.9 | Replacing a user function block | 553 |
| 6.3.10 | Deleting a user function block | 553 |
| 6.3.11 | Comparing user function blocks | 554 |
| 6.3.12 | Printing a user function block | 556 |
| 6.4 | Timing and counter relay example | 557 |
| **7.** | **System settings** | **561** |
| 7.1 | System options - Base device with display and buttons | 562 |
| 7.2 | Display | 563 |
| 7.3 | Device ID | 564 |
| 7.4 | Splash screen | 565 |
| 7.5 | NET | 566 |
| 7.6 | Ethernet | 568 |

7.7 Update 569
7.8 Switch languages 570
7.8.1 Selecting a menu language on a base device with a display 570
7.8.2 Setting the menu language in the easySoft 7 570
7.9 Setting the startup behavior 571
7.9.1 Enabling / disabling the RUN START option 572
7.9.1.1 Configuration on base devices with a display 572
7.9.2 Enabling / disabling the CARD START option 572
7.9.2.1 Configuration on base devices with a display 573
7.9.2.2 Configuring the easySoft 7 573
7.10 Debounce 574
7.10.1 Configuring input debouncing on a base device with a display 574
7.10.2 Configuring input debouncing in easySoft 7 574
7.11 Download comments 575
7.12 P buttons 576
7.12.1 Configuring the P buttons on a base device with a display 576
7.12.2 Configuring the P buttons in easySoft 7 576
7.13 Define program name 577
7.14 Retention function 578
7.14.1 Retention in the easySoft 7 579
7.15 Security – password protection 581
7.15.1 Configuring the password on a base device with a display 581
7.15.1.1 What happens if you forget your password or enter the wrong password? 583
7.15.2 Configuring the password in easySoft 7 584
7.16 Configuring the microSD card and device ID 586
7.17 Time and Date setting 587
7.17.1 Time and date on a base device with a display 587
7.17.2 Setting time and date in the easySoft 7 590

**8. easyE4 Inside 593**
8.1 Program execution 594
8.2 Transfering an existing circuit diagram 597
8.3 Device information 598
8.4 NET network 599

| | | |
|---|---|---|
| 8.5 | Operating states easyE4 | 602 |
| 8.6 | Device easyE4 time responses | 603 |
| 8.6.1 | Time behavior of the inputs and outputs | 603 |
| 8.6.2 | Base device timing | 604 |
| 8.6.2.1 | Delay time for operation with DC power supply | 604 |
| 8.6.2.2 | Delay time for operation with AC power supply | 606 |
| 8.6.3 | Timing characteristics of expansion devices | 608 |
| 8.6.3.1 | Delay time for AC expansion devices | 608 |
| **9.** | **Operating system diagnostic messages** | **609** |
| 9.1 | Transistor outputs (overload / short-circuit) | 611 |
| 9.2 | Diagnostics buffer | 612 |
| 9.3 | LED status messages on the device | 613 |
| **10.** | **Connection to other devices** | **615** |
| 10.1 | Setting up a NET | 625 |
| 10.1.1 | Access on the NET | 626 |
| 10.1.2 | Communication via NET | 627 |
| 10.1.3 | NET settings | 629 |
| 10.2 | Modbus TCP | 632 |
| 10.2.1 | General | 632 |
| 10.2.2 | Programming communication with Modbus TCP | 633 |
| 10.2.2.1 | Read Coils 0x01: | 634 |
| 10.2.2.2 | Read Discrete Inputs 0x02: | 635 |
| 10.2.2.3 | Read Holding Registers 0x03: | 636 |
| 10.2.2.4 | Read Input Registers 0x04: | 637 |
| 10.2.2.5 | Write Single Registers 0x06: | 638 |
| 10.2.2.6 | Write Multiple Registers 0x10: | 638 |
| 10.2.3 | Modbus TCP error handling | 639 |
| 10.2.4 | Modbus map | 643 |
| 10.3 | Setting up a web server | 646 |
| 10.3.1 | Configuring the web server function in easySoft 7 | 648 |
| 10.4 | Web client | 652 |
| 10.5 | E-mail function | 670 |
| 10.6 | Convenient visualization for easyE4 | 680 |

**11. Faults** .......... **681**
11.1 Messages from the operating system .......... 682
11.2 Possible situations when creating programs .......... 683
11.3 Event .......... 684
11.4 Functionality of the NET faulty .......... 685

**12. Maintenance** .......... **687**
12.1 Cleaning and maintenance .......... 688
12.2 Repairs .......... 689
12.3 Storage, transport and disposal .......... 690
12.3.1 Storage and transport .......... 690
12.3.2 Disposal .......... 691

**Appendix** .......... **693**
A.1 Dimension and weight specifications .......... 694
A.2 Approvals and declarations .......... 698
A.3 Technical data .......... 699
A.3.1 Data sheets .......... 699
A.3.1.1 Base devices .......... 699
A.3.1.2 Expansions .......... 699
A.4 Required memory for function blocks .......... 705
A.5 Further information .......... 709
A.6 Sample Projects .......... 711
A.6.1 easyE4_Lauflicht_EDP.e70 application example .......... 711
Alphabetical index .......... 714
List of Figures .......... 726
Glossary .......... 737

# 0.1 About this manual

This manual contains all the information you will need in order to use the easyE4 safely and effectively.

The Manual easyE4 manual is considered an integral part of the devices and must always be readily available in the device's close proximity so that users have access to it.

As an integrated part, the easySoft 7 Help groups together the relevant sections in the easySoft 7.

This Manual describes all of the devices' lifecycle stages: transportation, installation, commissioning, operation, maintenance, storage, and disposal.
It assumes you have electrical engineering knowledge and skills.

Make sure to always use the latest documentation for your device.

Manual easyE4 MN050009_EN

The latest version of this documentation, as well as additional references, is available for download on the Internet.

http://www.eaton.eu/doc

Please send any comments, recommendations, or suggestions regarding this document to: automation@eaton.com

## 0.1.1 List of revisions

The following significant amendments have been introduced since previous issues:

| Publication date | Page | Keyword | New | Modification | Deleted |
|---|---|---|---|---|---|
| 11/2018 | | New edition | ✓ | | |
| 11/2018 V1.1 | A3<br>A5<br>24 | Real-time clock characteristic curve<br>Sample program<br>Cat No. MEMORY-SUD-A1 | | ✓ | |
| 1/2019 V1.2 | ff | Corrections | | | |
| 2/2019 | | Added models EASY-E4-AC-... and EASY-E4-DC-4PE1; added AC, AV, PM, and RE function blocks | ✓ | | |
| 4/2019 | | Webserver, E-Mail function, time response, micro SD card | | ✓ | |
| 10/2019 | ff | Device versions with push in terminals,EASY-E4-AC (cULus) | ✓ | ✓ | |

## 0.1 About this manual

### 0.1.2 Target group

This manual is intended for electricians and electrical engineers, as well as for the people who will be in charge of performing the electrical installation and people who will be using the control relays as operating and monitoring devices or as integrated operating and control devices in their own applications.

This manual is intended for people who:

- Want to use an easyE4 control relay
- Develop an application with easySoft 7.
- Want to test or commission a developed application
- Maintain an application with easySoft 7.
- Want to diagnose faults in an application

easyE4 devices must be installed and connected exclusively by electricians and people who are familiar with electrical installation work.

**CAUTION**

Installation requires qualified electrician

**Follow the safety instructions for the easyE4!**
The section on safety instructions must be read and understood by everyone who will be working with the easyE4 before the actual work is performed.

**WARNING**
**Incomplete operator manual copies**
Working with individual pages taken out from the operator manual may lead to bodily injury and property damage due to missing safety information.

▶ Always work with the latest and full document.

### 0.1.3 Legal disclaimer

All the information in this manual has been prepared to the best of our knowledge and in accordance with the state of the art. However, this does not exclude the possibility of there being errors or inaccuracies. We assume no liability for the correctness and completeness of this information. In particular, this information does not guarantee any particular properties.

Do not use the easyE4 before reading and understanding this manual.

It is assumed that the user of this manual is thoroughly familiar with the information found in the manuals for incorporating the control relay into automation processes.

Hazards posed by the control relay cannot be ruled out if the safety instructions are not observed – especially if the control relay is installed and commissioned by inadequately qualified personnel or if it is used improperly. Eaton assumes no liability for any damages resulting from cases such as these.

The use of sample programs and of the easySoft 7 programming software is subject to the following instructions and rules of use:

1. The program examples provided were created to the best of our knowledge and belief and in accordance with the current state-of-the-art. The program examples provided were created to the best of our knowledge and belief and in accordance with the current state-of-the-art. However, errors cannot be totally excluded, and the example programs do not cover all function blocks and applications that are available for the control relays.
2. Electrical engineering skills and know-how are required in order to be able to program and commission control relays. An incorrectly wired or incorrectly configured control relay will pose a property damage risk and an injury hazard when active components such as motors and pressure cylinders are being driven.
3. When using the provided sample programs and generating a program with SWDeasySoft 7, the user has the sole responsibility to observe the following:
   - All relevant rules and practices for preparing circuit diagrams for the circuit relays as specified in the latest documents for these relays.
   - All occupational health and safety and accident prevention directives, standards, and regulations applicable to the commissioning, circuit diagram creation for, and use of the control relays for your planned application, in particular those imposed by employers' liability insurance associations (Berufsgenossenschaften).
   - Acknowledged rule of technology and state of science.
   - All other general due diligence regarding the prevention of damages to life and physical condition of persons as well as material damage.

4. The manufacturer cannot accept any liability for any damages that are caused by customers not using the program examples provided in accordance with the conditions of use specified here under points 1 to 3.

### 0.1.4 Short designations

The following general terms are used throughout this manual:

| Short designation | Explanation |
|---|---|
| easyE4 | Entire series, used to refer to all the devices in the product family |
| EASY-E4-... | Used to refer to the devices in the series |
| EASY-E4-...-12...C1(P) | Base devices from the product family with an LCD display and a keypad |
| EASY-E4-...-12...C1 | Type With screw terminal connection |
| EASY-E4-...-12...C1P | Type With push in connection |
| EASY-E4-...-12...CX1(P) | Base devices from the product family with diagnostic LEDs |
| EASY-E4-...-12...CX1 | Type With screw terminal connection |
| EASY-E4-...-12...CX1P | Type With screw terminal connection |
| Expansions EASY-E4-...-...E1(P) | All input and output expansions as devices in the product family |
| EASY-E4-...-...E1 | Type With screw terminal connection |
| EASY-E4-...-...E1P | Type With screw terminal connection |

For the exact designation for your easyE4, please refer to the inscription on the device.

### 0.1.5 Writing conventions

Table 1: Format conventions used throughout this manual

| Award | Description |
|---|---|
| `Monospaced` Font | Used for displays, elements at the file level, source code command lines |
| Button | Used for the button labels on the device and in easySoft 7 |
| *Menu path\submenu\...\item* | Used for paths to views and dialog boxes in easySoft 7 |
| *Menu/command* | Used for commands found in the menu |
| **<name>** | Angle brackets are used to indicate variable values that you must replace with your own values |
| `13:08` | Flashing values on the display are shown in gray in this manual |

#### 0.1.5.1 Warning labels

**Risk of personal injury warning.**

**DANGER**

Warns of hazardous situations that result in serious injury or death.

**WARNING**

Warns of the possibility of hazardous situations that could result in serious injury or even death.

**DANGER!**

Dangerous Electrical Voltage!

**CAUTION**

Warns of the possibility of hazardous situations that can cause injury.

**Property damage warning**

*NOTICE*

Warns about the possibility of material damage.

**Prohibited use**

**Prohibited uses, actions, etc.**

Explanation

## 0.1 About this manual

**Bids**

| | |
|---|---|
| | **Bid**<br>Explanation |

**Notes**

→

▶ Indicates instructions to be followed

Additional information, background information, information worth knowing, useful additional information

#### 0.1.5.2 Documents with additional information

Manuals, for example. These will be listed with the corresponding title and Eaton number after the icon.

Publication title For identifying the Eaton publication code

External Internet addresses. They will be shown after the icon.

Destination address

Links to external content are shown in blue.

# 1. easyE4 control relays description

## 1.1 Use as intended

The easyE4 device is a programmable switching and controller device that is used to replace relay and contactor controls.

It is intended exclusively for monitoring, operating, and controlling machines and systems, as well as building and automation services for commercial buildings.

Any other use must be discussed and agreed upon with the manufacturer in advance.

The easyE4 are approved for use in closed spaces.

**Bid**

The easyE4 device must be used only in locations for which the device is approved. Make sure to read and follow the information and labels on the nameplate for the device, as well as section Approvals and standards in the appendix.

**Prohibited uses, actions, etc.**

It is strictly prohibited to use the device to implement safety-relevant functions (in the sense of personal and machine protection) or safety-related controls (such as burner, emergency stop, and two-hand safety controls).

## 1.2 Function

The easyE4 device is an electronic control relay.

With their compact cover dimensions – and a heavy-duty, flat, anti-glare front – the base devices and expansions are ideal for industrial applications.

### 1.2.1 Features

- Logic gates
- Time and counter functions
- Time switch functions
- Arithmetic functions
- PID controllers
- Control relays with 16-character x 6-line LCD display (128 x 96 pixels) and keypad available.
- Function expansions can be implemented with insertable microSD cards
- Integrated operating system, can be loaded
- Built-in Ethernet interface
- Requires little space; can be used in an upright position as well
- Device construction for mounting rails
- Real Time Clock (RTC)
- Programming languages: Ladder diagram (LD), Function Block Diagram (FBD), Structured Text (ST), and easy Device Programming (EDP) on device and in easySoft 7

easyE4 base devices combine the functions of a control relay and an input device in one single unit

The Ethernet port makes it possible to integrate the base device into a network.

This allows the design of systems using high-speed controllers with decentralized intelligence.

There is ladder diagram language version called easy Device Programming (EDP) that you can use to put together a circuit diagram on the device.
In the case of devices with a display, you can enter the program as a circuit diagram directly on the device by using the corresponding buttons. You can also program it on your computer with the easySoft 7 programming software program (this option can also be used for base devices without a display).

For example, you can:

- Connect N/O and N/C contacts in series and in parallel
- Connect output relays and markers.

- Define outputs as coils, impulse relays, rising or falling edge-triggered relays or as latching relays.
- ...

You can use the function blocks to run arithmetic functions, compare values, count up, count down, etc. All the function blocks available are provided in a list

→ Section "Function blocks", page 216

If you wish to wire a easyE4 device via your PC, i.e. create a circuit diagram, use the easySoft 7

→ Section "Programming software easySoft 7", page 32.

If you want to connect a easyE4 device to a visualization, use the XV-102-AO-35TQRB-1E4 3.5" HMI touch display

→ Section "Convenient visualization for easyE4", page 680.

## 1.3 Device models - versions and part nos.

All easyE4 devices come with an operating system (firmware).

The base devices in the easyE4 series feature:

- a microSD memory card slot
- an Ethernet port (10/100 Mbit/s) that can be used as a communication or network interface

The functionality for each base device can be customized with up to 11 expansions from the easyE4 series.

### 1.3.1 Base device versions

The available base device versions are different from each other in terms of:

- The type of operating voltage - UC, DC or AC
- The type of outputs - Relays or Transistor
- The type of terminals - screw terminals or push in terminals

and

- The type of controls - a display and buttons or an LED display

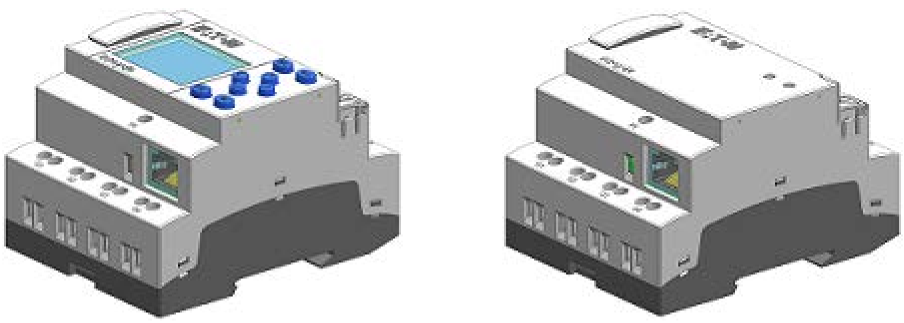
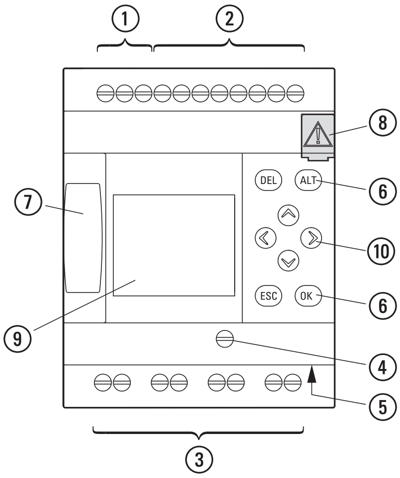
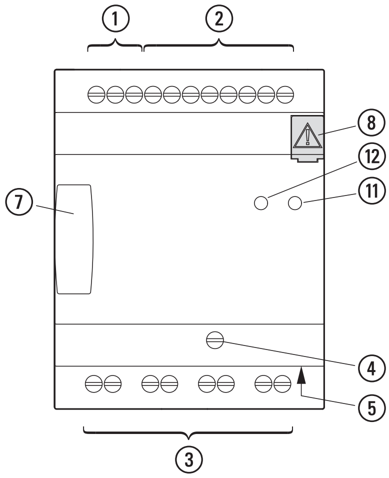


Figure 1: Device model with EASY-E4-...-12...C1(P) display and button controls or with EASY-E4-...-12...CX1(P) LED display for diagnostics

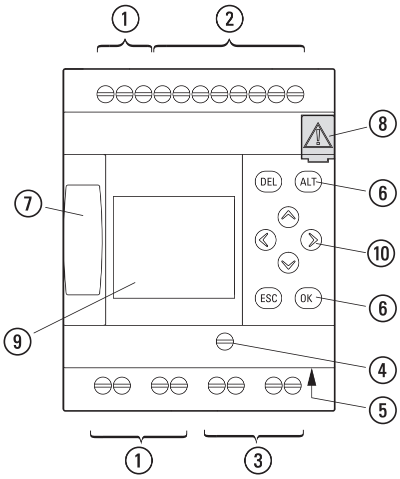
EASY-E4-UC-12RC1(P),
EASY-E4-AC-12RC1(P)



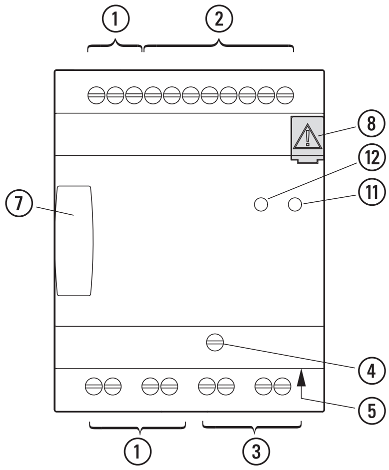

EASY-E4-UC-12RCX1(P),
EASY-E4-AC-12RCX1(P)



EASY-E4-DC-12TC1(P)




EASY-E4-DC-12TCX1(P)



| | | | | | |
|---|---|---|---|---|---|
| ① | Power Supply | ⑥ | Pushbuttons | ⑪ | LED POW/RUN |
| ② | Input points | ⑦ | Slot for microSD memory card | ⑫ | LED ETHERNET/NET |
| ③ | Outputs | ⑧ | Covering cap | | |
| ④ | Ethernet connection Functional earth | ⑨ | Display | | |
| ⑤ | Ethernet socket | ⑩ | Cursor buttons | | |

### 1.3.2 Expansion versions

The available input and output expansion devices are different from each other in terms of:

- The type of operating voltage - UC, DC or AC,
- The type and number of inputs/outputs - relay or transistor
- The corresponding function - temperature, for example
- The type of terminals - screw terminals or push in terminals

and

- in terms of width - 4 or 2 (space units SU).

EASY-E4-UC-16RE1(P),
EASY-E4-DC-16TE1(P),
EASY-E4-AC-16RE1(P)

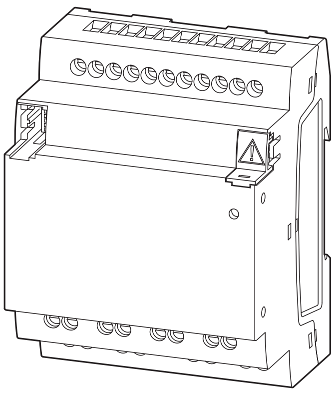


Figure 2: Device model with 4SU

EASY-E4-UC-8RE1(P),
EASY-E4-DC-4PE1(P),
EASY-E4-DC-6AE1(P),
EASY-E4-DC-8TE1(P),
EASY-E4-AC-8RE1(P)

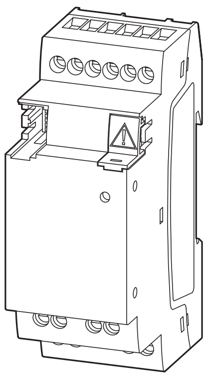
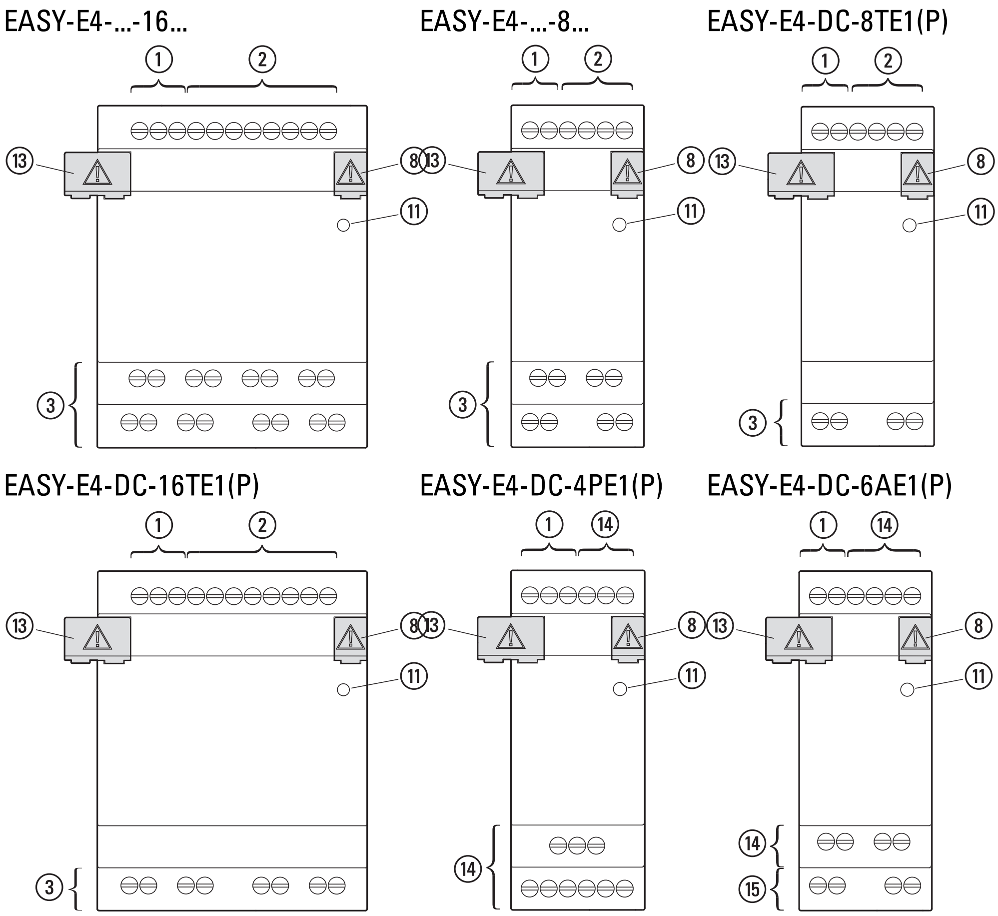


Figure 3: Device models in 2SU




| | | | |
|---|---|---|---|
| ① | Power Supply | ⑬ | Bus connector plug |
| ② | Input points | ⑭ | Analog inputs |
| ③ | Outputs | ⑮ | Analog outputs |
| ⑧ | Covering cap | | |
| ⑪ | LED POW/RUN/ Status | | |

### 1.3.2.1 Overview of available easyE4 devices

Make sure to take advantage of the EATON online catalog. Enter "easy" into the search box and the catalog will take you directly to the corresponding product group in the Automation, Control and visualization section.

http://www.eaton.eu/ecat

**easyE4 control relays**

- With screw terminals or
  EASY-E4-..-....1P push in terminals

| Catalog No. and type | Description |
|---|---|
| 197211 - EASY-E4-UC-12RC1<br>197504 - EASY-E4-UC-12RC1P | Base device with display;<br>12/24 VDC, 24 VAC;<br>digital inputs: 8, of which 4 can be used as analog inputs;<br>digital outputs: 4 relay outputs |
| 197212 - EASY-E4-UC-12RCX1<br>197505 - EASY-E4-UC-12RCX1P | Base device with diagnostic LED;<br>12/24 VDC, 24 VAC;<br>digital inputs: 8, of which 4 can be used as analog inputs;<br>digital outputs: 4 relay outputs |
| 197213 - EASY-E4-DC-12TC1<br>197506 - EASY-E4-DC-12TC1P | Base device with display;<br>24 VDC;<br>digital inputs: 8, of which 4 can be used as analog inputs;<br>digital outputs: 4 transistor outputs |
| 197214 - EASY-E4-DC-12TCX1<br>197507 - EASY-E4-DC-12TCX1P | Base device with diagnostic LED;<br>24 VDC;<br>digital inputs: 8, of which 4 can be used as analog inputs;<br>digital outputs: 4 transistor outputs |
| 197215 - EASY-E4-AC-12RC1<br>197508 - EASY-E4-AC-12RC1P | Base device with display;<br>100 - 240 V AC, 100 - 240 V DC (cULus 100 - 110 V DC),<br>digital inputs: 8; digital outputs: 4 relay outputs |
| 97216 - EASY-E4-AC-12RCX1<br>97509 - EASY-E4-AC-12RCX1P | Base device with diagnostic LED;<br>100 – 240 VAC, 100 – 240 VDC (cULus 100 - 110 V DC),<br>digital inputs: 8; digital outputs: 4 relay outputs |

I/O expansion for easyE4 control relays

- With EASY-E4-...-...E1screw terminals or EASY-E4-...-...E1P push in terminals

| Catalog No. and type | Description |
|---|---|
| 197217 - EASY-E4-UC-8RE1<br>197510 - EASY-E4-UC-8RE1P | 12/24 V DC, 24 V AC,<br>digital inputs: 4,<br>digital output: 4 relays |
| 197218 - EASY-E4-UC-16RE1<br>197511 - EASY-E4-UC-16RE1P | 12/24 V DC, 24 V AC,<br>digital inputs: 8,<br>digital output: 8 relays |
| 197219 - EASY-E4-DC-8TE1<br>197512 - EASY-E4-DC-8TE1P | 24 V DC,<br>digital inputs: 4,<br>digital output: 4 transistors |
| 197220 - EASY-E4-DC-16TE1<br>197513 - EASY-E4-DC-16TE1P | 24 V DC,<br>digital inputs: 8,<br>digital output: 8 transistors |
| 197221 - EASY-E4-AC-8RE1<br>197514 - EASY-E4-AC-8RE1P | 100 - 240 V AC, 100 - 240 V DC (cULus 100 - 110 V DC), digital inputs: 4,<br>digital output: 4 relays, |
| 197222 - EASY-E4-AC-16RE1<br>197515 - EASY-E4-AC-16RE1P | 100 - 240 V AC, 100 - 240 V DC (cULus 100 - 110 V DC), digital inputs: 8,<br>digital output: 8 relays |
| 197223 - EASY-E4-DC-6AE1<br>197516 - EASY-E4-DC-6AE1P | 24 V DC,<br>analog inputs: 4,<br>analog outputs: 2 |
| 197224 - EASY-E4-DC-4PE1<br>197517 - EASY-E4-DC-4PE1P | with temperature measuring<br>Pt100, Pt1000 or Ni1000<br>24 VDC;<br>analog inputs: 4;<br>outputs: None |

## 1.4 What the different parts of the part number mean

The part number includes information that specifies the version and model of the specific device being used.

The Part number can be found at the front of the easyE4.

Table 2: Key to part numbers

| easy-E4 | - | .C | - | .. | | ... | - | x1(P) |
|---|---|---|---|---|---|---|---|---|
| Power rating | | Type of supply voltage | | Number of inputs/outputs | | Type of output<br>R-Relay<br>T-Transistor<br>A-Analog<br>P-Temperature | | E-Expansion<br>CX-Base device with LED diagnostics<br>C-Base device with display and buttons<br>1-Version<br>P-Type With push in connection instead of With screw terminal connection. |

## 1.5 Accessory devices

In addition to the expansions, there are additional accessories available for easyE4 control relays.

| |
|---|
| *NOTICE*<br>Only use original accessories. |

→ Order accessories through your supplier or through the EATON online catalog

www.eaton.eu/ecat

Example:

| Cat No. and type | Description |
|---|---|
| 198513 XV-102-AO-35TQRB-1E4 | Touch display for easyE4, 24 VDC, 3.5 Zoll, TFTcolor, Ethernet |
| 198514 XV100-BOX-E4-DC1 | Starter package consisting of EASY-E4-DC-12TC1, XV-102-AO-35TQRB-1E4 touch display, Ethernet switch, three patch cables, and license for easySoft 7 |
| 198515 XV100-BOX-E4-UC1 | Starter package consisting of XV-102-AO-35TQRB-1E4, EASY-E4-UC-12RC1, Ethernet switch, three patch cables, and license for easySoft 7 |
| 191087 MEMORY-SUD-A1 | microSD<br>2 GB memory card with adapter, I Grade, without an operating system |
| 197226 EASYSOFT-SWLIC | Programming software license easySoft 7 |
| 061360 ZB4-101-GF1 | ZB4-101-GF1 Device foot for screw mounting |
| 197225 EASY-E4-CONNECT1 | EASY-E4-CONNECT1 spare parts package<br>Consists of three (3) connectors and three (3) end covers for the easyE4 series between the control relay and input/output expansions |
| 272484 - TR-G2/24 | Transformer, 230 V, 12/24 V, 2/1 A |

## 1.6 Nameplate

The device can be identified by checking the nameplate on its side.
This nameplate includes the following information:

- Manufacturer
- Version
- Operational voltage
- Heat dissipation information
- Type approval and certification marks and information
- Information relevant to UL listing

In addition to the device's part number and MAC address, the QR code in the front also contains additional information:

- Serial number
- Production Date

## 1.7 Support

To get fast and effective support, make sure to always provide Customer Service with the following information:

- Part number
- Information from the QR code
- Ambient conditions at the location of use
- Fuse or other protective element used to protect the device
- Supply voltage conditions
- Device operating system (firmware) version
- If applicable, easySoft 7 build No., version

## 1.8 Programming software easySoft 7

easyE4 control relays are designed to be programmed with the easySoft 7 programming software program. This program was developed specifically for this series of devices, and makes it possible to quickly, conveniently, and easily integrate available functions into a circuit diagram and use the result as a control program.
The program is available free of charge. However, you will need a software license in order to be able to use all of its functions.

→ Some functions are not available in the demo version.

easySoft 7 can also be used to:

- Test your circuit diagram by simulating the power flow (offline test).
- Transfer your circuit diagram to a connected and operational easyE4 base device.
- Monitor the power flow and view operand states after transferring the circuit diagram (online test)
- Print out your circuit diagram so that you can document it in detail

In addition, the program makes it possible for you to use a password to protect your projects and, accordingly, your know-how.

The easySoft 7 Help is an integral part of easySoft 7 and is designed to help you use the programming software.

**Tutorials**

For helpful videos that explain how to use specific functions, please visit the product page at http://www.eaton.eu/easy.

**Application examples**

Support has provided a number of applications that are available for download as ZIP files from the Software Download Center.

Download Center - Software
http://www.eaton.eu/software/Anwendungsbeispiele/easy/Deutsch
http://www.eaton.eu/software/Application Samples/easy/English

These examples come with a task description, the circuit diagram, and the easySoft 7 project (in the EDP and LD programming languages as of this writing).

# 1.9 Safety regulations

## 1.9.1 Basics

The device has been designed according to the state of the art and all generally accepted safety rules and standards. However, this alone cannot eliminate all potential hazards, which is why it is necessary for you to be aware of all hazards and residual risks.

Do not run the device unless it is in perfect technical condition. Make sure to always operate it as specified in this document and for the intended purpose.

**Follow the safety instructions for the easyE4!**
The section on safety instructions must be read and understood by everyone who will be working with the easyE4 before the actual work is performed.

*NOTICE*
Pay attention to the hazard severity levels used throughout this documentation whenever a hazard is indicated. The hazard symbol and signal word used and the corresponding text will provide information regarding the specific hazard and how to avoid or prevent it.

## 1.9.2 Mandatory requirements, personnel requirements

### 1.9.2.1 Occupational safety

All generally accepted occupational health and safety rules and standards (internal and national) must be complied with, as must be all applicable laws and regulations in the relevant country.

### 1.9.2.2 Personnel qualifications

The personnel responsible for installation, operation, maintenance, and repairs must have the necessary qualifications for the work they will be performing. They must be appropriately trained and/or briefed and be informed of all hazards and risks associated with the device.

#### 1.9.2.3 Device documentation

This manual is considered an integral part of the device and must always be readily available in the device's close proximity so that users have access to it.

Make sure that every person who will be working with the device, regardless of the life-cycle stage involved, has read and understood the relevant parts of the documentation for the device.

Additional parts of the documentation and information for the easyE4, including the installation instructions, can be found at the Eaton Download Center - Documentation and at the product pages on the Internet

http://www.eaton.eu/doc

http://www.eaton.eu/easy

**WARNING**

**Incomplete operator manual copies**

Working with individual pages taken out from the operator manual may lead to bodily injury and property damage due to missing safety information.

▶ Always work with the latest and full document.

#### 1.9.2.4 Installation, maintenance, and disposal

Make sure that the device is connected, installed, serviced, and disposed of professionally and in line with all relevant standards and safety rules.

**CAUTION**

Installation requires qualified electrician

**Important!**

Dispose of recyclables as required by your local recycling regulations.

Devices no longer being used must be professionally disposed of as per local regulations. To learn more, please visit:

Eaton.com/recycling

#### 1.9.2.5 Prerequisites for proper operation

In order for the device to be able to meet the contractually stipulated terms, the following must be observed:

- Only qualified personnel should be allowed to work with the device.
- The personnel working with the device must have read and understood all documents for the device and must follow all the instructions in them.
- The required ambient conditions must be met.
- Maintenance work must be carried out correctly.

→ Make sure to read the → "Legal disclaimer", page 15.

We assume no liability for damages, consequential damages, and/or accidents caused by the following:

- Failure to follow any applicable occupational health and safety rules, standards, and/or regulations
- Device failures or function disturbances
- Improper use and/or handling
- Not following the instructions or observing the information in the documentation for the device
- Alterations, changes, and repairs to the device

### 1.9.3 Device-specific hazards

> **CAUTION**
> **DESTRUCTION**
> The easyE4 should only be opened by the manufacturer or by an authorized center. Operate the device until only with the enclosure fully closed and sealed.

> **CAUTION**
> **ELECTROSTATIC DISCHARGE**
> Do not touch components (e.g., connector pins) that are electrostatic-sensitive.
>
> ▶ Discharge (by touching a grounded metal object) any static charge accumulated in your body before touching the device.
>
> Electrostatic discharges may damage or ruin assembly parts. Because of this, it is necessary to take precautions whenever handling the cards. Please refer to the guidelines for electrostatic-sensitive components for more information (ESD guidelines).

> **CAUTION**
> **INTERFERENCES**
> The values specified in the technical data, as well as the device's electromagnetic compatibility (EMC), cannot be guaranteed if the following are used: unsuitable cables, improperly assembled and terminated cables, and/or wiring that does not conform to the applicable standards.
> Only use cables assembled and terminated by professionals.
> The cables being used must be assembled and terminated as required by the port/interface description in this document.
> When wiring the devices, follow all instructions regarding how to wire the corresponding port/interface.
> All general Directives and standards must be complied with.

> **CAUTION**
> **INTERFERENCES**
> Screw all plug-in connections or lock them into place in order to improve screening.
> Signal cables must not be routed in the same cable duct with power cables.
> Before putting the system into operation, check all cable connections to make sure that everything has been wired properly.
> Make sure that all voltages and signals have the required values as specified in the technical data.

**CAUTION**
**SAFELY DIVERTING ELECTRICAL INTERFERENCE CURRENTS**

The devices must be connected to a central earth point with a conductor that is as short and has as low a resistance as possible.

- Ground connection characteristics:
  Wire cross-sectional area ≧ 1.5 mm², length ≦ 350 mm

The easyE4 needs to be connected to the conductive structure in, e.g., the control panel using the central earth point (earthing screw). This method of earthing is mandatory required for proper function.

**DANGER**
**STRAY CURRENTS**

Large equalizing currents between the functional earthing system and the ground system of different devices may result in fire or in malfunctions due to signal interference.

▶ If necessary, route an equipotential bonding conductor, with a cross-sectional area that is several times larger than that of the cable shielding, parallel to the cable.

**CAUTION**
**DATA LOSS**

If the SD card is being written to and a voltage drop occurs or the card is removed, data may be lost or the SD card may be ruined.

▶ Insert the SD card only when the easyE4 is de-energized.

Avoid writing to SD cards. Reasons:

- SD cards have a limited number of write cycles.
- If there is a voltage drop while a write operation is in progress, data loss is highly likely to occur.

▶ Remove the SD card only when the easyE4 is de-energized.

▶ Before switching off the device, make sure that there are no programs writing to the SD card.

**CAUTION**
**SHORT-CIRCUIT HAZARD**

If the device is or has been exposed to environmental fluctuations (ambient temperature, air humidity), condensation may form on or inside it. As long as this condensation is present, there will be a short-circuit hazard.
Do not switch on the device when it has condensation in or on it.
If the device has condensation in or on it, or if the panel has been

exposed to environmental fluctuations, let the panel settle into the existing ambient temperature before switching it on.

**CAUTION**
**UV LIGHT**
Plastics will become brittle when exposed to UV light. This artificial aging will reduce the easyE4 unit's lifespan. Protect the device from direct sunlight and other sources of UV radiation.

**CAUTION**
**POINTY, SHARP OBJECTS AND CORROSIVE LIQUIDS**
When cleaning the device:

- Do not use any pointy or sharp objects (e.g., knives).
- Do not use aggressive or abrasive cleaning products or solvents.

Make sure that no liquids get into the device (short-circuit hazard) and that the device is not damaged in any way.

**CAUTION**
**INSTALLATION CUT-OUT**
The mounting cutout must be located in a position that will not defeat the purpose of stabilizing webs or other reinforcing elements in the control panel. If necessary, reinforcing elements must be installed/added.

**CAUTION**
**MECHANICAL FORCES ON THE ETHERNET PORT**
Communications may be affected, and the connection's mechanical components may be damaged, if the Ethernet interface is subjected to strong vibrations or the RJ45 plug-in connection is subjected to pulling.

- Protect the RJ45 plug-in connection from strong vibrations.
- Protect the RJ45 plug-in connection from tensile forces at the socket.

**CAUTION**

Installation requires qualified electrician

# 1.10 Engineering

The easyE4 series makes it possible to combine multiple voltage variants. Each easyE4 base device can be wired with up to 11 EASY-E4-...-...E1(P) expansions with different power supplies.

## 1.10.1 Length of input cables

Severe interference can cause a signal 1 on the inputs without a proper signal being applied. Observe therefore the following maximum cable lengths:

Base device inputs I1–I6

- 40 m for AC supply voltages; 100 m for DC supply voltages

Base device inputs I7–I8

- 100 m for AC supply voltages; 100 m for DC supply voltages

Expansion device inputs

- 40 m for AC supply voltages; 100 m for DC supply voltages

In addition, the following applies to base devices and expansion devices:

With longer cables, connect a diode (e.g. 1N4007) with a minimum reverse voltage of 1 kV and a let-through current of 1 A in series to the device inputs. Ensure that the diode is pointing to the input; otherwise the device will not be able to detect the 1 state.

### 1.10.2 Analog Signals

**DANGER**
Analog signals are more sensitive to interference than digital signals, therefore the signal cables should be carefully routed and connected. An incorrect connection can lead to unwanted switching states.

The following measures must be adhered to in order to prevent any deviations in analog values.

**Tips for analog signals**

- Use shielded cables.
- Keep signal cables as short as possible and not longer than 10 m.
- In the case of short cable lengths, terminate the signal cables' shield to the 0 V terminal on both sides across the entire area.
  In the case of longer signal cables, the shield should only be terminated on one end, i.e., on the side of the EASY-E4-... devices.
  Otherwise compensation currents between both grounding points may flow, leading to the interference of analog signals.
- Lay signal cables separately from heavy current cables.
- Connect inductive loads that you are switching via the outputs of the EASY-E4-... base devices to a separate power supply or use a suppressor circuit for motors and valves.
  If loads from motors, solenoid valves or contactors are operated with the same power feed as EASY-E4-... devices, switching may give rise to interference on the analog input signals.
- Make sure that the reference potential is galvanically connected.

### 1.10.3 Notes on connecting EASY-E4-AC-... devices

**Special considerations for EASY-E4-AC-... expansions**

Connect inputs I1–I8 on AC base devices and I1–I4 on expansion devices in accordance with all applicable VDE, IEC, UL, and CSA safety rules using the same phase conductor that delivers the supply voltage. Otherwise, the device will not detect the switching level or may be destroyed by overvoltage.

During wiring, make sure to meet all cable protection requirements.

**AC base devices**

Input signal voltage range

- Off signal: 0 to 40 V.
- On signal: 79 to 264 V

Input current

- I1 to I6: 0.5 mA/0.25 mA at 230 V/115 V.
- I7, I8: 6 mA/4 mA at 230 V/115 V.

**In addition, the following applies to I1–I6 on base devices:**

With longer cables, connect a diode (e.g. 1N4007) with a minimum reverse voltage of 1 kV and a let-through current of 1 A in series to the device inputs. Ensure that the diode is pointing to the input; otherwise the device will not be able to detect the 1 state.

**Connecting inputs I7/I8**

Neon lamps with a maximum residual current of 2 mA / 1 mA at 230 V/115 V can be connected to I7 and I8.

→ Always use neon bulbs that are operated with a separate N connection.

**WARNING**
Do not use reed relay contacts on I7, I8. These may burn or melt due to the high inrush current of I7, I8.

Two-wire proximity switches have a residual current in the 0 state. If this residual current is too high, the device only detects the 1 state at the input.

With two-wire proximity switches or sensors with similar residual current consumption use therefore the inputs I7 and I8.

Use an additional input circuit if several inputs are required with a higher input current.

**AC expansions**

Connect the inputs, for example, to pushbutton actuators, switches or relay/contactor contacts.

Input signal voltage range

- Off signal: 0 to 40 V.
- On signal: 79 to 264 V

Input current

- I1 to I8: 0.5 mA/0.25 mA at 230 V/115 V.

In addition, the following applies to AC expansion devices:

With longer cables, connect a diode (e.g. 1N4007) with a minimum reverse voltage of 1 kV and a let-through current of 1 A in series to the device inputs. Ensure that the diode is pointing to the input; otherwise the device will not be able to detect the 1 state.

# 2. Installation

**CAUTION**

Installation requires qualified electrician

easyE4 devices must be installed and wired exclusively by an electrician or a person who is familiar with electrical installation rules and practices.

The devices are installed in the following order:

1. Mounting base device
2. Assemble the base device and expansion devices into a block (optional)
3. Connecting the power supply
4. Connecting inputs
5. Connecting outputs
6. Connect to Ethernet

**DANGER OF ELECTRICAL SHOCK!**
Dangerous Electrical Voltage!
All installation work must be carried out with the entire installation in a de-energized state.

Always comply with all applicable country-specific safety rules and regulations:

1. Switch off and isolate
2. Secure against retriggering
3. Verify isolation from the supply
4. Earthing and short-circuiting
5. Covering or providing barriers to adjacent live parts

What to do before turning the device back on

- Remove all tools and materials
- Leave the danger zone
- Remove the short-circuiting and grounding at the area where work was performed and then elsewhere
- Disconnect the ground wire from the system components first, then from the ground
- Do not touch system components or cables without a ground wire (if there was one previously) anymore
- Reinstall all safety covers, safety enclosures, safety labels, and safety signs

- Do not remove safety measures at switching points until you get the all-clear for the areas where work was performed
- If carrying out work that involves more than one worker, make absolutely sure that nobody is still in the danger zone

## 2.1 Prerequisites for the location of use

The device must be used exclusively in locations for which it has been approved/certified.

The supply voltage must be guaranteed and must conform to the relevant specifications-

Nameplate → page 30 and
the specifications on the → Section "Technical data", page 699 for the individual devices → page 699

> **CAUTION**
> **INSTALLATION CUT-OUT**
> The mounting cutout must be located in a position that will not defeat the purpose of stabilizing webs or other reinforcing elements in the control panel. If necessary, reinforcing elements must be installed/added.

### 2.1.1 Installation position

easyE4 devices are intended to be flush mounted in control cabinets, control panels, service distribution boards, or control consoles from behind.

The following must be taken into account when selecting the installation position:

- The controls and connectors must remain accessible even after the device has been installed.
- easyE4 devices can be installed in a horizontal or vertical position.

→ The slot for the microSD memory card is located under a cover on the base device.
Please observe the clearance required in order to be able to remove the microSD and use the buttons.

#### 2.1.1.1 Temperatures

Make sure that the device does not overheat.

Do not expose the device to direct sunlight or other sources of heat.
The minimum clearance to components that radiate heat, such as transformers under heavy loads, is 15 cm.

**CAUTION**
**UV LIGHT**
Plastics will become brittle when exposed to UV light. This artificial aging will reduce the easyE4 unit's lifespan. Protect the device from direct sunlight and other sources of UV radiation.

The environmental ambient conditions for operation must not exceed the specified values:

| **Ambient climatic conditions** | |
|---|---|
| Air pressure (in operation) | 795 - 1080 hPa |
| | Max. 2000 m above sea level |
| Temperature | |
| Operation | - 25 – +55 °C (-13 – +131 °F)<br>The display is readable between ϑ -5°C (-23°F) ≦ T ≦ 50°C (122°F). |
| Storage / Transport | - 40 – +70 °C (-40 – +158 °F) |
| Humidity | Relative humidity 5 - 95 % |
| Condensation | Prevent condensation by means of suitable measures |

#### 2.1.1.2 Aeration and de-aeration

- The device uses natural convection-based passive cooling, i.e., it does not use fans.
- Make sure that there will be enough volume for air changes inside the control panel, etc.
  The specified clearance around the easyE4 is: a, b, c ≧ 30 mm (1.2")
- When installing the easyE4 in complex systems together with other assemblies, you must ensure that there will be enough air circulation in order to prevent overheating.
  Ambient temperature for natural convection: ϑ -25°C (-13°F) ≦ T ≦ 55°C (131°F)
  The display (option) will be readable between ϑ -5°C (-23°F) ≦ T ≦ 50°C (122°F).
  The panel builder is responsible for the temperature rise calculation. Eaton will provide heat dissipation data for the easyE4 as necessary for design verification in accordance with IEC EN 61439.

## 2.2 Unpacking and checking the equipment supplied

- Check the easyE4's packaging for transit damage.
- Carefully remove the packaging in order to avoid damaging the device.
- Check the package contents for visible transit damage.
- Use the information in Installation instructions to make sure that the contents are complete.

→ Keep the original packaging so that you will be able to use it in the future if you need to transport or ship the device.
Make sure to also keep the documents enclosed with the device and/or to give them to the end customer.

The package for the easyE4 series comes with:

Table 3: Std. packeasyE4 control relays

| Unit | Description |
|---|---|
| 1 x | EASY-E4-...-12...C1(P) or EASY-E4-...-12...CX1(P) |
| 1 x | Installation instructions IL050020ZU |

| Unit | Description |
|---|---|
| 1 x | EASY-E4-...-...E1(P) |
| 1 x | Bus connector plug EASY-E4-CONNECT1 |
| 1 x | Installation instructions IL050021ZU |

The easyE4 is sturdily built, but the components inside it are sensitive to excessively strong vibrations and/or mechanical shock.

Accordingly, make sure to protect the easyE4 from mechanical loads that exceed the scope of the unit's intended use.

The device should only be transported in its original packaging after being packed properly.

### Missing parts or damage

If you notice anything wrong, please contact your distributor or
Eaton Service +49 (0) 180 5 223822 (de,en)

## 2.3 Mounting

> *NOTICE*
> Arrange for a professional technician to mount the device.

> **CAUTION**
> **INSTALLATION CUT-OUT**
> The mounting cutout must be located in a position that will not defeat the purpose of stabilizing webs or other reinforcing elements in the control panel. If necessary, reinforcing elements must be installed/added.

▶ Check to make sure that the installation clearances are being observed
→ Section " Installation position", page 44

▶ Make sure that the mounting cutout has the right size.

**Mounting EASY-E4-...**

Mounting on ICE/EN 60715 mounting rail OR

With screws and ZB4-101-GF1 mounting feet.

### 2.3.1 Mounting easyE4 control relays

Install a easyE4 control relays in an enclosure, switch cabinet or distribution board so that the power supply and terminal connections cannot be touched accidentally during operation.

The easyE4 control relays can be mounted either vertically or horizontally.

For ease of wiring, leave a clearance of at least 3 cm between the device terminals and the wall or adjacent devices.

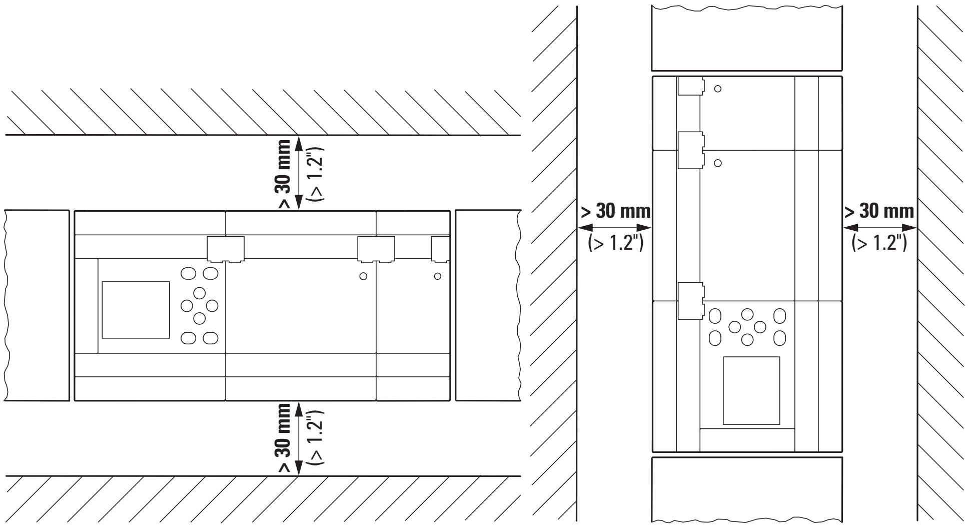



Figure 4: Mounting distance min. 3 cm

▶ Snap the base device and every expansion onto a mounting rail or mount every device using ZB4-101-GF1 device feet

Local expansion units are connected directly next to the basic unit.

You can use the EASY-E4-CONNECT1 connector to connect the easyE4 base device to up to 11 expansions and assemble them into a single device block.

Expansion devices come with an EASY-E4-CONNECT1 connector as standard.

You can use the expansion devices to:

- Increase the number of inputs/outputs
- Combine various voltages
- Process analog/digital signals

You can use all digital and analog expansion devices regardless of the corresponding operating voltage.

Each expansion needs to be mounted individually on the mounting rail or with screws and device feet, much like the base device. Once you are done mounting all the devices, use the connector to combine them into a single device block.

▶ Connect the base device to the expansion, and the expansions to each other, with the EASY-E4-CONNECT1 connector.

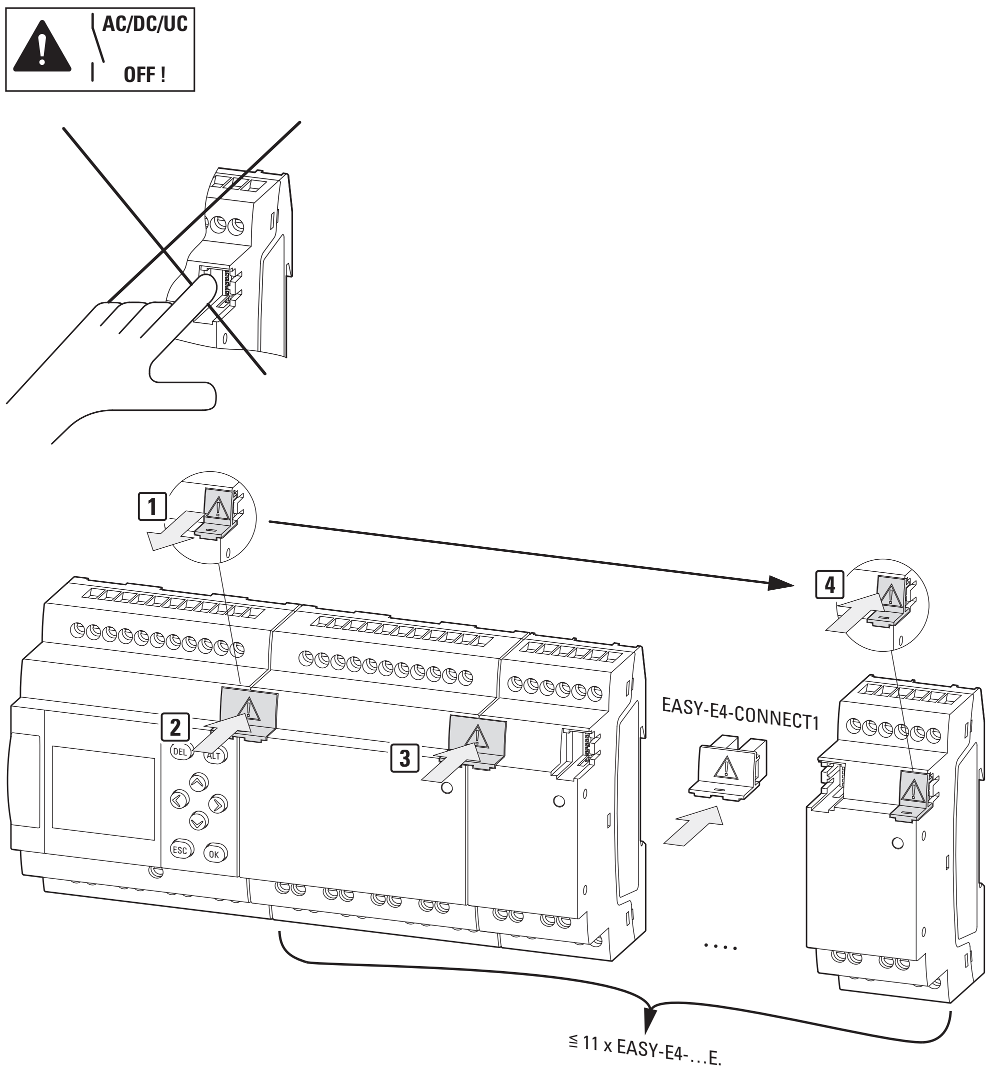



Figure 5: Assembling a base device with expansions

#### 2.3.1.1 Installation on mounting rail

1. Position the base device against the mounting rail's upper lip in an inclined position.
2. Lightly push the device down and against the mounting rail until it snaps into place over the mounting rail's lower lip.

The device will clip into place automatically.

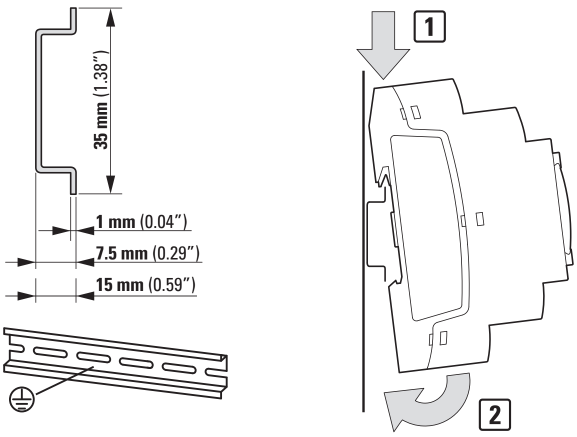



Figure 6: Installation on IEC/EN 60715 mounting rail

3. Check that the device is seated firmly.

The device is mounted vertically on a mounting rail in the same way.

### Mounting the first expansion (optional)

1. Position the expansion device against the mounting rail's upper lip in an inclined position to the right of the base device.
2. Slide the expansion until it is resting flush against the base device.
3. Lightly push the device down and against the mounting rail until it snaps into place over the mounting rail's lower lip.
4. Remove the end cover from the base device and store it in a safe place.
5. Connect the base device to the expansion using the connector.

### Mounting additional expansions (optional)

1. Position the expansion device against the mounting rail's upper lip in an inclined position to the right of the previous expansion.
2. Slide the expansion until it is resting flush against the previous expansion.
3. Lightly push the device down and against the mounting rail until it snaps into place over the mounting rail's lower lip.
4. Connect the expansions to each other using the connector.
5. Repeat for all additional expansions – up to 11 EASY-E4-...-...E1(P)

### Finishing up the process

1. Take the base device's end cover and install it on the right side of the last expansion.

There will be the following electrical isolation at the local expansion connection between the base device and the expansion device:

- Basic isolation 400 V AC (+10 %).
- Safe isolation 240 V AC (+10 %).

Basic device and expansion unit can be provided with different power supplies.

#### 2.3.1.2 Screw mounting

Fixing brackets ZB4-101-GF1 that can be inserted on the rear of the easyE4 devices are required for screw fixing.
These feet are available as an accessory – please refer to → Section " Accessory devices", page 29.

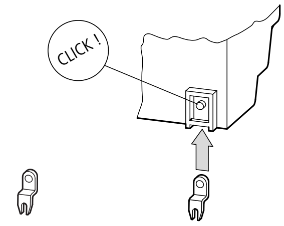



Figure 7: Inserting a fastening bracket.

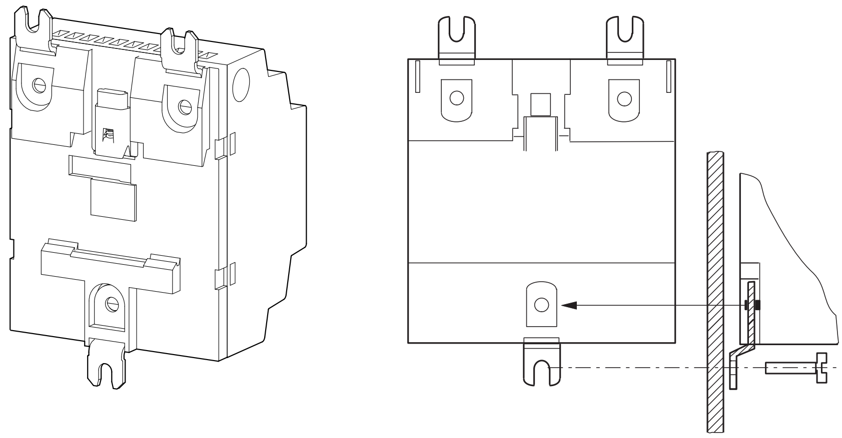


Figure 8: Screw mounting configuration for a device

→ The base devices and EASY-E4-...-16... 4U expansions, such as EASY-E4-UC-16RE1(P), require three feet each.
EASY-E4-...-8... 2U expansions, such as EASY-E4-DC-8TE1(P), EASY-E4-DC-6AE1(P), and EASY-E4-DC-4PE1(P), require two feet each.

### 2.3.1.3 Dismounting of a device

- Disconnect all the connections and wires for the device
- If the device is a standalone base device, you can remove it directly
- If you are working with a block consisting of a base device and expansion devices, remove the EASY-E4-CONNECT1 connectors

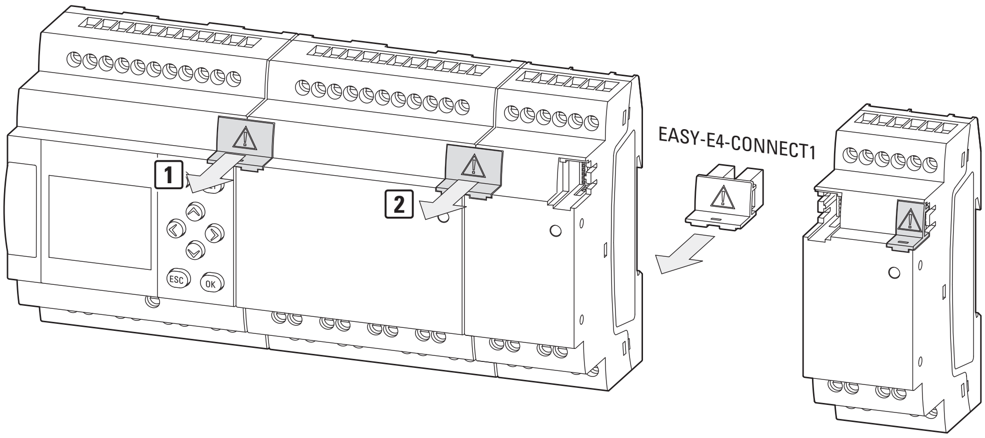



Figure 9: Remove adjacent connectors

- Remove the device from the mounting rail

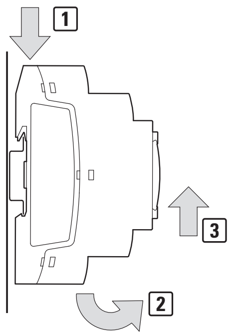



Figure 10: Dismantling

- Screw mounting option:
  Unscrew the screws on the device feet.

# 2.4 Connection terminals

All devices are available in a version with screw terminals and a version with push in terminals.
A flat-blade screwdriver is required in order to work with these terminals:

- With screw terminal connection
  Slot-head screwdriver, width 0.8 mm x 3.5 mm.
- With push in connection
  Slot-head screwdriver, width 0.4 mm x 2.5 mm.

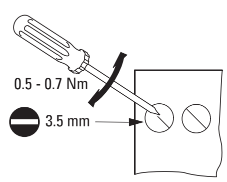
## 2.4.1 Screw terminal connection

EASY-E4-...-12...C1, EASY-E4-...-12...CX1, and EASY-E4-...-...E1 devices are designed for use with screw terminals.

The stripping length for the individual conductors, or the length of the wire end sleeve at the individual conductors, is 6.5 mm (0.26“) for this connection.

▶ Connect the individual conductors with a tightening torque of 0.5 - 0.7 Nm.

| **6,5 mm** (0.26") | | Terminal capacity in mm² |
|---|---|---|
| | Solid | 0.2 up to 4 |
| | Flexible | 0.2 up to 2.5 |
| | Conductor cross section AWG | min 22 - max 12 |
| | Solid cable with ferrule | 0.2 up to 2.5 |
| | Flexible with ferrule | |

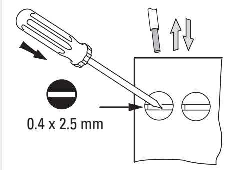


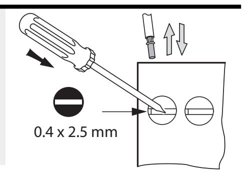
### 2.4.2 Push in connection

EASY-E4-...-12...C1P, EASY-E4-...-12...CX1P, and EASY-E4-...-...E1P devices are designed for use with push in terminals.

The stripping length for the individual conductors, or the length of the wire end sleeve at the individual conductors, is 8 mm (0.31“) for this connection.

▶ Push the individual conductors directly into the push in terminal until the terminal locks in place. If necessary, use a flat-blade screwdriver to help.

| **8 mm** (0.31”) | | Terminal capacity in mm² |   |
|---|---|---|---|
| | Solid | 0.2 up to 2.5 | |
| | Flexible | | |
| | Conductor cross section AWG | min 24 - max 14 | |

| **8 mm** (0.31”) | | Terminal capacity in mm² |   |
|---|---|---|---|
| | Solid cable with ferrule | 0.25 up to 1.5 | |
| | Flexible with ferrule | | |

### 2.4.3 Connecting the power supply

#### Cable protection

Connect a circuit protection device (F1) rated for at least 1 A (slow) to all base devices.

You may need a higher circuit protection rating depending on the type and connection of the expansion devices (F1).
You can use a shared, adequately sized circuit protection device for the base device and expansion(s) that takes into account the amount - a maximum of 11 and the terminal - with UC, DC, and AC power.

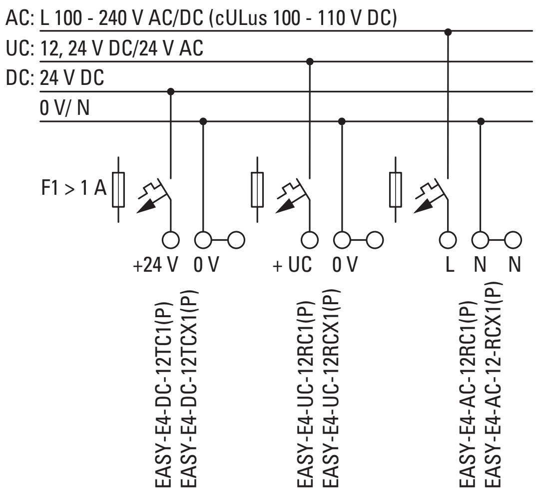



Figure 11: Connecting the power supply for base devices

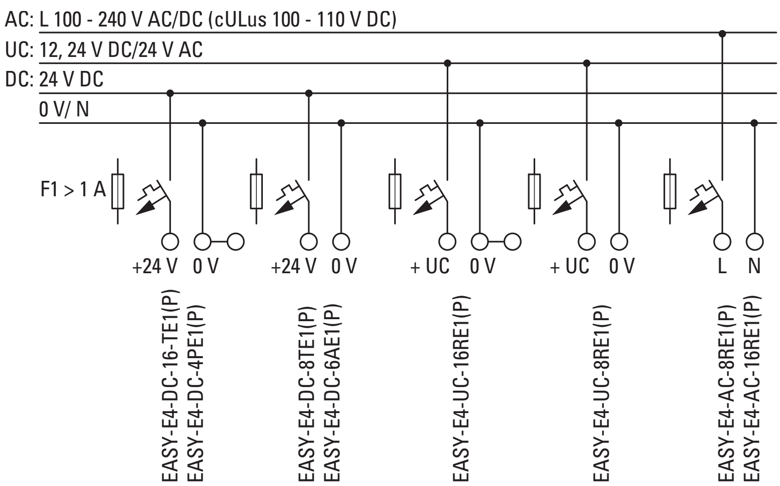



Figure 12: Connecting the power supply for expansions

**System test**

The devices run a system test after the supply voltage has been switched on.

The system test lasts 1 s for the base device. After this time elapses, the device will enter RUN or STOP mode depending on the specific device and configured settings.

> *NOTICE*
> When the basic devices and expansion units are switched on, they behave like a capacitor, so that an inrush current higher than the rated input current is present. Take this inrush current into account when designing the electrical equipment by using slow fuses and suitable switches. Never use reed relay contacts to switch the power supply as these may burn or stick.

You can find the required connection specifications for your device model from the corresponding data sheet, → Section "Technical data", page 699

### 2.4.3.1 Special notes on connecting EASY-E4-AC-... devices

> **DANGER!**
> Connect inputs I1–I8 on AC base devices and I1–I4 on expansion devices in accordance with all applicable VDE, IEC, UL, and CSA safety rules using the same phase conductor that delivers the supply voltage. The device will otherwise not detect the switching level or may be destroyed by overvoltage.
> Inputs I5–I8 on expansion EASY-E4-AC-16RE1(P) can be connected to a different phase.
> Ensure that the L and N conductor are not reversed.

### 2.4.4 Connect digital inputs

The inputs of the easyE4 devices switch electronically.
Once you have connected a contact via an input terminal, you can reuse it as a contact in your circuit diagram as often as you like.

Connect the contacts, for example buttons or switches, to the input terminals of the easyE4 device.

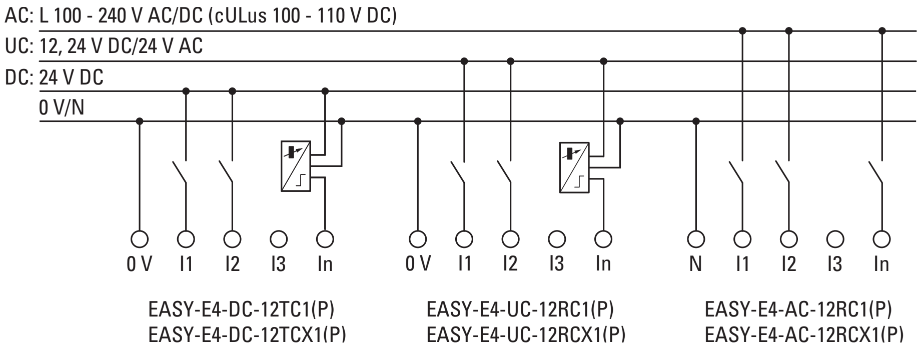



Figure 13: Connecting the digital inputs on base devices

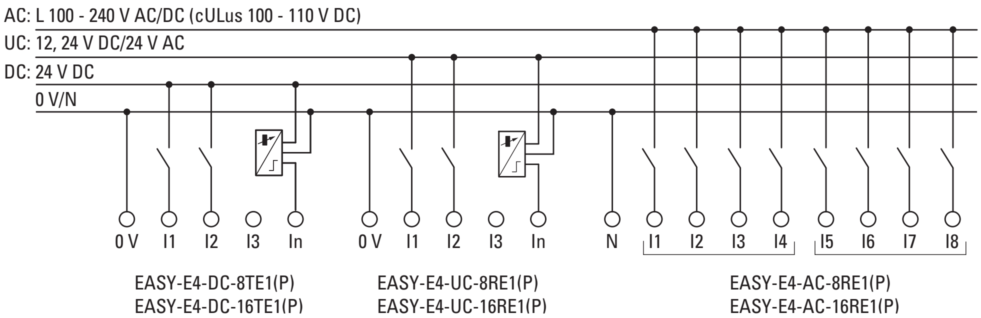



Figure 14: Connecting the digital inputs on expansions

As per the hardware characteristics, base devices have 8 digital inputs available (I1 – I8)
Meanwhile, expansion devices feature 4 (I1 – I4) or 8 (I1 – I8) inputs.

#### Special considerations for EASY-E4-AC-... expansions

**DANGER!**
Connect inputs I1–I4 on AC expansion devices in accordance with all applicable VDE, IEC, UL, and CSA safety rules using the same phase conductor that delivers the supply voltage. Otherwise, the device will not detect the switching level or may be destroyed by overvoltage.

Inputs I5–I8 on expansion EASY-E4-AC-16RE1(P) can be connected to a different phase as I1–I4.
Ensure that the L and N conductor are not reversed.
Adjacent AC devices can be powered with different phases.

Table 4: AC phase assignment

| | | EASY-E4-AC-12RC1(P), EASY-E4-AC-12RC1, EASY-E4-AC-8RE1(P) | EASY-E4-AC-16RE1(P) | |
|---|---|---|---|---|
| $L_{Ue}$ | $N_{Ue}$ | I1-I8 | I1-I4 | I5 - I 8 |
| L1 | | L1 | L1 | L1 |
| L1 | N | L1 | L1 | L2 |
| L1 | | L1 | L1 | L3 |
| L2 | | L2 | L2 | L2 |
| L2 | N | L2 | L2 | L1 |
| L2 | | L2 | L2 | L3 |
| L3 | | L3 | L3 | L3 |
| L3 | N | L3 | L3 | L1 |
| L3 | | L3 | L3 | L2 |

**Example showing how to read the table**

| $L_{Ue}$ | $N_{Ue}$ | I1-I8 | I1-I4 | I5-I8 |
|---|---|---|---|---|
| L1 | | L1 | L1 | L1 |
| L1 | N | L1 | L1 | L2 |
| L1 | | L1 | L1 | L3 |
| L2 | | L2 | L2 | L2 |
| L2 | N | L2 | L2 | L1 |
| L2 | | L2 | L2 | L3 |
| L3 | | L3 | L3 | L3 |
| L3 | N | L3 | L3 | L1 |
| L3 | | L3 | L3 | L2 |

If expansion device EASY-E4-AC-16RE1(P)

is being powered with phase L1, then inputs I1-I4 must also be driven with L1.
Inputs I5-I8 can be driven with the same phase L1, but also with either phase L2 or L3.

#### 2.4.4.1 Connect digital counter inputs

Only possible on base devices.

Base devices with DC and UC voltage come with special counting and measuring functions on inputs I1 to I4.
These functions are connected directly to function blocks.

→ The following applies to EASY-E4-UC-...:
The voltage supplied to the EASY-E4-UC-... must be a DC voltage, since only DC signals will be processed.

You can process the following:

- 4 individual high-speed counter signals (one single counting direction), I1, I2, I3, I4
- 2 incremental counters, I1, I2 and I3, I4
- Frequencies I1, I2, I3, I4

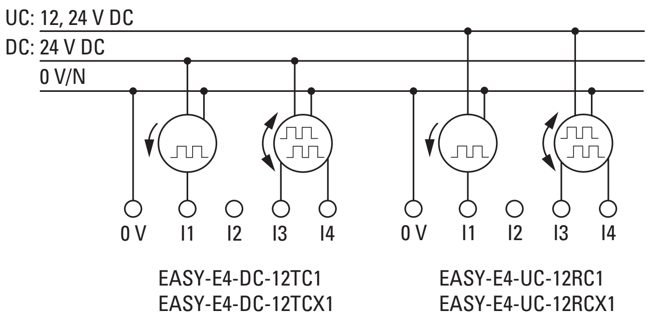



Figure 15: Connect digital counter inputs

→ Input cable length
Strong interference on long cables can result in inputs reaching their switching level. Please do not exceed the maximum cable lengths specified in the technical data for the connected, shielded sensors.

### 2.4.5 Connecting analog inputs

Only possible on base devices.

Base devices with DC and UC voltage can read analog voltages within a range of 0 to 10 V via inputs I5, I6, I7, and I8 on the EASY-E4-... base device.

The signal has a 12 bit resolution, value range 0 - 4 095.

The following applies:

- I5 = IA01
- I6 = IA02
- I7 = IA03
- I8 = IA04

The analog voltage inputs can also be used as digital inputs.

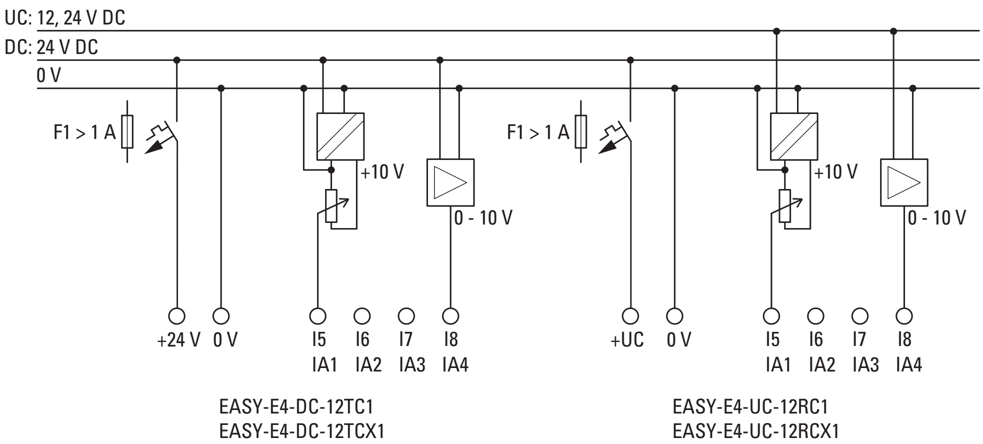



Figure 16: Connecting the analog inputs on base devices

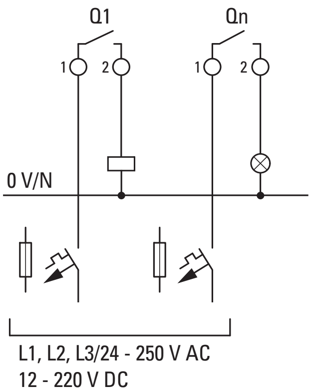
→ SETPOINT encoder:
Use a potentiometer with a resistance ≤ 1 kΩ, e.g., 1 kΩ, 0.25 W.

**DANGER**
Analog signals are more sensitive to interference than digital signals. Consequently, greater care must be taken when laying and connecting the signal lines. The measures described below must be adhered to in order to prevent any deviations in analog values. An incorrect connection can lead to unwanted switching states.

In order to prevent fluctuating analog values, you should take the measures specified for Engineering → Section "Analog Signals", page 40

### 2.4.6 Connecting relay outputs

The EASY-E4-UC-... and EASY-E4-AC-... base and expansion devices feature relay outputs.




| | |
|---|---|
| EASY-E4-UC-12RC1 | EASY-E4-UC-8RE1 |
| EASY-E4-UC-12RCX1 | EASY-E4-UC-16RE1 |
| EASY-E4-AC-12RC1 | EASY-E4-AC-8RE1 |
| EASY-E4-AC-12RCX1 | EASY-E4-AC-16RE1 |

Figure 17: Connecting relay outputs

**DANGER**

Make sure to observe the technical data for the relays.
Do not exceed the upper voltage limit of 250 VAC on a relay contact.
If the voltage exceeds this threshold, flashover may occur at the contact, resulting in damage to the device or a connected load.

### 2.4.7 Connecting transistor outputs

EASY-E4-DC-... devices feature transistor outputs.

A separate power supply feed must be provided for the base device transistor outputs.

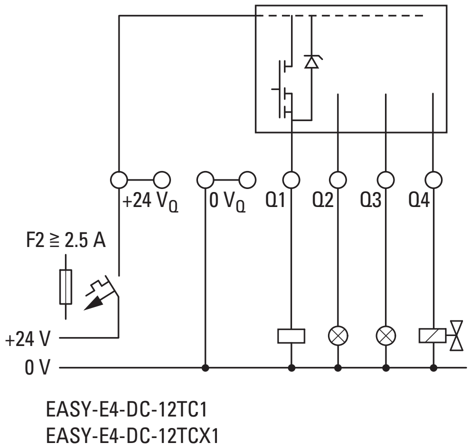



Figure 18: Connecting base device transistor outputs

Transistor outputs on easyE4 expansion devices are powered via the power supply for the expansion device. In other words, transistor outputs have the same potential as the expansion device's inputs.

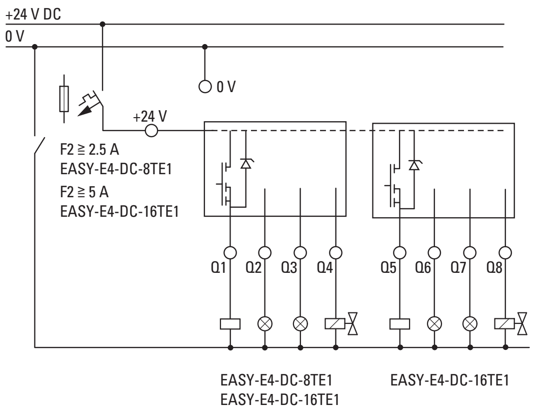



Figure 19: Connecting expansion device transistor outputs

→ Suppressor circuit for transistor outputs on EASY-E4-... devices.

When inductive loads without a suppressor circuit are switched off, overvoltages are produced. Accordingly, use an appropriate suppressor circuit for the transistor outputs in order to handle these peak currents and prevent electronic components from overheating in a worst-case scenario.

Depending on the actual inductive load (I, L):
If the +24 $V_{DC}$ power supply is switched off via a contact in the event of an emergency stop, and if more than one driven output with an inductive load can be switched off, you must provide these inductive loads with a suppressor circuit.

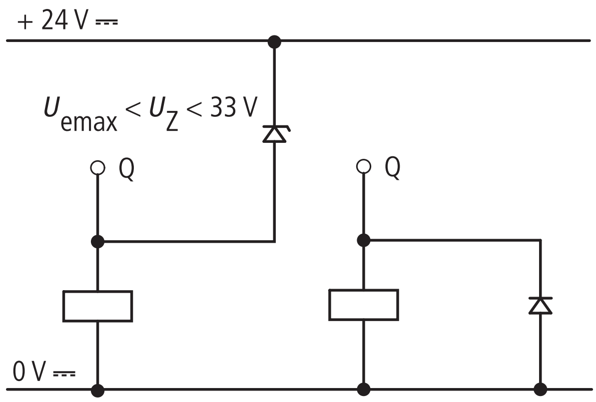


Figure 20: Inductive load with suppressor circuit

#### 2.4.7.1 Transistor output behavior in the event of a short circuit/overload

The following applies to easyE4 devices with transistor outputs:

In the event of a short circuit or overload at a transistor output, this output will switch off and a general fault alarm ID (please refer to fault IDs) will be set to 1. The output will switch back on up to the maximum temperature after a cooling time that depends on the ambient temperature and the current level. If the fault continues, the output will switch off and on until the fault is rectified or the power supply is switched off.

#### 2.4.7.2 Connecting outputs in parallel

Only outputs of the same group (Q1 to Q4 or Q5 to Q8) can be connected in parallel; e.g. Q1 and Q3 or Q5, Q7 and Q8. Parallel switched outputs must be similtaneously energized.

If the outputs are not switched on and off automatically, or if outputs from both groups are connected in parallel, this may result in malfunctions such as those occurring in the case of overloads.

### 2.4.8 Analog I/O expansion device

The analog inputs in expansion EASY-E4-DC-6AE1(P) cannot be used as digital inputs.

The EASY-E4-DC-6AE1(P) features four analog inputs and two analog outputs. You can use easySoft 7 to set the operating mode for each analog input and analog output.

The following options are available:

| Resolution, analog | Resolution, digital | value |
|---|---|---|
| 0 – 10 V | 12 bit | 0 - 4095 |
| 4 – 20 mA | 12 bit | 820 - 4095 |
| 0 – 20 mA | 12 bit | 0 - 4095 |

For all analog inputs, you can configure noise suppression, averaging, and an update rate in easySoft 7.

*Project view*

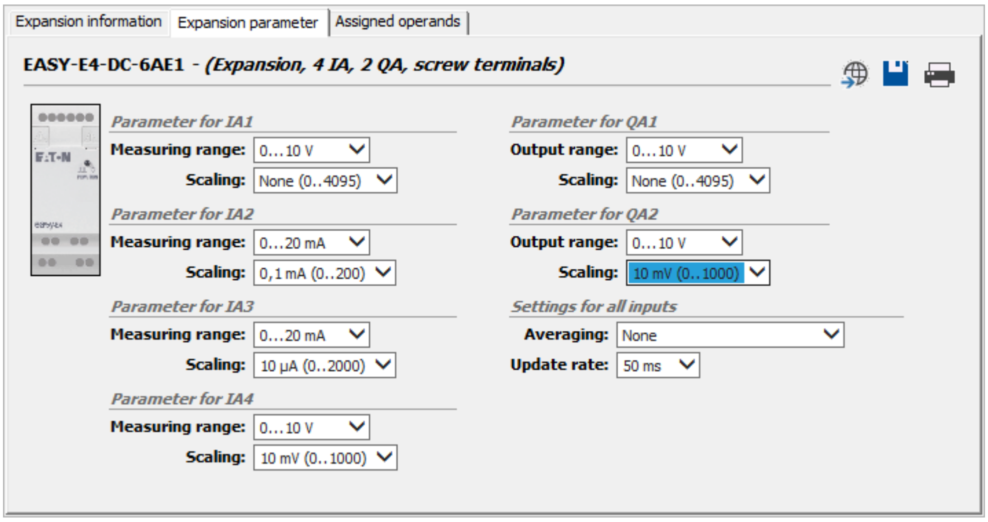



Figure 21: Device parameters tab, using the EASY-E4-DC-6AE1 as an example

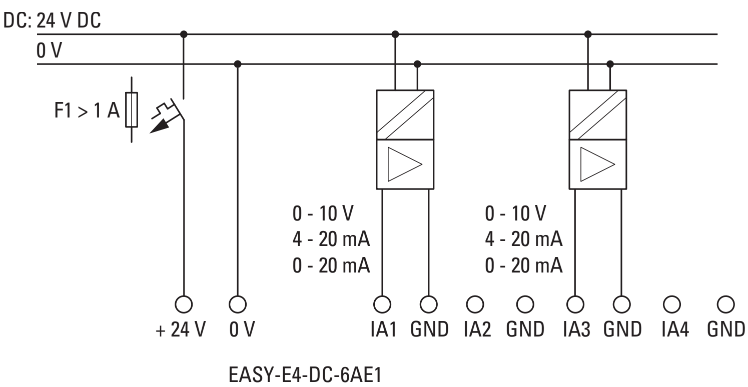



Figure 22: Connecting analog inputs EASY-E4-DC-6AE1(P)

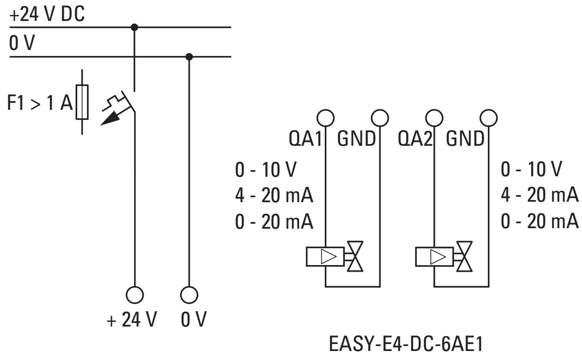



Figure 23: Connecting analog outputs EASY-E4-DC-6AE1(P)

**DANGER**
Analog signals are more sensitive to interference than digital signals, therefore the signal cables should be carefully routed and connected. An incorrect connection can lead to unwanted switching states.

In order to prevent fluctuating analog values, you should take the measures specified for Engineering → Section "Analog Signals", page 40

### 2.4.9 Analog input expansion with temperature measuring

Temperature inputs cannot be used as digital inputs.

The EASY-E4-DC-4PE1(P) analog input expansion features four (4) analog RTD inputs that can be used to integrate Pt100, Pt1000, or Ni1000 temperature sensors.

The Pt100, Pt1000, or Ni1000 inputs are suitable for 2-wire and 3-wire connections. In addition, unshielded or shielded cables with a length of up 30 m can be used for the connection. Finally, averaging for temperature readings can be set up.

When connecting temperature sensors, make sure to use the right configuration depending on whether you are using a 2-wire or 3-wire connection. If the temperature sensors are connected using a 2-wire connection, the corresponding input terminals must be connected to each other, i.e., input terminals 2 and 3 for T1, input terminals 5 and 6 for T2, input terminals 8 and 9 for T3, and input terminals 11 and 12 for T4.

→ When inputs on an EASY-E4-DC-4PE1(P) are not used, all three input terminals must be connected to each other.

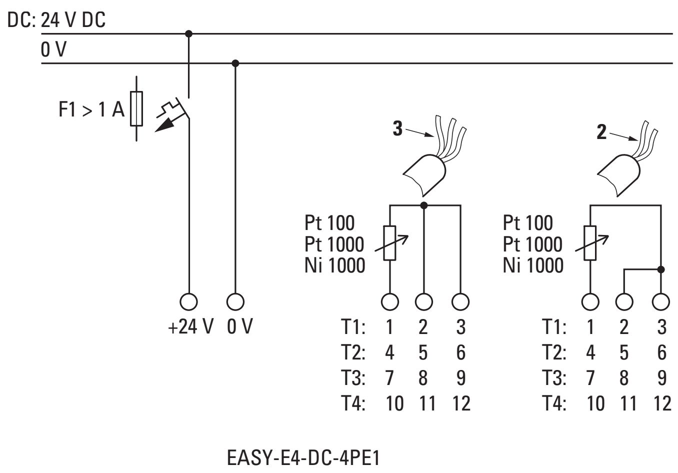



Figure 24: Connecting analog inputs EASY-E4-DC-4PE1(P)

| **DANGER** |
|---|
| Analog signals are more sensitive to interference than digital signals, therefore the signal cables should be carefully routed and connected. An incorrect connection can lead to unwanted switching states. Unshielded signal cables must be routed separately from AC cables. |

In order to prevent fluctuating analog values, you should take the measures specified for Engineering → Section "Analog Signals", page 40

easySoft 7 is required in order to be able to configure the connected RTD sensors.

*Project view*

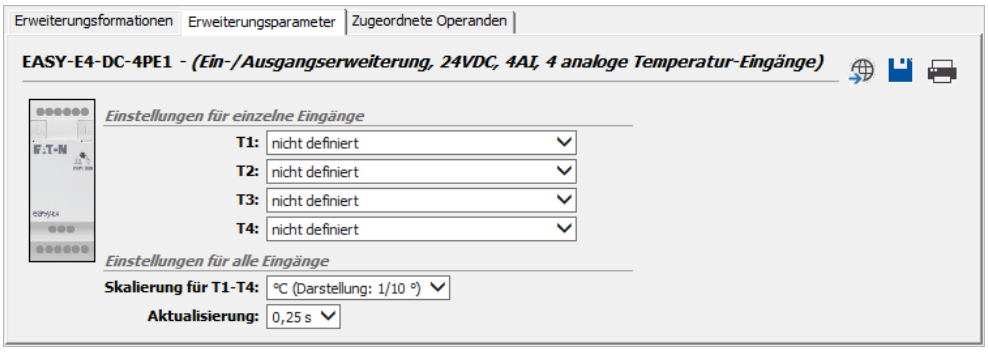



Figure 25: Expansion parameter tab, using the EASY-E4-DC-4PE1 as an example

The temperature sensor connections determine which inputs will be used. Up to four different RTD sensors of type Pt100, Pt1000, or Ni1000 with an individual temperature range can be connected to each EASY-E4-DC-4PE1(P) expansion device.

Inputs that do not have a sensor connected to them will be "undefined."
By default, all inputs will be undefined and accordingly will be switched off.

The temperature ranges for the EASY-E4-DC-4PE1(P) depend on the selected sensor.

| Temperature range | Sensor style | Temperature °C |
|---|---|---|
| 1 | Pt1000 / Pt100 | -100 … +200 (-148 - +392°F) |
| 2 | Pt1000 / Pt100 | -100 … +400 (-148 - +752°F) |
| 3 | Pt1000 / Pt100 | -100 … +800 (-148 - +1472°F) |
| 1 | Ni1000 | -50 … +100 ( -58..+212°F) |
| 2 | Ni1000 | -50 … +250 ( -58..+482°F) |

Values will be represented as a signed decimal with the following resolution (with the specifics depending on the selected format):

| Representation Sensor model | Temperature value in °C | Indicated value at selected representation | | | | |
|---|---|---|---|---|---|---|
| | | Degrees Celsius °C | | Degrees Fahrenheit °F | | Nonlinear value |
| | | 1/10 | 1 | 1/10 | 1 | |
| Pt100, Pt1000 | -100 up to +200 | -1000 up to 2000 | -100 up to +200 | -1480 up to +3920 | -148 up to +392 | 0 - 4095 |
| Pt100, Pt1000 | -100 up to +400 | -1000 up to 4000 | -100 up to +400 | -1480 up to +7520 | -148 up to +752 | 0 – 4095 |
| Pt100, Pt1000 | -100 up to +800 | -1000 up to 8000 | -100 up to +800 | -1480 up to +14720 | -148 to +1472 | 0 – 4095 |
| Ni1000 | -50 up to +100 | -500 up to 1000 | -50 up to +100 | -580 up to +2120 | -148 up to +212 | 0 – 4095 |
| Ni1000 | -50 up to +250 | -500 up to 2500 | -50 up to +250 | -580 up to +4820 | -148 up to +482 | 0 – 4095 |

The selected scaling and update settings will apply to all the temperature inputs in the corresponding module.

The scaling and the unit (Celsius, Fahrenheit) can be selected for inputs T1 through T4. If no scaling is specified, the corresponding raw value will be given with a 12-bit resolution (dimensionless, 0 – 4095).

Reading scaling: The scaling

Update - Sampling time for all inputs being used:

- 250 ms (no averaging)
- 1 s (averaging over 4 measuring cycles)
- 2.5 s (averaging over 10 measuring cycles)
- 10 s (averaging over 40 measuring cycles)

As soon as the device is switched on, the temperature will be measured and transmitted by all active sensors. However, the reading will not be averaged until after the set sampling time.

The expansion module features a DIAG output for function monitoring and diagnostic purposes. This means that each temperature input can be individually mapped to an operand within a range of ID25 to ID96.

| Designation | Event |
|---|---|
| DIAG | General diagnostic indicating that a diagnostic event is present |
| DIAG 1 | Configured measuring range exceeded at at least one temperature input, or connection cable discontinuity |
| DIAG 2 | Configured measuring range fallen below at at least one temperature input, or a short-circuit has occurred |
| T1 | <Mapped operand> |
| T2 | <Mapped operand> |
| T3 | <Mapped operand> |
| T4 | <Mapped operand> |

The temperature module will write to the easyE4 base device's diagnostic buffer.

### 2.4.10 Terminal configurations for individual devices

#### Base devices

| EASY-E4-UC-12RC1(P), EASY-E4-UC-12RCX1(P) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply | +UC | 0V | 0V | | | | | | | | |
| Input | | | | I1 | I2 | I3 | I4 | I5 | I6 | I7 | I8 |
| Output | | | | Q1/1 | Q1/2 | Q2/1 | Q2/2 | Q3/1 | Q3/2 | Q4/1 | Q4/2 |

| EASY-E4-DC-12TC1(P), EASY-E4-DC-12TCX1(P) | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply | +24V | 0V | 0V | | | | | | | | | |
| Input | | | | | I1 | I2 | I3 | I4 | I5 | I6 | I7 | I8 |
| Output power supply | +24VQ | +24VQ | 0V | 0V | | | | | | | | |
| Output | | | | | Q1 | Q2 | Q3 | Q4 | | | | |

| EASY-E4-AC-12RC1(P), EASY-E4-AC-12RCX1(P) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply | L | N | N | | | | | | | | |
| Input | | | | I1 | I2 | I3 | I4 | I5 | I6 | I7 | I8 |
| Output | | | | Q1/1 | Q1/2 | Q2/1 | Q2/2 | Q3/1 | Q3/2 | Q4/1 | Q4/2 |

#### Expansions

UC input expansions with relay outputs

| EASY-E4-UC-16RE1(P) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply | +UC | 0V | 0V | | | | | | | | |
| Input | | | | I1 | I2 | I3 | I4 | I5 | I6 | I7 | I8 |
| Output | | | | Q1/1 | Q1/2 | Q2/1 | Q2/2 | Q3/1 | Q3/2 | Q4/1 | Q4/2 |
| Output | | | | Q5/1 | Q5/2 | Q6/1 | Q6/2 | Q7/1 | Q7/2 | Q8/1 | Q8/2 |

| EASY-E4-UC-8RE1(P) | | | | | | |
|---|---|---|---|---|---|---|
| Power Supply | +UC | 0V | | | | |
| Input | | | I1 | I2 | I3 | I4 |
| Output | | | Q1/1 | Q1/2 | Q2/1 | Q2/2 |
| Output | | | Q3/1 | Q3/2 | Q4/1 | Q4/2 |

DC input expansions with transistor outputs

| EASY-E4-DC-8TE1(P) | | | | | | |
|---|---|---|---|---|---|---|
| Power Supply | +24V | 0V | | | | |
| Input | | | I1 | I2 | I3 | I4 |
| Output | | | Q1 | Q2 | Q3 | Q4 |

| EASY-E4-DC-16TE1(P) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply | +24V | 0V | 0V | | | | | | | | |
| Input | | | | I1 | I2 | I3 | I4 | I5 | I6 | I7 | I8 |
| Output | | | | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q7 | Q8 |

AC input expansions with relay outputs

| EASY-E4-AC-8RE1(P) | | | | | | |
|---|---|---|---|---|---|---|
| Power Supply | L | N | | | | |
| Input | | | I1 | I2 | I3 | I4 |
| Output | | | Q1/1 | Q1/2 | Q2/1 | Q2/2 |
| Output | | | Q5/1 | Q5/2 | Q6/1 | Q6/2 |

| EASY-E4-AC-16RE1(P) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply | L | N | | | | | | | | | |
| Input | | | I1 | I2 | I3 | I4 | | I5 | I6 | I7 | I8 |
| Output | | | Q1/1 | Q1/2 | Q2/1 | Q2/2 | | Q3/1 | Q3/2 | Q4/1 | Q4/2 |
| Output | | | Q5/1 | Q5/2 | Q6/1 | Q6/2 | | Q7/1 | Q7/2 | Q8/1 | Q8/2 |

Analog input expansion

| EASY-E4-DC-6AE1(P) | | | | | | |
|---|---|---|---|---|---|---|
| Power Supply | +24V | 0V | | | | |
| Input | | | IA1 | GND | IA2 | GND |
| Input | | | IA3 | GND | IA4 | GND |
| Output | | | QA1 | GND | QA2 | GND |

Analog input expansion with temperature measuring

| EASY-E4-DC-4PE1(P) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Power Supply | +24V | 0V | 0V | | | | | | |
| Input | | | | IA1-1 | IA1-2 | IA1-3 | | | |
| Input | | | | IA2-4 | IA2-5 | IA2-6 | | | |
| Input | | | | IA3-7 | IA3-8 | IA3-9 | IA4-10 | IA4-11 | IA4-12 |

# 2.5 External connections

With their ports, the base devices make it possible to connect a variety of peripheral devices and components.

## 2.5.1 External connection layouts

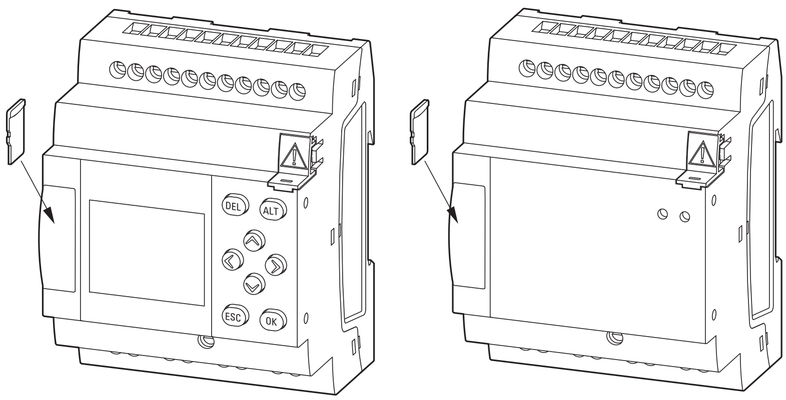



Figure 26: Slot for microSD

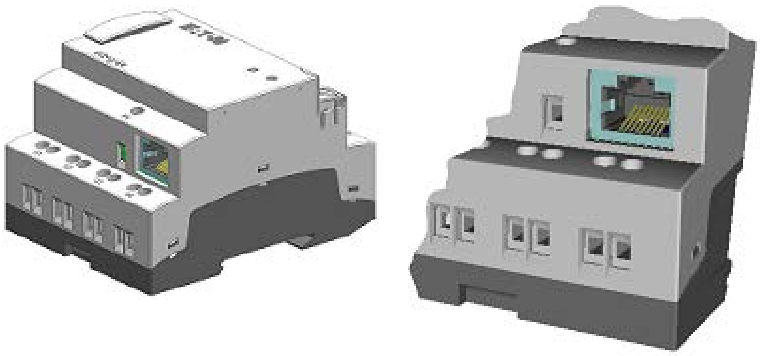


Figure 27: Ethernet port on base device

### 2.5.2 Memory card

The slot for the microSD is located at the front of the base device.

#### Inserting a microSD card

Memory cards cannot be inserted the wrong way around.
Do not use force when inserting the card.

- Pull out the slot.
- Push the microSD card into the slot until you feel it lock into place.
- Close the slot.

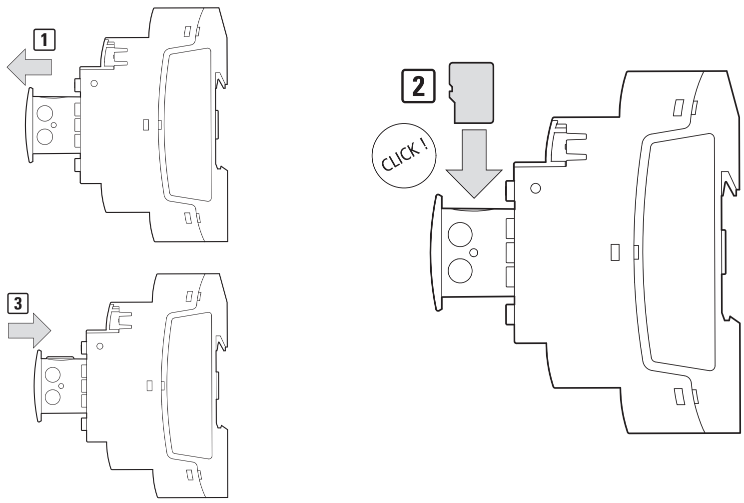



Figure 28: Inserting a memory card

### Remove microSD

- Pull out the slot.
- Push the microSD card into the slot.

The memory card will be released and come out a bit

- Remove the memory card.
- Store the microSD in its case in order to protect it.
- Close the slot

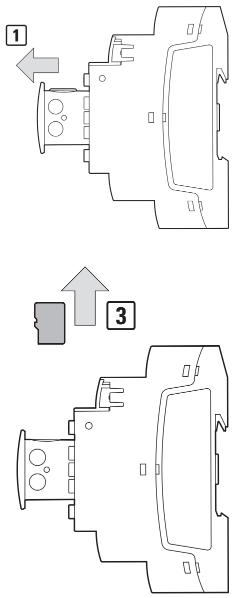



Figure 29: Removing the memory card

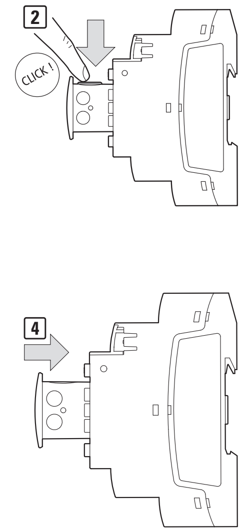



Figure 30: Removing the memory card

### 2.5.3 Ethernet

Every easyE4 base device features an Ethernet port.
This Ethernet port is a Cat 5 port.
Make sure to use compatible standard RJ45 Ethernet cables only.

The Ethernet port on the base device serves as a communication interface.

The Ethernet controllers support transfer rates of 10 Mbit/s and 100 Mbit/s.

Figure 31: RJ-45 socket, 8-pole

→ If you integrate the EASY-E4-... into an Ethernet network, you will need to connect the functional earth to the corresponding terminal.

To commission the communications between the EASY-E4-... control relay and the device to which the Ethernet cable is connected, follow the description for the connected device.

New easyE4 base devices will come with the Auto IP setting configured by default. In order to configure the settings differently on the EASY-E4-...-12...C1(P), use the menu structure and go to *System Options\Ethernet* → Section "Ethernet", page 568

### 2.5.3.1 Connecting the Ethernet cable

EASY-E4-...-12...C1(P) and EASY-E4-...-12...CX1(P) devices are designed for use with both screw terminals and push in terminals.

For more information on these terminal types, please refer to → Section "Connection terminals", page 54

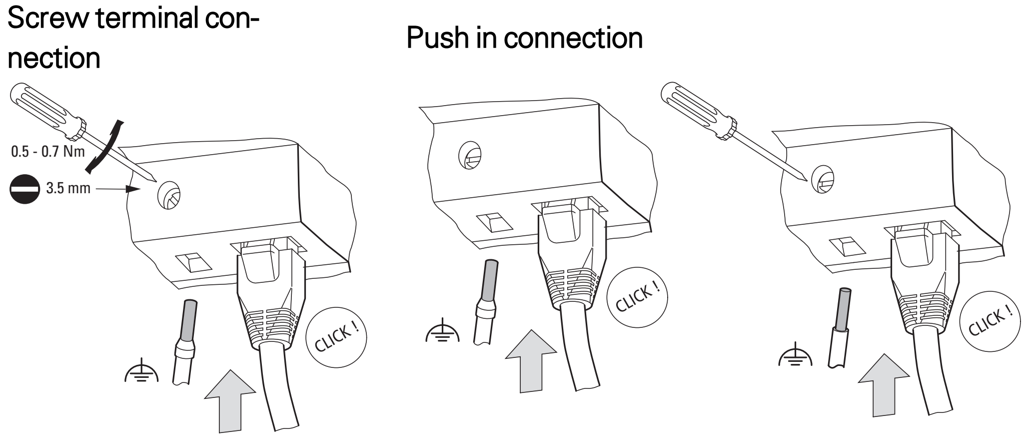
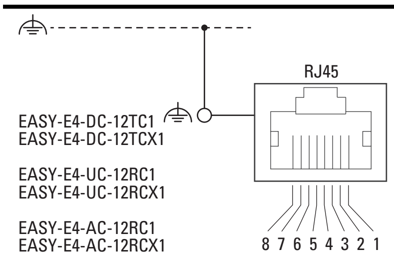



Figure 32: Connecting the Ethernet cable




| PIN | Ethernet 10/100 MBit |
|---|---|
| 1 | Tx +— |
| 2 | Tx —— |
| 3 | Rx +Rx – |
| 4 | — |
| 5 | — |
| 6 | Rx – |
| 7 | — |
| 8 | — |

- Connect the functional earth
- Plug in the Ethernet cable

### Removing the Ethernet cable

With screw terminal connection

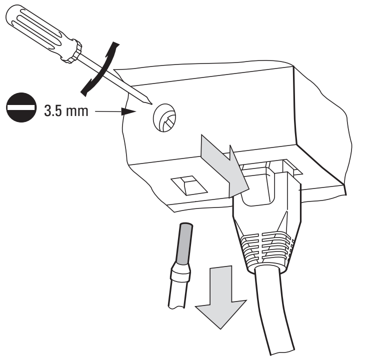



Figure 33: Removing the Ethernet cable

With push in connection

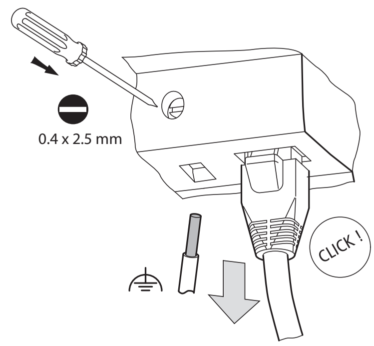



Figure 34: Removing the Ethernet cable

## 2.6 Programming software license

The programming software (version 7 and higher) is available for download.

Please note that easyE4 devices can only be programmed with easySoft 7.

The easySoft 7 programming software program is available for free. However, you will need to buy a software license in order to be able to use all of the software's functions.

→ You can order an easySoft 7 programming software license through your supplier or through the EATON online catalog EASYSOFT-SWLIC, catalog no. 197226.

http://www.eaton.eu/ecat

Once you purchase a software license, you will receive a license product certificate that you can then use to request a license key online. This license key can be used to unlock all of the software's functions.

Installation requirements

- An easySoft 7 version
- A PC that meets all system requirements and for which the person installing the software has administrator privileges
- A 24-digit license key

→ If a valid license key is not entered during installation, the software will be installed in demo mode.
This means that the software will be fully installed, but with the following limitations:

- You will not be able to load any programs onto a connected device (no online function)
- There will not be any card manager functions available for the microSD memory card

However, you will still be able to simulate programs.
You can always add a license key later on.

### 2.6.1 System requirements

**Hardware**

- Recommended minimum resolution of 1280 x 1024 pixels
- Min. 250 MB of free space on the hard drive

**Software**

One of the following operating systems:

- Windows 7 (32 + 64 Bit) >= SP1,
- Windows 8/8.1 (32 + 64 Bit) or
- Windows 10 (32 + 64 Bit)

### 2.6.2 Licensing

When you purchase an EASYSOFT-SWLIC (catalog no. 197226), you will be purchasing a license product certificate for easySoft 7.
This license product certificate will include a 36-digit certificate number that you can use to request a 24-digit license key online.

→ During the installation process, you will be asked for the 24-digit license key for your easySoft 7. If you do not enter a license key, the program will be installed in demo mode.
You can add a license key at a later point if necessary.


Figure 35: license product certificate

### 2.6.2.1 Getting a license key

To get a license key with your license product certificate, follow the instructions at:

http://www.eaton-automation.com/license


Figure 36: Input screen for the license product certificate No.

Once you enter the 36-digit certificate number from your license product certificate, a dialog box will appear. For your own security, enter the owner information into this dialog box.

Once you enter all your information, a 24-digit license key will be sent to the e-mail address you provided.

The e-mail will contain the following information:

- License type:SW-EASYSOFT
- License product certificate number: 7-digit number for your certificate
- License key: Automatically generated 24-digit code
- Information regarding the owner's registration

→ The 24-digit license key is requested during the installation process.

### 2.6.3 Adding a license key later on

If you installed the demo version of easySoft 7, you can add a valid license key later on in order to unlock the full version.

▶ Go to easySoft 7 the *? menu* and click on License.

A dialog box for entering the license key will appear.

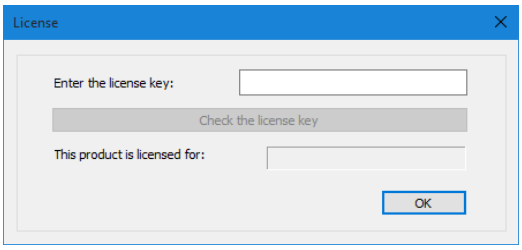



Figure 37: License dialog box

▶ Now enter the 24-digit license key
that you received by e-mail.

### 2.6.4 Software updates and hardware changes

Once you have licensed the programming software, you can download the latest easySoft 7 version from the Eaton Download Center - Software and install it – the license information will remain.

▶ If you change hardware, use your license key and redeem it again.

easySoft 7 can check whether there are any updates for the version installed. This requires for the PC to have an active Internet connection.

*Menu?*

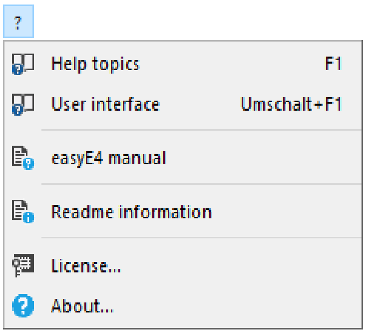



Figure 38: Options in ? menu

### 2.6.5 Installation instructions

Before starting with the installation, close all open applications.

To install easySoft 7, you will need to have local admin privileges for your system.

**Download**

- Download the full version of the easySoft 7 program from the Software Download Center.

Download Center - Software
http://www.eaton.eu/software

- Select the "Software" category, then the easySoft 7 software, then the product version, and finally your language.
- Click on the product version you want in order to download it.
- Save the installation package file on your PC.

**Installation**

→ During the installation process, you will be asked for the 24-digit license key for your easySoft 7. If you do not enter a license key, the program will be installed in demo mode.
You can add a license key later on if necessary.

- Follow the on screen instructions of the installation package.


Figure 39: Step 1

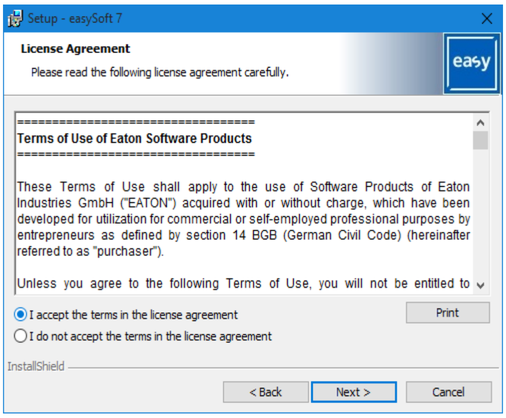



Figure 40: Step 2 License agreement

You can also print out the terms of use in their entirety.

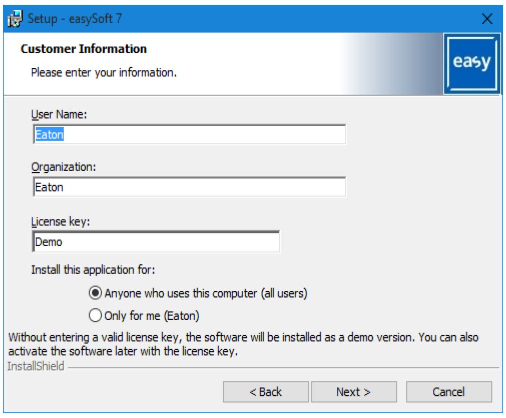



Figure 41: Step 3 License key

To install the full version of the software, enter your 24-digit license key here.

→ If a valid license key is not entered during installation, the software will be installed in demo mode.
You can add a license key later on – please refer to → Section "Adding a license key later on", page 81.

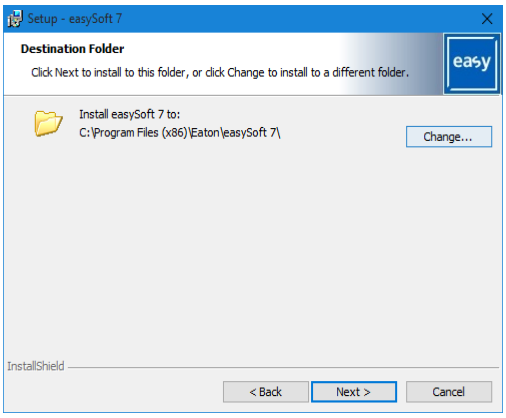



Figure 42: Step 4 Destination folder

Shows the path where the program files will be stored.

You can click on Browse... to set a different storage location where you want the easySoft 7 programming software program to be installed.

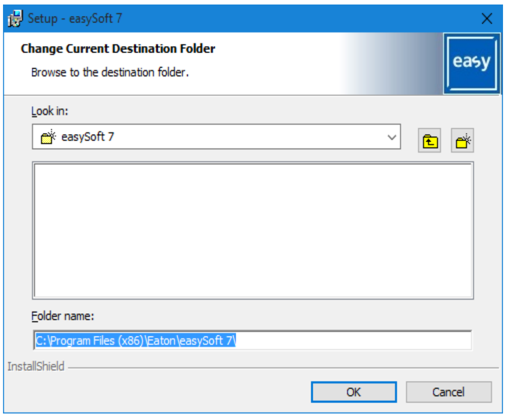



Figure 43: Step 4.1 Changing the destination folder

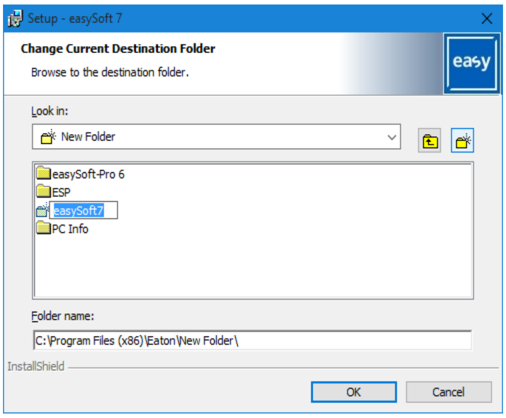



Figure 44: Step 4.2 Creating your own destination folder

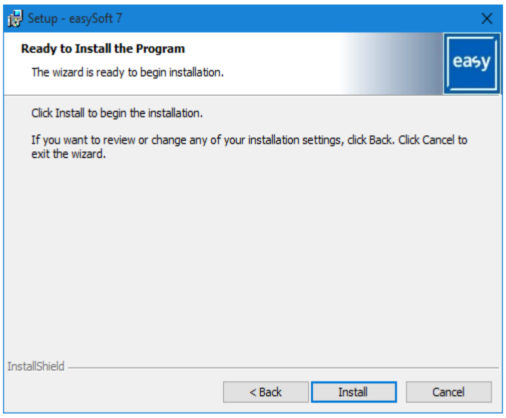



Figure 45: Step 6 Starting the installation

A confirmation prompt will appear.
The installation will start as soon as you confirm this prompt.

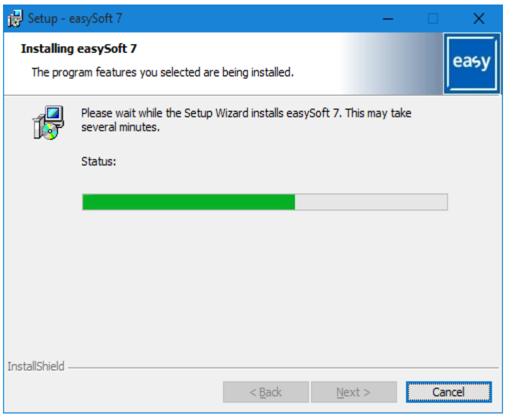



Figure 46: Step 7 Progress display

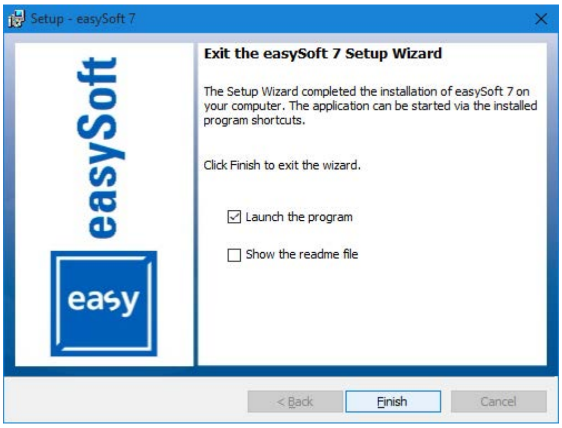



Figure 47: Step 8 Finishing

The easySoft 7 icon will be added to your Desktop during the installation process.

▶ Click on the easySoft 7 icon to open easySoft 7.

or



Figure 48: easySoft 7 icon depending on the screen resolution

# 3. Commissioning

**CAUTION**
**SHORT-CIRCUIT HAZARD**
If the device is or has been exposed to environmental fluctuations (ambient temperature, air humidity), condensation may form on or inside it. As long as this condensation is present, there will be a short-circuit hazard. Do not switch on the device when it has condensation in or on it.
If the device has condensation in or on it, or if the panel has been exposed to environmental fluctuations, let the panel settle into the existing ambient temperature before switching it on.

The easyE4 devices with/without display and operator functions can be commissioned. However, a display and operating facility is required in order to follow all explanations in this chapter.

The following applies to devices without a display and controls: You can use easySoft 7 for the display and controls or use a remote display. To this end, the control relay offers a point-to-point Ethernet connection or a connection in a network via easySoft 7.

## 3.1 Initial commissioning

Carry out the following steps once:

- ▶ Configure the device's system settings as necessary, including the menu language.
  Please refer to→ Section "Switch languages", page 570
- ▶ Install the required software package easySoft 7.
- ▶ Transfer program to the easyE4 device

## 3.2 Daily operation

Once the easyE4 has been initially commissioned, it will run whenever it is connected to the supply voltage.

In other words, it does not have to be separately switched on and off.

→ Reducing the level of brightness will increase the display backlight's lifespan.
The corresponding setting can be adjusted in the operating system.

→ Follow the instructions in the following section if your base device until will not boot up and/or if an error message appears:→ Section "Faults ", page 681

# 3.3 Switch on

Before switching the device on, check the power supplies, inputs, outputs, and any expansion devices and Ethernet connections to make sure that they are properly connected.

### 3.3.1 Startup behavior of easyE4 control relays with LED indicators

If there is no program, the control relay will start in STOP mode.

These devices without a display and controls feature 2 LEDs that indicate the state of the Ethernet port and the device status.

If there is an executable program on the easyE4 control relays, the device will start in RUN mode.

→ In addition to having a valid program on the control relay, please make sure that there are no peripheral faults that will lead to STOP mode.

Device models without a display feature LED indicators in the front:

- 1. POW/RUN
- 2. ETHERNET (base devices only)

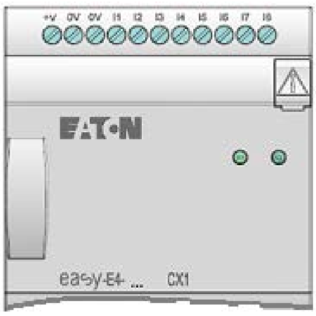



Figure 49: LED status indication

**LED POW/RUN base device**

The POW/RUN LED indicates the state of the POW power supply as well as the STOP or RUN state.

| | |
|---|---|
| Off | Malfunction or no supply voltage |
| Green, continuous light | Supply voltage OK, RUN mode |
| Green,<br>Flashing, 1 Hz | Supply voltage OK, STOP mode |
| Green,<br>Flashing, 4 Hz | Fault at one of the expansions,<br>between the easyE4 device and the EASY-E4-CONNECT1 connector |

### LED ETHERNET/NET (base device only)

| | |
|---|---|
| Off | No Ethernet cable connected; supply voltage off<br>The port is not enabled; the easyE4 device does not have an IP address |
| Yellow, continuous light | Ethernet cable connected |
| Green, continuous light | There is an IP address, but the NET has not been configured |
| Red, continuous light | Ethernet conflict or error, e.g.: duplicate IP address, address collision |
| Green, flashing,<br>2 flashes, pause, etc. | NET data flow working; one or more NET stations missing |
| Green, flashing,<br>1 flash, pause, etc. | NET data flow working; all NET stations working |

### LED POW/RUN status expansion unit

| | |
|---|---|
| Off | Malfunction or no supply voltage |
| Green, continuous light | Supply voltage OK, address assigned, expansion bus working correctly |
| Green,<br>Flashing, 1 Hz | Supply voltage OK, no data exchange with base device |
| Green,<br>Flashing, 3 Hz | Supply voltage OK, no data exchange with base device,<br>diagnostic bit will be set, device not working |
| Green,<br>Flashing, 10 Hz | Device waiting for firmware update |
| Green,<br>Flashing, 0.5 Hz | Firmware update in progress |

### 3.3.2 Startup behavior of easyE4 control relays with a display and keypad

If there is no program, the control relay will start in STOP mode.

All the information on the display will be shown in English if the device is configured with its factory settings.

If there is an executable program on the easyE4 control relays, the device will start in RUN mode.

In addition to having a valid program on the control relay, please make sure that there are no peripheral faults that will lead to STOP mode.

easyE4 base device with integrated display

- If there is no splash screen on the memory card
  after being switched on, the easyE4 base device will show the Eaton logo and then the status display. This status display provides information on the device's status.
- If there is a splash screen on the memory card
  after being switched on, the easyE4 device will show the splash screen and then the status display. This status display provides information on the device's status.

If there is no executable program on the easyE4 control relays, the device will start in STOP mode.

All the information on the display will be shown in English if the device is configured with its factory settings. Once the device is ready for operation, the status display will appear.

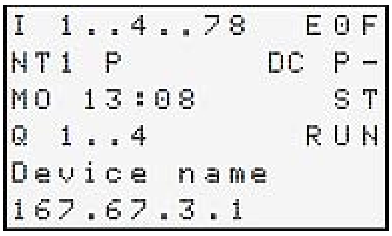



Figure 50: Example of status display on display

### 3.3.2.1 Changing the menu language

To change the menu language on the device, follow the steps below:

- ▶ Press the OK button.

The main menu will appear.

*Main menu*

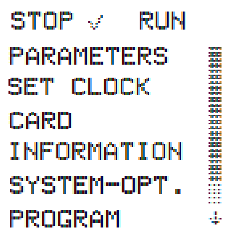



Figure 51: Main menu in English

- ▶ Use the ∧ ∨ cursor buttons to scroll to the SYSTEM OPTIONS menu option.
- ▶ Press the OK button.

The SYSTEM OPTIONS menu will appear.

*Main menu\System Options\Menu Language*

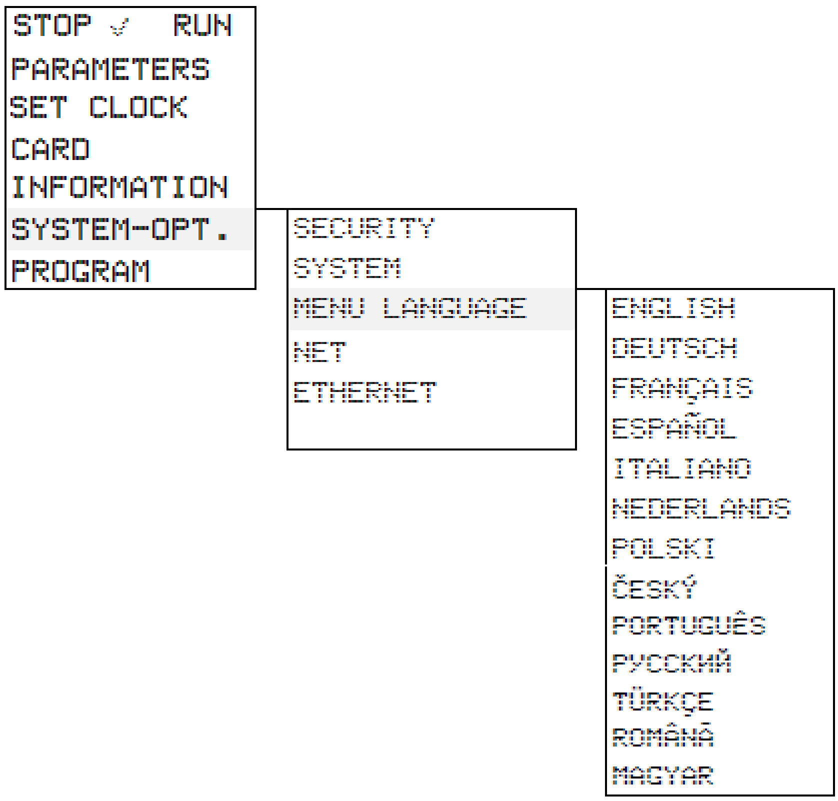



Figure 52: Menu path in English

- Use the ⮝ ⮟ cursor buttons to scroll to the MENU LANGUAGE menu option.
- Press the OK button.
- Use the cursor buttons ⮝ ⮟ to scroll to the language you want.
- Confirm with the OK button.
- Exit the menu with the ESC button.

The display will be switched to the language you selected.

### 3.3.3 Startup behavior of base devices with connected expansion devices

Make sure that all required expansion devices are connected to the expansion bus and to the base device.

- As far as possible, switch all easyE4 devices on at the same time.
- Check whether the program you want is found on the base device (display or easySoft 7).
- If there is no program on the base device, load the program you want (with the memory card or easySoft 7) onto the base device.
- Start the base device in RUN mode.
- Find out what the operating state of the base device and the expansions is.

→ All expansion devices must be selected in the program. The expansion devices must be connected in the same order in the program and in the physical block.

If a device is missing, or if a device different from the one in the program is being used, the easyE4 base device will remain in STOP mode. The easyE4 base device will also do this if you install more devices than the ones found in the program.

**DANGER**
If you have already integrated devices into a system, secure any parts of the system connected to the working area to prevent access and ensure that no one can be injured if, for example,
motors start up unexpectedly.

### 3.3.4 Status display on easyE4 control relays with display and keypad

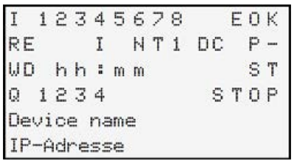
After being switched on, the easyE4 base device will start with the status display after the boot logo.
The status display has six lines, with each one containing 16 characters.

Press the Alt button to switch between displays.

- The first time you press ALT, the time will be replaced by the date.
- Pressing the ALT button again will switch to display 2

**Row** — **Status indicator 1** — **Status indicator 2**

1
2
3
4
5
6



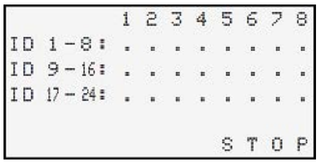




Figure 53: Start displays for easyE4 base device in English

| Status indicator 1 | | |
|---|---|---|
| Line 1 | The Ethernet status for the base device without LED indicators will be shown for diagnostic purposes | |
| I........ | | Inputs; the number will be shown during activity (1, 2, 3,..,8) |
| | EOF | The Ethernet port is not enabled;<br>no Ethernet cable connected; supply voltage off<br>The port is not enabled; the easyE4 device does not have an IP address |
| | ECN | Ethernet cable connected |
| | EOK | There is an Ethernet IP address, but the NET has not been configured |
| | ENW | NET data flow working; all NET stations working |
| | ENM | NET data flow working; one or more NET stations missing |
| | EER | Ethernet conflict or error, e.g.: duplicate IP address, address collision |

| | | |
|---|---|---|
| Line 2 | Settings in current program | |
| | RE | Retention active |
| | I | Debounce active |
| | NT | NET stations with NET ID (1 this case) |
| | DC | Used to display the type of power used by the base device (AC or DC) |
| | P | P Buttons inactive (-) or active (+) |

| | | |
|---|---|---|
| Line 3 | Current device setting | |
| | WD | Weekday |
| | hh:mm | Device time |
| 1xALT | DD-MM-YYY | Displays the device date with the configured format |
| | ST | Configured startup behavior for the device;<br>nothing displayed – automatic starting is possible |

| | | |
|---|---|---|
| Line 4 | | |
| | Q | Outputs; the number will be shown during activity (1, 2, 3, etc.) |
| | RUN/STOP | Current device operating mode |

| | |
|---|---|
| Line 5 | The device's MAC address or device name; displayed only if a name has been assigned |
| Line 6 | IP address; displayed only if an address has been assigned |

| Status indicator 2 | |
|---|---|
| | Displays set diagnostic bits ID1 through ID24: state display with "0" and "1" for each bit |
| Line 1 | Bit number for each block |
| Line 2 | ID 1 ... ID 8: |
| Line 3 | ID 9 ... ID 16 |
| Line 4 | ID 17 ... ID 24 |
| Line 5 | Free |
| Line 6 | Current device operating mode |

▶ Press the ALT button.

Shows additional indicators.

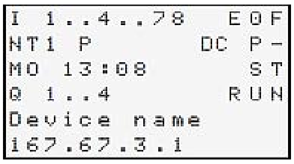


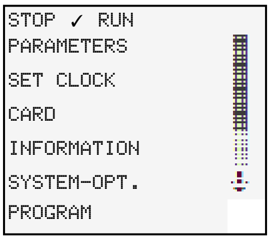

Figure 54: Example of status display on display

You can use the main menu from the status display in order to access the individual submenus.

▶ Press the OK button.

The main menu will appear.

Table 5: *Main menu*




## See also

→ chapter "3 Operation", page 127

### 3.3.5 Commissioning the Ethernet network

If you only want to communicate with one single easyE4, use an Ethernet cable to connect the easyE4 Ethernet port to your computer – please refer to → "Connecting the Ethernet cable", page 76

easySoft 7 communications make it possible to search for connected easyE4 that are switched on and establish communications with them.

#### 3.3.5.1 Network operation

Install the Ethernet network as required by your network architecture (switch, router, firewall, VPN, etc.)

If you want to run the easyE4 with other devices on the Ethernet network and communicate through the Internet, you will need to incorporate network security measures outside of the easyE4.

→ Make sure that the network area in which you run the easyE4 devices is secure.
You can do this by using VPN connections or other network security measures such as a firewall, a WLAN without Internet access, etc.

**WARNING**
Make sure that it is absolutely impossible to have unauthorized access to your easyE4 devices through a network. Unauthorized access may result in injury and/or property damage.

Eaton recommends implementing measures for protecting against cyberattacks.

→ Eaton cyber security

http://www.eaton.com/cybersecurity

**See also**

→ Section "Establish Ethernet connection", page 158

### 3.3.6 Remote operation

If you want to put the easyE4 device into operation without being present at the machine or system, make sure that you always know what exactly will happen when you do so.
Make absolutely sure that remote operation will not endanger anyone.

**See also**

→ Section "Setting up a web server", page 646
→ Section "Modbus TCP", page 632
→ Section "Setting up a NET", page 625

## 3.4 Overview of switch-on behavior

The following figure shows what happens when the device is turned on.

- RUN start
- Card start

As soon as the easyE4 device starts, the options will be read.

The easyE4 basic device will check whether a microSD has been inserted and whether there is a starting program on the microSD. The device will then switch to RUN or STOP mode depending on these parameters.

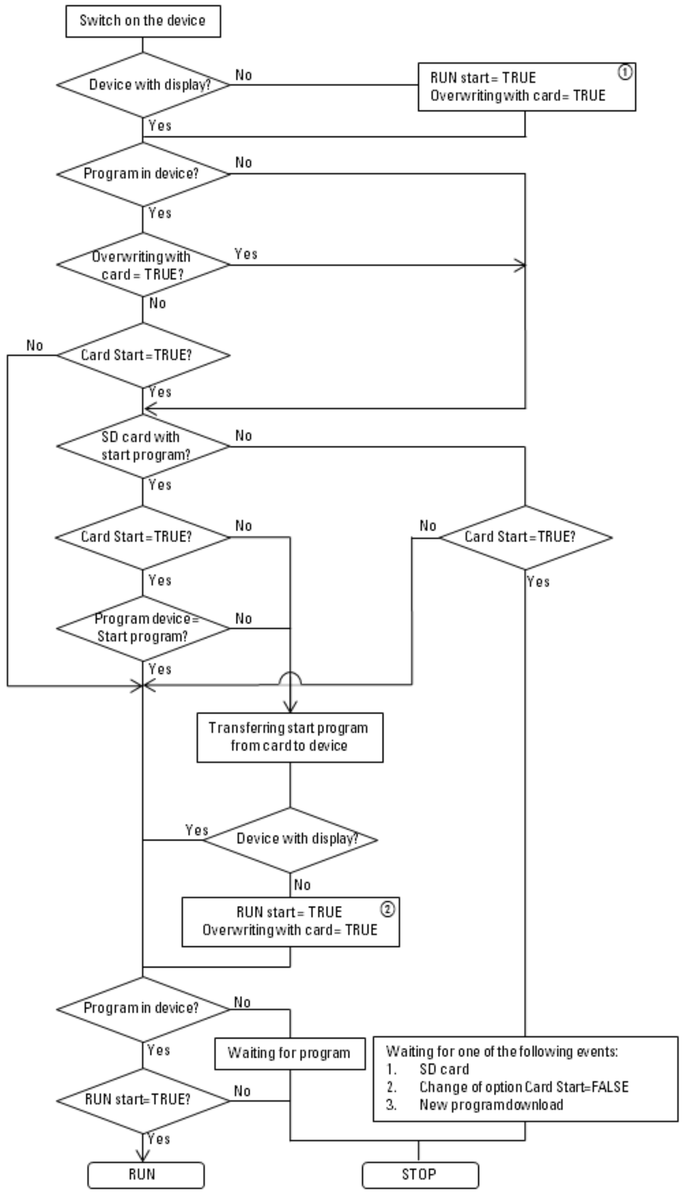



Figure 55: Startup procedure with device initialization

① RUN start: The device should be able to start without easySoft 7
Allow overwriting via card: If an microSD card with a starting program has been inserted, the device should load from the microSD card

② Options set again, since they could be overwritten by the loaded program

## 3.5 Establishing an Ethernet connection and transferring a program

There is a connection via Ethernet that can be used to access the easyE4 base device for programming purposes.

**Physical connection**

Ethernet uses point-to-point connections, meaning that whenever more than two devices are connected, there needs to be a switch with a port for each device. It is recommended to use appropriate industrial switches that have been developed for use in industrial environments. You can use any standard switch and Ethernet cable with RJ45 connectors.
You can also use an Ethernet connection to program the individual devices.

**Basic information on assigning IP addresses**

Internet Protocol (IP) version 4 (IPv4) addresses are used in order to enable easyE4 base devices to communicate on an Ethernet network.

An IPv4 addresses are 32 bits (4 bytes) long and are used to uniquely identify networks, subnets, and individual computers that work with the TCP/IP protocol. A distinction is drawn between private address spaces for local networks (intranet) and public addresses (Internet).

A gateway is required in order to be able to address addresses outside of the local network.

The way that devices communicate with each other on a local Ethernet network can be compared to the way neighbors communicate with each other. The neighbors all live on the same street. Each one of them has their own house with a unique house number.

In this example, the street corresponds to the network portion of an IP address, and needs to be the same for all the devices on the subnet. Meanwhile, the house number corresponds to the host (device) portion of an IP address, which needs to be unique for every device on the subnet.

The network part of the IP address is obtained by AND'ing the subnet mask and IP address. This means that the subnet mask determines the other IP addresses that it will be possible to address on a local Ethernet network.

For example, in order for a PC with IP address 192.168.178.100 and subnet mask 255.255.254.0 to communicate with an easyE4, the easyE4 base device's subnet mask must be the same, and the IP address must fall within an address range of 192.168.(178–179).(1–254). The network part is always the same.

Table 6: Sample addresses for PC

| PC | Decimal | Binary |
|---|---|---|
| IP Address | 192.168.178.100 | 11000000 10101000 10110010 01100100 |

| PC | Decimal | Binary | |
|---|---|---|---|
| Subnet Mask | 255.255.254.0 | 11111111 11111111 11111110 00000000 | AND |
| Network section | 192.168.178.192 | 11000000 10101000 10110010 00000000 | |

Table 7: Sample addresses for easyE4

| easyE4 | Decimal | Binary | |
|---|---|---|---|
| IP Address | 192.168.178.1 | 11000000 10101000 10110010 00000001 | |
| Subnet Mask | 255.255.254.0 | 11111111 11111111 11111110 00000000 | AND |
| Network section | 192.168.178.192 | 11000000 10101000 10110010 00000000 | |
| | | | |
| IP Address | 192.168.178.254 | 11000000 10101000 10110010 11111110 | |
| Subnet Mask | 255.255.254.0 | 11111111 11111111 11111110 00000000 | AND |
| Network section | 192.168.178.192 | 11000000 10101000 10110010 00000000 | |
| | | | |
| IP Address | 192.168.179.1 | 11000000 10101000 10110011 00000001 | |
| Subnet Mask | 255.255.254.0 | 11111111 11111111 11111110 00000000 | AND |
| Network section | 192.168.178.192 | 11000000 10101000 10110010 00000000 | |
| | | | |
| IP Address | 192.168.179.254 | 11000000 10101000 10110011 11111110 | |
| Subnet Mask | 255.255.254.0 | 11111111 11111111 11111110 00000000 | AND |
| Network section | 192.168.178.192 | 11000000 10101000 10110010 00000000 | |

→ Please note that there are IP addresses that are not allowed to be used, as they are reserved for special purposes (e.g., broadcast and loopback IP addresses).

Additional information can be found in the Internet Assigned Numbers Authority's (IANA) RFC 6890 - Special-Purpose IP Address Registries.

**See also**

→ Section "Connecting the Ethernet cable", page 76
→ Section "Establish Ethernet connection", page 158

Prerequisites that must be met in order to be able to access an easyE4 control relays:

- The PC must have an Ethernet port that is free and has been configured
- The PC's Ethernet port must be on the same subnet as the easyE4 base device.
- The easyE4 control relays must be connected to the PC with a standard Ethernet cable featuring an RJ45 connector
- An Ethernet address must have been assigned to the easyE4 base device via DHCP, auto-IP, or manually.

▶ For devices with a display, write down the address of the easyE4 base device by going to *INFORMATION\ACTUAL CONFIG* and scrolling to IP ADDRESS.

▶ Open the Communication view in the easySoft 7 programming software.

**Communication view**

*Communication\Connection view*

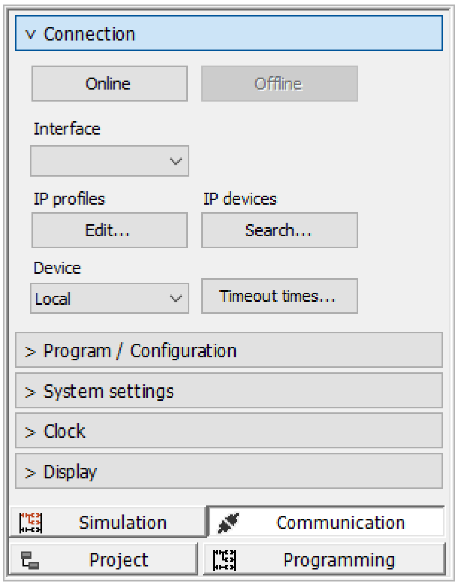



Figure 56: Establish Ethernet connection

- Click on Connection/IP profiles/Search... to open the Search for devices dialog box.
- Start a New search.

**Search for devices dialog box**

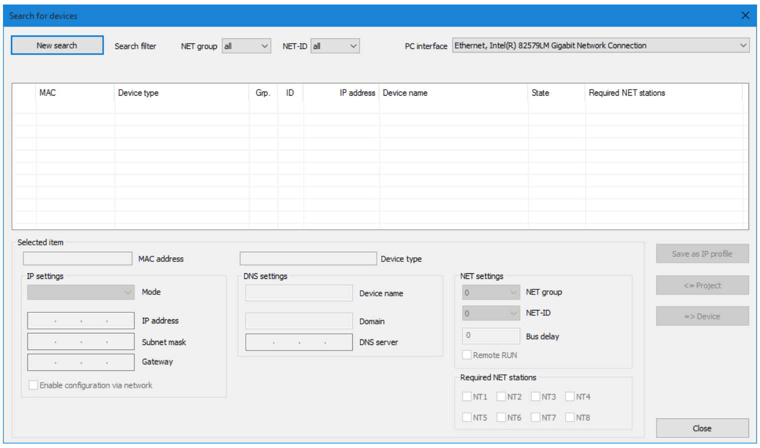



Figure 57: Search for devices with an IP address

If there is an Ethernet connection, the easyE4 base device will be found and listed in the table with the corresponding parameters.

- Click on the Save as IP profile button to save the IP profile for the found easyE4 base device.

*Search for devices dialog box*

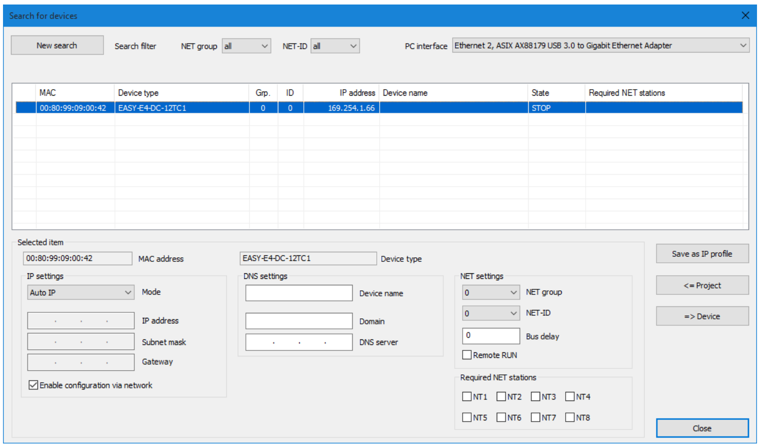



Figure 58: Saving the found device's IP profile

A prompt will appear saying that the IP address for the easyE4 base device has been created as a new profile.

- Close the Search for devices dialog box.

**Changes in the Interface drop-down menu**

The easyE4 base device's IP address will be found under Interface.

If a connection to multiple easyE4 base devices had already been established, there will be more entries available. In this case, select the IP address of the easyE4 base device you want under Interface.

*Communication\Connection view*

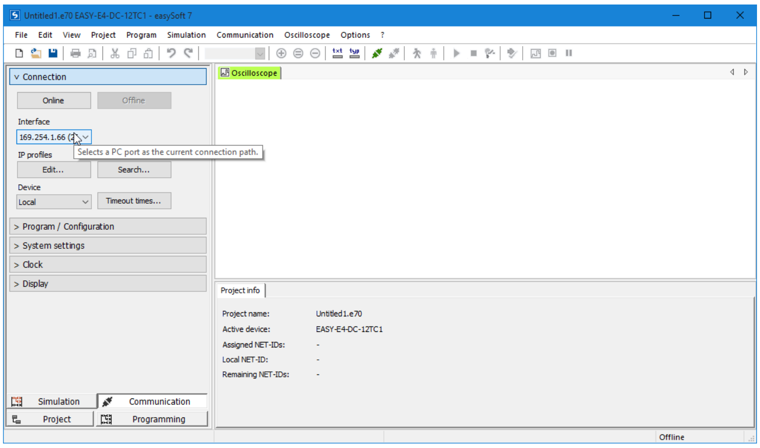



Figure 59: Selecting the easyE4 device's IP address

▶ Click on the Online button to establish a connection between your PC and the easyE4 base device.

→ If the easyE4 base device is protected with a password, a corresponding prompt will appear and ask you to enter the password before you can access the device.

If the password is correct, the connection to the device will be established.

As soon as the connection is established, the status line will show ONLINE.

▶ Transfer your *.e70 program by clicking on the PC => Device button in the Program / Configuration section.

*Communication\Connection view*

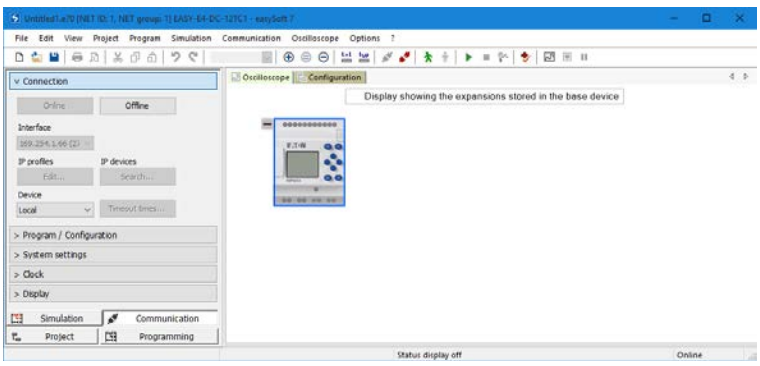



Figure 60: Connection to the easyE4 device established and program transferred

→ For additional help working with easySoft 7, read the various help topics in the easySoft 7 Help. To access it, press the F1 key on your keyboard.

When the project is loaded, the Ethernet settings in the project will be transferred to the easyE4 base device, or, more specifically: The parameter set under the Ethernet tab in the Project view will be transferred. Depending on how this set has been configured, the behavior of the Ethernet connection may change right after the project is loaded, and this may result in the device being disconnected. If the connection needs to be established again, you will need to repeat the steps above.

**See also**

→ Section "Connecting the Ethernet cable", page 76
→ Section "Establish Ethernet connection", page 158

# 3.6 Automatic booting from the card

easyE4 base devices can be booted from the memory card.

To be able to do this, the following prerequisites must be met:

- The microSD memory card must contain at least one compiled PRG program.
- One of the programs must have been set as the starting program, i.e., the microSD memory card must contain a BOOT.TXT file.
- If there is a program on the base device already, the Allow overwriting via card option must be enabled in the program.

If all these prerequisites are met, the device will boot from the card as follows:

▶ Insert the memory card while the device is de-energized.

▶ Switch on the supply voltage.

▶ Since the RUN start option is enabled by default, the device will automatically switch to RUN mode.

As soon as the easyE4 device switches to RUN mode, it will check whether there is a program in its internal memory. If there are none, the following step will be skipped. If there is one, the device will check whether the following option is enabled in *Project view/System settings tab/Memory card / device ID section/Program/device ID*: ☑ Allow overwriting via card – please refer to → "System settings tab" as well.

If this option is enabled, the starting program specified in the BOOT.TXT file will be copied from the card to the internal device memory and run.

The steps carried out when the device is turned on are shown in detail in the corresponding flowchart – please refer to → "Overview of switch-on behavior", page 98.

**Meeting the applicable prerequisites**

There are three different ways to prepare a microSD memory card for booting. Following is a description:

1. Preparing the card in the PC for booting with easySoft 7
   The microSD memory card must be inserted into a slot on the PC and written to from there.
2. Preparing the card in the device for booting with easySoft 7
   The microSD memory card is already in the device and is written to from the PC.
3. Preparing the card for booting on the device itself
   The microSD memory card is already in the device and is prepared for booting on the device itself. easySoft 7 is not needed in this case.

## 3.6.1 Preparing the card in the PC for booting with easySoft 7

Only possible with easySoft 7.

Prerequisites

- Licensed easySoft 7 version on the PC

- ▶ Insert the microSD memory card into a card reader on your PC.
- ▶ Open easySoft 7 and open the project you want to transfer, e.g., <test.e70>.
- ▶ If you want the starting program on the card to overwrite the current program on the device again later on, make sure to enable the Allow overwriting via card option in *Project view/System settings tab*.
- ▶ Start setting up the card by clicking on the *Project/Card...* menu option.
  If this is the first time you click on this menu option, make sure to select the drive corresponding to the microSD card.

The Card setup dialog box will appear.

### Transfer program

*Project/Card... menu option*

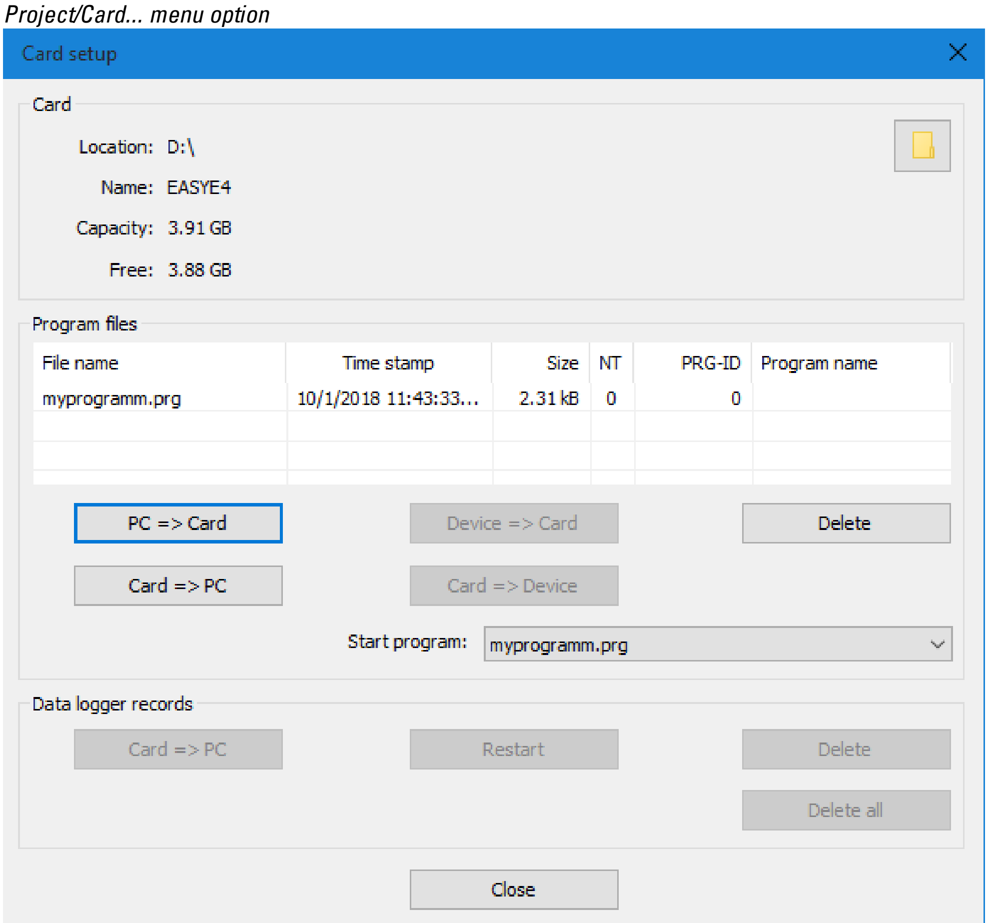
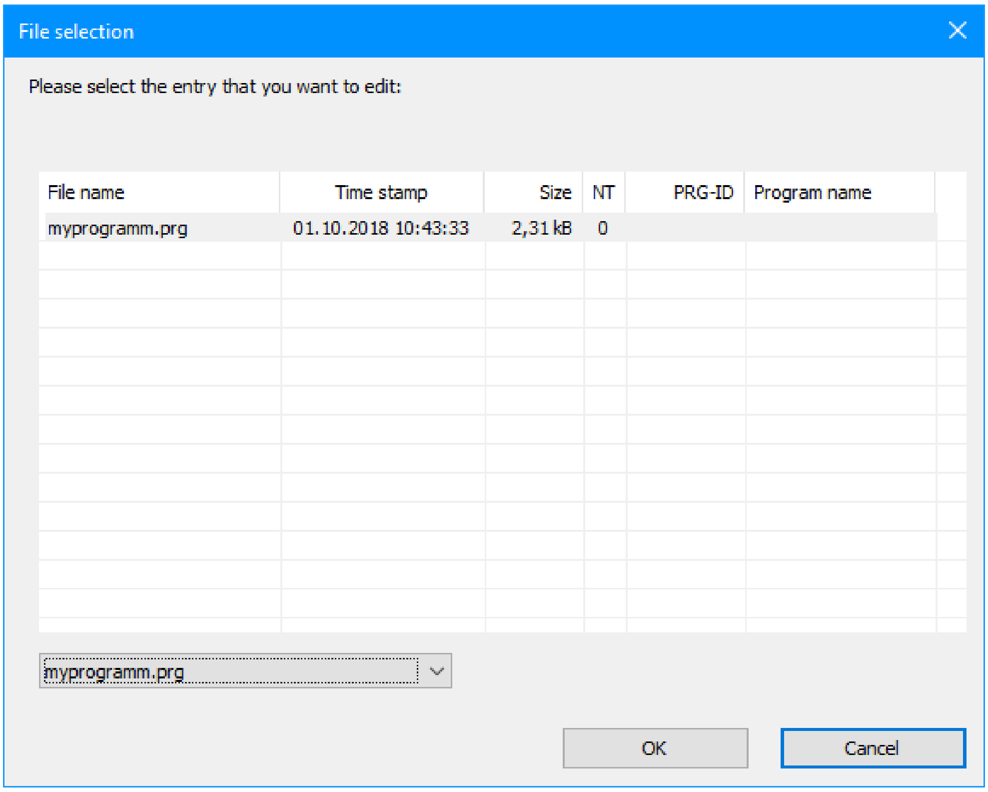



Figure 61: Offline dialog box for memory card

- ▶ Click the PC -> Card button.

The “File selection” window opens.




- If the list does not show any files, this means that there are no programs on the card.
  Enter the name you want for the program into the drop-down menu, e.g., <test>. This name can be different from the *.e70 file's name. You can also select a name from the menu.
- Confirm your selection with OK. This will transfer the program for the device selected in the Project view to the card.

If the project is a NET application, the "Selection of NET station" dialog box will appear.

- Select the NET station with the program that you want to transfer to the microSD memory card, e.g., <NET station NT1>.

A plausibility check will then be run – please refer to → "Running a plausibility check," page 1. If this plausibility check is completed successfully, the following prompt will appear.

### Setting a program as the starting program

"Do you want to enter the program as a start program on the card as well?"

- ▶ If you confirm by clicking on YES, the program will be set as a starting program for booting. Accordingly, a BOOT.TXT file that contains the name of the starting program will be generated. In addition, the name of the starting program will appear in the "Card setup" dialog box, in the Start program drop-down menu.

The .e70 program will be compiled into a .PRG program and shown in the list.

### Optional: Checking the microSD memory card

You can use Explorer to check the contents of the microSD memory card. It should now contain both the transferred program and the BOOT.TXT file.

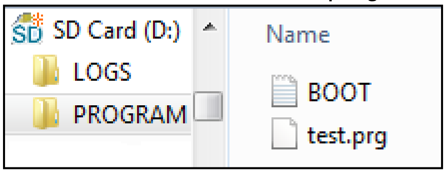



Figure 62: microSD memory card drive with PROGRAM folder contains BOOT.TXT and compiled test.prg program

The card is now prepared with all the prerequisites for booting. You can now use automatic booting from the card – please refer to → "Automatic booting from the card", page 106.

### 3.6.2 Preparing the card in the easyE4 device for booting with easySoft 7

Prerequisites

- Licensed easySoft 7 version on the PC

- ▶ Insert the card into the device while the latter is de-energized.
- ▶ Switch on the supply voltage.
- ▶ Open easySoft 7 and open the project you want to transfer, e.g., <myProgram.e70>.
- ▶ If you want the starting program on the card to overwrite the current program on the device again later on, make sure to enable the Allow overwriting via card option in *Project view/System settings tab*.
- ▶ Establish online communications between the PC and the device – please refer to → "Establish connection to device", page 618
- ▶ If there is a program on the device already, make sure that the Allow overwriting via card option is enabled in the program. To do so, enable the Allow overwriting via card option in *Project view/System settings*.
- ▶ Go to the *Communication view/Program/Configuration* section and click on the Card… button.

The "Card setup" dialog box will appear.

*Project/Card... menu option*

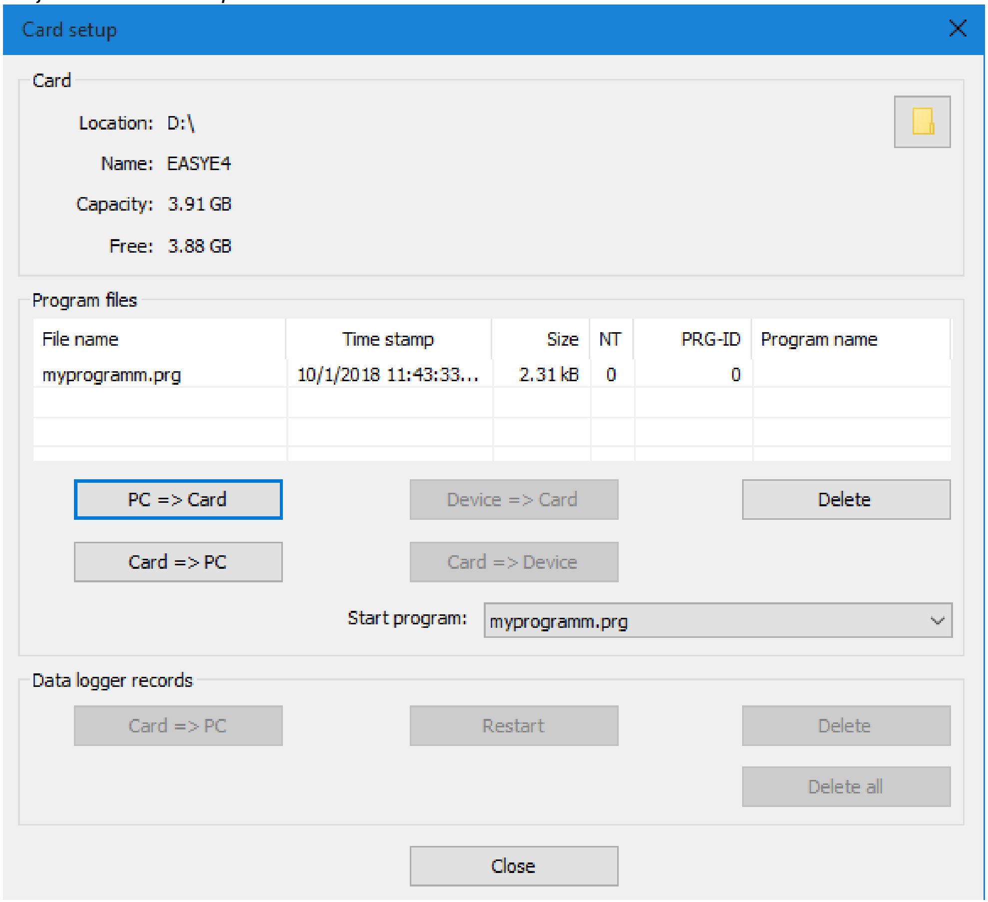
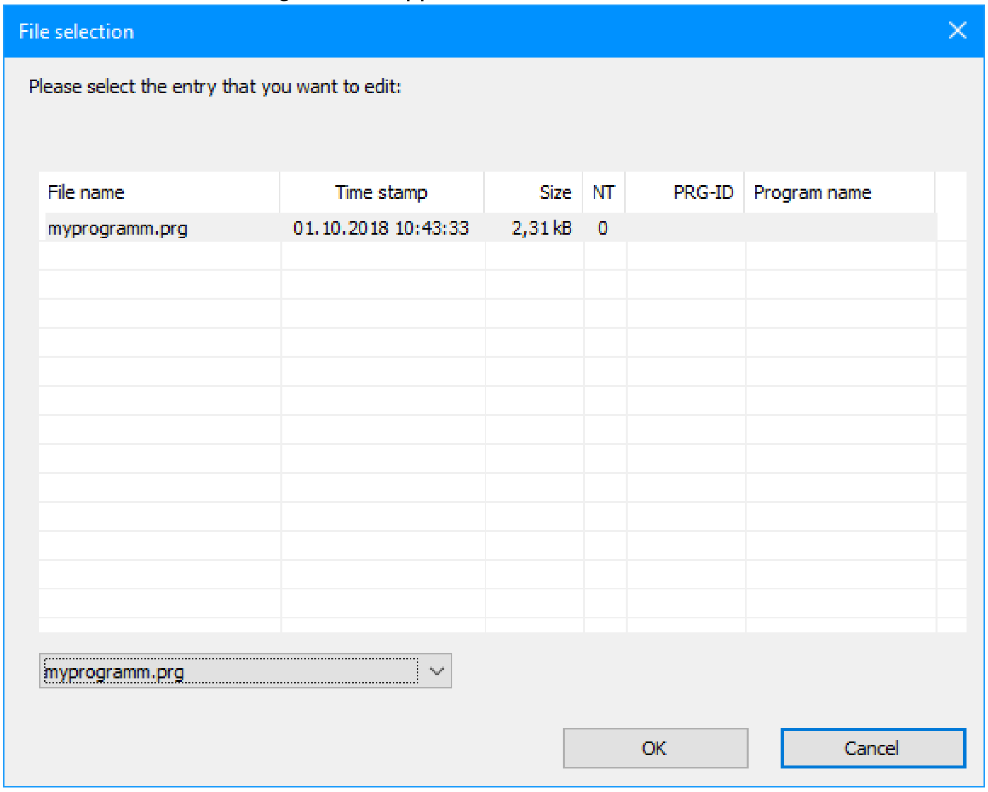



Figure 63: Offline dialog box for memory card

**Transfer program**

▶ Click the PC -> Card button.

The "File selection" dialog box will appear.




▶ If the list does not show any files, this means that there are no programs on the card.
Enter the name you want for the program into the drop-down menu, e.g., <test>.
This name can be different from the *.e70 file's name in easySoft 7. You can also select a name from the menu.

▶ Confirm your selection with OK. This will transfer the program for the device selected in the Project view to the card.

If the project is a NET application, the "Selection of NET station" dialog box will appear.

▶ Select the NET station with the program that you want to transfer to the microSD memory card, e.g., <NET station NT1>.

A plausibility check will then be run – please refer to → "Plausibility check", page 551. If the plausibility check is completed successfully, the following prompt will appear.

### Setting a program as the starting program

"Do you want to enter the program as a start program on the card as well?"

▶ If you confirm by clicking on YES, the program will be set as a starting program for booting. Accordingly, a BOOT.TXT file that contains the name of the starting program will be generated. In addition, the name of the starting program will appear in the "Card setup" dialog box, in the Start program drop-down menu.

The .e70 program will be compiled into a .PRG program and shown in the list.

### Optional: Checking the microSD memory card

You can use Explorer to check the contents of the microSD memory card. It should now contain both the transferred program and the BOOT.TXT file.

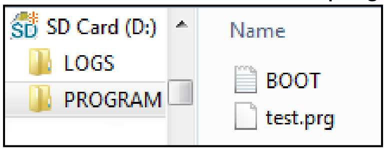



Figure 64: microSD memory card drive with PROGRAM folder contains BOOT.TXT and compiled test.prg program

The card is now prepared with all the prerequisites for booting. You can now use automatic booting from the card – please refer to → "Automatic booting from the card", page 106.

### 3.6.3 Preparing the card for booting on the easyE4 device itself

Prerequisites

- The microSD memory card must contain at least one compiled PRG program.

The easyE4 device must be in STOP mode before it can be configured. If it is not, the device will point this out.

▶ Insert the memory card while the device is de-energized.

▶ Switch on the supply voltage.

▶ Go to the main menu.

▶ Open the CARD menu option.

Table 8: *Main menu*

```
STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM-OPT.
PROGRAM
```

▶ Go to PROGRAM

Table 9: *Card*

```
PROGRAM
LOG RECORDING
MANAGE SD CARD
INFORMATION
```

▶ Go to START PROGRAM

Table 10: *Card\Program*

```
SET BOOT PROG.
DELETE PROGR.
CARD -> DEVICE
DEVICE -> CARD
```

▶ Select the starting program from the list containing the names of all the programs stored on the memory card.
The ✓ checkmark at the end of a line indicates the program with which the easyE4 device will start as soon as RUN mode is active.

Table 11: *Card\Program\Start program*

```
Name Program 1 ✓
Name Program 2
Name Program 3
...
```

→ If the display is empty, this means that no programs have been stored on the memory card.

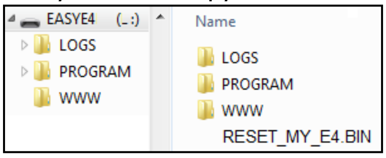
▶ Switch off the power supply.

The card is now prepared with all the prerequisites for booting. You can now use automatic booting from the card – please refer to → "Automatic booting from the card", page 106.

## 3.7 Reset – resetting the device to factory settings

To carry out a reset, follow the steps below:

- ▶ Create an empty file on the PC (with a text editor, for example) and rename it RESET_MY_E4.BIN.
- ▶ Use your PC to copy the file directly to the root of the microSD memory card.




- ▶ Switch off the easyE4 base device.
- ▶ Insert the microSD memory card.
- ▶ Switch on the easyE4 base device.
- ▶ Now turn off the easyE4 base device and remove the microSD memory card.

The easyE4 base device will be reset.
The program, password, and all settings will be deleted, and the network interface will work with auto-IP.

## 3.8 Updating firmware

As of firmware version V1.10, the firmware for the expansions in the easyE4 series can also be updated in addition to the base devices.
The process varies for base devices and expansions.

To update the firmware, you will need to use a microSD memory card. It is worth noting that you can also overwrite the firmware on your base devices with older firmware on the microSD memory card.

Eaton Industries GmbH, Bonn provides firmware updates *.zip files via its Download Center - Software page (under Firmware Updates).

Download Center - Software
http://www.eaton.eu/software/Firmware Updates/easy
http://www.eaton.eu/software/OS Updates/easy

In addition to the *.fw file that contains the firmware update, a configuration file (*.ini) is also stored in the same folder (ROOT) for base devices with bootloader version 1.01 and lower. This configuration file uses appropriate entries to control the update behavior of the base devices.

This configuration file is intended to enable series manufacturers to update the firmware for multiple devices in a row with a microSD memory card.

No configuration file is needed for expansion devices.

→ If the firmware on the easyE4 base device is the same version as the update, no update will be carried out.

Observe the documents belonging to the update in the download center.

## Updating the firmware for base devices

All base devices can be updated with newer firmware.

To find out which generation your easyE4 device belongs to, you can check the device menu or, during online communication with the easyE4 base device, the information in *Communication view/HW Info tab*.

If there is a program on the base device, the program will be left unchanged when the firmware is updated. Retentive data will remain unchanged as well.

- Download the firmware you want from Download Center – Software to your computer.
- Connect an empty microSD memory card (FAT format) to your computer.
- Use your computer to unzip the downloaded firmware to the ROOT directory in the microSD memory card.

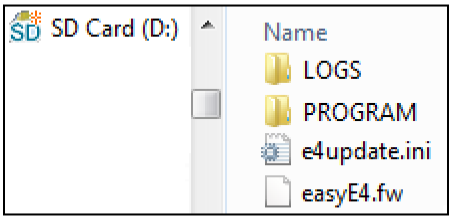



Figure 65: microSD memory card content when using bootloader version 1.01

The specific files that will be unzipped and that are required for the firmware update will depend on the bootloader version on the base device, as shown below:

| Unzipped files | Boot loader version 1.00 | Boot loader version 1.01 |
|---|---|---|
| Firmware file "EASYE4.FW" | √ | √ |
| Configuration file "e4update.ini" | – | √ |

During online communication with the easyE4 base device, the bootloader version that is found on the device will be shown in *Communication view/HW Info tab* .

- **If the bootloader version is version 1.01, check the parameters in the INI file**
  Check the parameters in the “e4update.ini” configuration file and adjust them as necessary. In firmware version V1.12 and higher, the following values will be set by default:
  `forceupdate=0 (default)` (dominant value)
  and
  `updateonce=1 (default)`

| force update | update once | |
|---|---|---|
| 0 | 0 | No update is run. |
| 0 | 1 | The update will be run once (default). |
| 1<br>1 | 0<br>1 | An update from the microSD memory card will always be run. |

→ Once the update has run, the entry for updateonce is automatically set to 0 in the configuration file.
This means that the default settings will cause the firmware to be updated once only.

For additional updates from the microSD memory card, you will need to edit the "e4update.ini" configuration file manually by setting `forceupdate=1`.

▶ Switch off the easyE4 base device.

▶ Insert the microSD memory card with the new firmware into the microSD card holder and slide the holder into the device.

→ Make sure that the power is stable and that the device is not turned off while the firmware is being updated (if it is, the operating system may be corrupted). Then run the firmware update again.

▶ Switch on the easyE4 base device.

Bootloader version 1.01: The configuration in the "e4update.ini" file will be read in the easyE4 bootloader and a compatibility check will be run. An update will not be carried out if the firmware on the device and the firmware on the card match.

Bootloader version 1.00: The firmware will be transferred from the microSD memory card to the base device.

If the firmware on the device is being updated, a corresponding message will be shown on the display or the POW/RUN/STATUS LED will indicate that an update is being carried out.

- Quickly flashing POW/RUN/Status LED:
  The device is looking for the firmware on the microSD memory device.
- The LED POW/RUN/Status flashes slowly and rhythmically,
  the update is running.

The new firmware starts then.

→ You can go to *INFORMATION\SYSTEM* to see what the current firmware version is.

- Switch off the supply voltage.
- Remove the microSD memory card with the firmware from the device.

→ If the firmware transferred from the microSD memory card is older than the firmware selected in the project, the project will not be able to start. The project may contain functions that the firmware currently on the device does not support.

**The following applies when using bootloader version 1.01:**

If you do not remove the microSD memory card, the parameters in the "e4update.ini" configuration file will be read every time the device is turned on and, if applicable, the firmware will be updated.

**The following applies when using bootloader version 1.00:**

If you do not remove the microSD memory card, the program will not start (when turning on the device) until after the firmware is transferred again from the microSD memory card.

## Firmware update expansion device

An expansion device update must run via the device menu of a easyE4 base device.

Expansion devices belonging to the first easyE4 generation (with firmware version 1.00) cannot be updated, as these devices do not have a bootloader physically. To find out which firmware version is found on the device, check the *Communication view/HW Info tab* during online communication. .

You can reach the device menu through

- a base device with display, or
- in the communication view of the easySoft 7 under Display\Display + keys or
- from a remote display with the web server.

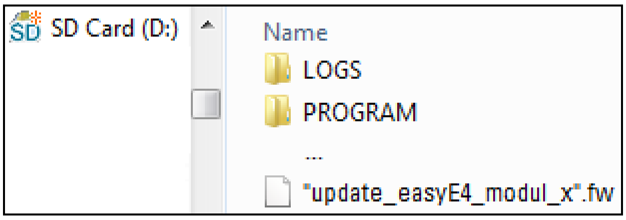
An update must be run separately for each expansion device.

Just as with updates for base devices, this requires for the required unzipped "*.FW" firmware file to be stored on the microSD memory card.

▶ Load the firmware you want on your computer.

▶ Connect an empty microSD memory card (FAT format) to your computer.

▶ Use your computer to unzip the downloaded firmware to the ROOT directory in the microSD memory card.




The unzipped file must be a firmware system file that matches the easyE4 expansion device (*.FW).

→ No entry is required in a configuration file for an update.

To update the firmware, the easyE4 expansion device must be connected to the base device with the EASY-E4-CONNECT1 plug connector.
The number of the easyE4 expansion is determined based on the position after the base devices in the assembly block, starting with 1 from the left. The maximum number 11 can be assigned to an expansion in the block.
An update must be carried out separately for each expansion device.

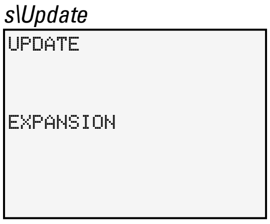
### An expansion is updated from the base device with display

Expansion devices belonging to the first easyE4 generation (with firmware version 1.00) cannot be updated, as these devices do not have a bootloader physically.

Take the following steps to update an expansion from a base device with display:

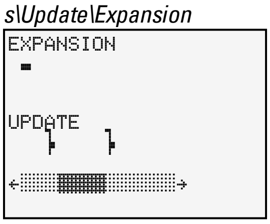
- Go to the main menu.
- Open the menu path SYSTEM OPTIONS\UPDATE\EXPANSION.

Table 12: *System option-s\Update*




- Select the number of the easyE4 expansion in the block; 1 to 11 are possible.

Table 13: *System option-s\Update\Expansion*




- Select the corresponding firmware file.
- Press the OK button to select.

A confirmation prompt is displayed.

- You can return to the previous menu by selecting "No".
- The update starts immediately by selecting "Yes".

"Update" flashes in the display.

After the update has ended, the display returns to the menu SYSTEM OPTIONS\UPDATE\EXPANSION.
Repeat the process for other easyE4 expansion devices.

→ You can only view the hardware information (HW Info), i.e., which firmware version is on the easyE4 expansion device, via easySoft 7.
To do so, go to the Communication view and connect to your easyE4 block. In the workspace Configuration, the FW version is displayed in the HW info.

**See also**

→ Section "Transfer program to the easyE4 device", page 153
→ Section "Transferring programs from and to the microSD memory card", page 187
→ Section "Overview of switch-on behavior", page 98
→ Section "Splash screen", page 565
→ Section "DL - Data logger", page 475

## 3.9 microSDMemory card

easyE4 base devices can be used with a microSD memory card.

The easyE4 device supports microSD memory cards with a capacity of 128 MB to 32 GB (SD and SDHC, FAT12/16/32, Class 2 or 4 ).

→ The following card manager functions for microSD memory cards and the online function are not available in demo mode.

Moreover, using a memory card makes it possible to use the following functions:

1. Automatic booting from the memory card
   The easyE4 can load and run a starting program from the memory card.
2. Resetting to factory settings
   starting program
3. Download new firmware
4. Setting a splash screen for the EASY-E4-...-12...C1(P) display
   You can store a boot.bmp file on the memory card so that it will be shown on the display when starting the easyE4 and when inserting the card
5. Transferring user programs, Storing multiple programs
6. Logging data
   → Section "DL - Data logger", page 475

In order to be able to transfer programs or use the data logger function, the microSD memory card must be formatted accordingly.
The actual transfer is carried out in easySoft 7, in the Project view.
The DL - Data logger function block can be used for logging data and states.

### 3.9.1 microSD Ejecting the memory card

As an alternative to removing the memory card from the device, you can eject it with easySoft 7.

There are two ways to do this:

Method 1

- In easySoft 7, go to *Communication view/Program/Configuration*.
- Click the Card... button.
- In the Card setup dialog box, click on the Release button.

Method 2

- In easySoft 7, go to *Communication view/Display/Display + buttons*.
- Press OK to exit the display.
- Make sure that the device is in STOP mode.
- Use the P buttons to select the CARD menu option and confirm your selection with

OK.

- Use the P buttons to select the MANAGE SD CARD menu option and confirm your selection with OK.
- Use the P buttons to select the RELEASE CARD menu option and confirm your selection with OK.

After this, the microSD memory card will no longer be accessible.

## 3.10 Setting a splash screen for the EASY-E4-...-12...C1(P) display

You can create your own custom monochromatic image externally in any program. Simply make sure that the image is in BMP format and is named boot.bmp.

The size is set at 128 x 96 pixels (width x height) or, alternatively, 128 x 64 pixels. You can use two colors, which will be shown as shades of gray.
Make sure that the file always keeps the name boot.bmp!

*Splash screen*


Figure 66: boot.bmp

- Transfer the image to the microSD memory card.
- Store the boot.bmp file directly on the memory card.

*microSD memory card on the PC*

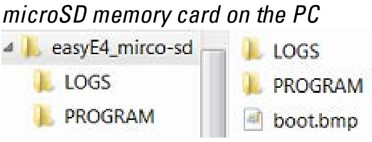



Figure 67: Storing the boot.bmp file

As soon as the easyE4 device is switched on,
the boot.bmp will be shown as a splash screen for the defined duration.

→ In order for the splash screen to keep working, the microSD memory card must remain in the device.

# 4. Operation

The way in which the various base devices are operated varies.
Only EASY-E4-...-12...C1(P) base devices with a display and buttons can be operated directly.
Meanwhile, EASY-E4-...-12...CX1(P) base devices with LED indicators used for diagnostic purposes, as well as all expansion devices, are limited to providing information via their LEDs' flashing patterns.

→ Section "Startup behavior of easyE4 control relays with LED indicators", page 88

## 4.1 Base device with display and buttons


Figure 68: Display and keypad

### 4.1.1 LCD Display

Monochrome device display with six lines, each with 16 characters (128 x 96 pixels).

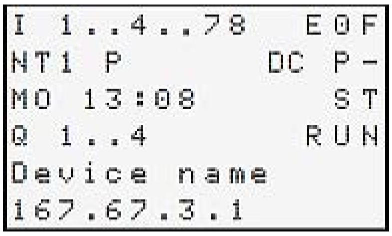



Figure 69: Example of status display on display

The display can show texts, values, parameters, and pseudographics.

Once the device is switched on, as well as during ongoing operation, the device will switch to sleep mode and display the status display after a configurable time without any operator activity.

#### 4.1.1.1 Display color backlight

The display backlight can be illuminated white, red, or green or be switched off in order to signal specific device states.

The backlight brightness has three different available levels.

▶ Press the OK button on the device in order to open the main menu from the status display.

Either the cursor position or the available action on the display will flash. A checkmark ✓ will be shown to indicate which option is currently selected. Please note that since the display has six lines only, you may need to use the ⋀ ⋁ cursor buttons to scroll to the remaining available lines on a screen.

The display's settings can be configured on the easyE4 device in the *SYSTEM OPTIONS\SYSTEM\DISPLAY* menu, → Section "Display", page 563

### 4.1.2 Keyboard

| | |
|---|---|
| DEL | Deleting in the circuit diagram |
| ALT | Special functions in circuit diagram, Status display |
| Cursor buttons ⋀ < > ⋁ | Move cursor<br>Select menu items,<br>Change numbers, contacts and values |
| ESC | Back, Cancel |
| OK | Next menu level, Save your entry |

Once the device is switched on, as well as during ongoing operation, the device will switch to sleep mode and display the status display after a configurable time without any operator activity.

▶ Press the OK button on the device in order to open the main menu from the status display.

▶ Use the ⋀ ⋁ cursor buttons to scroll through the individual menu options.

▶ To confirm a selection, press OK. to open the corresponding menu path.

▶ If necessary, use the < > cursor buttons while in a line to toggle between the right and left display areas.
If this option is available, the <> character will appear.

### 4.1.3 Selecting menus and entering values

| | |
|---|---|
| 2x ALT | Calling the System menu |
| ESC | Return to the previous menu level<br>Cancel entries since last OK |
| OK | Move to the next menu level<br>Call menu item<br>Activate, change, save entries |
| Cursor buttons ↑ ← → ↓ | Change menu item,<br>Change the value,<br>Activate, change, save entries |

How the P button functions are mapped to the cursor buttons:

| | |
|---|---|
| ← | Input P1 |
| ↑ | Input P2 |
| → | Input P3 |
| ↓ | Input P4 |

- Press the OK button on the EASY-E4-... in order to open the menu from the status display.
- Use the ↑ ↓ cursor buttons to scroll through the individual menu options.
- To confirm a selection, press OK. to open the corresponding menu path.
- If necessary, use the ← → cursor buttons while in a line to toggle between the right and left display areas.
  If this option is available, the ◇ character will appear.

### 4.1.4 Cursor display

The cursor buttons ⮉ ⮋ ⮈ ⮊ in the easyE4 program perform three functions:

- Move
- Enter
- Connect

The current mode is indicated by the appearance of the flashing cursor.

Current selection flashes on the easyE4 display

| | |
|---|---|
| ▮ | In Move mode, use ⮉ ⮋ ⮈ ⮊ to position the cursor on the circuit diagram in order to select a rung, a contact or a relay coil or the selection position of a coil function or a NET-ID. |
| I 01 | Use the OK button to switch to **Entry** mode so that you can enter or change a value at the current cursor position.<br>Press the ESC button in **Entry** mode to restore to the last changes of an entry. |
| ↙ | Press the ALT button to switch to **Connect** mode for wiring contacts and relays.<br>Press the ALT button again to return to **Move**. |
| ESC | Press the ESC button to leave the program (circuit diagram and parameter display). |

### 4.1.5 Entering of values

| | | |
|---|---|---|
| ⮉ ⮈ ⮊ ⮋ | ⮈ ⮊ | Used to select a position. |
| | ⮉ ⮋ | Use to select values and/or a setting |
| ESC | | Cancel, retain previous value |
| OK | | Store settings |

## 4.2 Operating modes of the easyE4

A easyE4 device recognizes operating modes RUN and STOP.

### 4.2.1 RUN mode

In RUN mode, the program stored on the device will be executed immediately

after the device is switched on and will continue to be executed until you select STOP, a system fault occurs, or the supply voltage is switched off.
The outputs will be driven based on the relevant switch logic. The parameter values will be retained in the event of a power failure, and all you will need to do is reset the real-time clock in the event that the backup time elapses. → "Back-upof real-time clock", page 702

In RUN operating mode:

- the process image of the inputs are read.
- The program is executed.
- The NET is run (Ethernet, web server, and Modbus TCP).
- the process image of the outputs is transferred to the physical outputs.

The easyE4 devices with a display do not start with RUN mode if you deactivate the RUN MODE startup behavior.

easyE4 devices with LED indicators have a different startup behavior. On them, the RUN START and CARD START functions will be enabled automatically, as manually starting the device is not possible.
For more information on the CARD START function, please refer to → Section "Setting the startup behavior", page 571

### 4.2.2 STOP mode

When the device is in STOP mode, the program will not be executed. In order to be able to do programming on the circuit diagram, modify system parameters, or configure communications, the device must be in this operating mode.
In addition, the program can be stored on the microSD memory card or be loaded from it while the device is in this mode.

**WARNING! AUTOMATIC STARTUP!**
Configure your machine/plant so that the automatic starting of the easyE4 device never causes unintentional starting of the machine/plant concerned.
Create your program in such a way that a defined and safe startup procedure is always provided after the power supply is switched on.

The main menu on the display is used to switch between operating modes, i.e., from RUN to STOP and vice versa,→ Section "STOP RUN operating mode menu", page 135

If a program has not been stored on the easyE4, it will not be possible to switch to RUN mode.
It will not be possible to do configuration work either.

In order to configure it, the program must be stopped.

STOP ✓ RUN

Operating mode changes may be protected with a password.

Ongoing operation

Table 14: *Main menu*

| |
|---|
| STOP RUN ✓ |
| PARAMETERS |
| SET CLOCK |
| CARD |
| INFORMATION |
| SYSTEM-OPT. |
| PROGRAM |

To work on the easyE4

Table 15: *Main menu*

| |
|---|
| STOP ✓ RUN |
| PARAMETERS |
| SET CLOCK |
| CARD |
| INFORMATION |
| SYSTEM-OPT. |
| PROGRAM |

# 4.3 Operation of the menu selection and value entry

## 4.3.1 How to navigate the device menus

| | |
|---|---|
| OK | Select; confirm value |
| ESC | Cancel; go back |
| DEL | Delete |
| ALT | Depending on the starting point:<br>- Toggle display<br>- Jump to the start or end of the menu<br>- Jump to next line |
| ◁ | Go left |
| ◁ | Go right |
| △ | Go up; increment value |
| ◁ | Go down; decrement value |

## 4.3.2 Operating principle in the circuit diagram and function block editor

| Button | Effect |
|---|---|
| DEL | Delete rung, contact, relay or empty rung in the circuit diagram |
| ALT | Toggle between N/C and N/O contact, connect contacts, relays and rungs, add rungs |
| △ ▽ | Change value; move cursor up, down |
| ◁ ▷ | Change position; move cursor left, right |
| ESC | Undo setting from last OK, current display, leave menu |
| OK | Change contact / relay. Insert new; save settings |
| ◁ | P1 input when used as P button |
| ▷ | P2 input when used as P button |
| △ | P3 input when used as P button |
| ▽ | P4 input when used as P button |

### 4.3.3 Selecting a device menu

You can use the main menu from the status display in order to access the individual sub-menus.

▶ Press the OK button.

The main menu will appear.
The scrollbar on the right side will indicate whether there are additional menu options.

Please note that since the display has six lines only, you may need to use the ⌃ ⌄ cursor buttons to scroll to the remaining available lines on a screen

Table 16: *Main menu*

```
STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM-OPT.
PROGRAM
```

A horizontal scroll bar indicates that other selection options are available. You may be able to reach them using the cursor keys ‹ ›.

# 4.4 Overview of the menus on the device

Following is an overview of the menu structure and the various submenus that can be accessed from the main menu.

## 4.4.1 Main menu

Ongoing operation

Table 17: *Main menu*

| |
|---|
| STOP RUN ✓ |
| PARAMETERS |
| SET CLOCK |
| CARD |
| INFORMATION |
| SYSTEM-OPT. |
| PROGRAM |

To work on the easyE4

Table 18: *Main menu*

| |
|---|
| STOP ✓ RUN |
| PARAMETERS |
| SET CLOCK |
| CARD |
| INFORMATION |
| SYSTEM-OPT. |
| PROGRAM |

## 4.4.2 STOP RUN operating mode menu

This submenu can be used to switch between operating modes.

| |
|---|
| STOP ✓ RUN |

| |
|---|
| STOP RUN ✓ |

Table 19: *STOP*

| |
|---|
| STOP? |
| YES |
| NO |

Table 20: *RUN*

| |
|---|
| RUN? |
| YES |
| NO |

**See also**

→ Section "Operating modes of the easyE4", page 131

### 4.4.3 Menu Parameter

This submenu lists the function blocks that are being used in the current program. This makes it possible to change the values of constants in the program at runtime without having to stop or re-transfer the program.

When the password is activated and the +/- basic parameter for each function block is set, you can allow or deny the operator of the system the possibility to change the values.

Function blocks with basic parameters that you have set to + via the +/- character in the function block editor are displayed in the PARAMETERS menu and can be changed. It is only possible to change constants. Other operands cannot be changed. It is also possible to change parameters via the PARAMETERS menu if you have saved the program and therefore password protected the function block editor.

To immediately apply a change to the individual constants, press the OK button. Press the ESC button to cancel the change instead.

List of function blocks in the current program, e.g.,

The current program does not use any function blocks

Table 21: *Parameter*

```
STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM-OPT.
PROGRAM
```

Table 22: *Parameters*

```
T 01      ⊔  S    +

C 02




L:1          STOP
```

Table 23: *Parameter*

```
NO FUNCTION BLOCKS
INCLUDED!
```

Pressing the OK button will show the parameters for the individual function blocks in an additional submenu that can be used to modify these parameters with the cursor buttons.

Table 24: *Time function block example*

```
T 01      ⊔    S     +

>I1      000  800
>I2      009  200
QV>      000  000
..
```

### 4.4.4 Set clock menu

This submenu can be used to set the date and time, select the display format for the date, and adjust the daylight saving time and radio clock settings on the easyE4 device.

Opens additional menus

Table 25: *Set clock*

```
STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM-OPT.
PROGRAM
```

Table 26: *Set clock*

```
TIME/DATE
DST
RADIO CLOCK
ASTRON. CLOCK
```

Table 27: *Set Clock-\Time/Date*

```
   DD-MM-YYYY
FR 13.08.2018
   12:03:04
```

Table 28: *Set clock\Daylight saving time*

```
NO                ✓
DST
US
RULE
```

Table 29: *Set Clock\Radio Clock*

```
RADIO
CLOCK
ACTIVE     : YES
INPUT      : I001
OFFSET     : +000'
```

Table 30: *Set clock\astron. clock*

```
ASTRON.       CLOCK
LAT     N000. 0000000
LNG     E000. 0000000
OFFSET:       +000'
```

**See also**

→ Section "Time and Date setting", page 587

### 4.4.5 Menu Card

This submenu will only be available if a memory card is detected in the slot.

Opens additional menus

Table 31: *Main menu*

| |
|---|
| STOP ✓ RUN |
| PARAMETERS |
| SET CLOCK |
| CARD |
| INFORMATION |
| SYSTEM-OPT. |
| PROGRAM |

Table 32: *Card*

| |
|---|
| PROGRAM |
| LOG RECORDING |
| MANAGE SD CARD |
| INFORMATION |

Table 33: *Card\Program*

| |
|---|
| SET BOOT PROG. |
| DELETE PROGR. |
| CARD -> DEVICE |
| DEVICE -> CARD |

Table 34: *Card\Log recording*

| |
|---|
| START NEW LOG |
| DELETE OLD LOG |
| DELETE ACT. LOG |

Table 35: *Card\Manage SD card*

| |
|---|
| FORMAT |
| RELEASE CARD |

Table 36: *Card\Information*

| | |
|---|---|
| EXISTING: | YES |
| FORMATTED: | YES |
| SIZE | xxxMB |
| FREE | XXXMB |

**See also**

→ Section "microSD memory card", page 1

→ Section "Transferring programs from and to the microSD memory card", page 187

→ Section "Configuring the microSD card and device ID", page 586

### 4.4.6 MenuInformation

Shows the current status of the easyE4 device.

Opens additional menus,
The submenu is only provided in English

Table 37: *Main menu*

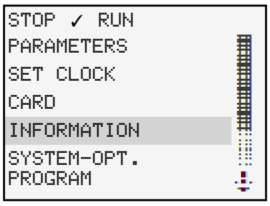
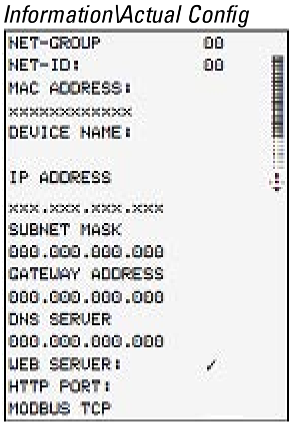
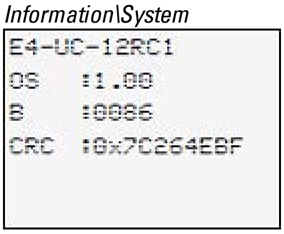



Table 38: *Information*


*Information\Actual Config*




*Information\System*




Shows the versions on the easyE4
Specification of the part number
OS: Operating system version
B: Firmware build version
CRC: Cyclic redundancy check result

To update the operating system

→ Section "microSD memory card", page 1
→ Section "Device information", page 598

### 4.4.7 System options menu

This menu can be used to configure the basic settings for the system.

Opens additional menus

Table 39: *Main menu*

| |
|---|
| STOP ✓ RUN |
| PARAMETERS |
| SET CLOCK |
| CARD |
| INFORMATION |
| SYSTEM-OPT. |
| PROGRAM |

Table 40: *System options*

| |
|---|
| SECURITY |
| SYSTEM |
| MENU LANGUAGE |
| DELETE PROGR. |
| NET |
| ETHERNET |
| UPDATE |

Table 41: *System options\Security*

| |
|---|
| PASSWORD |
| RANGE |

Table 42: *System options\System*

| |
|---|
| DEBOUNCE |
| P BUTTONS ✓ |
| RUN MODE |
| CARD MODE |
| LOAD CARD |
| INDICATOR |
| DEVICE ID |
| BOOT LOGO |

Table 43: *System Options\Menu Language*

| |
|---|
| ENGLISH |
| DEUTSCH ✓ |
| FRANCAIS |
| ESPAÑOL |
| ITALIANO |
| NEDERLANDS |
| POLSKI |
| ČESKÝ |
| PORTUGUÊS |
| РУССКИЙ |
| TÜRKÇE |
| ROMÂNĂ |
| MAGYAR |

Deletes the program in the easyE4 device.

Table 44: *System options\Delete progr.*

| |
|---|
| DELETE PROGR.? |
| YES |
| NO |

The submenu is only provided in English

*System options\Net*

| | |
|---|---|
| NET-GROUP: | 00 |
| NET-ID: | 00 |
| BUSDELAY: | 000 |
| REMOTE RUN | |

| | |
|---|---|
| The submenu is only provided in English | *System options\Ethernet*<br>ADDRESS MODE<br>IP ADDRESS<br>SUBNET MASK<br>GATEWAY ADDRESS<br>DNS SERVER |
| Available in OS Version 1.10 and higher | Table 45: *System option-s\Update*<br>UPDATE<br><br>EXPANSION |

**See also**

→ Section "System settings", page 561
→ Section "Security – password protection", page 581
→ Section "Setting up a NET", page 625
→ Section "Setting up a web server", page 646
→ Section "Modbus TCP", page 632
→ Section "E-mail function", page 670
→ Section "microSD memory card", page 1

### 4.4.8 Program menu

→ This menu will only be available if the easyE4 is configured with its factory settings and/or when a program created with the EDP programming language has been stored on the easyE4 device.

You can use this menu to create a program with the EDP programming language directly on the easyE4 device.

Table 46: *Main menu*

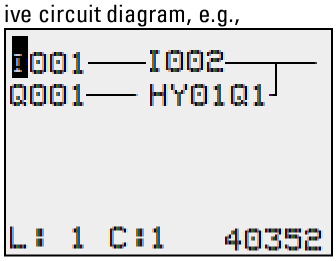
```
STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM-OPT.
PROGRAM
```

Opens another menu

Table 47: *Program*

```
CIRCUIT DIAGR.
FUNCTION BLOCKS
```

Used to display and edit the active circuit diagram, e.g.,




Pressing the OK button will show the parameters for the individual function blocks in an additional submenu that can be used to modify these parameters with the device's cursor buttons.

Table 48: *Programs\Function Blocks*

```
T 01      ⊔   S    +

C 02



L:1            STOP
```

Table 49: *Timer module example*

```
T 01      ⊔     S     +

>I1       000   800
>I2       009   200
QV>       000   000
..
```

### Menu options in the status line for work with the circuit diagram and the function blocks

When you complete your work on the circuit diagram and exit the menu with the ESC button, you will be able to select from the CANCEL, SEARCH, GO TO, and SAVE options by scrolling through them with the ▲ ▼ cursor buttons in the bottommost line.

After editing the function blocks, the CANCEL and SAVE options will be available.

Table 50: *Program\Circuit diagram*

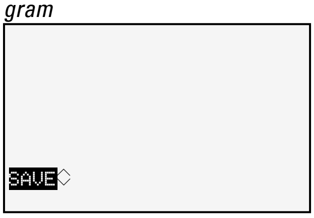
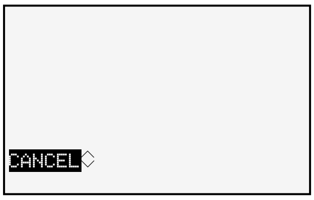



Table 51: *Program\Function blocks*



## 4.5 Your first EDP program

This section is intended to guide you step-by-step through the process of creating your first program with the easy Device Programming (EDP) language in order to wire a circuit diagram. This should enable you to become familiar with all the relevant rules and use an easyE4 device for your own projects in no time. Just like with conventional wiring, you will be using contacts and relays in the program. This means that the easyE4 device makes it possible to eliminate the use of these components in a variety of ways, including the use of function blocks.

The easyE4 program will take care of the entire wiring for these components.

All you have to do is then connect to the easyE4 any switches, sensors, lamps or contactors you wish to use.

→ Use easySoft 7 to create your own program

Opens another menu

Table 52: *Main menu*

```
STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM-OPT.
PROGRAM
```

Table 53: *Program*

```
CIRCUIT DIAGR.
FUNCTION BLOCKS
```

Prerequisites for entering a circuit diagram

- The easyE4 device is in STOP operating mode.
- The display must be showing the status display.

▶ Press the OK button to get to the main menu from the status display.

▶ Use the ▲ ▼ buttons to scroll to the Programs menu option.

▶ Open the menu option by pressing the OK button.

The *PROGRAMS\CIRCUIT DIAGRAM* menu option will now be selected on the easyE4 device.

In general, press the OK button to switch to the next menu level, and press the ESC button to move one level back.

▶ Press the OK button twice to enter the circuit diagram display via menu options <PROGRAM… ->CIRCUIT DIAGRAM>. This is where you will create the circuit diagram.

## Circuit diagram display

The content of the circuit diagram is displayed in the first 5 lines. This window can be moved over the circuit diagram. At the moment the circuit diagram is empty.
The cursor flashes at the top left, which is where you will start to wire your circuit diagram.

*Circuit diagram display*

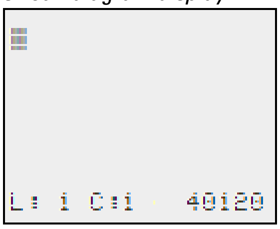



Figure 70: Empty circuit diagram

The last line shows the cursor's position:

- L: = Rung (Line).
- C: = Contact or coil field (Column).
- Amount of free memory in bytes.

### 4.5.1 Draw a wiring diagram

The circuit diagram supports four contacts and one coil in series. The display shows 6 fields of the circuit diagram.

Use the ⓐ ⓥ ⓒ ⓓ cursor buttons to move the cursor over the invisible circuit diagram grid.

*Navigating in the circuit diagram*

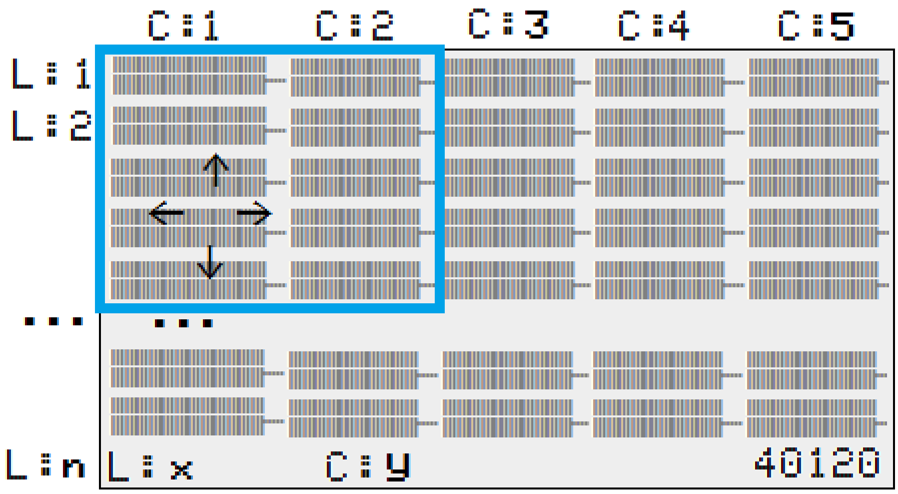



Figure 71: Fields in the circuit diagram

The first four columns C are contact fields, the fifth column is a coil field. Each line L is a circuit connection.
The easyE4 automatically connects the contact to the power supply.

The following example is provided for a lighting control. The easyE4 device takes on the wiring and the tasks of the circuit shown below.

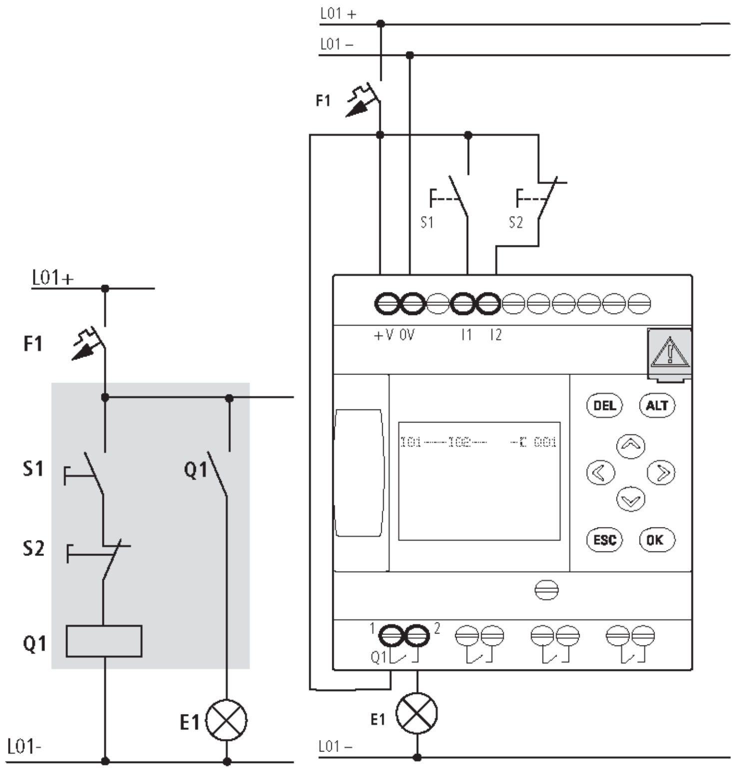



Figure 72: Lighting control circuit

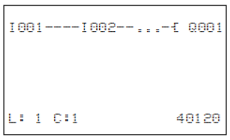



Figure 73: Circuit diagram with inputs I01, I02 and output Q1

▶ Now wire the circuit diagram as described below.

With this example the switches S1 and S2 are at the input. `I001` and `I002` are the switch contacts for the input terminals in the circuit diagram .

The relay Q1 is represented in the circuit diagram by the relay coil -[ `Q001`.

The symbol -[ identifies the coil's function, in this case a relay coil acting as a contactor. `Q001` is one of the easyE4device's outputs.

### From the first contact to the output coil

With easyE4 devices wire from the input to the output. The first input contact is `I001`.

▶ Press the OK button.

easyE4 will insert the first contact, `I001`, at the cursor position.

The `I` flashes and can be changed, for example,
to a `P` for a button input by using the cursor buttons ⌃ or ⌄. However, nothing needs to be changed at this point.

▶ Press the OK button twice,
to move the cursor across the 001 to the second contact field.

You could also move the cursor to the next contact field using the cursor button.

▶ Press the OK button.

The easyE4 device will once again insert an I001 contact at the cursor position.

▶ Press the OK. button so that the cursor will jump to the next position.

▶ Use the ⮉ or ⮋ cursor buttons to select the number 002.

→ Pressing DEL will delete the contact at the cursor position.

▶ Press the OK button to move the cursor to the third contact field.

You do not need a third relay contact, so you can now wire the contacts directly up to the coil field.

**Wiring**

easyE4 device displays a small arrow in the circuit diagram when creating the wiring.

Press the ALT button to activate the arrow and press the cursor buttons ⮉ ⮋ ⮈ ⮊ to move it. Pressing the ALT button once more switches the cursor back to Move mode.

→ The ALT button also has two other functions depending on the cursor position:

- In the left contact field, you can press the ALT button to insert a new empty circuit connection.
- The contact under the cursor can be changed between a N/O and N/C contact by pressing the ALT button.

The wiring arrow works between contacts and relays.
When you move the arrow onto a contact or relay coil, it changes back to the cursor and can be reactivated if required.

→ The easyE4 device automatically connects adjacent contacts up to the coil.

▶ Press the ALT to wire the cursor from I002 through to the coil field.

The cursor changes into a flashing wiring arrow and automatically jumps to the next logical wiring position.

▶ Press the cursor button ⮊.

Contact I002 will be connected up to the coil field.

→ You can use the DEL button to erase a connection at the cursor or arrow position. Where connections intersect, the vertical connections are deleted first, then, if you press the DEL button again, the horizontal connections are deleted.

▶ Press the cursor button ⮚ once more.

The cursor will move to the coil field.

▶ Press the OK button.

The specified coil function [ and the output relay Q001 are correct and do not have to be changed.

Your result will look as follows: Your first wired and functional circuit diagram

```
I001---I002---------[ Q001



L:1          C:5                40120
```

Figure 74:  Completed circuit diagram

You can use the cursor buttons to access the part that is not visible.

▶ Press the ESC button to leave the circuit diagram display.

Line 6 will show the SAVE menu option.

**Saving**

```
I001---I002--------------[ Q001



SAVE       ↕
```

Figure 75: SAVE menu option in the status line

▶ Press the OK button to confirm.

The circuit diagram is stored.

▶ Press the ESC button twice to return to the main menu.

You can test the circuit diagram if the S1 and S2 buttons are connected.

### 4.5.2 Testing the circuit diagram

▶ Go back to the main menu.

▶ Select the STOP RUN menu option.

The current operating mode is indicated on the display of the easyE4 device by a tick ✓ at RUN or STOP stop. Pressing the OK button enables you to toggle between the modes.

▶ Press the OK pushbutton in order to change to RUN.

→ The Status display also shows the current mode set.

### 4.5.3 Control options in RUN mode

There are two control options in RUN mode. Control of:

1. Inputs and outputs with Status display
2. power flow with power flow display

**Status display during RUN mode**

▶ Change to the status display and press pushbutton S1.
Do not execute pushbutton S2.

The contacts for inputs I001 and I002 are activated and relay QS1 picks up, indicated by the highlighted numbers.

**Test using the power flow display**

▶ Change to the circuit diagram display and press pushbutton S1.

The relay picks up and the easyE4 displays the power flow with a double line.

```
I001===I002===========[ Q001




 L:1          C:1                RUN
```

Figure 76: Power flow display 1

Power flow display: Inputs I001 and I002 are closed, relay Q1 has picked up

▶ Press pushbutton S2, that has been connected as a break contact.

The power flow is interrupted and relay Q1 drops out.

```
I001===I002------------[ Q001




 L:1          C:1                RUN
```

Figure 77: Power flow display 2

Power flow display: Input I001 is closed, input I002 is open,
relay Q1 has dropped out

▶ Press the ESC button to return to the Status display.

→ The circuit diagram does not have to be completed in its entirety so that it is possible to test parts of it. The easyE4 device simply ignores any incomplete wiring that is not yet working and only uses the finished wiring.

### Power flow display with zoom function

A reduced display of the circuit diagram is possible for a better overview. To do this, follow the steps below:

▶ Change to the circuit diagram display and press the ALT button.

The circuit diagram display is reduced.

▶ Press pushbutton S1.

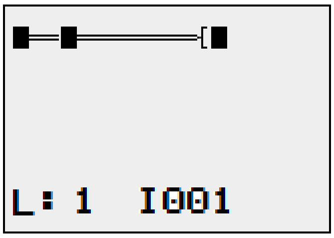



Figure 78: Display with zoom, power flow

Power flow display in Zoom function: Input I001 and I002 are closed, relay Q1 picked up

■ Contact closed, coil is triggered.

□ Contact opened, coil not triggered.

▶ Press pushbutton S2, that has been connected as a break contact.

The power flow is interrupted and relay Q1 drops out.

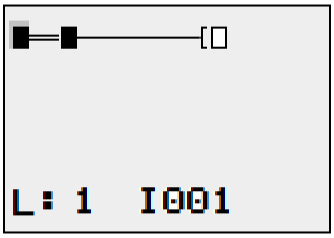



Figure 79: Display with zoom, power flow interrupted

Use the cursor buttons to move from contact to contact or coil.

▶ Press the cursor button .

The cursor moves to the second contact.

Press the ALT button. The zoom function will be turned off and the display will switch to the display status with contact and/or coil designations.

Power flow display: Input I01 is closed, input I02 is open, relay Q1 has dropped out.

### 4.5.4 Delete Program

The DELETE PROGRAM function not only deletes the circuit diagram but all elements of a program. These are as follows:

- Circuit Diagram
- Function block list
- Function block diagram
- Screens

The system settings and operating parameters are reset to the default settings and also a possible NET parameterization.

Proceed as follows to delete the program in the easyE4 device:

easyE4 must be in STOP mode in order to extend, delete or modify the circuit diagram.

▶ Switch the easyE4 device to STOP mode.

▶ From the main menu, go to the SYSTEM OPTIONS menu.

Opens additional menus

Table 54: *Main menu*

| |
|---|
| STOP ✓ RUN |
| PARAMETERS |
| SET CLOCK |
| CARD |
| INFORMATION |
| SYSTEM-OPT. |
| PROGRAM |

Table 55: *System options*

| |
|---|
| SECURITY |
| SYSTEM |
| MENU LANGUAGE |
| DELETE PROGR. |
| NET |
| ETHERNET |
| UPDATE |

Table 56: *System option-s\Delete progr.*

| |
|---|
| DELETE PROGR.? |
| YES |
| NO |

▶ Select DELETE PROGRAM.

The easyE4 device will show a confirmation prompt.

▶ Select YES.

▶ Press the OK button to delete the program

Or

▶ Press the ESC button to cancel

Pressing the ESC button once more returns you to the previous menu level.

## 4.6 Transfer program to the easyE4 device

There are two options for directly transferring a finished .e70 program to an easyE4 device.

- With a microSD memory card
- With a direct Ethernet connection between the PC and easyE4

### 4.6.1 Transfer with a microSD memory card

Prerequisites

- You will temporarily need a suitable microSD memory card with a maximum storage capacity of 32 GB.
- A PC on which the easySoft 7 programming software is installed, → Section "Installation instructions", page 83

▶ Insert the microSD memory card into a drive on your PC (with a suitable adapter if necessary).

▶ Open the easySoft 7 programming software on your PC.

▶ Create an application program and save it.

Use the help in the ? menu by accessing the help topics with the F1 key or open the easyE4 manual.

Or

▶ Open a sample program.→ Section "Sample Projects", page 711

→ Make sure to stay in the Project view, as the Project menu will only be available there.

**Application examples**

Support has provided a number of applications that are available for download as ZIP files from the Software Download Center.

Download Center - Software
http://www.eaton.eu/software/Anwendungsbeispiele/easy/Deutsch
http://www.eaton.eu/software/Application Samples/easy/English

These examples come with a task description, the circuit diagram, and the easySoft 7 project (in the EDP and LD programming languages as of this writing).

▶ Click on the Project\ Card... menu option.

*easySoft 7 Project view*

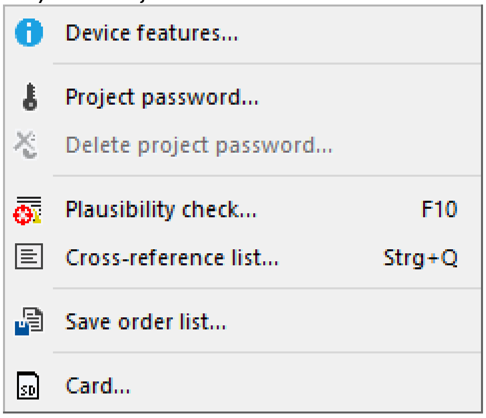
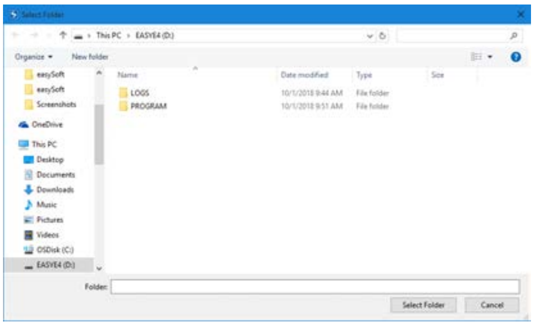



Figure 80: Sample program open

In the Select Folder dialog box that appears, select a folder where the LOGS and PROGRM folders that easySoft 7 needs should be created.

▶ Select the drive where the memory card is located

Exit the dialog box by clicking on Select Folder.



The Card setup dialog box will appear.

*easySoft 7 Project View\Project\Card...*

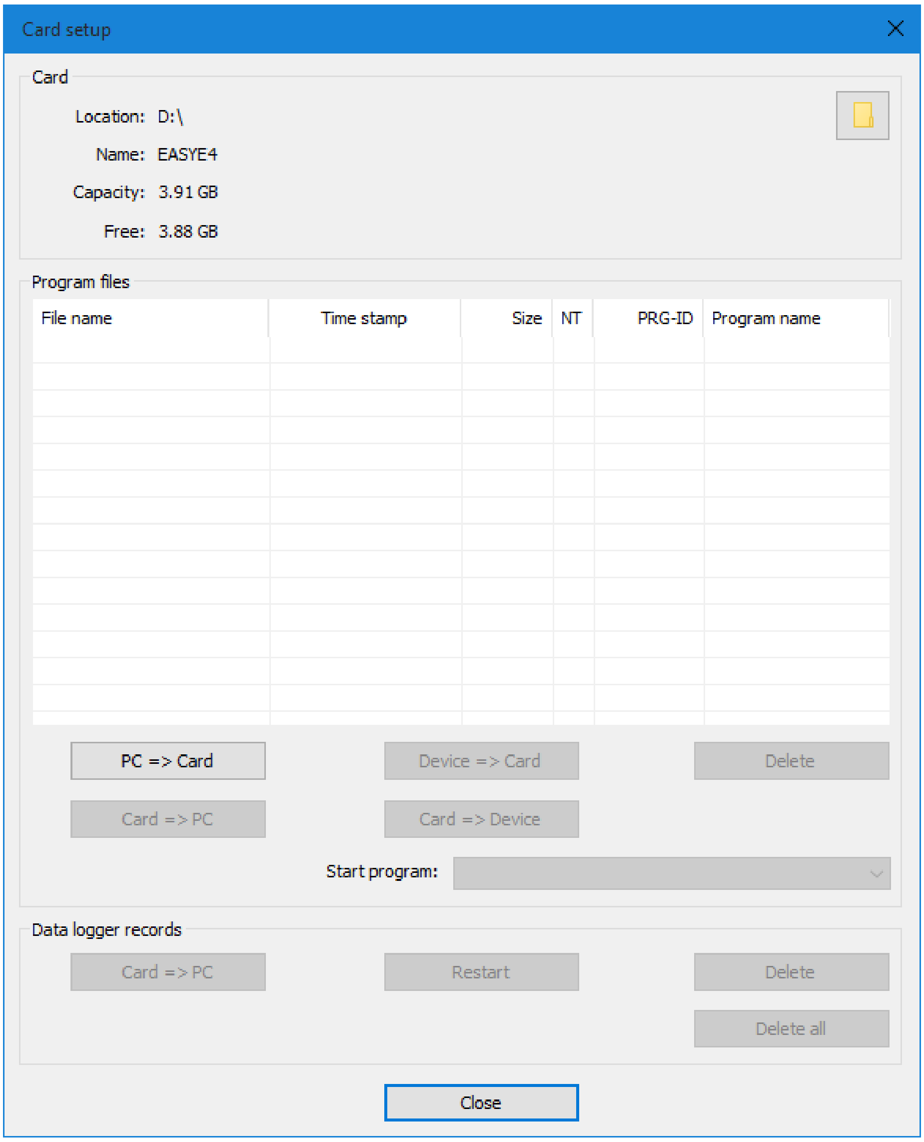
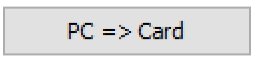



Figure 81: Card setup dialog box

You can use the Card section to specify the storage location, i.e., the drive, where the microSD memory card is located.
In addition, this section will show the available information about the memory card.

▶ Click on the PC => Card button to select the transfer option you want.




The File selection dialog box will appear.

Use this dialog box to enter the name that the program should have on the easyE4 device.
Please make sure that this name does not have more than 14 characters (only letters and numbers are permitted).

▶ Enter a new entry into the input field.

*easySoft 7 Project View\Project\Card...*

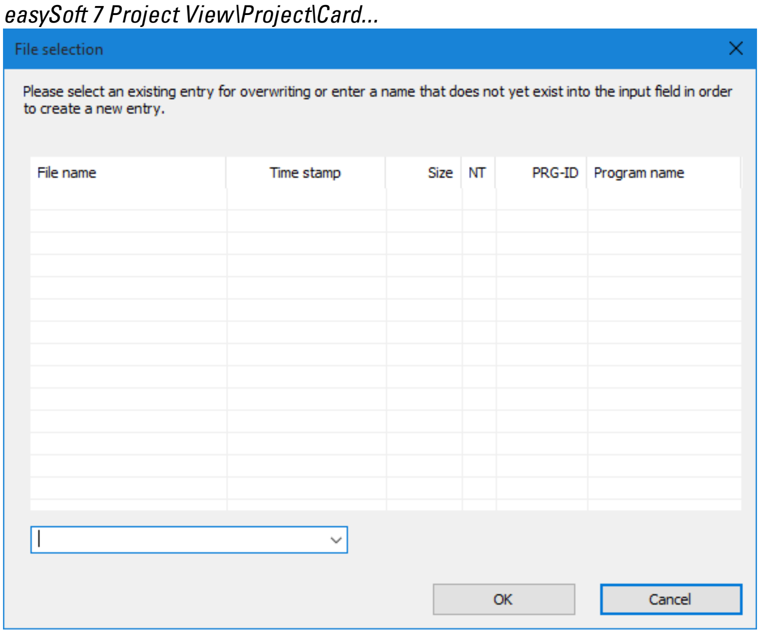



Figure 82: File selection dialog box

The following safety prompt will appear:
Do you want to enter the program as a start program on the card as well?

→ This prompt is intended for cases in which the easyE4 device will start running this program as soon as a supply voltage is applied. If you select the Yes option, please make sure to take into account the potential automatic startup and the settings configured in the program

▶ Click on Yes for the test with a sample program from this manual.

*easySoft 7 Project View\Project\Card...*

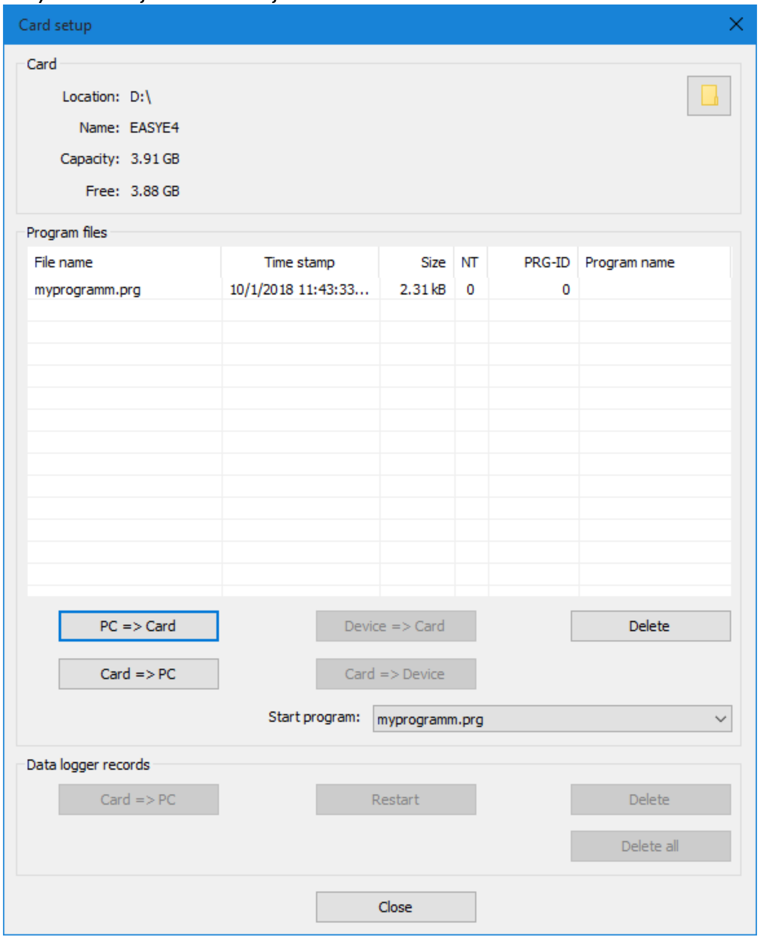



Figure 83: The program was transferred to the memory card.

- Close the window
- Remove the microSD memory card from the drive.
- Insert the microSD memory card into the slot on the easyE4 base device.
  → Section "Inserting a microSD card", page 73

The easyE4 device will be ready for operation.

- Apply the corresponding supply voltage while observing all relevant safety instructions.
- The easyE4 device will start running the program (depending on the operating mode).

Or

- Transfer the program from the microSD memory card to the device
  if you did not define the program as the starting program. → page 187

### 4.6.2 Establish Ethernet connection

Establishing a connection between a PC and an easyE4 base device

In order to be able to establish a connection, your PC must first have the required Ethernet capability. This capability can be in the form of a local Ethernet port on the PC or a standard adapter (e.g., USB-to-Ethernet adapter).

The PC's and the easyE4 base device's IP addresses must fall within the same range, i.e., the addresses' first two or three numbers must be the same, while the last number must be different and not equal to 0.

- ▶ Check the easyE4 device's IP address.
- ▶ To do this, open the *INFORMATION\ACTUAL CONFIG* menu and scroll to IP ADDRESS.

Table 57: *Main menu*

| |
|---|
| STOP ✓ RUN |
| PARAMETERS |
| SET CLOCK |
| CARD |
| INFORMATION |
| SYSTEM-OPT. |
| PROGRAM |

Table 58: *Information*

| |
|---|
| ACTUAL CONFIG |
| SYSTEM |

The submenu is only provided in English

*Information\Actual Config*

| | |
|---|---|
| NET-GROUP | 00 |
| NET-ID: | 00 |
| MAC ADDRESS: | |
| xxxxxxxxxxxx | |
| DEVICE NAME: | |
| | |
| IP ADDRESS | |
| xxx.xxx.xxx.xxx | |
| SUBNET MASK | |
| 000.000.000.000 | |
| GATEWAY ADDRESS | |
| 000.000.000.000 | |
| DNS SERVER | |
| 000.000.000.000 | |
| WEB SERVER: | ✓ |
| HTTP PORT: | |
| MODBUS TCP | |

- ▶ If an IP address has not been assigned yet, enter one.
- ▶ To do this, open the *SYSTEM OPTIONS\ETHERNET\IP Address* menu.

Table 59: *Main menu*

| |
|---|
| STOP ✓ RUN |
| PARAMETERS |
| SET CLOCK |
| CARD |
| INFORMATION |
| SYSTEM-OPT. |
| PROGRAM |

Table 60: *System options*

| |
|---|
| SECURITY |
| SYSTEM |
| MENU LANGUAGE |
| DELETE PROGR. |
| NET |
| ETHERNET |
| UPDATE |

Table 61: *System options\Ethernet*

| |
|---|
| ADDRESS MODE |
| IP ADDRESS |
| SUBNET MASK |
| GATEWAY ADDRESS |
| DNS SERVER |

Table 62: *System Options\Ethernet\IP Address*

| |
|---|
| IP-ADDRESS |
| 000.000.000.000 |

- ▶ Use the cursor buttons to enter the device's IP address.

Table 63: *System Options\Ethernet\Address mode*

| | |
|---|---|
| AUTO IP | ✓ |
| DHCP | |
| STATIC IP | |

▶ Select the network setting you want.

▶ Set up a new Ethernet connection on your PC's operating system.

To do this, go to Windows Network and Sharing Center and set up a LAN connection over Internet Protocol Version 4 (TCP/IPv4). Then enter an IP address in the same range you used for the device, but with a different device number.

*Example using Windows*

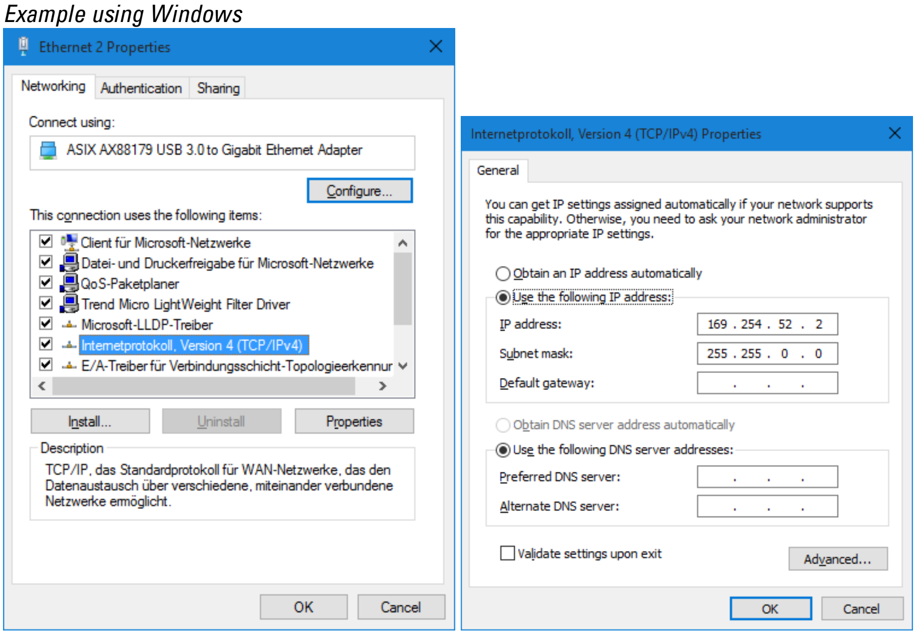



Figure 84: Ethernet connection on PC

You can now connect to your easySoft 7 device with the easyE4 programming software.

**See also**

→ Section "Establishing an Ethernet connection and transferring a program", page 100

# 5. Programming on the device

This chapter describes how to wire easyE4 contacts and coils with the display and keypad on an EASY-E4-...-12...C1(P).

## 5.1 Program

An easyE4 program consists of the required system settings for the easyE4 device, NET, password, and operating parameters,
as well as the following:

- Circuit diagram (program on the easyE4)
- Function block list
- Function block diagram

Programs have file extension .e70. However, please note that this file extension will not be shown on the display.

The programs themselves can be created very easily with easySoft 7 and then transferred to the easyE4 device.
easySoft 7 Help provides corresponding support.

## 5.2 Circuit diagram display

The circuit diagram, i.e., the program with which the EASY-E4-...-12...C1(P) works, can be displayed in the main menu under Program.

*Programs/Circuit Diagram*

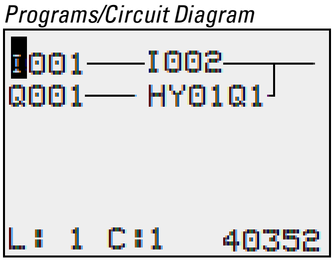



Figure 85: Circuit diagram display

You wire contacts and coils of relays in the easyE4 circuit diagram from left to right, from the contact to the coil.
The circuit diagram is created on a hidden wiring grid containing contact fields, coil fields and rungs. It is then wired up with connections.

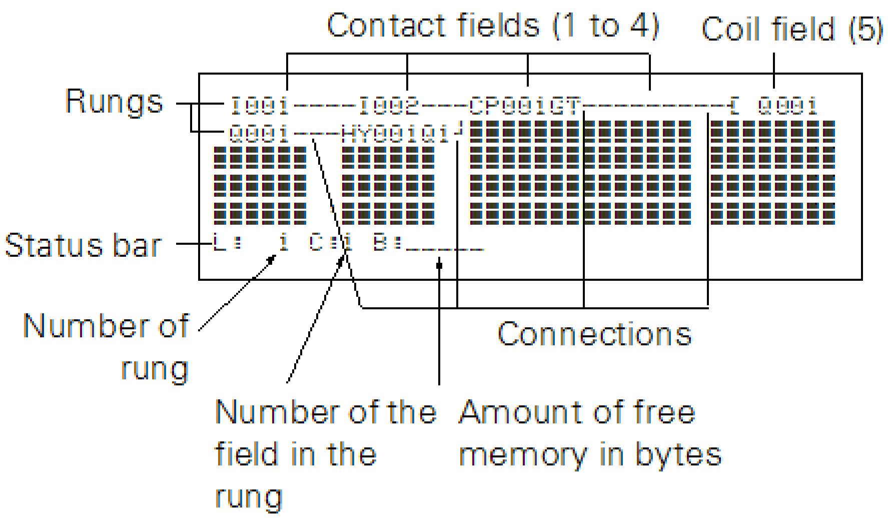
- Insert contacts in the four contact fields. The first contact field on the left is automatically connected to the voltage.
- The coil field can be used to enter the relay coil being driven together with the corresponding coil identifier and coil function. The coil identifier consists of a coil name,

coil number, and, in the case of function blocks, the function identifier. The coil function defines the method of operation of the coil.

The cursor buttons ⮝ ⮟ can be used to change the contact fields. The number of the circuit connection and the contact are displayed in the lower status line.

There are 256 rungs available in the circuit diagram, for wiring contacts and coils.

For greater legibility the circuit diagram of the easyE4 device shows two contacts or one contact plus coil in a row on each rung. A total of 16 characters per circuit connection and five circuit connections plus the status line can be displayed simultaneously.




Program display on the display

- Connections are used to produce the electrical contact between relay contacts and the coils. They can be created across several rungs. Each point of intersection is a connection.
- The number of free bytes is displayed so that you can recognize how much memory is available for the circuit diagram and function blocks.

→ The circuit diagram display has two functions:

- It is used to edit the circuit diagram in STOP mode.
- It is used in RUN mode to check the circuit diagram using the power flow display.

# 5.3 Circuit diagram elements

A circuit diagram is a series of commands that the easyE4 device processes cyclically in RUN operating mode.

Coils and contacts are interconnected in the circuit diagram. In RUN mode, a coil is switched on or off depending on the power flow and coil function set.

This section provides more information on how contacts and coils are mapped to operands; please refer to → "Using operands in a program", page 200.

## 5.3.1 Function blocks

Function blocks are program elements with special functions. Examples: Timing relays, time switch, data block comparators. Function blocks are provided with or without contacts and coils. How a safety or standard function block is added to the safety or standard circuit diagram and parameterized is described in
→ Section "Working with function blocks", page 191

In RUN mode the function blocks are processed according to the circuit diagram and the results are updated accordingly.

Examples:
Timing relay = Function block with contacts and coils
Time switch = Function block with contacts

## 5.3.2 Relays

Relays are switching devices that are emulated in easyE4 device electronically, which activate their contacts according to the assigned function. A relay consists of at least one coil and contact.

### 5.3.3 Contacts

Contacts are used to change the power flow inside the easyE4 circuit diagram. Contacts such as N/O contacts are set to 1 when they are closed and 0 when they are opened. In the easyE4 circuit diagram you can wire contacts as make or break contacts. N/C contacts are indicated with a horizontal line above the operand concerned.

An easyE4 device operates with different contacts that you arrange in any order in the contact fields of the circuit diagram.

Table 64: Usable contacts

| | Switching contact | | Look |
|---|---|---|---|
| | N/O contact, i.e., normally open contact | | I, Q, M, A, … |
| | N/C contact, i.e., normally closed contact | | I, S, S, A, … |

A detailed list of all contacts used in the circuit diagram is provided on → Section "Function blocks", page 216

### 5.3.4 Coils

Coils are the actuating mechanisms of relays. The results of the wiring are transferred to the coils when the device is in RUN mode. These switch to the On (1) or Off (0) state according to these results. The options for setting output and marker relays are listed with the description of each coil function.

A easyE4 device is provided with different types of relays and function blocks which can be wired in a circuit diagram via their coils (inputs).

#### Coil functions

You can configure the switching behavior of relays with coil functions and parameters.

→ If you want to map coils from your circuit diagram to the easyE4 device, use the coils with a contactor function in your device!

The following coil functions are available for all coils:

Table 65: Coil function

| Display | Coil function | Example | → Page |
|---|---|---|---|
| -[ | Contactor function | -[Q01,-[D02,-[S04,-[:01,-[M07,.. | → page 166 |
| ʃ | Impulse relay function | ʃQ03,ʃM04,ʃD08,ʃS07,ʃ:01,.. | → page 166 |
| S | Set | SQ08,SM02,SD03,SS04.. | → page 167 |
| R | Reset | RQ04,RM05,RD07,RS03.. | → page 167 |
| ]- | Contactor function with negated result | ]-Q04,]-M96.. | → page 168 |
| ┌ | Cycle pulse on rising edge | ┌M01.. | → page 168 |
| └ | Cycle pulse on falling edge | └M42.. | → page 169 |

→ With non-retentive coil functions such as -[ (contactor), ]- (negated contactor), ┌ (rising) and └ (falling edge evaluation): Each coil must only be used once. The last coil in the circuit diagram determines the state of the relay. Exception: When working with jumps, the same coil can be used twice.
Retentive coil functions such as S, R, ʃ can be used several times.

The available coil functions for the various function blocks are described in the relevant sections. Please refer to → Section "Working with function blocks", page 191

### Coil with contactor function

The output signal follows the input signal directly, the relay operates as a contactor.

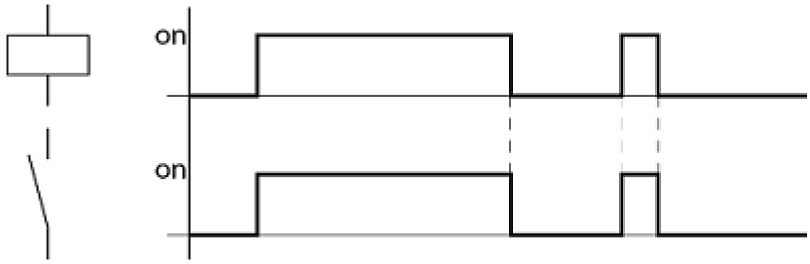



Figure 86: Contactor function signal diagram

### Coil with the impulse relay function

The relay coil switches whenever the input signal changes from 0 to 1. The relay behaves like a bistable flip-flop.

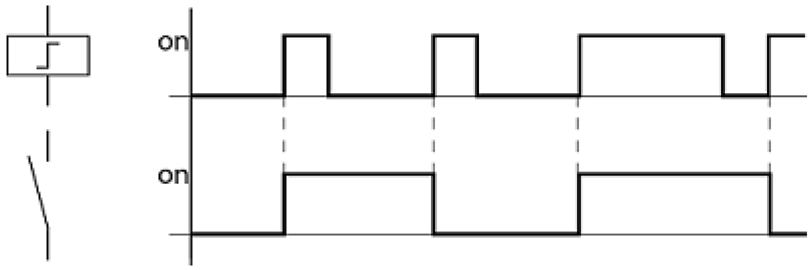



Figure 87: Impulse relay signal diagram

A coil is automatically switched off in the event of a power failure and in STOP mode. Exception: retentive coils retain the 1 state.

**See also**

→ Section "Retention function", page 578

#### "Set" and "Reset" coil functions

"Set" the coil function S and "Reset" R coil functions are normally used in pairs.

The relay picks up when the coil is set (A) and remains in this state until it is reset (B) by the coil function.

The power supply is switched off (C), the coil is not retentive.

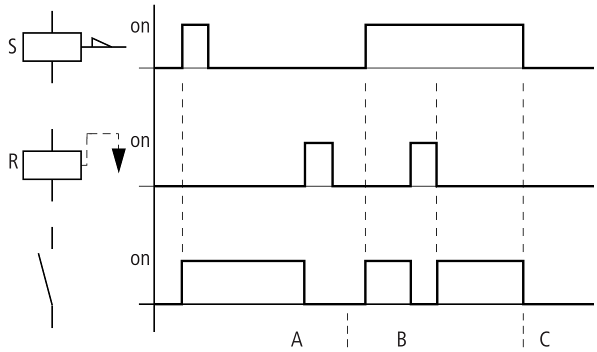



Figure 88: Set and Reset signal diagram

If both coils are triggered at the same time, priority is given to the coil further down in the circuit diagram. This is shown in the above signal diagram in section B. Has a higher rung number(The Reset coil in the example above.)

```
I 05----------------S Q 01
I 10----------------R Q 01
```

Figure 89: Simultaneous triggering of Q 01

The above example shows the Reset coil with priority when the Set and Reset coil are triggered at the same time.

#### Coil negation (inverse contactor function) ↄ

The output signal will correspond to the inverted input signal. The relay will work like a contactor with negated contacts. If the coil is driven with a state of 1, it will switch its make contacts to a state of 0.

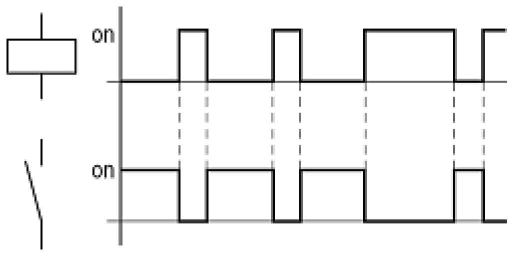



Figure 90: Inverse contactor function signal diagram

#### Evaluating a rising edge (cycle pulse)

This function is used if the coil is only meant to switch on a rising edge. When the coil status changes from 0 to 1, the coil switches its make contacts to 1 for one cycle.

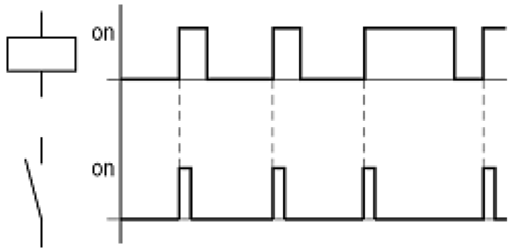



Figure 91: Signal diagram of cycle pulse with rising edge

#### Evaluating a falling edge (cycle pulse)

This function is used if the coil is only meant to switch on a falling edge. When the coil status drops out from 1 to 0, the coil switches its make contacts to 1 for one cycle.

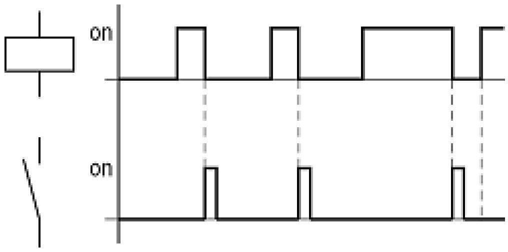


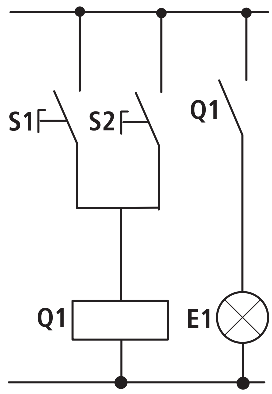

Figure 92: Signal diagram of cycle pulse with negative edge

→ A set coil is automatically switched off in the event of a power failure and in STOP mode.
Retentive coils keep their logic state.

## 5.4 Working with contacts and coils

Switches, pushbuttons and relays from a conventional hardwired circuit diagram are wired in the easyE4 circuit diagram via input contacts and relay coils.

**Hardwired**




**Wired with an easyE4 device**

easyE4 connection

Make contact S1 to input terminal I1
Make contact S2 to input terminal I2
Connect load E1 to easy output Q1

S1 or S2 switch on E1.

**easyE4 circuit diagram:**

```
I 001--------------------Q 001
I 002--┘
```

Figure 93: Circuit diagram with inputs

Circuit diagram with inputs `I 001`, `I 002` and output `Q 001`

First specify which input and output terminals you wish to use in your circuit.

The signal states on the input terminals are detected in the circuit diagram with the input contacts I, R or RN. The outputs are switched in the circuit diagram with the output relays Q, S or SN.

The jump destination has a special position for the input contacts and the jump location for the output relays. These are used for structuring a circuit diagram.

Following is a description of how to wire various contacts and coils for the various relay types and function blocks (inputs) in the circuit diagram.

### 5.4.1 Entering and modifying contacts

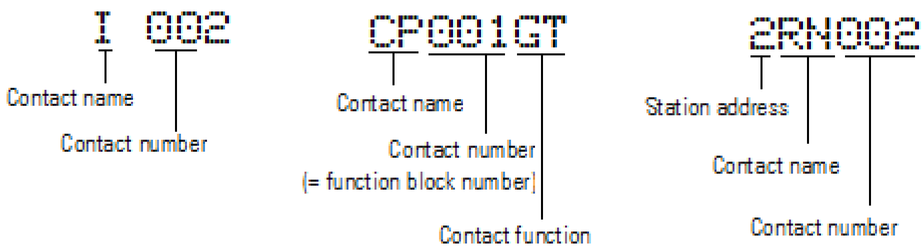



Figure 94: Contact legend

You choose an input contact in the easyE4 device by means of the contact name and the contact number.

Example: Base device input contact or function block contact consists of the abbreviated function block name, the number, and the contact function.

Example: Contact of "comparator" function block

→ How a safety or standard function block is added to the circuit diagram and parameterized is described in → Section "Function blocks", page 216.

If a NET station's contact is used in a circuit diagram, the station's NET-ID (address) will be placed before the contact name, →"Wiring another NET station's contact or coil in a circuit diagram" section, page 121.

Example: Contact of a NET station

### 5.4.2 Changing an N/O contact to an N/C contact

**DANGER**
Persons, systems and machines may be put at risk if an N/C contact is misinterpreted. When using N/C contacts in the program always evaluate the PRSNT and DIAG diagnostic bits of this module.

You can define each contact in the circuit diagram as a N/O or N/C.

- Switch to Entry mode and move the cursor over the contact name.
- Press the ALT button. The N/O contact will change to a N/C contact.
- Press the OK button 2 × to confirm the change.

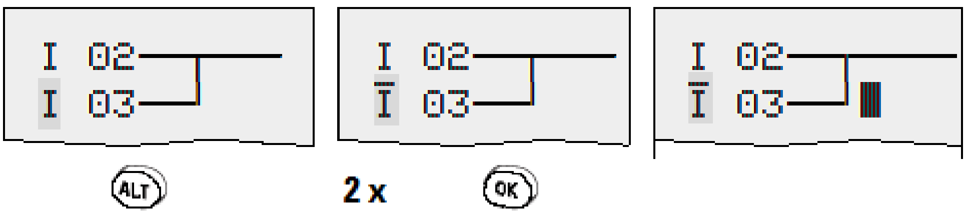



Figure 95: Change contact I 03 from N/O to N/C

Bear in mind that the active state of an N/C contact is 0. The 0 state of a contact may, however, be present if the station is missing or is operating incorrectly. The use of an N/C contact in the circuit diagram without evaluating the diagnostics bit may cause incorrect interpretations.

### 5.4.3 Entering and modifying coils

With a relay coil or a function block, you choose the coil function, coil name, coil number as well as the function block coil. With coils of an NET station, choose the address (NET ID) in front of the coil name.

→ The coil number in the figures on the left must be the same as the function block number!

**Examples**

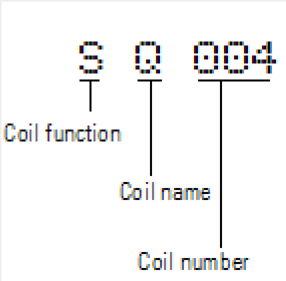



Figure 96: Relay coil "Output Q"

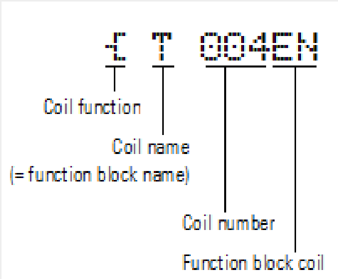



Figure 97: Relay coil for timing relay function block with control coil

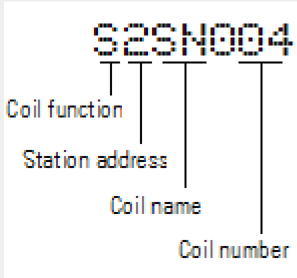



Figure 98: Relay coil of a NET station

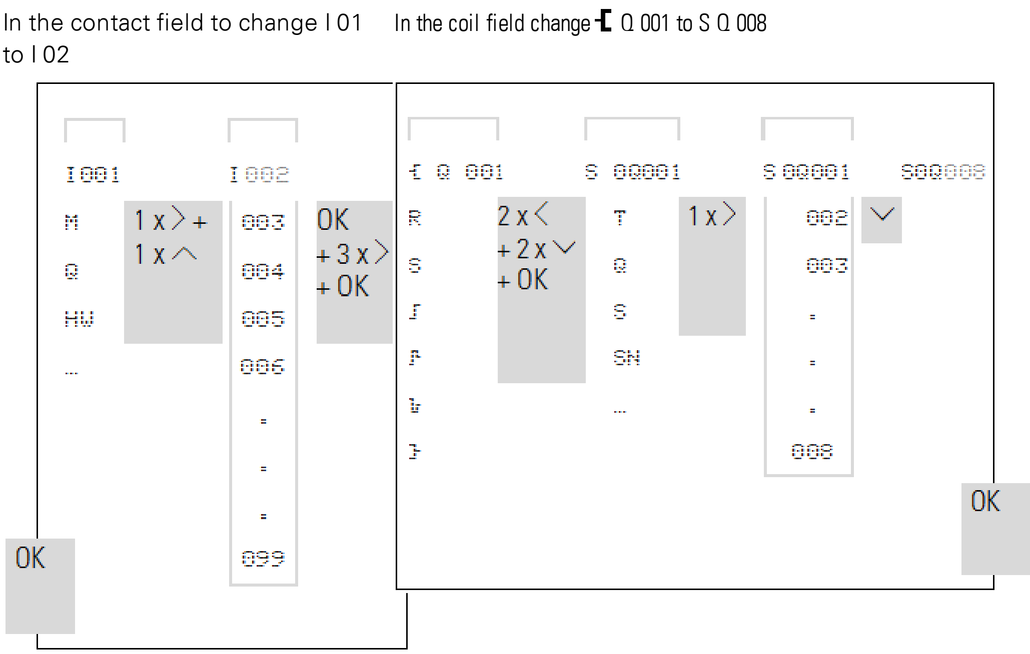
→ A complete list of all contacts and coils,
→ Section "Function blocks", page 216

The values for contact and coil fields can be changed in input mode.
The value that can be changed will flash.

I001 The easyE4 device proposes contact I 001 or the coil Ɛ Q 001 for entry in an empty field.

- Use the ⮝ ⮟ ⮜ ⮞ buttons to move the cursor to a free contact or coil field.
- Press the OK button to switch to Entry mode.
- Use ⮜ ⮞ to select the position you wish to change and press the OK to move to the next position
(a selected position is shown in light grey in the following figure).
- Use the ⮝ ⮟ cursor buttons to change the value at the position.

The easyE4 device will exit input mode as soon as you exit a contact or coil field with the ⮜ ⮞ cursor buttons or the OK button.




## 5.4.4 Deleting contacts and coils

- Use the ⊗ ⊗ ⊗ ⊗ buttons to move the cursor to a free contact or coil field.
- Press the DEL pushbutton.

The contact or the coil will be deleted, together with any connections.

### 5.4.5 Creating and modifying connections

Contacts and relay coils are connected with the wiring arrow in Connect mode. The easyE4 device displays the cursor in this mode as a wiring arrow.

- Use to move the cursor to the contact or coil field from which you wish to create a connection.

Do not position the cursor on the first contact field.
Here the ALT button has a different function (add rung).

- Press the ALT button to switch to Connect mode.
- Use the cursor buttons to move the arrow between the contact and coil fields and use the cursor buttons to move it between the rungs.
- Press the ALT button to leave Connect mode.

The easyE4 device closes the mode automatically as soon as you have moved the arrow to an occupied contact or coil field.

In a rung, the easyE4 device connects contacts and the connection to the relay coil automatically if no empty fields are between them.

Never wire backwards.

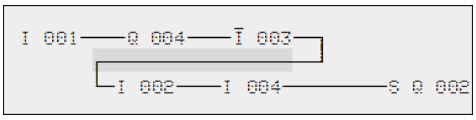



Figure 99: Circuit diagram with five contacts, invalid

When using more than four contacts in series, use one of the 96 M or 128 M markers.

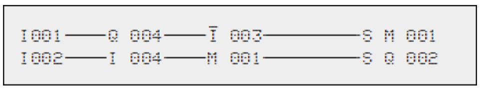



Figure 100: Circuit diagram with M marker relay

### 5.4.6 Deleting connections

- Move the cursor onto the contact field or coil field to the right of the connection that you want to delete.
- Press the ALT button to switch to Connect mode.
- Press the DEL pushbutton.

The easyE4 device deletes a connection branch.
Closed adjacent connections will be retained.

- Close the delete operation with the ALT button or by moving the cursor to a contact or coil field.

### 5.4.7 Adding a rung

The circuit diagram display shows three of the 256 rungs at the same time. Rungs outside of the display, including empty rungs, are scrolled by easyE4 automatically in the circuit diagram display if you move the cursor beyond the top or bottom of the display.

A new rung is added below the last connection or inserted above the cursor position:

- Position the cursor on the **first** contact field of a rung.
- Press the ALT button.

The existing rung with all its additional connections is “shifted” downwards. The cursor is then positioned directly in the new rung.

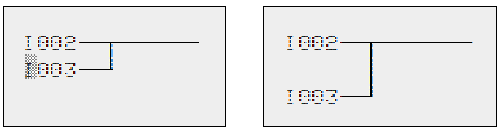



Figure 101: Inserting a new rung

### 5.4.8 Deleting a rung

The easyE4 device only removes empty rungs (without contacts or coils).

- Delete all contacts and coils from the rung.
- Position the cursor on the first contact field of the empty rung.
- Press the DEL pushbutton.

The subsequent rung(s) will be “pulled up”
and any existing links between rungs will be retained.

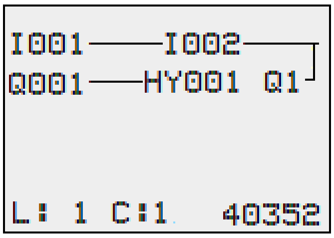
### 5.4.9 Got to a rung

The GO TO function can be used to go to a different rung.

- Press the ESC button.
- Use the ⌃ ⌄ cursor buttons to select the GO TO menu.
- Press the OK button.
- Use the ⌃ ⌄ cursor buttons to select the rung you want (L…).

Always the first contact of the rung is displayed.

- Press the OK button.




→ You can use the "GO TO" function to jump to any rung, up to the last wired rung.

### 5.4.10 Saving the circuit diagram

- Press the ESC button.

A menu will appear in the status line.

- Use the ⌃ ⌄ cursor buttons to switch to the SAVE menu.
- Press the OK button.

The entire program, circuit diagram and function blocks will be saved.

After saving, you will be returned to the previous menu from which you have opened the circuit diagram.

### 5.4.11 Exiting the circuit diagram without saving

- In order to exit a circuit diagram without saving, press ESC.

A menu will appear in the status line.

- Use the ⮝ ⮟ cursor buttons to switch to the CANCEL menu.
- Press the OK button.

The circuit diagram is closed without saving.

### 5.4.12 Searching for contacts and coils

Boolean operands or function blocks that are wired as contacts or coils can be found in the following way:

- Press the ESC button.
- Use the ⮝ ⮟ cursor buttons to switch to the SEARCH menu.
- Press the OK button.
- Use the cursor buttons ⮝ ⮟ ⮜ ⮞ to select a contact or coil as well as the required number.

For a function block you select the function block name and the number.

- Confirm the search with the OK pushbutton.

The search starts at the point where the search is activated, continues to the end of the circuit diagram. It applies only to this area.
If the required contact or coil is located above the point of calling, start the search at the beginning of the circuit diagram.
If the search is successful, you will automatically reach the required contact or coil field in the circuit diagram.

### 5.4.13 Switching with the Cursor Buttons

You can use the four cursor buttons on the easyE4 device as hardwired inputs in the circuit diagram.

The P buttons can be used for testing circuits or for manual operation. The button function is a useful addition for service and commissioning tasks.

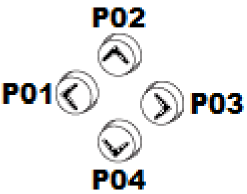



Figure 102: The cursor buttons are wired in the circuit diagram as contacts P 01 to P 04.

**Requirement**:

The P buttons must have been enabled in the system menu.

**Example 1**

This standard circuit diagram example enables a lamp at output Q1 to either be switched on or off via the inputs I1 and I2 or via the cursor buttons ∧∨.

```
I001-┬---------------------------S Q001
P002-┘
I002-┬---------------------------R Q001
P004-┘
```

Figure 103: Switch Q1 via I1, I2, ∧, or ∨

**Example 2**

This circuit diagram example causes output Q1 to be actuated by input I1. I5 switches to cursor operation and disconnects the rung I 01 via M̄ 01.

```
I005-------------------------S M001
I001----M̄ 01-┬---------------S Q001
P001----M 01-┘
```

Figure 104: I5 switches to cursor buttons.

→ The easyE4 device only evaluates P button entries if the status display is shown.

The Status menu display shows whether the P buttons are used in the circuit diagram.

Display in Status display:

- P: Button function wired and active,
- P2: Button function wired, active and P2 button ∧ pressed

- P-: Button function wired, not active,
- Empty field: P pushbuttons not used.

```
I   1. ... 6.8....
                 P2
M 0 1 4:5 5
Q 02. .6. 8      RUN
M AC: ... .........
n icht verbunden
```

### 5.4.14 Checking the circuit diagram

The easyE4 device features an integrated power flow display with which you can follow the switching states of contacts, relay and function block coils during operation. The circuit diagram display performs two functions depending on the mode:

- STOP: Create the circuit diagram.
- RUN: Power flow display.

▶ Create the small parallel circuit below and save it.

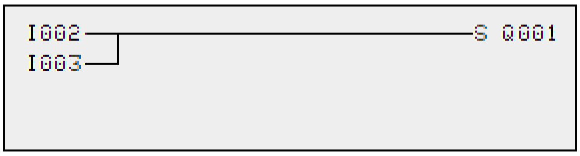



Figure 105: Paralleling link

▶ Switch easyE4 to RUN mode via the main menu.

▶ Switch back to the circuit diagram display.

You are now unable to edit the circuit diagram.

→ If you change to the circuit diagram display but cannot change a circuit diagram, first check whether the easyE4 device is in STOP mode.

▶ Switch on I3.

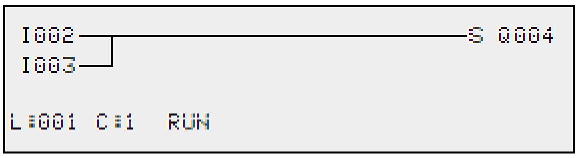



Figure 106: Power flow display

In the power flow display, energized connections are thicker than non-energized connections.

You can follow energized connections across all rungs by scrolling the display up and down.

The power flow display indicates the controller is in RUN mode.

The power flow display does not show signal changes in the millisecond range due to the inherent technical delay of LCD displays.

### 5.4.15 Jumps

Jumps can be used for structuring a circuit diagram. They can replace the function of a selector switch, for example, for Manual/Automatic mode or for different machine programs.

Jumps consist of a jump location and a jump destination (label). Jumps exist in the

- circuit diagram, for skipping rungs:
  Jump location and jump destination are located in the same circuit diagram
- Function block editor, for skipping function blocks:
  Jump location is located in the circuit diagram and jump destination in the function block editor
  The use of jumps in the function block diagram is explained in → "LB - Jump label", page 491 and → "JC - Conditional jump", page 486.

The easyE4 device allows the use of up to 32 jumps.

#### Circuit diagram elements for jumps in the circuit diagram

| | |
|---|---|
| Contact (N/O1) | |
| Numbers | 001 up to 032 |
| Coils | -[ |
| Numbers | 001 up to 032 |
| Coil function | -[, ]-, ┘, ┌, └ |
| 1) can only be used as first leftmost contact | |

#### Function of jumps

If the jump coil is triggered, the rungs after the jump coil are no longer processed. Jumps are always made forwards, i.e. the jump ends on the first contact with the same number as that of the coil.

- Coil = Jump when 1
- Contact only at the first left-hand contact position = Jump destination

The jump destination is always an N/O contact with the status 1.

→ Backward jumps cannot be executed due to the way in which easyE4 works. If the jump label does not come after the jump coil, the jump will be made to the end of the circuit diagram. The last rung is also skipped.
Multiple usage of the same jump coil and the same contact is possible as long as this is done in pairs, this means:
Coil -[:1/jumped area/contact :1,
Coil -[:1/jumped area/contact :1,
etc.

*NOTICE*
If rungs are skipped, the states of the coils are retained. The time of started timing relays continues to run.

**Power flow display of skipped area**

Jumped sections are indicated by the coils in the power flow display. All coils after the jump coil are shown with the symbol of the jump coil.

**Example for jumps**

A selector switch is used to select two different sequences.

Sequence 1: Switch on motor 1 immediately.

Sequence 2: Activate barrier 2, wait time, then switch on motor 1.

Contacts and relays used:

I1 sequence 1

I2 sequence 2

I3 guard 2 moved out

I12 motor protective circuit breaker switched on

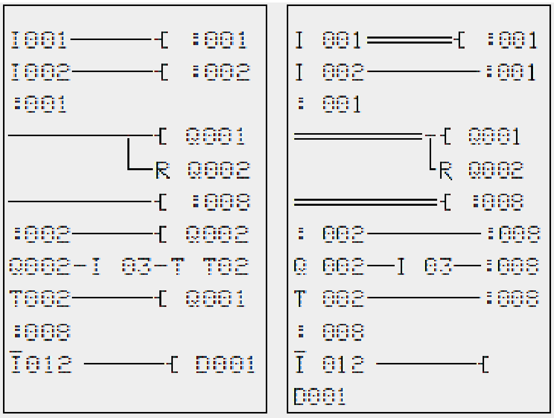
Q1 motor 1

Q2 guard 2

T 01 Wait time 30.00 s, on-delayed

D 01 Text “Motor protective circuit breaker tripped”

Circuit Diagram: Power flow display: I001 selected:




Range from jump label 1 processed.

Jump to label 8.
Range to jump label 8 skipped.

Jump label 8,
circuit diagram processed from this point on.

### 5.4.16 Wiring NET operands in the circuit diagram

A NET with several stations always allows the reading of all inputs and outputs. This is possible regardless of whether a circuit diagram is being processed or not on the NET station to be read. The inputs and outputs are addressed in the NET by using the preceding NET-ID of the station. The inputs and outputs of a NET station are identified with nI.. and nQ.

The permissible access by stations to the inputs and outputs of other stations depends on the operating mode of the devices on the NET, in which the following applications are possible:

| Operating devices on the NET | Usable NET operands of data type… | | |
|---|---|---|---|
| | Bit | Byte | 32 Bit (DWord) |
| NET marker | nN.. | nB.. | nW.., nD... |
| All NET stations each process a circuit diagram. | nI.., nR.., nQ.., nS.., nRN.., nSN... | | |
| n = NET-ID | | | |

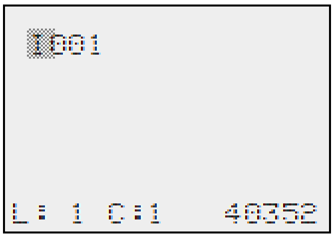
#### Wiring a contact or coil of another NET station in the circuit diagram

Prerequisites
You must have selected an I.., Q.., R.., RN.., or SN.. operand in the circuit diagram and be in input mode.

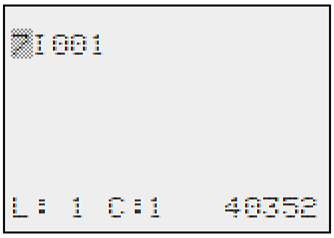
This mode is displayed by a flashing operand.

▶ Use the cursor button < to move the cursor to the position to the left of the operand.

A flashing zero appears as the initial value.




- Use the ︿ or ﹀ cursor buttons to select the NET-ID you want (NET-ID 7 in this example).
- Click on OK.




The local I.., or Q.. operand has been changed to a NET operand nI.., nR.., nQ.. and nS...

**Several NET stations with their own circuit diagram**

The relevant NET stations each process a circuit diagram.

- Every station has read access to all inputs and outputs of the other stations.
- The station only has write access to its local outputs and outputs of its local expansion unit.
  Example: Station 1 uses the state of Q1 of station 2 in its circuit diagram. Station 1, however, cannot set Q1 of station 2 to 1.
- Send NET (SN) and Receive NET( RN) is used for exchanging bits. These operands are always used in pairs.
- Put (PT) and Get (GT) are used in order to exchange double word operands via the NET.
  For more information on the manufacturer function blocks: → Section "Working with function blocks", page 191
  → Section "Function blocks", page 216

### SN-RN combination for bit exchange on the NET

- Writing via SN

The NET operand SN (Send NET) is used for sending bit data from one NET station to another. To do this you select the SN operand in a coil field.

- Reading using RN

The RN (Receive NET) NET operand is used to receive the bit data that another NET station has sent. To do this you select the RN operand in a contact field.

As the RN and SN must always be used in pairs, the following rule applies:

- on the sending station and receiving station use the same operand number for each SN/RN pair to be formed.
- in the circuit diagram of the sending station you set for the SN operand (coil) the station number (NET ID) of the receiving station.
- in the circuit diagram of the receiving station you set for the RN operand (contact) the station number (NET ID) of the sending station.

**SN-RN example**

NET station 2 sends the status of the P button P01 via SN1 to NET station 1.

| | | | |
|---|---|---|---|
| 2 slave | | 1 slave | |
| Destination Id | | Source ID | |
| 1 | SN ⟷ | 2 | RN |

The relevant circuit diagram is as follows:

```
P001 ---------------------------{1SN001
```

On NET station 1 the status of P1 is associated via RN01 as a count pulse for the counter relay C01.

```
2RN001 -------------------------{C001C_
```

### NET operand GT.. (receive), PT.. (send) and SC.. (set date and time)

The function blocks are of data type 32-bit. They will only work if the NET is working properly.→ Section "Operating system diagnostic messages", page 609

More information about the function blocks: → Section "Function blocks", page 216

### NET marker

N.., nB.., nW.., nD...

Every station that the NET marker describes can read any of the other stations.

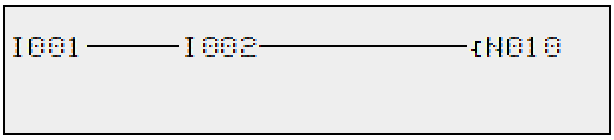



Figure 107: 1 slave

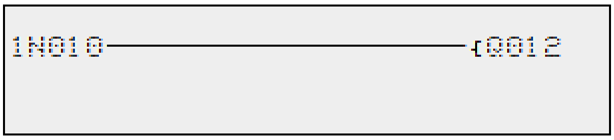



Figure 108: 2 slave

## 5.5 Transferring programs from and to the microSD memory card

easyE4 base devices can be used with a microSD memory card.

For more information on the various ways in which this memory card can be used, please refer to: → "microSD storage cards" section, page 1

Programs are normally transferred from easySoft 7 to the device so that they can be run on the device.
If the easyE4 base device has a microSD memory card, the program can also be stored on this memory card.→ Section "Automatic booting from the card", page 1

You can store multiple programs on a single memory card.
One of these programs can be set as the boot program, i.e., this boot program will be automatically transferred to the device and run as soon as there is a supply voltage present (the device is turned on) and there is no program on the device itself.

Programs can be transferred on the easyE4 device itself. They can also be transferred with easySoft 7 if the latter is connected to the easyE4.

### 5.5.1 Configuration on base devices with a display

The program needs to be transferred using the Card menu option.

In order to be able to do this, the program must be in STOP mode. If it is not, the device will point this out.

- ▶ Go to the main menu.
- ▶ Open the CARD menu option.

Table 66: *Main menu*

```
STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM-OPT.
PROGRAM
```

The device menu for the memory card will appear with additional menu options.

Table 67: *Card*

```
PROGRAM
LOG RECORDING
MANAGE SD CARD
INFORMATION
```

| | |
|---|---|
| PROGRAM | Used to manage the programs on the device |
| LOGS | Data can be written to a binary file by using the DL (Data Logger) manufacturer function block. These logs can be managed here. |
| MANAGE CARD | Used to format and eject the card |
| INFORMATION | Provides information on the card size and the amount of free space left |

#### 5.5.1.1 PROGRAM submenu

Requirement:
The following option must be enabled when creating the program in easySoft: Allow overwriting via card

You can use this submenu to manage the programs on the easyE4.

The program transfer menu offers the following options:

Table 68: *Card\Program*

```
SET BOOT PROG.
DELETE PROGR.
CARD -> DEVICE
DEVICE -> CARD
```

### STARTPROGRAM

If you select this submenu, a list with the names of all the programs stored on the memory card will appear.

Table 69: *Card\Program\Start program*

```
Name Program 1 ✓
Name Program 2
Name Program 3
...
```

The checkmark ✓ at the end of a line is used to indicate the program with which the easyE4 device will start as soon as there is a supply voltage.

→ If the display is empty, this means that no programs have been stored on the memory card.

▶ Select the starting program you want.

### DELETE PROGRAM

If you select this submenu, a list with the names of all the programs stored on the memory card will appear.

The ✓ checkmark at the end of a line indicates which program is currently set as the starting program. Meanwhile, the program that is currently selected will flash.

▶ Select the program that you want to delete.

A confirmation prompt will appear. The program will not be deleted until you select Yes and press OK as a confirmation.

#### CARD -> DEVICE

If you select this submenu, a list with the names of all the programs stored on the memory card will appear.

The ✓ checkmark at the end of a line indicates which program is currently set as the program to be transferred to the device. Meanwhile, the program that is currently selected will flash.

- Select the program that you want to transfer to the device.
- Confirm your selection by clicking the OK button.

A confirmation prompt will appear. The program will not be deleted until you select Yes and press OK as a confirmation.

#### DEVICE -> CARD

The current program will be transferred from the device to the memory card.

After you select this submenu, another menu will be offered for selection.

| | |
|---|---|
| SAVE PROG. | Overwrites the selected program with the program from the easyE4 |
| SAVE AS | Makes it possible to save the current program on the easyE4 under a new name |

#### See also

→ Section "microSD memory card", page 1
→ Section "Transferring programs from and to the microSD memory card", page 187

## 5.6 Working with function blocks

Only the EDP programming language can be used on the device (if you want to program using the LD, FBD, or ST language instead, you will need to use easySoft 7). The rest of this section goes over the basics involved in working with function blocks on the device.

Function blocks are subdivided into manufacturer function blocks, interrupt function blocks, and user function blocks.

Manufacturer function blocks, which are function blocks provided by Eaton, can be used directly on the device in the circuit diagram. Meanwhile,
interrupt function blocks and user function blocks, which you can create yourself, will only be available when using the LD, FBD, or ST programming language and can only be used by using easySoft 7 to transfer the program to the device.

For a detailed description of all available function blocks, please refer to the "Function blocks" section.

The manufacturer function blocks are used to simulate some of the devices used in conventional open-loop and closed-loop control systems. You can first use the function block in the circuit diagram and then define the ACTUAL and Setpoint parameters for the inputs and outputs in the function block editor.
Or vice versa: you create the function block in the function block editor, define the parameters and use it then in the circuit diagram. With easyE4 devices you can insert up to 255 manufacturer function blocks in the function block list.

→ easyE4 devices will not limit this number automatically. This means that you must check the maximum number of manufacturer function blocks yourself in order to avoid a function block error.

### 5.6.1 Adding function blocks to the circuit diagram for the first time

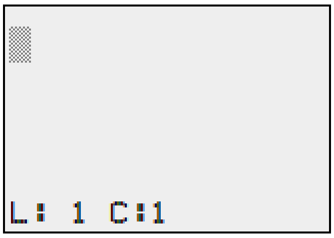
Prerequisites
In order to be able to select the *PROGRAMS* menu option, one of the two following prerequisites must be met:

- The card must contain a compiled *.PRG program that uses the EDP programming language.
- The card must not contain a compiled *.PRG program.

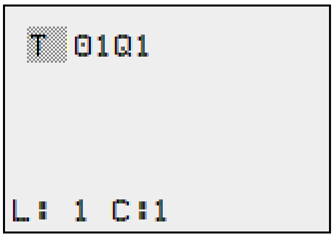
To add a function block to a circuit diagram for the first time, follow the steps below:

▶ Switch to the circuit diagram display by selecting
*Main menu -> PROGRAMS -> CIRCUIT DIAGRAM.*

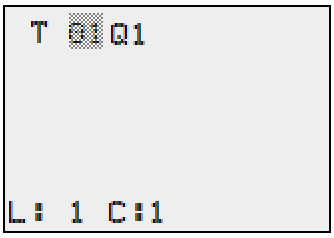
- Use the ↑ ↓ ← → buttons to move the cursor to a free contact or coil field.
- Press the OK button to switch to Entry mode.




- Then use the ↑ ↓ cursor buttons to select the function block you want (e.g., a timing relay with the T identifier).

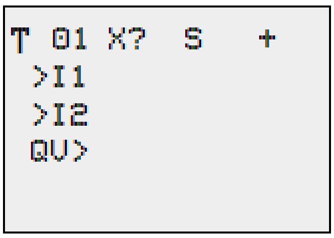



- While the function block identifier is flashing, press the OK button or the → button to move to the function block number
- Press the OK button.




The display switches to the function block editor. Here you normally define all function block parameters. As you have reached the function block editor via the circuit diagram in this case, you can only set the basic parameters.

The figure on the left shows the function block editor of the Timing relay function block.




The basic parameters may vary according to the function block. All manufacturer function blocks have the basic parameter +/-. Through the +/- symbol you can switch the parameter display on and off in operating mode RUN to allow (+) or prevent (-) changes to be made to reference points (constants). You must at least confirm the +/- character with the OK button.

→ Parameter sets can only be enabled or protected via the FUNCTION RELAYS menu, or via the circuit diagram with the “+” enable and with “–” inhibit parameter set characters.

- Use the cursor buttons ◀ ▶ to select the parameter to be changed, for example the time range “S”.
- Use the cursor buttons ▲ ▼ to change the parameter value to a different time range such as M:S.
- Press the OK button to exit the parameter dialog if you wish to save the parameters or press the ESC button, if you do not wish to parameterize the function block and add it to the circuit diagram.

After saving or canceling, the cursor returns to the position in the circuit diagram where you last left it.

In order to finish configuring the manufacturer function block (e.g., by assigning a reference value), open the function block editor as follows:

- Press the ESC button in order to save the circuit diagram with the newly added function block.
- Answer the subsequent SAVE prompt with the OK button.

The circuit diagram is saved and the easyE4 device changes to the next higher menu level.

### 5.6.2 Function block list

The function block list can be used to access the function block editor.

- Go to
  *Main menu -> PROGRAMS -> FUNCTION BLOCKS.*

This lists all function blocks that were ever used in the circuit diagram, and also those that were already deleted in the circuit diagram.
If no function blocks are shown, the list is empty.

The function block list in the example below contains the AR, CP, and T manufacturer function blocks (the manufacturer function blocks are created in the order in which they were edited).

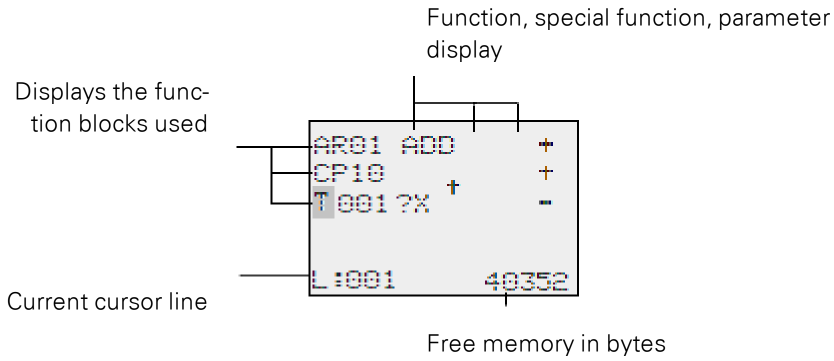
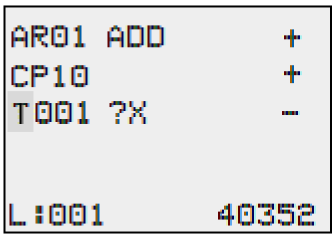



Figure 109: Explanation of the function block list

- Use the cursor buttons ⏶ ⏷ ⏴ ⏵ to select the required function block from the function block list, in this case timing relay T01.




- Confirm the selection by clicking the OK button.

The timing relay is displayed in the function block editor.

### 5.6.3 Configuring parameters in the function block editor

The function block can be fully parameterized in the function block editor.
This is accessed via the function block list.
Access is blocked if the program is password protected.

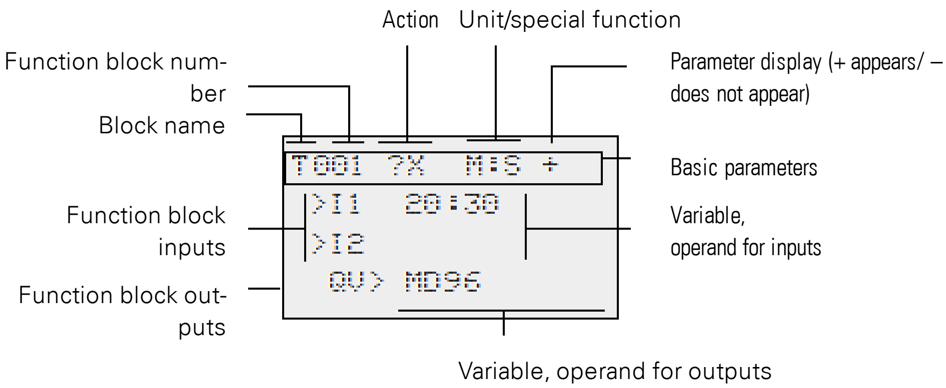



Figure 110: Manufacturer function block display in the function block editor

**Example: Timing relay function block**

| | |
|---|---|
| Function Block: | timing relay |
| Switch function: | On-delayed with random time |
| Time range: | M:S (Minute:Second) |
| Reference time >I1: | 20 min 30 s |
| Actual time QV>: | Copied to MD96 |

```
T001  ?X   M:S +
 >I1   20:30
 >I2

  QV> MD96
```

### Assigning operands at a manufacturer function block's input

The following operands can be assigned to the input of a manufacturer function block:

- Constants, e.g.: 42,
- Markers such as MD, MW, MB,
- Analog output QA,
- Analog inputs IA,
- The QV outputs of all manufacturer function blocks

You can set the parameters of the function block as follows:

▶ Use the ∧ ∨ cursor buttons to scroll through the function block inputs' constants.

▶ Change the values for a parameter set:

  ▶ OK button: Switches to input mode.

  ▶ Change whole number places with the ‹ › cursor buttons.

  ▶ Use the ∧ ∨ cursor buttons to change the value of a whole number place.

▶ OK button: Saves the constant immediately

▶ Press the ESC button to leave the parameter display.

ESC button:
Keep the previous setting and exit the parameter display.

→ Ensure that the input of a function block is not assigned impermissible values during operation.

This risk exists if you apply negative values to an input even though the corresponding function block only accepts positive values.
If, for example, the T timing relay function block is driven with a negative time reference, it will no longer switch as expected.

You should therefore take care to exclude such situations, as the easyE4 device cannot foresee these when the parameters are assigned.

If, for example, you have set the output QV of the AR arithmetic function block at input I1 of the T manufacturer function block, you should connect a CP comparator in between in order to signal the occurrence of a negative value.

In most applications, a thorough simulation is enough to prevent any impermissible values at the function block input.

#### Assigning operands at a manufacturer function block's output

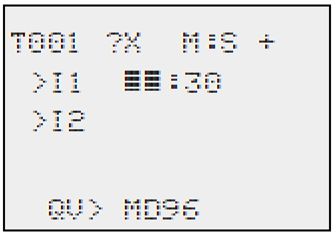
The following operands can be assigned to the output of a QV manufacturer function block:

- Markers such as MD, MW, MB,
- or the analog output QA.

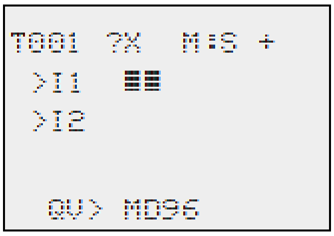
#### Deleting operands at function block inputs/outputs

Position the cursor on the required operand.

▶ Press the DEL pushbutton.




The operand is deleted.




#### Behavior of the function block editor with different operating modes

When working with the function block editor, the mode of the device is important.

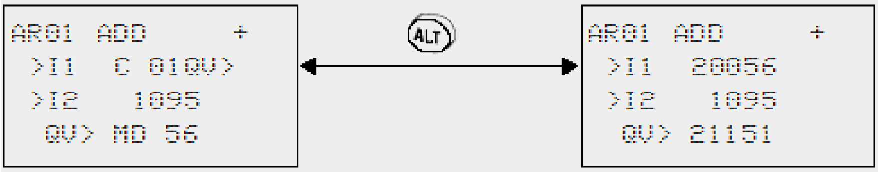
1. STOP: You will be able to access all of the manufacturer function block's parameters.
2. RUN:
   - Only access to the basic parameters is possible.
   - It is only possible to change input values at manufacturer function blocks if they are constants. The modified constants are used directly for further pro-

cessing in the circuit diagram.

- You can toggle between reference values and actual values by pressing ALT.

**Example**

- >I1= Actual value, here from the output of the counter C 01
- >I2= Constant 1095.
- QV> = Marker double word MD56.




Setpoints Actual Values

### 5.6.4 PARAMETERS menu

This menu item can only be activated in RUN mode.

Manufacturer function blocks with basic parameters that you have set to + via the +/- character in the function block editor are displayed in the PARAMETERS menu and can be changed. It is only possible to change constants. Other operands cannot be changed.

It is also possible to change parameters via the PARAMETERS menu if you have saved the program and therefore password protected the function block editor. This is the point of this menu. When the password is activated and the +/- basic parameter for each function block is set, you can allow or deny the operator of the system the possibility to change the values.

▶ Move from the Status display to the Parameters display by pressing OK -> PARAMETERS.

▶ Follow the operating steps described in → Section "Assigning operands at a manufacturer function block's input", page 195

### 5.6.5 Deleting function blocks

To remove a function block, you must remove it from the circuit diagram and from the function block list.

Requirement: The easyE4 device must be in STOP mode.

▶ Switch to the circuit diagram display by selecting *Main menu -> PROGRAMS -> CIRCUIT DIAGRAM.*

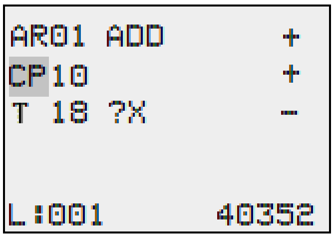
▶ Move the cursor in the circuit diagram to all the contact fields and the coil field in which the function block to be deleted is used and press the DEL button each time.

### Deleting function blocks from the function block list

In order to prevent accidental deletion, a function block continues to be managed in the function block list, even if it was already removed in the circuit diagram. To delete the function block permanently and therefore to free up more memory, you must remove it from the function block list.

- Go to
  *Main menu -> PROGRAMS -> FUNCTION BLOCKS.-> Function block list*
- Select the function block to be deleted in the function block list, in this case CP10.
- Press the DEL pushbutton.

The function block is removed from the function block list.




- Press the ESC button in order to save the function block list with the deleted function block.
- Confirm with the OK button.
- In the function block list select the required function block.

  In this example, select the data block comparator AR01 in Adder mode.
- Press the OK button.

Depending on the display selected the function block is shown with the ACTUAL values and the result, or with the set operands and constants.

If you want to switch from the operand value display to the actual value display or vice versa while checking the manufacturer function block, press the ALT button.

- Press the ALT button again.

Here are some useful tips:

### Tips for working with manufacturer function blocks

- Current ACTUAL values are deleted when you switch off the power supply or switch the easyE4 device to STOP mode.
  Exception: Retentive data keeps its state, → Section "Retention function", page 578.
  The most recent actual values are transferred to the operands every cycle. The data function block is an exception.
- If you want to prevent other people from modifying the parameters of the manufacturer function blocks, change the access enable symbol from + to – when cre-

ating the circuit diagram and setting parameters. You can then protect the circuit diagram with a password.

- Since every function block in the function block list takes up space even if it is no longer being used and has been deleted from the circuit diagram, you should clean up things every once in a while.
  Check the function block diagram for manufacturer function blocks that are no longer needed and delete them.
- The manufacturer function blocks are designed so that an output value of a function block can be assigned directly with an input of another function block. The 32-bit data format is used automatically. This also enables the transfer of negative values.

→ The following applies to RUN mode:
A easyE4 device processes the manufacturer function block after a pass through the circuit diagram. This takes the last status of the coils into account.

## 5.7 Using operands in a program

Only operands can be processed in a program. Accordingly, device input values, device output values, device P button states, diagnostic alarms, and backlight LED outputs need to be stored in operands. All operands can be mapped to markers, and markers are also considered operands. In fact, these markers can be used for bit, byte, word, and double word access in the program and for implementing simple calculations and connections.

### Elementary data types

Following is a list of the various elementary data types. These data types are independent of the programming language you select.

| Type/(description) | Length in bits | Format | Value range | Example |
|---|---|---|---|---|
| BOOL/(Bit) | 1 | Binary (bool) | 0/1, FALSE/TRUE | TRUE (1) |
| BYTE/(Byte) | 8 | Decimal number (unsigned) | 0…255 | 128 |
| WORD/(Word) | 16 | Decimal number (unsigned) | 0 - 65535 | 1023 |
| DWORD/(Double Word) | 32 | Decimal number (signed) | -2 147 483 648… +2 147 483 647 | - 65535 |

### Permissible operands at a glance

Table 70: Permissible operands

| Operand | Explanation | Data width: | Data type |
|---|---|---|---|
| I | Input | 1 bit | BOOL |
| Q | Output | 1 bit | BOOL |
| P | P buttons | 1 bit | BOOL |
| ID | Diagnostic alarm | 1 bit | BOOL |
| IA | Analog Input | 32 bit | DINT |
| QA | Analog output | 32 bit | DINT |
| M | Markers | 1 bit | BOOL |
| MB | Marker byte | 8 bit | BYTE |
| MW | Marker word | 16 bit | WORD |
| MD | Marker double word | 32 bit | DINT |
| LE | LED output | 1 bit | BOOL |
| RN | Input bit via NET (receive) | 1 bit | BOOL |
| SN | Output bit via NET (send) | 1 bit | BOOL |
| N | Network marker | 1 bit | BOOL |
| NB | Network marker byte | 8 bit | BYTE |
| NW | Network marker word | 16 bit | WORD |
| ND | Network marker double word | 32 bit | DINT |

| Use | Operand range |
|---|---|
| Local bit operands | I1…I16[1)]<br>I17…I128<br>Q1…Q16[1)]<br>Q17…Q128<br>P1…P8<br>M1…M512 (EDP: M1…M128)<br>ID1…ID24[1)]<br>ID25…ID96<br>LE1…LE3 |
| Local value operands | IA1...IA4[1)]<br>IA5...IA48<br>QA1...QA4[1)]<br>QA5...QA48<br>MB1…MB512<br>MW1…MW512<br>MD1…MD256 |
| N operands bit | N1…N512 (EDP: N1…N128)<br>xRN1…xRN32<br>xSN1…xSN32 |
| N operands value | NB1…NB64<br>NW1…NW32<br>ND1…ND16 |
| 1) base device permanently assigned | Figure 111: |

## Connection rules for operands

The following operands can be connected to the various inputs and outputs, as well as to each other, regardless of the programming language you select:

| Operators | Bit inputs | Bit outputs |
|---|---|---|
| Constant 0, constant 1 | x | x |
| M – Markers | x | x |
| RN - Input bit via NET | x | – |
| SN - Output bit via NET (send) | x | x |
| N - Network marker bit | x | x |
| nN - NET station n marker | x | x |
| ID: Diagnostic alarm | x | – |
| LE - Output backlight | x | x |
| P buttons | x | – |
| I - Bit input | x | – |
| Q - Bit output from another FB | x | x |

| Assigning operands | Value inputs | Value outputs |
|---|---|---|
| Constant | x | x |
| Markers: MB, MD, MW | x | x |

| Assigning operands | Value inputs | Value outputs |
|---|---|---|
| Analog inputs IA | x | x |
| Analog output QA | x | x |
| Numeric output from another QV FB | x | x |

## Overview of operands for numeric formats

The values of the data types marker byte (MB) and marker word (MB) are processed as unsigned. In order to store negative values, you must use a marker double word.

This fact must be particularly taken into account if the output of a function block can take on a negative value. You must temporarily store this value in a marker double word in order to transfer it to the input of a function block, otherwise the sign information will be lost.

The easyE4 device processes calculations with a signed 31-bit value.

The value range is: -2147483648 to +2147483647

In the case of a 31-bit value, the 32nd bit is the sign bit.

Bit 32 = status 0 -> positive number.

Bit 32 = status 1 -> negative number.

**Example**

$00000000000000000000010000010010_{bin} = 412_{hex} = 1042_{dez}$

$11111110110111001111010001000111_{bin} = FEDCF447_{hex} = -19073977_{dez}$

## Organizing marker ranges

The term "marker" is used to represent marker bits (M). Marker bits (M) are used to store the Boolean states 0 or 1. A marker bit is also called an auxiliary relay.

easyE4 devices also manage the marker bits in marker bytes (MB), marker words (MW) and in marker double words (MD). A marker byte consists of 8 marker bits, a marker word of 16 marker bits and a marker double word of 32 marker bits.

In order to store the state for a contact, you can use a specific bit and, accordingly a specific byte as well. For instance, marker bit 9 is included in marker byte 2, marker word 1, and marker double word 1. You can use the following conversion table to determine which word contains a bit or which bits encompass a specific double word.

Bear in mind that after the division, a rounding up to the next higher integer is necessary, even if the decimal number is below 0.5.

The easyE4 has 1024 bytes available for data storage.
This data memory can be accessed by bit, byte, word, or double word.

This means that four different operands with their own address can be used to access the exact same data range. Accordingly, it is extremely important to pay close attention to each operand's address in order to avoid accidental duplicate access.

Markers can be used to access data as follows (with the corresponding address range):

- M 1...512
- MB 1...512
- MW 1...512
- MD1...256

Avoid accidental double marker assignments.
Otherwise you might address the 512 bit markers simultaneously via the first 64 marker bytes, 32 marker words or 16 marker double words and thus produce uncontrollable states. When write accesses are made successively within an MD, such as to MD1, MW2, MB4 or M32, the last write operation is retained.

Observing the following wiring rules will prevent the double assignment of marker bits.
For easyE4, use:
Marker bytes, starting at MB13
Marker words, starting at MW07
Marker double words, starting at MD04

The following operand tables show how this works.

## Operand table

To read the operand table, observe the following information:

The most significant marker bit, byte, word or double word is located on the left and the least significant on the right. Only double words have a sign bit. The other data formats do not.

Example 1: Bit81 is contained in MB11, MW6 and DW3.

Example 2: Byte21 is contained in MW11 and DW6 and contains bits Bit161 to Bit168.

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 128…-121 | 120…-113 | 112…-105 | 104…-97 | 96…8-9 | 88…8-1 | 80…7-3 | 72…6-5 | 64…5-7 | 56…4-9 | 48…4-1 | 40…3-3 | 32…2-5 | 24…1-7 | 16…9 | 8…1 |
| Byte | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| Wor-d | 8 | | 7 | | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | |
| DW-ord | 4 | | | | 3 | | | | 2 | | | | 1 | | | |
| | | | | | | | | | | | | | | | | |
| Bit | 256…-249 | 248…-241 | 240…-233 | 232…-225 | 224…-217 | 216…-209 | 208…-201 | 200…-193 | 192…-185 | 184…-177 | 176…-169 | 168…-161 | 160…-153 | 152…-145 | 144…-137 | 136…-129 |
| Byte | 32 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 |
| Wor-d | 16 | | 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | |
| DW-ord | 8 | | | | 7 | | | | 6 | | | | 5 | | | |
| | | | | | | | | | | | | | | | | |
| Bit | 384…-377 | 376…-369 | 368…-361 | 360…-353 | 352…-345 | 344…-337 | 336…-329 | 328…-321 | 320…-313 | 312…-305 | 304…-297 | 296…-289 | 288…-281 | 280…-273 | 272…-265 | 264…-257 |
| Byte | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 |
| Wor-d | 24 | | 23 | | 22 | | 21 | | 20 | | 19 | | 18 | | 17 | |
| DW-ord | 12 | | | | 11 | | | | 10 | | | | 9 | | | |
| | | | | | | | | | | | | | | | | |
| Bit | 512…-505 | 504…-497 | 496…-489 | 488…-481 | 480…-473 | 472…-465 | 464…-457 | 456…-449 | 448…-441 | 440…-433 | 432…-425 | 424…-417 | 416…-409 | 408…-401 | 400…-393 | 392…-385 |
| Byte | 64 | 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 |
| Wor-d | 32 | | 31 | | 30 | | 29 | | 28 | | 27 | | 26 | | 25 | |
| DW-ord | 16 | | | | 15 | | | | 14 | | | | 13 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 80 | 79 | 78 | 77 | 76 | 75 | 74 | 73 | 72 | 71 | 70 | 69 | 68 | 67 | 66 | 65 |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Word | 40 | | 39 | | 38 | | 37 | | 36 | | 35 | | 34 | | 33 | |
| DWord | 20 | | | | 19 | | | | 18 | | | | 17 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 96 | 95 | 94 | 93 | 92 | 91 | 90 | 89 | 88 | 87 | 86 | 85 | 84 | 83 | 82 | 81 |
| Word | 48 | | 47 | | 46 | | 45 | | 44 | | 43 | | 42 | | 41 | |
| DWord | 24 | | | | 23 | | | | 22 | | | | 21 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 112 | 111 | 110 | 109 | 108 | 107 | 106 | 105 | 104 | 103 | 102 | 101 | 100 | 99 | 98 | 97 |
| Word | 56 | | 55 | | 54 | | 53 | | 52 | | 51 | | 50 | | 49 | |
| DWord | 28 | | | | 27 | | | | 26 | | | | 25 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 128 | 127 | 126 | 125 | 124 | 123 | 122 | 121 | 120 | 119 | 118 | 117 | 116 | 115 | 114 | 113 |
| Word | 64 | | 63 | | 62 | | 61 | | 60 | | 59 | | 58 | | 57 | |
| DWord | 32 | | | | 31 | | | | 30 | | | | 29 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 144 | 143 | 142 | 141 | 140 | 139 | 138 | 137 | 136 | 135 | 134 | 133 | 132 | 131 | 130 | 129 |
| Word | 72 | | 71 | | 70 | | 69 | | 68 | | 67 | | 66 | | 65 | |
| DWord | 36 | | | | 35 | | | | 34 | | | | 33 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 160 | 159 | 158 | 157 | 156 | 155 | 154 | 153 | 152 | 151 | 150 | 149 | 148 | 147 | 146 | 145 |
| Word | 80 | | 79 | | 78 | | 77 | | 76 | | 75 | | 74 | | 73 | |
| DWord | 40 | | | | 39 | | | | 38 | | | | 37 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 176 | 175 | 174 | 173 | 172 | 171 | 170 | 169 | 168 | 167 | 166 | 165 | 164 | 163 | 162 | 161 |
| Word | 88 | | 87 | | 86 | | 85 | | 84 | | 83 | | 82 | | 81 | |
| DWord | 44 | | | | 43 | | | | 42 | | | | 41 | | | |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | |
| Byte | 192 | 191 | 190 | 189 | 188 | 187 | 186 | 185 | 184 | 183 | 182 | 181 | 180 | 179 | 178 | 177 |
| Word | 96 | | 95 | | 94 | | 93 | | 92 | | 91 | | 90 | | 89 | |
| DWord | 48 | | | | 47 | | | | 46 | | | | 45 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 208 | 207 | 206 | 205 | 204 | 203 | 202 | 201 | 200 | 199 | 198 | 197 | 196 | 195 | 194 | 193 |
| Word | 104 | | 103 | | 102 | | 101 | | 100 | | 99 | | 98 | | 97 | |
| DWord | 52 | | | | 51 | | | | 50 | | | | 49 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 224 | 223 | 222 | 221 | 220 | 219 | 218 | 217 | 216 | 215 | 214 | 213 | 212 | 211 | 210 | 209 |
| Word | 112 | | 111 | | 110 | | 109 | | 108 | | 107 | | 106 | | 105 | |
| DWord | 56 | | | | 55 | | | | 54 | | | | 53 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 240 | 239 | 238 | 237 | 236 | 235 | 234 | 233 | 232 | 231 | 230 | 229 | 228 | 227 | 226 | 225 |
| Word | 120 | | 119 | | 118 | | 117 | | 116 | | 115 | | 114 | | 113 | |
| DWord | 60 | | | | 59 | | | | 58 | | | | 57 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 256 | 255 | 254 | 253 | 252 | 251 | 250 | 249 | 248 | 247 | 246 | 245 | 244 | 243 | 242 | 241 |
| Word | 128 | | 127 | | 126 | | 125 | | 124 | | 123 | | 122 | | 121 | |
| DWord | 64 | | | | 63 | | | | 62 | | | | 61 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 272 | 271 | 270 | 269 | 268 | 267 | 266 | 265 | 264 | 263 | 262 | 261 | 260 | 259 | 258 | 257 |
| Word | 136 | | 135 | | 134 | | 133 | | 132 | | 131 | | 130 | | 129 | |
| DWord | 68 | | | | 67 | | | | 66 | | | | 65 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 288 | 287 | 286 | 285 | 284 | 283 | 282 | 281 | 280 | 279 | 278 | 277 | 276 | 275 | 274 | 273 |
| Word | 144 | | 143 | | 142 | | 141 | | 140 | | 139 | | 138 | | 137 | |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DW-ord | 72 | | | | 71 | | | | 70 | | | | 69 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 304 | 303 | 302 | 301 | 300 | 299 | 298 | 297 | 296 | 295 | 294 | 293 | 292 | 291 | 290 | 289 |
| Word | 152 | | 151 | | 150 | | 149 | | 148 | | 147 | | 146 | | 145 | |
| DW-ord | 76 | | | | 75 | | | | 74 | | | | 73 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 320 | 319 | 318 | 317 | 316 | 315 | 314 | 313 | 312 | 311 | 310 | 309 | 308 | 307 | 306 | 305 |
| Word | 160 | | 159 | | 158 | | 157 | | 156 | | 155 | | 154 | | 153 | |
| DW-ord | 80 | | | | 79 | | | | 78 | | | | 77 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 336 | 335 | 334 | 333 | 332 | 331 | 330 | 329 | 328 | 327 | 326 | 325 | 324 | 323 | 322 | 321 |
| Word | 168 | | 167 | | 166 | | 165 | | 164 | | 163 | | 162 | | 161 | |
| DW-ord | 84 | | | | 83 | | | | 82 | | | | 81 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 352 | 351 | 350 | 349 | 348 | 347 | 346 | 345 | 344 | 343 | 342 | 341 | 340 | 339 | 338 | 337 |
| Word | 176 | | 175 | | 174 | | 173 | | 172 | | 171 | | 170 | | 169 | |
| DW-ord | 88 | | | | 87 | | | | 86 | | | | 85 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 368 | 367 | 366 | 365 | 364 | 363 | 362 | 361 | 360 | 359 | 358 | 357 | 356 | 355 | 354 | 353 |
| Word | 184 | | 183 | | 182 | | 181 | | 180 | | 179 | | 178 | | 177 | |
| DW-ord | 92 | | | | 91 | | | | 90 | | | | 89 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 384 | 383 | 382 | 381 | 380 | 379 | 378 | 377 | 376 | 375 | 374 | 373 | 372 | 371 | 370 | 369 |
| Word | 192 | | 191 | | 190 | | 189 | | 188 | | 187 | | 186 | | 185 | |
| DW-ord | 96 | | | | 95 | | | | 94 | | | | 93 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 400 | 399 | 398 | 397 | 396 | 395 | 394 | 393 | 392 | 391 | 390 | 389 | 388 | 387 | 386 | 385 |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Wor-d | 200 | | 199 | | 198 | | 197 | | 196 | | 195 | | 194 | | 193 | |
| DW-ord | 100 | | | | 99 | | | | 98 | | | | 97 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 416 | 415 | 414 | 413 | 412 | 411 | 410 | 409 | 408 | 407 | 406 | 405 | 404 | 403 | 402 | 401 |
| Wor-d | 208 | | 207 | | 206 | | 205 | | 204 | | 203 | | 202 | | 201 | |
| DW-ord | 104 | | | | 103 | | | | 102 | | | | 101 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 432 | 431 | 430 | 429 | 428 | 427 | 426 | 425 | 424 | 423 | 422 | 421 | 420 | 419 | 418 | 417 |
| Wor-d | 216 | | 215 | | 214 | | 213 | | 212 | | 211 | | 210 | | 209 | |
| DW-ord | 108 | | | | 107 | | | | 106 | | | | 105 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 448 | 447 | 446 | 445 | 444 | 443 | 442 | 441 | 440 | 439 | 438 | 437 | 436 | 435 | 434 | 433 |
| Wor-d | 224 | | 223 | | 222 | | 221 | | 220 | | 219 | | 218 | | 217 | |
| DW-ord | 112 | | | | 111 | | | | 110 | | | | 109 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 464 | 463 | 462 | 461 | 460 | 459 | 458 | 457 | 456 | 455 | 454 | 453 | 452 | 451 | 450 | 449 |
| Wor-d | 232 | | 231 | | 230 | | 229 | | 228 | | 227 | | 226 | | 225 | |
| DW-ord | 116 | | | | 115 | | | | 114 | | | | 113 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 480 | 479 | 478 | 477 | 476 | 475 | 474 | 473 | 472 | 471 | 470 | 469 | 468 | 467 | 466 | 465 |
| Wor-d | 240 | | 239 | | 238 | | 237 | | 236 | | 235 | | 234 | | 233 | |
| DW-ord | 120 | | | | 119 | | | | 118 | | | | 117 | | | |
| | | | | | | | | | | | | | | | | |
| Byte | 496 | 495 | 494 | 493 | 492 | 491 | 490 | 489 | 488 | 487 | 486 | 485 | 484 | 483 | 482 | 481 |
| Wor-d | 248 | | 247 | | 246 | | 245 | | 244 | | 243 | | 242 | | 241 | |
| DW-ord | 124 | | | | 123 | | | | 122 | | | | 121 | | | |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Byte | 512 | 511 | 510 | 509 | 508 | 507 | 506 | 505 | 504 | 503 | 502 | 501 | 500 | 499 | 498 | 497 |
| Word | 256 | | 255 | | 254 | | 253 | | 252 | | 251 | | 250 | | 249 | |
| DWord | 128 | | | | 127 | | | | 126 | | | | 125 | | | |
| | | | | | | | | | | | | | | | | |
| Word | 264 | | 263 | | 262 | | 261 | | 260 | | 259 | | 258 | | 257 | |
| DWord | 132 | | | | 131 | | | | 130 | | | | 129 | | | |
| | | | | | | | | | | | | | | | | |
| Word | 272 | | 271 | | 270 | | 269 | | 268 | | 267 | | 266 | | 265 | |
| DWord | 136 | | | | 135 | | | | 134 | | | | 133 | | | |
| | | | | | | | | | | | | | | | | |
| Word | 280 | | 279 | | 278 | | 277 | | 276 | | 275 | | 274 | | 273 | |
| DWord | 140 | | | | 139 | | | | 138 | | | | 137 | | | |
| | | | | | | | | | | | | | | | | |
| Word | 288 | | 287 | | 286 | | 285 | | 284 | | 283 | | 282 | | 281 | |
| DWord | 144 | | | | 143 | | | | 142 | | | | 141 | | | |
| | | | | | | | | | | | | | | | | |
| Word | 296 | | 295 | | 294 | | 293 | | 292 | | 291 | | 290 | | 289 | |
| DWord | 148 | | | | 147 | | | | 146 | | | | 145 | | | |
| | | | | | | | | | | | | | | | | |
| Word | 304 | | 303 | | 302 | | 301 | | 300 | | 299 | | 298 | | 297 | |
| DWord | 152 | | | | 151 | | | | 150 | | | | 149 | | | |
| | | | | | | | | | | | | | | | | |
| Word | 312 | | 311 | | 310 | | 309 | | 308 | | 307 | | 306 | | 305 | |
| DWord | 156 | | | | 155 | | | | 154 | | | | 153 | | | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Word | 320 | 319 | 318 | 317 | 316 | 315 | 314 | 313 |
| DWord | 160 | | 159 | | 158 | | 157 | |
| | | | | | | | | |
| Word | 328 | 327 | 326 | 325 | 324 | 323 | 322 | 321 |
| DWord | 164 | | 163 | | 162 | | 161 | |
| | | | | | | | | |
| Word | 336 | 335 | 334 | 333 | 332 | 331 | 330 | 329 |
| DWord | 168 | | 167 | | 166 | | 165 | |
| | | | | | | | | |
| Word | 344 | 343 | 342 | 341 | 340 | 339 | 338 | 337 |
| DWord | 172 | | 171 | | 170 | | 169 | |
| | | | | | | | | |
| Word | 352 | 351 | 350 | 349 | 348 | 347 | 346 | 345 |
| DWord | 176 | | 175 | | 174 | | 173 | |
| | | | | | | | | |
| Word | 360 | 359 | 358 | 357 | 356 | 355 | 354 | 353 |
| DWord | 180 | | 179 | | 178 | | 177 | |
| | | | | | | | | |
| Word | 368 | 367 | 366 | 365 | 364 | 363 | 362 | 361 |
| DWord | 184 | | 183 | | 182 | | 181 | |
| | | | | | | | | |
| Word | 376 | 375 | 374 | 373 | 372 | 371 | 370 | 369 |
| DWord | 188 | | 187 | | 186 | | 185 | |
| | | | | | | | | |
| Wor- | 384 | 383 | 382 | 381 | 380 | 379 | 378 | 377 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| d | | | | | | | | |
| DW-ord | 192 | | 191 | | 190 | | 189 | |
| | | | | | | | | |
| Word | 392 | 391 | 390 | 389 | 388 | 387 | 386 | 385 |
| DW-ord | 196 | | 195 | | 194 | | 193 | |
| | | | | | | | | |
| Word | 400 | 399 | 398 | 397 | 396 | 395 | 394 | 393 |
| DW-ord | 200 | | 199 | | 198 | | 197 | |
| | | | | | | | | |
| Word | 408 | 407 | 406 | 405 | 404 | 403 | 402 | 401 |
| DW-ord | 204 | | 203 | | 202 | | 101 | |
| | | | | | | | | |
| Word | 416 | 415 | 414 | 413 | 412 | 411 | 410 | 409 |
| DW-ord | 208 | | 207 | | 206 | | 205 | |
| | | | | | | | | |
| Word | 424 | 423 | 422 | 421 | 420 | 419 | 418 | 417 |
| DW-ord | 212 | | 211 | | 210 | | 209 | |
| | | | | | | | | |
| Word | 432 | 431 | 430 | 429 | 428 | 427 | 426 | 425 |
| DW-ord | 216 | | 215 | | 214 | | 213 | |
| | | | | | | | | |
| Word | 440 | 439 | 438 | 437 | 436 | 435 | 434 | 433 |
| DW-ord | 220 | | 219 | | 218 | | 217 | |
| | | | | | | | | |
| Word | 448 | 447 | 446 | 445 | 444 | 443 | 442 | 441 |
| DW- | 224 | | 223 | | 222 | | 221 | |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ord | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | |
| Wor-d | 456 | | 455 | | 454 | | 453 | | 452 | | 451 | | 450 | | 449 | |
| DW-ord | 228 | | | | 227 | | | | 226 | | | | 225 | | | |
| | | | | | | | | | | | | | | | | |
| Wor-d | 464 | | 463 | | 462 | | 461 | | 460 | | 459 | | 458 | | 457 | |
| DW-ord | 232 | | | | 231 | | | | 230 | | | | 229 | | | |
| | | | | | | | | | | | | | | | | |
| Wor-d | 472 | | 471 | | 470 | | 469 | | 468 | | 467 | | 466 | | 465 | |
| DW-ord | 236 | | | | 235 | | | | 234 | | | | 233 | | | |
| | | | | | | | | | | | | | | | | |
| Wor-d | 480 | | 479 | | 478 | | 477 | | 476 | | 475 | | 474 | | 473 | |
| DW-ord | 240 | | | | 239 | | | | 238 | | | | 237 | | | |
| | | | | | | | | | | | | | | | | |
| Wor-d | 488 | | 487 | | 486 | | 485 | | 484 | | 483 | | 482 | | 481 | |
| DW-ord | 244 | | | | 243 | | | | 242 | | | | 241 | | | |
| | | | | | | | | | | | | | | | | |
| Wor-d | 496 | | 495 | | 494 | | 493 | | 492 | | 491 | | 490 | | 489 | |
| DW-ord | 248 | | | | 247 | | | | 246 | | | | 245 | | | |
| | | | | | | | | | | | | | | | | |
| Wor-d | 504 | | 503 | | 502 | | 501 | | 500 | | 499 | | 498 | | 497 | |
| DW-ord | 252 | | | | 251 | | | | 250 | | | | 249 | | | |
| | | | | | | | | | | | | | | | | |
| Wor-d | 512 | | 511 | | 510 | | 509 | | 508 | | 507 | | 506 | | 505 | |
| DW-ord | 256 | | | | 255 | | | | 254 | | | | 253 | | | |
| | | | | | | | | | | | | | | | | |

## Retentive markers

You can declare a freely definable contiguous range between marker bytes as retentive.

| Device | Marker range that can be declared as retentive |
|---|---|
| easyE4 | MB01 - MB400 |

For information on how to configure markers so as to store data in a non-volatile manner, please refer to → Section "Retention function", page 578.

## Internal marker ranges in function blocks

Function blocks with a main program that can contain subroutines need to provide their own marker ranges for the program as well. These marker ranges cannot be accessed externally. Following are the function blocks that have their own marker ranges:

UF, IC, IE, IT

| function block | Marker range |
|---|---|
| UF | 16 marker double words |
| IE<br>IC<br>IT | 32 marker bits |

Available operands for UF - user function block

IC - Counter-controlled interrupt function block Interaction between main program and interrupt program

IE - Edge-controlled interrupt function block

IT - Time-controlled interrupt function block Interaction between main program and interrupt program

**See also**

Cross-reference list - Finding operands in a program

# 6. Function blocks

Function blocks offer pre-defined solutions for frequently occurring programming tasks. Whether a function block is available or not will depend on the programming language you select and the firmware version used in the project.

**Manufacturer function blocks**

Manufacturer function blocks are available in easySoft 7 and directly on the device


Timer modules

- HW - 7-day time switch (Hour Week) → page 218
- HY - Year time switch (Hora Year) → page 268
- OT - Operating hours counter → page 238
- RC - Real-time clock → page 242
- T - Timing relay → page 246
- WT - Weekly timer (WeekTable) → page 268
- YT - Year time switch (Year Table) → page 261
- AC - Astronomic clock → page 272

Counter Function Blocks

- C - Counter relay → page 281
- CF - Frequency counter → page 287
- CH - High-speed counter → page 293
- CI - Incremental Counter → page 299

Arithmetic and analog function blocks

- A - Analog value comparator → page 306
- AR - Arithmetic → page 312
- AV - Average → page 317
- CP – Comparator → page 326
- LS - Value scaling → page 330
- MM - Min-/Max function → page 335
- PM - Performance map → page 339
- PW - Pulse width modulation → page 345

Open-loop and closed-loop function blocks

- DC - PID controller → page 351
- FT - PT1-Signal smoothing filter → page 358
- PO - Pulse output → page 364
- TC - Three step controller → page 379
- VC - Value limitation → page 384

Data and register function blocks

- BC - Block compare → page 388
- BT - Block transfer → page 395

DB - Data function block → page 401
MX - Data multiplexer → page 406
RE - Recipe records → page 410
SR - Shift register → page 415
TB - Table function → page 423
NET Function Blocks
GT - Get values from NET → page 428
PT - Put values to NET → page 432
SC - Synchronizing clock via NET → page 436
Other function blocks
AL - Alarm function block → page 440
BV - Boolean operation → page 444
D - Text display → page 448
D - Text display editor → page 458
DL - Data logger → page 475
JC - Conditional jump → page 486
MR - MasterReset → page 493
NC - Numerical converter → page 497
ST - Set cycle time → page 503


**interrupt function blocks**

Interrupt function blocks are available only in easySoft 7


IC - Counter-controlled interrupt → page 506
IE - Edge-controlled interrupt → page 519
IT - Time-controlled interrupt function block → page 525


**User function blocks – used to create custom function blocks**

User function blocks are only available in easySoft 7


UF - User function block → page 533

# 6.1 Manufacturer function blocks

## 6.1.1 Timer modules

### 6.1.1.1 HW - 7-day time switch (Hour Week)

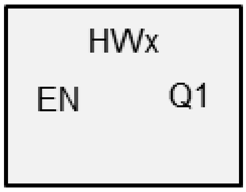
easyE4 devices feature a real-time clock with a date and time functionality.
When combined with the HW, HY or WT, YT function blocks, this real-time clock makes it possible to implement the functionality of a weekly timer and year time switch.

→ Section "Time and Date setting", page 587

The AC manufacturer function block, Astronomic clock, can be used to program switching operations based on sunrise and sunset times. In order for this to work properly, the settings for the device clock and the device location's time zone and geographic coordinates must be correctly selected in this tab.

#### General

easyE4 Base devices provide 32 weekly timer HW01…HW32 (Hour Week).
Each weekly timer provides 4 channels. These channels all act jointly on the function block output Q1 of the weekly timer.




#### Operating principle

Each of the 32 weekly timer, HW01 through HW32, features four channels that can each be configured with an ON event and an OFF event in the parameter configuration for the block. All channels act jointly on function block output Q1.

The following abbreviations are used for the individual days of the week:
Monday = Mon, Tuesday = Tue, Wednesday = Wed, Thursday = Thur, Fri = Fri, Saturday = Sat, Sunday = Sun.

#### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |

### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: if the on condition is fulfilled. | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers [2)]<br>NET stations n | x |

| Assigning operands | Value outputs |
|---|---|
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Channel A - D | A maximum of four channels can be configured (all four channels will act on output Q1).<br>There is an ON time and an OFF time for each channel. In addition, you can select one or two days of the week when these times will apply. | Note: If the off time is earlier than the on time, the control relay will not switch off until the following day. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Edit interrupt routine | Clicking on this button will open the interrupt routine | |
| Simulation possible | | |

### Parameter definition at the function block

If + Call enabled is selected for the function block under *Function block diagram/Parameters/*, it will be possible to change the switching times in the PARAMETER menu on the device while in RUN / STOP mode.

The time to be entered must be between 00:00 and 23:59.

Table 71: Incomplete and automatic parameter definition at the function block

| Day | Hour | Minute | Result |
|---|---|---|---|
| - | - | - | A switch point will not be set if you have not set the weekday or the time.<br>Device display: -- --:-- |
| DY1 e.g. Mo | - | - | If you only set the weekday for the On time, the programming software will automatically set the Hour and Minute values to 00. The contact remains active, if the Off Time has not been set.<br>Device display example: Mo 00:00/-- --:-- |
| DY2 e.g. Fr | - | - | If you only set the weekday for the Off time, the programming software will automatically set the Weekday for the On time to Sunday and Hour and Minute values to 00.<br>Device display: Su 00:00/Fr --:-- |

DYx = Weekday

It is therefore not possible to enter the time only. If you delete the weekday (DEL button) whilst making an entry or during operation or simulation, the time will be deleted automatically. Entering the time automatically causes Sunday to be entered as the default weekday.

### Other

### Retention

The function block does not recognize retentive data.

### Behavior in the event of a power failure

The time is backed up and refreshed in the event of a power supply failure. In this case, the time switches no longer switch and the contacts are kept open, Q1=0.
Information on the battery back-up time are provided on → Section "Back-upof real-time clock", page 702

→ After being switched on, the control relay will always update its switching state based on all existing switching time settings and will switch Q1 accordingly.

Example 1: Daily on/off switching

(channel A ON - FR 10:00; OFF - SU 18:00)

If the function block output Q1 is to switch on and off daily for a certain number of weekdays, use one channel.

- For one channel define at DY1 the weekday and at ON the time for the initial on switching.
- Then in the same channel define at DY2 the weekday and at OFF the time for the last off switching.

The time switch is required to switch from 10:00 to 18:00 from Fridays to Sundays.

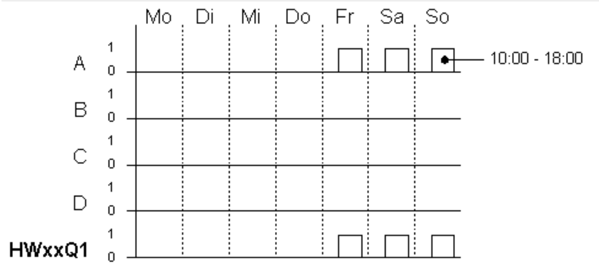



Figure 112: Signal diagram

The HW time switch must be assigned the following parameters:

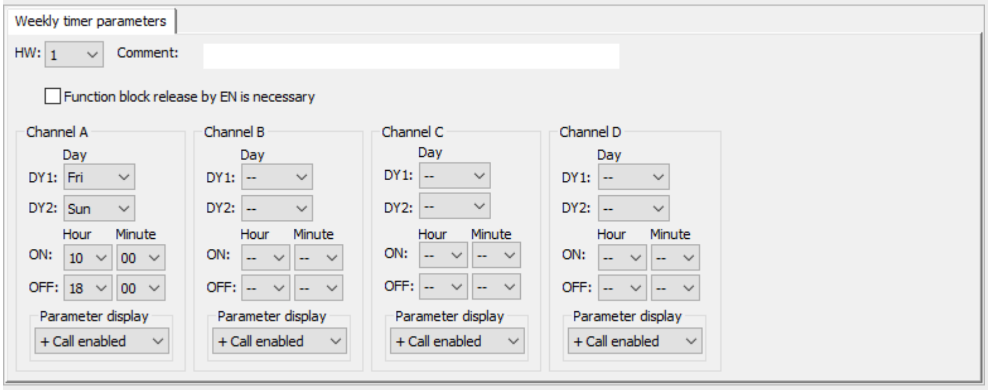



Figure 113: Tab with parameters in the Programming view

Example 2: Switching at specific times

The time switch is required to switch from Mondays to Fridays between 6:30 and 9:00 and between 17:00 and 22:30.

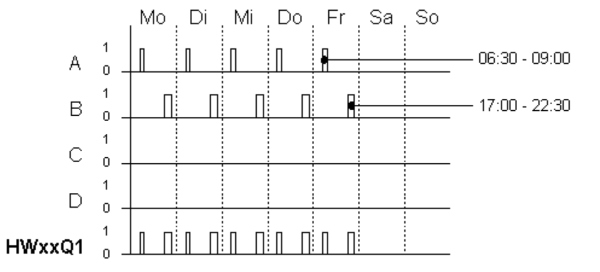



Figure 114: Signal diagram

The HW time switch must be assigned the following parameters:

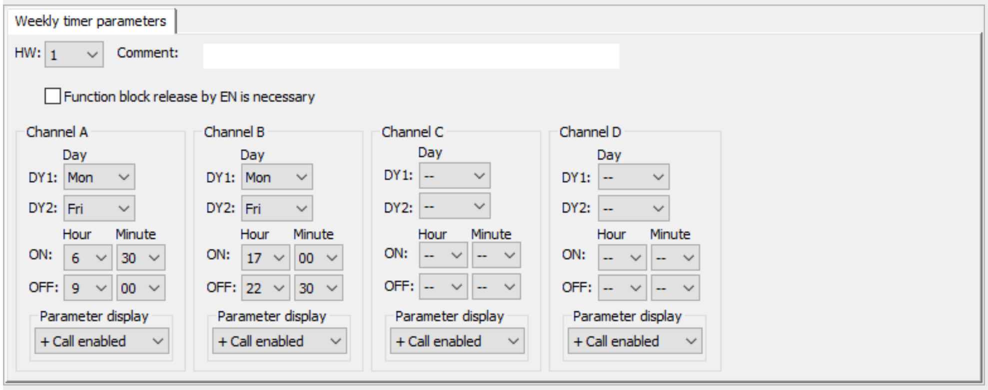



Figure 115: Tab with parameters in the Programming view

Example 3: Switching on one day and off switching on another day

If the contact Q1 is to remain switched on for a certain number of weekdays, use two channels.

- For one channel define at DY1 the weekday and at ON the time for the initial on switching. DY2 and OFF are left without any parameters for this first channel.
- Then in the next channel define at DY1 the weekday and at OFF the time for the switch-off. DY2 and ON are without any parameters for this second channel.

The time switch is required to switch on at 18:00 on Tuesdays and switch off at 6:00 on Saturdays.

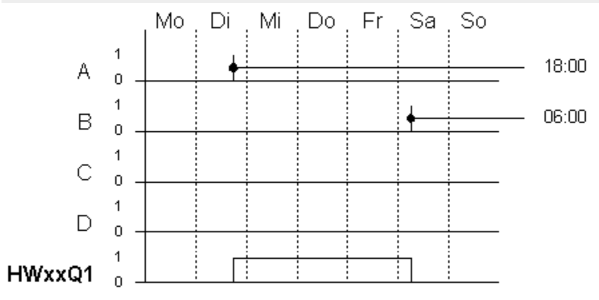



Figure 116: Signal diagram

The HW time switch must be assigned the following parameters:

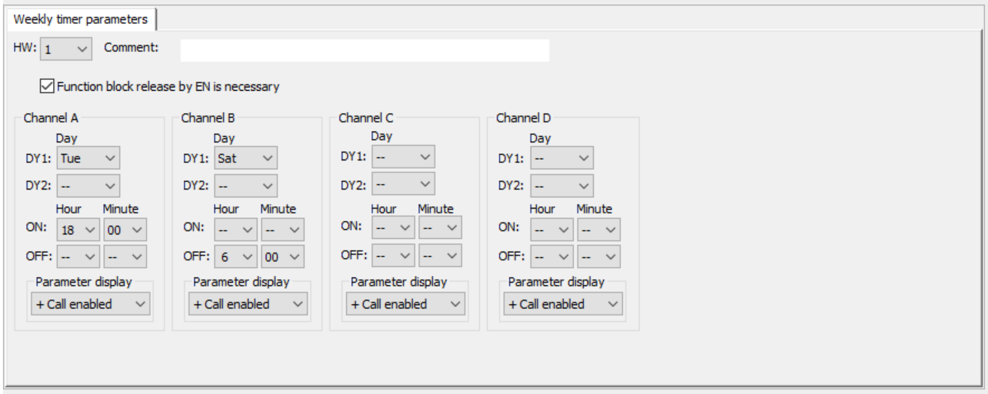



Figure 117: Tab with parameters in the Programming view

Example 4: Time overlap

The time settings of a time switch overlap. The clock switches on at 16:00 on Monday, whereas on Tuesday and Wednesday it switches on at 10:00. The off time for Mondays to Wednesdays is 22:00.

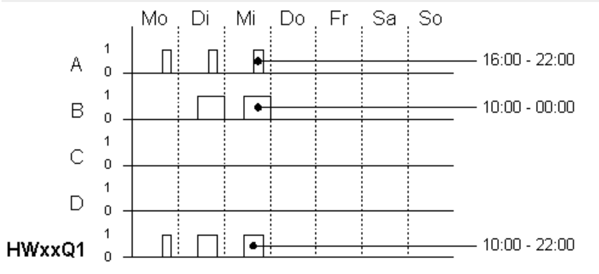



Figure 118: Signal diagram

The first on time at one of the four channels switches output Q1 to 1. The first off time of a channel switches output Q1 to 0. If the on time and off time are the same, the output Q1 is switched off.

The HW time switch must be assigned the following parameters:

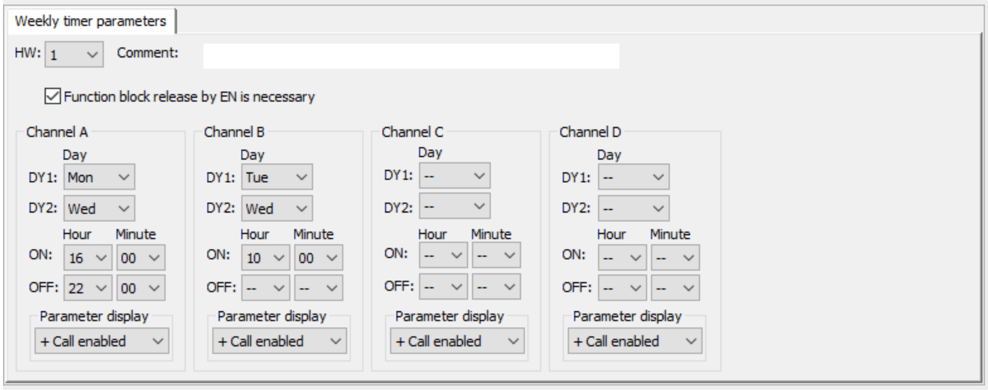



Figure 119: Tab with parameters in the Programming view Settings Time overlap

Example 5: 24 hours

The time switch is to switch for 24 hours. On time at 00:00 on Monday and off time at 00:00 on Tuesday.

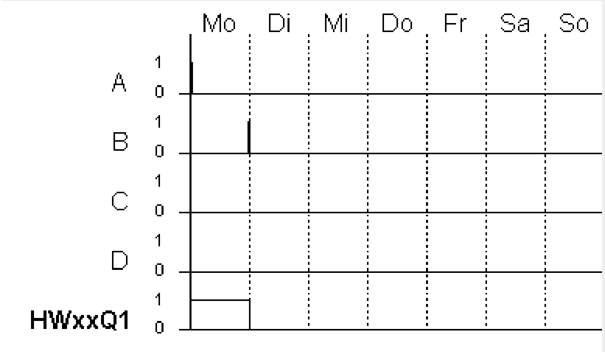



Figure 120: Signal diagram

The HW time switch must be assigned the following parameters:

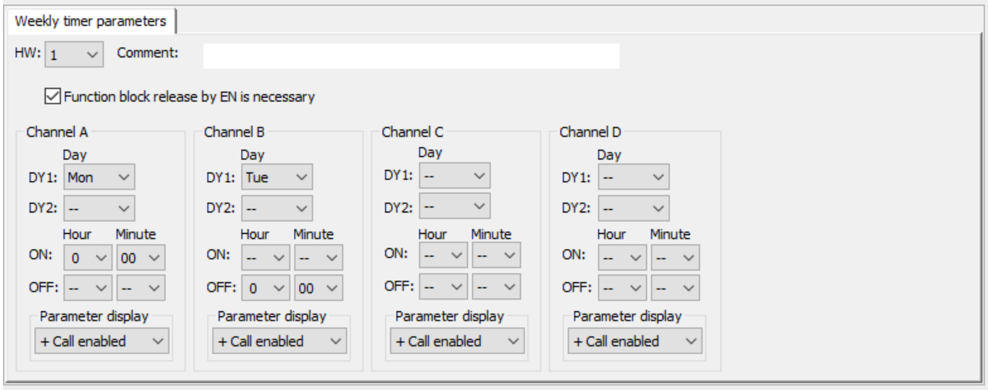



Figure 121: Tab with parameters in the Programming view 24 hours setting

Example 6: Overnight switching

The time switch is set for one day, e.g. Mondays, for an on time of ON=22:00 and an off time of OFF=6:00.

The HW time switch must be assigned the following parameters:

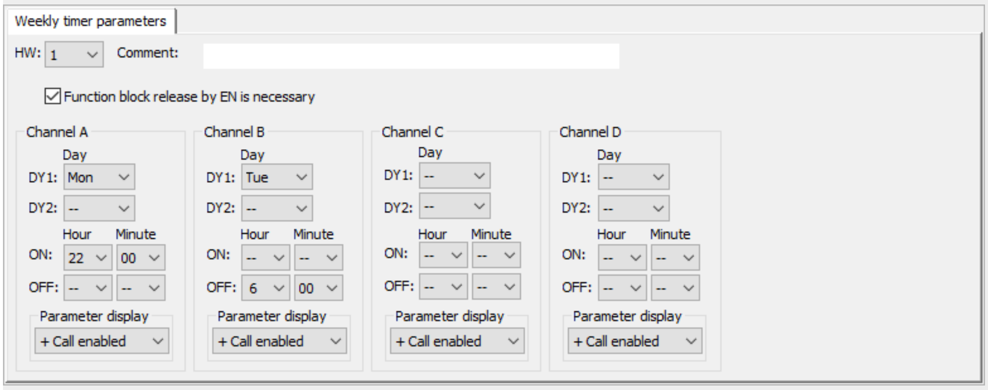



Figure 122: Tab with parameters in the Programming view

## See also

→ Section "HY - Year time switch (Hora Year)", page 228
→ Section "OT - Operating hours counter ", page 238
→ Section "RC - Real-time clock", page 242
→ Section "T - Timing relay", page 246
→ Section "WT - Weekly timer (WeekTable)", page 268
→ Section "YT - Year time switch (Year Table)", page 261
→ Section "AC - Astronomic clock ", page 272

### 6.1.1.2 HY - Year time switch (Hora Year)

easyE4 devices feature a real-time clock with a date and time functionality.
When combined with the HW, HY or WT, YT function blocks, this real-time clock makes it possible to implement the functionality of a weekly timer and year time switch.

→ Section "Time and Date setting", page 587

The AC manufacturer function block, Astronomic clock, can be used to program switching operations based on sunrise and sunset times. In order for this to work properly, the settings for the device clock and the device location's time zone and geographic coordinates must be correctly selected in this tab.

If you have to implement special on and off switching functions on public holidays, vacations, company holidays, school holidays and special events, these can be implemented easily with the year time switch.

The channels are set via the PARAMETER menu or via easySoft 7.

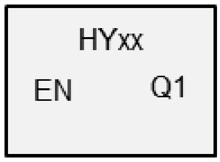
The year time switch can:

- Switch at recurring intervals by switching on and off for individual days, months, or years.
- Switch for continuous periods of time by remaining continuously switched on from the start of any user-defined day until the end of any user-defined day, month, or year.

→ The parameters for the switch-on and switch-off times for recurring intervals are configured in one single channel for each.
The parameters for the switch-on and switch-off times for a continuous period of time are configured in two neighbouring channels. If you enter the ON information on channel A, the OFF information must be entered on channel B; likewise, if the ON information is on channel B, the OFF information must be on channel C.

#### General

easyE4 Base devices provide 32 year time switches HY01…HY32 (Hour Year). Accordingly, 128 switching times are available.
Each time switch is provided with four channels A, B, C and D. You can choose an on and off switching time for every channel. These channels all act jointly on function block output Q1 of the year time switch.



### Operating principle

Each of the 32 year time switches, HY01 through HY32, features four channels that can each be configured with an ON event and an OFF event in the parameter configuration for the block. An ON time and an OFF time that are accurate to the day can be selected for each channel. All channels act jointly on function block output Q1.

### Behavior in the event of a power failure

The time and date are backed up in the event of a power supply failure and continue to run. This means that it will continue to run in the event of a power failure, although the time switch relays will not switch. The contacts are kept open when de-energized. Information on the battery back-up time are provided on → Section "Back-upof real-time clock", page 702

.

→ **Switching behavior with overlapping channel settings:**
If the set ranges overlap, the year time switch activates the contact with the first detected ON signal irrespective of which channel is supplying this ON. In the same way, the year time switch switches the contact off with the first detected OFF, irrespective of whether another channel still supplies the ON signal.
Please note that the time switches can only be configured up to the year 2099.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)] NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: if the on condition is fulfilled. | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| Parameter set | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Channel A - D | A maximum of four channels can be configured (all four channels will act on output Q1). There is an ON time and an OFF time that are accurate to the day for each channel. | |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Parameterization

If you select the function block in the easySoft 7 Programming view by clicking on it, a table with the various parameters will appear under the tab.

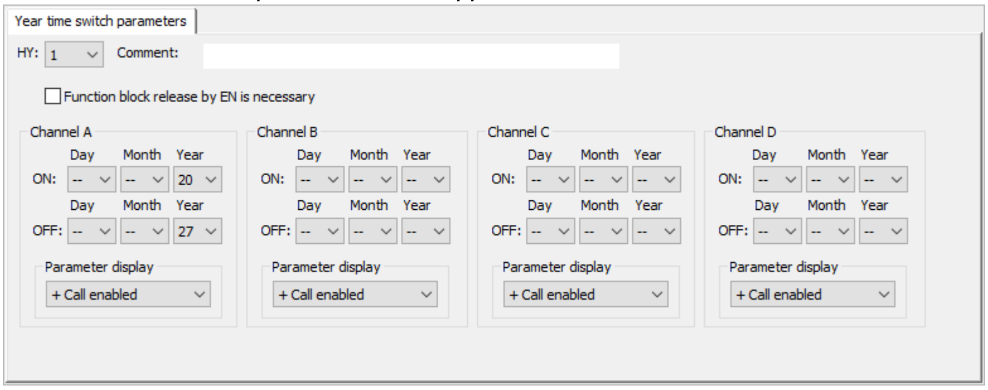



Figure 123: Year time switch parameters tab with and example in which a year range is being selected

### Setting time range parameters

If + Call enabled is selected for the function block under *Function block diagram/Parameters/*, it will be possible to change the switching times in the PARAMETER menu on the device while in RUN / STOP mode.

Time ranges are defined by setting an ON and an OFF time.
The contact therefore always switches from ON to OFF, as shown in the following Parameter examples. → "Example 1: Select year range", page 234.

Please note:
The year time switch only operates correctly if you observe the following rules:

- The On year must be before the Off year,
- ON and OFF times, the same time parameters must be defined.

Examples of correct time parameter configurations:

- ON = --/--/Year, OFF = --/--/Year,
- ON = --/Month/Year, OFF = --/Month/Year,
- ON = Day/Month/Year, OFF = Day/Month/Year

### Setting time range parameters in which the on phase is from the beginning of the year to the end of the year (whole year(s)):

Channel A

ON: -- -- 22, OFF: -- -- 30 means:

The year time switch should switch on at 00:00 on day 01.01.2022 and switch off when the OFF year has elapsed at 00:00 on day 01.01.2031. The parameters are set in one channel. Refer to the → "Example 1: Select year range", page 234 below for this time range.

**Setting time range parameters in which the on phase is from the beginning of the month to the end of the month (whole month(s)):**

First channel ON: -- 04 --, OFF: -- 10 -- means:

This year time switch will switch on on April 01st at 00:00 and after the OFF month elapses will switch off on November 01st at 00:00. The parameters are set in one channel. Compare this to → "Example 2: Select month ranges", page 234 for this time range found below.

**Setting time range parameters in which the on phase is from the beginning of the day to the end of the day for each month in each year (whole day(s)):**

First channel ON: 02 -- --, OFF: 25 -- -- means:

The year time switch should switch on at 00:00 h on day 2 of each month, and switch off when the OFF day has elapsed at 00:00 h on day 26. The parameters are set in one channel. Refer to the → "Example 3: Select day ranges", page 235 below for this time range.

**Setting time range parameters in which the on phase is from the beginning of the day to the end of the day for specified months and years year (day, month, year):**

First channel ON: 02 04 25; OFF: 25 09 25 means:

The year time switch will switch on on 04/02/2015 at 00:00:01 and off on 09/26/2029 at 00:00:00. Outside of the configured time range, the time switch will remain off.

**Setting overlapping time ranges:**

Refer to the → "Example 7: Overlapping ranges", page 237 below for these time ranges.

In these cases, a time cannot be configured for switching, and switching will always occur for the entire day, from 00:00 to 24:00. This is a set configuration that cannot be modified at runtime.

#### Other

#### Retention

The function block does not recognize retentive data.

#### Examples HY - Year time switch in easySoft 7

#### Example 1: Select year range

The year time switch HY01 should switch on at 1 January 2020, 00:00 h, and remain switched on until 1 January 2028, 00:00 h.

The HY year time switch must be assigned the following parameters:

*Programming view/HY01/Year time switch parameters tab*

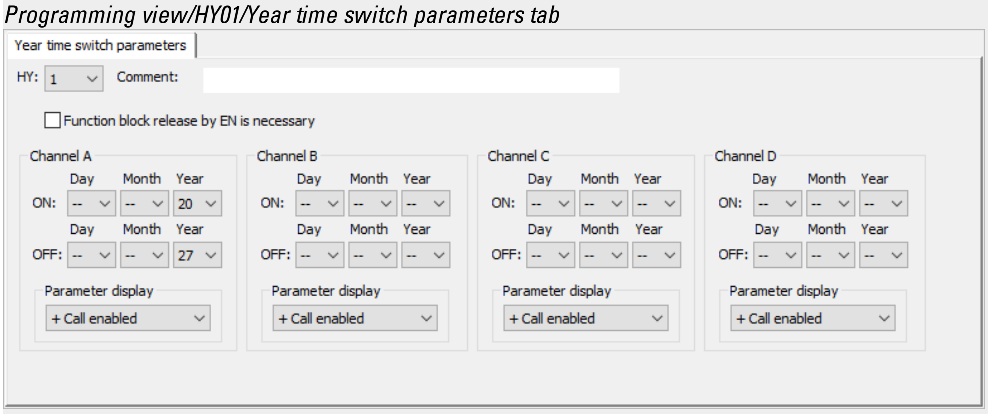



Figure 124: Entry screen in the programming software

#### Example 2: Select month ranges

The year time switch HY01 should switch on at 1 March , 00:00 h, and remain switched on until 1 November, 00:00 h.

The HY year time switch must be assigned the following parameters:

*Programming view/HY01/Year time switch parameters tab*

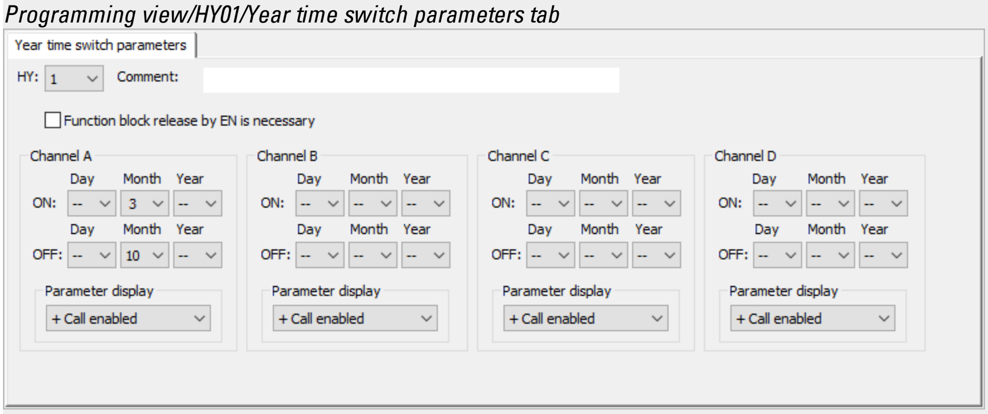



Figure 125: Entry screen in the programming software

### Example 3: Select day ranges

The year time switch HY01 is required to switch on at 00:00 on day 1 of each month and switch off at 00:00 on day 29 of each month.

The HY year time switch must be assigned the following parameters:

*Programming view/HY01/Year time switch parameters tab*

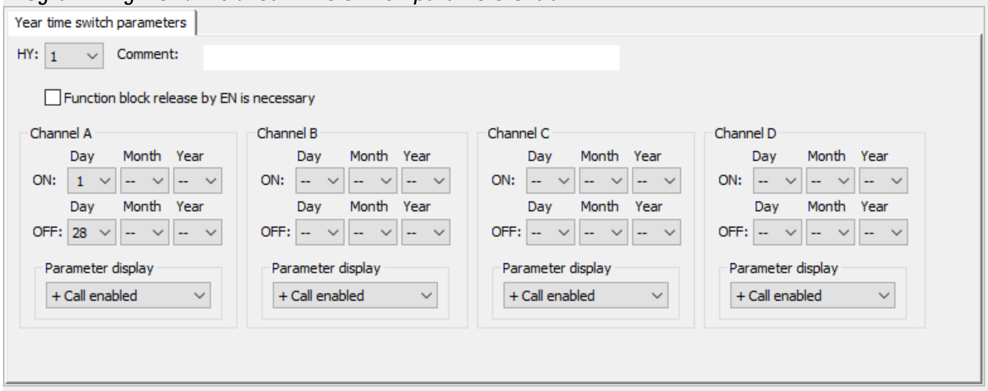



Figure 126: Entry screen in the programming software

### Example 4: Select "public holidays"

The year time switch HY01 is required to switch on at 00:00 on day 25.12 of each year and switch off at 00:00 on day 28.12 of each year.

The HY year time switch must be assigned the following parameters:

*Programming view/HY01/Year time switch parameters tab*

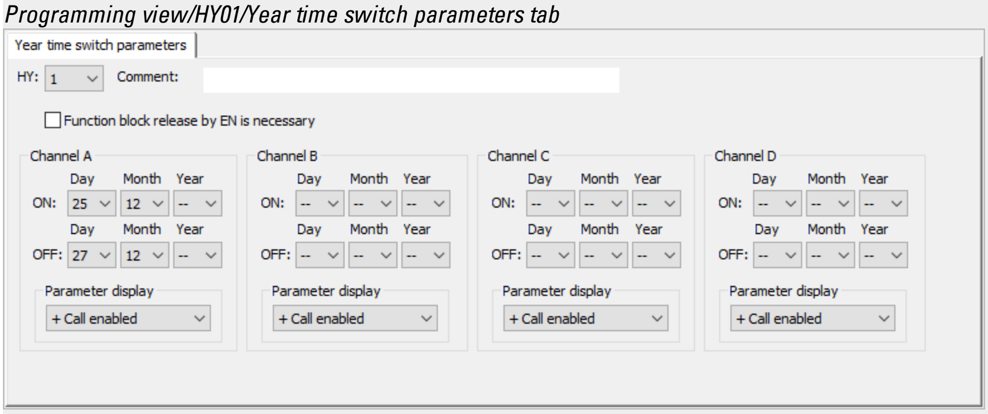



Figure 127: Entry screen in the programming software

**Example 5: Select time range**

The year time switch HY01 is required to switch on at 00:00 on day 01.05 of each year and stay on continuously until 00:00 on 2.11 of each year.

The HY year time switch must be assigned the following parameters:

*Programming view/HY01/Year time switch parameters tab*

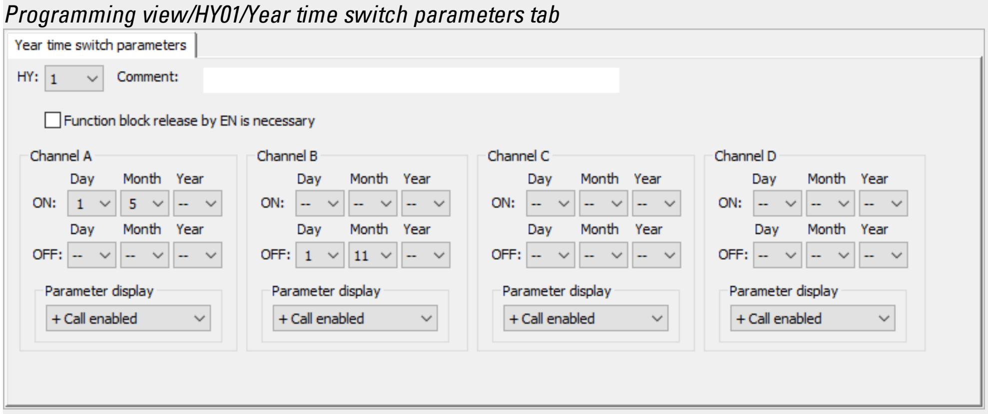



Figure 128: Entry screen in the programming software

**Example 6: Specific days of specific months**

The year time switch HY01 is required to switch on at 0:00 on day 9 of months 6, 7, 8, 9 and 10 of each year and switch off at 00:00 on day 17 of the month.

The HY year time switch must be assigned the following parameters:

*Programming view/HY01/Year time switch parameters tab*

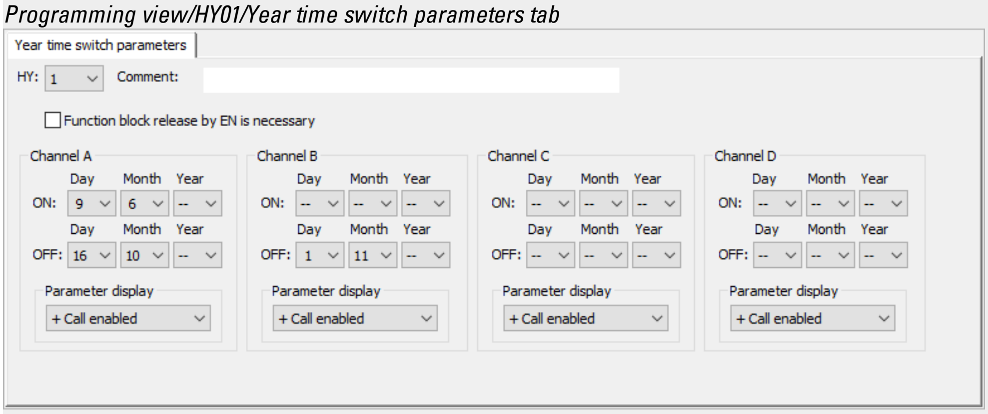



Figure 129: Entry screen in the programming software

**Example 7: Overlapping ranges**

Channel A of the year time switch HY01 switches on at 00:00 on day 3 of months 5, 6, 7, 8, 9, 10 and stays on until 00:00 on day 27 of the same months.

Channel B of the year time switch HY01 switches on at 00:00 on day 2 of months 6, 7, 8, 9, 10, 12 11 and stays on until 00:00 on day 19 of the same month.

The HY year time switch must be assigned the following parameters:

*Programming view/HY01/Year time switch parameters tab*

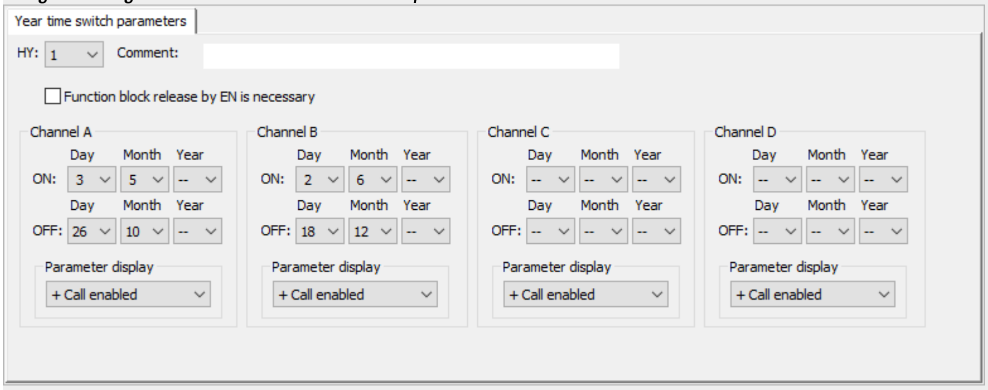



Figure 130: Entry screen in the programming software

Resulting behavior of contact HY01 Q1: The time switch comes on at 00:00 on day 3 and goes off at 00:00 on day 27. The time switch comes on at 00:00 on day 2 of the months June to December, and goes off at 00:00 on day 19 of the same months.

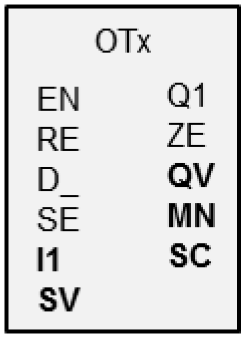
**See also**

→ Section "HW - 7-day time switch (Hour Week)", page 218
→ Section "OT - Operating hours counter ", page 238
→ Section "RC - Real-time clock", page 242
→ Section "T - Timing relay", page 246
→ Section "WT - Weekly timer (WeekTable)", page 268
→ Section "YT - Year time switch (Year Table)", page 261
→ Section "AC - Astronomic clock ", page 272

### 6.1.1.3 OT - Operating hours counter

#### General

easyE4 base devices provide 4 operating hours counter function blocks, OT01 through OT04. These function blocks output minutes and seconds in addition to hours. A comparison with a reference value that can be entered makes it possible, for instance, to signal when maintenance work is due.
The counter states are retained even when the device is switched off.




#### Operating principle

The operating hours counter will run as long as input EN has a state of 1. The operating hours counter's seconds will be output at SC, the minutes at MN, and the hours at QV. The second and minute values will run from 0 to 59, and the hour values will run from 0 to 596,523 h.

The operating hours counter features a comparison function. The corresponding reference value needs to be connected to I1. With every call, the value of the operating hours counter will be compared with the value at I1. The operating hours counter features a direction input, D_.

If the operating hours exceed the reference value at I1 when counting up, function block output Q1 will switch to 1 as long as the number of operating hours is greater than or equal to the reference value.
If, on the other hand, the operating hours fall below the reference value at I1 when counting down, function block output Q1 will switch to 1 until the number of operating hours is greater than the reference value.

The operating hours counter can be preset to any value you want. This value needs to be connected to SV and applied with a wising edge at SE.
The operating hours at QV will not be reset to zero unless reset input RE is activated.

→ An operating mode change between STOP/RUN, supply voltage ON/OFF, deleting the program, editing the program, loading a new program: None of these actions will clear the operating hours counter's actual value.
However, operating hours will not be added if the program is not running.
The actual value can only be cleared by using the reset input.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |
| RE | Reset<br>1: Resets the actual counter value back to zero. | |
| D_ | Count direction<br>1: down counting<br>0: up counting | Integer value range:<br>0…596 523 |
| SE | When there is a rising edge at SE, the value at SV is applied as the operating hours value and appears at QV | |
| **(DWord)** | | |
| I1 | The value at I1 is the reference value. If this reference value is greater than the operating hours value, output Q1 will be set. | |
| SV | When there is a rising edge at SE, the value at SV is applied as the operating hours value | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |

| Operators | Bit inputs |
|---|---|
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: If the operating hours counter reaches or falls below the reference value at I1 when counting down or reaches and exceeds it when counting up | |
| ZE | Zero<br>1: If operating hours counter = 0 | |
| **(DWord)** | | |
| QV | Actual operating hours counter value<br>Displayed in hours | Integer<br>Value range: 0…596 523 |
| MN | Minutes | Value range: 0…59 |
| SC | Seconds | Value range: 0…59 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

**Parameter set**

| Configuration/time range | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

#### Retention

The function block does not recognize retentive data.

**See also**

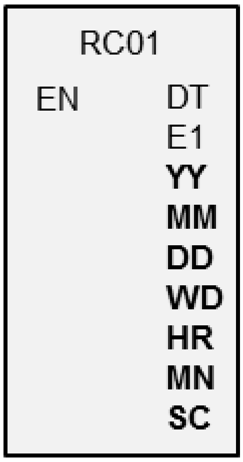
→ Section "HW - 7-day time switch (Hour Week)", page 218
→ Section "HY - Year time switch (Hora Year)", page 228
→ Section "RC - Real-time clock", page 242
→ Section "T - Timing relay", page 246
→ Section "WT - Weekly timer (WeekTable)", page 268
→ Section "YT - Year time switch (Year Table)", page 261
→ Section "AC - Astronomic clock ", page 272

#### 6.1.1.4 RC - Real-time clock

##### General

easyE4 Base devices provide exactly one real-time clock RC01.

This function block can be used to read the date and time value of the device's real-time clock. This value is output in seven individual parameters than can each be processed further individually. This makes it very easy to select recurring events with a downstream comparator function block.




##### Operating principle

If the function block is enabled, the date and time value from the device's real-time clock will be output at the function block outputs: YY (year), MM (month), DD (day), WD (day of the week), HR (hours), MN (minutes), SC (seconds).

Function block output DT signals whether the clock has been switched to daylight saving time.

##### The function block and its parameters

##### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |

##### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)] NET station n | x |

| Operators | Value inputs |
|---|---|
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| DT | 0: The output value is standard time<br>1: The output value is daylight saving time | |
| E1 | Error<br>0: Error-free operation<br>1: The value is invalid, since it is before the device's initial date | |
| **(DWord)** | | |
| YY | Date: year | Range of 00 to 99 |
| MM | Date: month | Range of 00 to 12 |
| DD | Date: day | Range of 00 to 31 |
| WD | Weekday | 0= Su; 1=Mo, 2=Tu, 3=We, 4=Th, 5=Fr, 6=Sa |
| HR | Time: hour | Range of 00 to 23 |
| MN | Time: minute | Range of 00 to 59 |
| SC | Time: second | Range of 00 to 59 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

☑ Function block release by EN is necessary

### Parameter set

| | Description | Note |
|---|---|---|
| | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

### Retention

The function block does not recognize retentive data.

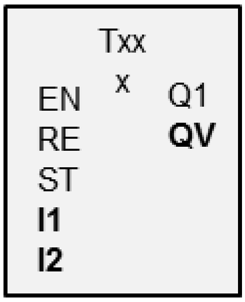
### See also

→ Section "HW - 7-day time switch (Hour Week)", page 218
→ Section "HY - Year time switch (Hora Year)", page 228
→ Section "OT - Operating hours counter ", page 238
→ Section "T - Timing relay", page 246
→ Section "WT - Weekly timer (WeekTable)", page 268
→ Section "YT - Year time switch (Year Table)", page 261
→ Section "AC - Astronomic clock ", page 272

### 6.1.1.5 T - Timing relay

#### General

easyE4 base devices provide 32 timing relays (timer) T01…T32.




You can use a time relay to delay the switching duration and the ON and OFF times of a switch contact. The times can be set from a range of 5 ms to 99 h 59 min.
As reference values, you can use positive values, e.g., from analog inputs or actual values from counter and timing relays.
Minimum time setting: 0.005 s (5 ms).

#### Operating principle

Each one of the 32 timing relays is a multifunction relay with various operating modes. This operating mode is selected during configuration and cannot be changed at runtime.

In addition, the time range can be configured as well. Seconds, minutes, and hours are available.

The operands with the time reference values are connected to inputs I1, I2 and the switching state and the actual value of the running timing relay are signaled at the outputs.

The timing relay is started via the trigger coil T…EN and can be selectively reset via the reset coil T…RE. The third coil T..ST terminates the run down of the actual time.

The EN input is used to start and stop the timing relay.

→ Enabling the function block by disabling the Function block release by Enable is necessary option is not possible in this case.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block.<br>Enable, the timing relay is started (Trigger coil) When a rising edge is detected, the timing relay (trigger) will be started at the same time. EN must be set to 1 without interruption until the time you want has elapsed.<br>The only operating mode in which detecting a rising edge is enough is the Single pulse mode. In this case, the function block will be activated for one cycle and started for this mode. | |
| RE | Reset<br>1: Resets the timing relay to a value of zero (reset coil) | |
| ST | Stop coil<br>1: stops the timing relay.<br>The started time will cease to time out whilst the ST is set to 1.<br>The stopped time will continue to time out if the signal is reset to 0.<br>If ST has a state of 1 when there is a rising edge at trigger coil EN, the assumption of the time reference value will be delayed while ST =1. | |
| **(DWord)** | | |
| I1 | Time setpoint 1 | Integer value range:<br>S: 1…999995 ms, resolution 5 ms<br>M:S: 1… 5999 s, resolution 10 ms<br>H:M: 1… 5999 min, resolution 1 Min. |
| I2 | Time reference value 2 for an operating mode with two reference values, e.g., flashing. This input will be ignored in the case of operating modes with only one single reference value. | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant 1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers 2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |

| Operators | Value inputs |
|---|---|
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

#### Interpretation of variable operands for the time reference values at inputs I1 and I2

#### Variable time values

If you associate the function block inputs T ..I1 and T ..I2 with operands, you can use variable setpoints. The setpoints are transferred as follows, depending on the time range selected:

- S, value in milliseconds. The last digit is rounded up or down to 0 or 5, maximum value = 999995 ms.
- M:S, value in seconds, maximum value = 5999 s.
- H:M, value in minutes, maximum value = 5999 min.

Examples of time range S:

- Operand value 9504 -> time value is 9.500 s.
- Operand value 45507 -> 45.510 s.

Examples of time range M:S:

- Operand value 5999 -> Time value is 99 min, 59 s.

Example of time range H:M:

- Operand value 5999 -> Time value is 99 h, 59 min.

#### Operating mode

This parameter defines the switch function of the timing relay.

| Device parameters | Operating mode easySoft 7 | Note |
|---|---|---|
| X | On-delayed | |
| ?X | On-delayed with random time | |
| ■ | Off-delayed | |
| ?■ | Off-delayed with random time | |
| X■ | On/off delayed | There are two time setpoints to be configured. |
| ?X□ | On/off-delayed with random time | With random time, 2 time setpoints |
| ⎍ | Single pulse | Normalizes input signals of different pulse lengths to a fixed pulse length at the switch contact of the timing relay. |
| ⊔ | Flashing<br>Time values:<br>S1=Pulse time,<br>S2= Pause time; | Time values<br>Two time reference values need to be configured.<br>I1=Pulse time,<br>I2= Pause time;<br>Synchronous flashing: I1 = I2<br>Mark-to-space ratio = 1:1<br>Asynchronous flashing: I1 ≠ I2<br>Mark-to-space ratio ≠ 1:1 |
| □ | Off-delayed with retriggering | Retriggerable reference value |
| ?□ | Off-delayed with retriggering and random time | Retriggerable reference value |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | Switching contact | |
| **(DWord)** | | |
| QV | Elapsed actual time in RUN mode | Integer value range:<br>0 to max. 99990 in time range: seconds; milliseconds; hours<br>depending on configured time range. |

### Parameter set

| Configuration/time range | Description | Note |
|---|---|---|
| S | Seconds:Milliseconds<br>Configurable as a constant:<br>00.005 to 999.995 (s.ms) | Resolution: 5 ms |
| M : S | Minutes:Seconds<br>Configurable as a constant:<br>00:01 to 99:59 (min:s) | Resolution: 1 s |
| H : M | Hours:Minutes | Resolution: 1 min |

| Configuration/time range | Description | Note |
|---|---|---|
| | Configurable as a constant:<br>00:01 to 99:59 (h:min) | |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

Note on the minimum time setting:
If a time value is less than the program cycle time, the elapsed time will not be recognized until the next cycle. This may cause unforeseeable switching states.

### Analog value and timing relay setpoint

If you wish to use variable values as a timing relay setpoint, such as an analog input, the following conversion rules apply, depending on the time base configured.

#### S time base

Equation: Time setpoint = ( Variable value/10) in [ms]

| Variable Value | Time setpoint in [s] | Time setpoint in [mm:ss] | Time setpoint in [hh:mm] |
|---|---|---|---|
| 0 (Minimum) | 00:000 | 00:00 | 00:00 |
| 100 | 00:108 | 01:04 | 01:04 |
| 300 | 00:308 | 05:00 | 05:00 |
| 500 | 00.507 | 08:02 | 10:06 |
| 4095 (Maximum) | 04:099 | 68:15 | 68:15 |

#### M:S time base

Rule: Time setpoint = Variable value/60

Integer = Number of minutes,

Residual = Number of seconds

#### Time base H:M

Rule: Time setpoint = Variable value/60

Integer = Number of hours,

Residual = Number of minutes,

You can only use analog values as setpoints if the value of the analog input is stable. Fluctuating analog values impair a reproducible timing response.

## Signal Diagrams

The fact that the function block features various operating modes means that it can work in more than one way as shown below.

**How the timing relay works with the on-delayed operating mode with and without random times**

Random switching

The contact of the timing relays switches randomly within the SETPOINT value range.

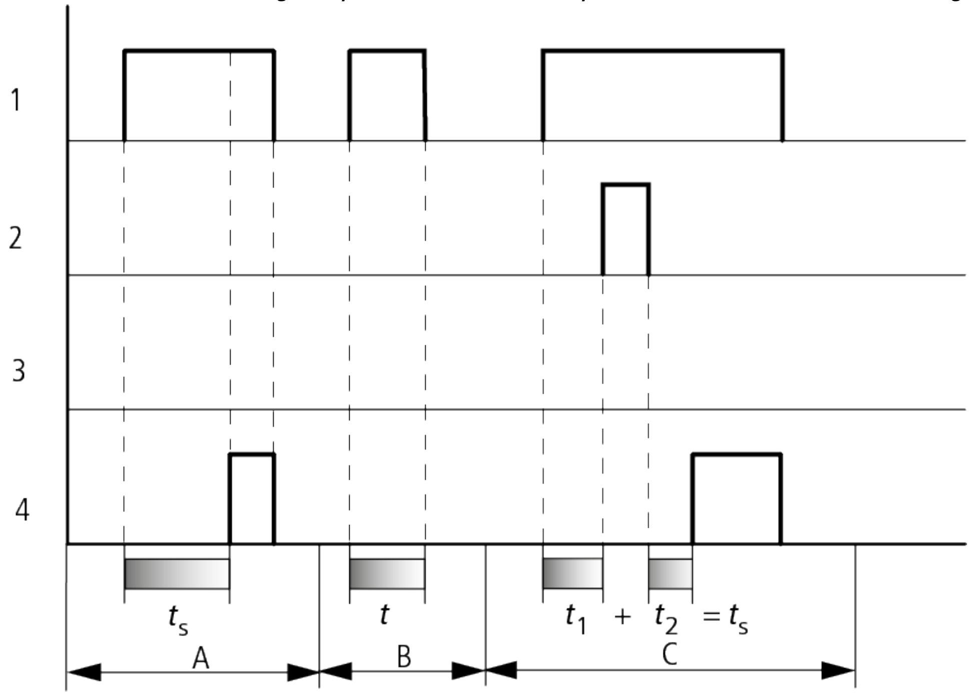



Figure 131: Signal diagram of timing relay, on-delayed (with and without random switching)

1: Trigger coil T..EN

2: Stop coil T..ST

3: Reset coil T..RE

4: Switching contact (N/O contact) T…Q1

ts: Setpoint time

Range A: The time runs down from the SET time value.

Range B: The time does not elapse because the trigger coil drops out prematurely.

Range C: The Stop coil stops the time from elapsing.

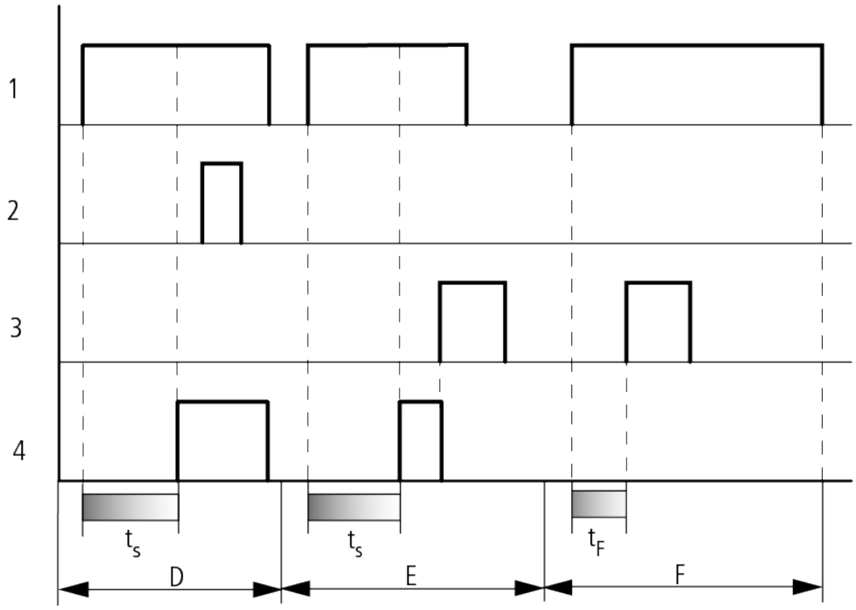



Figure 132: Signal diagram of timing relay, on-delayed (with and without random switching)

Range D: The Stop coil is inoperative after the time has elapsed.

Range E: The Reset coil resets the relay and the contact.

Range F: After the reset coil is activated, the switching contact is switched off and the internal time counter is reset. The function relay waits for a new trigger pulse.

**How the timing relay works with the off-delayed operating mode with and without random times**

Random switching, with and without retriggering

The contact of the timing relays switches randomly within the SETPOINT value range.

Retriggering

When the time is running and the trigger coil is reactivated or deactivated, the ACTUAL value is reset to zero. The SET time of the timing relay is timed out once more.

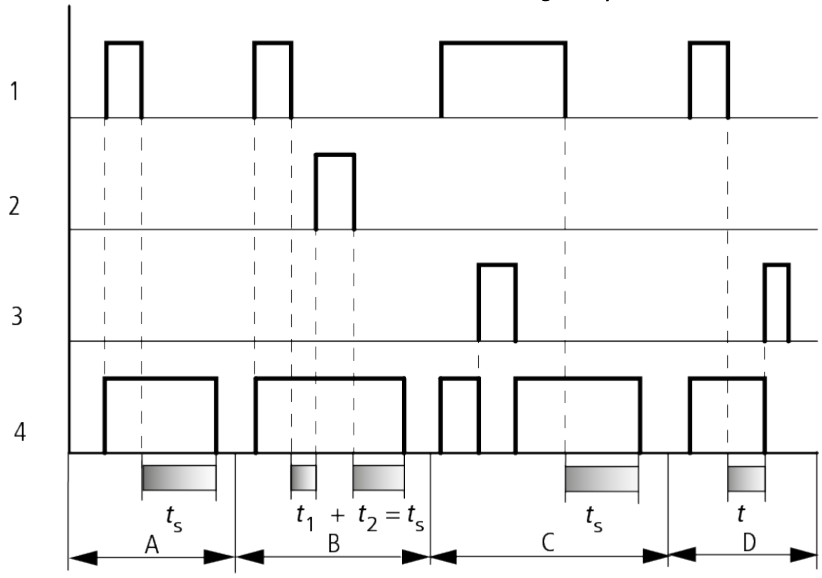



Figure 133: Signal diagram of timing relay, off-delayed

(with/without random switching, with/without retriggering)

1: Trigger coil T..EN

2: Stop coil T..ST

3: Reset coil T..RE

4: Switching contact (N/O contact) T…Q1

ts: Reference time

Range A: The time elapses after the trigger coil is deactivated.

Range B: The Stop coil stops the time from elapsing.

Range C: The Reset coil resets the relay and the contact.

After the reset coil drops out, the relay continues to work normally.

Range D: The Reset coil resets the relay and the contact when the function block is timing out.

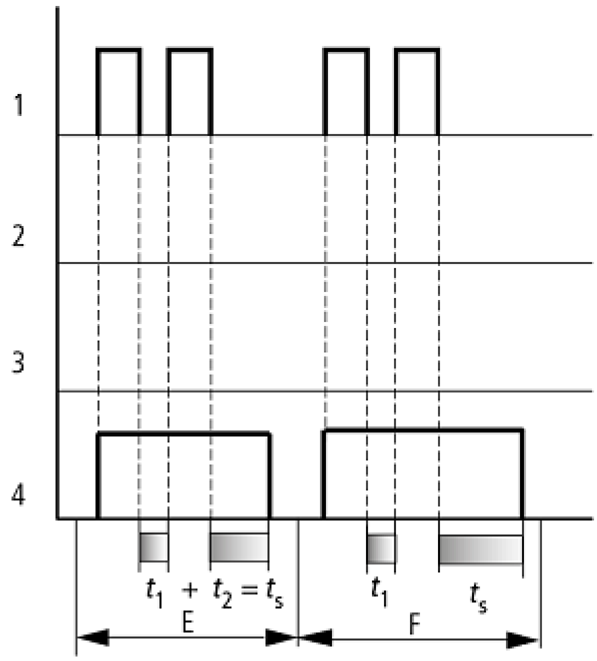



Figure 134: Signal diagram of timing relay, off-delayed

(with/without random switching, with/without retriggering)

Range E: The trigger coil drops out twice.

The SET time ts consists of t1 plus t2 (switch function not retriggerable).

Range F: The trigger coil drops out twice. The ACTUAL time t1 is cleared and the SET time ts elapses completely (retriggerable switch function).

**How the timing relay works with the on/off-delayed operating mode with and without random times**

Time value I1: on-delay time

Time value I2: Off-delay time

Random switching

The contact of the timing relay switches randomly within the SETPOINT value range.

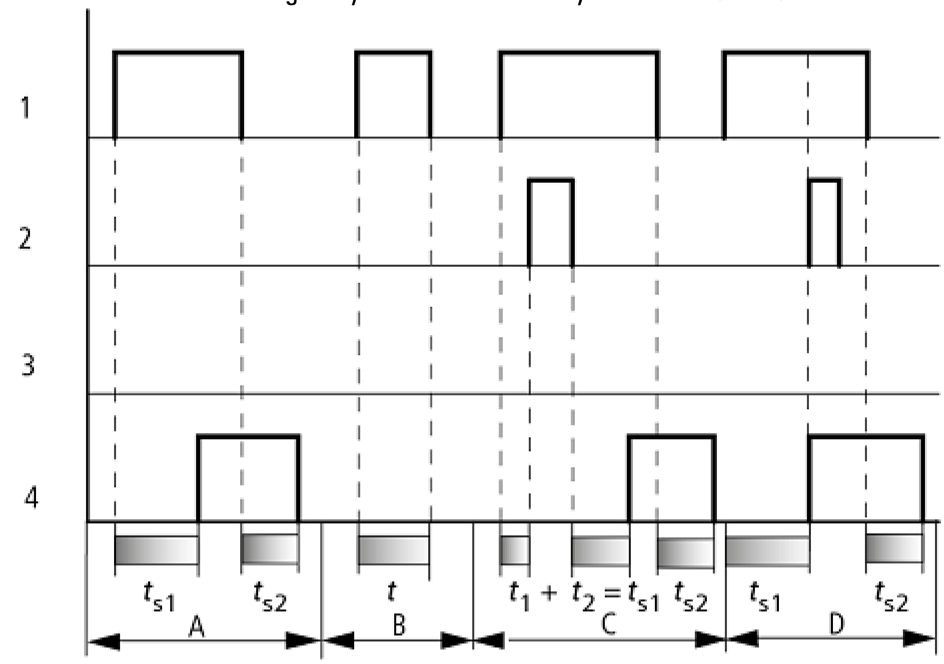



Figure 135: Operational diagrams timing relay, on and off delayed 1

1: Trigger coil T..EN

2: Stop coil T..ST

3: Reset coil T..RE

4: Switching contact (N/O contact) T…Q1

ts1: Pick-up time

ts2: Drop-out time

Range A: The relay processes the two times without any interruption.

Range B: The trigger coil drops out before the on-delay is reached.

Range C: The stop coil stops the timeout of the on-delay.

Range D: The stop coil has no effect in this range.

#### How the timing relay works with the single pulse operating mode with and without random times

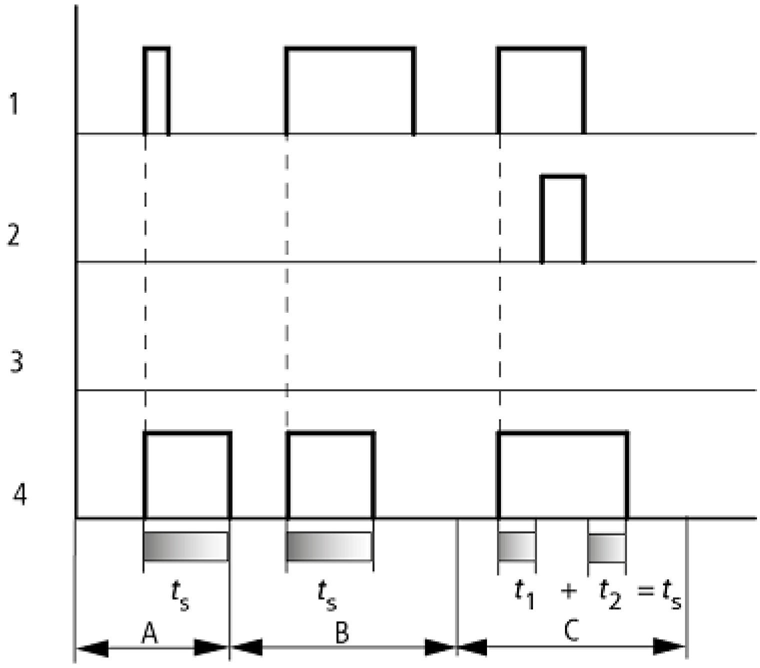



Figure 136: Signal diagram timing relay, single pulse 1

1: Trigger coil T..EN

2: Stop coil T..ST

3: Reset coil T..RE

4: Switching contact (N/O contact) T…Q1

Range A: The trigger signal is short and is lengthened.

Range B: The trigger signal is longer than the SET time.

Range C: The stop coil interrupts the timing out of the set time.

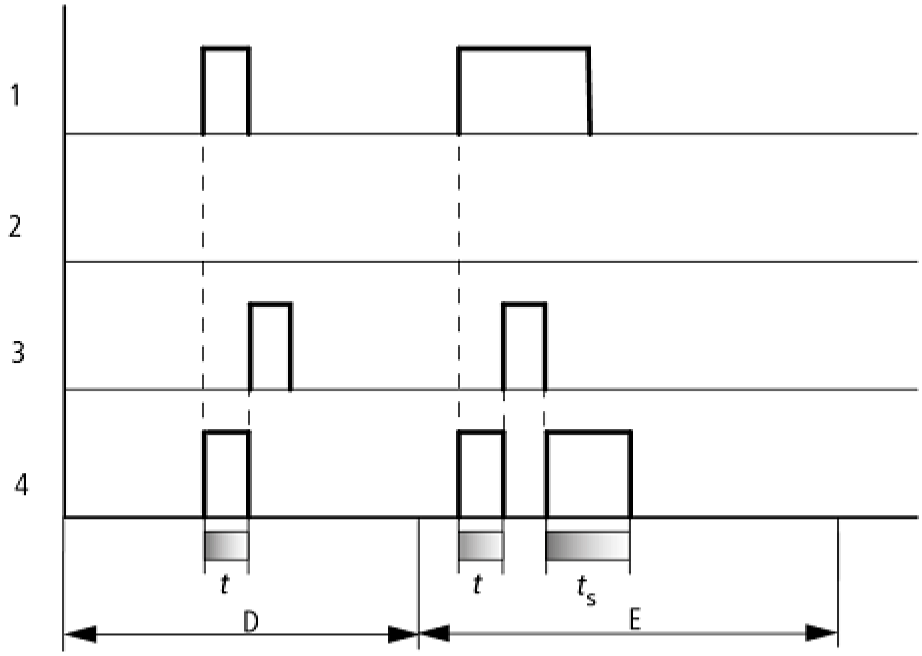



Figure 137: Signal diagram timing relay, single pulse 2

- Range D: The reset coil resets the timing relay.
- Range E: The reset coil resets the timing relay. The trigger coil is still energized after the reset coil is disconnected, whilst the delay time runs down..

### How the timing relay works with the flashing operating mode, synchronous and asynchronous

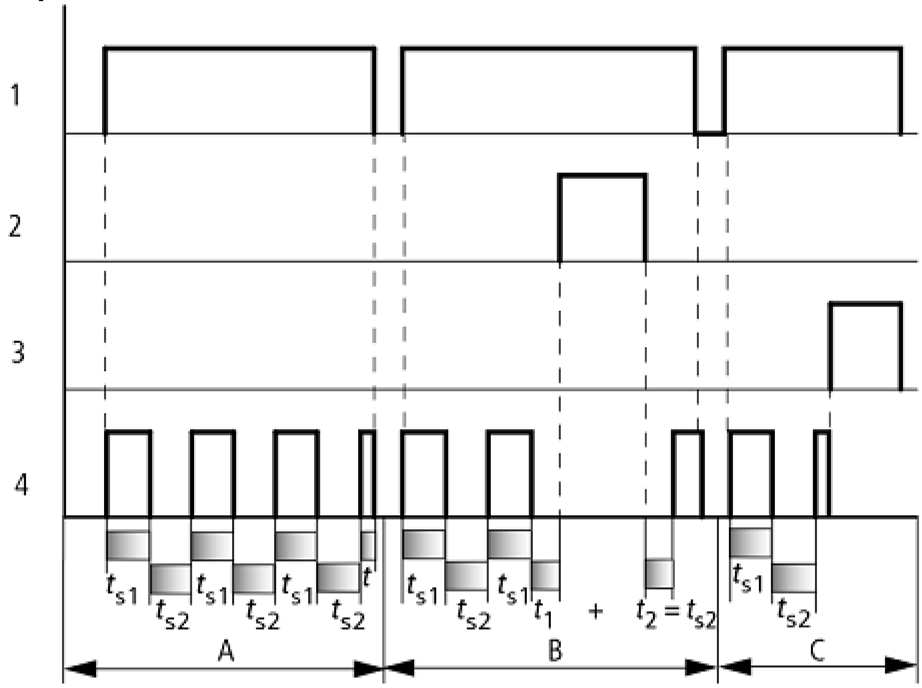



Figure 138: Signal diagram timing relay, single pulse 1

1: Trigger coil T..EN

2: Stop coil T..ST

3: Reset coil T..RE

4: Switching contact (N/O contact) T…Q1

Range A: The relay flashes for as long as the trigger coil is activated.

Range B: The stop coil interrupts the timing out of the set time.

Range C: The reset coil resets the relay.

#### Other

#### Retention

Selected timing relays can be run with retentive actual values. If a timing relay is retentive, the actual value is retained when the operating mode is changed from RUN to STOP and when the power supply is switched off.
When the control relay is restarted in RUN mode, the timing relay continues with the retentively stored actual value.

In the Project view, go to the System settings tab and select the timing relays, out of T1 through T32, that should be run with retentive values. A retentive actual value requires 4 bytes of memory.

| Operand | Description |
|---|---|
| Constant | 0.. to 99:59 (time range: "M : S"/"H : M") or 0 - 99.99 (time range: "S") |
| C | Output of a counter relay (e.g. C3QV).<br>If the actual counter value is greater than the maximum permissible setpoint of the configured time range, the setpoint will be limited to this maximum value.<br>Example: Say you have configured the time range M: S and the actual counter value is 31333. The device will limit the setpoint to 5999 min. |
| IA | Note the relationship described below between the permissible analog value and the timing relay setpoint. |
| T | Output of a timing relay (e.g. T4QV). |

#### Example of a timing relay when using the EDP programming language

```
I 10-----------------------------[ T 02EN

M 42-----------------------------[ T 02RE

M 43-----------------------------[ T 02ST
```

Figure 139: Wiring the function block coils

The trigger coil of the function block is connected here directly to the device inputs.

. A marker activates the reset coil, while another marker activates the
stop coil.

```
T 02Q1---------------------------[ Q 01
```

Figure 140: Wiring of the function block contact

The signal from the function block will go directly to the device output.

**See also**

→ Section "HW - 7-day time switch (Hour Week)", page 218
→ Section "HY - Year time switch (Hora Year)", page 228
→ Section "OT - Operating hours counter ", page 238
→ Section "RC - Real-time clock", page 242
→ Section "WT - Weekly timer (WeekTable)", page 268
→ Section "YT - Year time switch (Year Table)", page 261

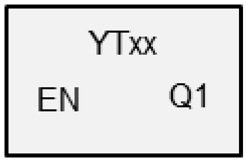
### 6.1.1.6 YT - Year time switch (Year Table)

easyE4 devices feature a real-time clock with a date and time functionality.
When combined with the HW, HY or WT, YT function blocks, this real-time clock makes it possible to implement the functionality of a weekly timer and year time switch.

→ Section "Time and Date setting", page 587

The AC manufacturer function block, Astronomic clock, can be used to program switching operations based on sunrise and sunset times. In order for this to work properly, the settings for the device clock and the device location's time zone and geographic coordinates must be correctly selected in this tab.

#### General




This function block is an enhanced version of the existing HY - year time switch function block.
easyE4 base devices provide 32 year time switches (new), YT01 through YT32 (Year Table).
Year time switches can be used to easily configure unique or recurring switching events.
The following operating modes are available within this context:

- Fixed date
- Fixed date for every year
- Weekday rule
- Easter rule

Variable holidays other than Easter cannot be selected.

#### Operating principle

Each of the 32 year time switches, YT01 through YT32, features eight channels that can each be configured with an ON event and an OFF event in the parameters for the function block. All channels act jointly on function block output Q1.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2] | x |
| nNB, nND, nND- NET markers [2] NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2] | x |
| SN - Output bit via NET (send)[2] | x |
| N - Net marker bit[2] | x |
| nN - NET marker bit[2] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: if the on condition is fulfilled. | Can be used to directly connect an output that implements the configured switching times |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| Parameter set | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Channel A - D | A maximum of four channels can be configured (all four channels will act on output Q1). There is an ON time and an OFF time that are accurate to the day for each channel. | |
| Parameter display<br>+ Call enabled | Constants can be edited on the | |

| Parameter set | Description | Note |
|---|---|---|
| | device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Parameterization

If you select the function block in the easySoft 7 Programming view by clicking on it, a table with the various parameters will appear under the tab.

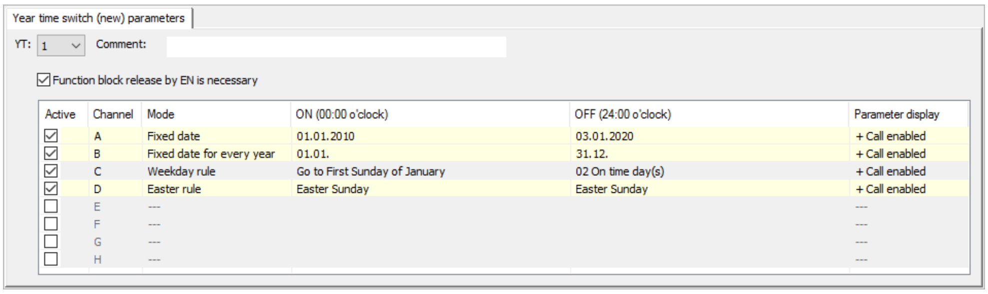



Figure 141: Year time switch (new) parameters tab for YT function block with example showing all four modes

If + Call enabled is selected for the function block under *Function block diagram/Parameters/*, it will be possible to change the switching times in the PARAMETER menu on the device while in RUN / STOP mode.

One of the following operating modes can be selected for each of the channels, A through H:

- Fixed date
  Will switch one; the ON and OFF times and specified with a number of years
- Fixed date for every year
  ON and OFF times with specified day and month but no year
- Weekday rule
  A cyclical switching operation that is carried out on a defined day of the week during a defined month. For example: the "first Sunday of January"
- Easter rule
  You can select an ON time and an OFF time that repeat annually and are relative to Easter. Easter does not have a fixed date, and is instead based on the lunar calendar. The selectable reference points for the ON and OFF times are Good Friday, Easter Sunday, Easter Monday, and day(s) before/after Easter Sunday.
  Reference points other than Easter cannot be selected.

In these cases, a time cannot be configured for switching, and switching will always occur for the entire day, from 00:00 to 24:00. This is a set configuration that cannot be modified at runtime.

This example uses all four available modes.

### Other

#### Retention

The function block does not recognize retentive data.

#### Examples YT - Year time switch in easySoft 7

#### Example 1: Select year range

Year time switch YT01 should switch on on January 1st, 2020, at 00:00, and remain switched on until January 1st, 2028, 00:00.

The YT year time switch must be assigned the following parameters:

*Programming view/YT01/Year time switch (new) parameters tab*

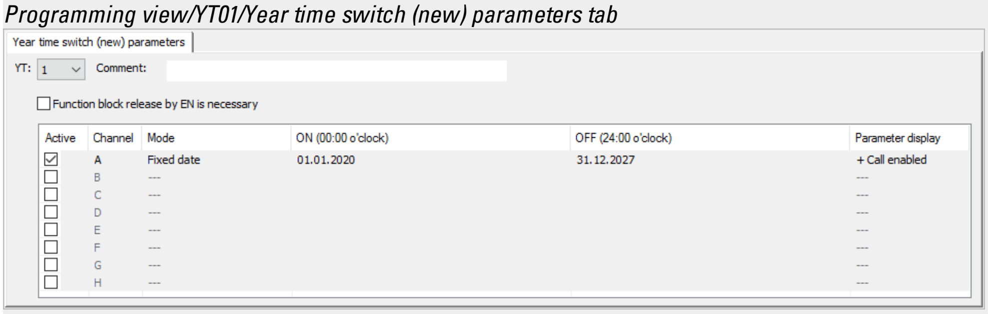



Figure 142: Entry screen in the programming software

#### Example 2: Select month ranges

Year time switch YT01 should switch on on March 1st, at 00:00, and remain switched on until November 1st, at 00:00.

The YT year time switch must be assigned the following parameters:

*Programming view/YT01/Year time switch (new) parameters tab*

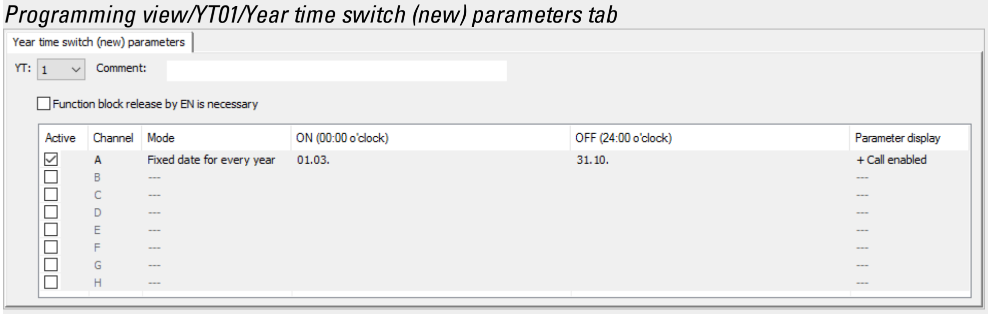



Figure 143: Entry screen in the programming software

#### Example 3: Select “public holidays"

Year time switch YT01 should switch on on 12/05 of every year at 00:00 and remain switched on until 12/28 of every year at 00:00.

The YT year time switch must be assigned the following parameters:

*Programming view/YT01/Year time switch (new) parameters tab*

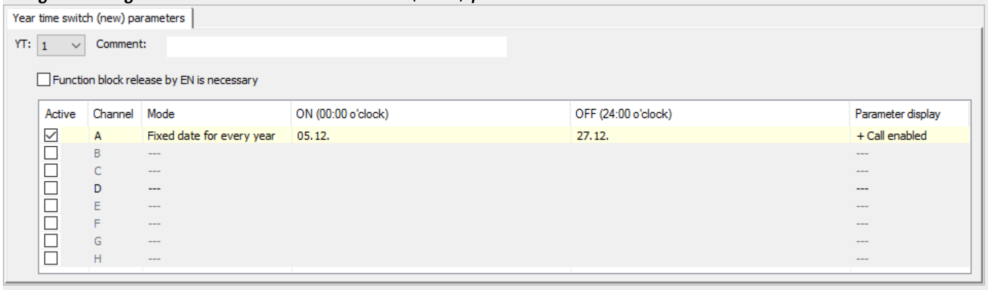



Figure 144: Entry screen in the programming software

### Example 4: Select time range

Year time switch YT01 should switch on on 05/01 of every year at 00:00 and remain switched on until 11/02 of every year at 00:00.

The YT year time switch must be assigned the following parameters:

*Programming view/YT01/Year time switch (new) parameters tab*

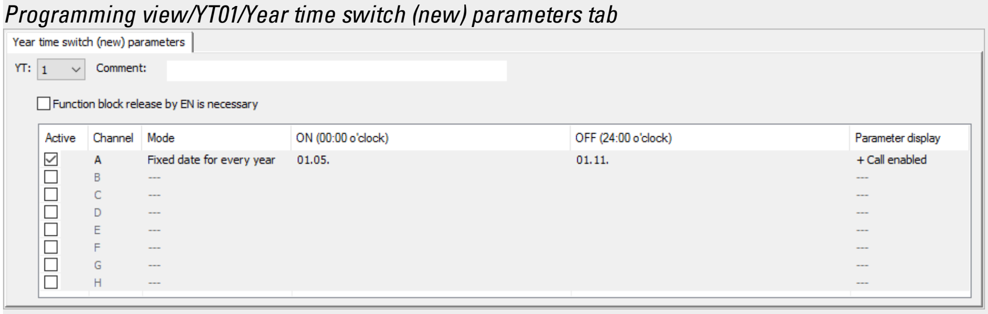



Figure 145: Entry screen in the programming software

### Example 5: Specific days of specific months

Year time switch YT01 should switch on on the 9th of months 6, 7, 8, 9, and 10 every year at 00:00 and switch off on the 17th at 00:00.

The YT year time switch must be assigned the following parameters:

*Programming view/YT01/Year time switch (new) parameters tab*

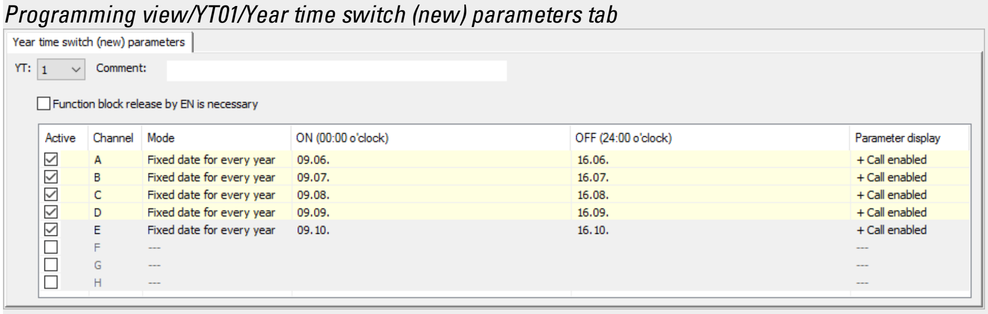



Figure 146: Entry screen in the programming software

**See also**

→ Section "HW - 7-day time switch (Hour Week)", page 218
→ Section "HY - Year time switch (Hora Year)", page 228
→ Section "OT - Operating hours counter ", page 238
→ Section "RC - Real-time clock", page 242
→ Section "T - Timing relay", page 246
→ Section "WT - Weekly timer (WeekTable)", page 268
→ Section "AC - Astronomic clock ", page 272

### 6.1.1.7 WT - Weekly timer (WeekTable)

easyE4 devices feature a real-time clock with a date and time functionality.
When combined with the HW, HY or WT, YT function blocks, this real-time clock makes it possible to implement the functionality of a weekly timer and year time switch.

→ Section "Time and Date setting", page 587

The AC manufacturer function block, Astronomic clock, can be used to program switching operations based on sunrise and sunset times. In order for this to work properly, the settings for the device clock and the device location's time zone and geographic coordinates must be correctly selected in this tab.

#### General


This function block is an enhanced version of the existing HW - 7-day time switch function block.
easyE4 Base devices provide 32 weekly timers, WT01 through WT32. WT weekly timers can be used to easily configure recurring switching events. The function block was specifically designed for implementing switching events that occur at set weekly cycles.
It can also take into account different procedures for business days and weekends.

#### Operating principle

Each of the 32 weekly timers, WT01 through WT032, can be configured with eight switching events that will be executed at the same time and on any specified day of the week. The corresponding settings are accurate to the minute and cannot be modified at runtime, i.e., they must be viewed as a set configuration.

#### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: if the on condition is fulfilled. | Can be used to directly connect an output that implements the configured switching times |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers [2)]<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

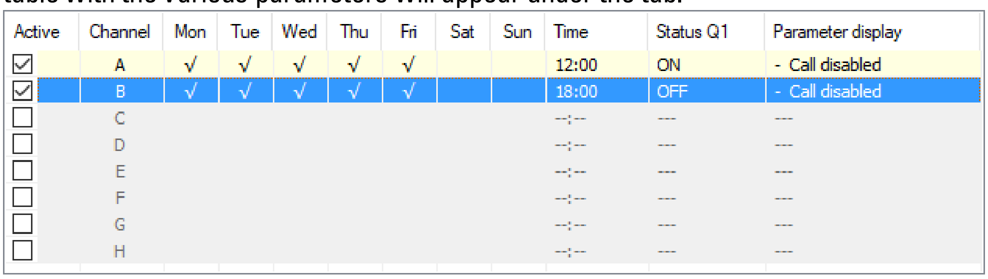
| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)[2] SN - Output bit via NET (send) | x |
| N - Network marker bit[2] | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| Parameter set | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Channel A - D | A maximum of eight channels can be configured (all eight channels will act on output Q1). There is an ON time and an OFF time that are accurate to the day for each channel. | |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Parameterization

If you select the function block in the easySoft 7 Programming view by clicking on it, a table with the various parameters will appear under the tab.



| Active | Channel | Mon | Tue | Wed | Thu | Fri | Sat | Sun | Time | Status Q1 | Parameter display |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ☑ | A | √ | √ | √ | √ | √ | | | 12:00 | ON | - Call disabled |
| ☑ | B | √ | √ | √ | √ | √ | | | 18:00 | OFF | - Call disabled |
| ☐ | C | | | | | | | | --:-- | --- | --- |
| ☐ | D | | | | | | | | --:-- | --- | --- |
| ☐ | E | | | | | | | | --:-- | --- | --- |
| ☐ | F | | | | | | | | --:-- | --- | --- |
| ☐ | G | | | | | | | | --:-- | --- | --- |
| ☐ | H | | | | | | | | --:-- | --- | --- |

Figure 147: Weekly timer (new) parameters tab with an example

If + Call enabled is selected for the function block under *Function block diagram/Parameters/*, it will be possible to change the switching times in the PARAMETER menu on the device while in RUN / STOP mode.

Channels A through H are available for an ON or OFF switching operation. The time to be entered must be between 00:00 and 23:59.

In the example, the ON time on business days is 12:00, and the OFF time is 18:00. A channel is required for each switching action. Channel A switches on every day of the week at 17:30, while channel B switches off at 20:00.

**See also**

→ Section "HW - 7-day time switch (Hour Week)", page 218
→ Section "HY - Year time switch (Hora Year)", page 228
→ Section "OT - Operating hours counter ", page 238
→ Section "RC - Real-time clock", page 242
→ Section "T - Timing relay", page 246
→ Section "YT - Year time switch (Year Table)", page 261
→ Section "AC - Astronomic clock ", page 272

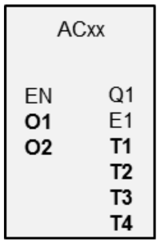
#### 6.1.1.8 AC - Astronomic clock

Only available on easySoft Version 7.10 or higher.

If this function block is not being shown in the leftmost pane in easySoft 7, make sure that you are using firmware version 1.10 or higher for the project.

**General**

easyE4 base devices provide 32 astronomic clock function blocks, AC01 through AC32. Output Q1 in these function blocks is switched on during the time between sunrise and sunset.




**Operating principle**

The astronomic clock function block calculates both sunrise and sunset times based on the geographical position of the device's location and the current device time (both need to be entered in order for the function block to work correctly). The device location can be set in the *Project view/Clock tab*, while the device time can be checked and changed directly on the device or in the *Communication view/Clock section*.

The astronomic clock  function block is meant for use in latitudes of -65 to +65. Please note that the sunrise and sunset time calculations will be too inaccurate outside of this range ( at a latitude of 60, the times will be inaccurate by up to 5 minutes; at a latitude of 65.7, the times will be inaccurate by about 12 minutes).

A time offset for the sunrise time and the sunset time can be set at function block inputs O1 and O2 respectively. This means that the time when Q1 will be switched can be moved up or delayed in order to, for example, account for a heating system's lag and post-run times.

If a daylight saving time has been set up in the *Project view/Clock tab*, this setting will also be taken into account when switching function block output Q1.

The function block inputs' and function block outputs' resolution is in minutes.

Time zone data changed at runtime will immediately affect the function block's behavior accordingly.

→ The device location and device time must be correctly specified.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | |
| O1 | Offset for calculating the sunrise time, in minutes | Integer value range: -720…+720 |
| O2 | Offset for calculating the sunset time, in minutes | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2) NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| (Bit) | | |
| Q1 | 1: During the time between sunrise and sunset | |
| E1 | Error<br>1: If the device location's latitude exceeds the value range – please refer to *Project view/Clock tab*<br><br>or if O1, O2 exceeds the value range. | Integer value range starting from the prime meridian:<br><br>Longitude<br>-180…+180 (W…O)<br><br>Latitude<br>-89.899…+89.899 (S…N)<br>(-89°54'…+89°54') |
| (DWord) | | |
| T1 | Hours in the switch-on time calculated on the basis of the calculated sunrise and the value at O1 | Integer value range:<br>0…23 |
| T2 | Minutes in the switch-on time calculated on the basis of the calculated sunrise and the value at O1 | Integer value range:<br>0…59 |
| T1 | Hours in the switch-off time calculated on the basis of the calculated sunset and the value at O2 | Integer value range:<br>0…23 |
| T1 | Minutes in the switch-off time calculated on the basis of the calculated sunset and the value at O2 | Integer value range:<br>0…59 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

**Parameter set**

| | Description | Note |
|---|---|---|
| **Parameter set** | | |
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

#### Retention

The function block does not recognize retentive data.

#### Examples showing how the AC function block will work in different regions of the world

The gray area in the diagrams shows the time of day when Q1 will have a value of 1. Accordingly, the examples show how longitude and latitude affect function block output Q1.

There is no offset in the following examples, i.e., O1=0, O2=0;

#### Bonn in Germany

The following is the geodata for the Bonn location in Germany:

- Latitude: 50.7344111
- Longitude: 7.0854634

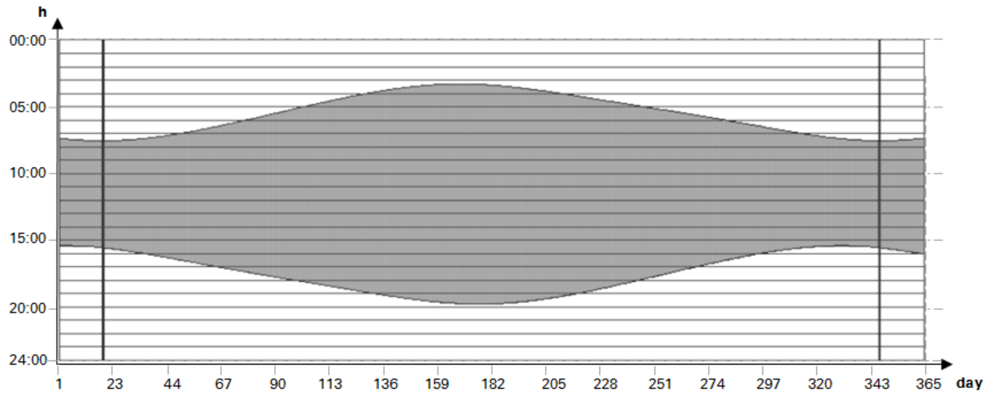



Figure 148: Sunrise and sunset in Bonn

#### Drevja in Norway

The following is the geodata for the Drevja location in Norway:

- Latidude: 65.9780775
- Longitude: 13.2348074

The sun does not set during the summer months (day 165 to 180).

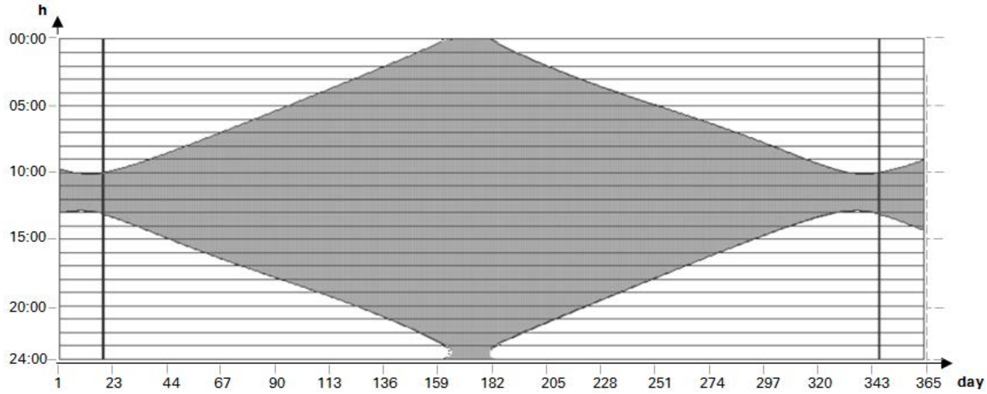



Figure 149: Sunrise and sunset in Drevja

### Douala in Cameroon

The following is the geodata for the Douala location in Cameroon:

- Latidude: 4.0047314
- Longitude: 9.7329299

The sunrise and sunset times remain the same throughout the whole year with little variation.

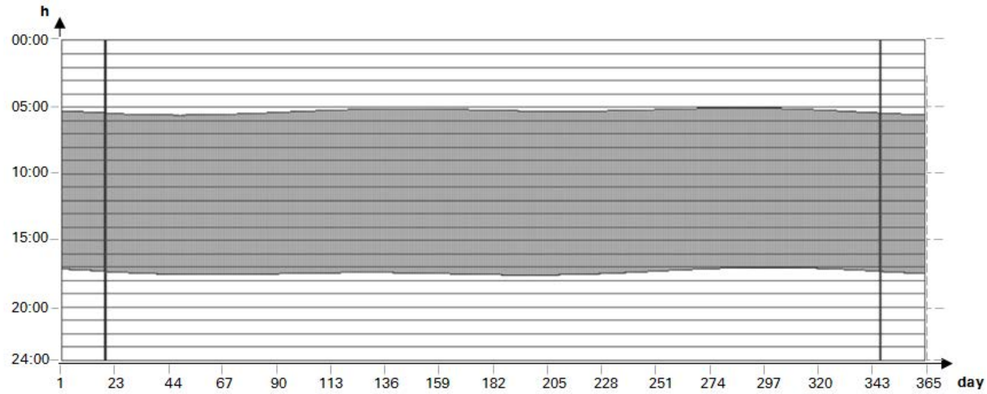



Figure 150: Offset; O1=-2; O2=2; Q1=1 will switch on 2 hours before sunrise and off 2 hours after sunset

**Examples showing how the AC function block will behave with various offset values in O1 and O2**

The gray area in the diagrams shows the time of day when Q1 will have a value of 1. These examples are meant to show how the O1 and O2 offsets affect function block output Q1.

The same geodata applies to all the examples below:

- Latitude: 50.7344111
- Longitude: 7.0854634

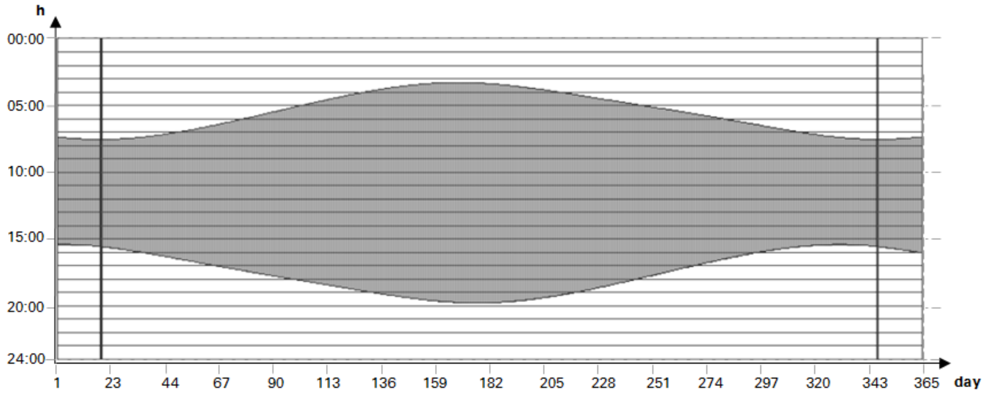



Figure 151: No offset; O1=0; O2=0; Q1=1 between sunrise and sunset

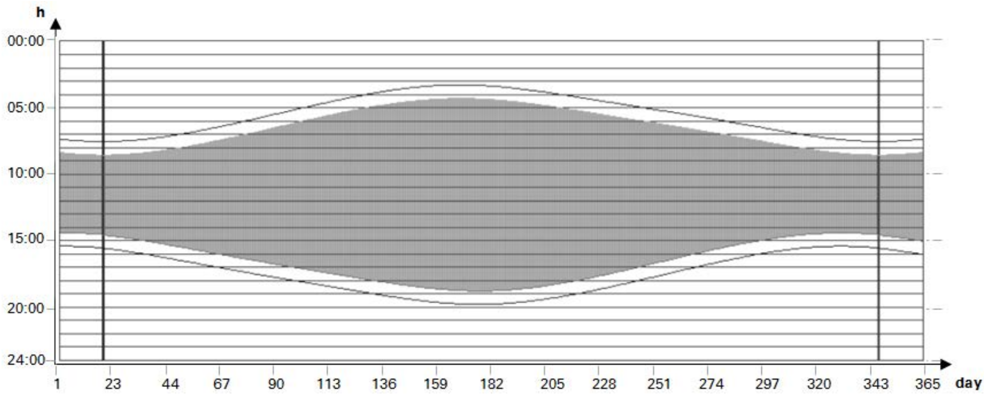



Figure 152: Offset; O1=1; O2= -1; Q1 will switch on 1 hour after sunrise and off 1 hour before sunset

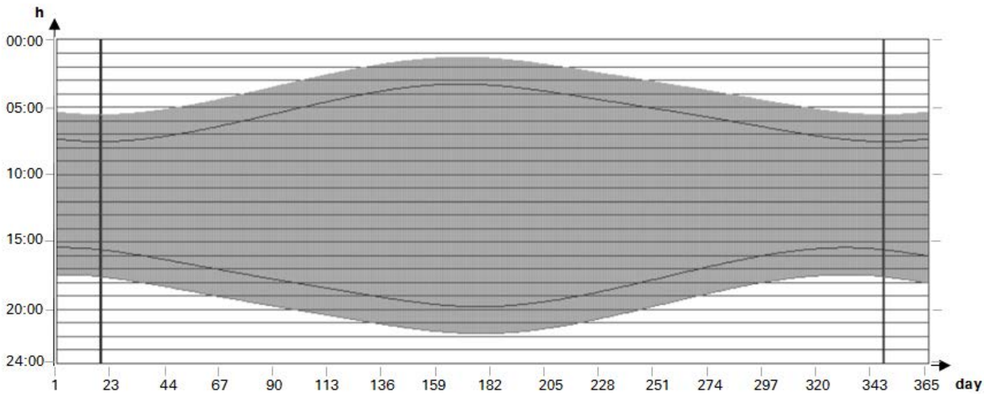



Figure 153: Offset; O1=-2; O2=2; Q1=1 will switch on 2 hours before sunrise and off 2 hours after sunset

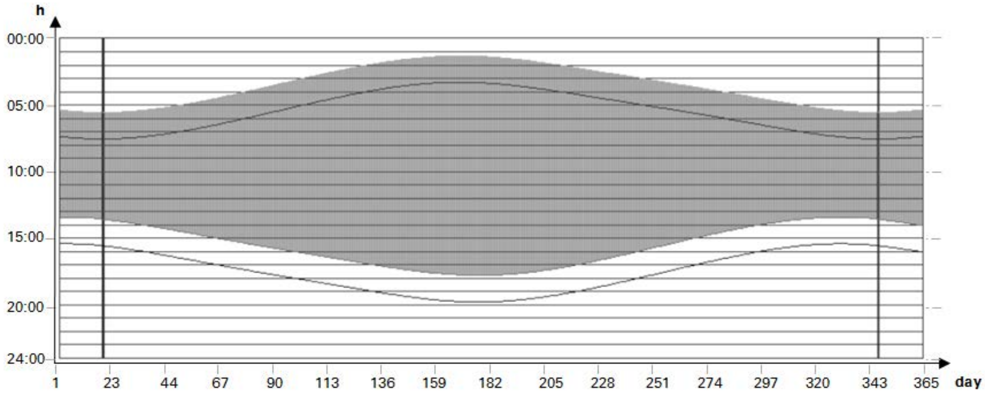



Figure 154: Offset; O1=-2; O2=-2; Q1=1 will switch on 2 hours before sunrise and off 2 hours before sunset

### Overlap between switch-on and switch-off times

The following geodata applies to the examples below:

- Latidude: 60
- Longitude: 0
- Offset O1 = -4
- Offset O2 = 4

During the summer months, the switch-on time will overlap with the switch-off time. This will cause function block output Q1=1 to always remain on during these months.

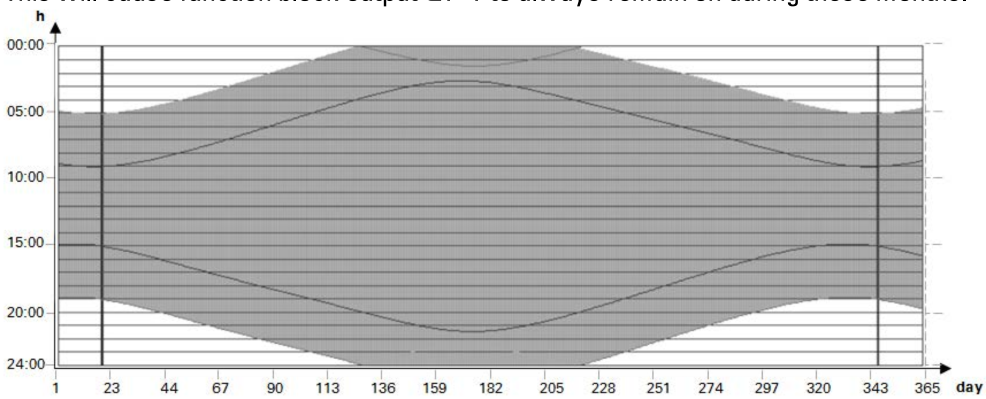



Figure 155: Q1 does not switch off during the summer months

### Switch-off time before switch-on time

The following geodata applies to the examples below:

- Latidude: 60
- Longitude: 0
- Offset O1 = 5
- Offset O2 = -7

In the winter months, the switch-off time is before the switch-on time. This will cause function block output Q1=0 to always remain off during these months.

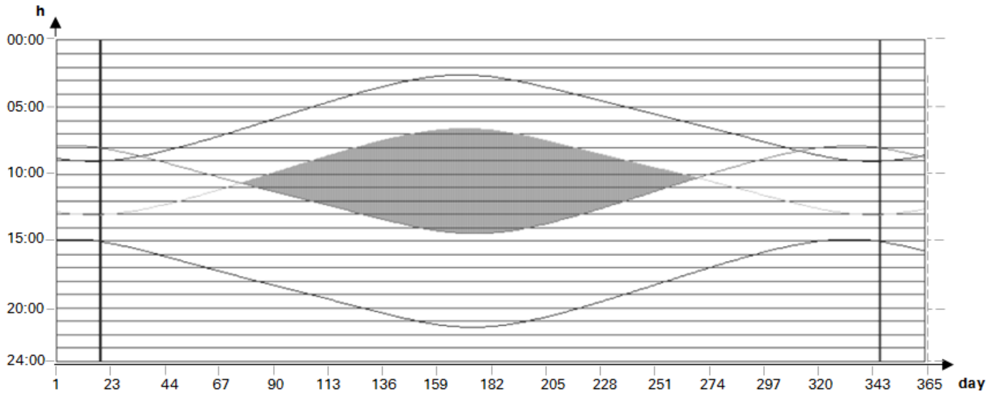



Figure 156: Q1 does not switch on during the winter months

### See also

→ Section "HW - 7-day time switch (Hour Week)", page 218
→ Section "HY - Year time switch (Hora Year)", page 228
→ Section "OT - Operating hours counter ", page 238
→ Section "RC - Real-time clock", page 242
→ Section "T - Timing relay", page 246
→ Section "WT - Weekly timer (WeekTable)", page 268

## 6.1.2 Counter Function Blocks

### 6.1.2.1 C - Counter relay

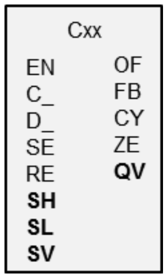
This counter relay function block counts pulses that are received at counter input C_. The count direction can be set.

Counting is carried out cyclically. The time of a pulse must therefore be greater than twice the cycle time.
For shorter pulses, use the CH - High-speed counter function block → page 293.

You can set a lower and higher limit setpoint as comparison values for the counter relay function block, as well as a start value.

#### General

easyE4 base devices provide 32 counter relays, C01 through C32. Each counter relay can count up and down and functions as a double word counter.




#### Operating principle

The contacts will switch according to the actual value. The appropriate function block outputs switch according to the actual value determined. You can set a starting value using input SV.

The counter relays C01…C32 are cycle time dependent.

The following applies to the EDP programming language:
The time for a single count pulse must be longer than twice the cycle time. For shorter pulses, use the CH high-speed counter function block.

| |
|---|
| *NOTICE*<br>Avoid unforeseeable switching states.<br>Switch C, CF, CH, CI function blocks only at one single point in the program.<br>Otherwise, previous counts will be overwritten. |

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| C_ | Counter input, counts with every rising edge | |
| D_ | Count direction<br>0: up counting<br>1: down counting | |
| SE | The starting value at SV is applied whenever there is a rising edge at this input | |
| RE | Reset<br>1: QV=0 | Reset the counter to zero |
| **(DWord)** | | |
| SH | Upper Threshold Value | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| SL | Lower threshold value | |
| SV | Start value (Pre Set) | When there is a rising edge at SE, this value will be applied as the counter value.<br>Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |

| Operators | Bit inputs |
|---|---|
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| OF | Overflow<br>1: if QV ≥ SH | OF=1, if the actual value QV is greater than or equal to the upper threshold value; |
| FB | Fall below<br>1: if QV ≤ SL | FB=1, if the actual value QV is less than or equal to the lower threshold value; |
| CY | Carry<br>1: if QV > value range | If the value range is exceeded, the switch contact switches to status 1 for one cycle per rising edge detected. The function block retains the value of the last valid operation before the contact CY is set. |
| ZE | Zero<br>1: if QV = 0 | |
| **(DWord)** | | |
| QV | Current counter value in RUN mode | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| **Parameter set** | | |
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

## Other

### Signal diagrams

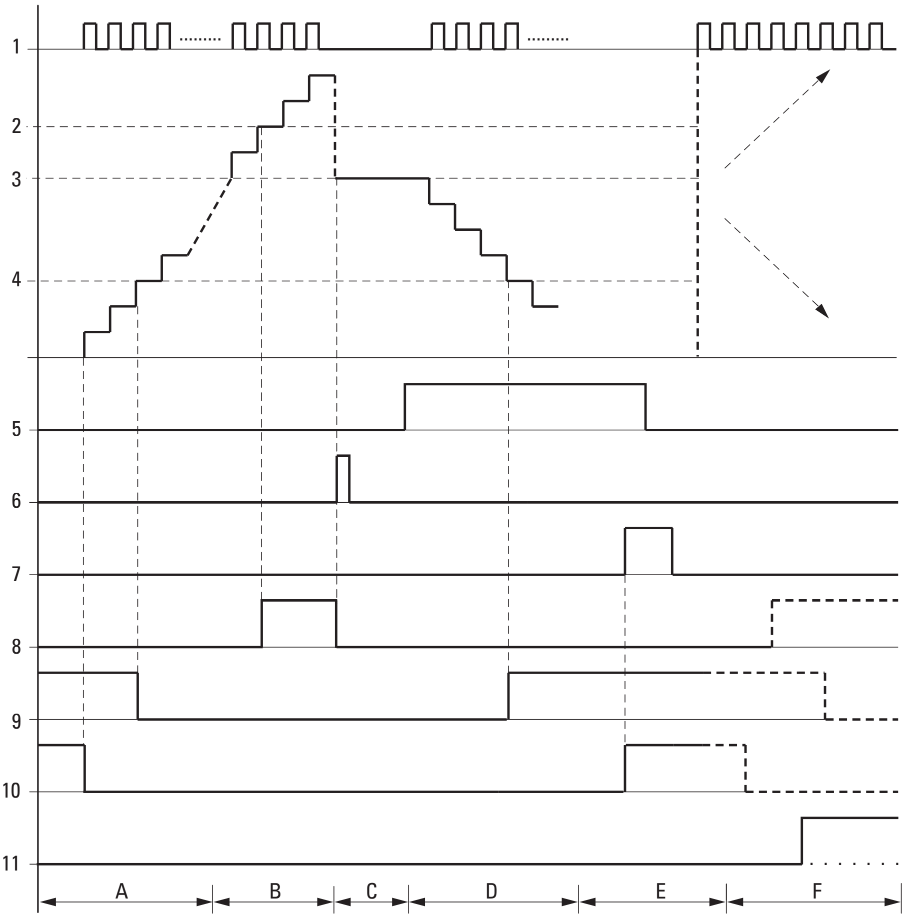



Figure 157: Signal diagram of counter relay

Legend for Figure

1: Counter input C..C_

2: Upper threshold value SH

3: Start value SV

4: Lower threshold value SL.

5: Counting direction, coil C..D

6: Transfer start value, coil C..SE.

7: Reset coil C..RE

8: Contact (N/O) C..OF: Upper limit threshold reached or exceeded.

9: Contact (N/O) C..FB: Lower threshold value reached or undershot.

10: C..ZE = 1, if actual value is zero

11: C..CY = 1, if the value is out of range.

- Range A:

– The counter range has the value zero.

– The contacts C..ZE (actual value equal to zero) and C..FB (lower threshold value undershot) are active.

– The counter receives pulses and increases the actual value.

– C..ZE drops out as well as C..FB after the lower threshold value is reached.

• Range B:

– The counter relay counts upwards and reaches the upper threshold value.

The “upper setpoint value reached” contact C..OF becomes active.

• Range C:

The coil C..SE is briefly actuated and the actual value is set to the start value.

The contacts go to the respective position.

• Range D:

– The counting direction coil C..D_ is actuated. If counting pulses are present, downward counting is initiated.

If the lower threshold value is undershot, the contact C..FB becomes active.

• Range E:

– The reset coil C..RE is activated. The actual value will be set to zero.

– The contact C..ZE is active.

• Range F:

– The actual value goes outside the value range of the counter relay.

– The contacts OF, FB, ZE become active according to the direction of the values (positive or negative).

### Retention

Counter relays can be operated with retentive actual values. You can select the number of retentive counter relays in *Project view/System settings tab*, please refer to → Page 1 System Settings tab. The retentive ACTUAL value will require 4 bytes of memory space. If a counter relay is retentive, the actual value will be retained when the operating mode switches from RUN to STOP and when the power supply is switched off. If the device is started in RUN mode, the counter relay will continue to work with the non-volatile actual value.

### See also

→ Section "CF - Frequency counter", page 287
→ Section "CH - High-speed counter", page 293
→ Section "CI - Incremental Counter", page 299
→ Section "Timing and counter relay example", page 557

### 6.1.2.2 CF - Frequency counter

CF function blocks can be used to carry out frequency measurements that are independent of the cycle time.
There are four frequency counters available. The digital inputs I1 to I4 are permanently assigned to the frequency counter function blocks provided:

- I1: Counter input for counter CF01.
- I2: Counter input for counter CF02.
- I3: Counter input for counter CF03.
- I4: Counter input for counter CF04.

You can set a lower and higher limit setpoint comparison values for the frequency counter function block.

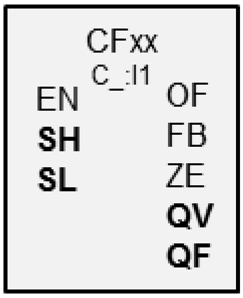
→ Square wave count pulses are required to ensure a 1:1 mark to space ratio.

The function block operates in the integer range from 4 to 5000 (1 kHz = 1000).

The value range cannot be exceeded as the maximum measured value is less than the value range.

#### General

easyE4 base devices provide 4 frequency counters CF01…CF04. These high-speed frequency counters are internally connected with the digital inputs I01…I04 and operate independently of the cycle time. The contacts will switch according to the actual value.




#### Operating principle

For the entire configured measuring interval, the pulses at the input are counted independently of the cycle time and are then used to determine the frequency. The number of pulses counted within the measuring interval is then provided as a value at function block output QV. Finally, as a result, output QF delivers a value equal to ten times the frequency so that the measurement can be accurate to the decimal place despite the fact that the value range is of type integer.

This means that the frequency is the value at QF multiplied by 0.1.

F= QF*0.1

The CF01…CF04 frequency counters are not dependent on the cycle time.

The minimum counter frequency is 4 Hz.

The maximum counter frequency is 5 kHz.

Only square wave signals are permissible.

The mark-to-space ratio is 1:1.

The counter wiring must observe the following digital input assignment:

- I01 counter input for frequency counter CF01
- I02 counter input for counter CF02
- I03 counter input for counter CF03
- I04 counter input for counter CF04

*NOTICE*

Avoid unforeseeable switching states.

Switch C, CF, CH, CI function blocks only at one single point in the program.

Otherwise, previous counts will be overwritten.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | |
| SH | Upper Threshold Value | Integer value range: -2,147,483,648 to +2,147,483,647 |
| SL | Lower threshold value | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)] NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

#### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| OF | Overflow<br>1: if QV ≥ SH | |
| FB | Fall Below<br>1: if QV ≤ SL | |
| ZE | Zero<br>1: if QV = 0 | |
| **(DWord)** | | |
| QV | QV outputs the number of pulses detected per measuring interval | The function block operates in the integer value range from 0…50 000. |
| QF | QF outputs the measured frequency*10. | The function block operates in the integer value range from 4…50 000.<br>The following formula applies: 10 000 = 1 kHz.<br>The measurable frequency range is 0.4…5000 Hz. |

#### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

#### Example with CF01 with 50 Hz at the input

There is a square wave signal with a frequency of 50 Hz at device input I01. Outputs QV and QF of function block CF01 will have the following values depending on the chosen measuring interval:

| Measuring interval | QV | QF | f an I01 |
|---|---|---|---|
| 0.1s | 5 | 500 | 50 Hz |
| 0.5s | 25 | 500 | 50 Hz |
| 1.0s | 50 | 500 | 50 Hz |
| 2.0s | 100 | 500 | 50 Hz |
| 5.0s | 250 | 500 | 50 Hz |
| 10.0s | 500 | 500 | 50 Hz |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Measuring interval | <table><tr><th>Measuring interval</th><th>Maximum value at QV</th></tr><tr><td>0.1s</td><td>500</td></tr><tr><td>0.5s</td><td>2 500</td></tr><tr><td>1.0s</td><td>5 000</td></tr><tr><td>2.0s</td><td>10 000</td></tr><tr><td>5.0s</td><td>25 000</td></tr><tr><td>10.0s</td><td>50 000</td></tr></table> | The longer the measuring interval, the smaller the frequency being measured can be. |
| Simulation possible | | |

### Other

### Signal diagram

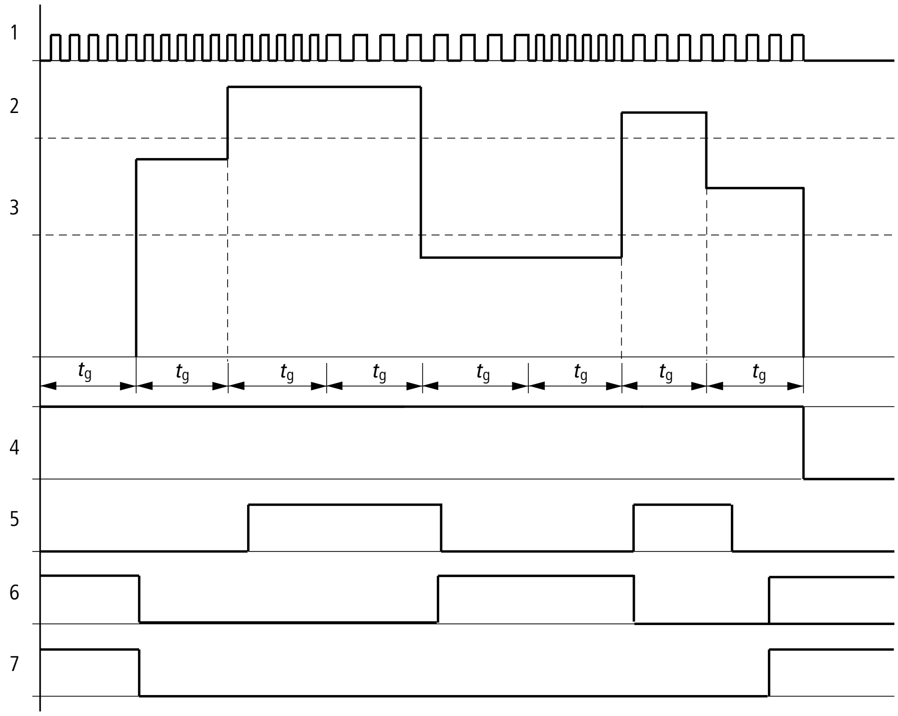



Figure 158: Signal diagram of frequency counter

1: One of the device inputs I01 to I04

2: Upper threshold value SH

3: Lower threshold value SL

4: Enable CF..EN

5: Contact (N/O) CF..OF: Upper threshold value exceeded

6: Contact (N/O) CF..FB: Lower threshold value undershot.

7: Actual value equal to zero CF..ZE

8. tg: gate time ( = measuring interval) for the frequency measurement

The first measurements are made after the CF..EN enable signal has been activated. The value is output after the gate time has timed out. The contacts are set in accordance with the measured frequency. If the CF..EN enable signal is removed, the output value is set to zero.

### Retention

The frequency counter does not have any retentive actual values since the frequency is continuously remeasured.

### See also

→ Section "CF - Frequency counter", page 287
→ Section "CH - High-speed counter", page 293
→ Section "CI - Incremental Counter", page 299
→ Section "Timing and counter relay example", page 557

### 6.1.2.3 CH - High-speed counter

CH function blocks can be used to quickly count rising edges forward and backward. You can set a lower and higher limit as comparison values for the high-speed counter function block, as well as a start value.

There are four high-speed counters available.

→ Square wave count pulses are required to ensure a 1:1 mark-to-space ratio.

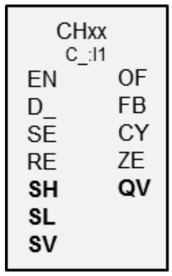
The maximum count frequency is 5000 Hz.

→ Note that the digital inputs I1 to I4 are permanently assigned to the frequency counter function blocks provided:

- I1: Counter input for counter CH01.
- I2: Counter input for counter CH02.
- I3: Counter input for counter CH03.
- I4: Counter input for counter CH04.

#### General

easyE4 Base devices provide 4 high-speed counters CH01…CH 04.
The high-speed up and down counters are internally hard-wired with the digital inputs I01...I04 and operate independently of the cycle time.




#### Operating principle

The contacts will switch according to the actual value. The appropriate function block outputs switch according to the actual value determined. The counter relays enable a preset start value to be defined at the SV input.

Only square wave signals are permissible.

The mark-to-space ratio is 1:1.

The counter wiring must observe the following digital input assignment:

- I01 Counter input for the CH01 counter relay
- I02 counter input for CH02 counter relay
- I03 counter input for CH03 counter relay
- I04 counter input for CH04 counter relay

| | |
|---|---|
| | *NOTICE*<br>Avoid unforeseeable switching states.<br>Switch C, CF, CH, CI function blocks only at one single point in the program.<br>Otherwise, previous counts will be overwritten. |

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| D_ | Count direction<br>0: up counting<br>1: down counting | |
| SE | The starting value at SV is applied whenever there is a rising edge at this input | |
| RE | Reset<br>1: QV=0 | |
| **(DWord)** | | |
| SH | Upper Threshold Value | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| SL | Lower threshold value | |
| SV | Start value (Pre Set) | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |

1) Only on function blocks T, AC
2) Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| OF | Overflow<br>1: if QV ≥ SH | OF=1, if the actual value is greater than or equal to the upper threshold value; |
| FB | Fall below<br>1: if QV ≤ SL | FB=1, if the actual value is less than or equal to the lower threshold value; |
| CY | Carry<br>1: if QV > value range | If the value range is exceeded, the switch contact switches to status 1 for one cycle per rising edge detected. The function block retains the value of the last valid operation before the contact CY is set. |
| ZE | Zero<br>1: if QV = 0 | |
| **(DWord)** | | |
| QV | Current counter value in RUN mode | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |

| Assigning operands | Value outputs |
|---|---|
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| **Parameter set** | | |
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

### Signal diagram

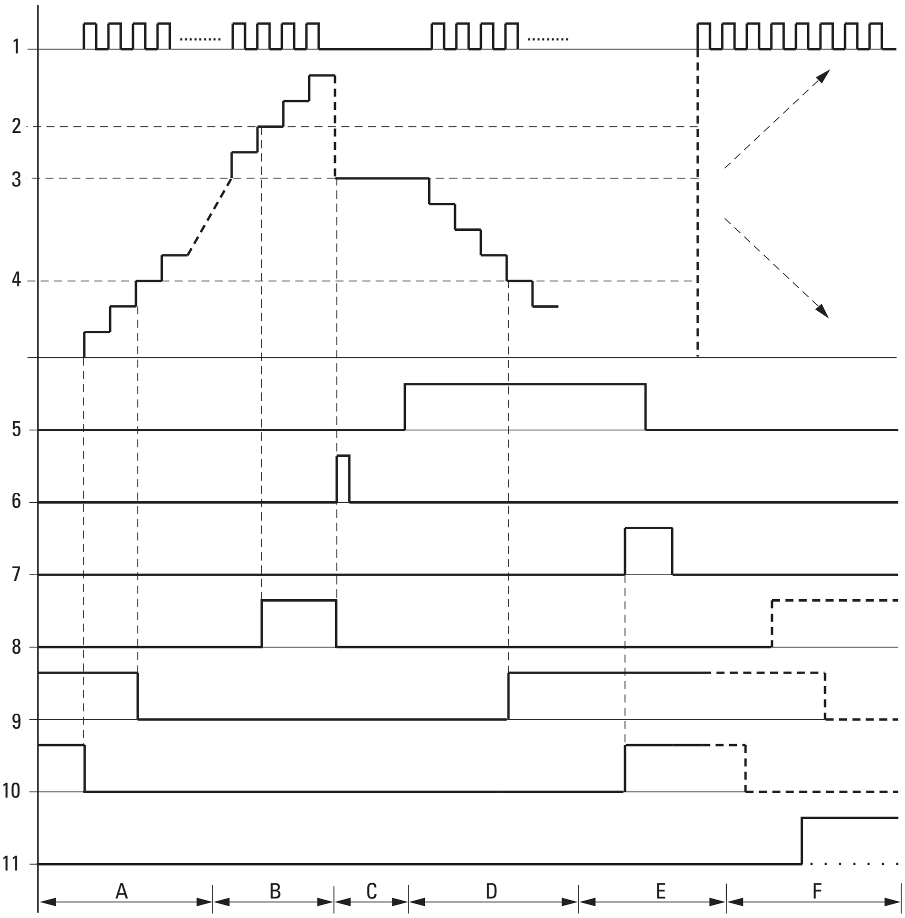



Figure 159: Signal diagram High-speed counter

Legend for Figure

1: One of the I01 through I04 device inputs

2: Upper threshold value SH.

3: Start value SV.

4: Lower threshold value SL.

5: Count direction, coil CH…D

6: Transfer start value, coil CH..SE.

7: Reset coil CH…RE

8: Contact (N/O) CH..OF: Upper limit threshold reached or exceeded.

9: Contact (N/O) CH..FB: Lower threshold value reached or undershot.

10: CH..ZE = 1, if actual value is zero.

11: CH..CY = 1, if the value is out of range

- Range A:

– The counter range has the value zero.

– The contacts CH..ZE (actual value equal to zero) and CH..FB (lower threshold value undershot) are active.

– The counter receives pulses and increases the actual value.

– CH..ZE drops out as well as CH..FB after the lower threshold value is reached.

• Range B:

– The counter relay counts upwards and reaches the upper threshold value.

The contact “upper threshold value” CH…OF becomes active.

• Range C:

– The coil CH..SE is briefly actuated and the actual value is set to the start value.

The contacts go to the respective position.

• Range D:

– The counting direction coil CH..D_ is actuated. If counting pulses are present, downward counting is initiated.

– If the lower threshold value is undershot, the contact CH..FB becomes active.

• Range E:

– The reset coil CH..RE is activated. The actual value will be set to zero.

– The contact CH…ZE is active.

• Range F:

– The actual value goes outside the value range of the counter relay.

– The contacts OF, FB, ZE become active according to the direction of the values (positive or negative).

### Retention

Counter relays can be operated with retentive actual values. You can select the number of retentive counter relays in *Project view/System settings tab*, please refer to → Page 1 System Settings tab. The retentive ACTUAL value will require 4 bytes of memory space. If a counter relay is retentive, the actual value will be retained when the operating mode switches from RUN to STOP and when the power supply is switched off. If the device is started in RUN mode, the counter relay will continue to work with the non-volatile actual value.

### See also

→ "C - Counter relay", page 281
→ "CH - High-speed counter", page 293
→ "CI - Incremental Counter", page 299
→ "Timing and counter relay example", page 557

### 6.1.2.4 CI - Incremental Counter

CI function blocks can be used to quickly count rising and falling edges forward and backward. This counting is independent of the cycle time.
You can set a lower and higher limit as comparison values for the incremental value counter, as well as a start value.
There are two incremental counters available.

→ Square wave count pulses are required to ensure a 1:1 mark to space ratio.
The signals at channels A and B must have an offset of 90°; otherwise the count direction will not be detected.

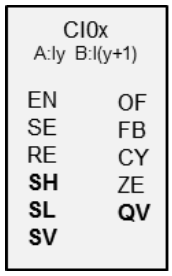
The maximum count frequency is 5000 Hz.

→ Note that the digital inputs I1 to I4 are permanently assigned to the incremental counter function blocks provided:

- I1: Counter input for counter CI01, channel A.
- I2: Counter input for counter CI01, channel B.
- I3: Counter input for counter CI02, channel A.
- I4: Counter input for counter CI02, channel B.

#### General information

easyE4 base devices provide 2 high-speed incremental encoder counters, CI01 through CI02. The high-speed up and down counters are internally hardwired with the digital inputs I01...I02 or I03...I04 and operate independently of the cycle time.




#### Operating principle

Incremental counters interpret rising and falling edges in order to identify the count direction. The count will be in the direction of the rising and falling edges.

The counter wiring must observe the following digital device input assignment:

I01 Counter input for the counter CI01 channel A.

I02 Counter input for counter CI01 channel B

I03 Counter input for counter CI02 channel A

I04 Counter input for counter CI02 channel B

The contacts will switch according to the actual value. The appropriate function block outputs switch according to the actual value determined. The counter relays enable a preset start value to be defined at the SV input.

Only square wave signals are permissible.

The mark-to-space ratio is 1:1.

The signals of channels A and B must be offset by 90°. Otherwise the count direction cannot be determined.

**Positive count direction**

If the rising edge at channel A is detected before the rising edge at channel B, the counter will count up. This means that the counter will be incremented by 1 after there is a rising edge at channel A followed by a rising edge at channel B. The same applies to the falling edges in sequence at channel A and channel B. The counter relay's result will be incremented and output at output QV.

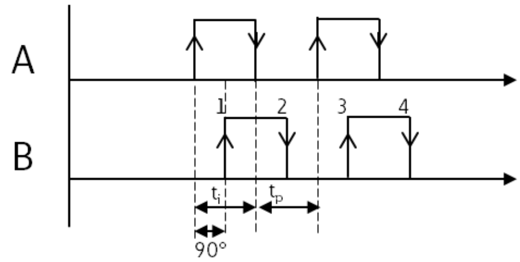



Figure 160: CI function block counting up; QV=QV+4

**Negative count direction**

If the rising edge at channel B is detected before the rising edge at channel A, the counter will count up. This means that the counter will be decremented by 1 after there is a rising edge at channel A followed by a rising edge at channel B. The counter relay's result will be decremented and output at output QV.

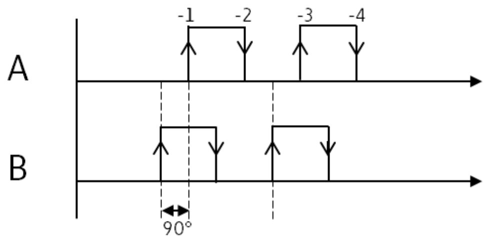



Figure 161: CI function block counting down; QV=QV-4

| |
|---|
| *NOTICE*<br>Avoid unforeseeable switching states.<br>Switch C, CF, CH, CI function blocks only at one single point in the program.<br>Otherwise, previous counts will be overwritten. |

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| SE | The starting value at SV is applied whenever there is a rising edge at this input | |
| RE | Reset<br>1: QV=0 | |
| **(DWord)** | | |
| SH | Upper Threshold Value | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| SL | Lower threshold value | |
| SV | Start value (Pre Set) | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |

| Operators | Bit inputs |
|---|---|
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| OF | Overflow<br>1: if QV ≥ SH | OF=1, if the actual value is greater than or equal to the upper threshold value; |
| FB | Fall below<br>1: if QV ≤ SL | FB=1, if the actual value is less than or equal to the lower threshold value; |
| CY | Carry<br>1: if QV > value range | If the value range is exceeded, the switch contact switches to status 1 for one cycle per rising edge detected. The function block retains the value of the last valid operation before the contact CY is set. |
| ZE | Zero<br>1: if QV = 0 | |
| **(DWord)** | | |
| QV | Current counter value in RUN mode | The pulses at channel A and channel B are counted. 2 pulses are counted per counting period.<br>Example: 2 pulses at channel A and 2 pulses at channel B; value at CI..QV = 4 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| Parameter set | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display + Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

## Other

### Signal diagram

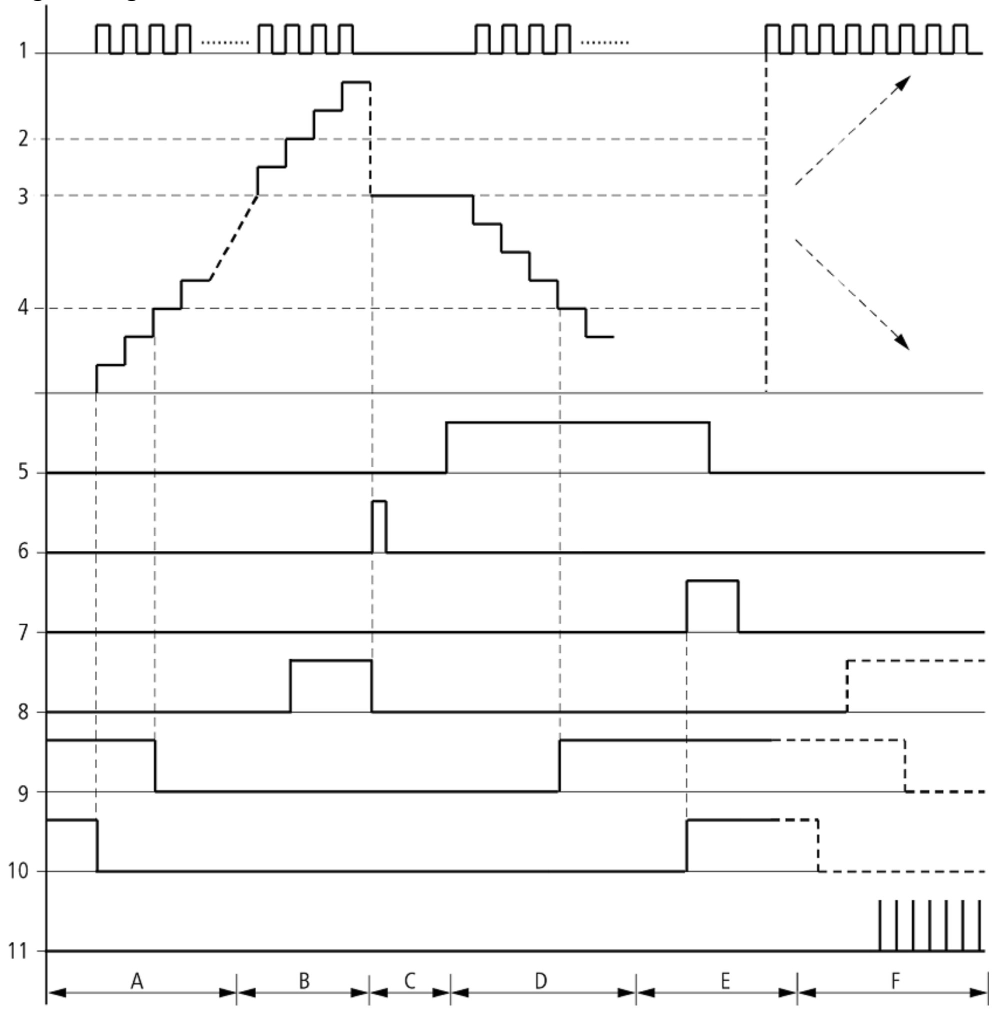



Figure 162: Signal diagram High-speed incremental value counter

Legend for Figure

1: One of the I01 through I04 device inputs

2: Upper threshold value SH

3: Start value SV

4: Lower threshold value SL

5: Transfer start value, coil CI..SE.

6: Reset coil CI…RE

7: Contact (N/O) CI..OF: Upper limit threshold reached or exceeded.

8: Contact (N/O) CI..FB: Lower threshold value reached or undershot.

9: CI..ZE = 1, if actual value is zero

10: CI..CY = 1, if the value is out of range.

• Range A:

– The counter range has the value zero.

– The contacts CI..ZE (actual value equal to zero) and CI..FB (lower threshold value undershot) are active.

– The counter relay receives pulses at I01 and I02 or at I03 and I04 and increments the actual value.

– CI..ZE drops out as well as CI..FB after the lower threshold value is reached.

• Range B:

– The counter relay counts upwards and reaches the upper threshold value.

The “upper setpoint value reached” contact CI..OF becomes active.

• Range C:

– The coil CI..SE is briefly actuated and the actual value is set to the start value.

The contacts go to the respective position.

• Range D:

– The counter relay receives pulses at I02 or I04 and decrements the actual value. The function block counts down.

– If the lower threshold value is undershot, the contact CI..FB becomes active.

• Range E:

– The reset coil CI..RE is activated. The actual value will be set to zero.

– The contact CI..ZE is active.

• Range F:

– The actual value goes outside the value range of the counter relay.

– The contacts OF, FB, ZE become active according to the direction of the values (positive or negative).

**Retention**

Counter relays can be operated with retentive actual values. You can select the number of retentive counter relays in *Project view/System settings tab*, please refer to → Page 1 System Settings tab. The retentive ACTUAL value will require 4 bytes of memory space. If a counter relay is retentive, the actual value will be retained when the operating mode switches from RUN to STOP and when the power supply is switched off. If the device is started in RUN mode, the counter relay will continue to work with the non-volatile actual value.

**See also**

→ Section "CI - Incremental Counter", page 299
→ Section "CF - Frequency counter", page 287
→ Section "CH - High-speed counter", page 293
→ Section "Timing and counter relay example", page 557

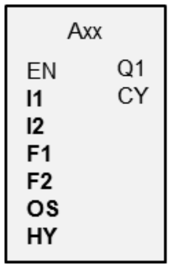
### 6.1.3 Arithmetic and analog function blocks

#### 6.1.3.1 A - Analog value comparator

An analog value comparator or threshold value switch is used, for example, to compare analog values or marker contents and switch when defined threshold values are reached.

**General**

easyE4 base devices provide 32 analog comparators, A01 through A32.
Analog comparators can be used to compare analog input values with a reference value.




**Operating principle**

The following comparisons are available:
Function block input I1 greater than, equal to, or less than function block input I2.

Using the factors F1 and F2 as inputs enables you to amplify and adjust the values of the function block inputs.

Function block input OS can be used as an offset for input I1.

The HY function block input is used for the positive and negative switching hysteresis of the input I2.

The contact Q1 switches if the condition of the comparison mode you have selected is fulfilled.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The check-box for the Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | |
| I1 | Comparison value 1 | Integer value range: -2,147,483,648 to +2,147,483,647 |
| I2 | Comparison value 2 | |
| F1 | Gain factor for I1 (I1 = F1 * value)<br>Default value = 1 | |
| F2 | Gain factor for I2 (I2 = F2 * value)<br>Default value = 1 | |
| OS | Offset for the value at I1,<br>$I1_{OS}$ = OS + actual value at I1; | |
| HY | Switching hysteresis for value at I2.<br>To calculate the hysteresis band (dead band) limited by the upper and lower hysteresis threshold), the function block takes into account the value HY as well as positive and negative components.<br>$I2_{HY}$ = Actual value at I2+ HY,<br>$I2_{HY}$ = Actual value at I2 - HY); | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |

1) Only on function blocks T, AC
2) Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating modes

| | Description | Note |
|---|---|---|
| LT: less than (I1 < I2) | Less than (I1 < I2) | |
| EQ: equal to (I1 = I2) | Equal to (I1 = I2) | |
| GT: greater than (I1 > I2) | Greater than (I1 > I2) | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | Status 1 if condition is fulfilled<br>(e.g. I1 < I2 with LT mode active) | |
| CY | $-2^{31} \leq I1 * F1 + OS \leq (2^{31}-1) \Rightarrow CY = 0$<br>$-2^{31} \leq I2 * F2 + HY \leq (2^{31}-1) \Rightarrow CY = 0$<br>$-2^{31} \leq I2 * F2 - HY \leq (2^{31}-1) \Rightarrow CY = 0$<br>Status 1 if the above permissible value range of the function block is exceeded. | If an out of range value is indicated with CY = 1, Q1 will remain 0. |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers [2)]<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

## Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

Signal diagrams

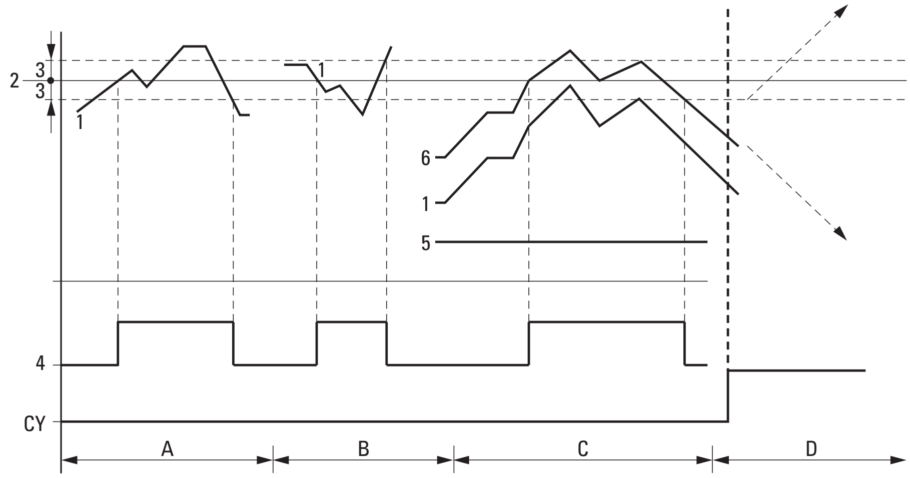



Figure 163: Analog comparator signal diagram

Legend for Figure

1: Actual value at I1

2: Setpoint value on I2

3: Hysteresis on HY

4: Switching contact Q1 (N/O contact)

5: Offset for value I1

6: Actual value plus offset

• Range A: Compare I1 greater than I2

– The actual value I1 increases.

– The contact switches when the actual reaches the setpoint value.

– The actual value changes and falls below the value of the setpoint value minus the hysteresis.

– The contact goes to the normal position.

• Range B: Compare I1 less than I2

– The actual value drops.

– The contact switches if the actual value reaches the setpoint value.

– The actual value changes and rises above the value of the setpoint value plus hysteresis.

– The contact goes to the normal position.

• Range C: Compare I1 with Offset greater than I2

– This example behaves as described in Range A. The offset value is added to the actual value.

– Compare I1 equal to I2 The contact switches on.

– If I1 is equal to I2, i.e., if the actual value is equal to the reference value: The contact will switch off.

– If the hysteresis limit is exceeded with the actual value rising.

– If the hysteresis limit is undershot with the actual value decreasing.

• Range D: I1 with offset leaves the permissible value range. The contact CY closes. CY opens as soon as I1 with offset is once more within the value range.

### Retention

The function block does not recognize retentive data.

### Example of a analog value comparator function block when using the EDP programming language

```
I01----A01Q1-----------------[ Q01
I02----A01CY-----------------S Q02
```

### Example of an AR configuration on a device display

When using the function block in the circuit diagram for the first time, use OK to automatically enter the display of function blocks on the device display, as shown in the following figure.

```
A02 GT +
>I1
>F1
>I2
>F2
>OS
>HY
```

Figure 164: Parameters on the display

Enter the function block settings here. The display contains the following elements:

| | |
|---|---|
| A02 | Function block: Analog comparator, number 02 |
| GT | Operating mode: Greater than |
| + | Parameter set can be called via the PARAMETERS menu |
| >I1 | Comparison value 1, is compared with the comparison value 2 at >I2, Value range: -2147483648… 2147483647 |
| >F1 | Gain factor for >I1 (>I1 = >F1. Value)<br>Value range: -2147483648… 2147483647 |
| >I2 | Comparison value 2 I1, Value range: -2147483648… 2147483647 |
| >F2 | Gain factor for >I2 (>I2 = >F2. Value) Value range: -2147483648 … 2147483647 |
| >OS | Offset (zero point offset) for the value of >I1 Value range: -2147483648… 2147483647 |
| >HY | Positive and negative switching hysteresis for comparison value I2, Value range: -2147483648… 2147483647 |

### See also

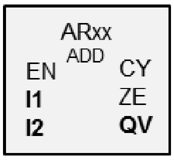
→ Section "AR - Arithmetic", page 312
→ Section "CP – Comparator", page 326
→ Section "LS - Value scaling", page 330
→ Section "MM - Min-/Max function", page 335
→ Section "PW - Pulse width modulation", page 345
→ Section "PM - Performance map ", page 339

### 6.1.3.2 AR - Arithmetic

The Arithmetic function block performs all four basic calculations.
The function block is provided with the two Boolean outputs mentioned above for controlling the calculation result that you wire as contacts in the circuit diagram.

#### General

easyE4 Base devices provide 32 arithmetic function blocks, AR01 through AR32. These function blocks can be used to carry out the four basic arithmetic operations: addition, subtraction, multiplication, and division.




#### Operating principle

The function block will apply the selected arithmetic operation to the values at function block inputs I1 and I2. If the calculation result exceeds the value range that can be represented, overflow signal contact CY will close and function block output QV will contain the value of the last valid operation. When the function block is called for the first time, the value at function block output QV will equal zero.

#### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | |
| I1 | Operand 1 | Integer value range: -2,147,483,648 to +2,147,483,647 |
| I2 | Operand 2 | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |

1) Only on function blocks T, AC
2) Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating modes

| | Description | Note |
|---|---|---|
| ADD - Adder | Adding<br>(I1 + I2 = QV) | 2174483647 + 1 = QV contains the last permissible value as an overflow has occurred. The carry bit AR..CY is set to 1. |
| SUB - Subtracter | Subtracting<br>(I1 - I2 = QV) | -2174483648 - 3 = QV contains the last permissible value as an overflow has occurred. The carry bit AR..CY is set to 1. |
| MUL - Multiplier | Multiplying<br>(I1 * I2 = QV) | 1000042 * 2401 = QV contains the last permissible value as an overflow has occurred. The carry bit AR..CY is set to 1. |
| DIV - Divider | Dividing<br>(I1 : I2 = QV) | 1024: 0 = QV contains the last permissible value as an overflow occurred. The carry bit AR..CY is set to 1.<br>10 : 100 = 0 |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| CY | Status 1 if the above value range is exceeded. | |
| ZE | Status 1 if the value of the function block output QV (the calculation result) equals zero | |
| **(DWord)** | | |
| QV | Current counter value in RUN mode | Integer value range:<br>-2,147,483,648 to<br>+2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| **Parameter set** | | |
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Addition example

42 + 1000 = 1042

2147483647 + 1 = Last valid value before this arithmetic operation, due to overflow (Carry) AR..CY = 1

-2048 + 1000 = -1048

**Subtraction example**

1134 - 42 = 1092

-2147483648 - 3 = Last valid value before this arithmetic operation, due to overflow (Carry) AR..CY = 1

-4096 - 1000 = -5096

-4096 - (-1000) = -3096

**Multiplication example**

12 x 12 = 144

1000042 x 2401 = Last valid value of this arithmetic operation, due to overflow (Carry) correct value = 2401100842 AR..CY = 1

-1000 x 10 = -10000

**Division example**

1024 : 256 = 4

1024 : 35 = 29 (The places after the decimal point are omitted.)

1024 : 0 = Last valid value before this arithmetic operation, due to overflow (Carry) (mathematically correct “Infinity”) AR..CY = 1

-1000 : 10 = -100

1000 : (-10) = -100

-1000 : (-10) = 100

10 : 100 = 0

**Example of an arithmetic function block when using the EDP programming language**

```
I 01----AR01CY------------------[ Q 01
I 02----AR02ZE------------------S Q 02
```

Figure 165: Wiring the contacts

**Example of an AR configuration on a device display**

When using the function block in the circuit diagram for the first time, use OK to automatically enter the display of function blocks on the device display, as shown in the following figure.

```
AR04 ADD +
>I1
>I2
QV>
```

Figure 166: Parameters on the device display

Enter the function block settings here. The display contains the following elements:

| | |
|---|---|
| AR04 arithmetic function block | Function block:Arithmetic |

| | |
|---|---|
| ADD + | Mode:Adder |
| + | Parameter set can be called via the PARAMETERS menu |
| >I1 | First value is associated with the value at I2 via the arithmetic operation.<br>Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| >I2 | Second value;<br>Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| >QV | Supplies the calculation result.<br>Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

### See also

→ Section "AR - Arithmetic", page 312
→ Section "CP – Comparator", page 326
→ Section "LS - Value scaling", page 330
→ Section "MM - Min-/Max function", page 335
→ Section "PW - Pulse width modulation", page 345

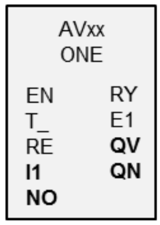
### 6.1.3.3 AV - Average

Only available on easySoft Version 7.10 or higher.

If this function block is not being shown in the leftmost pane in easySoft 7, make sure that you are using firmware version 1.10 or higher for the project.

#### General

easyE4 base devices provide 32 average function blocks, AV01 through AV32. Averaging is a method used to smooth data series, and is primarily used, for example, to smooth temperatures or production data recorded over several hours or days by removing high frequency components. Please note, however, that this function block is not intended for signal smoothing or for controllers – the FT function block should be used in those cases instead.




#### Operating principle

The average function block takes the values at function block input I1 and calculates the corresponding moving average. Every time there is a rising edge at function block input T_, the value at I1 is read and included in the calculation of the average value. Meanwhile, the maximum number of values to be included in the calculation must be specified using function block input NO. If this maximum number is reached, there will be two possibilities depending on the selected operating mode.

#### One-time mode

When using one-time mode, the function block will stop calculating the average value when done with the calculation, and function block output RY will be set to 1. This operating mode is primarily intended for calculating the average of a specific value range at periodic intervals. Accordingly, it is, for example, suitable for calculating the average day temperature every day (in which case a value of 24 would be selected for NO). The maximum absolute error is 0.5.

#### Continuous mode

When using continuous mode, the function block will continue calculating the average value with every new rising edge at T_. In this case, the moving average will be calculated for the window defined with NO, with the oldest value being eliminated and the newest one being added every time there is a rising edge. In other words, this makes it possible, with every new rising edge, to "look into the past" a number of edges = NO. Since it is not possible to store all the values in the aforementioned window, the calculation is made with an approximate calculation instead. Please note that, just like with one-time mode, function block output RY will also be set to 1 in this case as soon as the number of values NO is reached. This operating mode is suitable, among other things, for continuously calculating the average value of a temperature for a specific period of time (and a value of 24 would also be used for NO in this case).

The formulas used for the calculations are provided further below.
Even though the average value will be calculated only after the number of values to be included (NO) is reached, it will already be output at function block output QV during the startup phase (n < NO).
Please note that the selected number of values to be included (NO) should not be too large, as the larger NO is, the smaller smoothing factor SF will be, and the less the current value at I1 will be taken into account.
The currently calculated average value will be output at function block output QV. Meanwhile, function block output QN will indicate how many values were read at I1 and used for the calculation.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| T_ | Trigger input<br>When there is a rising edge at T_, the value at function block input I1 is taken and used in the average value calculation. | |
| RE | 1: Resets the number of values to be included, as well as the calculated average value; QN=0, QV=0, RY=0. | |
| **(DWord)** | | |
| I1 | Input value | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| NO | Maximum number of values that should be included when calculating the average value. | Integer value range:<br>0…+2 147 483 647 |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |

| Operators | Bit inputs |
|---|---|
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating modes

| | Description | Note |
|---|---|---|
| One-time mode | The average value calculation will be completed as soon as the specified maximum number of input values to be included (NO) is reached. | |
| Continuous operation | The average value will be calculated continuously even if the maximum number of input values to be included (NO) is reached. | |

The default setting is one-time mode.

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| RY | 1: The average value calculation has been completed, since the specified number of input values to be included has been reached. | |
| E1 | Error<br>1: If the value range for I1 or NO is exceeded. | |
| **(DWord)** | | |
| QV | The current average value | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| QN | The current number of values that were used in the averaging calculation | Integer value range:<br>0…+2 147 483 647 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| **Parameter set** | | |
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything.<br>The "Function block release by EN is necessary" option is enabled by default. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

#### How average values are calculated in the AV function block

For these examples, the maximum number of values used to calculate the average value (NO) will be 24.

Moreover, the examples use the sample temperature values in the table below and assume that they were (multiplied by 100) present and measured at function block input I1.

#### One-time mode

In one-time mode, the moving average is calculated using the following formula:

One-time mode average value $CMA(n) = ROUND\ [\ CMA_{n-1} + (I1_n - CMA_{n-1})/(n+1)\ ]$

CMA(n) = Currently calculated simple moving average
n = 1…NO
$I1_n$ = Value at function block input I1; e.g., temperature value

#### Continuous operation

In continuous mode, the smoothing factor is calculated first.

Smoothing factor SF = 2 / (NO+1)

SF = Smoothing factor; value between 0 and 1
NO = Maximum number of values to be included in the calculation

The average value is then calculated using the formula below:

Continuous operation average value $EMA(n) = ROUND\ [\ EMA_{n-1} + SF * (I1_n - EMA_{n-1})\ ]$

EMA(n) = Currently calculated exponential moving average
n = 1…NO
SF = Smoothing factor; value between 0 and 1
$I1_n$ = Value at function block input I1; e.g., temperature value

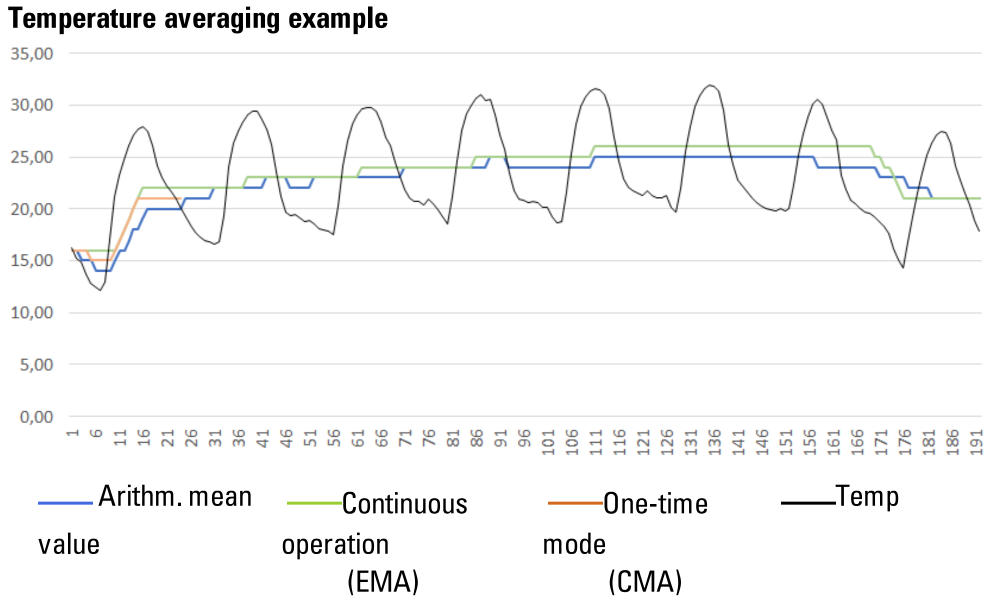



Figure 167: Sample curve for hourly temperature measurement, over 7 days

### One-time mode

In the example, the average value in one-time mode is calculated for the 24th value / CMA (23) as follows:

$$CMA(23) = ROUND\left[ CMA(22) + \frac{I1(23) - CMA(22)}{23 + 1} \right]$$

CMA(23) = ROUND [ 1889 + (2004 - 1889)/24 ] = ROUND [1893,792] = 1894

### Continuous operation

The smoothing factor in the example is SF = 2/(24+1) = 0.08.

In the example, the average value in continuous mode is calculated for the 24th value:

EMA(23) = ROUND [ EMA(22)+0.08*( I1(23) – EMA(22) ]

EMA(23) = ROUND [ 2035 + 0.08* (2004 – 2035) ]

EMA(23) = ROUND [2032.52 ] = 2033

Table 72: Example temperature values

| Day | Hour | Temperature | Total Temp | Arithmetic Mean value | Duration-operation | Once-operation |
|---|---|---|---|---|---|---|
| 20 | 0 | 16 | 16 | 16.00 | 16 | 16 |
| 20 | 1 | 15 | 31 | 15.50 | 16 | 16 |
| 20 | 2 | 15 | 46 | 15.33 | 16 | 15 |
| 20 | 3 | 14 | 60 | 15.00 | 16 | 15 |
| 20 | 4 | 13 | 73 | 14.60 | 15 | 15 |
| 20 | 5 | 12 | 85 | 14.17 | 15 | 14 |
| 20 | 6 | 12 | 97 | 13.86 | 15 | 14 |
| 20 | 7 | 13 | 110 | 13.75 | 15 | 14 |
| 20 | 8 | 17 | 127 | 14.11 | 15 | 14 |
| 20 | 9 | 21 | 148 | 14.80 | 15 | 15 |
| 20 | 10 | 23 | 171 | 15.55 | 16 | 16 |
| 20 | 11 | 25 | 196 | 16.33 | 17 | 16 |
| 20 | 12 | 26 | 222 | 17.08 | 18 | 17 |
| 20 | 13 | 27 | 249 | 17.79 | 18 | 18 |
| 20 | 14 | 28 | 277 | 18.47 | 19 | 18 |
| 20 | 15 | 28 | 305 | 19.06 | 20 | 19 |
| 20 | 16 | 27 | 332 | 19.53 | 20 | 20 |
| 20 | 17 | 26 | 358 | 19.89 | 21 | 20 |
| 20 | 18 | 24 | 382 | 20.11 | 21 | 20 |
| 20 | 19 | 23 | 405 | 20.25 | 21 | 20 |
| 20 | 20 | 22 | 427 | 20.33 | 21 | 20 |
| 20 | 21 | 22 | 449 | 20.41 | 21 | 20 |
| 20 | 22 | 21 | 470 | 20.43 | 21 | 20 |
| 20 | 23 | 20 | 490 | 20.42 | 21 | 20 |
| 20 | 0 | 19 | 493 | 20.54 | 21 | – |
| 21 | 1 | 18 | 496 | 20.67 | 21 | – |
| 21 | 2 | 18 | 499 | 20.79 | 21 | – |
| 21 | 3 | 17 | 502 | 20.92 | 20 | – |
| 21 | 4 | 17 | 506 | 21.08 | 20 | – |
| 21 | 5 | 17 | 511 | 21.29 | 20 | – |
| 21 | 6 | 17 | 516 | 21.50 | 20 | – |
| … | | … | … | … | … | – |

### Retention

The function block does not recognize retentive data.

**See also**

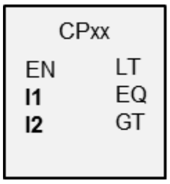
→ Section "A - Analog value comparator", page 306
→ Section "CP – Comparator", page 326
→ Section "LS - Value scaling", page 330
→ Section "MM - Min-/Max function", page 335
→ Section "PM - Performance map ", page 339
→ Section "PW - Pulse width modulation", page 345
→ Section "FT - PT1-Signal smoothing filter ", page 358

#### 6.1.3.4 CP – Comparator

This function block is used to compare variables and/or constants with each other.

**General**

easyE4 base devices provide 32 comparator function blocks (Compare) CP01 to CP32.
Comparators are used to compare variables and constants with each other and output the relationship between them: Less than / Equal to / Greater than.




**Operating principle**

The function block compares values present at the inputs I1 and I2. The following contacts close depending on the comparison result:

- I1 greater than I2, GT contact closes.
- I1 equal to I2, EQ contact closes.
- I1 less than I2, LT contact closes.

**The function block and its parameters**

**Function block inputs**

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | |
| I1 | Comparison reference value | Integer value range: -2,147,483,648 to +2,147,483,647 |
| I2 | Comparison value | |

**Assigning operands**

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)] NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |

| Operators | Value inputs |
|---|---|
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| LT | Less Than<br>1: if I1 < I2 | |
| EQ | Equal<br>1: if I1 = I2 | |
| GT | Greater Than<br>1: if I1 > I2 | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)[2] SN - Output bit via NET (send) | x |
| N - Network marker bit[2] | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| **Parameter set** | | |
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display + Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

### Retention

The function block does not recognize retentive data.

### Example of a comparator function block when using the EDP programming language

The contact of the function block is sent to markers.

```
CP12LT-----------------------[ M 21
CP12LT-----------------------[ M 22
CP12GT-----------------------┬ R M 21
                             └ R M 22
```

Figure 168: Wiring the contacts

### Example of a CP configuration on a device display

When using the function block in the circuit diagram for the first time, use OK to automatically enter the display of function blocks on the device display, as shown in the following figure.

```
CP12 +
>I1
>I2
```

Figure 169: Parameters on the display

Enter the function block settings here. The display contains the following elements:

| | |
|---|---|
| CP12 comparator | Function block: Value comparator, number 12 |
| + | Parameter set can be called via the PARAMETERS menu |
| >I1 | Reference value to which the comparison is made<br>Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| >I2 | Comparison value; I2 is compared with I1<br>Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

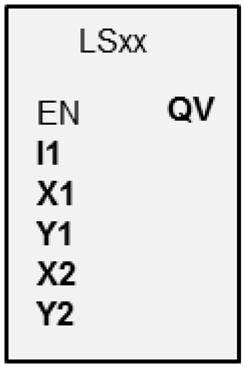
**See also**

→ Section "A - Analog value comparator", page 306
→ Section "AR - Arithmetic", page 312
→ Section "AV - Average", page 317
→ Section "LS - Value scaling", page 330
→ Section "MM - Min-/Max function", page 335
→ Section "PW - Pulse width modulation", page 345
→ Section "PM - Performance map ", page 339

### 6.1.3.5 LS - Value scaling

#### General

easyE4 base devices provider 32 value scaling function blocks, LS01 through LS32.




These function blocks can be used to transfer values from one value range to another. More specifically, a value scaling function block will take one of the mathematical relationships you have specified and use it to scale the value at input LS..I1 in order to then output it, either as smaller value or larger value, at output LS..QV. The mathematical relationship is based on a straight line defined by the coordinate pairs X1, Y1 and X2, Y2 (see under “The mathematical relationship is:”).
A typical application is the conversion of values, such as 0…20 mA to 4…20 mA. An easy800 device is provided with 32 value scaling function blocks.

#### Operating principle

"EN = 1" starts the function block.
"EN = 0" initiates a reset in which the output **QV** is reset to 0.

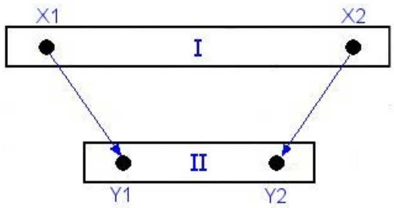



Figure 170: Figure: Scaling the input values - reducing

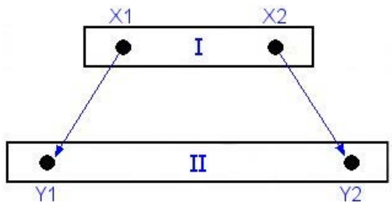



Figure 171: Scaling the input values - increasing

① Source range

② Destination range

The mathematical relationship is follows:

$$Y = m * X + Y0$$

$$m = \frac{Y2 - Y1}{X2 - X1} \qquad Y0 = \frac{X2 * Y1 - X1 * Y2}{X2 - X1}$$

m = Gradient

Y0 = Y offset when X = 0

X1, Y1 = First value pair

X2, Y2 = Second value pair

g = Straight line with positive gradient

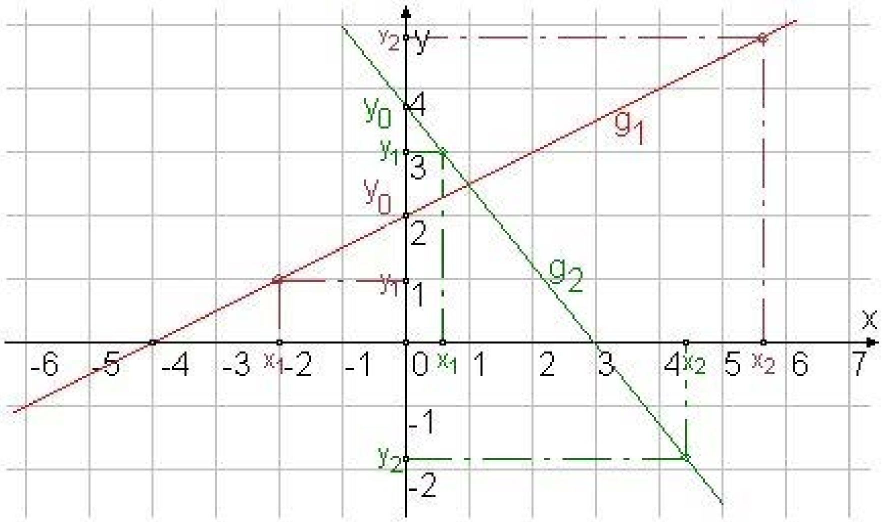



Figure 172: Mathematical interrelationship

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | **(DWord)** |
| I1 | Input value, value range: 32-bit | Integer value range: -2,147,483,648 to +2,147,483,647 |
| X1 | First scale; data point 1 | Value range: 32 bits |
| Y1 | First scale; data point 2 | |
| X2 | Second scale; data point 1 | |
| Y2 | Second scale; data point 2 | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)] NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |

| Operators | Bit inputs |
| --- | --- |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
| --- | --- | --- |
| **(DWord)** | | |
| QV | Delivers the scaled input value | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
| --- | --- |
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
| --- | --- |
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

**Parameter set**

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Interrupt source | Used to select device inputs I1 through I8 as a trigger for the interrupt | |
| Edit interrupt routine | Clicking on the button will open the interrupt routine in the Programming view | |
| Simulation possible | | |

**Other**

**Retention**

The function block does not recognize retentive data.

**Application example LS**

An analog pressure sensor I1 in a tank delivers a value ranging from 0 (empty) to 10000 (full). When the cylindrical, upright tank is completely full, it holds 600 liters. The purpose is for the current fill level to be converted to liters. The relationship between the pressure and the fill level, and accordingly the volume as well, is linear, meaning that an LS function block can be used.

The parameters would need to be configured as follows: X1=0, X2= 10000, Y1=0, Y2=600

QV will then deliver the fill level in liters.

**See also**

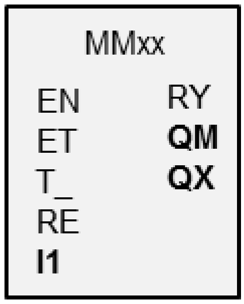
→ Section "LS - Value scaling", page 330
→ Section "AR - Arithmetic", page 312
→ Section "CP – Comparator", page 326
→ Section "MM - Min-/Max function", page 335
→ Section "PW - Pulse width modulation", page 345
→ Section "PM - Performance map ", page 339

### 6.1.3.6 MM - Min-/Max function

#### General

easyE4 base devices provide 32 Min-/Max function blocks, MM01 through MM32.

These function blocks can be used to determine the maximum value and the minimum value of a changing analog value. This makes it, for example, easy to conveniently determine the magnitude of the pressure fluctuations inside a system.




#### Operating principle

If the function block is enabled, the current value at function block input I1 will be compared with the existing minimum value and maximum value. If the current value falls below or exceeds these values respectively, it will accordingly be set as the new minimum value or maximum value. Only one minimum value and one maximum value will be stored in the function block at any one time.

Both values will be zero when the measurement starts. Moreover, they can be reset to zero using the RE input.

You can run the calculation cyclically or only when there are rising edges at function block input T_. The typical application is monitoring a process value cyclically.

#### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |
| ET | Enable trigger<br>0: Calculates Min/Max every time the function block is called; trigger input T_ is deactivated<br>1: Calculates Min/Max only when there is a rising edge at T_; trigger input T_ is activated | Typically, an ET = 0 automatic trigger configuration is used |
| T_ | Trigger input<br>Min/Max are calculated when there is a rising edge at T_, provided that ET = 1 | The fastest this can be done is every second cycle, since a switch from 0 to 1 is required at T_. |
| RE | 1: Sets the internal Min/Max values to 0 | |
| **(DWord)** | | |
| I1 | Analog value used for the Min/Max comparison | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| RY | Event message signaling that a new minimum or maximum value has been entered | This message will only be displayed for one cycle |
| **(DWord)** | | |
| QM | Minimum value of I1 that was sampled during the active period | |
| QX | Maximum value of I1 that was sampled during the active period | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

### Retention

The function block does not recognize retentive data.

**See also**

→ Section "MM - Min-/Max function", page 335
→ Section "AR - Arithmetic", page 312
→ Section "CP – Comparator", page 326
→ Section "LS - Value scaling", page 330
→ Section "PW - Pulse width modulation", page 345
→ Section "PM - Performance map ", page 339

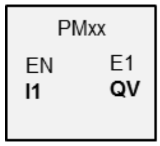
### 6.1.3.7 PM - Performance map

Only available on easySoft Version 7.10 or higher.

If this function block is not being shown in the leftmost pane in easySoft 7, make sure that you are using firmware version 1.10 or higher for the project.

#### General

easyE4 base devices provide 4 performance map function blocks, PM01 through PM04. This characteristic curve function is implemented by reading the value at function block input I1, looking up the corresponding output value in a reference value table, and outputting this output value at function block output QV.




#### Operating principle

The performance map function block can be used to implement a characteristic curve function. This characteristic curve function is implemented by reading the value at function block input I1, looking up the corresponding output value in a reference value table, and outputting this output value at function block output QV. The reference value table needs to be filled with at least 2 and at most 32 values for I1 and QV beforehand. If a value that is not found in the table is present at the function block input, the operating mode being used will determine which value fits the best and will be output at the function block output.

An example is used in the following sections in order to illustrate which operating modes are available and how they interpret the values at the function block input.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | |
| I1 | Input value | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating modes

The operating mode will decide on the output value if the value at function block input I1 does not match one of the I1 values in the reference value table exactly.

| | Description |
|---|---|
| Interpolating | The average value between the next higher and next lower value for I1 in the reference value table will be output at function block output QV. |
| Next higher value | The function block will look for the next higher value for I1 in the reference value table and output the corresponding assigned value at function block output QV. |
| Next lower value | The function block will look for the next lower value for I1 in the reference value table and output the corresponding assigned value at function block output QV. |
| Nearest value | The function block will look for the nearest value for I1 in the reference value table and output the corresponding assigned value at function block output QV.<br>If the value at I1 is exactly in the middle between two reference values in the table, the higher value will be output. |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| E1 | Error<br>1: If QV exceeds the value range | |
| **(DWord)** | | |
| QV | Output value determined from the reference value table based on input value I1. | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| **Parameter set** | | |
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything.<br>The "Function block release by EN is necessary" option is enabled by default. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

#### Retention

The function block does not recognize retentive data.

#### PM function block example: How the operating mode affects the results

Say that the following characteristic curve needs to be implemented using the PM function block. To do this, 32 mapped values are defined in the reference value table.

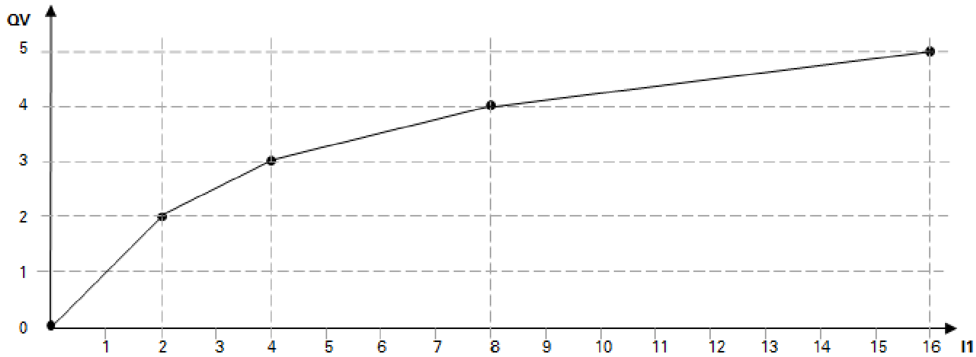



Figure 173: Example of a characteristic curve for the PM function block

Example of a reference value table in which QV values are mapped to I1 values

| | I1 | QV |
|---|---|---|
| 1 | 0 | 0 |
| 2 | 2 | 2 |
| 3 | 4 | 3 |
| 4 | 8 | 4 |
| 5 | 16 | 5 |
| ... | ... | ... |
| 31 | 26 | 10 |
| 32 | 30 | 12 |

The following table shows how the operating mode affects the values at function block output QV if the characteristic curve from the example is implemented using the defined reference value table. With the following values at the function block input:

| Value at I1 | Value at QV as a function of the operating mode |
|---|---|
| 1 | Interpolating: 1<br>Next higher value: 2<br>Next lower value: 0<br>Nearest value: 2 |
| 3 | Interpolating: 3<br>Next higher value: 3<br>Next lower value: 2<br>Nearest value: 3 |
| 5 | Interpolating: 4 |

| Value at I1 | Value at QV as a function of the operating mode |
|---|---|
| | Next higher value: 4<br>Next lower value: 3<br>Nearest value: 3 |
| 8 | Interpolating: 4<br>Next higher value: 4<br>Next lower value: 4<br>Nearest value: 4 |
| 27 | Interpolating: 11<br>Next higher value: 12<br>Next lower value: 10<br>Nearest value: 10 |

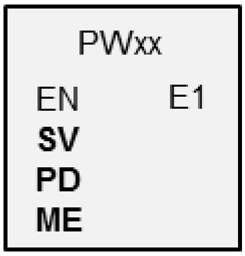
### See also

→ Section "A - Analog value comparator", page 306
→ Section "AV - Average", page 317
→ Section "CP – Comparator", page 326
→ Section "LS - Value scaling", page 330
→ Section "MM - Min-/Max function", page 335
→ Section "PW - Pulse width modulation", page 345

### 6.1.3.8 PW - Pulse width modulation

#### General

easyE4 base devices provide two function blocks for pulse width modulation PW01…PW02.
These function blocks generate a pulse train with a constant period. The main application for PW function blocks is when working with devices that feature transistor outputs.
However, PW function blocks can also be used with devices featuring relay outputs. Due to the relay's ON and OFF times, a long period duration and a long minimum ON duration are both possible with these devices.
Function block PW is used, among other things, to set up PID controllers with a pulse output for proportional actuators.
When doing this, the PW function block is used together with the DC (PID controller) function block.




#### Operating principle

The PW function block modulates the pulse duty factor of a square wave signal, and thus changes the mark to space ratio and pulse duration.
The period duration of the signal stays constant.

You define the period duration at the PD input. The on time or pulse duration is proportional to the manipulated variable at the SV input. You can also define the minimum on duration at the ME input.

A hardware output is assigned to each function block:

PW01 -> Q01, PW02 -> Q02

The function block causes the direct output of the calculated value at the hardware output.

→ If you use a PW function block with its permanently assigned Q1 or Q2 output, you cannot make any additional associations of these outputs in the program.
A status change at Q1 or Q2 that is caused by the circuit diagram is suppressed in favor of the higher priority status change caused by the function block.

**DANGER**
**UNFORESEEABLE SWITCHING STATES AT OUTPUT**
When using the PW function block, strictly observe the separate assignment of the outputs if other hardware-dependent function blocks are used, such as the PO function block.

If this is not observed, unforeseeable switching states may occur at the output concerned.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block.<br>0: Output Q1 or Q2 switches to a state of 0. | The checkbox for the Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | |
| SV | Manipulated variable<br>Value range: 0 to 4095 (12 bit) of this value range corresponds to 0...100% of the period duration. Notes on the normalization of the manipulated variable are provided in the previous section Manipulated variable SV. | No signals are output at Q1 and Q2 if SV=0 or SV<ME, and the output concerned will stay at 0. |
| PD | Period duration [ms]<br>When the value is 0, no pulses are output at Q1 and Q2.<br>The minimum period duration for a device with transistor outputs is 5 ms.<br>(the resulting max. frequency is 200 Hz). | Value range: 0…65535 |
| ME | Minimum On duration [ms]<br>The minimum on duration for devices with transistor outputs is 0.1 ms.<br>This on duration applies if ME is not set or set to 0.<br>If ME=1, the on duration is 1ms etc..<br>The minimum on duration possible essentially depends on the downstream electronics. | Value range: 0…65535 |

### Manipulated variable SV

The value range from 0 to 4095 of the manipulated variable SV corresponds to 0 to 100% of the period duration.

If you wish to control the pulse duration with the DC.. PID controller, you can associate the DC..QV output directly with the input PW..SV. This kind of application does not require any normalization since the DC..QV covers the same value range between 0 and 4095.

If you wish to control the pulse duration via an analog input that has a value range between 0 and 1023, you will have to normalize this value first of all. If you require an integral normalization factor, in this case the factor of 4, only multiplication with the AR arithmetic function block is required.

If you wish to control the pulse duration with a signed value you will have to normalize this value first of all with an LS value scaling function block.

If the actual value specified via SV for the pulse duration is shorter than the minimum on duration, the output at Q1 or Q2 is accordingly 0. The status of the PW..E1 contact must be observed.

If the off duration of the pulse at the output is less than the minimum off duration, Q1 or Q2 are permanently on. The status of the PW..E1 contact must be observed.

### Parameter limits for period duration and minimum ON duration

Table 73: Parameter limit values for period duration and minimum on duration

| | Period duration [ms] | Minimum On duration [ms] | Note |
|---|---|---|---|
| **Base device** | | | |
| EASY-E4-UC-... | Min. zz<br>Max. 65535 | Min. zz<br>Max. 65535 | Period duration<br>When the value is 0, no pulses are output at Q1 and Q2.<br>minimum contact closing time<br>Can be selected within the valid limits |
| EASY-E4-DC-... | Min. zz<br>Max. 65535 | Min. zz<br>Max. 65535 | |
| EASY-E4-AC-... | Min. zz<br>Max. 65535 | Min. zz<br>Max. 65535 | |

### Minimum period duration PD

The minimum period duration is 5 ms.

### Minimum contact closing time ME

The minimum on duration is limited to zz (100) µs when a very low manipulated variable is set at input SV.

### Minimum on duration = Minimum off duration

The above hardware restriction also applies to the minimum off duration if a very high manipulated variable at input SV generates a correspondingly long pulse duration. If a large mark-space ratio of this kind is almost as long as the period duration, the minimum off duration is 100 µs.

If the duration is below the set minimum on duration or minimum off duration, the Boolean control output E1 is set to 1. This control output E1 is used for monitoring during commissioning and does not have to be connected.

### Period duration to minimum ON duration ratio

The ratio of the period duration/minimum on duration (P/M) determines which percentage of manipulated variables have no effect.
The minimum on duration must be set as low as possible so that P/M is as high as possible.

If, however, a very short on duration has no effect on the connected actuator, it is advisable to suppress these short on phases in order to reduce the load on the hardware. In this case, the period duration should not be set too low.

### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| E1 | Error output<br>1: if the value is below the minimum on duration or minimum off duration. | The range limits are checked irrespective of the edge change on the Boolean EN input. |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| **Parameter set** | | |
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation not possible | | |

**See also**

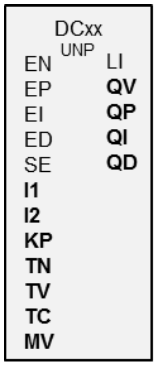
→ Section "PW - Pulse width modulation", page 345
→ Section "AR - Arithmetic", page 312
→ Section "CP – Comparator", page 326
→ Section "LS - Value scaling", page 330
→ Section "MM - Min-/Max function", page 335
→ Section "PM - Performance map ", page 339

### 6.1.4 Open-loop and closed-loop function blocks

#### 6.1.4.1 DC - PID controller

**General**

easyE4 The base devices provide 32 PID controller function blocks DC01...DC32.




**Operating principle**

A closed-loop control circuit with a PID controller consists of the following components:

- Setpoint (reference variable),
- Actual value (controlled variable),
- System deviation = (setpoint–actual value),
- PID controller,
- Control system (e.g. PTn system),
- Disturbance variables.

The PID controller operates on the basis of the equation of the PID algorithm. According to this, the manipulated variable Y(t) is the result of the calculation of the proportional component, an integral component and a differential component.

PID controller equation:

$$Y(t) = YP(t) + YI(t) + YD(t)$$

| | |
|---|---|
| Y(t)= | calculated manipulated variable with scan time t |
| YP(t)= | Value of the proportional component of the manipulated variable with scan time t |
| YI(t) = | Value of the integral component of the manipulated variable with scan time t |
| YD(t)= | Value of the differential component of the manipulated variable with scan time t |

**Proportional component**

The proportional component YP is the product of the gain (Kp) and the control difference (e). The control difference is the difference between the setpoint (Xs) and the actual value (Xi) at a specified scan time. The equation used by the device for the proportional component is as follows:

| | | |
|---|---|---|
| YP(t) = Kp * [Xs(t) - Xi(t)] | Kp = | Proportional gain |
| | Xs(t) = | SETPOINT value at scan time t |
| | Xi(t) = | ACTUAL value at scan time t |

#### Integral component

The integral component YI is proportional to the sum of the control difference over time. The equation used by the device for the integral component is as follows:

YI(t) = Kp * Tc / Tn * [Xs(t) - Xi(t)] + YI (t-1)]

$K_p$ = Proportional gain
Tc = Scan time
Tn = Reset time (also known as integration time).
Xs(t) = Setpoint with scan time t
Xi(t) = Actual value with scan time t
YI(t—1)= Value of the integral component of the manipulated variable with scan time t – 1

#### Differential component

The differential component YD is proportional to the change in the control difference. So as to avoid step changes or jumps in the manipulated variable caused by the differential behavior when the setpoint is changed, the change of the actual value (the process variable) is calculated and not the change in the control difference. This is shown by the following equation:

YD(t) = Kp x Tv / Tc x (Xi (t-1) - Xi(t))

$K_p$ = Proportional gain
Tc = Scan time
Tv = Rate time of the control system (also called the differential time)
Xi(t) = Actual value with scan time t
Xi(t-1) = Actual value with scan time t - 1

In order for a PID controller to work, it must be enabled with DC_EN =1. The PID controller will provide manipulated variable QV as an output variable. If the DC..EN coil is not active, the entire PID controller will be disabled and reset. The manipulated variable at the QV output will assume a value of 0. Function block inputs DC_EP, DC_EI, and DC_ED all need to be active in order for the proportional term, integral term, and derivative term to be calculated.

Example: If only coils DC…EP and DC…EI are activated, the PID controller operates as a PI controller.

A deactivation of the I and D component is linked with a reset. The controller is assigned parameters with the standard variables Kp [%], TN [0.1 s] and TV [0.1 s].

The device calculates the manipulated variable every time the scan time TC has elapsed. If the scan time is zero, the manipulated variable is calculated every cycle.

The controller can also be run in UNP and BIP modes, and also controlled in Manual mode.

### Manual mode of the PID controller

A value must be present at the MV function block input in order to set the manipulated variable directly. If the function block input DC…SE is activated, the value at MV is transferred directly as manipulated variable QV. This value is present for as long as the DC…SE coil is activated or the value at the MV input is changed. If DC..SE is no longer triggered, the control algorithm is active again.

→ Extreme changes in the manipulated variable can occur when the manual manipulated variable is transferred or deactivated.

→ If the function block is running in UNI (unipolar) mode, a negative signed manipulated variable value will be output as the value zero.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |
| EP | 1: Activates the proportional term | |
| EI | 1: Activates the integral term | |
| DF | 1: Activates the derivative term | |
| SE | 1: Accept manual manipulated variable | |
| **(DWord)** | | |
| I1 | Setpoint | Value range: -32768 ... +32767 |
| I2 | Actual value | Value range: -32768 ... +32767 |
| KP | Proportional gain Kp [%] | Value range: 0 ... 65535<br>The value 100 corresponds to a KP (factor) of 1. |
| TN | Reset time Tn [0.1 s] | Value range: 0 ... 65535 |
| TV | Rate time Tv [0.1 s] | Value range: 0 ... 65535 |
| TC | Scan time = Time between function block calls. Value range: 0.1 s - 6553.5 s.<br>If the value 0 is specified, the scan time is determined by the program cycle time. | |
| MV | Manual manipulated variable<br>Value range: -4096 ... +4095 | When UNP mode is selected: and negative values are entered at MV, the function block returns a zero at output QV. |

### KP Proportional gain factor

The input KP is used to define a proportional gain factor.
The value <100> corresponds to a KP factor of 1, the value 50 corresponds to a KP of 0.5 etc.

### Scan time Tc

The TC input defines the time between the function block calls. Values between 0.1s and 6553.5s can be defined.
If the TC scan time is set to 0, the program cycle time defines the time difference between the function block calls. This may cause irregularities in the control response as the program cycle time is not always constant. The ST (set cycle time) function block should be used to set a constant program cycle time, please refer to → "ST - Set cycle time", page 503.

→ A combination of two devices easyE4 is ideal for applications requiring lengthy calculations or visualizations such as PID closed-loop control tasks with the PID controller and also visualization functions at the same time.
In these kinds of applications, move the time consuming calculations to a second device, possibly without an integrated display, which you can connect via NET.

### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |

| Operators | Bit inputs |
|---|---|
| nN - NET marker bit[2] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating mode

| | Description | Note |
|---|---|---|
| **Operating mode** | | |
| UNP | The manipulated variable is output as a unipolar 12-bit value. . | Value range: 0 ... 4095 |
| BIP | The manipulated variable is output as a bipolar 13-bit value. | Value range: -4096 ... +4095 |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| LI | 1: if the value range of the medium-voltage is exceeded. | |
| **(DWord)** | | |
| QV | Manipulated variable | Integer value range<br>for operating mode UNP: 0...+4095 (12 Bit)<br>for operating mode BIP: -4096...+4095 (13 Bit) |
| QP | Proportional component of the manipulated variable<br>Can be used for diagnostic purposes | |
| QI | Integral component of the manipulated variable<br>Can be used for diagnostic purposes | |
| QD | Differential component of the manipulated variable<br>Can be used for diagnostic purposes | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers [2]<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

### Retention

The function block does not recognize retentive data.

### Example of a PID controller with the programming method EDP

```
M 51----------------------------┬-[ DC02EN
                                ├-[ DC02EP
                                ├-[ DC02EI
                                └-[ DC02ED
M 52------------------------------[ DC02SE
```

Figure 174: Wiring the function block coils

The function block coils are activated by markers.

```
DC02LI----------------------------[ S M 96
```

Figure 175: Wiring of the function block contact

The message of the function block is sent to a marker.

**Example of a PID controller configuration on a device display**

When using the function block in the circuit diagram for the first time, use OK to automatically enter the display of function blocks on the device display, as shown in the following figure.

```
DC02 UNP +
>I1
>I2
>KP
>TN
>TV
>TC
>MV
QV>
```

Figure 176: Parameters on the device display

Enter the function block settings here. The display contains the following elements:

| | |
|---|---|
| DC02 | Function block: PID controller, number 02 |
| UNP | Operating mode: Unipolar |
| + | Parameter set can be called via the PARAMETERS menu |
| >I1 | SETPOINT of the PID control: -32768…+32767 |
| >I2 | ACTUAL value of the PID control: -32768…+32767 |
| >KP | Proportional gain Kp; 0…65535,<br>in %; Example: The value 1500 is processed in the function block as 15. |
| >TN | Reset Time Tn: 0… 65535,<br>in 100 ms; Example: The value 250 is processed in the function block as 25 s. |
| >TV | Rate time TV: 0…65535,<br>in 100 ms; Example: The value 20 is processed in the function block as 2 s. |
| >TC | Scan Time Tc: 0…65535, in 100 ms |
| >MV | Manual manipulated variable: -4096… +4095 |
| QV> | Manipulated variable:<br>• unipolar: 0…4095<br>• bipolar: -4096…+4095 |

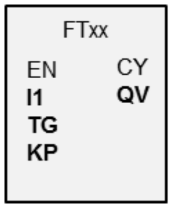
**See also**

→ Section "DC - PID controller", page 351
→ Section "FT - PT1-Signal smoothing filter ", page 358
→ Section "TC - Three step controller", page 379
→ Section "VC - Value limitation ", page 384
→ Section "BC - Block compare", page 388
→ Section "BV - Boolean operation", page 444
→ Section "PO - Pulse output", page 364

### 6.1.4.2 FT - PT1-Signal smoothing filter

#### General

easyE4Base devices provide 32 PT1 signal smoothing filter function blocks FT01...FT32.
The function block smoothes noisy signals such as analog input signals. It operates as a low pass filter.




#### Operating principle

The signal to be smoothed is added via input I1. The smoothed output value is transferred to QV.

EN=1 starts the function block. EN=0 initiates a reset in which output QV is set to a value of 0.

You can use the TG input to set the recovery time. This recovery time is the time over which smoothing will be applied, and it should not be longer than necessary, as signals will be delayed more than is actually necessary for smoothing. Please note that delays are an unavoidable side effect of signal smoothing.

The input KP is used to define a proportional gain factor. The input signal I1 is multiplied with this factor. The value <100> corresponds to a KP of 1.

The PT1 delayed output value is provided at output QV.

If the function block is called for the first time when the device is started or after a reset, this will result in the delay value being initialized with the input value (the PT1 delay does not start with a value of zero). In other words, the output value at QV will equal the input value at I1 during the first processing cycle. This will speed up the PT1 starting behavior.

#### Step response of the function block

The step response of the FT-PT1 function is an e function. After a time t = Tg the normalized output value is 0.63 QV/QVmax.

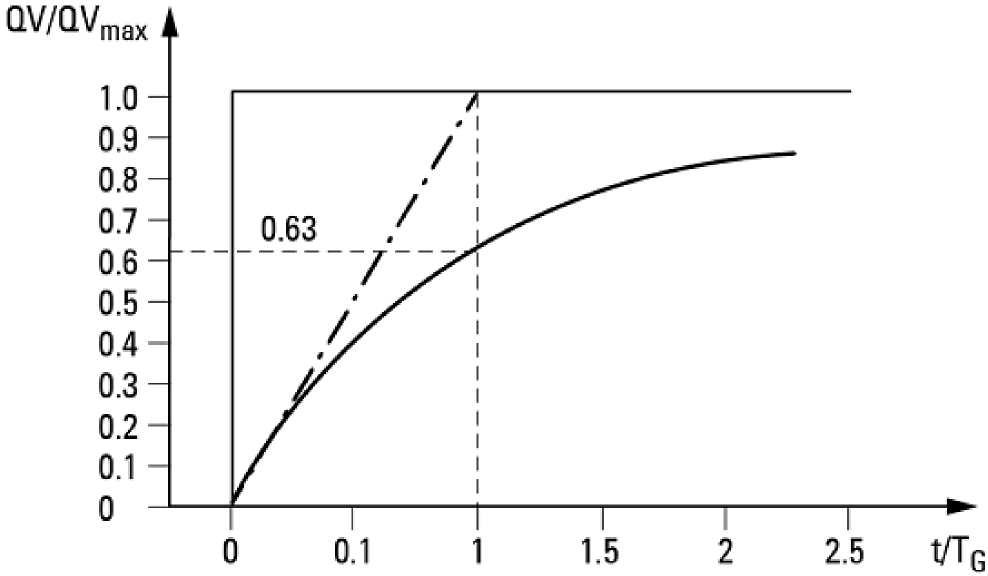



Figure 177: Response of the FT function block

——— Output value of the FT-PT1 signal smoothing filter function block, Tangent

— · — · Tangent

The output value is based on the following equation:

$$Y(t) = [T_A/T_G] \cdot [\ K_P \cdot (\ X(t)\text{-}Y(t\text{-}1)\ )\ ]$$

Y(t) = Calculated output value at time t
$T_A$ =Scan time (calculated internally)
$T_G$ = Recovery time
$K_P$ = Proportional gain
X(t) = Actual value at time t
Y(t-1) = Calculated output value at time t-1

### Scan time

Scan time $T_A$ depends on the set recovery time value.

| At recovery time $T_G$ | Internal calculation of the scan time $T_A$ |
|---|---|
| $T_G \leqq 1000$ ms | $T_A$= 10 ms |
| $T_G > 1000$ ms | $T_A$= $T_G$/100 |

### Cycle time to scan time

The ratio of cycle time $_{cyc}$ to sampling time $T_A$ should be such that the sampling time is much larger than the cycle time (by a factor of approximately 10): $T_A$=10 · $t_{cyc}$. The sample time is set indirectly with the value for recovery time $T_G$ (please refer to the table above).

The following therefore applies: $t_{cyc} << T_A$.

For applications in which this cannot be fulfilled, the cycle time should be set with the ST (set cycle time) function block so that the scan time is an integral multiple of the cycle time.

$$t_{cyc} \cdot n = T_A$$

mit n = 1,2,3, ...

The function block always works with a scan time that is an integral multiple of the cycle time. This can cause the set recovery time to be lengthened.

→ In the case of time-consuming application scenarios in which, for instance, a signal smoothing filter and a PID controller are used and visualization tasks need to be executed simultaneously, the cycle time may be extended to an extent that may not be tolerable for controller tasks. In these kinds of applications, move the time-consuming calculations to a second device connected via easyNet – please refer to

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |
| **(DWord)** | | |
| I1 | Input value | Value range: -32768…+32767 |
| TG | Recovery time TG [0.1 s] | Value range: 0…65535<br>The value 10 corresponds to a recovery time of 1000 ms. |
| KP | Proportional gain Kp [%]<br>Value range: 0 ... 65535 | Value range: 0…65535<br>The value 100 corresponds to a KP (factor) of 1.<br>The value 50 corresponds to a KP of 0.5. |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |

| Operators | Value inputs |
|---|---|
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET 2) | x |
| SN - Output bit via NET (send) 2) | x |
| N - Net marker bit 2) | x |
| nN - NET marker bit 2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| CY | Carry<br>1: If output value QV falls outside the valid value range. | Value range: -32768…+32767 |
| **(DWord)** | | |
| QV | Delayed output value | Value range: -32768…+32767 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

### Retention

The function block does not recognize retentive data.

### Example of an FT-PT1-signal smoothing filter when using the EDP programming language

```
M 40------------------------------[ FT01EN
```

Figure 178: Wiring the function block coils

### Example of an FT-PT1 signal smoothing filter configuration on a device display

When using the function block in the circuit diagram for the first time, use OK to automatically enter the general display of function block parameters, as shown in the figure on the left. Enter the function block settings here.

```
FT17 +
>I1
>TG
>KP
QV>
```

Figure 179: Parameters shown on display

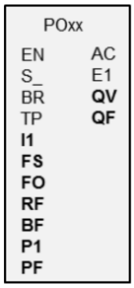
The display contains the following elements:

| | |
|---|---|
| FT17 signal smoothing filter | Function block: Signal smoothing, number 17 |
| + | Parameter set can be called via the PARAMETERS menu |
| >I1 | Input value: -32768… +32767 |
| >$T_G$ | Recovery time: 0… 65535 resolved in 100 ms<br>Example: The value 250 is processed in the function block as 25 s. |
| >$K_P$ | Proportional gain: 0… 65535 in %; Example: With KP=1500 the function block performs the calculation with $K_p$ =15 |
| QV> | Output value: -32768 … +32767, smoothed |

**See also**

→ Section "FT - PT1-Signal smoothing filter ", page 358
→ Section "TC - Three step controller", page 379
→ Section "VC - Value limitation ", page 384
→ Section "BC - Block compare", page 388
→ Section "BV - Boolean operation", page 444
→ Section "PO - Pulse output", page 364

### 6.1.4.3 PO - Pulse output




#### General

The DC versions of easyE4 base devices provide 2 pulse output function blocks, PO01 through PO02. These function blocks make it possibly to quickly output 24 V pulses in order to drive stepper motors.  Pulse output PO01 is hardwired to device output Q1 and PO02 to device output Q2.

→ If you use a PO function block with its hardwired device output Q1 or Q2, do not not assign that device output again in the program. Doing so will not have any effect, as a state change by the function block has a higher priority.

**WARNING**
**UNFORESEEABLE SWITCHING STATES AT OUTPUT**
When using the PO function block, strictly observe the separate assignment of the outputs if other hardware-dependent function blocks are used, such as the PW function block.

If this is not observed, unforeseeable switching states may occur at the output concerned.

### Operating principle

You can use the pulse output function block to generate a defined number of pulses at device output Q1 or Q2.. This series of pulses is called a pulse train. You can change the frequency within the pulse sequence. You can generate several pulse sequences in defined intervals. You can use these pulse trains to control a stepper motor in the three possible individual sequences: acceleration, operation, and deceleration. The function block also has Jog mode as well as normal mode.

Each function block is assigned a set device output for the fast pulses:

Function block PO01: -> device output Q01

Function block PO02: -> device output Q02

The device outputs Q1 and Q2 used must not be processed again in the circuit diagram. The reason for this is that the PO function blocks overwrite all other status changes at the device outputs Q01 and Q02.
A suitable power output stage that is compatible with the stepper motor being used is required in order to be able to drive the stepper motor.

The step information needs to be fed to the power output stage's input logic. The input logic for both signals should be optocoupled and process an input voltage of +24V.

The parameter definition for a stepper motor and therefore the function block largely depends on the rated load to be moved. This defines the framework for the maximum start and operating frequency.

The function block is active if the PO..EN coil is actuated. After you have parameterized the function block, you can actuate the coil PO..S_. This will start normal operation. Alternatively, you can also actuate the coil PO..TP and start the function block in Jog mode.

### Pulse profiles

The PO function block enables very simple pulse profiles to be generated in order to control a stepper motor with the sequences acceleration [1], operation [2] and braking [3]. In order to do this, a PO function block supplies a user-defined number of square wave pulses (50% relative ON duration) at the permanently assigned high-speed device output Q1 or Q2. I1 for normal operation or P1 for inching operation.

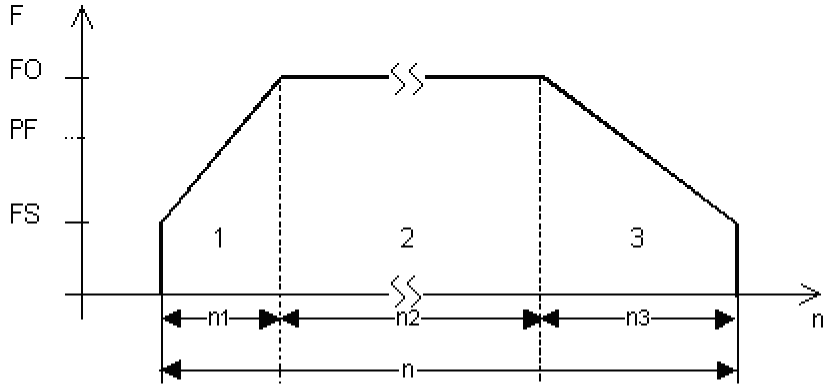



Figure 180: Typical pulse profile for a stepper motor in normal mode

n1: Number of acceleration pulses

n: Total number of pulses

n2: Number of pulses for operation

n3: Number of deceleration pulses

QF: Current frequency

FS: Start frequency

FO: Operating frequency

PF: Jog frequency

### Start frequency at function block input FS

The maximum start frequency that can be configured depends on the load torque. Enter a start frequency value that will enable the stepper motor to move the load even at low speeds. You will normally be able to find information on the maximum start frequency without taking the load torque into account in the technical data for the motor. When the load torque is taken into account, the start frequency must only be high enough for the motor not to lose any pulses during acceleration and not to be driven by the load during deceleration.

→ If the value of FS is set too low, this may cause oscillations between the motor and the load. Jumps at the start or end of the positioning section may occur if FS is set too high.

### Operating frequency at function block input FO

The maximum operating frequency also depends on the load torque.
The motor generally achieves its maximum power and torque at very low speeds. The higher the speed, the weaker the motor will be.

### Jog frequency at function block input PF

The maximum frequency that the motor should be able to reach in jog mode.

### Number of steps at function block input P1

The number of steps that the motor should be able to execute in jog mode.

### Frequency change per step in the acceleration phase RF

During the acceleration phase [1], the motor's step sequence frequency is increased continuously from the start frequency to the operating frequency.
The frequency change per step is used to define the number of steps that the acceleration phase will take with the configured starting and operating frequencies.

### Frequency change per step in the braking phase RF

During the deceleration phase [3], the motor's step sequence frequency is decreased continuously from the operating frequency to the start frequency.
The frequency change per step is used to define the number of steps that the deceleration phase will take with the configured starting and operating frequencies.

### Number of pulses (total number of pulses) I1

The total number of pulses needs to be configured based on the distance being covered on the basis of the specified step angle per step.

→ In normal operation the function block will always move a distance specified by the Total number of pulses.

This value for the total number of pulses and the calculated number of pulses for the acceleration and braking sequence are used by the function block to calculate the number of pulses for the operating sequence [2].

### Number of pulses for acceleration and braking

The PO function block automatically calculates the number of pulses required for acceleration and braking using the frequency change values FS->FO and FO->FS you have set.
You can calculate the number of pulses for the acceleration and braking sequence using the following equations.

$$n_{RRF} = \frac{(FO - FS)}{RF} * 1000 \qquad n_{RBF} = \frac{(FO - FS)}{BF} * 1000$$

FO: Operating frequency [Hz]; FS: Start frequency [Hz],

$n_{RRF}$ : Number of pulses in the acceleration sequence

$n_{RBF}$ : Number of pulses in the braking sequence

RF: Frequency change in the acceleration phase [mHz/step]

BF: Frequency change in the braking phase [mHz/step]

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| EN | Enable of the function block on status 1.<br>The operations **Start positioning job** (S_) or **Jog mode** (TP) can be executed when the function block is enabled.<br>Disable the function block on status 0.<br>The function block **Reset** is carried out when there is falling edge from 1 to 0. | **Caution!** In normal mode, always stop positioning jobs with the BR function block input. In this case, the step sequence frequency will be reduced according to the deceleration ramp and the motor will decelerate smoothly.<br>A stop with EN= 0 would cause an abrupt stop of the motor and a possible loss of the reference point if this would be dragged further by the moved load. |
| S_ | The positioning job will start when there is a rising edge.<br>An active positioning job will be signaled with AC =1. | When a positioning job is activated, the acceleration, operating and braking phases are executed in succession.<br>If a positioning job is already activated, a renewed rising edge 0 -> 1 at S_ will not cause a new job to be started. |
| BR | Aborts the ongoing positioning job when there is a rising edge. | After the positioning job is aborted, the function block now runs the braking phase, i.e. a delayed motor stop is executed.<br>The bit output AC is only set to 0 when the braking phase has been completed. |
| TP | Activates the Jog mode on status 1<br>The on duration TP = 1 determines the type of Jog mode. | Two operating mode are available in Jog mode for diagnostics and testing.<br>**1. Positioning with a preset number of steps**<br>TP on duration ≦ 0.5 seconds<br>The motor moves by the number of steps defined at P1.<br>**2. Positioning with a specified jog frequency - manual mode**<br>TP on duration > 0.5 seconds<br>The motor will be accelerated to the jog frequency set at PF. |
| **(DWord)** | | |
| I1 | Number of pulses | For number of pulses enter the total number pulses for the entire sequence, consisting of the three individual acceleration, operating and braking phases.<br>Integer value range:<br>0…+2 147 483 647 |
| FS | Start Frequency | Integer value range:<br>0…5000 Hz |
| FO | Operating frequency | Integer value range:<br>0…5000 Hz |

| | Description | Note |
|---|---|---|
| RF | Frequency change in the acceleration phase [mHz/step] | Integer value range:<br>0…65 535<br>Value for changing the frequency during the acceleration in 0.001Hz per step.<br>Example: f-REF = 0 - 10 V<br>0 = No frequency change<br>100 = Frequency increased by 0.1 Hz per step |
| BF | Frequency change in the braking phase [mHz/step] | Integer value range:<br>0…65 535<br>Value for changing the frequency during braking in 0.001Hz per step.<br>Example: f-REF = 0 - 10 V<br>0 = No frequency change<br>1000 = Frequency reduction by 1 Hz per step |
| P1 | Number of steps in jog mode | If you only define a very low number of steps, the start pulse at the TP function block input must likewise only be very short. Otherwise the function block will output several pulse sequences that will cause the distance A to be covered several times. In extreme cases generate short start pulses at TP using a T... timing relay.<br>Integer value range:<br>0…65 535 |
| PF | Jog frequency | Integer value range:<br>0…5000 Hz |

### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |

1) Only on function blocks T, AC
2) Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| AC | 1: Signals an ongoing positioning job<br>Pulses are output at device output Q1 or Q2. also during jog mode or during the braking ramp and also after S_ is set to 0.<br><br>0: Signals that there is no ongoing positioning job. | |
| E1 | Error output<br>1: in the event of incorrect parameters, such as:<br>- FO < FS (operating frequency < start frequency)<br>- PF < FS (jog frequency < start frequency) | No positioning commands if the function block detects incorrect parameters.<br>If the function block detects an incorrect parameter change during an active positioning job, the step sequence frequency is reduced according to the braking ramp and the motor is gently braked to a stop. |
| **(DWord)** | | |
| QV | Actual number of steps completed | Integer value range:<br>0…+2 147 483 647 |
| QF | Actual output frequency | Integer value range:<br>0…5000 Hz |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers [2)]<br>NET stations n | x |

| Assigning operands | Value outputs |
|---|---|
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)[2)] SN - Output bit via NET (send) | x |
| N - Network marker bit[2)] | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| Parameter set | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation not possible | | |

### Other

### Retention

The function block does not recognize retentive data.

### Normal mode with signal diagram

For normal mode, specify the number of pulses based on the distance that must be traversed.
In addition, configure the start frequency and the operating frequency as a function of the load torque and of the motor being used.
Set the slope for the starting and braking ramps by using the corresponding RF and BF frequency change inputs. The function block will interpret the frequency change parameter value as a change in mHz per step. For example RF = 2000 means that the frequency will increase 2 Hz per step during the acceleration phase.

### Parameters for normal operation

▶ Configure the following function block inputs as shown:
- I1 - Number of pulses; e.g. 10000 (value range 0...2147483647)
- FS Start frequency; e.g. 200 Hz (value range 0-5000Hz)
- FO - Operating frequency; e.g. 3000 Hz (value range 0-5000Hz)
- RF - Frequency change per step in the acceleration phase; e.g., 500 mHz/step, i.e., the frequency is increased 0.5 Hz per step (value range of 0 to 65535)
- BF - Frequency change per step in the deceleration phase; e.g., 2000 mHz/step, i.e., the frequency is decreased 2 Hz per step (value range of 0 to 65535)

▶ Connect function block inputs EN, S_, and BR to a contact suitable for driving them. Adding a comment for the selected operands can make the program easier to understand.

▶ Switch input EN=1.

▶ Start a positioning job with a rising edge at bit input S_.

▶ Check the acceptance of the job at device output AC.

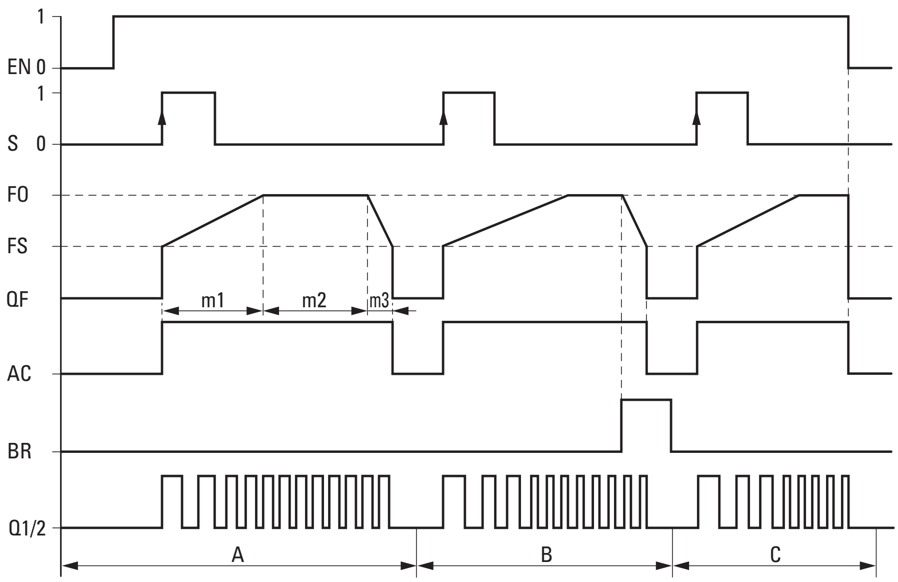


Figure 181: Signal diagram for PO pulse output with specified number of pulses for I1 - possible normal mode phases

EN: Enable coil

S: Start coil for pulse sequence

FO: Operating frequency,

FS: start frequency,

QF: Current output frequency

m1 = Acceleration phase, m2 = Operating phase, m3 = Braking phase

AC: Positioning job active

BR: Stop coil for pulse sequence

Q1/2: Pulse sequence at the device output Q1 and/or Q2

• Range A: The pulse sequence is present at the device output until the number of pulses defined at I1 has been reached.

• Range B: Activating the coil PO..BR initiates the braking phase and reduces the frequency of the pulse sequence.

• Range C: A voltage drop at coil PO..EN immediately switches off the pulse sequence from the device output.

### Jog mode with signal diagram

You can use the PO function block in jog mode for commissioning. You can either start a positioning job with a specified number of steps P1 or a specified jog frequency PF. The decisive factor will be whether the on time for TP is TP ≤ 0.5 seconds or TP > 0.5 seconds.

### Parameters for jog operation

- ▶ Configure the following function block inputs:
  - FS start frequency; e.g., 200 Hz (value range of 0 to 5000 Hz); the operating frequency is not needed for operation, but for the plausibility check instead.
  - RF - Frequency change per step in the acceleration phase; e.g., 500 mHz/step, i.e., the frequency is increased 0.5 Hz per step (value range of 0 to 65535)
  - BF - Frequency change per step in the deceleration phase; e.g., 2000 mHz/step, i.e., the frequency is decreased 2 Hz per step (value range of 0 to 65535)
- ▶ To move at the specified jog frequency at function block input PF (e.g., 1000 Hz (o to 5000 Hz), configure the maximum frequency that the motor should reach in jog mode.
  The jog frequency must be greater than the start frequency: PF > FS; e.g., 1000 Hz.
- ▶ To run at the specified number of steps, configure the maximum number of steps that the motor should carry out in jog mode at function block input P1.
- ▶ Connect inputs EN and TP each to a contact suitable for activation.
- ▶ Check the acceptance of the job at device output AC.

Jog mode will be executed as described below as a function of the duty cycle at function block input TP.

### Positioning with a specified number of steps P1 (defined distance) TP on time ≦ 0.5 seconds

When using this operating mode, set the distance using the number of steps P1.

▶ Switch the TP input to 1 for a duration ≦ 0.5 seconds.

The motor will start with starting frequency FS, move the set number of steps, and then stop automatically. The start ramp and deceleration ramp will be ignored in this case.

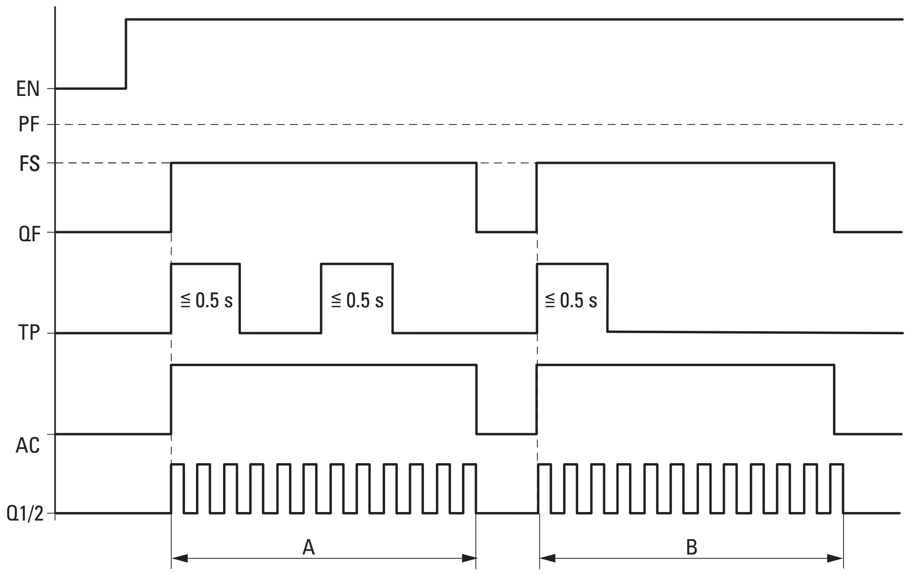



Figure 182: Signal diagram for jog mode with specified number of steps P1

PF: Jog frequency

FS: Start frequency

QF: Current output frequency

TP: Jog command

AC: Positioning job active

A: Pulse output until number P1 reached, triggered by TP if on duration ≦ 0.5 sec.

**Positioning with specified jog frequency PF (defined maximum frequency) TP on time > 0.5 seconds**

When using this operating mode, you can control the distance manually by keeping the state at function block input TP at "1" for a time > 0.5 seconds.

- Switch the TP input to 1 for a duration > 0.5 seconds.

The motor starts moving for the duration of 0.5 s with start frequency FS and is then accelerated to jog frequency PF with frequency change RF.

- End jog mode with TP = 0.

**Number of steps in jog mode P1 reached**

If the number of steps P1 is reached after the deceleration phase ends, device output Q1/2 will be switched off.

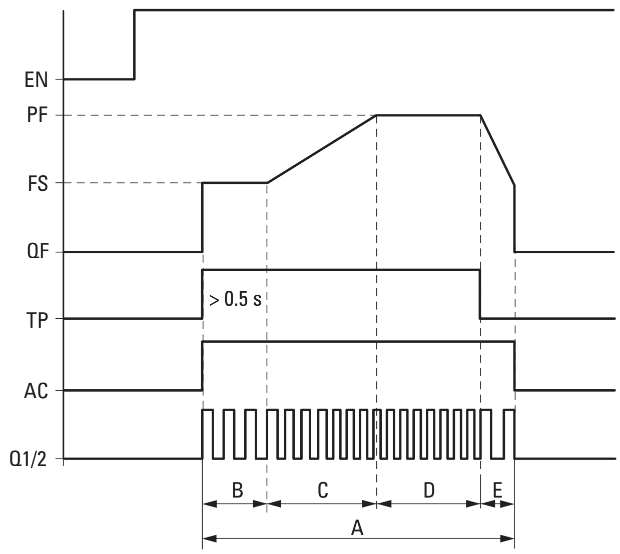



Figure 183: Signal diagram for jog mode with specified jog frequency, P1 after deceleration phase reached

PF: Jog frequency

FS: Start frequency

QF: Current output frequency

TP: Jog command

AC: Positioning job active

A: Complete distance if active TP on time is longer than 0.5 sec.

B: During the first 0.5 sec, the distance is traversed with specified start frequency FS.

C: This is followed by the acceleration phase with RF all the way to the jog frequency.

D: Motion continues with jog frequency PF.

E: The deceleration phase is initiated with jog command TP = 0 and the frequency of the pulse sequence is reduced to the start frequency with BF.

### Number of steps in jog mode P1 not reached:

If the number of steps P1 has not been reached after the deceleration phase ends, the motor will be driven with start frequency FS until the specified number of steps is reached. Device output Q1/2 will not be switched off until then.

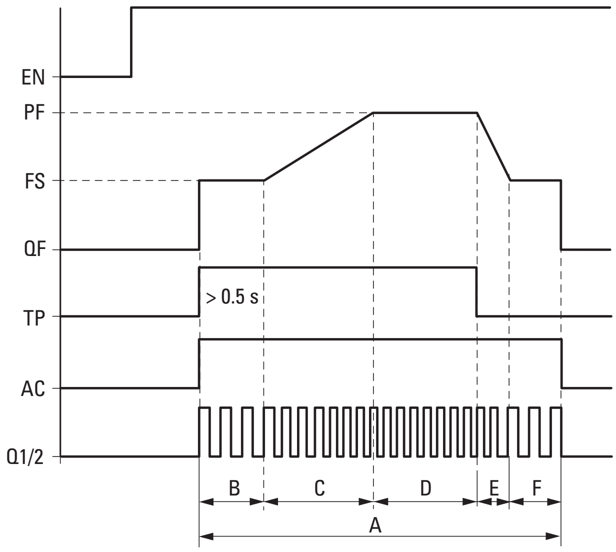



Figure 184: Signal diagram for jog mode with specified jog frequency, P1 not reached after deceleration phase

PF: Jog frequency

FS: Start frequency

QF: Current output frequency

TP: Jog command

AC: Positioning job active

A: Complete distance if active TP on time is longer than 0.5 sec.

B: During the first 0.5 sec, the distance is traversed with specified start frequency FS.

C: This is followed by the acceleration phase with RF all the way to the jog frequency.

D: Motion continues with jog frequency PF.

E: The deceleration phase is initiated with jog command TP = 0 and the frequency of the pulse sequence is reduced to the start frequency with BF.

F: Distance after deceleration phase until the specified number of steps P1 is reached with start frequency FS.

### How the run-on distance (phase F) is determined

P1 and the on time for jog mode TP=1 need to be added to the sample parameters defined in "Parameters for jog operation."

FS = start frequency = 200 Hz

PF = 1000 Hz

RF = Acceleration frequency change = 500 mHz/step

BF = Frequency change in the braking phase = 2000 mHz/step

P1 = Number of steps in jog mode = 6000
TP=1 on duration = 3 seconds

The sample parameters yield the following distance:

A: Complete distance = P1 = Number of steps in jog mode
B: Start phase with FS for 0.5 seconds = 100 steps
C: Acceleration phase with RF= 0.5 Hz/step to reach PF-FS= 800 Hz = 1600 steps
D: Jog frequency = 1000 Hz with an assumed on time of 3 seconds for TP=1 = 3000 steps
E: Deceleration phase with BF = 2 Hz/step to reach PF-FS= 800 Hz = 400 steps
F: P1-(B+C+D+E) = 6000 - 5100 = 900 steps

The run-on distance (phase F) is 900 steps.

#### Connecting a pulse output function block

Prerequisites

- A control relay with 24 VDC must be selected for the project.

#### Evaluation of a pulse output contact

You can use bit outputs AC (positioning job in progress) and E1 (error) to check whether a positioning or jog mode job has been activated. You can use error output E1 to check whether your parameters are correct.

#### Resetting a pulse output function block

▶ To reset (Reset) the pulse output function block, switch the EN bit input from 1 to 0

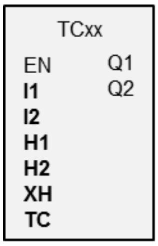
#### See also

→ Section "PO - Pulse output", page 364
→ Section "FT - PT1-Signal smoothing filter ", page 358
→ Section "TC - Three step controller", page 379
→ Section "VC - Value limitation ", page 384
→ Section "BC - Block compare", page 388
→ Section "BV - Boolean operation", page 444
→ Section "PO - Pulse output", page 364

### 6.1.4.4 TC - Three step controller

#### General

easyE4 base devices provide 32 three-step controller function blocks, TC01 through TC32.
TC three-step controllers feature three states for the manipulated variable. These states are implemented with two function block outputs Q1, Q2, of which either none or only one can be closed. I1 is the setpoint and I2 is the actual value. X = I1 - I2 yields control deviation X, which is applied on the actual controller. The controller will then determine the manipulated variable for function block outputs Q1, Q2.



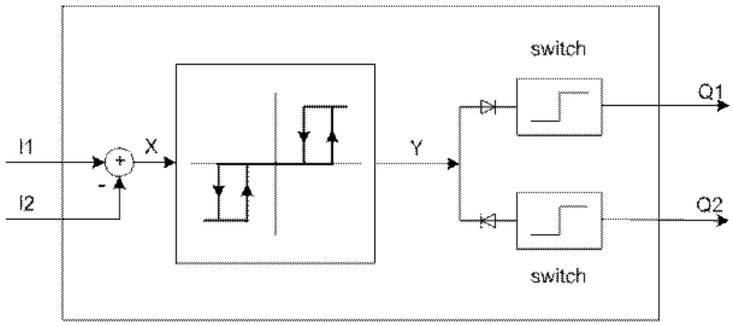




Figure 185: Three-step controller schematic diagram

I1: Setpoint

I2: Actual value

#### Operating principle

The following timing diagram shows how the three-step controller behaves.

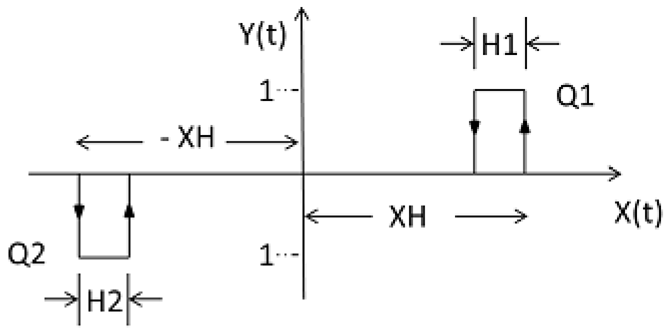



Figure 186: Timing diagram for three-step controller

XH/ -XH: Distance X from switching point

H1: Hysteresis 1 for XH

H2: Hysteresis 2 for -XH

Y(t): Operating points for Q1/ Q2

Q1: Switch output X = positive

Q2: Switch output X = negative

### Operating ranges

- $X > XH$
  Q1 switches one to $X < (XH - H1)$
- $X < -XH$
  Q2 switches one to $X > -XH + H2$

If the switching conditions for Q1 and Q2 are not met, both outputs will be switched off to 0.

Only Q1, or Q2, or neither one can be switched on at any one time.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |
| **(DWord)** | | |
| I1 | Setpoint | -32768…+32767 |
| I2 | Actual value | -32768…+32767 |
| H1 Chaser light on times | Hysteresis value 1 | 0…32767 |
| H2 Chaser light on times | Hysteresis value 2 | 0…32767 |
| XH | Distance from switching point | 0…32767<br>Contact distance |
| TC | Cycle time | 0…65535<br>In 0.1 ms; value of 10 = 1 s. If the value = 0, the function block will go through every cycle. |

### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Numeric inputs |
|---|---|
| | I1, I2, H1, H2, XH, TC |
| Constant | x |
| Markers MD, MW, MB | x |
| Analog inputs IA | x |
| Analog output QA | x |
| Numeric output from a different FB | x |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| | EN |
| Constant 0, constant 1 | x |
| M – Markers<br>RN - Input bit via NET<br>SN _ Output bit via NET (send)<br>N - Network marker bit<br>nN - NET station n marker<br>ID: Diagnostic alarm<br>LE - Output backlight | x<br>x<br>x<br>x<br>x<br>x<br>x |
| I Bit input | x |
| Q Bit output from another FB | x |

### Function block outputs

| | Description |
|---|---|
| **(Bit)** | |
| Q1 | Switch output 1 |
| Q2 | Switch output 2 |

### Assigning operands

You can assign the following operands to the function block outputs that are bit outputs:

| Operators | Bit outputs |
|---|---|
| | Q1, Q2 |
| Constant 0, constant 1 | x |
| M – Markers<br>RN - Input bit via NET<br>SN _ Output bit via NET (send)<br>N - Network marker bit<br>nN<br>ID: Diagnostic alarm<br>LE - display brightness indicators | x<br>x<br>x<br>x<br>x<br>x<br>x |
| I Bit input | x |
| Q Bit output from another FB | x |

#### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display + Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

#### Other

#### Retention

The function block does not recognize retentive data.

#### Heating and cooling

Actual value I1 (temperature) is higher than setpoint I2, meaning that cooling is required.

Function block output Q1 =1 switches the cooling system on as soon as (I1-I2) > XH.

Actual value I1 (temperature) is lower than setpoint I2, meaning that heating is required.

Function block output Q2 = 1 switches the heating system on as soon as (I1-I2) > XH

Hysteresis values H1 and H2 determine how long cooling or heating is required and, accordingly, the cooling/heating energy content.

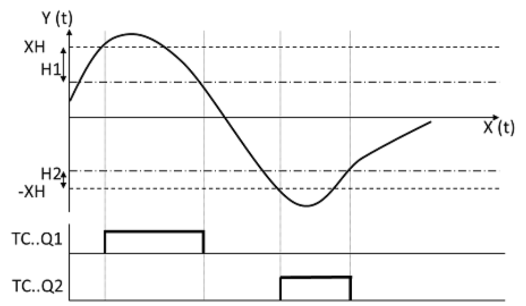



Figure 187: Signal diagram for three-step controller

**Level control**

The contents of a liquid tank must not fall below or exceed a specific liquid level.

The ACTUAL value (level) is higher than the SETPOINT, meaning that liquid needs to be drained. Function block output Q1 switches the drain valve on.

The ACTUAL value (level) is lower than the SETPOINT, meaning that liquid needs to be replenished. Function block output Q2 switches the supply valve on.

Hysteresis values H1 and H2 define how long liquid needs to be drained or replenished. This also means that they define the volume before draining and after replenishing.

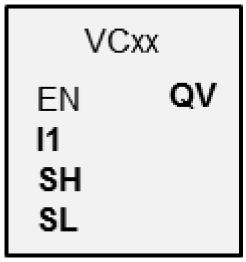
**See also**

→ Section "TC - Three step controller", page 379
→ Section "FT - PT1-Signal smoothing filter ", page 358
→ Section "VC - Value limitation ", page 384
→ Section "BC - Block compare", page 388
→ Section "BV - Boolean operation", page 444
→ Section "PO - Pulse output", page 364

### 6.1.4.5 VC - Value limitation

#### General

easyE4 base devices provide 32 value limitation function blocks, VC01 through VC32.
These function blocks can be used to output values within specific limits.




#### Operating principle

The lower and upper limits are set using function block inputs SL (Low) and SH (High). The value at function block output QV will follow the value at function block input I1 as long as the latter falls within the limits. Values outside of the range will be truncated accordingly.

EN = 0 will carry out a reset and function block output QV will be set to a value of 0.

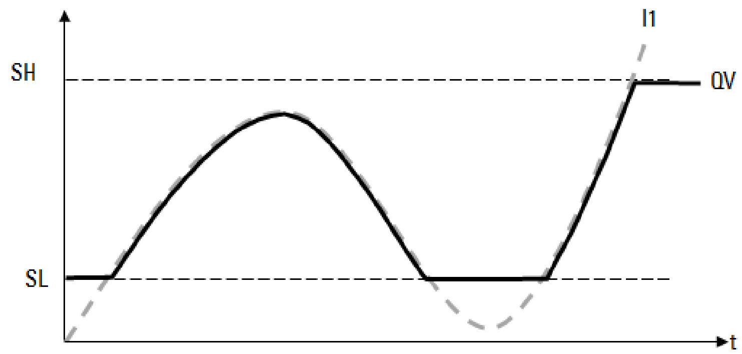



Figure 188: Figure: Restriction of the input values to the specified limits.

| | |
|---|---|
| SL: Lower limit | I1: Input function at I1 |
| SH. Upper limit | QV: Bounded output function at QV |

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | |
| I1 | Input value | Integer value range: -2,147,483,648 to +2,147,483,647 |
| SH | Upper Threshold Value | |
| SL | Lower threshold value | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)] NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

#### Function block outputs

| | Description | Note |
|---|---|---|
| **(DWord)** | | |
| QV | Outputs the value at input I1 within the set limits. | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

#### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

#### Parameter set

| Parameter set | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

#### Other

#### Retention

The function block does not recognize retentive data.

#### See also

→ Section "DC - PID controller", page 351
→ Section "FT - PT1-Signal smoothing filter ", page 358
→ Section "TC - Three step controller", page 379
→ Section "BC - Block compare", page 388
→ Section "BV - Boolean operation", page 444
→ Section "PO - Pulse output", page 364

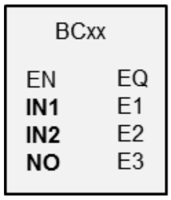
### 6.1.5 Data and register function blocks

#### 6.1.5.1 BC - Block compare

The data block comparator (BC = Block Compare) function block compares two contiguous marker ranges. For this you define the number of bytes to be compared. The comparison is carried out in byte format for marker types MB, MW and MD.

**General**

easyE4 Base devices provide 32 block comparison function blocks,
BC01 through BC32. This function block compares values from two contiguous marker ranges. Moreover, this comparison can be carried out within the entire marker range (1024 bytes). Addresses are byte-based, including the range that can only be addressed with words or double words (MB513 through MB1024). (MB513 through MB1024), refer to section → Section "Organizing marker ranges", page 202.




**Operating principle**

The reference data block starts at the source address specified at input I1. This data block will be compared with the data block that starts at the destination address specified at I2. You can use constants or operands, in which case the data value of the operand at runtime will be used as the corresponding address.

The NO input is used to specify the size of the data block (number of elements) in bytes.

If the comparison of two data blocks finds that there is no difference between them, the Boolean output EQ will be set to 1.

The following operands can be used:

- Constant NU
- ACTUAL value ..QV.. of a function block
- Analog input IA.. or analog output QA..
- Timer constant

**Example value 0**

A value <0> at input I1 means that the reference data block for the comparison starts at MB01. A value of <100> at I2 means that the target data block for the compare operation begins at MB101.

**Marker byte example**

You wish to compare the content of marker bytes MB11-MB14 with the content of marker bytes MB381-MB384 (MD96). A value <10> at input I1 means that the reference data block for the comparison starts at MB11. A value <380> at I2 means that the destination data block for the comparison starts at MB381.

→ Marker addresses are always specified using byte addresses.

→ easySoft 7 no longer supports addresses without an offset.

#### Update

After importing projects created with earlier versions of the programming software easySoft, check whether the "without offset" address type was used. If it was, you will need to reprogram the relevant parameters and replace the marker operands with constants.

#### Offset calculation for addressing marker words

Offset = MW (x-1)*2

#### Offset calculation for addressing marker double words

Offset = MD (x-1)*4

#### Parameter error due to incorrect number or offset definition

Bad parameter configurations will be output at program runtime via error outputs E1 through E3.
This type of parameter configuration error will occur, for example, when the number of elements exceeds the source or destination range or, due to an offset error, the source or destination range falls outside the available marker range.

#### Application example

Comparing marker data blocks

| | |
|---|---|
| I1 | MB23 |
| I2 | MB30 |
| NO | NU 4 |

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | |
| I1 | Source range | First marker address (MB, MW, or MD) of the reference data block or offset added to marker byte MB01 when using one of the operands specified above. |
| I2 | Destination range | First marker address (MB, MW or MD) of the destination data block or offset to marker byte MB01 when defining one of the operands stated in the table. |
| NO | Number of elements to be compared: max. 192 bytes | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2) NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |

| Operators | Bit inputs |
|---|---|
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EQ | 1: if the data ranges are identical.<br>0: if the data ranges are not identical. | |
| E1 | Error output<br>1: if the number of elements exceeds the source or target range. | The range limits are checked irrespective of the edge change on the Boolean EN input. |
| E2 | Error output<br>1: if the source and target range overlap. | The range limits are checked irrespective of the edge change on the Boolean EN input. |
| E3 | Error output<br>1: if the source or destination range are outside of the available marker range (offset error, or input NO is not configured i.e. has the value 0. | The range limits are checked irrespective of the edge change on the Boolean EN input. |
| EQ | Equal<br>1: if the data ranges are identical.<br>0: if the data ranges are not identical. | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |

| Assigning operands | Bit outputs |
|---|---|
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

**Parameter set**

| Parameter set | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

#### Other

#### Retention

The function block does not recognize retentive data.

#### Example of a analog value comparator function block when using the EDP programming language

```
I 05----------------------------Ɛ BC11EN
```

Figure 189: Wiring the enable coil

```
BC11E1┐
BC11E2┤
BC11E3┘-----------------------Ɛ M 48
BC11EQ-BC11EN-----------------Ɛ M 49
```

Figure 190: Wiring the contacts

#### Example of an AR configuration on a device display

When using the function block in the circuit diagram for the first time, use OK to automatically enter the display of function blocks on the device display, as shown in the following figure.

```
BC11 +
>I1
>I2
>NO
```

Figure 191: Parameters on the display

Enter the function block settings here. The display contains the following elements:

| | |
|---|---|
| BC11 block compare | Function block: Data block comparator, number 11 |
| + | Parameter set can be called via the PARAMETERS menu |
| >I1 | Start of comparison range 1; the data block with the start address present at input I1 is compared with the data block with the start address present at input I2. |
| >I2 | Start of comparison range 2 |
| >NO | Number of elements to be compared in bytes per range, number: 1 - 383 |

**See also**

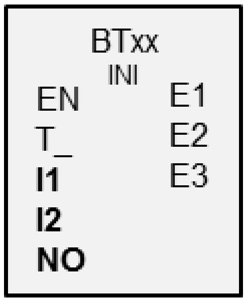
→ Section "BC - Block compare", page 388
→ Section "FT - PT1-Signal smoothing filter ", page 358
→ Section "TC - Three step controller", page 379
→ Section "VC - Value limitation ", page 384
→ Section "BV - Boolean operation", page 444
→ Section "PO - Pulse output", page 364
→ Section "RE - Recipe records ", page 410

### 6.1.5.2 BT - Block transfer

The block transfer function block is used to transfer values from one marker range to another (copy data). The marker ranges can be overwritten with a particular value (data initialization). The following marker types can be transferred and overwritten: MB, MW and MD.

#### General

easyE4 base devices provide 32 Block Transfer function blocks BT01...BT32.
The block transfer function copies values from one marker range to a different, contiguous market range when the CPY (copy) operating mode is enabled. The source range and target range are allowed to overlap.
When the INI (initialization) mode is enabled instead, the function block copies the content of a marker byte to a different, continuous marker range.
The transfer is carried out bytewise.




#### Operating principle

Data is transferred from the source address specified at function block input I1 to the destination address specified at function block input I2. The NO input is used to specify the size of the data block in bytes.

#### Transfer with offset

The source address for the copy or initialization is specified at function block input I1, while the destination address is specified at function block input I2. Within this context, the numeric value of the operand at runtime will be interpreted as the offset to be added to marker byte MB01.

**Example with a value of "0"**

A value 0 at input I1 means that the source data block for the transfer starts at MB01. A value 10 at I2 means that the destination data block for the transfer starts at MB11.

→ With the offset information you can address marker ranges (for example MB380), which you can not address when using marker operands (direct addressing).

#### Marker byte example

You wish to transfer the content of marker bytes MB1-MB4 with the content of marker bytes MB381-MB384 (MD96). A value 0 at input I1 means that the source data block for the transfer starts at MB01. A value 380 at I2 means that the destination address for the transfer starts at MB381.

#### Offset calculation for addressing marker words

Offset = MW (x-1)*2

#### Offset calculation for addressing marker double words

Offset = MD (x-1)*4

#### Parameter error due to incorrect number or offset definition

Incorrect parameters are indicated whilst the program is running via the E1 - E3 error outputs.
Parameter errors of this kind occur, for example, if the number of elements to be transferred exceeds the source or destination range, or when, due to an offset error, the source and destination range are outside of the available marker range.

→ Transfer function blocks always copy or initialize marker bytes, and never marker words or double words. This transfer behavior does not depend on the values at I1 and I2 (source address and destination address). You can copy a marker double word, e.g., MD 12 to MD 96, by copying 4 marker bytes with the function block.

#### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| (Bit) | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| T_ | Triggering the transfer on a rising edge | |
| (DWord) | | |
| I1 | Source address | Offset to marker byte MB01 when using for the definition one of the aforementioned operands |
| I2 | Destination address | Offset to marker byte MB01 when using for the definition one of the operands stated in the table |
| NO | Number of elements to be initialized or copied. | Integer value range<br>Operating mode INI: 1…+1024 Byte<br>Operating mode CPY: 1…+1024 Byte |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating modes

| | Description | Note |
|---|---|---|
| INI | Initialize | Initializes the destination range with a byte value that is stored at the source address. The length of the source range is fixed at one byte. NO defines the length of the destination range. |
| CPY | Copy | Copies a data block from a source to a destination range. NO defines the size of the data block to be copied. |

### Copy mode, operating mode = CPY

In Copy mode, the function block copies the complete data range of the size specified at NO from the source range to the destination range. You specify the start of the source range and the destination range via I1 (source address) and I2 (destination address).

**Example transfer with offset**

Copying a marker data block (2 bytes) with a variable offset definition for the marker ranges.
The content of the marker bytes MB14 + MB15 are to be copied with a variable offset that is defined via the output QV of counter relay C3.

| | |
|---|---|
| I1 | NU14 |
| I2 | C 3 |
| NO | NU 2 |

**Initialization mode, operating mode = INI**

In initialization mode, the function block takes a byte value that is stored at the source address (input I1) and copies it to a destination range. The destination range is specified at input I2, and the length is defined by the value at input NO. The exact same source byte value (MB) is then copied to every single byte in the destination range.

→ If you select a type MD or MW marker operand as the source address, the function block always uses the content of the least significant byte for the initialization. If, for example, you assign MD 6 to I1, the function block initializes with the content of marker byte MB21.

**Function block outputs**

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| E1 | Error output<br>1: if the number of elements exceeds the source or destination range. | The range limits are checked irrespective of the edge change on the Boolean T_ input. No data blocks are initialized or copied if an error occurs. |
| E2 | Cannot be evaluated<br>Originally used as an error output in previous versions and kept due to compatibility reasons. | The source and target ranges are allowed to overlap for a copy operation; no error message will be generated at E2. |
| E3 | Error output<br>1: if the source or destination range are outside of the available marker range (offset error, or input NO is not configured i.e. has the value 0. | The range limits are checked irrespective of the edge change on the Boolean T_ input. No data blocks are initialized or copied if an error occurs. |

**Assigning operands**

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |

| Assigning operands | Value outputs |
|---|---|
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)[2] SN - Output bit via NET (send) | x |
| N - Network marker bit[2] | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

**Parameter set**

| | Description | Note |
|---|---|---|
| **Parameter set** | | |
| Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display + Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

**Other**

**Retention**

The function block does not recognize retentive data.

**Example of a configuration for a BT block transfer function block on the device display**

When using the function block in the circuit diagram for the first time, use OK to automatically enter the display of function blocks on the device display, as shown in the following figure.

```
BT07 INI +
>I1
>I2
>NO
```

Figure 192: Parameters on the display

Enter the function block settings here. The display contains the following elements:

| | |
|---|---|
| BT07 block transfer | Function block: block transfer, number 07 |
| INI | Operating mode: INI - Initialize |
| + | Parameter set can be called via the PARAMETERS menu |
| >I1 | Start address of source range or initialization markers (MB,MW,MD) |
| >I2 | Start address destination range |
| >NO | Number of elements to be written in bytes per range, number: 1…383 |

#### Example of a block transfer function block when using the EDP programming language

The trigger coil is connected to a device input.

```
I 05-----------------------------[ BT07T_
```

Figure 193: Wiring the trigger coil

```
BT07E1┐
BT07E2┤
BT07E3--------------------------[ M 42
```

Figure 194: Wiring the contacts

The messages of the function block are sent as a group message to a marker M42.

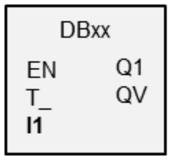
**See also**

→ Section "BC - Block compare", page 388
→ Section "BT - Block transfer", page 395
→ Section "MX - Data multiplexer", page 406
→ Section "SR - Shift register", page 415
→ Section "TB - Table function", page 423
→ Section "RE - Recipe records ", page 410
→ Section "Retention function", page 578
→ Section "Organizing marker ranges", page 202

### 6.1.5.3 DB - Data function block

#### General

easyE4 base devices provide 32 data function blocks DB01…DB32.
This function block makes it possible to copy bytes, words, or double words to an operand for only one cycle.




#### Operating principle

When there is a rising edge at function block input T_, the value at function block input I1 will be passed to an operand connected to function block output QV. The operand will keep that value until the next time it is overwritten,
meaning it can be used, for example, to save reference values for function blocks.

→ Note that the data function block only transfers the value in the program cycle in which it detects a rising edge. When the operand linked with output QV is overwritten by the program after the value is transferred, the value transferred with the data function block is lost.

#### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| T_ | A rising edge at this input will transfer the value of function block input I1 to the operand connected to QV. | |
| **(DWord)** | | |
| I1 | Value that is transferred to output QV when the function block is triggered. | Integer value range: -2,147,483,648 to +2,147,483,647 |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)] | x |

| Operators | Value inputs |
|---|---|
| NET station n | |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: transfer confirmation, if function block output T_ is 1. | |
| **(DWord)** | | |
| QV | Passes the value at function block input I1 to the operand connected to QV during the program cycle in which a rising edge is detected at T_. | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| **Parameter set** | | |
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| ☑ No edge evaluation of T bit input | | |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

### Retention

Data blocks can be operated with retentive actual values.
To select the number of data blocks, go to *Project view/System settings/Retention area*. The retentive actual value will require 4 bytes of memory space. If a data block is retentive, the actual value will be retained when the operating mode changes from RUN to STOP and when the power supply is switched off. If the device is started in RUN mode, the data block will continue to work with the actual value stored in non-volatile memory.

**Signal diagram**

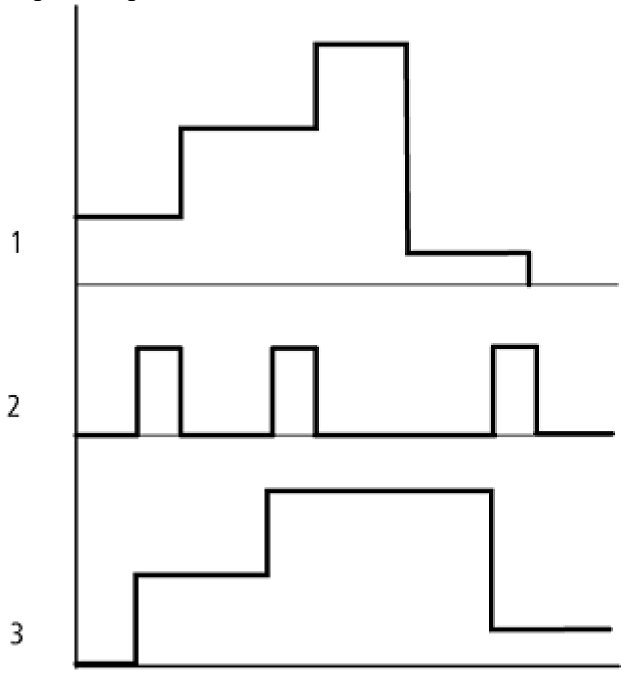



Figure 195: Signal diagram of data function block

Legend for Figure

1: Value at input DB..>I1

2: Trigger coil DB…T_

3: Value on DB…QV>

### Example of a data function block with programming method EDP

The trigger coil is addressed via the network.

```
GT01Q1-------------------------[ DB16T
```

Figure 196: Wiring the trigger coil

The output of the data function block DB16Q1 is assigned to the input D02 EN of the text display function block.

```
DB16Q1-------------------------[ D 02EN
```

Figure 197: Wiring of the function block contact

### Example of a DB configuration on a device display

When using the function block in the circuit diagram for the first time, use OK to automatically enter the display of function blocks on the device display, as shown in the following figure.

```
DB16 +
>I1
QV>
```

Figure 198: Parameters on the display

Enter the function block settings here. The display contains the following elements:

| | |
|---|---|
| DB16 data function block | Function block: Data function block, number 16 |
| + | Parameter set can be called via the PARAMETERS menu |
| >I1 | Input value<br>Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| >I2 | Outputs the value of DB..I1 when triggered.<br>Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

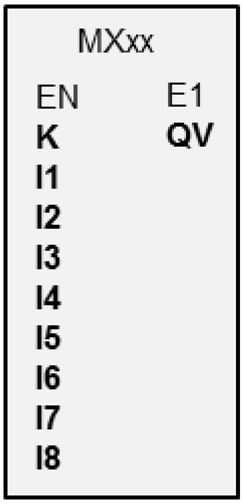
**See also**

→ Section "BC - Block compare", page 388
→ Section "BT - Block transfer", page 395
→ Section "MX - Data multiplexer", page 406
→ Section "SR - Shift register", page 415
→ Section "TB - Table function", page 423
→ Section "RE - Recipe records ", page 410
→ Section "Retention function", page 578
→ Section "Organizing marker ranges", page 202

#### 6.1.5.4 MX - Data multiplexer

**General**

easyE4 base devices provide 32 data multiplexer function blocks, MX01 through MX32. You can use a data multiplexer to select a value from eight input values, I1 through I8. The function block will then provide this value at output QV for further processing.
Use input K (channel number) to specify which input will be connected through to the output. Channel number 0 will connect input I1 to QV, while channel number 7 will connect input I8 to QV.
The MX data multiplexer can be used for the sequential control of up eight to positioning sections that are transferred to the I1 input of the PO Pulse output function block.




**Operating principle**

If there is a signal state of 1 at the EN coil, the data value of the operand connected to input Ix will be connected through to output QV. In this case, the value at function block input K will reference input Ix.

You can change the channel number and accordingly connect a different input value through to QV even when the EN input is set.
If there is a signal state of "0" at the EN coil, output QV will be set to a signal state of "0". The function block will carry out a selection of one out of eight.

**The function block and its parameters**

**Function block inputs**

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the<br>☑ Function block release by EN is necessary<br>parameter must first be enabled |
| **(DWord)** | | |
| K | Channel number<br>References the function block input you want (I1 through I8).<br>channel Function Block Input<br>0 I1<br>1 I2 | Integer value range: 0…7 |

| | Description | | Note |
|---|---|---|---|
| | 2 | i3 | |
| | 3 | I4 | |
| | 4 | I5 | |
| | 5 | I6 | |
| | 6 | I7 | |
| | 7 | I8 | |
| I1…I8 | Input value | | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |

| | Description | Note |
|---|---|---|
| E1 | Error output<br>1: with incorrect parameters , if<br>0 > K or K > 7 | Output QV is reset to 0 if there is a parameter error. |
| **(DWord)** | | |
| QV | Output value of the selected channel | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

#### Other

#### Retention

The function block does not recognize retentive data.

#### See also

→ Section "BC - Block compare", page 388
→ Section "BT - Block transfer", page 395
→ Section "MX - Data multiplexer", page 406
→ Section "SR - Shift register", page 415
→ Section "TB - Table function", page 423
→ Section "RE - Recipe records ", page 410
→ Section "Retention function", page 578
→ Section "Organizing marker ranges", page 202

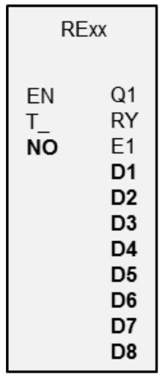
### 6.1.5.5 RE - Recipe records

Only available on easySoft Version 7.10 or higher.

If this function block is not being shown in the leftmost pane in easySoft 7, make sure that you are using firmware version 1.10 or higher for the project.

#### General

easyE4 base devices provide 8 recipes RE01…RE08.
Normally, the word "recipe" is used to refer to a combination of ingredients with quantities, temperatures, and times that is used to make a product – especially a food dish. In manufacturing, a Recipe usually refers to a specific product model or a specific method. and describes a group of various parameters for the relevant product or method type. These parameters are in turn filled with concrete values, which results in one or more data records.




In real-life applications, recipes are used to make it possible to quickly switch between production processes in production systems. The selection is made by the corresponding device operator on the device screen, and, if necessary, it is also possible to enable the operator to edit parameters for a production process.

Recipes cannot be edited at runtime. Neither the recipe parameters nor the values in the records can be changed.

#### Operating principle

When there is a rising edge at T_, the value at function block input NO will be read. NO determines which record (i.e., which recipe) will be read in the function block and output at function block outputs D1 through D8. A maximum of one recipe with a maximum of 32 records can be stored per instance (each record consists of 8 values).

If there is no value at function block input NO, or if there is a value that addresses a record that does not exist and the value is applied with T=1, the function block will signal an error at E1. This error at E1 will then be cleared as soon as there is a correct value at NO. Please note that the values in a record can only be edited in easySoft 7.

| NO | Data1 | Data2 | Data3 | Data4 | Data5 | Data6 | Data7 | Data8 |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 4500 | 3572 | 1564389 | 0967 | 5447 | 79 |
| 2 | 100 | 250 | 455 | 3478 | 34 | 46 | 3 | 44 |
| 3 | 2200 | 1750 | -333 | 45 | 55 | 1750 | 255 | 266 |
| 4 | -6000 | 21474836 | -74836 | 0 | 647 | 232 | 78 | -32999 |
| 5 | -84987 | -31789 | -5255 | -45 | 768 | 235 | 66 | -234 |
| … | … | … | … | … | … | … | … | … |
| 32 | -89365 | -356978 | 21 | -13 | 34999 | -476 | 35879 | -637 |

Records cannot be modified at runtime.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block.<br>0: Resets all function block outputs. | The checkbox for the<br>☑ Function block release by EN is necessary<br>parameter must first be enabled |
| T_ | Trigger input<br>When there is a rising edge at T_, the value at function block input NO will be read. There must be a valid value at function block input NO before T is set to 1 – otherwise, the function block will signal an error at E1. | |
| **(DWord)** | | |
| NO | The number of the recipe with the record that should be output at function block outputs D1 through D8. | Number of records 1…32 |

### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |

| Operators | Bit inputs |
|---|---|
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: if T_=1;<br>0: if T_=0 or E1=1 or EN=0; | |
| RY | 1: The record for the requested recipe with number NO has been loaded.<br>0: No recipes are loaded.<br>0: The value at NO has changed, but the record for the recipe has not been loaded and is not present at D1 through D8. | |
| E1 | Error<br>1: If the recipe with requested number NO does not exist or the value range of NO is exceeded.<br>0: As soon as there is a value at NO with which the record in a valid recipe can be addressed. | |
| **(DWord)** | | |
| D1…D8 | Values in the record of the recipe selected with NO. | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |

| Assigning operands | Bit outputs |
|---|---|
| N - Network marker bit[2)] | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

## Parameter set

| | Description | Note |
|---|---|---|
| **Parameter set** | | |
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything.<br>The "Function block release by EN is necessary" option is enabled by default. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

#### Other

#### Retention

Recipes are parts of the parameter set and are accordingly stored retentively as part of the project.

#### See also

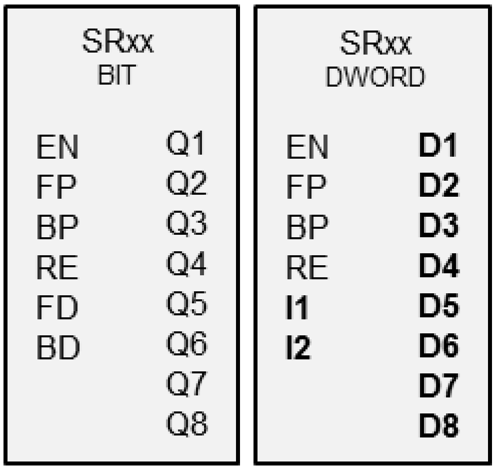
→ Section "BC - Block compare", page 388
→ Section "BT - Block transfer", page 395
→ Section "DB - Data function block", page 401
→ Section "MX - Data multiplexer", page 406
→ Section "SR - Shift register", page 415
→ Section "TB - Table function", page 423

### 6.1.5.6 SR - Shift register

#### General

easyE4 base devices provide 32 shift register function blocks, SR01 … SR32.

These function blocks can be used to shift bits or double words by one position with every clock pulse. You can select the BIT or DWORD operating mode with a parameter. To set the shift direction, you will need to activate either function block input FP (forward pulse) or function block input BP (backward pulse). The values to be accepted in the shift register are located at different inputs depending on the shift direction and operating mode.




The shift register has a linear structure. If, for example, a clock-pulsed bit operation adds a value is at one end of the register, another value is dropped at the other end.

### Operating principle

#### SR function block - shift register (BIT)

A rising edge at FP (ForwardPulse) causes the bit value to be accepted at data input FD (ForwardData) into the first register field Q1. The original contents of the register fields are then moved by one field in the direction of the next higher field numbers.

A rising edge at the BP (BackwardPulse) transfers the bit value at the BD (BackwardData) data input to the last register field Q8. The original contents of the register fields are then moved by one field in the direction of the next lower field numbers.

**Example: Shift register BIT mode, forward**

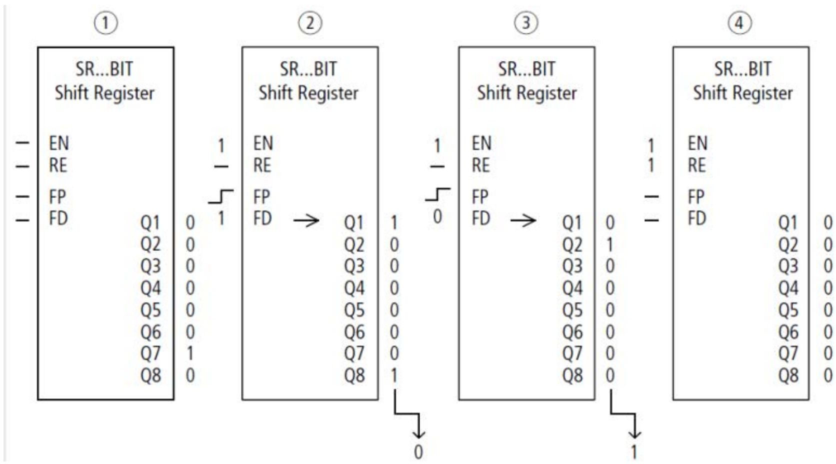



Figure 199: Shift register SR..: Forwards operation in BIT operating mode

① Initial situation
The "Function block release by EN is necessary" option is enabled.
– SR..EN is not activated, the function block is not active.
– SR..Q7 contains a data bit 1, a 0 is contained in the other register fields.

② Transfer of a data bit
– SR..EN is activated, the function block is active.
– SR..FD has the value 1.
– With the forwards pulse from SR..FP the register field SR..Q1 shifts the content of all register fields one place forwards and accepts the 1 from SR..FD.

③ Transfer of a data bit
– SR..EN is activated, the function block is active.
– SR..FD has the value 0.
– With the forwards pulse from SR..FP the register field SR..Q1 shifts the content of all register fields one place forwards and accepts the 0 from SR..FD.

④ Reset of the register
– SR..EN is activated, the function block is active.
– Activating SR..RE clears the content of the register.

#### SR function block - shift register (DWORD)

A rising edge at FP (ForwardPulse) causes the double word value at data input I1 to be transferred to the first register field D1. The original contents of the register fields are then moved by one field in the direction of the next higher field numbers. A rising edge at the BP (BackwardPulse) transfers the double word value at data input I2 to the last register field D8. The original contents of the register fields are then moved by one field in the direction of the next lower field numbers.

**Example: Shift register DW mode, backward**

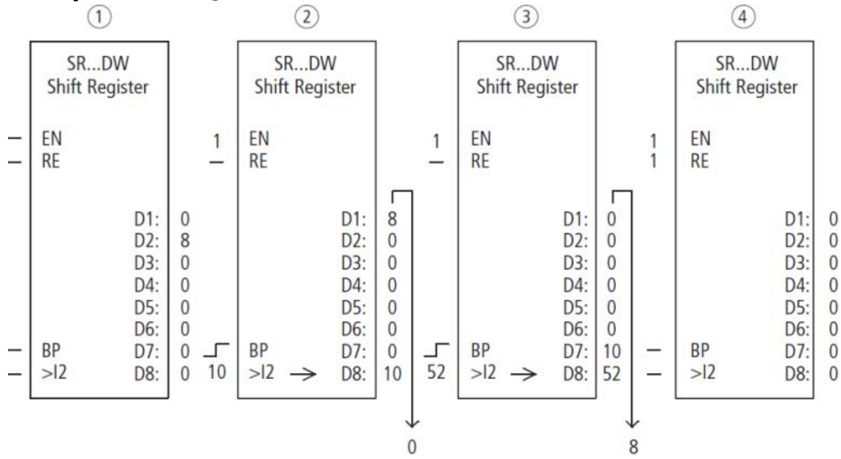



Figure 200: Shift register SR..: Backwards operation in DW operating mode

1. Initial situation
The "Function block release by EN is necessary" option is enabled.
– SR..EN is not activated, the function block is not active.
– SR..D2 contains the value 8, a 0 is contained in the other register fields.
2. Transfer of value
– SR..EN is activated, the function block is active.
– SR..I2 has the value 10.
– With the backwards pulse from SR..BP the register field SR..D8 shifts the content of all register fields one place back and accepts the 10 from SR..I2.
3. Transfer of value
– SR..EN is activated, the function block is active.
– SR..I2 has the value 52.
– With the backwards pulse from SR..BP the register field SR..D8 shifts the content of all register fields one place back and accepts the 52 from SR..I2.
4. Reset of the register
– SR..EN is activated, the function block is active.
– Activating SR..RE clears the content of the register.

## The function block and its parameters

### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| FP | Forward Pulse<br>Single | |

| | Description | Note |
|---|---|---|
| | Clock input, shift register forward | |
| BP | Backward Pulse<br>Clock input, shift register backward | |
| RE | Reset<br>1 clears the entire output register Q1...Q8 and D1...D8. | |
| FD | Bit data input, shift register forward | |
| BD | Bit data input, shift register backward | |
| **(DWord)** | | |
| I1 | Input value for shift direction forwards | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| I2 | Input value for shift direction backwards | |

### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

**Operating modes**

| | Description | Note |
|---|---|---|
| BIT | Marker bit shift operation | |
| DW | Marker double word shift operation | |

The factory setting of this parameter is BIT.

→ The operating mode is determined by selecting different function blocks:
SR - shift register (BIT) or
SR - shift register (DWORD)
and not with the parameters for the function block as is usually the case.

→ If the BIT operating mode is selected, the inputs I1, I2 and outputs D1-D8 are displayed. They have no function in BIT mode! If they are assigned operands, these will have no effect. The wiring of the SR function block (BIT) is carried out in the circuit diagram.

**Function block outputs**

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1…Q8 | Output of the bit register fields 1 - 8 | |
| **(DWord)** | | |
| D1…D8 | Register values for the shift register 1 through 8 | Integer value range: -2,147,483,648 to +2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |

| Assigning operands | Bit outputs |
|---|---|
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

### Retention

The function block does not recognize retentive data.

**Application example**

Different workpieces are moved along a production line with several machining stations. An operator determines the work required for the individual workpieces, creates a production code for it and writes it into a shift register. The workpieces reach the machining stations in this order. When the workpiece is changed, the stations read the required production steps from their permanently assigned register field. When workpiece 1 moves to the first station, the forwards pulse input coil SR01FP and the shift register SR01 receive production code 01 at input SR1I1 from the double word marker MD11. Production code 1 is now at the register field SR01D1 for the first machining station which reads it from double word marker MD01. The finished workpiece is then transferred to station 2. The shift register accepts the production code 2 for the next workpiece.

Production code 1 moves one place forwards, as does the remaining register content. It now stands at register output SR01D2. Double word marker MD02 gets it to production station 2. The process is repeated for each further workpiece and for each further machining station until the finished workpieces leave the line.

**When using the EDP programming language, the coils are connected in the circuit diagram:**

The enable coil SR01EN is permanently active, the function block is not switched off.

Marker M09 switches the forwards pulse input coil SR01FP.

```
-------------------------------[ SR01EN

M 09---------------------------[ SR01FP
```

Figure 201: Circuit diagram with EDP programming language for user example 2

**SR01 configuration on device display**

When using the function block in the circuit diagram for the first time, use OK to automatically enter the display of function blocks on the device display, as shown in the following figure. After defining shift register number 01, you set here the following parameters:

- The DW operating mode for the double word marker format.
- The double word markers for receiving the production code.

```
SR01 DW +
>I1 MD11
>I2
D1> MD01
D2> MD02
D3> MD03
D4>
D5>
D6>
D7>
D8>
```

Figure 202: Parameters on the device display

Enter the function block settings here. The display contains the following elements:

| | |
|---|---|
| SR01 shift register | Function block: SR shift register, number 01 |
| DW | Operating mode: Double word |
| + | Parameter set can be called via the PARAMETERS menu |
| >I1 | Input value DW forwards:<br>Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| >I2 | Input value DW backwards:<br>Integer value range:<br>-2,147,483,648 to +2,147,483,647 |
| D1> | Register value 1 of the shift register,<br>Integer value range:<br>-2,147,483,648 to +2,147,483,647<br>for all registers |
| D2> | Register value 2 |
| D3> | Register value 3 |
| D4> | Register value 4 |
| D5> | Register value 5 |
| D6> | Register value 6 |
| D7> | Register value 7 |
| D8> | Register value 8 |

### See also

→ Section "BC - Block compare", page 388
→ Section "BT - Block transfer", page 395
→ Section "SR - Shift register", page 415
→ Section "MX - Data multiplexer", page 406
→ Section "TB - Table function", page 423
→ Section "RE - Recipe records ", page 410
→ Section "Retention function", page 578
→ Section "Organizing marker ranges", page 202

### 6.1.5.7 TB - Table function

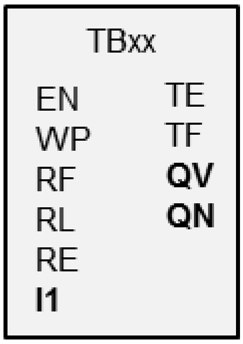
#### General

easyE4 base devices provide 32 table function function blocks, TB01 through TB32.

The table function block allows you to easily create and read table entries in the form of double words (32 bit).

You can select a LIFO or FIFO function for read operations.

A table can contain up to 16 double words.




#### Operating principle

#### Writing to a table

The function block is enabled when EN=1. When the function block is enabled and there is a rising edge at the function block input, the current value at function block input I1 will be added to the table. Every time there is an edge, a double word (32 bits) will be assigned a value.
It is permissible to simultaneously activate the EN coil and the WP coil with a rising edge.
Every new table entry will be appended after the last entry until the sixteenth entry is reached. At the same time, function block output QN will be incremented by 1. QN indicates the current number of entries. If I1 is added successfully, the input value that was just entered will be output at function block output QV.
Once the maximum number of 16 table entries is reached, no more data will be added to the table. If you want to keep making table entries in this scenario, you will first need to clear the entire table with a rising edge at function block input RE. Function block output QN will be set to 0 in this case.

#### Reading from the table

A table can be read from the table's beginning or end.
When there is a rising edge at the RF coil, the oldest value in the table will be read and output at output QV (FIFO function).
The read operation deletes this value from the table and the actual number of entries is decremented by 1 at the QN output.

A rising edge at the RL coil causes the most recent value entered in the table to be output at QV (LIFO function).
The read operation deletes this value from the table and the actual number of entries is decremented by 1 at the QN output.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| WP | Trigger coil<br>Rising edge: The value at I1 will be added to the table and output at function block output QV. QN will be incremented by 1. | |
| RF | Trigger coil<br>Rising edge: The oldest value in the table will be output at function block output QV (FIFO function). QN will be decremented by 1 with each read operation. | |
| RL | Trigger coil<br>Rising edge: The most recent value in the table will be output at function block output QV (LIFO function). QN will be decremented by 1 with each read operation. | |
| RE | Reset<br>Rising edge: The entire table will be cleared. Function block output QN will be set to 0. | |
| **(DWord)** | | |
| I1 | Input value to be transferred to the table. | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |

1) Only on function blocks T, AC
2) Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### The function block and its parameters

#### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| TE | 1: if the table is empty. | |
| TF | 1: if the table is full. | |
| **(DWord)** | | |
| QV | Read operation:<br>The value read from the start or end of the table.<br>Write operation: The input value just entered. | |
| QN | Current number of table entries | Integer value range: 0…16 |

#### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers [2)]<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)[2)] SN - Output bit via NET (send) | x |

| Assigning operands | Bit outputs |
|---|---|
| N - Network marker bit[2)] | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

**Parameter set**

| Configuration/time range | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

#### Retention

The function block does not recognize retentive data.

#### Example of a table function when using the EDP programming language

While in the Programming view:

- Position a TB function block on a coil field in your circuit diagram.
- In the Properties field window select the required function block number on the Circuit Diagram Element tab.
- Assign a numerical operand to the function block input I1 so that a numerical value can be transferred.
- Connect the coils TBxxEN, TBxxWP, TBxxRF etc. with the contact suitable for activation.
- If required, write a comment for the selected operand.

If you wish to check if a table is full or empty, you must also wire this function block as a contact.

- Position the function block on a contact field and select the same function block number in the Circuit Diagram Element tab that you have assigned to the coil.
- If required, change the switch function of the contact from break to make contact.

▶ If necessary, position the function block on a contact field and associate TBxxTE (table empty) and TBxxTF (table full) with a Boolean operand suitable for evaluation tasks.

Whether you position the function relay first of all in a coil field or contact field or whether you make the entries in the Parameters tab of a coil or a contact is not important. It is only important that you have selected the same function block number if you also want to configure the same function block.

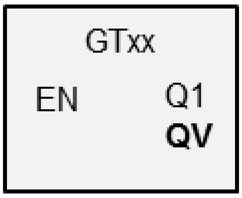
**See also**

→ Section "BC - Block compare", page 388
→ Section "BT - Block transfer", page 395
→ Section "TB - Table function", page 423
→ Section "MX - Data multiplexer", page 406
→ Section "SR - Shift register", page 415
→ Section "RE - Recipe records ", page 410
→ Section "Retention function", page 578
→ Section "Organizing marker ranges", page 202

### 6.1.6 NET Function Blocks

#### 6.1.6.1 GT - Get values from NET

**General**




easyE4 Base devices provide 32 function blocks GT01…GT32 (GET).
The list of operands and function blocks will only include this function block if the project view features a NET consisting of at least two devices.

With function block GT you can poll a 32-bit value from the NET network. The function block automatically fetches the data designated for itself as soon as it is provided on the NET by another NET station using PUT function block PT.

**Operating principle**

The GET function block can be used to read a value from the NET. This value is sent beforehand by the corresponding PUT function block of another NET station. The sent value can be the content of a function block output, a marker byte, word or double word.

Each GET function block is assigned exactly one PUT function block in the parameters for the function block. At runtime, only one single EN enable signal is required, and the received value will be provided each cycle.

→ The SC function block only functions if the NET is running properly.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant 1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers 2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET 2) | x |
| SN - Output bit via NET (send) 2) | x |
| N - Net marker bit 2) | x |
| nN - NET marker bit 2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

#### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: if a new value transferred from the NET network is present. Only valid for one processing cycle | |

| | Description | Note |
|---|---|---|
| **(DWord)** | | |
| QV | Value received from NET | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Function Block Input | NET-ID: The number of the transmitting NET station. | Value range: 01…08 |
| | PT: Number of the transmit function block (e.g. PT 20) by which the transmitting NET station puts a value onto the NET. | Possible function block numbers: 01...32 |
| Simulation not possible | | |

To set parameters, follow the steps below:

- ▶ Uniquely define the sender providing the value for the GET function block. To do this, go to Programming view/Get value from the NET parameters tab /Function block input section/NET-ID and select the number of the transmitting NET station.
- ▶ Select also the number of the transmitting PUT function block from the PT drop-down menu.
- ▶ Connect the QV function block output to an operand to which you want to pass the received value.

### Other

#### Retention

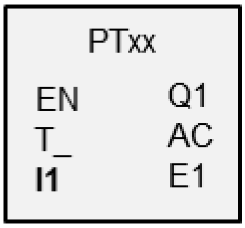
The function block does not recognize retentive data.

#### See also

→ Section "PT - Put values to NET", page 432
→ Section "GT - Get values from NET", page 428
→ Section "Setting up a NET", page 625

### 6.1.6.2 PT - Put values to NET

#### General




easyE4 base devices provide 32 function blocks PT01…PT32 (PUT).

The list of operands and function blocks will only include this function block if the project view features a NET consisting of at least two devices.

These function blocks can be used to pass an operand, which must not be longer than 32 bits, to the NET. The operand value is transferred and automatically read by the corresponding GET function block GT of another NET station.

#### Operating principle

The operand being transmitted needs to be connected to function block input I1. To do this, you can use the output from another function block, e.g., an arithmetic function block. Using a marker double word such as MD1 will make it possible to simultaneously transmit 32 marker bits, M01 through M32.

To transmit marker bits M01 through M96, you would need three PUT function blocks in order to transmit marker double words MD1, MD2, and MD3

You can trigger the transmission with a rising edge at function block input T_. In order for the function block to do another transmission, it would then need to detect another rising edge.

As an alternative, you can have the device do transmissions that depend on the cycle time, i.e., by specifying the number of cycles after which the transmission should occur. This makes it possible to optimize net loads and to transmit values that are not subject to change as frequently less often.

These options are selected in the parameters for the function block.

→ The SC function block only functions if the NET is running properly.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |
| T_ | Trigger coil<br>When there is a rising edge, the function block will temporarily save the input value at I1 and pass it to the NET. | |
| **(DWord)** | | |
| I1 | Input value that is to be put onto the NET. | Integer value range:<br>-2,147,483,648<br>to<br>+2,147,483,647 |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

#### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: if the status of the trigger coil PT..T_ is also 1. | |
| AC | 1: As soon as a transmit job is activated, or as soon as one is aborted with an error message at output E1. | This bit output is used for checking whether the required value was transferred to the NET. |
| E1 | Error - NET transmission error<br>1: If the value could not be sent and the previously set AC output changes from a state of 1 back to 0.<br>The output will remain with a state of 1 until a new transmit job is activated. | |

#### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

#### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set |

| | Description | Note |
|---|---|---|
| | | to 0 or 1 depending on the function block. |
| ☑ No edge evaluation of T bit input | When this checkbox is enabled, the data will be transmitted to the NET based on cycle times. It will be transmitted every nth cycle, where n can be defined in the parameters for the function block.<br>If the checkbox is disabled, manual transmission triggering with an edge at function block input T_ will be required. | |
| Write data to NET after each … <n> cycle | Can only be selected if ☑ No edge evaluation of T bit input is enabled. | |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation not possible | | |

### Other

### Retention

The function block does not recognize retentive data.

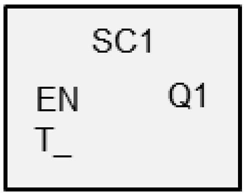
**See also**

→ Section "GT - Get values from NET", page 428
→ Section "PT - Put values to NET", page 432
→ Section "Setting up a NET", page 625

### 6.1.6.3 SC - Synchronizing clock via NET

#### General

easyE4 Base devices provide exactly one function block SC01 (Send Clock).
This function block allows you to selectively place the date and time onto the network. All other NET stations accept the date and time of the transmitting station and set their device real-time clock accordingly.




#### Operating principle

If the function block's trigger coil is activated, the current date, day of the week, and time of the transmitting station will be sent to the NET network. The transmitting station will perform this operation as soon as the seconds counter of the device's real-time clock goes through zero to the next minute. The other stations accept this value. This process can be repeated as often as desired. In this case, the function block input trigger coil must be changed from a state of "0" to a state of "1" again.

#### Accuracy of time synchronization

The maximum time deviation between the functional stations is 5 s.

→ The SC function block only functions if the NET is running properly.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |
| T_ | Trigger coil<br>When there is a rising edge, the function block will transmit the current date, day of the week, and time to the NET. | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: If the transmit job has been completed. | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation not possible | | |

### Other

### Retention

The function block does not recognize retentive data.

**User example**

The trigger pulse is set at 03:32:21 (hh:mm:ss). The other stations are synchronized at 03:33:00. The time is accepted by all stations.

**See also**

→ Section "GT - Get values from NET", page 428
→ Section "PT - Put values to NET", page 432
→ Section "Setting up a NET", page 625

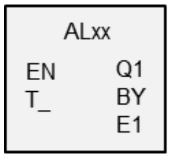
### 6.1.7 Other function blocks

#### 6.1.7.1 AL - Alarm function block

You can use the alarm function block to send e-mails to specific recipients in a targeted manner when specific events occur.

**General**

easyE4 base devices provide 32 alarm function blocks, AL01 through AL32. One e-mail with a defined subject and a defined 160-character message text can be sent with each function block.
This means that a maximum of 32 different messages can be sent to any recipient of your choice. The subject and the message text are both defined in the parameters for the AL function block.
The program triggers the actual sending.




**Operating principle**

In order for the e-mails to be sent, the LAN port on a suitable network must be configured and connected.

A rising edge at function block input T_ will cause the message to be sent. In order for this to work, however, function block output BY must equal to 0.
Message sending starts after each rising edge at T. A maximum of three send attempts will be made per trigger.
If the e-mail is sent successfully, there will be corresponding feedback signals at BY and E1. Otherwise, the job will be aborted, also with corresponding feedback signals at BY and E1.
Deactivating the function block will not result in e-mail sending operation being aborted.

BY will have a state of 1 as long as the job is in progress, preventing any new send jobs from being accepted. If the send job is not completed successfully, an error will be signaled at function block output E1 with E1 = 1.

The recipients and the e-mail server settings need to be set in the hardware configuration.
To do this, a base device needs to be selected in the Project view and the relevant parameters must be configured under the E-Mail tab.
For more information on this topic, please refer to → "E-mail function", page 670.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| T_ | A rising edge will start the communication job. | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers2) | x |
| nNB, nND, nND- NET markers 2) NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: If function block input EN = 1. | |
| E1 | Error output<br>The send job could not be completed successfully after three attempts. Cleared if the job is completed without errors or if the EN input is set to "0". | |
| BY | BUSY<br>1: The most recent send job is still in progress.<br>0: The most recent send job has been completed. | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| Parameter set | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| ☑ Web server active as long as there is a state of 1 | The web server will be selectively turned on and off based on AL_EN.<br>In order for this to work, the web server | Turning off the web server saves processing time |

| Parameter set | Description | Note |
|---|---|---|
| at input EN | must not be permanently enabled; go to → "Webserver tab", page 646 | |
| Parameter display + Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Type of information transmission | E-Mail; there are no other options as of this writing | |
| Recipient assignment | Used to select one of the three possible recipient groups. Each recipient group contains all the detailed information required in order to send an e-mail.<br>1 recipient group; there are no other options as of this writing | Each recipient group needs to be set up in the hardware configuration. To do this, click on "Project" – select a base device – click on the "E-Mail" tab. You can then configure the e-mail server and one or more e-mail recipients for each of the three available groups. |
| Subject | The e-mail's subject | |
| Message text | The text must not exceed 160 characters. | |
| Simulation possible | | |

### Other

### Retention

The function block does not recognize retentive data.

### See also

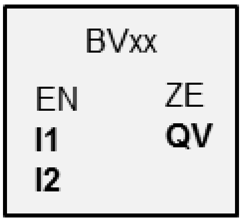
→ Section "E-mail function", page 670
→ Section "BV - Boolean operation", page 444
→ Section "D - Text display", page 448
→ Section "D - Text display editor", page 458
→ Section "DL - Data logger", page 475
→ Section "JC - Conditional jump", page 486
→ Section "LB - Jump label", page 491
→ Section "MR - MasterReset ", page 493
→ Section " NC - Numerical converter", page 497
→ Section "ST - Set cycle time", page 503

### 6.1.7.2 BV - Boolean operation

This function block can be used to define logical operators between the input signal and output signal.

#### General

easyE4 base devices provide 32 (Boolean operation) function blocks, BV01 through BV32. This means that the values at function block inputs BV…I1 and BV…I2 will be connected with a Boolean operator. This function block can be used to mask specific bits from values, detect bit patterns, or change bit patterns.




#### Operating principle

This function block makes it possible to apply Boolean operators to bit groups (bytes, words, or even double words). The size of the parameters at I1 and I2 must be the same, in which case a bitwise AND, OR, XOR, or NOT operator will be applied to them, with the result being output at QV.

#### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | |
| I1 | First value | If one operand assumes a negative value, such as -10 (dec) the processing unit forms the two's complement of the amount.<br>**Example** |
| I2 | Second value | -10 (dec) = 10000000 00000000 00000000 00001010 (bin)<br>Two's complement = 11111111 11111111 11111111 11110110 (bin) = FFFFFFF6 (hex)<br>Bit 32 is retained at 1 as a sign bit. |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |

| Operators | Value inputs |
|---|---|
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating modes

| | Description | Note |
|---|---|---|
| AND | AND operation | |
| OR | OR operation | |
| XOR | Exclusive OR operation (XOR eXclusive OR - exclusive OR, either or) | |
| NOT | Inverts the individual bits of the value at I1. The inverted value is shown as a signed decimal value. | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| ZE | Zero<br>1: if the value of the function block output QV (the operation result) equals zero | |
| **(DWord)** | | |
| QV | Result of the operation | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

### Retention

The function block does not recognize retentive data.

### Example I1 AND I2 = QV

| | decimal | binary |
|---|---|---|
| I1 | 13 219 | 0000 0000 0000 0000 0011 0011 1010 0011 |
| I2 | 57 193 | 0000 0000 0000 0000 1101 1111 0110 1001 |
| QV | 4 897 | 0000 0000 0000 0000 0001 0011 0010 0001 |

**Example I1 OR I2 = QV**

| | decimal | binary |
|---|---|---|
| I1 | 13 219 | 0000 0000 0000 0000 0011 0011 1010 0011 |
| I2 | 57 193 | 0000 0000 0000 0000 1101 1111 0110 1001 |
| QV | 65 515 | 0000 0000 0000 0000 1111 1111 1110 1011 |

**Example I1 XOR I2 = QV**

| | decimal | binary |
|---|---|---|
| I1 | 13 219 | 0000 0000 0000 0000 0011 0011 1010 0011 |
| I2 | 57 193 | 0000 0000 0000 0000 1101 1111 0110 1001 |
| QV | 60 618 | 0000 0000 0000 0000 1110 1100 1100 1010 |

**Example NOT I1 = QV**

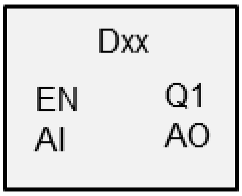
| | decimal | binary |
|---|---|---|
| I1 | 13 219 | 0000 0000 0000 0000 0011 0011 1010 0011 |
| I2 | – | |
| QV | -13 220 | 1111 1111 1111 1111 1100 1100 0101 1100 |

**See also**

→ Section "AL - Alarm function block", page 440
→ Section "D - Text display", page 448
→ Section "D - Text display editor", page 458
→ Section "DL - Data logger", page 475
→ Section "JC - Conditional jump", page 486
→ Section "LB - Jump label", page 491
→ Section "MR - MasterReset ", page 493
→ Section " NC - Numerical converter", page 497
→ Section "BV - Boolean operation", page 444

### 6.1.7.3 D - Text display

#### General




easyE4 base devices provide 32 text display function blocks, D01 through D32. Each of these function blocks can be used to output a custom text display on the easyE4 display or on another external display device and to allow custom input using the device's P buttons.

- Output options
  Each text display consists of 6 lines, with 16 characters each. In other words, 96 characters. These displays are created using a text editor in easySoft 7, and can be used to add graphic macros, texts, value displays, bar graphs, running texts, message texts, date and time entries, etc. on the work pane.
- Input options
  These can be used to enable operators to enter value input and use input buttons. Specific device P buttons can be used to implement user controls within this context.

Various character sets, such as Cyrillic, are available, as if the ability to switch between various user languages. The EN function block input is used to call the function block in the program, and enables the text display.

#### Operating principle

Only one text display function block instance (i.e., only one of the maximum 32) can be displayed at any one time. This must be implemented with the programming, i.e., only one of the text displays should be enabled with the EN input at any one time. If multiple function blocks are enabled instead, the display priority and rolling time will be used to determine which one is displayed. In this case, the program will follow the defined priorities to change from one enabled function block to the next every time after the set rolling time elapses.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The text page for the relevant function block instance will be shown. |
| AI | 1: Acknowledges an alarm message | A rising edge will reset an alarm. Only as long as the function block is still visible. |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)] NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

#### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | Returns the state of input EN. | |
| AO | Acknowledgement pulse for an alarm reset | Only as long as the function block is still visible |

#### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

#### Parameter set

| | Description | Note |
|---|---|---|
| Display priority | 001…032 | 001: highest priority<br>032 lowest priority |
| Rolling time [s] | 001…030 | Text display time<br>when the priority is the same |
| ☑ alarm | Highest priority; takes precedence over all other function blocks | The text display will remain on the device display until it is acknowledged via a rising edge at AI. |

#### Text display tab

The parameters for the text display function block can be configured in the Text display parameters tab. Before you can configure a specific function block (e.g., D02), you will need to select it in the Programming view. If it is the first time you are configuring the function block, the configuration dialog box will show an empty text display consisting of 6 lines with 16 characters each.

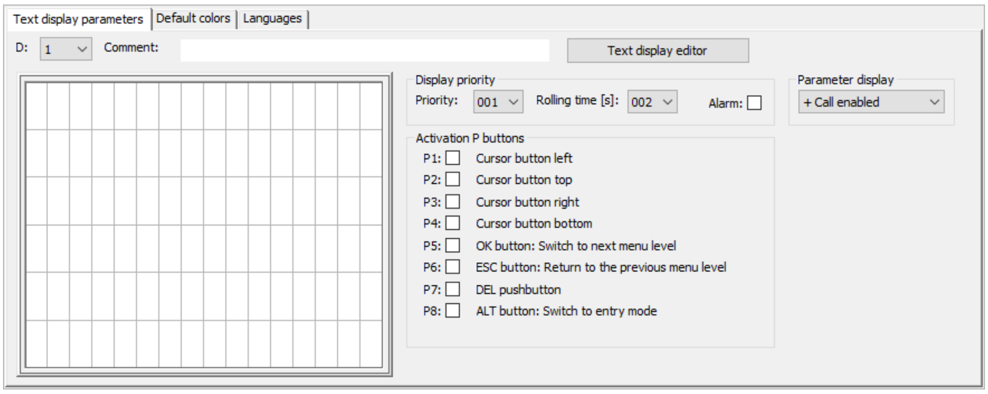



Figure 203: Text display parameters tab for text display function block in the Programming view

### Display priority

If multiple text displays from D01 through D32 are enabled simultaneously, the display priority will be used to determine the order in which they will be displayed on the easyE4 device display. The text display with display priority 001 will have the highest priority, while the text display with 032 will have the lowest priority. The text display with the highest priority will be displayed as long as it is enabled (EN=1). The text display with the next highest priority will not be displayed until the function block input of the previous text display is disabled (EN=0). If multiple text function blocks with the same priority are enabled, the corresponding texts will be displayed in a rolling pattern based on the rolling time. Moreover, when a text display with an alarm becomes enabled, it will be displayed on the device display immediately. (Please refer to the Alarm parameter as well.)

### Rolling time [s]

The rolling time parameter is used to specify how long text displays that have the exact same display priority will be displayed on the device display. In order for this parameter to do something, multiple text displays must be enabled simultaneously (EN=1). Please note that the rolling time is specified in seconds. Moreover, when a text display with an alarm becomes enabled, it will be displayed on the device display immediately. (Please refer to the Alarm parameter as well.)

### ☑ Alarm

If this checkbox is enabled, the corresponding text display will be displayed with absolute priority until the device operator acknowledges the alarm via a rising edge at input AI. This function is only effective for visible function blocks

If multiple text displays with an alarm are enabled simultaneously, the first one that was enabled will remain on the device display until it is acknowledged by a rising edge at input AI. The next text display will be shown then. Once all the text displays with an alarm have been acknowledged by a rising edge at input AI, the text display with the highest priority will be displayed on the device display.

It is important to keep in mind that the alarm acknowledgement function at function block input AI always expects a rising edge. In other words, the AI function block input does not need to be acknowledged immediately, but it does, however, need to be acknowledged by the next alarm reset at the latest.

**Activation P buttons**

The P buttons on the easyE4 device can be used for input and menu control purposes at runtime. You can use these parameters to individually define which buttons should be enabled, and the specific button configuration can be different for each individual text display. Please note that the buttons are required only if you need the operator to enter input or switch screens.

In order for the P buttons to work, they must be enabled with an enabled checkbox next to *Project view/System settings tab/P buttons* – please refer to System settings: → Section "P buttons", page 576 as well.

**Default colors tab**

The easyE4 device display is a monochrome display. Accordingly, the only device display setting that can be changed under the Default colors tab in this case is the backlight color:

- White
- Green
- Red

If you are using an external display device or are showing the device display via a web server, you will be able to configure additional color settings under the Default colors tab. To do this, you can select the predefined colors from the color table.

Left-click inside the color table to select the text color you want.

Right-click inside the color table to select the background color you want.

Please note that you can use the text display editor to configure additional color settings for each element. The color settings you configure in the editor will overwrite any settings you configure here.

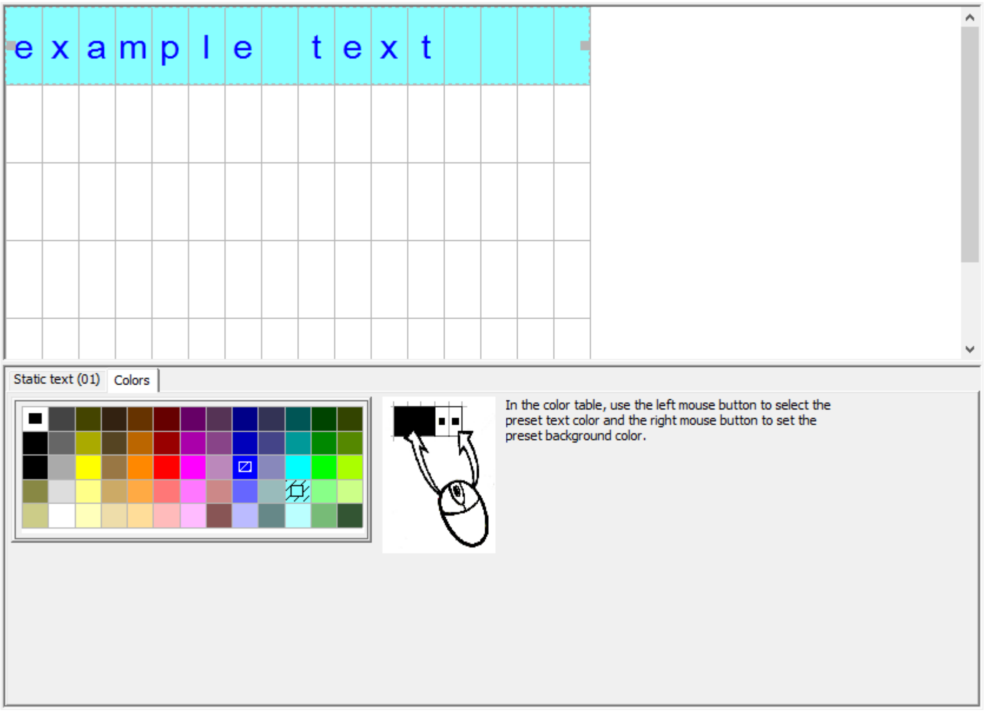



Figure 204: Default colors tab for text display

These will be the default color settings used in the text display editor.

**Languages tab**

You can configure the text display in such a way that the device operator will be able to change the language on the device display or on other external device displays. To do this, you will need to configure the appropriate settings under the Languages tab.

You can assign each language any name of your choice in the Language table column. After doing so, you will need to open the text display editor and enter the text in each language for each text element being used.

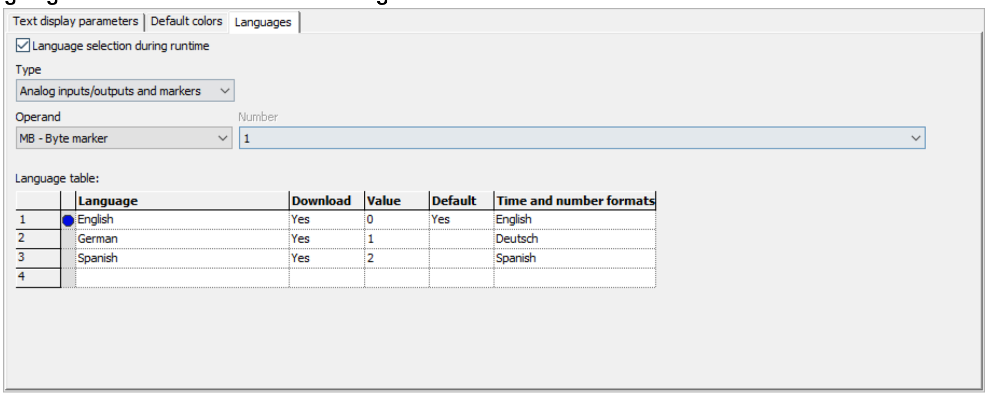



Figure 205: Text display function block, language tab

**☑ Language selection during runtime**

Allows device operators to switch language at runtime.

### Type and operand

User to select an operand that will be used to select the target language. You can use analog inputs or outputs from function blocks, marker bytes, marker words, marker double words, analog outputs, or analog inputs as an operand.

MB1 is selected in the following example. Look at the "Value" column, which is filled out by the system. When MB1 is assigned a value of 1 in the program, the language will be switched to English.

**Language table**

| Column | Description |
|---|---|
| Language | In the language table, you can assign any name of your choice to each language in the project. |
| Download | Selecting <Yes> in the Download column will make the texts for the corresponding language be loaded onto the device. You can enter these texts for each language in the text display editor under the tab for the selected display and entry element. |
| Value | If the assigned operand assumes this value at runtime, the display will switch to the corresponding language. |
| Default | You can select a language as the default language. Selecting <Yes> in the Default column will make the corresponding language be selected whenever the current value of the operand cannot be found in the Value column. In other words, the default language will be used when a specific language is not selected. |
| Time and number formats | The time and number formats for each language will be taken from the formats specified in the corresponding column. Every text that is configured will then need to be entered in each of the defined languages when defining the text element in the text editor. |

### Other

### Signal diagram for text displays with different priorities

The following signal diagram shows four different text displays with different priorities. The text display with the highest priority, D01, is displayed first. As soon as D01_EN=0, other enabled text displays are output – D02 in this example. And as soon as a text display with an alarm is enabled, e.g., D06_EN=1, the text display will be displayed and remain until the alarm is acknowledged with D06_AI=1. After the alarm is acknowledged, the enable text display with the highest priority or with an alarm will be displayed instead. In this example, D07 will be displayed until it is acknowledged with D07_AI=1, after which the display will switch to D02, i.e., the only remaining text display.

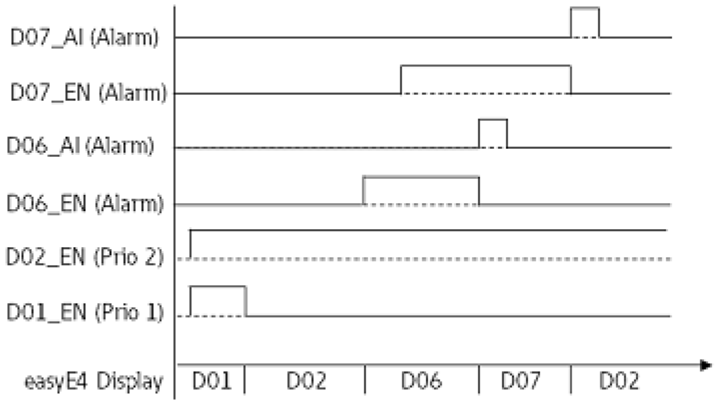



Figure 206: Signal diagram Text display

### Signal diagram for text displays with identical priorities

Text displays D03, D04, and D05 all have the same priority of 3. They will accordingly be displayed according to their configured rolling time as soon as no other text displays with a higher priority are enabled. In other words, D01_EN must be 0 and D02_EN must be 0 for this to happen in the example below. D03, D04, and D05 will be displayed in alternating order until a text display with a higher priority is enabled, e.g., D02_EN=1.

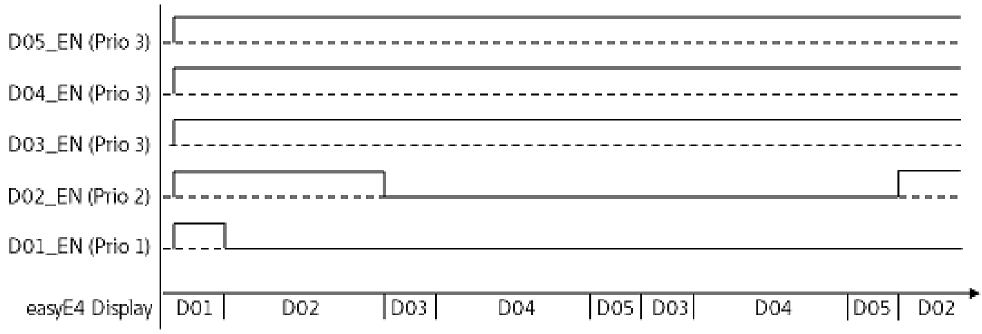



Figure 207: Signal diagram for text display with text function blocks with an identical priority of 3

Rolling time: D03 = 1s; D04 = 3s; D05 = 1s

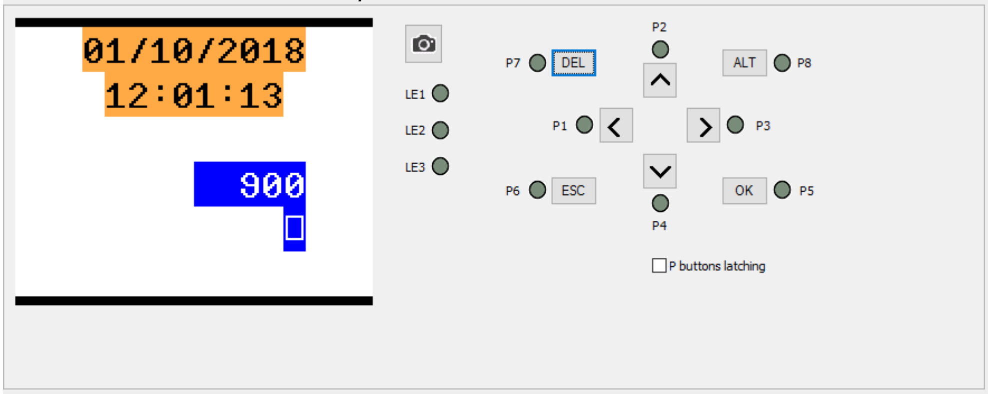
### Retention

The function block does not recognize retentive data.

### Example

**Entering data on the display via a D text function block**

If an easyE4 with a display is used with the text function block and the cursor buttons are enabled in the configuration, operators will be able to enter data using these buttons. To do this, the input mode needs to be accessed by pressing the ALT button. You can also simulate this with easySoft.




The input fields will then be highlighted in color or be shown with inverse colors.

To select an input field and enter data, the operator will need to use the arrow buttons. The currently active cursor position will flash.

UP: The numeric value at the current cursor position will be incremented

DOWN: The numeric value at the current cursor position will be decremented

RIGHT: The next smaller decimal place will be selected or the input value to the right or underneath will be selected

LEFT: The next larger decimal place will be selected or the input value to the left or above will be selected

In the example above, there are three input values on the screen: a value entry, a latching button, and a message text selection.

The value entry [with a value of 900 in the screenshot] consists of three decimal numbers in which the value for each number is entered individually. The latching button [the checkbox with the checkmark] is activated. The question marks show the 16-character area for the message text selection, in which the UP/DOWN buttons can be used to select one of the configured texts.

Once a new value is entered, it can be confirmed with OK. This will exit input mode.

**See also**

→ Section "AL - Alarm function block", page 440
→ Section "BV - Boolean operation", page 444
→ Section "D - Text display editor", page 458
→ Section "DL - Data logger", page 475
→ Section "JC - Conditional jump", page 486
→ Section "LB - Jump label", page 491
→ Section "MR - MasterReset ", page 493
→ Section " NC - Numerical converter", page 497
→ Section "D - Text display", page 448

#### 6.1.7.4 D - Text display editor

easySoft 7 features a text display editor that can be used to design text displays. In order to be able to access it, there must first be a text display function block in the Programming view work pane and the you must have clicked on the function block. Open the Text display parameters tab and click on the Text display editor button. The text display editor will be opened in a separate window.

**Properties Text display editorText display editor**

Text displays are put together with a text display editor that makes input with free text and actual values from various function blocks possible.
This editor features the following properties:

- 6 lines x 16 characters - 96 elements
- Texts can be positioned freely within the text display
- Analog value, timer value, and time value processing
- Message texts, times, dates, and checkboxes as input and output elements
- Simple value entry and controls
- User-controlled acknowledgements
- Ticker text with variable speeds
- Variable display times
- Prioritization by the user
- Multi-language capabilities
- Can also be used in user function blocks

### Working with the text display editor

In order to position a display or input element, follow the steps below:

- Select the display or input element you want from the list, e.g., Static text.
- While holding down the left mouse button, drag the element to the work pane and drop it where you want it to be positioned.
- Move your cursor over one of the element handles and drag the handle until the display or input element has the size you want.
- Configure the parameters using the tabs below. For example:*Static text (01) tab/Text field*<Sample text>.

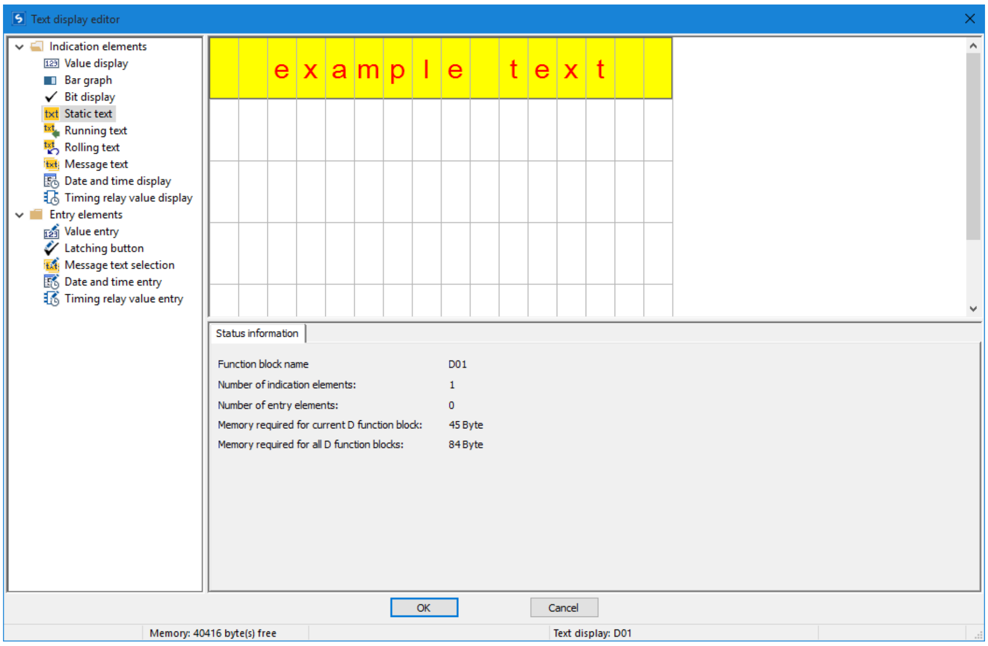



Figure 208: Text display editor with static text in the first line

1 List with display and input elements

2 Work pane with text display elements that have already been configured

3 Status information tab with parameters for the display and input elements

### Color management in the display text editor

Each element will get its own text color and background color as parameters.

If you use the inverse display mode, the colors will be swapped.
If you enter 0 as a color, "default color" will be used.

### Insert special characters

In addition to the characters on your keyboard, you can also enter special characters . To do so, use the CTRL+C/CTRL+V or ALT+ASCII code shortcuts.

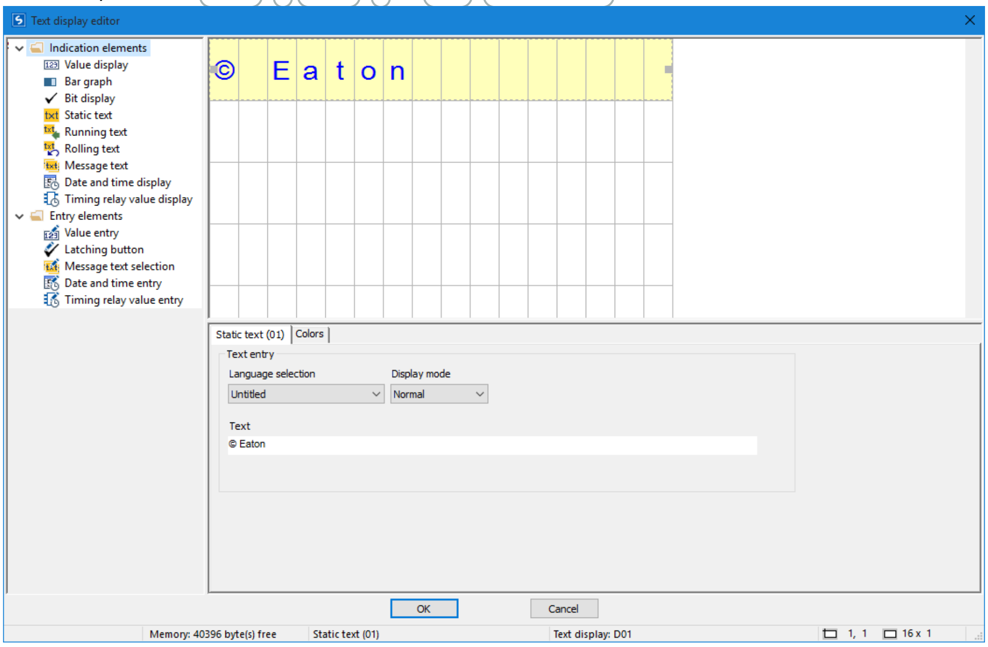
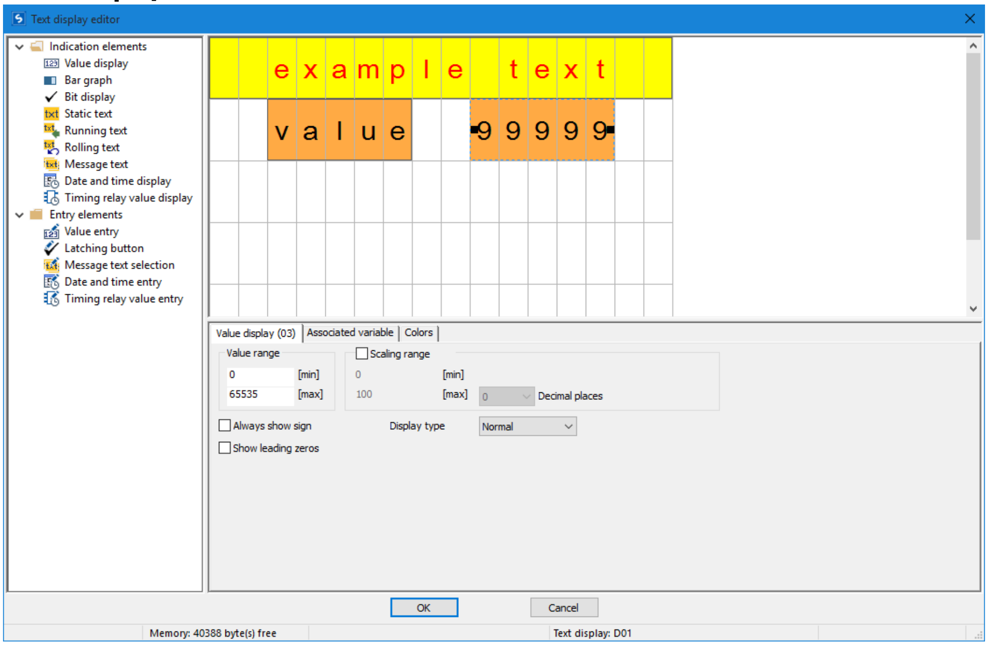



Figure 209: Character table Special characters

## Display and input elements

### Value display




Value displays can be very effectively combined with a static text. In the example above, the value display element has been placed to the right of the "Value" text in the preview pane. The display in the example is intended to have five digits, which is why the number of characters has been configured accordingly. (The number 9 symbolizes value displays.)

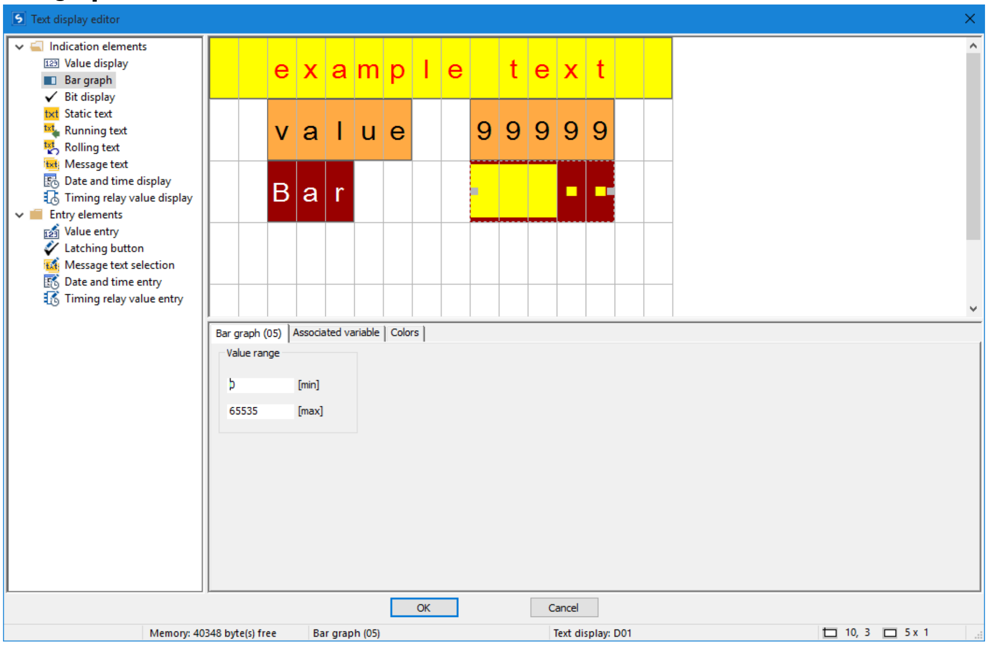
Value range: The default value range is 0-65535. If you want to make it smaller, you can enter the corresponding limits in this section. If the actual value falls outside the configured value range, the display will instead show the nearest value that still falls within the value range.

Scaling range: If you want the value to be scaled for display, enable the "Scaling range" option. Then enter the minimum and maximum values for the scaling.

Displays can be customized with a sign and/or leading zeros.

"Associated variable" tab: The settings in this tab can be used to select a byte, word, or double word value from the operand resources and the function block inputs and outputs so that it can be displayed.

### Bar graph




Bar graphs can be very effectively combined with a static text. In the example above, the bar graph element has been placed to the right of the "Bar" text in the preview pane. The display in the example is intended to have five digits, which is why the number of characters has been configured accordingly.

Value range: The default value range is 0-65535. If you want to make it smaller, you can enter the corresponding limits in this section. If the actual value ends up falling outside of the value range as a result, arrows pointing up or down will be used to indicate this.

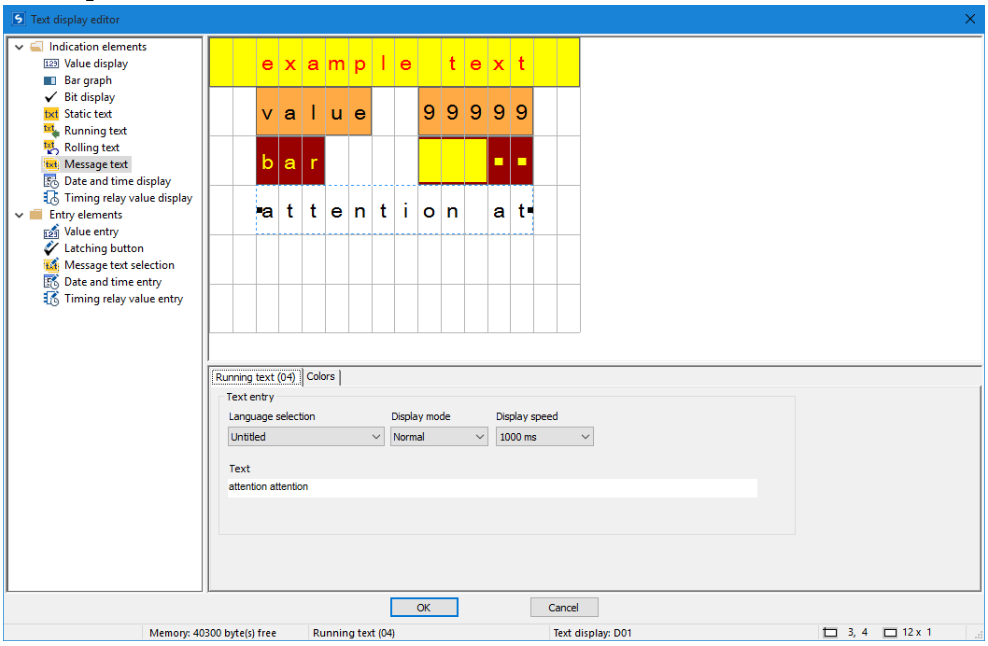
Associated variable tab: The settings in this tab can be used to select a byte, word, or double word value from the operand resources and the function block inputs and outputs so that it can be displayed.

### Static text

To place a static text in the first line, follow the steps below:

- ▶ Select the Static text option from the list, hold down the left mouse button and drag the display element to the work pane, then drop it in the line where you want it.
- ▶ Enter the text you want into the *Static text (01) tab/Text field*, e.g., <Sample Text>.
- ▶ Move your cursor over one of the element handles and drag the handle until the static text element has the size you want.

## Running text

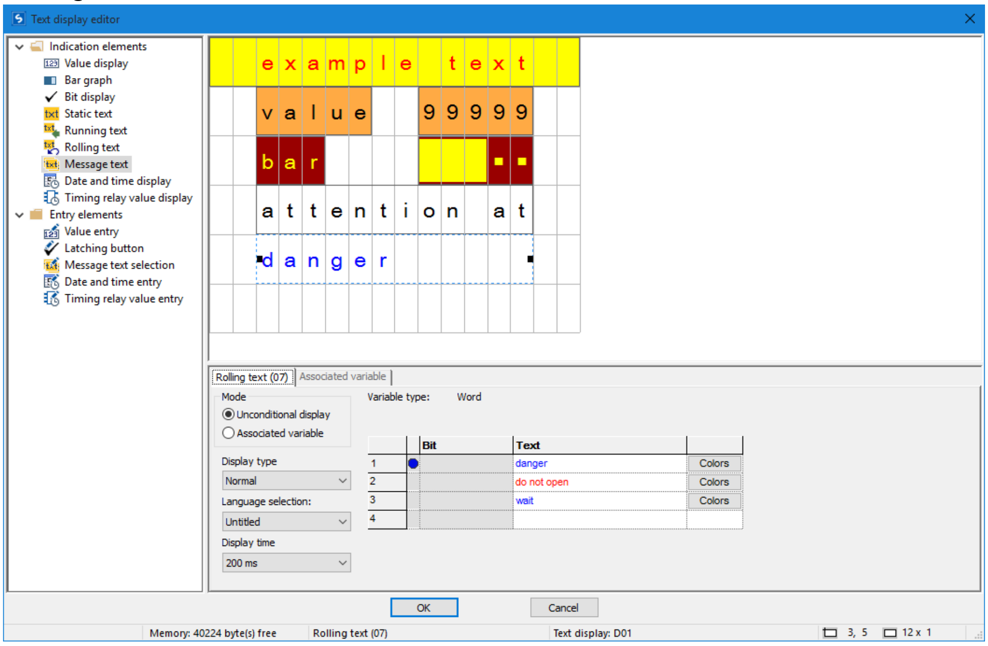



If you want to display a text that is longer than 16 characters, you can use the running text display element. This option can come in particularly handy when you want to draw the machine operator's attention to a text.

Simply select the running text display element from the list, hold down the left mouse button, and drag the element to the work pane. You can then move your cursor over one of the element handles and drag the handle until the running text element has the size you want.

You can use the settings in the corresponding tabs to enter the actual text and set the language, the display mode, and the display speed.

### Rolling text




Rolling texts can be used to display various texts in succession in a single line. You can display various messages or errors, which in this case will be shown in sequence one after the other at a set interval.

The required texts need to be entered into the table under the Rolling text tab, where you can also select the corresponding colors and the display type.

#### Unconditional display

When this operating mode is selected, the texts will be displayed in succession at the specified interval. This interval can be defined using the Display time parameter.

#### Associated variable

When this operating mode is selected, the text selection will depend on the application program. More specifically, an operand that can be selected under the Associated variable tab will be used to control how the texts are displayed. You can select local or network operands of type byte, word, or double word. Each text will then be automatically linked to a bit from the selected operand as you enter it.

- Text 1 will be assigned bit 1
- Text 2 will be assigned bit 2
- Text 3 will be assigned bit 3
- etc.

If bit 2 is set in the program at runtime, for example, text 2 will be displayed. If multiple bits in the operand are set simultaneously, the corresponding texts will be displayed in succession. In this case, the interval at which they are displayed will correspond to the value configured with the Display time parameter.

### Message text

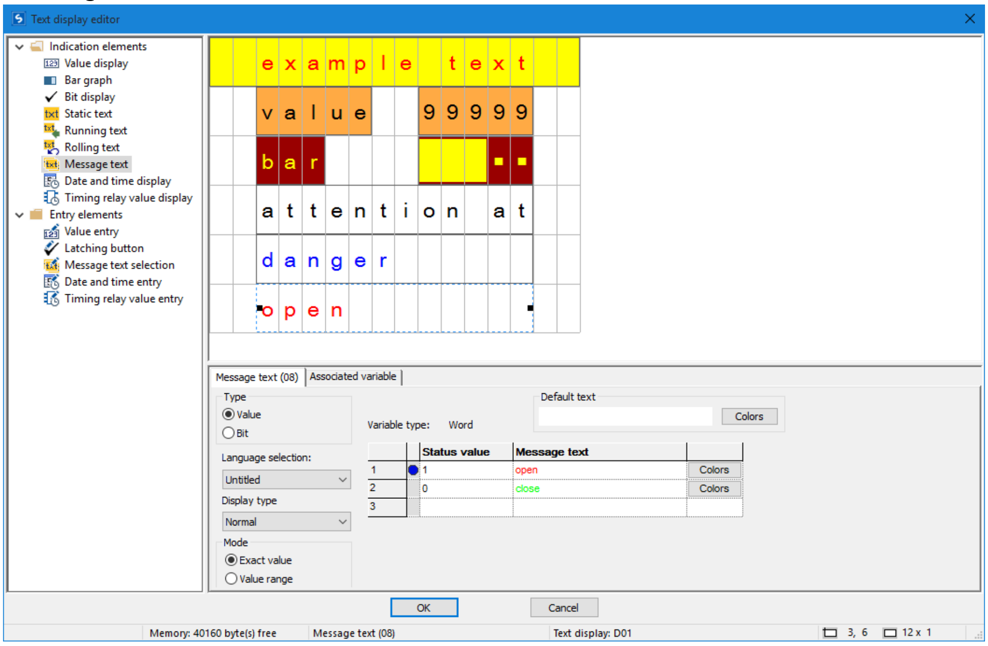



Figure 210: Example showing an exact value message text

Message texts can be used to display various texts in succession in a single line. This can come in handy, for instance, for operating or maintenance procedures in which the display needs to ask the operator or maintenance technician to carry out a step, then another step, then another, etc. In this case, the text can be changed every time the operator or maintenance technician performs the requested action (increment/decrement).

The required texts need to be entered into the table under the Message text tab, where you can also select the corresponding colors and the display type. To do this, you need to specify a status value for each system text. This status value (bit or decimal value) will then be used to call the corresponding message text in the user program (if you are using more than two texts, please make sure to select the "Value" type). An operand that can be selected under the Associated variable tab will be used to control how the texts are displayed. You can select local or network operands of type byte, word, or double word.

#### Default Text

The default text will be shown in exact value mode as soon as the value of the associated variable does not match any of the stored status values.

The default text  will be shown as soon as the value of the associated variable is less than the lowest specified status value.

#### Resize mode

- Exact value
  If you select the exact value, a text will only be displayed if the value is exactly equal to the configured value mode.

- Value range
  In value range mode, the value range of the associated variables will be the value range for possible status values – please refer to → "Elementary data types", page 200.
  This value range can be subdivided further so that the appropriate message text will be output based on the value of the associated variables. In this case, the subdivision will always start with the status value that was entered and end before the next status value that is entered. Moreover, the default text will be shown for any value that is less than the lowest specified status value. And for each value that is greater than or equal to the highest specified status value, the message text for that status value will be shown (all the way to the end of the value range).

This can come in handy, for instance, when trying to abstract analog values – please refer to the description for a fill level in the following example:

**Value range message text example**

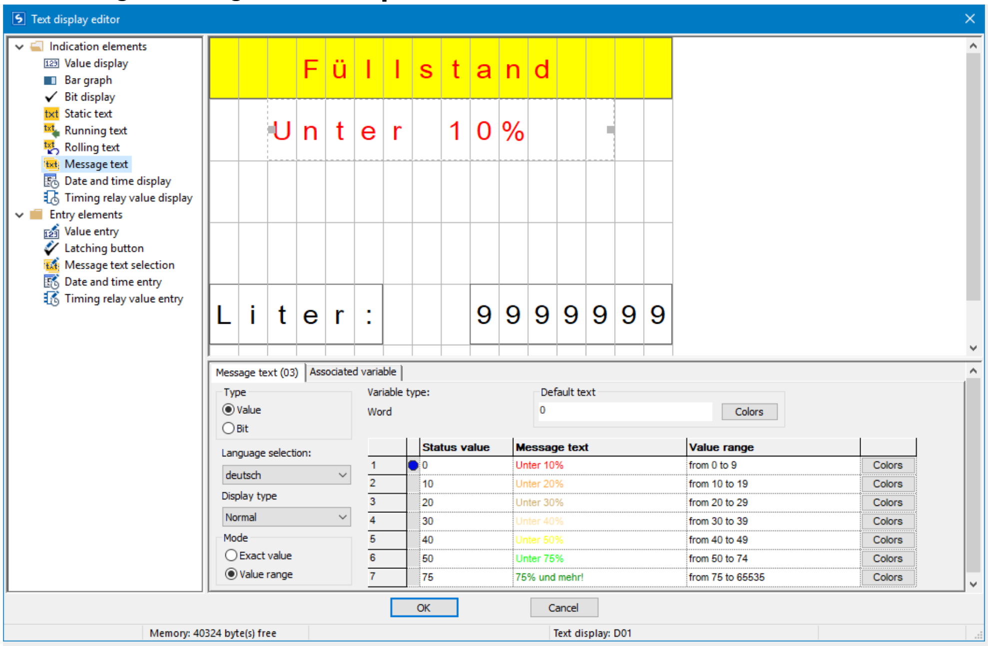



Figure 211: Value range message text example

The value range starts with the status value defined for the message text. This yields the following value ranges:

0…9 : Less than 10%

10…19 : Less than 20%

20…29: Less than 30%

30…39: Less than 40%

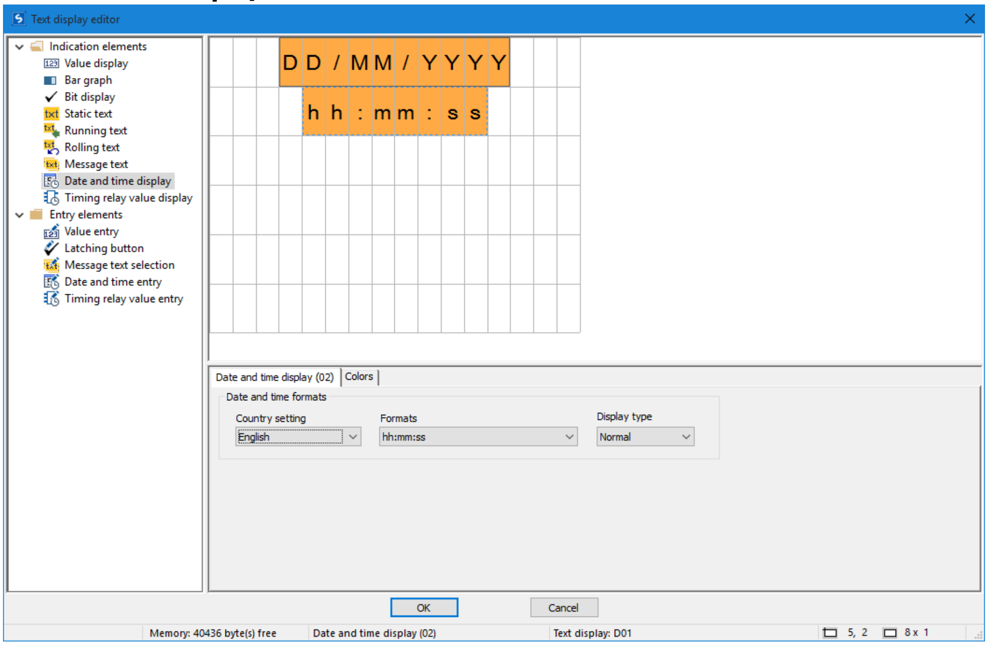
…

75…65535: Greater than 75%

The maximum value will depend on the type of associated variable. In this particular case, it is a marker word with a value range of 0 to 65535.

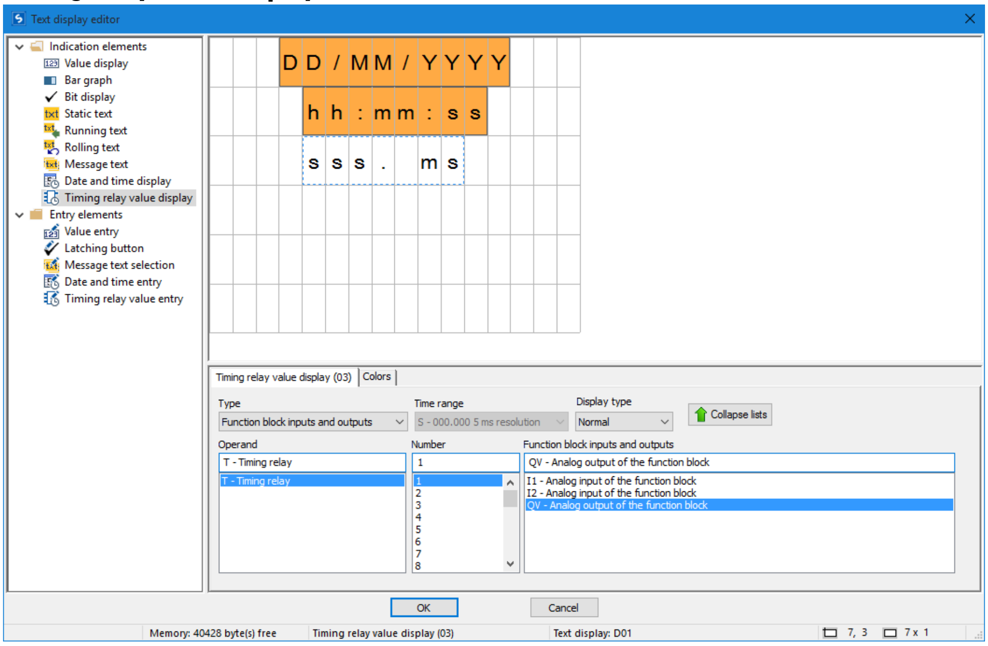
In this specific example, the default text will not be shown.

### Date and time display



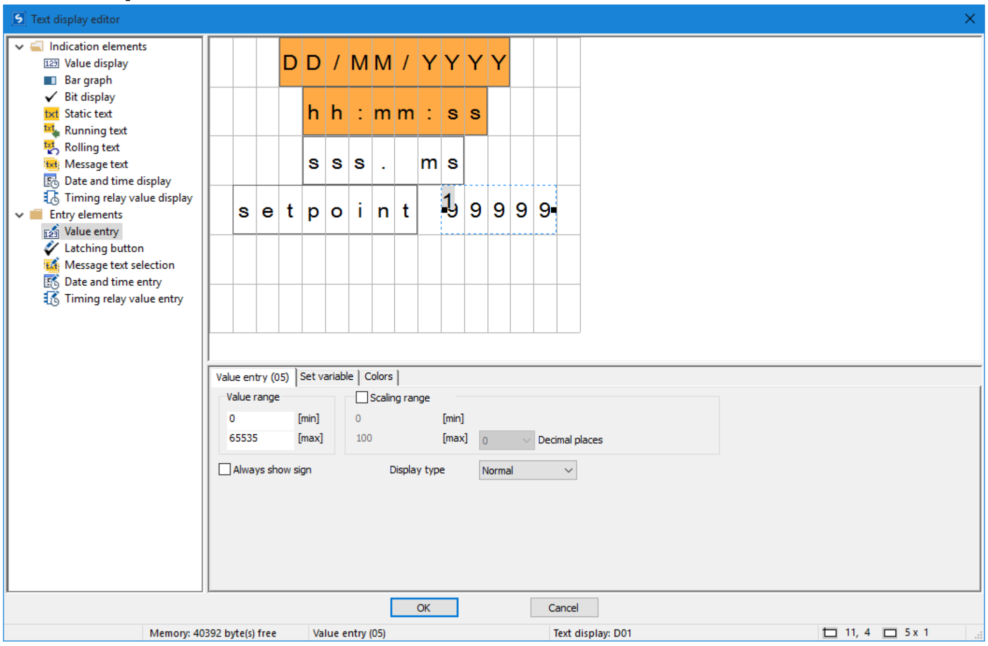

Date and time elements are available in various display formats. Drag a date and time display element to the screen and then select the format you want. The example above contains two data and time display elements configured with the same background color.

### Timing relay value display




Time functions can be implemented using the timing relay value display elements. You can conveniently display the reference value or the running time value in its own display element. Please note that the number of characters, and the size of the display element accordingly, is fixed. To configure the element, you will need to select the timing relay function block number you want and configure the parameters you want. You can also reference operands such as markers directly as a source for the display, in which case you will need to make sure that the data format used in the operand is the format for a timer value.

### Value entry




You can also use the display and the keyboard on the easyE4 to enable operators to enter input. To do this, drag a <value entry> element onto the screen. The value entry element is indicated with <99999>, with the superscript <1> indicating that the element is meant for value input. The <Setpoint> text in the example above is a separate display element of type static text and is used to describe the function of the value being entered.

After being entered, the value will be written to a "set variable" that can be selected under the corresponding tab. Moreover, this element can be configured with scaling, which can be activated by enabling the "Scaling range" option.

You can define the valid value range for the value written to the set variable in the "Value range" section. In this particular example, the full value range of 0 to 65535 that is allowed with a word width has been selected. In order to make things easier for operator, however, the value entered will fall within a range of 0 to 100 instead (this is a good idea, for example, when entering a container fill level for which the fill level percentage is sufficient in terms of accuracy). Accordingly, the scaling range is set at 0 to 100 in the example.

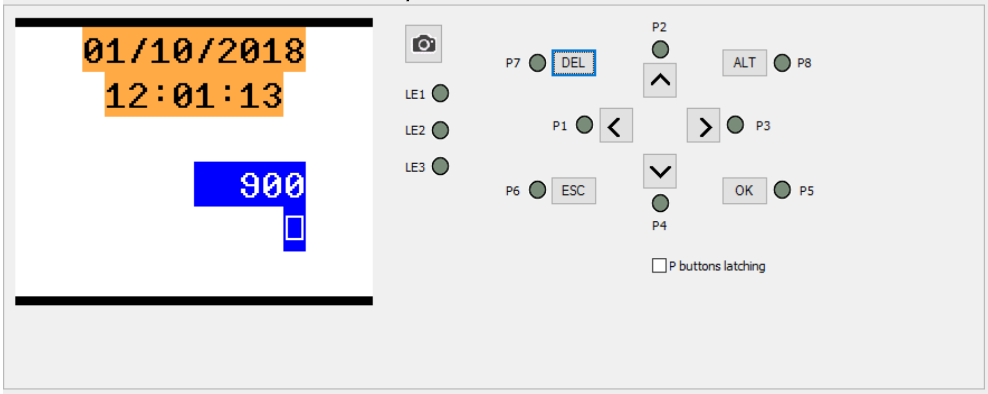
Example: If the operator enters a value of 40, a value of 65535*0.4=26214 will be written to the set variable.

☑ Scaling range

Enabling the checkbox will allow you to configure the scaling range for the value entry element. If you, for example, enter a value of <1000> into the [max] field, the value input will be limited to 4 digits, <9999>.

#### Example: Entering data on the display via a D text function block

If an easyE4 with a display is used with the text function block and the cursor buttons are enabled in the configuration, operators will be able to enter data using these buttons. To do this, it is first necessary to switch to input mode by pressing the ALT button. You can also simulate this with easySoft 7.




The input fields will then be highlighted in color or be shown with inverse colors.

To select an input field and enter data, the operator will need to use the arrow buttons. The active cursor position will flash.

UP: The numeric value at the current cursor position will be incremented

DOWN: The numeric value at the current cursor position will be decremented

RIGHT: The next smaller decimal place will be selected or the input value to the right or underneath will be selected

LEFT: The next larger decimal place will be selected or the input value to the left or above will be selected

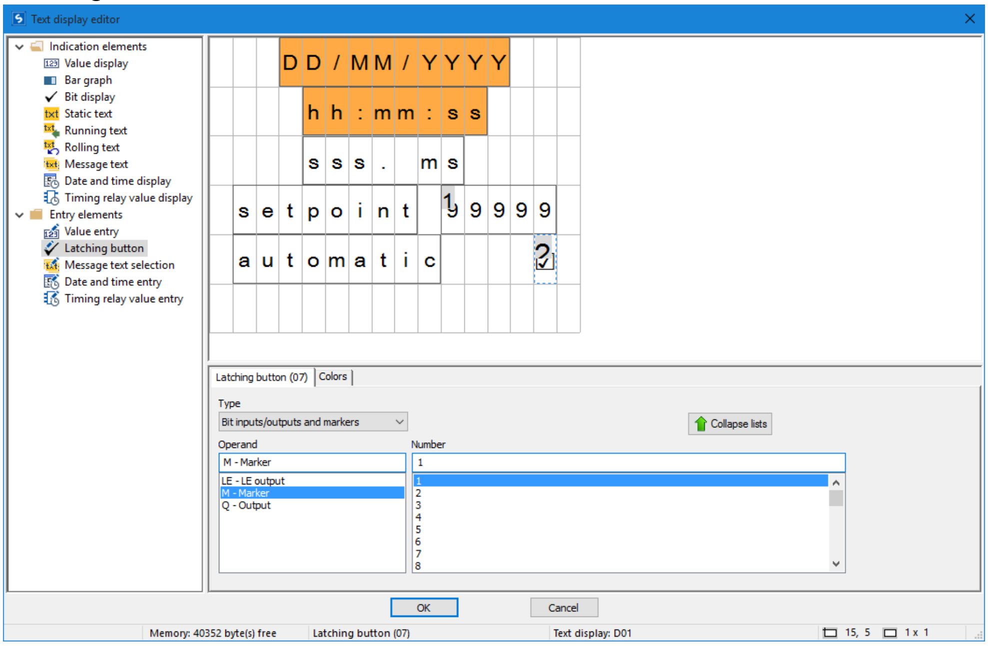
In the example above, there are three input values on the screen: a value entry, a latching button, and a message text selection.

The value entry [with a value of 132 in the screenshot] consists of three decimal numbers in which the value for each number is entered individually. The latching button [the checkbox with the checkmark] is activated.

Once a new value is entered, it can be confirmed with OK. This will exit input mode.

→ The values entered will be stored page by page.
If the text display contains multiple input elements that affect the same associated variable, clicking on OK will assign the value of the input element with the highest index to the associated variable.

### Latching button

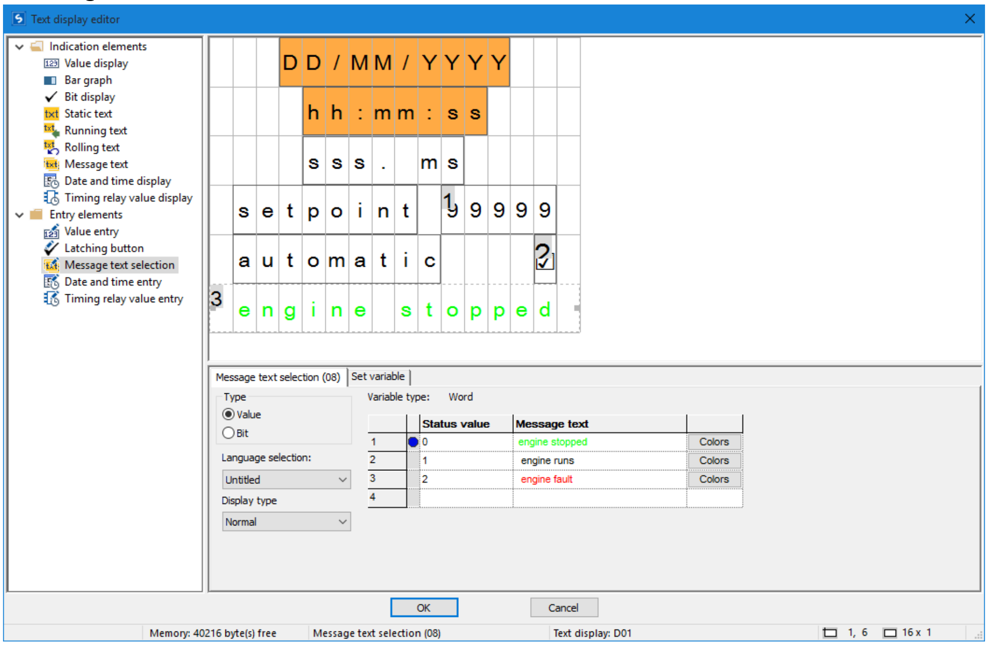



Latching button input elements can be used to visualize and enter binary values with the use of a checkbox and checkmark. Two different colors can be used based on a Boolean value. To configure this type of element, you will need to select a bit operand (marker bit 1 in the example above).

To enter input mode, press the <ALT> button at runtime or during the simulation. You will then be able to use the checkbox by pressing the P2 or P4 button, and the binary value will switch between 0 and 1 accordingly.

The superscript $^2$ on the ☑ indicates that it is the second parameter on the page that can be modified by input – please refer to → Section "Example: Entering data on the display via a D text function block", page 471.

### Message text selection




Normally, message texts are activated by the program on the easy. However, it is also possible to have operators call message texts as "input" for the program on the easy (when selecting operating modes, for example). (when selecting operating modes, for example). One example is a scenario in which a machine can produce various colors and the operator needs to select one: black socks, brown socks, blue socks.

This element is configured exactly the same way as a message text – please refer to → Section "Message text", page 465.

The only difference is that the message text selection element allows for operator input – please refer to → Section "Example: Entering data on the display via a D text function block", page 471.

### Date and time entry

This element is configured exactly the same way as a date and time display – please refer to → Section "Example: Entering data on the display via a D text function block", page 471

The only difference is that the element allows for operator input.

### Timing relay value entry

This element is configured exactly the same way as a timing relay value display – please refer to → Section "Example: Entering data on the display via a D text function block", page 471

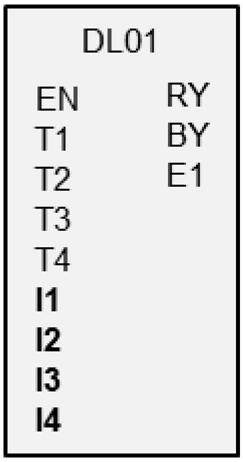
The only difference is that the element allows for operator input.

**See also**

→ Section "AL - Alarm function block", page 440
→ Section "BV - Boolean operation", page 444
→ Section "D - Text display", page 448
→ Section "DL - Data logger", page 475
→ Section "JC - Conditional jump", page 486
→ Section "LB - Jump label", page 491
→ Section "MR - MasterReset ", page 493
→ Section " NC - Numerical converter", page 497
→ Section "D - Text display editor", page 458

### 6.1.7.5 DL - Data logger

#### General




easyE4 Base devices provide exactly one data logger function block DL01.
You can use the data logger function block to write operating data with a timestamp to a log file on a memory card in the easyE4 base device. In order for this function block to work, a memory card must always be inserted in the device. You can select a filename for the log file when configuring the function block.
Digital function block inputs T1 through T4 and analog function block inputs I1 through I4 are always logged for each data set. In addition, the log will indicate which input triggered the logging operation.

#### Operating principle

Logging can be triggered with a rising edge at one of the trigger inputs T1 through T4 or a change at analog function block inputs I1 through I4. You can use the Delta ΔI parameter for each function block input (I1 through I4) to specify the data change magnitude starting from which logging will be triggered.

There are two storage modes available for selection:

1. All events will be saved in a single file [ring buffer], meaning that the file will grow with every save operation. Once the maximum number of data sets is reached, the entries at the beginning will be overwritten.
2. A new file will be created for each save operation until the number of log files is reached.

Any byte, word, or double word operand can be connected to analog inputs I1 through I4.

#### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |
| T1 | 1: Logging data set. | |
| T2 | 1: Logging data set. | |
| T3 | 1: Logging data set. | |
| T4 | 1: Logging data set. | |
| **(DWord)** | | |
| I1 | Analog value 1 for storage | |

| | Description | Note |
|---|---|---|
| I2 | Analog value 2 for storage | |
| I3 | Analog value 3 for storage | |
| I4 | Analog value 4 for storage | |

→ If too many log entries are made in a short amount of time, some of the entries may be lost. An important factor to consider within this context is the speed of the memory card being used. If logging is triggered via function block inputs T1 through T4, this problem can be avoided by only having one save operation be triggered if the function block is not in the "logging disabled" state.

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| RY | Ready<br>0: Logging is active<br>1: Logging inactive<br>RY = 0 always applies to ring buffers;<br>Until number of log files reached: Logging is active until the predefined number of files per log session have been filled with the specified number of records. | Logging can be disabled because:<br>• n log files have been written<br>• The memory card is full<br>• No memory card fitted<br>• Memory card faulty |
| BY | Busy<br>1: Logging not possible | Possible causes:<br>• The card is currently being written to<br>• The temporary internal buffer is full |
| E1 | Error output<br>1: Data loss | Possible causes:<br>• No memory card fitted<br>• The memory card does not have enough space for another log file<br>• Memory card faulty<br>• Temporary internal buffer is exceeded by at least one record |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| Directory name of log session | Enter the name of the folder that contains the log files here, such as <MYLOG>. A maximum of 8 characters are allowed, and they must conform to Microsoft DOS conventions. The default name is <EASYLOG>. | |
| Storage mode | Ring buffer<br>Until number of log files is reached | |
| Number of files per log session | A log session contains n log files | Integer value range for n: 0…1000 |
| Number of data sets per log file | A log file contains n data sets | Integer value range for n: 0…60 000 |
| Log when input values change | If the changes at DL_I are greater than or equal to ΔI, a data set will be logged. | Integer value range for ΔI: 0…65 535 |
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Storage mode

You can select between Ring buffer and Until number of log files is reached :

Ring buffer

If you select the ring buffer mode, the older values will be overwritten after a certain amount of time. If you select this option, you will be able to look at past values up to a certain limit, as old data will be lost at some point.

Until number of log files is reached
Up to n log files will be created in succession in a single folder. The name of the files will be <number>.log, where <number> will be an eight-digit number that is counted up starting from 00000001.log.

The individual log files are only visible on the file level.

The data logged with the DL data logger manufacturer function block is stored on the card in a folder. The folder is named as entered in *Programming view/Data logger parameter tab* under the folder name of the log session.

The data in the recording is binary-encoded and cannot be read with standard Windows PC tools. You can instead read it in easySoft 7, where you can view the logs on the card and export them to Excel. To do this, click on *Project menu/Card.../Data logger logs/icon* Card = PC button. In the process, the binary data of a log are converted to a *.csv format, compiled and stored in a file. They can be read and edited in Excel.

See example .....

### Number of files per log session

You can define the number of files you want to be logged per log session by using the Number of files per log session parameter. The maximum number is 1000. When this maximum number is reached, logging will stop (i.e., the log session will be terminated) and output RY will be set to 1. In order to start over, the log files must first be deleted from the storage device.

### Number of data sets per log file

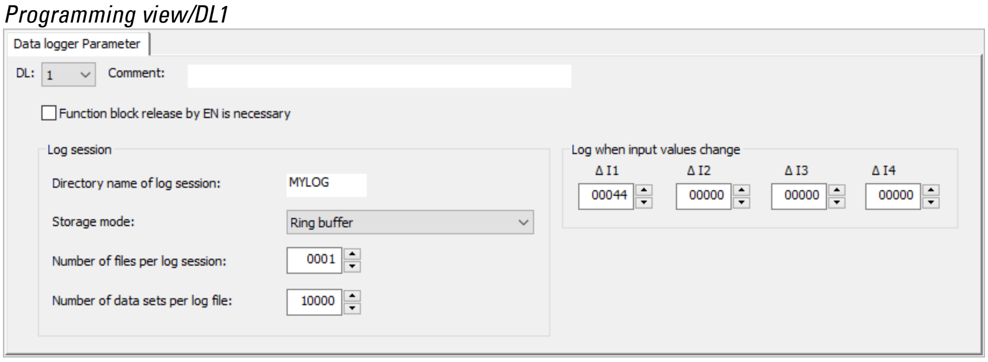
You can use this parameter to define the number of data sets you want to be logged per log file. The maximum number is 60,000. In this mode, logging will always be running, i.e., the log session will not be terminated automatically.

Make sure that the number of data sets you select is only as large as necessary so that the the time for logging will be kept as short as possible.

### Log when input values change

You can use the delta values in this section in order to specify the magnitude of change in the actual value (when compared to the most recently logged value) at which a new save operation should start. You can set a delta (ΔI1 through ΔI4) for the four analog values at DL_I1 through DL_I4. Please note that all data will always be logged with each log operation.

*Programming view/DL1*



#### Other

#### Retention

The function block does not recognize retentive data.

#### Create log files

The data logged with the DL data logger manufacturer function block is stored on the card in a folder. The folder is named as entered in *Programming view/Data logger parameter tab* under the folder name of the log session.

The data in the recording is binary-encoded and cannot be read with standard Windows PC tools. You can instead read it in easySoft 7, where you can view the logs on the card and export them to Excel. To do this, click on *Project menu/Card.../Data logger logs/icon* Card = PC button. In the process, the binary data of a log are converted to a *.csv format, compiled and stored in a file. They can be read and edited in Excel.

→ Please keep in mind that only one folder can be created for each log session, even if the number of files per log session is selected as greater than 1 and several binary files are also stored accordingly.

Task: Any time the device button P1 is pressed, this should be logged. A total of 3 log files with 3 data sets each should be logged. No further logging will occur afterwards.

To do so, follow the steps below:

- Switch to *Programming view*.
- Place a function block DL in the workspace.
- From the catalog, drag a N/O to the function block input DL01_T1.
- In the Contact tab, configure the operand as P device button

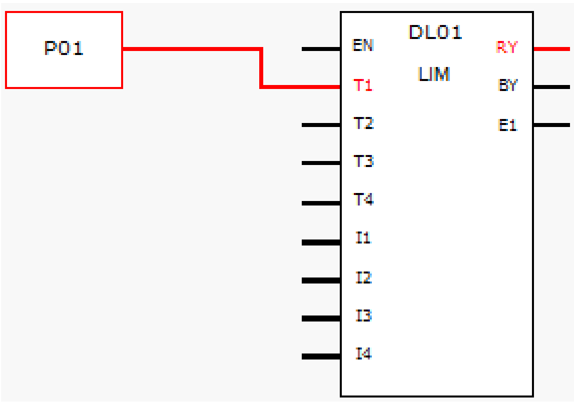



Figure 212: Workspace with function block and device button

- Click on the function block DL and configure as shown in the following illustration.

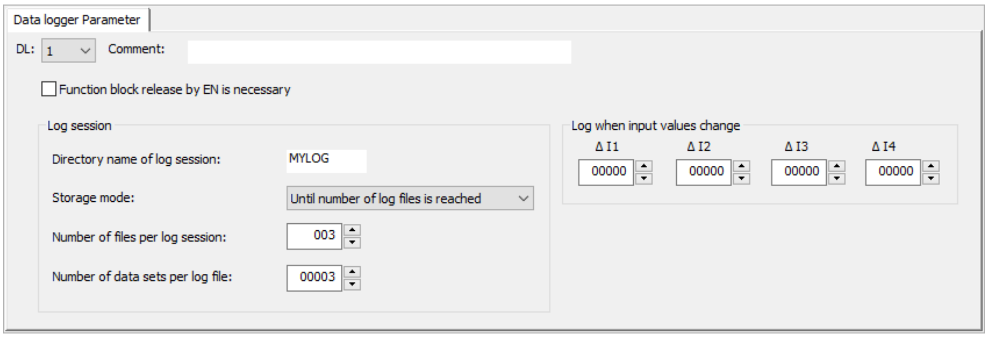



Figure 213: Data logger tab with set parameters for the programming view

- Place a function block DL in the workspace.
- Make sure that this option is enable with the check mark in the *Project view/System settings tab/P buttons*.
- Establish an online connection to the device.
- Save the program on the device.
- Start the program with *Communication view/Program/Communication/*RUN
- Switch the Status Display On using with *Communication menu bar/ Status display on*.
- On the device, press the P button P1 nine times.

The function block output RY=1 displays that logging has ended. The 9 logged data sets are on the SD card. No other data sets are considered.

**Read log files**

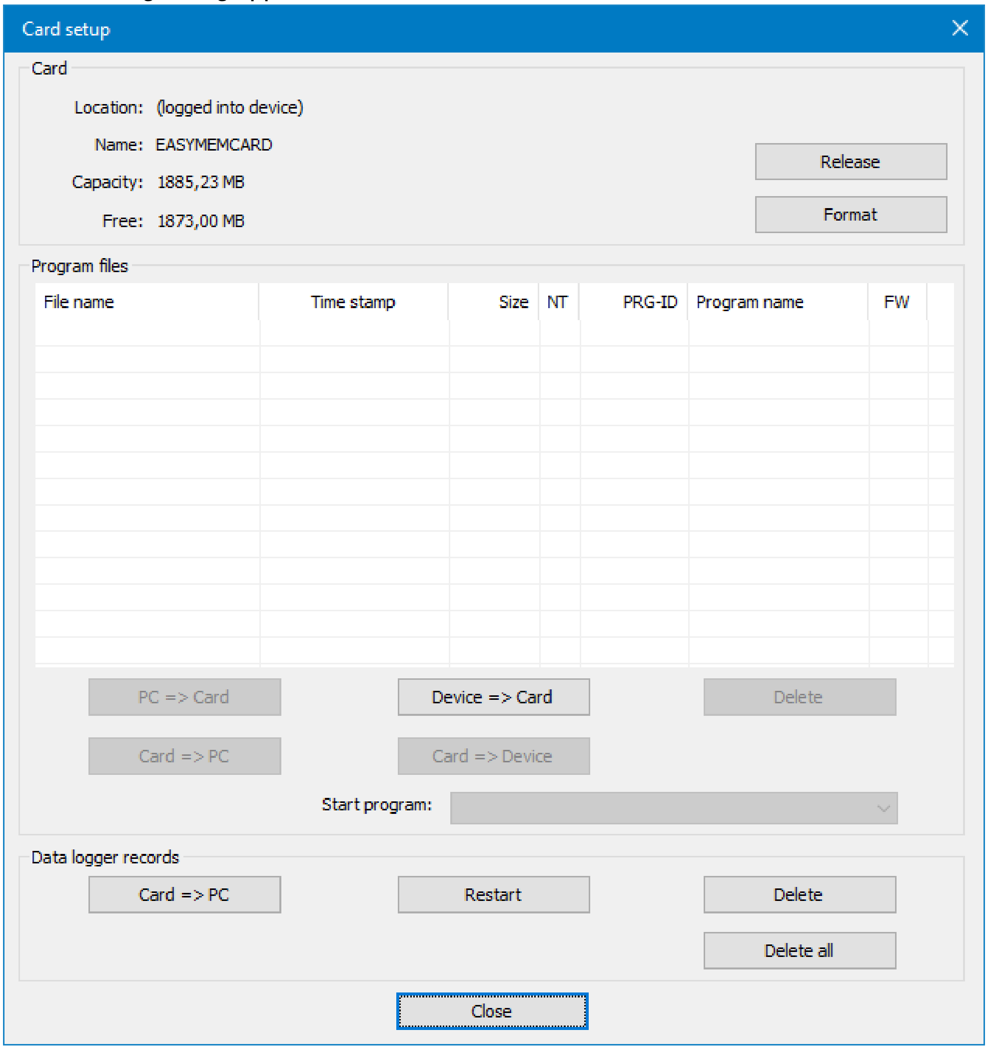
The data logged with the DL data logger manufacturer function block is stored on the card in a folder. The folder is named as entered in *Programming view/Data logger parameter tab* under the folder name of the log session.

The data in the recording is binary-encoded and cannot be read with standard Windows PC tools. You can instead read it in easySoft 7, where you can view the logs on the card and export them to Excel. To do this, click on *Project menu/Card.../Data logger logs/icon* Card = PC button. In the process, the binary data of a log are converted to a *.csv format, compiled and stored in a file. They can be read and edited in Excel.

→ Please keep in mind that only one folder can be created for each log session, even if the number of files per log session is selected as greater than 1 and several binary files are also stored accordingly.

- Make sure that the card is inserted in the log files.
- Establish an online connection to the device with *Communication view/Connection/*Online.
- Make sure that the Status indicator is switched off using *Communication menu bar/Status display OFF*.
- Click on the button on the *Card...* in the area Program/Configuration.

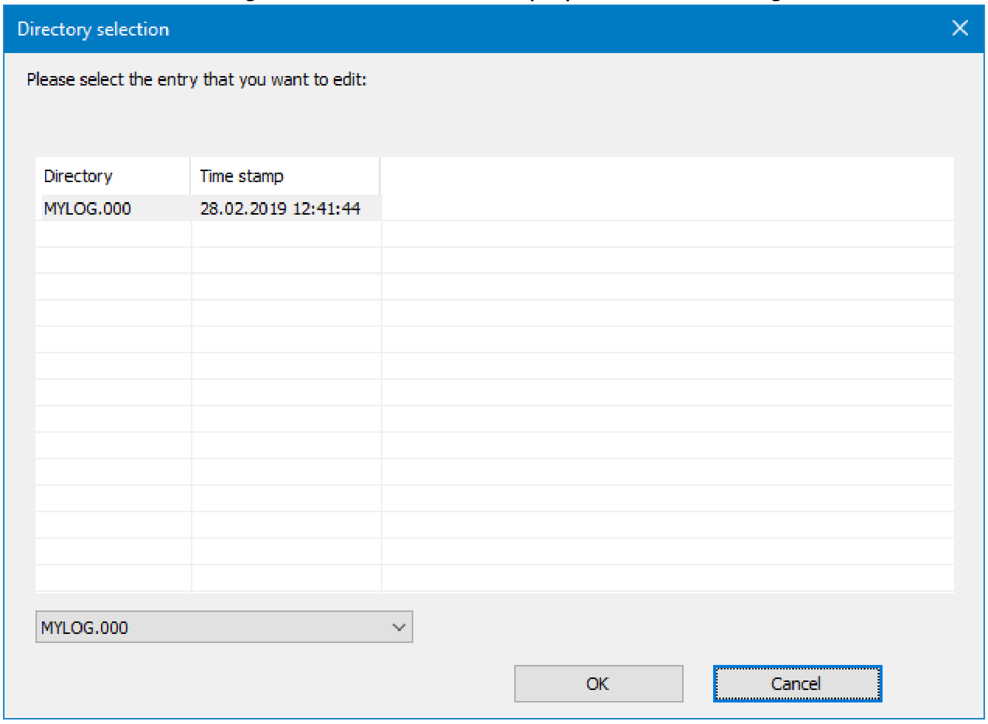
The following dialog appears.




| | |
|---|---|
| Card => PC | Transfers the log files from the memory card to a folder on the computer. |
| Restart | Clicking on this button will create a new directory, on the memory card, for a new data logger file. The new log file will immediately be located in this folder on the memory card. |
| Delete | Deletes the selected log file from the memory card. |
| Delete all | Deletes all the log files found on the memory card. |

▶ Click on Card => PC.
If this button cannot be enabled, make sure that the card is not released in the device.

All of the folders of log files on the card are displayed in the following window.




- Select the folder and confirm with OK.
- In the following Explorer window, choose a storage location and potentially a different file name than the one for MYLOG_000/

The content of all log files in the MYLOG folder are stored in the CSV file MYLOG_000.csv.

If the recording is exported to Excel, the content of several log files is exported into one Excel file. In the dialog, only one file is shown as well, such as MYLOG

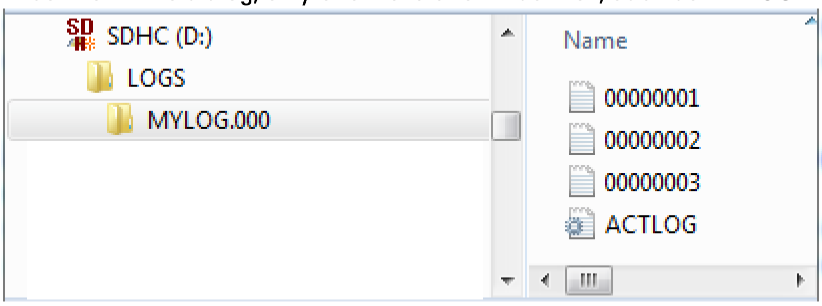



Figure 214: Card content after logging has ended

### Sample log file

The following information will be saved for each data set in the log file:

- Counters
- Date stamp
- Time stamp hh:mm:ss
- Time stamp ms
- States of function block trigger inputs T1 through T4 (DL01T1 through DL01T4 in this example)
- Values at analog function block inputs I1 through I4 (DL01I1 through DL01I4 in this example)

| Counter | Date | Time | Time (ms) | DL01T1 | DL01T2 | DL01T3 | DL01T4 | DL01I1 | DL01I2 | DL01I3 | DL01I4 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2018-07-26 | 12:08:40 | 365 | 1 | 0 | 0 | 0 | 1480 | 2321 | 0 | 0 |
| 1 | 2018-07-26 | 12:08:40 | 968 | 1 | 1 | 0 | 0 | 1480 | 2322 | 0 | 0 |
| 2 | 2018-07-26 | 12:08:42 | 965 | 1 | 0 | 0 | 0 | 1479 | 2321 | 0 | 0 |
| 3 | 2018-07-26 | 12:08:43 | 677 | 0 | 1 | 0 | 0 | 1479 | 2322 | 0 | 0 |
| 4 | 2018-07-26 | 12:08:45 | 579 | 1 | 0 | 0 | 0 | 1480 | 2322 | 0 | 0 |
| 5 | 2018-07-26 | 12:08:46 | 908 | 0 | 1 | 0 | 0 | 1480 | 2321 | 0 | 0 |
| 6 | 2018-07-26 | 12:08:51 | 529 | 1 | 0 | 0 | 0 | 1480 | 2321 | 0 | 0 |
| 7 | 2018-07-26 | 12:08:52 | 332 | 0 | 1 | 0 | 0 | 1477 | 2322 | 0 | 0 |
| 8 | 2018-07-26 | 12:08:53 | 367 | 1 | 0 | 0 | 0 | 1480 | 2319 | 0 | 0 |
| 9 | 2018-07-26 | 12:08:54 | 151 | 0 | 1 | 0 | 0 | 1479 | 2322 | 0 | 0 |
| 10 | 2018-07-26 | 12:08:54 | 729 | 1 | 0 | 0 | 0 | 1480 | 2322 | 0 | 0 |
| 11 | 2018-07-26 | 12:08:55 | 155 | 1 | 1 | 0 | 0 | 1480 | 2322 | 0 | 0 |
| 12 | 2018-07-26 | 12:08:56 | 675 | 1 | 0 | 0 | 0 | 1480 | 2322 | 0 | 0 |
| 13 | 2018-07-26 | 12:08:56 | 677 | 1 | 1 | 0 | 0 | 1480 | 2322 | 0 | 0 |
| 14 | 2018-07-26 | 12:08:57 | 598 | 1 | 0 | 0 | 0 | 1480 | 2321 | 0 | 0 |
| 15 | 2018-07-26 | 12:08:57 | 607 | 1 | 1 | 0 | 0 | 1480 | 2321 | 0 | 0 |
| 16 | 2018-07-26 | 12:08:58 | 493 | 0 | 1 | 0 | 0 | 1480 | 2322 | 0 | 0 |
| 17 | 2018-07-26 | 12:08:58 | 494 | 1 | 1 | 0 | 0 | 1480 | 2322 | 0 | 0 |
| 18 | 2018-07-26 | 12:08:59 | 355 | 1 | 1 | 0 | 0 | 1481 | 2321 | 0 | 0 |
| 19 | 2018-07-26 | 12:09:00 | 198 | 0 | 1 | 0 | 0 | 1481 | 2322 | 0 | 0 |
| 20 | 2018-07-26 | 12:09:00 | 201 | 1 | 1 | 0 | 0 | 1481 | 2322 | 0 | 0 |
| 21 | 2018-07-26 | 12:09:01 | 56 | 0 | 1 | 0 | 0 | 1481 | 2322 | 0 | 0 |
| 22 | 2018-07-26 | 12:09:01 | 60 | 1 | 1 | 0 | 0 | 1481 | 2322 | 0 | 0 |
| 23 | 2018-07-26 | 12:09:02 | 523 | 1 | 0 | 0 | 0 | 1481 | 2322 | 0 | 0 |
| 24 | 2018-07-26 | 12:09:02 | 525 | 1 | 1 | 0 | 0 | 1481 | 2322 | 0 | 0 |
| 25 | 2018-07-26 | 12:09:03 | 445 | 0 | 1 | 0 | 0 | 1480 | 2321 | 0 | 0 |
| 26 | 2018-07-26 | 12:09:03 | 447 | 1 | 1 | 0 | 0 | 1480 | 2321 | 0 | 0 |

There are 26 data sets logged in this log file. The logging for all the data sets was triggered by a rising edge at one of the digital inputs, e.g., data sets 0 through 2 by a DL01T01=1 trigger and data set 3 by a DL01T02=1 trigger. Log files do not include any information regarding the operating mode.

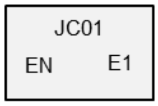
**See also**

→ Section "AL - Alarm function block", page 440
→ Section "BV - Boolean operation", page 444
→ Section "D - Text display", page 448
→ Section "D - Text display editor", page 458
→ Section "JC - Conditional jump", page 486
→ Section "LB - Jump label", page 491
→ Section "MR - MasterReset ", page 493
→ Section " NC - Numerical converter", page 497
→ Section "DL - Data logger", page 475

### 6.1.7.6 JC - Conditional jump

#### General

This function block is only available when using the EDP (easy Device Programming) programming language.
easyE4 base devices provide 32 conditional jump function blocks, JC01 through JC32. You can use JC function blocks to branch off forward to an LB jump label function block within the function block diagram and skip several function blocks while doing so.
The JC function block is used in the circuit diagram, while the LB function block is used in the function block diagram. You can use this approach to structure a program.




#### Operating principle

In order for a jump to be executed, function block input EN must have a state of 1. The jump target is defined using an LB jump label function block.

JC.. and LB.. must always be used in pairs.

When EN = 1, the program jumps forward over one or several function blocks. The next function block to be processed by the program is the first one following the jump label LB.. in the function block diagram.

When EN = 0, the next function block that the program processes is the one that you have added behind JC.. in the function block diagram.

If the associated jump label is not present for an activated jump or is positioned in front of the jump label (backward jump), the program jumps to the end of the function block diagram.

In both cases, the function block output will be set to state E1 = 1.

→ Please note that if there is a timing relay function block that has been started in the circuit diagram, the time will keep counting up even if the timing relay is skipped in the function block diagram with JC..

Display of function blocks in the function block diagram

#### Active function blocks

During simulation, the function block status display shows a red frame around an activated function block that is being processed in the program.
An inactive function block that is not being processed, for example, because the enable coil is set to 0, is shown in a black frame.

The following figure shows the function block JC.. as an example of an active function block. This functions here as the active jump label.

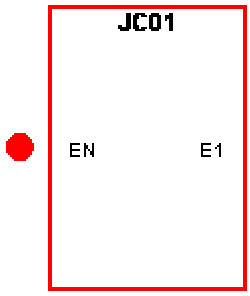



Figure 215: Activated function block in the function block status display

### Skipped function blocks

Function blocks, in the function block diagram change the intensity of their color due to an active » Conditional Jump « JC...

With a skipped function block:

- The red in a frame of an active function block turns to pink and
- the black in the frame of an inactive function block turns to grey.
- The last internal states and values are frozen, e.g. the calculation result of an arithmetic function block, that was calculated before a JC..- function block was activated.

Based on these intermediate states, a function block starts

- its recalculation as soon as it is no longer skipped,
- a bit input can be activated in the circuit diagram and
- a green dot can also be displayed in the simulation,

however, the function block does not change its internal states and values. It consequently does not also change the state of its outputs.

### Positioning in the function block diagram

Drag the conditional jump function block JC.. into the function block diagram and select in the Properties field window the required function block number between 1 and 32 on the Parameters tab.

The conditional jump function block JC.. is now shown at the end of the function block diagram.

Position the conditional jump function block JCxx in the function block diagram in front of the function block(s) to be skipped. To do this, activate the context menu of the JC.. function block and use the Move Function Block function.

Use of the conditional jump function block also requires the placement of a jump label (LABEL:xx) function block in the function block diagram.

### Association in the circuit diagram

Drag the conditional jump function block JC.. onto a coil field of the circuit diagram and in the Properties Field window select the function block number already used in the positioning. Connect the JC..EN coil with an appropriate contact for activation.

→ For greater clarity, position the conditional jump function block JC.. in the circuit diagram if possible directly in front of the function block(s) to be skipped.

If the error output is to be evaluated, position the function block in the circuit diagram again. This time use it as a contact and associate JC..E1 with a suitable Boolean operand.

### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2] | x |
| nNB, nND, nND- NET markers [2] NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |

1) Only on function blocks T, AC
2) Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2] | x |
| SN - Output bit via NET (send)[2] | x |
| N - Net marker bit[2] | x |
| nN - NET marker bit[2] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |

| Operators | Bit inputs |
|---|---|
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Function block outputs

| | Description | Note |
|---|---|---|
| (Bit) | | |
| | | |
| E1 | Error<br>1: if no associated jump label LB is present or is located in front of the jump location (backward jump) | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| Parameter set | Description | Note |
|---|---|---|
| – | | |

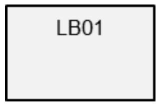
**See also**

→ Section "AL - Alarm function block", page 440
→ Section "BV - Boolean operation", page 444
→ Section "D - Text display", page 448
→ Section "D - Text display editor", page 458
→ Section "DL - Data logger", page 475
→ Section "LB - Jump label", page 491
→ Section "MR - MasterReset ", page 493
→ Section " NC - Numerical converter", page 497
→ Section "ST - Set cycle time", page 503

### 6.1.7.7 LB - Jump label

#### General

This function block is only available when using the EDP (easy Device Programming) programming language.
easyE4base devices provide 32 jump label function blocks, LB01 through LB32.
Within a function block diagram, an LB jump label is used as a jump target for a conditional jump implemented with the JC function block.
JC.. and LB.. must always be used in pairs.




#### Operating principle

The jump label function block does not have to be linked or assigned parameters. It only has to be placed at the appropriate position in the function block diagram.

A corresponding JC.. function block (conditional jump) must exist for every LB.. function block. Conditional jump JC01, for example, is always associated with jump label LB01.

Seen from the corresponding conditional jump function block, the jump label must always be downstream. In other words, it must be closer to the end of the function block diagram.

If the jump label is located upstream of the jump location (backward jump), the program will branch off to the end of the function block diagram. In this case, the conditional jump function block output will be set to state E1 = 1.

### Other

#### Linking and parameter setting

In the function block diagram view, drag the function block to the position you want. Then go to the Jump label parameters tab and select the same function block number you originally assigned to the corresponding conditional jump function block.

You can also move this function block later on. To do so, right-click on the function block you want to move and then select the *Move...* option.

#### See also

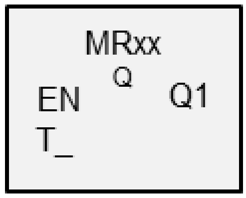
→ Section "AL - Alarm function block", page 440
→ Section "BV - Boolean operation", page 444
→ Section "D - Text display", page 448
→ Section "D - Text display editor", page 458
→ Section "DL - Data logger", page 475
→ Section "JC - Conditional jump", page 486
→ Section "MR - MasterReset ", page 493
→ Section " NC - Numerical converter", page 497
→ Section "ST - Set cycle time", page 503

### 6.1.7.8 MR - MasterReset

#### General

easyE4 base devices provide 32 master reset function blocks MR01 to MR32.

These function blocks can be used to set the markers and all device outputs to a state of 0.




#### Operating principle

Depending on the operating mode set, it is possible to reset either the outputs only, the markers only or both.

→ To ensure that all data ranges are reliably cleared, the master reset function block must be the last function block executed in your program. Otherwise subsequent function blocks may overwrite the data ranges again.

#### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |
| T_ | Trigger: The reset will be carried out when there is a rising edge. | |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |

1) Only on function blocks T, AC
2) Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating mode

| | Description | Note |
|---|---|---|
| **Operating mode** | | |
| Q = Reset outputs | Device outputs Q.., and QA..., as well as outputs LE.., SN.., will be reset to a state of 0. | Default settings |
| M = Reset marker | The following markers are reset to 0:<br>• Marker range MD01...MD256<br>• ND01..ND16<br>• Internal markers of existing function blocks UF, IC, IE and IT | |
| ALL = Reset both | Has an effect on the operands set at Q and M. | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | 1: If input T_ has a state of 1. | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers [2)]<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

### Retention

The function block does not recognize retentive data.

### Example of a master reset function block with the programming method EDP

```
I 05--------------------------[ MR07T_
```

Figure 216: Wiring the function block coils

The trigger coil is connected to a device input.

```
MR07Q1------------------------[ S M42
```

Figure 217: Wiring of the function block contact

The message of the function block is sent to a marker.

### Example of a master reset function block configuration on a device display

When using the function block in the circuit diagram for the first time, use OK to automatically enter the display of function blocks on the device display, as shown in the following figure.

```
MR16 Q +
```

Figure 218: Parameters on the device display

Enter the function block settings here. The display contains the following elements:

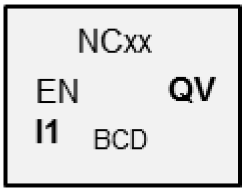
| | |
|---|---|
| MR16 master reset | Function block: Master reset, number 16 |
| Q | Operating mode: Reset outputs |
| + | Parameter set can be called via the PARAMETERS menu |

**See also**

→ Section "AL - Alarm function block", page 440
→ Section "BV - Boolean operation", page 444
→ Section "D - Text display", page 448
→ Section "D - Text display editor", page 458
→ Section "DL - Data logger", page 475
→ Section "JC - Conditional jump", page 486
→ Section "LB - Jump label", page 491
→ Section " NC - Numerical converter", page 497
→ Section "ST - Set cycle time", page 503

### 6.1.7.9 NC - Numerical converter

#### General




easyE4base devices provide 32 numerical conversion function blocks NC01...NC32.
A decimal number can be represented either as being binary-coded or BCD-coded. Depending on the operating mode you select, this function block will convert BCD-coded numbers to binary-coded numbers (BCD mode) or vice verse, i.e., binary-coded numbers to BCD-coded numbers (BIN mode).

#### Operating principle

EN=1 enables the function block so that the number conversion will be carried out every cycle. The following applies when using LD, FBD, ST: As soon as there is a changed value at I1, the new conversion value will appear at output QV. When using EDP, the converted value will be provided until the next cycle.

The maximum data size that can be connected to the inputs/outputs is a double word (32 bits). A BCD-coded number requires four bits (a nibble). This means that the biggest BCD-coded numbers that can be converted are 7-digit BCD-coded numbers, since the most significant nibble is used for the sign.

0000 means +
1111 means –

EN=0 initiates a reset in which the output QV is reset to 0.

#### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |
| **(DWord)** | | |
| I1 | Operand to be converted | Integer value range,<br>decimal not all the way due to BCD limitation<br>BCD: -9 999 999 … +9 999 999<br>Decimal: -161 061 273 … +161 061 273 |

### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating mode

### BCD mode

The BCD value at I1 will be converted to a binary value and output at output QV. The binary value is displayed as a decimal value.

### BIN mode

The binary value present at input I1 is converted to a BCD value and supplied at the output QV. The binary value is displayed as a decimal value.

| | Description | Note |
|---|---|---|
| BCD | Converts a BCD value to a binary value. | |
| BIN | Converts a binary value to a BCD value. | |

### Function block outputs

| | Description | Note |
|---|---|---|
| (DWord) | | |
| QV | Supplies the converted value. | Integer value range<br>Decimal: -161 061 273…+161 061 273<br>BCD: -9 999 999… +9 999 999 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| Configuration/time range | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation possible | | |

### Other

#### Retention

The function block does not recognize retentive data.

#### Example for BIN operating mode

For simulation purposes in easySoft 7, function block input I1 can be connected to a marker double word instead of to a binary source. The value of this marker double word can be entered in hexadecimal or decimal format. It will always be interpreted as binary at function block input I1.

| MD value | | I1 | | | QV |
|---|---|---|---|---|---|
| (dec) | (hex) | BIN | NC | BCD | (dec) |
| 9 | 9 | 0000 1001 | | 0000 1001 | 9 |
| 23 | 17 | 0001 0111 | | 0010 0011 | 35 |
| 37 | 25 | 0010 0101 | | 0011 0111 | 55 |
| 9 999 999 | 00 989 67F | 0000 0000<br>1001 1000<br>1001 0110<br>0111 1111 | | 0000 1001<br>1001 1001<br>1001 1001<br>1001 1001 | 161 061 273 |
| -9 999 999 | FF 676 981 | 1111 1111<br>0110 0111<br>0110 1001<br>1000 0001 | | 1111 0110<br>0110 0110<br>0110 0110<br>0110 0111 | -161 061 273 |
| | -10 000 000 | 1001 0000<br>0000 0000<br>0000 0000<br>0000 0000 | Value range exceeded | 1001 1001<br>1001 1001<br>1001 1001<br>1001 1001 | -161 061 273 |

The most significant nibble determines the sign.
For negative numbers, the two's complement
will be calculated.

Since each decimal value is represented with four bytes or eight nibbles and each nibble in the BCD code can assume a value of 9, the largest number that can be represented is 9999999. The smallest number that can be represented is -9,999,999.

However, since a BCD source cannot represent negative numbers, a negative numeric conversion at QV is simply a theoretical case.

→ Values greater than 9999999 are output as 161061273.
Values less than –9999999 are output as –161061273.
The working range of the function block has been exceeded.

**Example for BCD operating mode**

For simulation purposes in easySoft 7, function block input I1 can be connected to a marker double word instead of to a BCD source. The value of this marker double word can be entered in hexadecimal or decimal format. It will always be interpreted as BCD at function block input I1.

| MD value (dec) | (hex) | I1 BCD | NC | BIN | QV (dec) |
|---|---|---|---|---|---|
| 9 | 9 | 0000 1001 | | 0000 1001 | 9 |
| 23 | 17 | 0001 0111 | | 0001 0001 | 17 |
| 37 | 25 | 0010 0101 | | 0001 1001 | 25 |
| 18 585 | 4 899 | 0000 0000<br>0000 0000<br>0100 1000<br>1001 1001 | | 0000 0000<br>0000 0000<br>0001 0011<br>0010 0011 | 4 899 |
| 161 061 273 | 9 999 999 | 0000 1001<br>1001 1001<br>1001 1001<br>1001 1001 | | 0000 0000<br>1001 1000<br>1001 0110<br>0111 1111 | 9 999 999 |
| -161 061 273 | F6 666 667 | 1111 0110<br>1001 1001<br>1001 1001<br>1001 1001 | | 1111 1111<br>0110 0111<br>0110 1001<br>1000 0001 | -9 999 999 |
| 161 061 274 | | 1001 1001<br>1001 1001<br>1001 1001<br>1001 1001 | Value range exceeded | 1001 1001<br>1001 1001<br>1001 1001<br>1001 1001 | 9 999 999 |

→ The most significant nibble determines the sign. For negative numbers, the two's complement will be calculated.

→ Since each decimal value is represented with four bytes or eight nibbles and each nibble in the BCD code can assume a value of 9, the largest number that can be represented is 9999999. The smallest number that can be represented is -9,999,999.

However, since a BCD source cannot five a negative number at I1, a negative numeric conversion at QV is simply a theoretical case.

→ Values greater than 161061273 are output as 9999999.
Values less than –161061273 are output as –9999999.
The working range of the function block has been exceeded.

**Example of a numerical converter function block when using the EDP programming language**

Function block input NC..EN is connected directly to device terminal I5

```
I 05-----------------------------[ NC01EN
```

Figure 219: Wiring the function block coils

```
NC02 BCD +
>I1
QV>
```

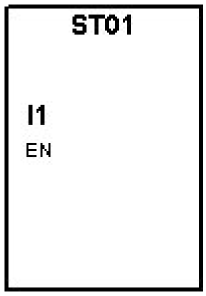
Figure 220: Setting of the parameters

**See also**

→ Section "AL - Alarm function block", page 440
→ Section "BV - Boolean operation", page 444
→ Section "D - Text display", page 448
→ Section "D - Text display editor", page 458
→ Section "DL - Data logger", page 475
→ Section "JC - Conditional jump", page 486
→ Section "LB - Jump label", page 491
→ Section "MR - MasterReset ", page 493
→ Section "ST - Set cycle time", page 503

### 6.1.7.10 ST - Set cycle time

#### General




easyE4 base devices provide 32 set cycle time function blocks, ST01 through ST32.

This function block allows a set cycle time to be defined.
This cycle time is adjusted automatically if the maximum cycle time used in the program is less than this specified value.
The maximum possible set cycle time is 1000 ms.

The set cycle time cannot be implemented if the cycle time of the program is longer than it.

#### The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| **(DWord)** | | |
| I1 | Required cycle time in ms | Integer value range: 0…1000 |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant 1) | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers 2) | x |
| nNB, nND, nND- NET markers 2) NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display + Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Simulation NOT possible | | |

### Other

### Retention

The function block does not recognize retentive data.

### Application example

A program consisting of the bit circuit diagram and function block generates a mean cycle time of approx. 12 ms. Setting the set cycle time to 30 ms will ensure that the cycle times are kept constant at this value.

**See also**

→ Section "AL - Alarm function block", page 440
→ Section "BV - Boolean operation", page 444
→ Section "D - Text display", page 448
→ Section "D - Text display editor", page 458
→ Section "DL - Data logger", page 475
→ Section "JC - Conditional jump", page 486
→ Section "LB - Jump label", page 491
→ Section "MR - MasterReset ", page 493
→ Section " NC - Numerical converter", page 497

# 6.2 interrupt function blocks

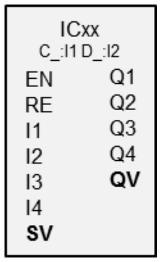
## 6.2.1 IC - Counter-controlled interrupt

Only possible with easySoft 7.

### 6.2.1.1 General

easyE4 base devices provide 8 counter-controlled interrupt function blocks, IC01 through IC08. This does not apply to the EDP programming language.
easyE4 makes it possible to quickly respond to various events. This makes it possible, for instance, to switch outputs on or off outside of the main program's routine. Only bit operators are allowed within an interrupt program.
The following events can trigger an interrupt:

- Reaching counter reference values, two-channel, device inputs I1 through I8, function blocks IC1 through IC8
- Frequency measurement, reference value exceeded or fallen below, device inputs I1 through I8, function blocks IC1 through IC8




**Execution time for an interrupt**
The time between the moment the event is detected and the moment there is a response at a device output is < 1 ms. To this end, the base device's QP physical output must be set.
If multiple interrupts are executed simultaneously, the times add up.

> *NOTICE*
> Use each device input from I1 to I8 only once in each interrupt function block. Otherwise, an error message will be output during the plausibility check and it will not be possible to load the program onto the device.

→ In total, no more than 8 interrupt sources are allowed to be processed in a single program. The valid interrupt sources are the IC, IE, IT interrupt function blocks and the CF, CH, and CI high-speed counters that are directly connected to the device inputs.

→ If there are multiple interrupt requests present simultaneously, the first detected interrupt program will be executed, after which the remaining ones will be executed based on the corresponding order.

→ While the interrupt program is being processed, any other incoming interrupts at the function block inputs of the same instance will not be detected.

### 6.2.1.2 Operating principle

A reference value is set at function block input SV. Depending on the operating mode, the function block will be assigned one or two of device inputs I1 through I8 in the corresponding parameters (at least one of them will be set as a counter input in the parameters). If the counter input reaches the reference value, the interrupt will be triggered. The system will switch from the main program to the interrupt program and the latter will be processed.

**Interaction between main program and interrupt program**

The states of function block inputs IC_I1 through IC_04 are passed to the interrupt program, where they can be processed further as I01 through I04.

Function block outputs IC_Q1 through IC_Q4 can be set from the interrupt program. The corresponding interrupt program outputs are Q01 through Q04.

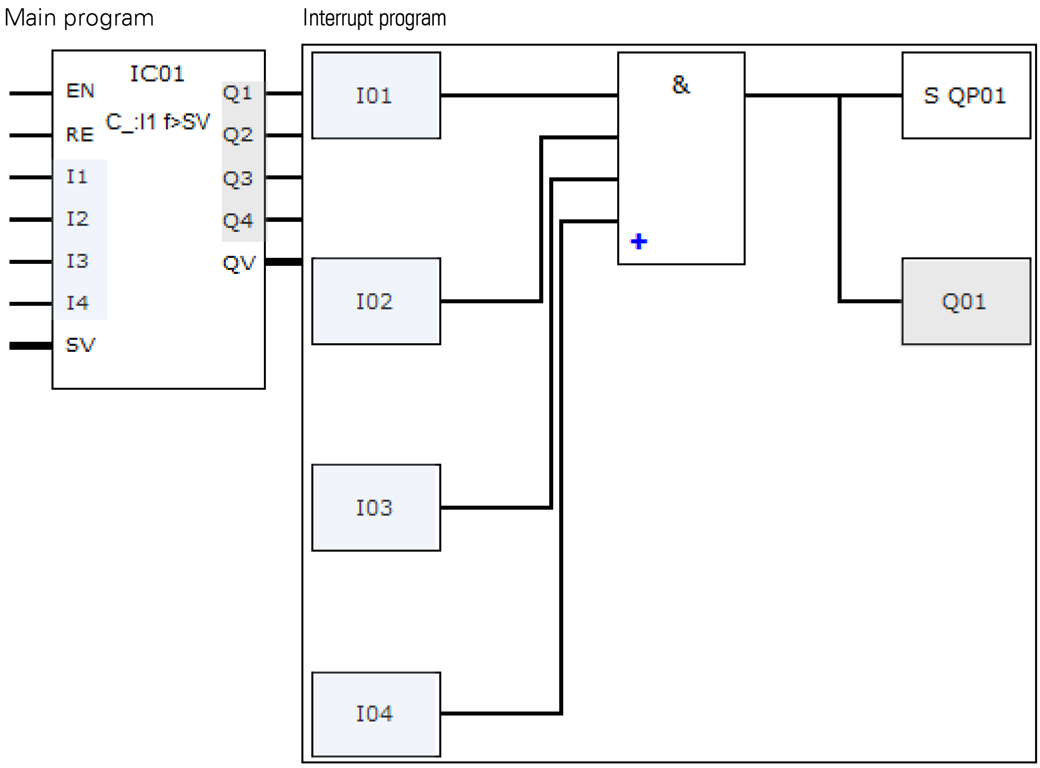



Figure 221: Input and output states being passed between the main program and interrupt program

If an output is defined as a physical output on base device in the parameters for the interrupt program, the output will be assigned an identifier of QP01 – QP04 and will act directly on device output Q1 – Q4.

The function block has its own 32-marker-bit marker range for processing the interrupt program.

**Available functions within an interrupt program**

Interrupt programs are not available when using the EDP programming language.

| Action | KOP | FBS | ST |
|---|---|---|---|
| New network | √ | √ | √ |
| Input/output inverter | √ | √ | √ |
| Contacts | Make, Break, Constant 1, Constant 0 | | |
| Coils | Contactor, Negated contactor, Set, Reset | | |
| Jump functions | Jump if 1, Jump if 0, Return if 1, Return if 0 | | |
| Logic gates | AND, AND NOT, OR, OR NOT, XOR, XNOR | | |
| Conditional statement | – | – | √ |
| Simple alternative | – | – | √ |
| Multiple alternatives | – | – | √ |

### 6.2.1.3 The function block and its parameters

**Function block inputs**

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary parameter must first be enabled |
| RE | 1: Sets the actual counter value to zero | |
| I1 | The states of the bit inputs from the main program will be provided to the interrupt program | |
| I2 | | |
| I3 | | |
| I4 | | |
| **(DWord)** | | |
| SV | Setpoint | Integer value range: -2,147,483,648 to +2,147,483,647 |

**Assigning operands**

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |

| Operators | Value inputs |
|---|---|
| nNB, nND, nND- NET markers 2)<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET2) | x |
| SN - Output bit via NET (send)2) | x |
| N - Net marker bit2) | x |
| nN - NET marker bit2) NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating modes

| (Bit) | Description | Note |
|---|---|---|
| Counter with external direction control | Pulse at device input I1 – I8, which is defined as a counter input in the parameters.<br><br>Duration signal at device input I1 – I8, which specifies the counting direction.<br>0: up counting<br>1: down counting | Maximum frequency 5kHz |
| Counter with 2 counter inputs | Pulse at device input I1 – I8 counts up.<br>Pulse at device input I1 – I8 counts down. | |
| incremental counter | Double evaluation<br>With automatic up/down counting direction detection, two counter inputs<br>I1…I8, counter input channel A, pulse<br>I1…I8, counter input channel B, pulse | |

| (Bit) | Description | Note |
|---|---|---|
| | When there is a complete channel A and B period (e.g., first channel A edge to next channel A edge), the value at IC..QV is incremented or decremented by 2 depending on the counting direction. | |
| Frequency counter; f > SV | I1…I8, timeout of the frequency reference<br>Measuring interval 0.01s, 500 Hz - 5000 Hz<br>Measuring interval 0.1 s, 50 Hz - 5000 Hz<br>Measuring interval 1.0 s 5 Hz - 5000 Hz | |
| Frequency counter; f < SV | I1 – I8, Frequency reference fallen below<br>Measuring interval 0.01s, 500 Hz - 5000 Hz<br>Measuring interval 0.1 s, 50 Hz - 5000 Hz<br>Measuring interval 1.0 s 5 Hz - 5000 Hz | |

→ In the case of pulse counter with external direction control, device inputs I1 through I4 must be used as pulse inputs and device inputs I5 through I8 must be used as direction inputs.
In the case of counters with 2 counter inputs, I1 through I4 should be used with first priority.
In the case of incremental encounters, I1 through I4 should be used with first priority.

→ In the case of incremental counters, channel A and channel B must deliver pulses with an offset of 90°.

IC function block with incremental counter operating mode, up or down count; Double evaluation

**Function block outputs**

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | Bit output used to provide operand states from the interrupt program to the main program. | |
| Q2 | | |
| Q3 | | |
| Q4 | | |
| **(DWord)** | | |
| QV | Current count | Integer value range:<br>-2,147,483,648 to +2,147,483,647 |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Edit interrupt routine | Clicking on the button will open the interrupt routine | |
| Simulation possible | | |

## 6.2.1.4 Other

### Retention

The function block does not recognize retentive data.

The interrupt program does not have any retentive data.

#### Monitoring the interrupt load

In total, no more than 8 interrupt sources are allowed to be processed in a single program. The valid interrupt sources are the IC, IE, IT interrupt function blocks and the CF, CH, and CI high-speed counters that are directly connected to the device inputs. For more information, please refer to → "CF - Frequency counter", page 287, → "CH - High-speed counter", page 293 , → "CI - Incremental Counter", page 299 as well.

For function blocks IE01 – IE08 and IC01 – IC08, device inputs I01 – I08 can be assigned freely.

For function blocks IT01 – IT08, an interrupt that is not yet in use is assigned in easySoft 7. The interrupt sources used by the CF, CH, and CI high-speed counters are also considered to be in use within this context.

Each device input and each interrupt source can only be used once.

Exceptions:

- For CI01, the instance of I02 can be used by an IT interrupt function block.
- For CI02, the instance of I04 can be used by an IT interrupt function block.
- For each interrupt function block IC, the instance of the second input of an IT function block can be used if the Counter with 2 counter inputs mode has not been selected.

These exceptions are taken into account by the plausibility check and by the program compilation routine in easySoft 7. The maximum number of 8 interrupts is also taken into account within this context.

| | **Device inputs** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **I01** | **I02** | **I03** | **I04** | **I05** | **I06** | **I07** | **I08** |
| **Interrupt source** | | | | | | | | |
| CF01 frequency counter | x | | | | | | | |
| CF02 frequency counter | | x | | | | | | |
| CF03 frequency counter | | | x | | | | | |
| CF04 frequency counter | | | | x | | | | |
| CH01 high-speed meter | x | | | | | | | |
| CH02 high-speed meter | | x | | | | | | |
| CH03 high-speed meter | | | x | | | | | |
| CH04 high-speed meter | | | | x | | | | |
| CI01 incremental counter | x | x | | | | | | |

| | Device inputs | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | I01 | I02 | I03 | I04 | I05 | I06 | I07 | I08 |
| CI02 incremental counter | | | x | x | | | | |
| IE01…IE08 | One input, I01 – I08 can be assigned freely (max. 8, none can be assigned more than once) | | | | | | | |
| IC01 – IC08 | Two inputs, I01 – I08 can be assigned freely (max. 8, none can be assigned more than once) | | | | | | | |
| IT01…IT08 | Automatically assigns the user interrupts 1 to 8 that are still available (only for instances of I01 – I08 that are not in use by other function blocks) | | | | | | | |

The time between the moment the triggering signal is detected and the moment there is a response at an output is < 1 ms. If multiple interrupts are executed simultaneously, the times add up.

**Measuring the interrupt load**

The runtime for each interrupt source is measured in µs. All measured times are added over a period of 100 ms. After each 100 ms, the total of all times is evaluated and the time measurement is cleared. If the interrupts used up more than 50% of the computing time, the application will be stopped.

The <System_CPU_overload> diagnostic message will be generated and ID19 will be set to 1.

For more information on how diagnostic messages can be called and processed, please refer to

**Available fixes for high interrupt loads**

If the interrupt load becomes too heavy, the following steps can be taken to reduce it:

- Reduce the number of function blocks
- Keep the interrupt routine as short as possible
- Reduce frequencies when using counters

**Example of a pulse counter with external direction control in easySoft 7**

Device input I1: Counter input C_

Device input I5: Counting direction D_

If the setpoint of <1750> is reached at device input I1, the system will jump to the interrupt program. Inside the interrupt program, QP04 will be used to set device output Q4 to 1 directly. Q01 will be used to set function block output Q1 to 1. The system will then jump back to the main program.

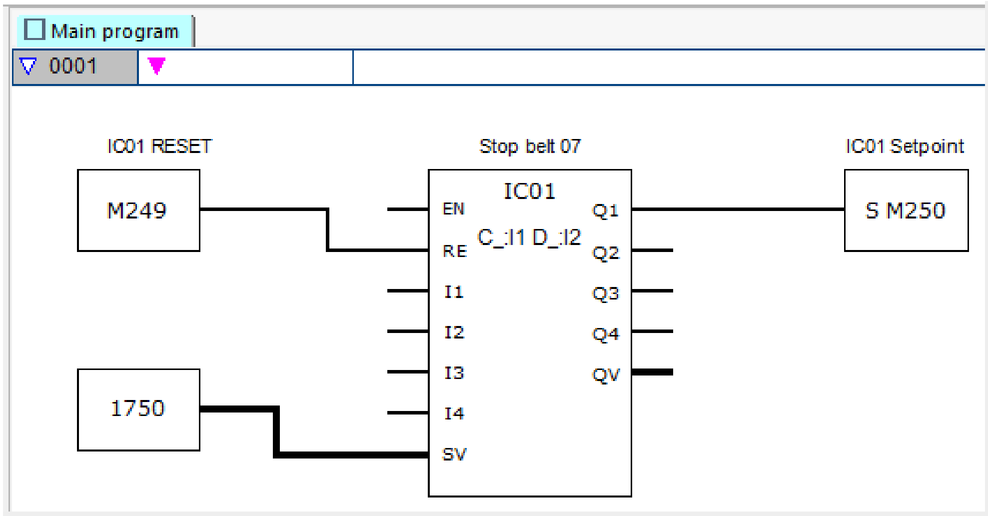



Figure 222: easySoft 7 Main program Pulse counter with external direction

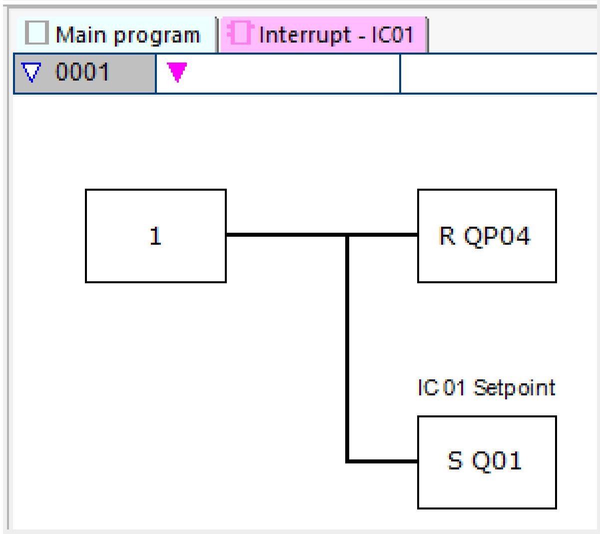



Figure 223: easySoft 7 Interrupt program Pulse counter with external direction

**Example with two counter inputs in easySoft 7**

Device input I1: Up counter input C+

Device input I2: Down counter input C-

If the actual value reaches the function block's setpoint, the interrupt will be triggered. The interrupt program will then set device output Q1 back to 0. In addition, Q01=1 will be used to set function block output Q1 to 1 and main program marker M250 to 1. This way, the container status will be signaled.

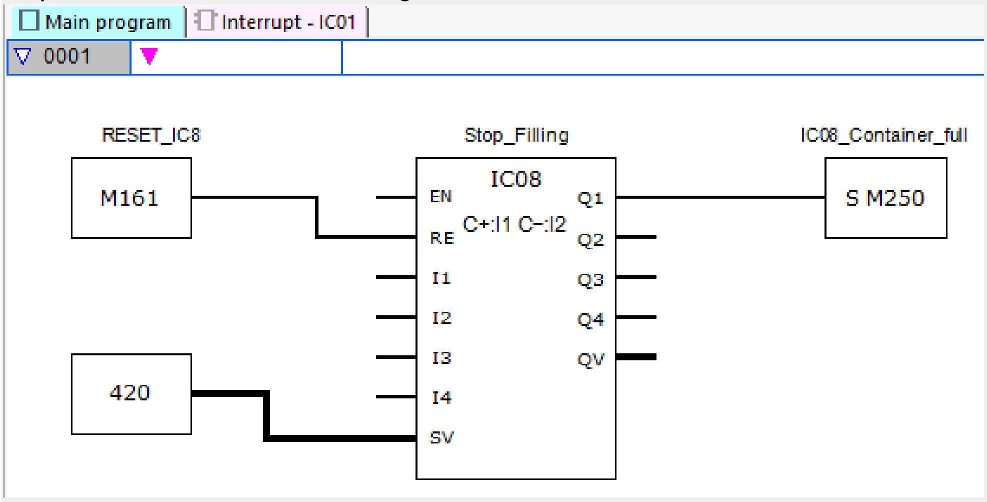



Figure 224: easySoft 7 Main program, two counter inputs

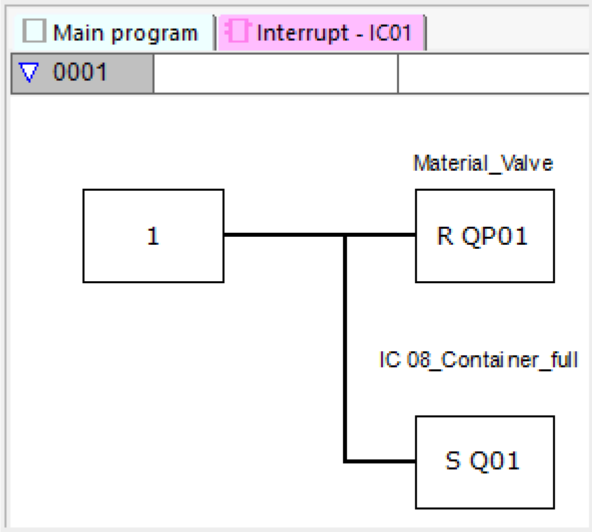



Figure 225: easySoft 7 Interrupt program, two counter inputs

### Example of an incremental counter in easySoft 7

Palletizing system with home positioning

The gripper must set the material down every time the target position with marker word MW512 is reached in the up direction. When Q01 is set in the interrupt program, marker M511 is set in the main program so that it can be used to move back to the home position.

Device input I3: channel A

Device input I4: channel B

The target position is specified on marker MW512.

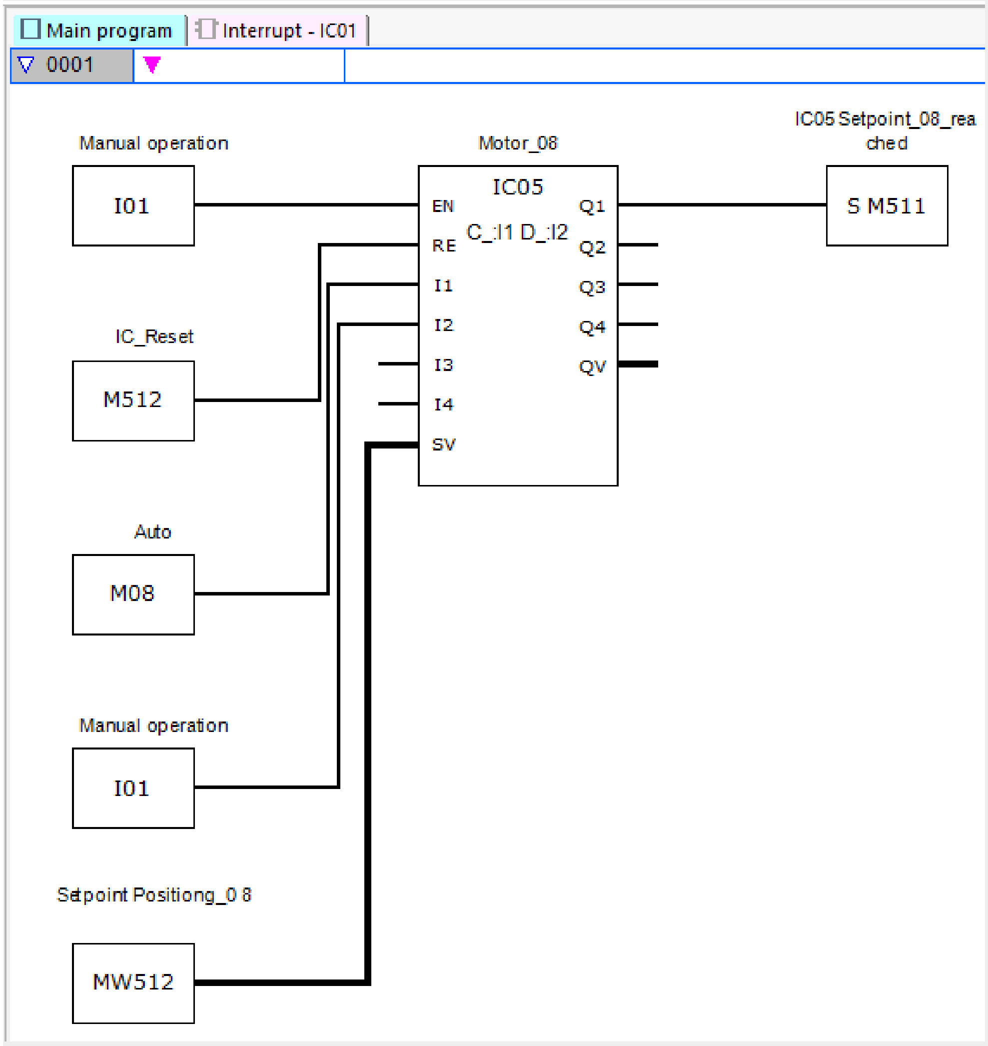



Figure 226: easySoft 7 Main program Incremental counter

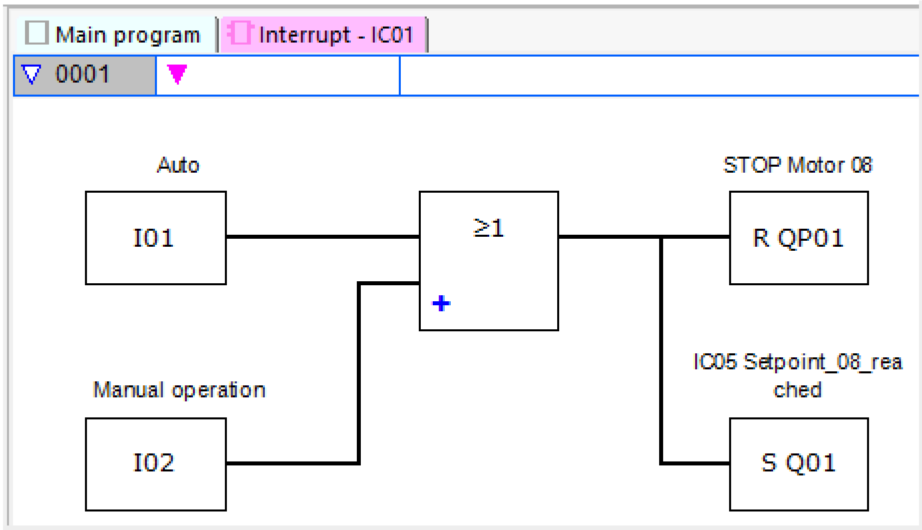



Figure 227: easySoft 7 Interrupt program Incremental counter

### Example frequency measurement in easySoft 7

Device input I1 is a measuring input.

If the frequency at device input I1 reaches the frequency of 1030 Hz, the interrupt is triggered. The interrupt program uses QP02 to reset device output Q2 and SQ01 to set marker M31 at function block output Q1. Marker 31 signals that the frequency has been reached.

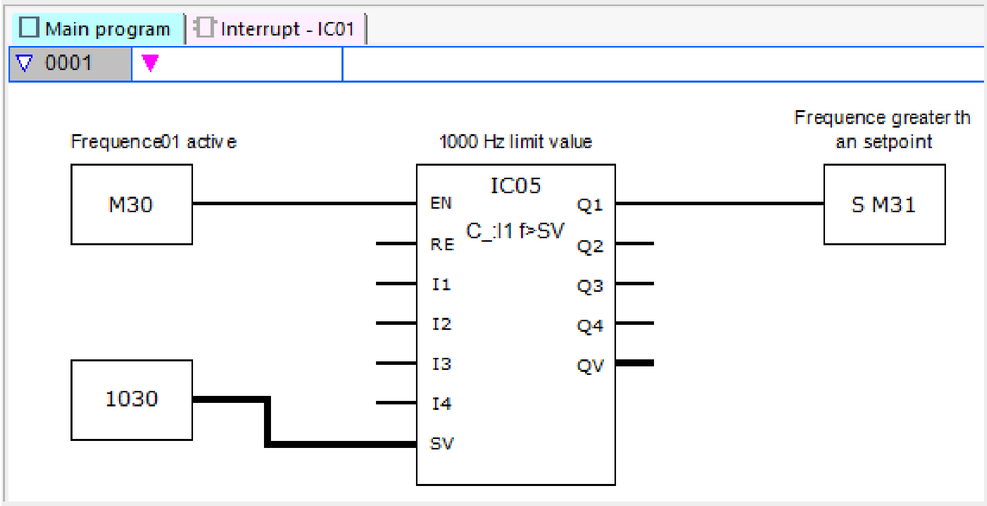



Figure 228: easySoft 7 Main program Frequency measurement

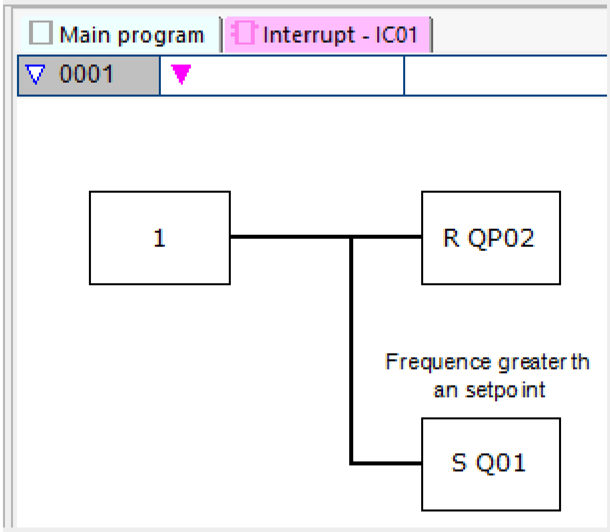



Figure 229: easySoft 7 Interrupt program Frequency measurement

### See also

→ Section "IE - Edge-controlled interrupt", page 519
→ Section "IT - Time-controlled interrupt function block", page 525

### 6.2.2 IE - Edge-controlled interrupt

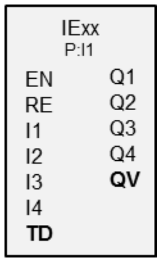
Only possible with easySoft 7.

#### 6.2.2.1 General

easyE4 base devices provide 8 edge-controlled interrupt function blocks, IE01 through IE08. This does not apply to the EDP programming language. easyE4 makes it possible to quickly respond to various events. This makes it possible, for instance, to switch outputs on or off outside of the main program's routine. Only bit operators are allowed within an interrupt program.
The following events can trigger an interrupt:

- Rising, falling, rising, and falling edges at device inputs I1 through I8, function block IE01 through IE08.

**Execution time for an interrupt**
The time between the moment the event is detected and the moment there is a response at a device output is < 1 ms. To this end, the base device's QP physical output must be set.
If multiple interrupts are executed simultaneously, the times add up.




*NOTICE*
Use each device input from I1 to I8 only once in each interrupt function block. Otherwise, an error message will be output during the plausibility check and it will not be possible to load the program onto the device.

→ In total, no more than 8 interrupt sources are allowed to be processed in a single program. The valid interrupt sources are the IC, IE, IT interrupt function blocks and the CF, CH, and CI high-speed counters that are directly connected to the device inputs.

→ If there are multiple interrupt requests present simultaneously, the first detected interrupt program will be executed, after which the remaining ones will be executed based on the corresponding order.

→ While the interrupt program is being processed and during a configured delay, any other incoming interrupts at the function block inputs of the same instance will not be detected.

### 6.2.2.2 Operating principle

You can use function block input TD to set a reference value for a delay you want. You will need to assign one of device inputs I1 through I8 to the function block as an interrupt source. The first edge at the assigned device input will trigger the interrupt directly if you did not configure a delay. Otherwise, the interrupt will be triggered after the configured delay elapses. The system will switch from the main program to the interrupt program and the latter will be processed.

**Interaction between main program and interrupt program**

The states of function block inputs IE_I1 through IE_04 are passed to the interrupt program, where they can be processed further as I01 through I04.

Function block outputs IE_Q1 through IE_Q4 can be set from the interrupt program. The corresponding interrupt program outputs are Q01 through Q04.

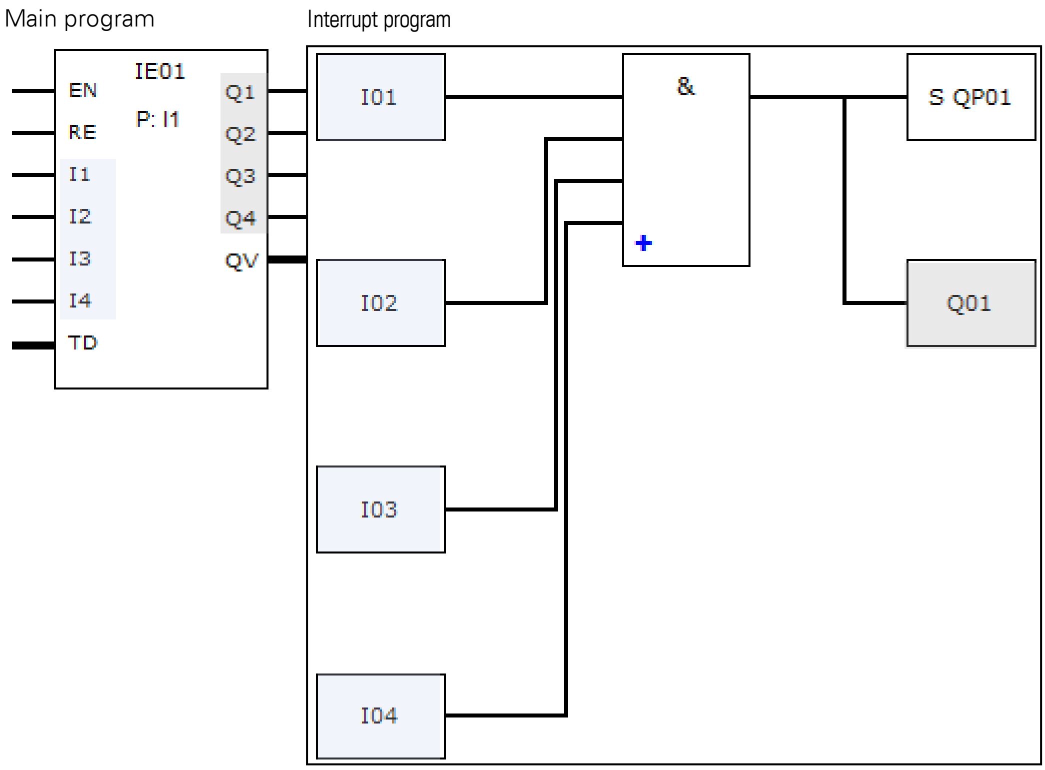



Figure 230: Input and output states being passed between the main program and interrupt program

If an output is defined as a physical output on base device in the parameters for the interrupt program, the output will be assigned an identifier of QP01 – QP04 and will act directly on device output Q1 – Q4.

The function block has its own 32-marker-bit marker range for processing the interrupt program.

**Available functions within an interrupt program**

Interrupt programs are not available when using the EDP programming language.

| Action | KOP | FBS | ST |
|---|---|---|---|
| New network | √ | √ | √ |
| Input/output inverter | √ | √ | √ |
| Contacts | Make, Break, Constant 1, Constant 0 | | |
| Coils | Contactor, Negated contactor, Set, Reset | | |
| Jump functions | Jump if 1, Jump if 0, Return if 1, Return if 0 | | |
| Logic gates | AND, AND NOT, OR, OR NOT, XOR, XNOR | | |
| Conditional statement | – | – | √ |
| Simple alternative | – | – | √ |
| Multiple alternatives | – | – | √ |

### 6.2.2.3 The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | The checkbox for the ☑ Function block release by EN is necessary<br>parameter must first be enabled |
| RE | 1: Sets the function block's internal counter for the delay back to the value at TD. | |
| I1 | Bit input used to provide operand states from the main program to the interrupt program | |
| I2 | | |
| I3 | | |
| I4 | | |
| **(DWord)** | | |
| TD | Delay until the interrupt program is started | Value range: 20 ms …999 990 ms<br>Resolution: 10 ms |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating modes

| | Description | Note |
|---|---|---|
| Rising edge | Rising edge: Runs the interrupt program once after delay TD. | |
| Falling edge | Falling edge: Runs the interrupt program once after delay TD. | |
| Both edges | Rising edge and falling edge at input: Runs the interrupt program after delay TD each time. | |

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | Bit output used to provide operand states from the interrupt program to the main program. | |
| Q2 | | |
| Q3 | | |
| Q4 | | |
| **(DWord)** | | |
| QV | Actual elapsed delay (TD) time | |

### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers [2)]<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Parameter set

| | Description | Note |
|---|---|---|
| ☑ Function block release by EN is necessary | If this checkbox is enabled, the state of function block input EN will be evaluated. If the checkbox is disabled instead, the function block will be enabled and function block input EN will not do anything. | This parameter ensures that when existing programs are copied, the functionality of the function blocks that are carried over will be retained. The parameter will be automatically set to 0 or 1 depending on the function block. |
| Parameter display + Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Interrupt source | Used to select device inputs I1 through I8 as a trigger for the interrupt | |
| Edit interrupt routine | Clicking on the button will open the interrupt routine in the Programming view | |
| Simulation possible | | |

## 6.2.2.4 Other

### Retention

The function block does not recognize retentive data.

The interrupt program does not have any retentive data.

### Example slope in easySoft 7

Rising edge operating mode

Cutting device at station 2. The interrupt is triggered with a pulse at function block input I1. Device output Q1 is set in the interrupt program and the product is cut. Device output Q2 is reset. The interrupt program Q01 passes marker 211 to the main program with Q1 and signals that the cutting device is active.

Once the product has been cut at station 2, there is a rising edge at input I1 of the base device. Processing must be initiated.

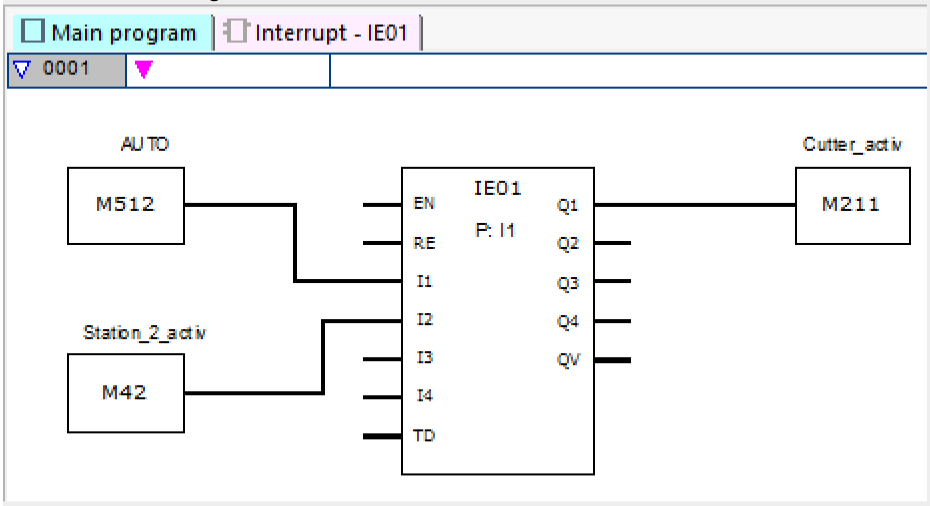



Figure 231: easySoft 7 Main program Slope

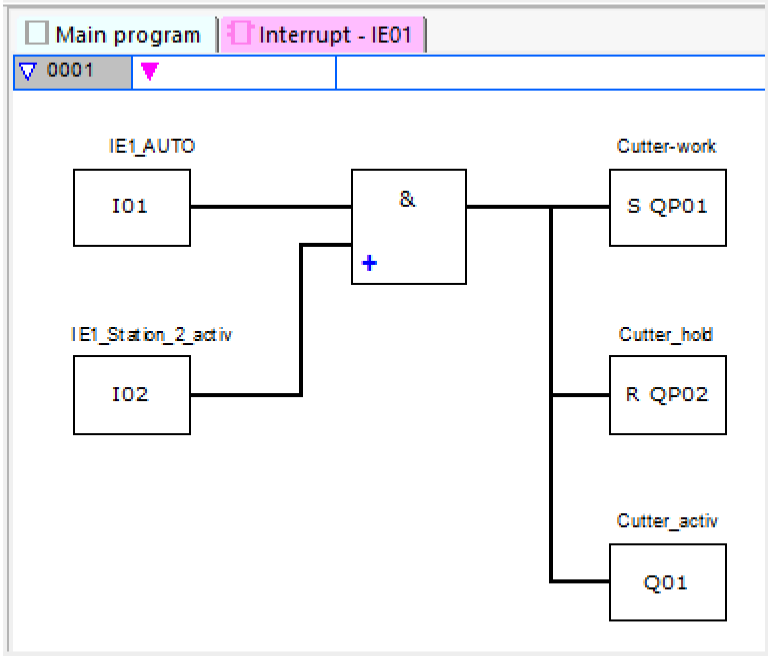



Figure 232: easySoft 7 Interrupt program Slope

### See also

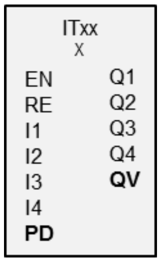
→ Section "IE - Edge-controlled interrupt", page 519
→ Section "IT - Time-controlled interrupt function block", page 525

### 6.2.3 IT - Time-controlled interrupt function block

Only possible with easySoft 7.

#### 6.2.3.1 General

easyE4 base devices provide 8 time-controlled interrupt function blocks, IT01 through IT08. This does not apply to the EDP programming language.
easyE4 makes it possible to quickly respond to various events. This makes it possible, for instance, to switch outputs on or off outside of the main program's routine. Only bit operators are allowed within an interrupt program.
Time-controlled interrupt function blocks can be used in on-delayed mode or interval mode.




**Execution time for an interrupt**
The time between the moment the event is detected and the moment there is a response at a device output is < 1 ms. Accordingly, QP - Physical output on base device must be set in the interrupt program.
If multiple interrupts are executed simultaneously, the times add up.

→ In total, no more than 8 interrupt sources are allowed to be processed in a single program. The valid interrupt sources are the IC, IE, IT interrupt function blocks and the CF, CH, and CI high-speed counters that are directly connected to the device inputs.

→ If there are multiple interrupt requests present simultaneously, the first detected interrupt program will be executed, after which the remaining ones will be executed based on the corresponding order.

#### 6.2.3.2 Operating principle

A reference value is set at function block input PD. As soon as function block input EN is set to 1, the time measurement starts. Depending on the operating mode being used, the system will jump to the interrupt program once or repeatedly as soon as the specified time at function block input PD is reached.

**Interaction between main program and interrupt program**

The states of function block inputs IT_I1 through IC_04 are passed to the interrupt program, where they can be processed further as I01 through I04.

Function block outputs IT_Q1 through IC_Q4 can be set from the interrupt program. The corresponding interrupt program outputs are Q01 through Q04.

Main program Interrupt program

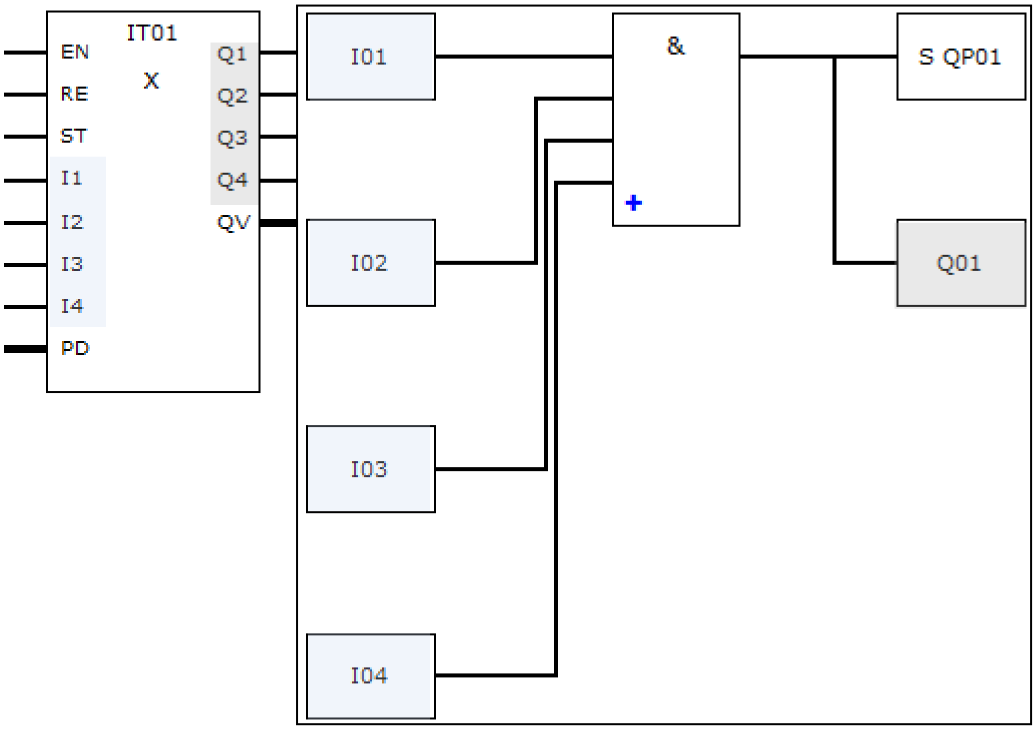



Figure 233: Input and output states being passed between the main program and interrupt program

If an output is defined as a physical output on base device in the parameters for the interrupt program, the output will be assigned an identifier of QP01 – QP04 and will act directly on device output Q1 – Q4.

The function block has its own 32-marker-bit marker range for processing the interrupt program.

**Available functions within an interrupt program**

Interrupt programs are not available when using the EDP programming language.

| Action | KOP | FBS | ST |
|---|---|---|---|
| New network | √ | √ | √ |
| Input/output inverter | √ | √ | √ |
| Contacts | Make, Break, Constant 1, Constant 0 | | |
| Coils | Contactor, Negated contactor, Set, Reset | | |
| Jump functions | Jump if 1, Jump if 0, Return if 1, Return if 0 | | |
| Logic gates | AND, AND NOT, OR, OR NOT, XOR, XNOR | | |
| Conditional statement | – | – | √ |
| Simple alternative | – | – | √ |
| Multiple alternatives | – | – | √ |

### 6.2.3.3 The function block and its parameters

#### Function block inputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| EN | 1: Activates the function block. | |
| RE | 1: Sets the actual time of the interrupt function block back to the time at PD. | |
| ST | 1: Stops the interrupt function block's time measurement.<br>0: The interrupt function block's time measurement will continue. | |
| I1 | The states of the bit inputs from the main program are provided to the interrupt program. | |
| I2 | | |
| I3 | | |
| I4 | | |
| **(DWord)** | | |
| PD | Pulse pause time:<br>Value of the delay that must elapse before the interrupt program is started. | Integer value range:<br>20…999 990 ms,<br>resolution 10 ms |

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

| Operators | Value inputs |
|---|---|
| Constant, timer constant[1)] | x |
| MD, MW, MB - Markers | x |
| NB, NW, ND - NET markers[2)] | x |
| nNB, nND, nND- NET markers [2)]<br>NET station n | x |
| IA - Analog input | x |
| QA - Analog output | x |
| QV - QV - Numeric output of a FB | x |
| 1) Only on function blocks T, AC<br>2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block inputs that are bit inputs:

| Operators | Bit inputs |
|---|---|
| Constant 0, constant 1 | x |
| M – Markers | x |
| RN - Input bit via NET[2)] | x |
| SN - Output bit via NET (send)[2)] | x |
| N - Net marker bit[2)] | x |
| nN - NET marker bit[2)] NET station n | x |

| Operators | Bit inputs |
|---|---|
| ID: Diagnostic alarm | x |
| LE - Output backlight | x |
| P device buttons | x |
| I - Bit input | x |
| Q - Bit output | x |
| Q - Bit output of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

### Operating modes

| | Description | Note |
|---|---|---|
| On-delayed | If the time set at function block input PD is reached, the system will jump to the interrupt program once | |
| Interval | If the time set at function block input PD is reached, the system will jump to the interrupt program. The time measurement will start again and, after it elapses, the system will jump to the interrupt program again. This will keep happening as long as function block input EN = 1. | |

IT interrupt function blocks feature two operating modes that work as described below:

- On-delayed
  The interrupt function block is enabled via function block input EN. The pulse/pause time at function block input PD starts to count down. When the pulse/pause time at function block input PD elapses, the interrupt is triggered immediately and the interrupt program is processed.
- Interval
  The interrupt function block is enabled via function block input EN. The pulse time at function block input PD starts to count down. When the pulse time at function block input PD elapses, the interrupt is triggered immediately and the interrupt program is processed. After this, the pause time at function block input PD starts to count down. When the pause time at function block input PD elapses, the interrupt is triggered immediately and the interrupt program is processed. This means that the interrupt is triggered twice: once at the end of the pulse and once at the end of the pause.

### Function block outputs

| | Description | Note |
|---|---|---|
| **(Bit)** | | |
| Q1 | Bit output used to provide operand states from the interrupt program to the main program. | |
| Q2 | | |
| Q3 | | |
| Q4 | | |

| | Description | Note |
|---|---|---|
| **(DWord)** | | |
| QV | Elapsed actual time of delay set at PD | |

**Assigning operands**

You can assign the following operands to the function block outputs that are numeric outputs:

| Assigning operands | Value outputs |
|---|---|
| MB, MD, MW - Markers | x |
| NB, NW, ND - NET markers 2)<br>NET stations n | x |
| QA - Analog output | x |
| I - Value input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

You can assign the following operands to the function block outputs that are bit outputs:

| Assigning operands | Bit outputs |
|---|---|
| M - Markers | x |
| SN - Output bit via NET (send)2) SN - Output bit via NET (send) | x |
| N - Network marker bit2) | x |
| LE - Output backlight | x |
| Q - Bit output | x |
| I - Bit input of a FB | x |
| 2) Only on projects with ≥ 2 base devices on NET | |

**Parameter set**

| | Description | Note |
|---|---|---|
| Parameter display<br>+ Call enabled | Constants can be edited on the device, as can function block parameters when using the EDP programming language. | |
| Edit interrupt routine | Clicking on the button will open the interrupt routine | |
| Simulation possible | | |

### 6.2.3.4 Other

**Retention**

The function block does not recognize retentive data.

The interrupt program does not have any retentive data.

**Monitoring the interrupt load**

In total, no more than 8 interrupt sources are allowed to be processed in a single program. The valid interrupt sources are the IC, IE, IT interrupt function blocks and the CF, CH, and CI high-speed counters that are directly connected to the device inputs. For more information, please refer to → "CF - Frequency counter", page 287, → "CH - High-speed counter", page 293 , → "CI - Incremental Counter", page 299 as well.

For function blocks IE01 – IE08 and IC01 – IC08, device inputs I01 – I08 can be assigned freely.

For function blocks IT01 – IT08, an interrupt that is not yet in use is assigned in easySoft 7. The interrupt sources used by the CF, CH, and CI high-speed counters are also considered to be in use within this context.

Each device input and each interrupt source can only be used once.

Exceptions:

- For CI01, the instance of I02 can be used by an IT interrupt function block.
- For CI02, the instance of I04 can be used by an IT interrupt function block.
- For each interrupt function block IC, the instance of the second input of an IT function block can be used if the Counter with 2 counter inputs mode has not been selected.

These exceptions are taken into account by the plausibility check and by the program compilation routine in easySoft 7. The maximum number of 8 interrupts is also taken into account within this context.

| | **Device inputs** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **I01** | **I02** | **I03** | **I04** | **I05** | **I06** | **I07** | **I08** |
| **Interrupt source** | | | | | | | | |
| CF01 frequency counter | x | | | | | | | |
| CF02 frequency counter | | x | | | | | | |
| CF03 frequency counter | | | x | | | | | |
| CF04 frequency counter | | | | x | | | | |
| CH01 high-speed meter | x | | | | | | | |
| CH02 high-speed meter | | x | | | | | | |
| CH03 high-speed meter | | | x | | | | | |
| CH04 high-speed meter | | | | x | | | | |
| CI01 incremental counter | x | x | | | | | | |
| CI02 incremental counter | | | x | x | | | | |
| IE01…IE08 | One input, I01 – I08 can be assigned freely (max. 8, none can be assigned more than once) | | | | | | | |
| IC01 – IC08 | Two inputs, I01 – I08 can be assigned freely (max. 8, none can be assigned more than | | | | | | | |

| | Device inputs | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | I01 | I02 | I03 | I04 | I05 | I06 | I07 | I08 |
| | once) | | | | | | | |
| IT01…IT08 | Automatically assigns the user interrupts 1 to 8 that are still available (only for instances of I01 – I08 that are not in use by other function blocks) | | | | | | | |

The time between the moment the triggering signal is detected and the moment there is a response at an output is < 1 ms. If multiple interrupts are executed simultaneously, the times add up.

**Measuring the interrupt load**

The runtime for each interrupt source is measured in µs. All measured times are added over a period of 100 ms. After each 100 ms, the total of all times is evaluated and the time measurement is cleared. If the interrupts used up more than 50% of the computing time, the application will be stopped.

The <System_CPU_overload> diagnostic message will be generated and ID19 will be set to 1.

For more information on how diagnostic messages can be called and processed, please refer to

**Available fixes for high interrupt loads**

If the interrupt load becomes too heavy, the following steps can be taken to reduce it:

- Reduce the number of function blocks
- Keep the interrupt routine as short as possible
- Reduce frequencies when using counters

**Example of a time-controlled interrupt function block in easySoft 7**

Output Q4 needs to be reset after a specific time. This time should be independent from the main program's cycle time so that the switch-off point in time is always the same.

Operating mode: On-delayed

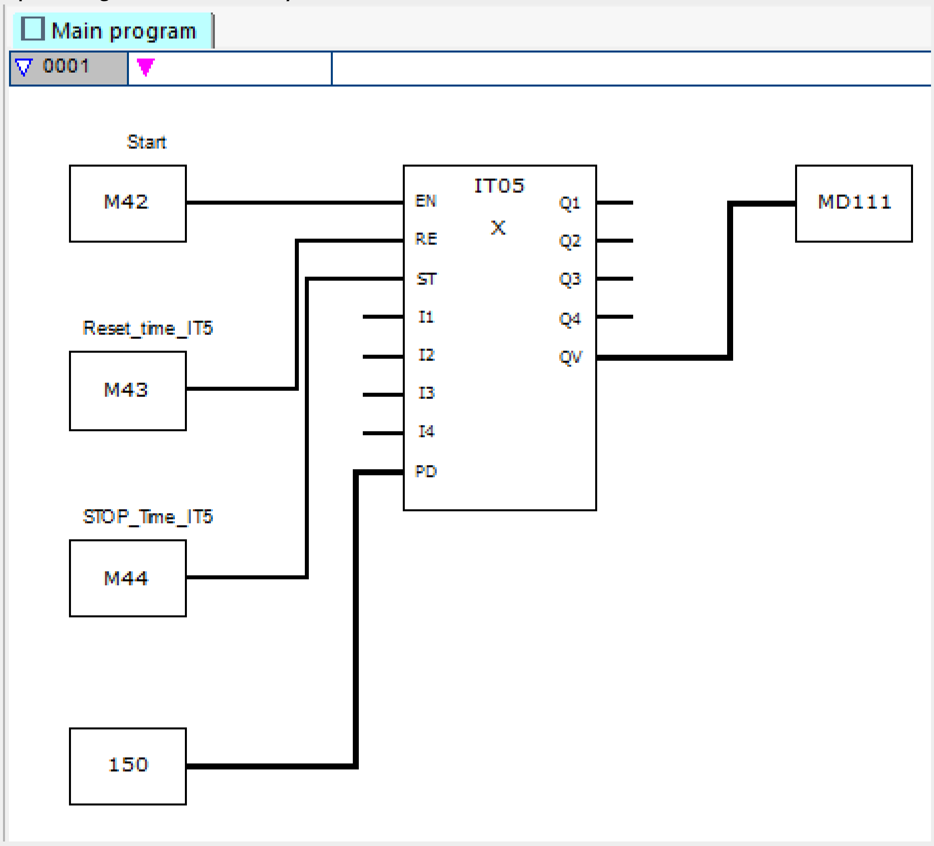



Figure 234: easySoft 7 Main program Time-controlled

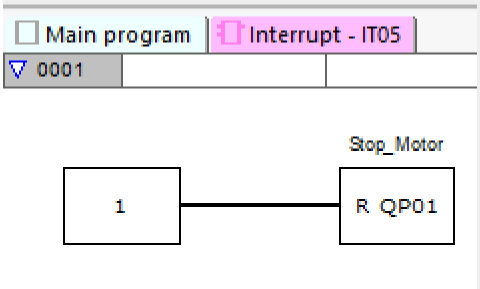



Figure 235: easySoft 7 Interrupt program Time-controlled

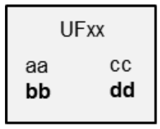
## See also

→ Section "IT - Time-controlled interrupt function block", page 525
→ Section "IE - Edge-controlled interrupt", page 519

# 6.3 UF - User function block

Only possible with easySoft 7.

## 6.3.1 General




easyE4 base devices provide 128 user function blocks, UF01 through UF128.
You can configure these function blocks yourself and then use them in the main program the same way as manufacturer function blocks.

User function blocks are used whenever a recurring functionality needs to be programmed with various parameters. For example, when similar machines need to be controlled, the actual control program for them can be written in a user function block and then called multiple times – separately for each machine. User function blocks also feature inputs and outputs that can be used to pass custom parameters for each call.

The programming language used in each user function block is independent from the programming language used for the corresponding main program. In other words, you can use user function blocks written in ST in an FBD or LD main program, for instance.
User function blocks have their own data range. In fact, there are 64 bytes, which can be used as bits, bytes, words, or double words, available for each instance (call) of a user function block. This means, for example, that marker M01 in the main program is not the same marker M01 in a user function block.
Parts of the markers can be declared as being retentive. In this case, it is important to keep in mind that the total number of retentive markers cannot exceed 400 bytes, and that this total includes the retentive markers from the main program and the retentive markers from all user function block instances.
Just like a main program, a user function block is made up of FBD/LD networks or ST source code. This means that a user function block can be created the same way as a main program, with the only differences consisting of the available operands (please refer to → Section "Configuring a user function block", page 536.

The maximum number of user function blocks that can be called in a single main program is 128.

#### 6.3.1.1 General information on user function blocks

The markers and function blocks used in a user function block have their own self-contained data range.This precludes any conflicts with data from other user function blocks or data from the main program. Likewise, the standard function blocks used in a user function block, as well as their parameter sets, are managed separately in the firmware for each function block instance.

The number of instances of a manufacturer function block that can be used in a user function block is the same as the number that can be used in the main program. The program is limited only by the available program memory.

When a main program is loaded onto an easyE4 device or into the current project, all the user function blocks used in the main program will be loaded as well.

### 6.3.2 Creating a user function block

Once you have created a project and selected a programming language, you can create a user function block.

▶ Click on the *Program/ Create user function block...* menu option
or
click on the  button in the toolbar.

The Create user function block dialog box will appear

*Program/ Create user function block... menu option*

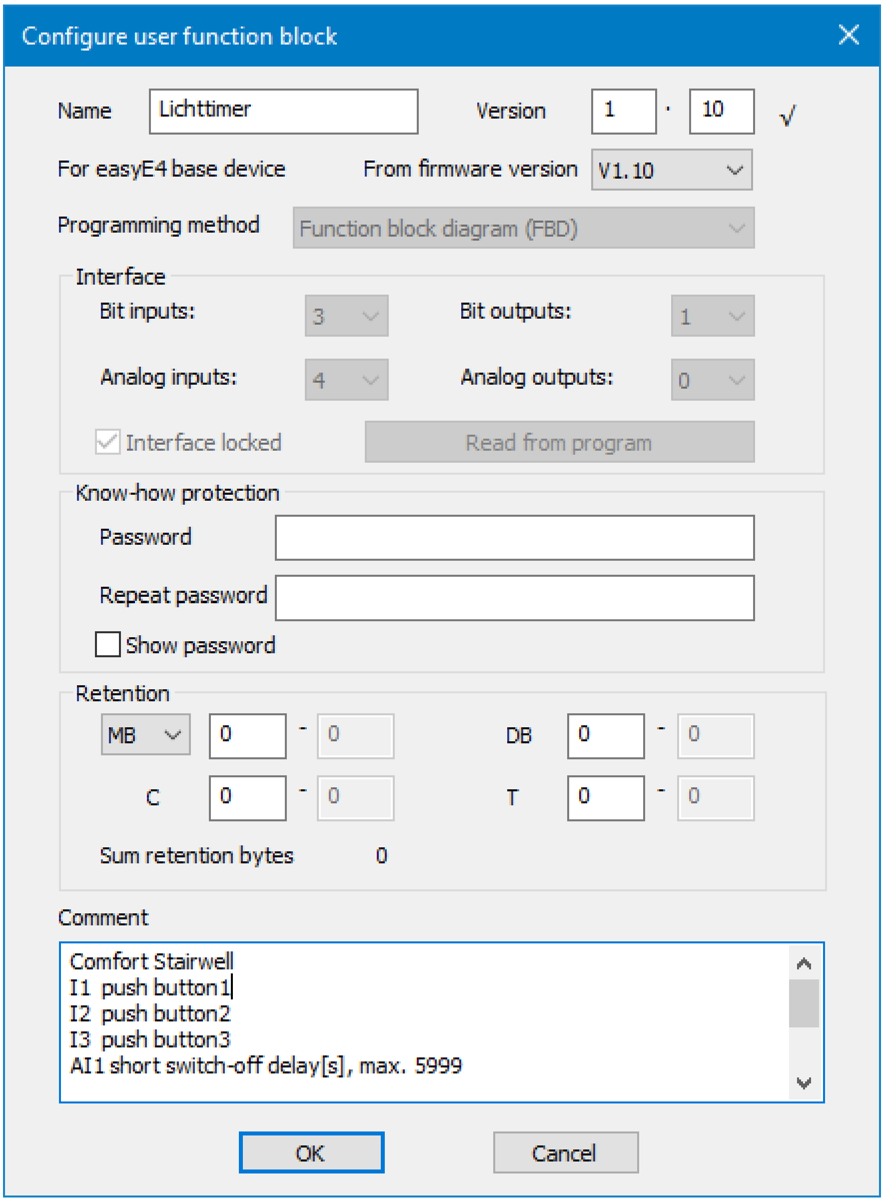



Figure 236: Create user function block

You must at least enter a name and version and select a programming method before creating the user function block. However, in order to be able to use the user function block effectively, it is also a good idea to configure the options in the Interface section. In short, this section is used to specify how big the number of parameters passed from the main program should be.

You can choose to configure all the other options later on if you want. The "Configuring a user function block" section goes into them in greater detail.

**Name and Version**

The name you choose for the user function block should not exceed a maximum of 10 characters. The following characters are allowed:

- Uppercase and lowercase letters
- Numbers
- Special characters # $ % & ` ( ) + , - ; = @ [ ] ^ _ ‘ { } ~

Spaces and the \ / . : * ? < > | special characters are not allowed. The name is not case-sensitive. If you enter a name that meets all requirements, a black checkmark will appear to the right of the Name field. If the name needs to be corrected instead, a red exclamation mark will appear. A new user function block will automatically be assigned version 1.00. You can enter any version number between 0.00 and 99.99.

**Programming method**

You can use this drop-down menu to select the programming language (LD, FBD, ST) for the user function block. The default is FBD. The programming language you select here will be independent from the programming language used for the main program. However, please note that once you create a user function block, you will no longer be able to change its programming language.

▶ Clicking on OK to close the "Create user function block" dialog box will create and save the new user function block.

Once you click on OK, the empty programming unit for the user function block will be opened and the Programming view work pane will show an additional tab with the same name as the user function block, e.g., <UF – Blinker1V1.00>.

If you switch to the Main program tab, the user function block will appear in the list of operands and function blocks, in the "User function blocks" folder.

### 6.3.3 Configuring a user function block

In order to configure a user function block, click on the tab with the same name as the user function block inside the work pane, e.g., <UF – Light timer V1.10>, and use one of the following methods:

▶ Click on the *Program/ Configure user function block...* menu option.

▶ Inside the pane, click on the tab with the same name as the user function block, e.g., <UF – Light timer V1.10>, and then click on the "Configure user function block" button in the toolbar.

▶ Right-click on the tab for the user function block in the work pane and select the Configure... option.

Or

- In the work pane, click on the <Main program> tab.
- Go to the *list of operands and function blocks / User function blocks folder* , right-click on the function block, and then select the Configure... option.

The Configure user function block dialog box will appear.

- Enter all parameters.
- Confirm your input and close the dialog box by clicking on OK or pressing the <Return> key.

The changes will be applied to the user function block.

The Name, Version, and Programming method fields have already been described in the "Creating a user function block" field. Please note that, although the "Configure user function block" dialog box will show the programming language you originally selected, it will no longer be possible to change it.

*Menu option Program/ Configure user function block*

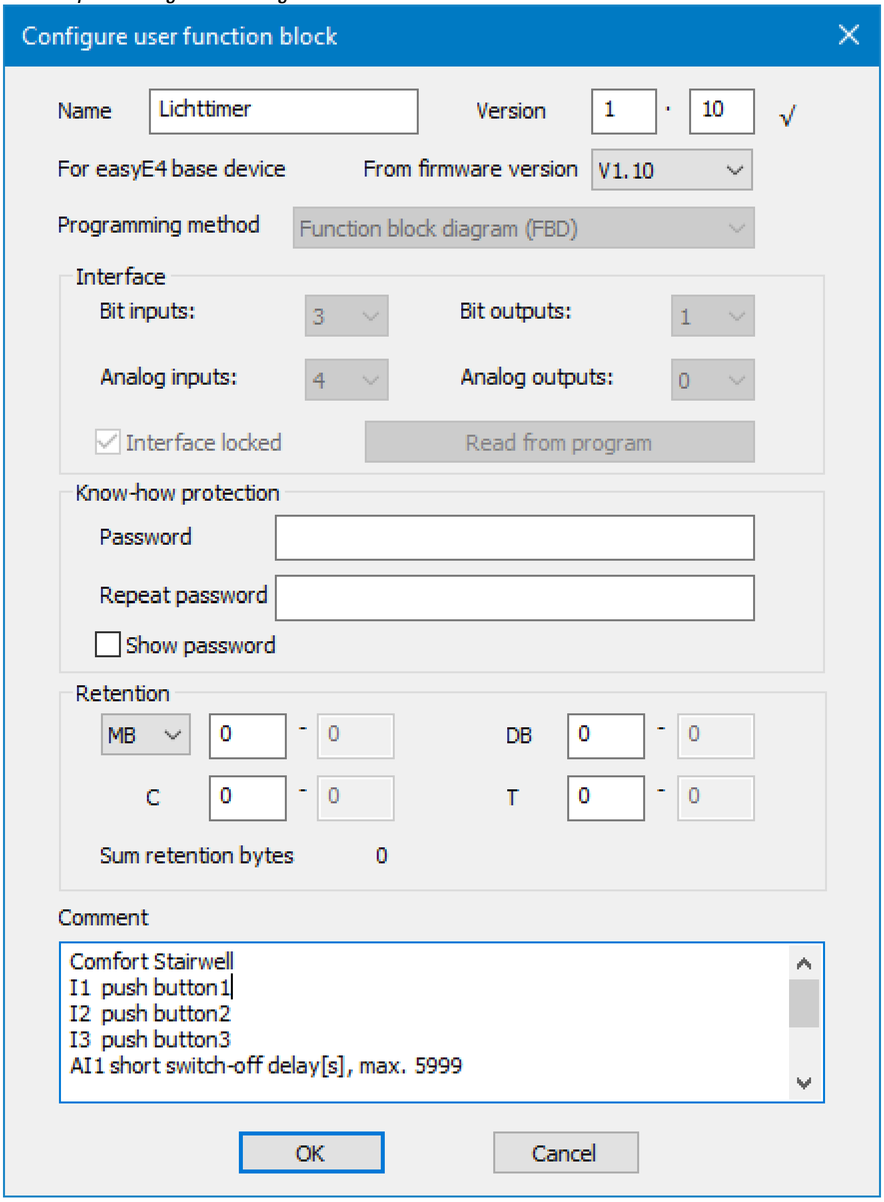



Figure 237: Configure user function block

### From firmware version

You can use this drop-down menu to specify the minimum firmware version for using the user function block. The function blocks and language elements available for the selected firmware version will be available.

→ Please note that, after selecting a firmware version, you will not be able to change it back to the older version.

### Interface section

You can use the options in this dialog box to define the number of digital and analog inputs and outputs for your user function block. These inputs and outputs will then form the interface between the user function block and the main program. You can configure a maximum of 12 digital bit inputs/outputs and a maximum of 8 analog inputs/outputs. Please note, however, that the total number of all inputs and outputs cannot exceed 12.

When the user function block is called in the main program, the inputs and outputs defined in the Interface section will be shown and you will be able to configure them.

**Read from program**

If the program for the user function block has already been written and it uses inputs and outputs, you can click on the "Read from program" button to have the system automatically determine the interface parameters. The highest input/output index used will be the one copied over, and any gaps in the circuit will be ignored. Please note that this button will not be available if:

- The inputs and outputs are configured correctly in conformity with the program for the user function block.
- The user function block has already been used in a main program in the project.

easySoft 7 will not check whether the inputs/outputs used in the user function block program are defined in the Interface section as well.

**Know-how protection section**

You can set a password in order to prevent a user function block from being viewed or edited. This password must not be longer than 32 Unicode characters. If the password entered into both fields matches, a black checkmark will appear and the OK button will be enabled again.

**Retention section**

It is a requirement of system and machine controllers for operating states or ACTUAL values to have retentive settings. What this means is that the values will be retained until the next time the ACTUAL value is overwritten.

There are two input fields (for the start and end values of the retention range) each for markers and for the following function blocks.

*Project view/ System settings tab*

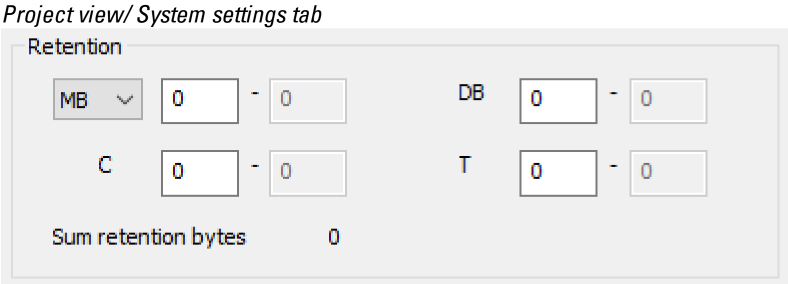



Figure 238: Screenshot of Retention section that can be found in the System settings tab in the Project view (including sample values)

Value range for the function blocks, instances that can be stored retentively:

- C - Counter relay : 01…32
- CH - High-speed counter : 01…04
- CI - Incremental counter : 01…02
- DB - Data function block : 01…32
- T - Timing relay : 01…32

For more information, please refer to the description for the relevant function block.

Marker value ranges:

- MB : 1 …512
- MW : 1…512
- MD : 1…256

The values from the input field will be automatically converted to MB marker bytes.

**Retention bytes**

The entire retentive marker range for an easyE4 must not exceed 400 bytes. The total of the retention bytes for the main program and user function blocks (UF) will be displayed in the Project view, in the System settings tab. If the retentive marker range exceeds 400 bytes, a red negative number will be shown in the Free field in order to indicate this.

**Retain retention during transfer**

Retentive ACTUAL values on the device are deleted:

- By any program change in the circuit diagram or function block diagram followed by its transfer to the device.
- When the program is deleted with the *Communication/Program/Configuration/Delete device* button
- By any change of the retentive range via *Project view/System settings/Retention*.
- By any changes to the parameters for the remote markers of a visualization device.
- When deleting the device from the workspace of the project view.

The following exception applies to retentive markers:

☑ Marker contents

If this option is enabled, the contents of the existing retentive marker range will be retained when the program is transferred. The ACTUAL marker values are retained.
In order for this to work, however, the marker range defined as being retentive must remain unchanged.

☑ Function block contents

If this option is enabled, the contents of the existing retentive operand range will be retained when the program is transferred.
In order for this to work, however, the function block defined as being retentive must remain unchanged.

### Comment section

This box can be used to enter an optional comment, e.g., in order to distinguish between various versions of a user function block.

## 6.3.4 Programming a user function block

Once you have created a user function block, the view will automatically switch to the Programming view for the user function block. Moreover, a tab with the user function block's name and version will appear next to the Main program tab in the work pane. This tab will be green as long as the user function block is not being used in a main program. As soon as the function block is used in the main program, however, the tab will change color to yellow.

User function blocks are programmed exactly the same way as a main program. The only difference is that there are slightly fewer operands available. The list of available operands and function blocks will automatically change to reflect this.

You should be in the Programming view for the user function block. Following is a screenshot showing an example in which a timing relay is programmed with the flashing operating mode.

*Programming view for UF Blinker1 tab*

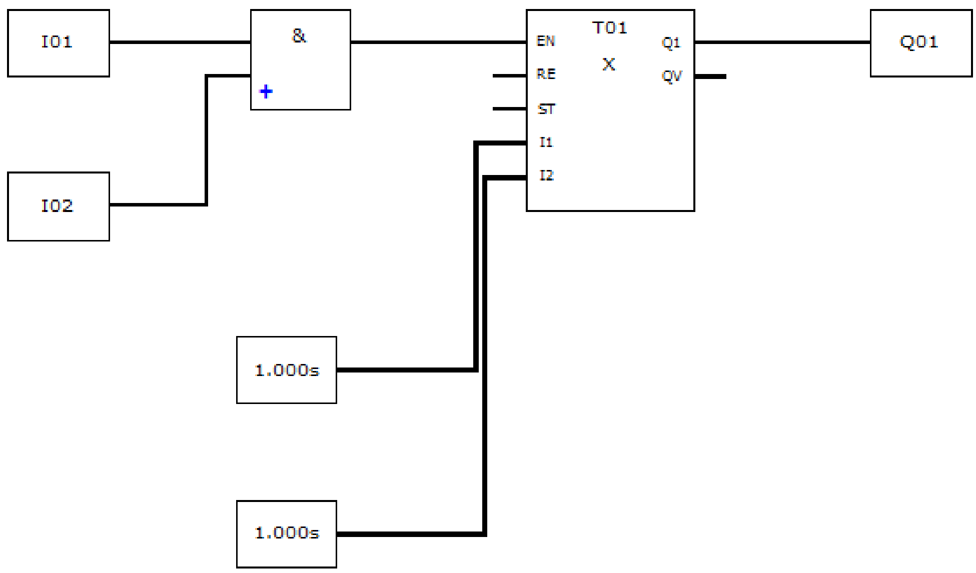



Figure 239: Programming view for UF Blinker1 user function block

- Run a plausibility check first.
- Save the user function block and switch to the Programming view for the main program.

The user function block will appear in the list of operands and function blocks with a green icon that means that it is not being used in the project yet.

#### 6.3.4.1 Programming view tabs

The tabs in the Programming view are intended to help you keep your project more manageable.

When applicable, tabs for user function blocks and interrupt function blocks will be found next to the tab for the main program. Different colors and icons will be used to differentiate between them:

| Color | Registers |
|---|---|
| Blue | Main program |
| Green | User function block that is not being used |
| Yellow | User function block that is being used |
| Magenta | Interrupt function block |

Inactive tabs will be shown with a brighter color. A total of 11 tabs can be displayed.

## Adding comments to user function blocks

It is recommended to add detailed comments to your user function blocks. This will enable users to understand how to use the function block even if they do not have a password. There are three ways to show the comments for a user function block:

1. In the Programming view, go to the *list of operands and function blocks / User function blocks folder*, right-click on the function block, and select the Show comments… option.
2. Open the user function block and select the *Program/Show user function block comments…* menu option.
3. Select the user function block in the main program.
   The comments will be shown in the tab.

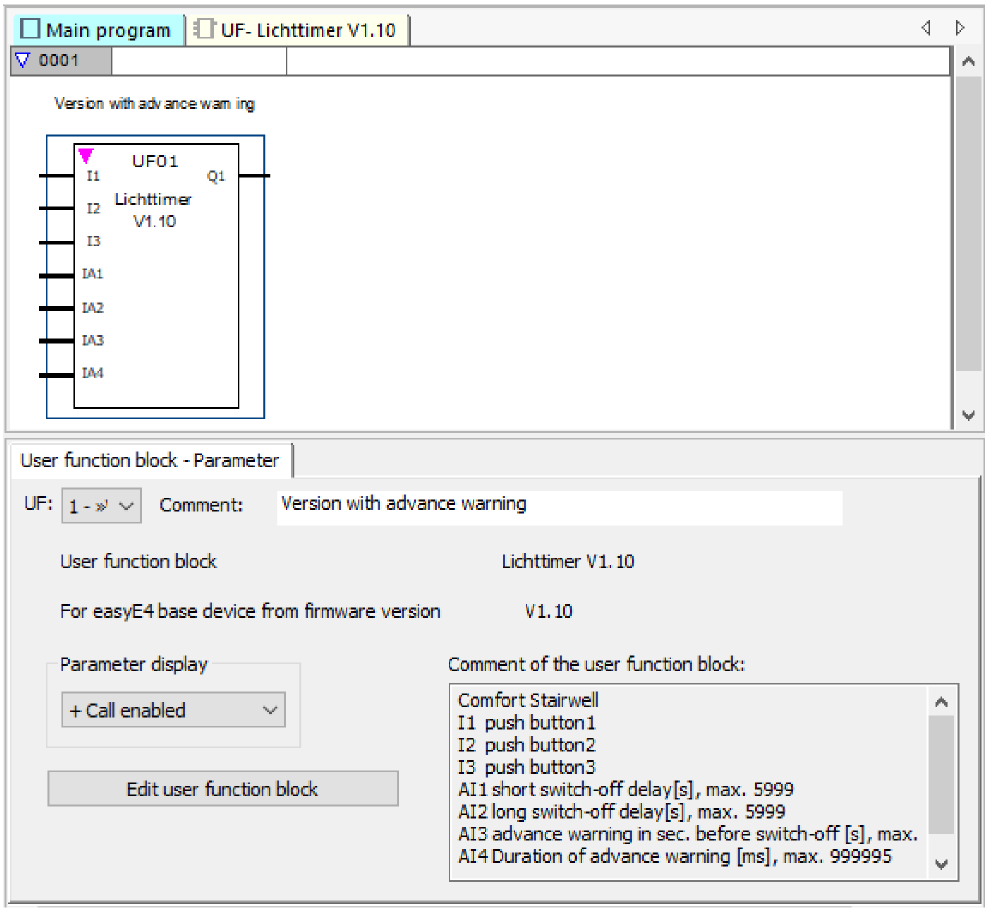



Figure 240: User function block comments being shown in the tab

The operand comments for a user function block are managed separately from the operand comments from the main program. This means, for example, that I1 ("Otto") in the user function block can have a comment different from I1 ("Emil") in the main program.

### 6.3.5 Calling a user function block in the main program

User function blocks can be called in the main program the same way manufacturer function blocks are called.

#### User function block in and FBD main program

In order to call a user function block in a main program that uses the FBD programming language, drag the function block like a normal function block to the work pane in the Programming view.

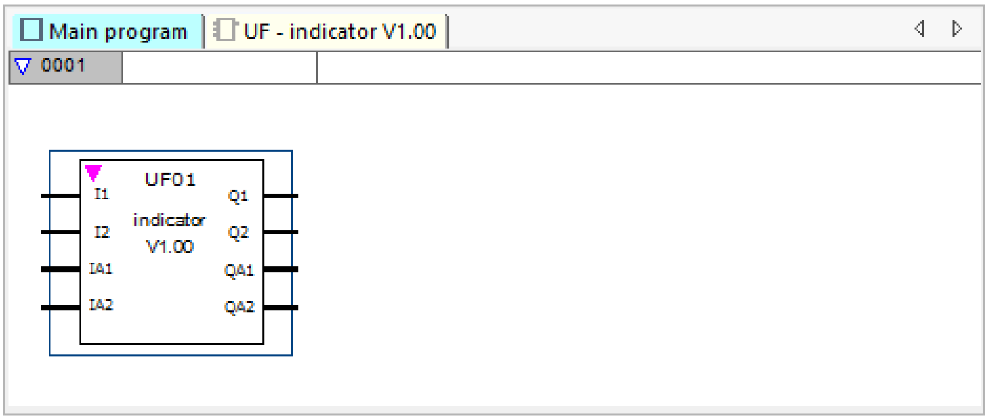



Figure 241: UF Blinker1 user function block used in the main program

The function block will be shown with its name, version, and configured inputs/outputs. The function block type will be "UF", followed by the instance number (01 to 128).

The function block will now appear in the leftmost pane with a yellow icon, and the tab in the work pane will also change color to yellow, which means that the function block is being used in the project.

### Inputs/outputs wiring

The digital and analog inputs and outputs can be connected the same way as for any other function block. The example shows the user function block's Q1 output being connected to a counter relay's C input.

*Programming view*

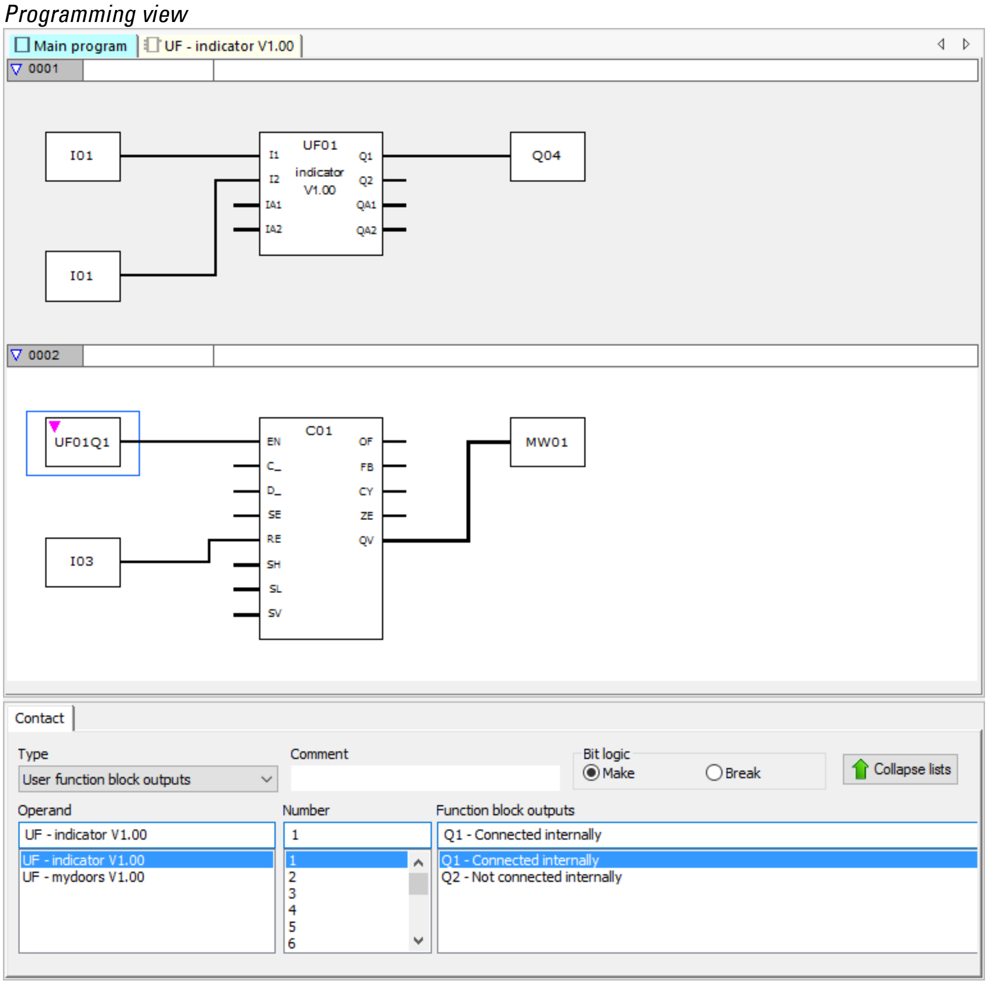



Figure 242: Inputs/outputs wiring

User function block inputs and outputs can be copied in the main program and pasted the same way as those from any other operand.

If a user function block call is copied and pasted, the new call will be assigned the next free instance number.

All the user function blocks used in a project's main programs will be part of the project file and will be saved together with the project.

If there are any user function blocks, the tabs will change accordingly:

*Programming view*

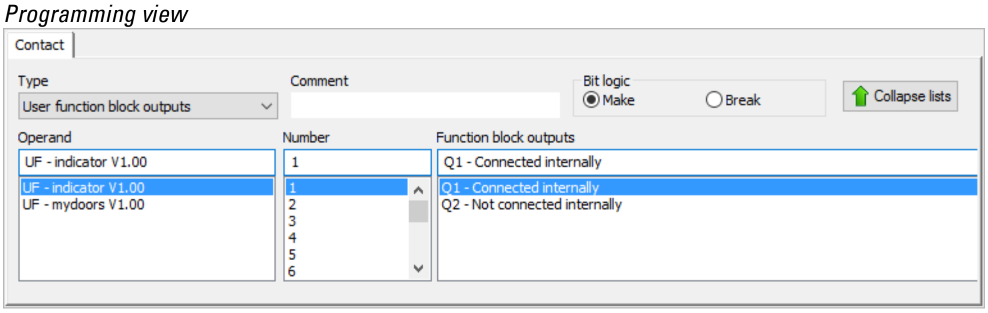



Figure 243: Contact tab

*Programming view*

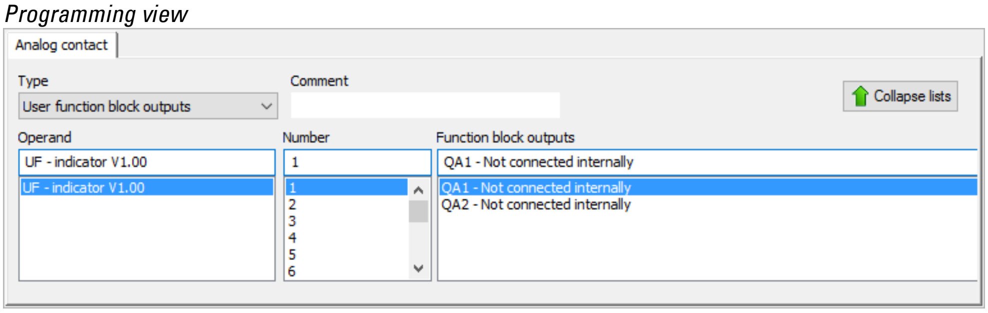



Figure 244: Analog contact tab

If there are any user function blocks with bit and/or analog outputs available, you will be able to select the "User function block outputs" option in the "Type" drop-down menu.

Meanwhile, the "Operand" drop-down menu will contain all registered user function blocks that have bit and/or analog outputs.

The "Number" drop-down menu will contain all available function block numbers within a range of 1 to 128, as well as the comment entered for the corresponding number. Please note that any instance numbers that have already been assigned to other types of user function blocks will not be available for selection.

Finally, the "Function block outputs" drop-down menu will list the various individual outputs, together with information specifying whether the contact is connected internally.

In addition, you will be able to select the bit logic for digital outputs.

*Programming view*

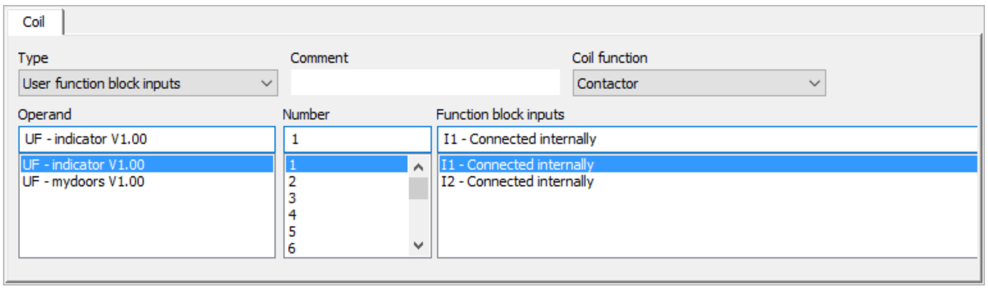



Figure 245: Coil tab

*Programming view*

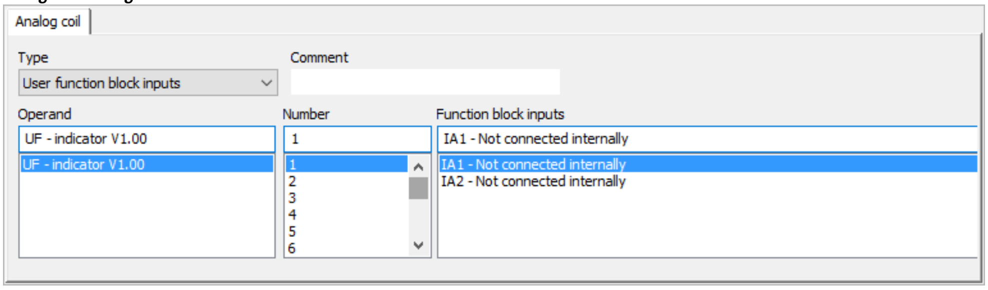



Figure 246: Analog coil tab

If there are any user function blocks with bit and/or analog inputs available, you will be able to select the "User function block inputs" option in the "Type" drop-down menu.

Meanwhile, the "Operand" drop-down menu will contain all registered user function blocks that have bit and/or analog inputs.

The "Number" drop-down menu will contain all available function block numbers within a range of 1 to 128, as well as the comment entered for the corresponding number. Please note that any instance numbers that have already been assigned to other types of user function blocks will not be available for selection.

Finally, the "Function block inputs" drop-down menu will list the various individual inputs, together with information specifying whether the coil is connected internally.

In addition, you will be able to select the coil function (Contactor, Set, Reset, etc.) for digital inputs.

#### 6.3.5.1 User function blocks in an ST main program

A user function block created with the FBD language can also be called in an ST main program and vice versa.

In this case, a template based on the user function block's parameters (as defined in the Interface section) will be generated in the ST program if the function block is dragged onto the program. You will be able to connect the inputs and outputs the same way as with manufacturer function blocks.

The NAME and VERSION you entered originally will be used to define the user function block's type and version. These two pseudo inputs cannot remain unconnected and are not allowed to be mapped outside of the function block call.

**Example**

```
;UF01 (
    NAME := "indicator",
    VERSION := "V1.00",
    I1 := I01,
    I2 := I02,
    IA1 := ,
    IA2 := ,
    Q1 => ,
    Q2 => ,
    QA1 => ,
    QA2 =>
 );
 C01 (
    EN := ,
    C_ := UF01Q1,
    D_ := ,
    SE := ,
    RE := I03,
    SH := ,
    SL := ,
    SV := ,
    OF => ,
    FB => ,
    CY => ,
    ZE => ,
    QV => MW01
 );
```

The example shows the user function block's Q1 output being connected to a counter relay's C input.

### 6.3.6 Saving a user function block

You can close an open user function block at any time. Likewise, you can save changes to the user function block at any time. Please note that if you close a modified user function block, the system will ask whether you want to save or discard the changes.

All user function blocks are stored in the \ProgramData\Eaton\easySoft 7\UserFBs folder.

The "Program->Close" menu option and the "Close" button will be available if the user function block is open and either the user function block view is open or the user function block is selected in the main program view.

The "Program->Save user function block" and the "Save user function block" button will be available if the user function block is open and has been modified and either the user function block view is open or the user function block is selected in the main program view.

## Operands available for user function blocks

When a user function block is selected, the list of operands and function blocks will show the operands that are available. The number of manufacturer function blocks will be smaller.

All operands within a user function block refer to a separate local memory area. Supported (local) operands:

| Operand | Maximum number |
|---|---|
| I | 12 |
| IA | 8 |
| Q | 12 |
| QA | 8 |
| M | 480 |
| MB | 64 |
| MW | 32 |
| MD | 16 |

These numbers specify the maximum possible number for I, IA, Q, and QA. However, the following restrictions apply:

- The total number of inputs (bit and analog) must not exceed 12
- The total number of outputs (bit and analog) must not exceed 12
- A maximum of 12 bit inputs and outputs can be used
- A maximum of 8 analog inputs and outputs can be used

Device-specific operands (ID, LE, P) and NET operands (N, NB, NW, ND, RN, SN) are not supported for user function blocks.

**Supported manufacturer function blocks:**

All standard function blocks can be used in a user function block, with the exception of function blocks that have a hardware interface or firmware reference (i.e., OT, CF, CH, CI, PW, PO, GT, PT, SC, AL, D, DL, and ST). Function blocks BC, BT, and MR can be used, but will only act on the user function block's local data arrays.

- The Copy, Cut, and Paste functions are supported the same way as in the main program. However, they can only be used between user function blocks.
- Just like in the main program, the keyboard can be used to enter the I, Q, IA, QA, M, MB, MW, and MD operands as contacts and coils.
- In addition, and just like in the main program, the keyboard can be used to create contacts and coils corresponding to the supported function blocks, inputs, and outputs. This applies both to entering an operand completely and to changing the index number for an operand.
- As soon as a change is made to a user function block, the Save user function block option in the main menu and the Save user function block button in the toolbar will become available.

### 6.3.7 Exporting a user function block

User function blocks can be saved in a separate folder as a uf7 file. The"Export user function block" menu option will become available if a user function block call is selected or the programming view for a user function block is open.

Before the user function block is exported, it will be subjected to a plausibility check. Please note that it will only be possible to export the function block if it does not contain any errors. Finally, if the function block is protected with a password and is not open, a prompt asking you to enter the password will appear.

A dialog box will appear and ask if you want to edit the user function block's name, version, password, and/or comment before exporting it.

Yes— Yes: The "Edit user function block settings" dialog box will appear. Please note that if a password was previously set for the user function block, you will need to enter it first. If you do not enter the password, a prompt will appear asking whether you still want to export the user function block.

No: No: The "Select user function block folder" dialog box will appear. You can then select the folder where you want the user function block's uf7 file to be saved.

→ If the selected folder contains any items (files, folders, archives) with the exact same name as the user function block being exported, you will not be able to notice this in the "Select user function block folder" dialog box. Accordingly, make sure to first check whether the selected folder is suitable for saving the file.

Clicking on the "Select Folder" button may result in any of the following scenarios if there is a problem:

In the five cases below, you will be able to select a different folder:

1. The selected drive is not ready or is write-protected
2. The selected drive does not have enough free memory
3. The selected folder cannot be accessed
4. The selected folder is write-protected
5. The selected folder already contains a folder named UserFB_V1_01.uf7

In the first three cases, you will be asked whether you want to overwrite the file or select a different folder.

1. The selected folder already contains an archive named UserFB_V1_01.uf7
2. The selected folder already contains a user function block named UserFB_V1_01.uf7
3. The selected folder contains a write-protected user function block named UserFB_V1_01.uf7.

If the aforementioned checks are all completed successfully, the user function block will be saved and the user interface will be refreshed in the Programming view and in the list of operands and function blocks if applicable.

#### 6.3.7.1 Plausibility check

When exporting a user function block, a user function block check that determines whether the user function block can be executed in the easyE4 device's current state will be triggered. This is especially necessary for user function blocks programmed using ST, as entering impermissible operands is possible in these cases.

The export function will generate a uf7 file only if the user function block is executable. In this case, the file will contain not only the actual user function block, but also all required management data.

This check can be run at any time on user function blocks that are in use or that are not in use in the project. The sole exception consists of password-protected user function blocks in use.

A user-function-block-specific plausibility check will not be run when copying and pasting between user function blocks. All checks are identical to those run on the main program.

Within the context of the plausibility check for a device, the system will check whether the total number of user function blocks per device is less than or equal to 128. If the plausibility check for a device outputs an error / warning for a user function block in the "Report output" dialog box and the view for the user function block is not active or open in the Programming view, double-clicking on the error / warning will activate, or, if applicable, open the Programming view for the user function block and show exactly where the error or warning is found.

Depending on the corresponding results, the following messages may appear after a plausibility check:

- FB input / FB output %2 is not part of the user function block's interface
- The number for an FB input / FB output was not assigned without gaps
- FB input %2 exceeds the maximum number of a total of 12 bit and analog inputs
- FB output %2 exceeds the maximum number of a total of 12 bit and analog outputs
- Operand %2 is not supported in user function blocks!
- The number for operand %2 falls outside the permissible value range for user function blocks.

### 6.3.8 Importing a user function block

The import function makes it possible to load user function blocks (uf7 files) from a folder. This function is available in the Programming view.

→ In order to be able to import user function blocks, all open user function blocks must be unmodified.
If they are not, the following message will be output: If open user function blocks are modified, an import will not be possible. Please save all modified user function blocks first..

▶ Select a uf7 file and click on "Open"

The selected user function block will be added to the user function block management group only if it meets certain criteria.

The following messages can occur:

- User function block is already found in easySoft 7.
  You do not need to import it. Do you want to select a different file?
- A user function block with different content is already found in easySoft 7. Since it is used in the project and the function block interfaces are different, the import is not allowed. Do you want to select a different file?
- A user function block with different content is already found in easySoft 7. This user function block is open for editing, meaning that an import is not possible. Do you want to select a different file?

The following applies to these three scenarios:

| | |
|---|---|
| No: | The import will be aborted |
| Yes— | You will be able to select a different file |

- A user function block with different content is already found in easySoft 7. Do you want to replace this user function block with the function block being imported?

| | |
|---|---|
| No: | You will be able to select a different file |
| Yes— | The existing function block will be replaced with the imported function block |

If the aforementioned checks are all completed successfully, the imported user function block will be added to the group of user function blocks in easySoft 7.

### 6.3.9 Replacing a user function block

This function makes it possible to replace a user function block that is being used in the project with a different user function block with an identical "Interface" configuration.

In order for this menu option to be available, a user function block call must be selected and the user function block must not be open for editing.

If there are any user function blocks with an "Interface" configuration that matches that of the selected function block and the latter is not open for editing, the "Replace user function block" dialog box will appear and the user function blocks that can be selected as replacements will be shown as a list in a pane.

You can then use the "Replace area" section to specify which user function block calls should be replaced:

- The selected user function block only
- All instances of the selected user function block in the current program
- All instances of the user function block in all programs

Clicking on the "Replace" button will replace the original user function block instance(s), i.e., the calls, contacts, and coils for the original user function block will be replaced with the selected user function block.

If there are no user function blocks with an "Interface" configuration that matches that of the selected function block available, or if they are open for editing, the following message will appear:

"There are no user function blocks that are suitable as a replacement, or they are currently open for editing."

### 6.3.10 Deleting a user function block

The function can be used to remove user function blocks from easySoft 7. Please note that you will only be able to delete user function blocks that are not being used in the project and that are not open for editing. If there are no user function blocks that can be deleted, the *Delete user function blocks* menu option will not be available.

The following options are available for deregistering / deleting a user function block:

1. *Program/Delete user function blocks… menu option*
2. *User function blocks folder in list of operands and function blocks/Delete user function blocks… context menu option*
3. *List of operands and function blocks <name>/Delete context menu option*

The following dialog box will appear if you use one of the first two options:

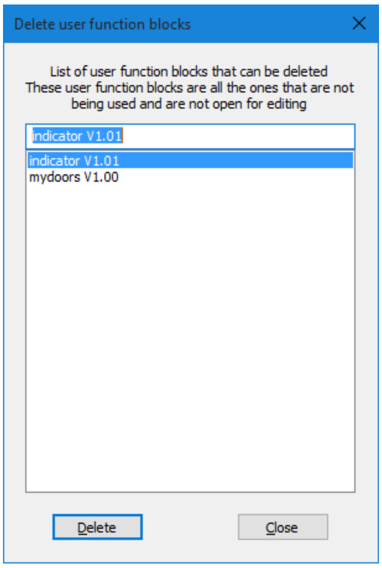



Figure 247: Delete user function blocks dialog box

You will get a list of all user function blocks that can be deleted, and can select any individual user function blocks you want. Once you select a function block and click on the Delete button, the function block will be deleted. After this, the user function block will no longer be part of easySoft 7 and will no longer be available in the *list of operands and function blocks/User function blocks folder*.

If you use the third option, the user function block you selected will be deleted and removed from the *list of operands and function blocks/User function blocks folder*.

### 6.3.11 Comparing user function blocks

The "Compare user function blocks..." menu option will be enabled as soon as you select a user function block. If the selected user function block is password-protected, you will have to enter the password.

Please note that you can only compare user function blocks that use the same programming language.

You can select whether you want to compare the user function block with a user function block registered in easySoft 7 or to one from a uf7 file (i.e., a user function block that has been previously exported). Accordingly, the following dialog box will appear:

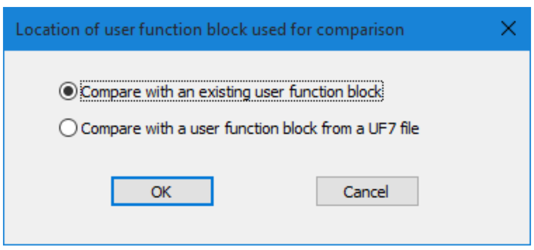



Figure 248: Location of user function block user for comparison dialog box

If you choose to compare the selected user function block with an existing user function block, a dialog box showing all available user function blocks with the same programming language will appear.

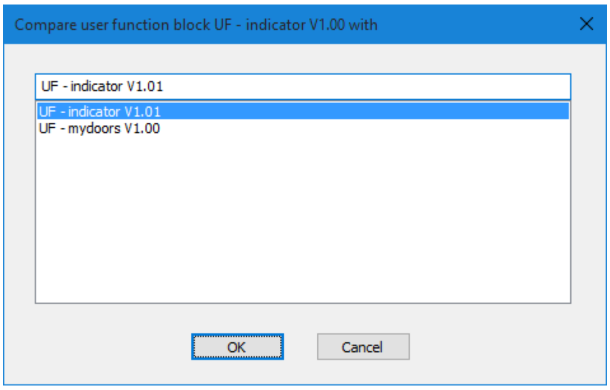



Figure 249: UF user function block

If you instead choose to compare the selected user function block with a previously exported user function block, the "Import user function block" dialog box will appear so that you can select a uf7 file.

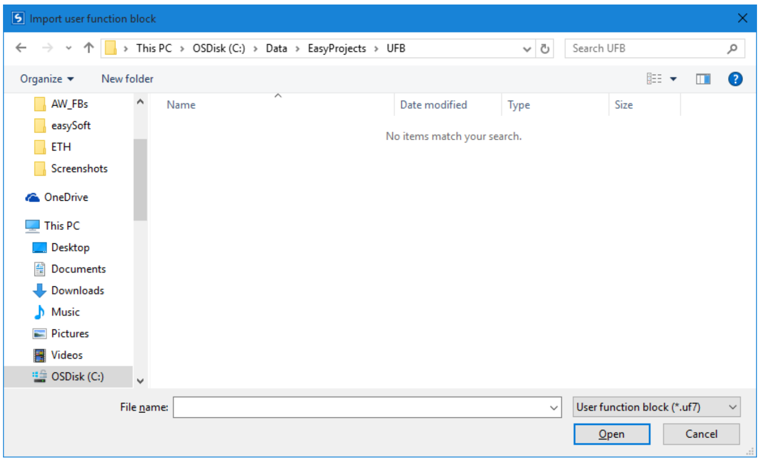



Figure 250: Import user function block

If a user function block is identical to the one being compared, or
if you are attempting to compare user function blocks that use different programming languages, you will get a message to this effect, after which you can select a different function block.

The comparison uses a text-based line-to-line comparison as a basis. The function units for each network will be grouped together in order to make the comparison easier to follow. The comparison itself will be visualized in a simplified ASCII character image.

Moreover, the gates and parallel branches in each network will be assigned a three-digit order number in ascending order based on where they are located in the network. You can use these numbers is order to identify how the gates/parallel branches are related to each other.

After the comparison, the results will be shown on your default HTML browser and saved to an output file. This output file will have the same name as the opened user function block and the HTML extension, and will be stored in the user's "My Documents" or "Documents" folder.

### 6.3.12 Printing a user function block

You can print both user function blocks that are being used in the project and user function blocks that are not being used in the project.

When you print out a user function block, the printout will contain all the parameters from the configuration dialog box, the program in the programming language used, and a list of cross references for the operands used.

This function comes with the option of viewing a page preview first.

## Opening a project with an existing user function block

If you open a project that has an existing user function block, the user function block will be automatically added to the list of available function blocks in easySoft 7. This also means that it will be available for other projects.

If you open a project that has a user function block and there is already another user function block with the same name in easySoft 7, a prompt to this effect will appear and you will have two options for solving the conflict:

1. You can cancel opening the project.
2. You can open the project, in which case the user function block in it will overwrite the user function block in easySoft 7.

To solve the conflict, you can also rename the user function block found in easySoft 7 and then open the project.

# 6.4 Timing and counter relay example

A warning light flashes when the counter reaches 10. In this example the function blocks C01 and T01 are wired in the standard circuit diagram and their inputs and outputs are defined.

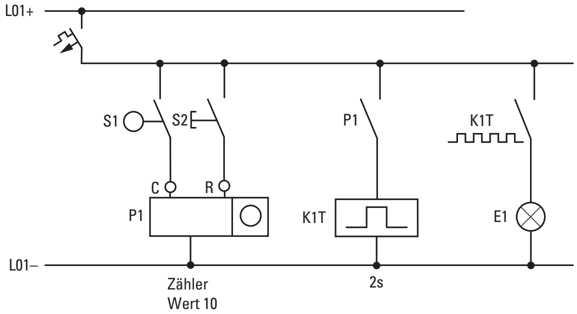



Figure 251: Hardwiring with relays

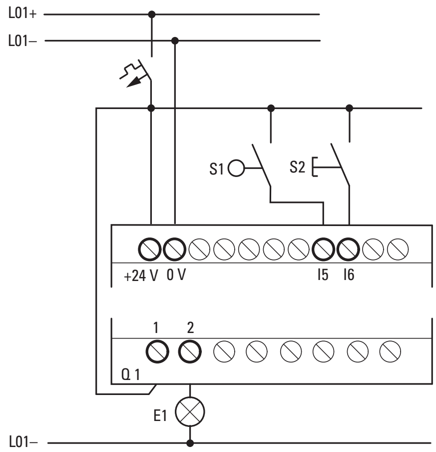



Figure 252: Wiring with EASY-E4-UC-..., for example

### Enter circuit diagram

▶ Enter the following circuit diagram while using the EDP programming language.

```
I 05-----------------------------[ C 01C

I 06-----------------------------[ C 01RE

C 01OF---------------------------[ T 01EN

T 01Q1---------------------------[ Q01
```

Figure 253: Wiring of counter and timing relay

### Entering function block parameters

If you enter the coils or contacts of a function block, the inputs/outputs of the function block are displayed that you can parameterize. The parameters can be entered via the function blocks menu.

The meaning of these parameters is explained under each function block type.

Entry:

The first part of the parameter set of a counter C01 is displayed.

- Move the cursor > over the + character to the value input behind `>SH`:
  `>SH` means: Function block input upper counter setpoint value.
  The + character means that the parameters of this timing relay can be modified using the PARAMETERS menu.
- Change the upper counter setpoint to 10:
  Use < > to move the cursor onto the tens digit.
  Use the ↑ and ↓ buttons to change the value of the digit.
- Press OK to save the value and ESC to return to the circuit diagram.

```
C 01 +
>SH +10
>SL +0
>SV +0
QV>+0
```

Figure 254: Enter parameter C01

Configuring the parameter for T01:

The timing relay works like a flashing relay. The function is set on the top right beside the number in the parameter display.

- The time base is set to the right of the “flashing” function. Leave the time base set to S for seconds.
- Move the cursor to the right over the + character in order to input the time SETPOINT value I1.

If you enter the same setpoint value at I1 and I2, the timing relay operates as a synchronous flasher.

The + character means that the parameters of this timing relay can be modified using the PARAMETERS menu.

- Confirm the value input with OK.
- Press ESC to leave entry.

```
T 01  n  S +
>I1 002,000
>I2 002,000
QV>
```

Figure 255: Enter ParameterT01

Testing the circuit diagram:

- Switch easyE4 to RUN operating mode and return to the program.

You can display every parameter set via the function relays menu.

- Move the cursor onto C 01 and press OK.

The parameter set for the counter is displayed with actual and setpoint values.

- Move the cursor ↓ downwards until you see the value QV.
- Switch input IS05. The ACTUAL value changes.

```
C 01 +
>SH +10
>SL +0
>SV +0
QV>+0
```

Figure 256: Testing the circuit diagram

If the ACTUAL and upper SETPOINT values of the counter are the same, the timing relay switches the warning light on and off every 2 seconds.

```
C 01 +
>SH +10
>SL +0
>SV +0
QV>+10
```

Figure 257: Testing the circuit diagram +10

Doubling the flashing frequency:

- Select the power flow display T 01 and change the constant of the setpoint time to 001,000.

When you press OK, the warning light will flash at twice the frequency.

```
T 01  n  S +
>I1 002,000
>I2 002,000
QV> 0.550
```

Figure 258: Doubling the flashing frequency

If the setpoint is a constant, it can also be modified via the PARAMETERS menu.

→ The actual value is only shown in RUN mode.

**See also**

→ Section "Timing and counter relay example", page 557
→ Section "CF - Frequency counter", page 287
→ Section "CH - High-speed counter", page 293
→ Section "CI - Incremental Counter", page 299

# 7. System settings

The "System settings" section groups together the various basic settings for the device, and accordingly can be used as a reference guide.

It is important to note where the relevant system setting can be configured, i.e., with the display on the EASY-E4-...-12...C1(P) under SYSTEM OPTIONS and/or only in easySoft 7 after selecting the device. Programming and integrating the easyE4 device into a group are also important within this context.

The following settings can only be configured with easySoft 7 as of this writing:


| | |
|---|---|
| **Connection to other devices** | |
| Setting up a NET | → page 625 |
| Modbus TCP | → page 632 |
| Setting up a web server | → page 646 |
| Web client | → page 652 |
| E-mail function | → page 670 |
| **Configuration** | |
| Download comments | → page 575 |
| Define program name | → page 577 |
| Retention function | → page 578 |
| Configuring the microSD card and device ID | → page 586 |

# 7.1 System options - Base device with display and buttons

The system options that can be configured on EASY-E4-...-12...C1(P) base devices include:

Table 74: *System options*

```
SECURITY
SYSTEM
MENU LANGUAGE
DELETE PROGR.
NET
ETHERNET
UPDATE
```

| | |
|---|---|
| Security | Access to the area used to assign a password and define password-protected areas<br>→ Section "Security – password protection", page 581 |
| System<br>Table 75: *System options\System*<br>DEBOUNCE<br>P BUTTONS ✓<br>RUN MODE<br>CARD MODE<br>LOAD CARD<br>INDICATOR<br>DEVICE ID<br>BOOT LOGO | Accessing the system settings:<br>Debounce, → Section "Debounce", page 574<br>P Buttons, → Section "P buttons", page 576<br>RUN Start, Card Start, → Section "Setting the startup behavior", page 571<br>Load Card, → Section "Configuring the microSD card and device ID", page 586<br>Display, Display settings, → Section "Display", page 563<br>Device ID, Device IDs, → Section "Device ID", page 564<br>Splash Screen, Used to set the display duration on the display, provided that a boot.bmp file has been stored on the memory card. → Section "Splash screen", page 565 |
| Menu language | Used to set the device menu language, → Section "Switch languages", page 570 |
| DELETE PROGR. | Deletes the program on the easyE4 from the device memory |
| NET | Used to configure a NET GROUP as a group of multiple devices, → Section "Setting up a NET", page 625<br>The submenu is only provided in English |
| ETHERNET | Used to configure the Ethernet settings on the device,<br>→ Section "Ethernet", page 568<br>The submenu is only provided in English |
| UPDATE | Operating System /Firmware Update for easyE4 expansion devices from Version 1.10.<br>→ Section "Update", page 569 |

# 7.2 Display

This menu can be used to configure the settings for the display.

Table 76: *System Options\System\Display*

```
BRIGHTNESS1        100
BRIGHTNESS2        50
TIMEOUT:           10m
COLOR:             0
```

| | |
|---|---|
| BRIGHTNESS1 | Display brightness when the device is being operated<br>Default value: 100,<br>can be changed in increments of 10 |
| BRIGHTNESS2 | Brightness for sleep mode<br>Default value: 50,<br>can be changed in increments of 10<br>Value: 0 will cause the display to be switched off in sleep mode |
| TIMEOUT | Used to set the time in minutes or seconds<br>after which the display will go to sleep if the easyE4 is not being actively operated |
| PAINT | Relevant to easyE4 remote operation<br>Color value of 0 – 15,<br>This setting will affect the way the device is displayed, e.g., in easySoft 7 or on the web server |

## 7.3 Device ID

Used to set / enter the individual device IDs for transferring programs.

Table 77:
*SYSTEM OPTIONS\DEVICE ID*

```
DEVICE-ID
XXX XXX XXX
```

Entering a device ID of <000 000 000> will disable the device ID check and the program ID check. If you do this, it will be possible to transfer all program types to the base device through a microSD memory card or through easySoft 7 regardless of whether an ID has been set in the program itself.

## 7.4 Splash screen

Once an boot.bmp image is stored on the microSD memory card, this setting can be used to specify how long the image will be displayed, in seconds, before the status display is shown.

Table 78: *System Options\Splash Screen*

```
DISPLAY TIME
3 s
```

**See also**

→ "Setting a splash screen for the EASY-E4-...-12...C1(P) display" section, page 1

# 7.5 NET

This submenu can be used to configure the NET addresses for the easyE4 device.

The other stations, i.e., easyE4 devices, must also be configured accordingly in order for it to be possible to establish a connection.

The item in the last line on the status display 1 will indicate whether there is an active NET connection.

Table 79: Net configuration on device

Table 80: *Main menu*

| |
|---|
| STOP ✓ RUN |
| PARAMETERS |
| SET CLOCK |
| CARD |
| INFORMATION |
| SYSTEM-OPT. |
| PROGRAM |

Table 81: *System options*

| |
|---|
| SECURITY |
| SYSTEM |
| MENU LANGUAGE |
| DELETE PROGR. |
| NET |
| ETHERNET |
| UPDATE |

Table 82: *System options\Net*

| | |
|---|---|
| NET-GROUP: | 00 |
| NET-ID: | 00 |
| BUSDELAY: | 000 |
| REMOTE RUN | |

The submenu is only provided in English

- Select the NET-GROUP you want using the cursor keys.
- Set the device's NET-ID.
- Select the network setting you want.

### NET-GROUP

Used to select the group for the selected base device.

| | |
|---|---|
| 0 | Base device running in standalone mode with the relevant I/O expansions (if any),<br>no NET group |
| 1-10 | Possible NET-GROUP |

### NET-ID

Used to assign a group device number from the NET GROUP to the selected base device.

| | |
|---|---|
| 0 | Base device running in standalone mode with the relevant I/O expansions (if any) |
| 1-8 | Available device IDs in the NET-GROUP |

### Bus delay

The bus delay is used to define the time after which a station on the NET will send its data to other stations.

This bus delay needs to be adjusted as appropriate for the number of stations and the values being transmitted. Please note than an excessively short bus delay will result in data collisions.

The permissible value range for the bus delay is 10 ms to 255 ms.

Cyclical data will be sent every 10 ms or when there is a data change, but not before the bus delay has elapsed. Using the default value of 60 ms will normally be sufficient to prevent transmission overloads.

### Remote RUN

If this field is enabled, the NET stations of a group with NET-IDs 02 through 08 will take their current RUN or STOP operating mode from the NET station with NET-ID 1.

### See also

→ Section "Setting up a NET", page 625

# 7.6 Ethernet

This submenu can be used to configure the addresses for the easyE4 device.

The other station must also be configured accordingly in order for it to be possible to establish a connection.

The last line on the status display will indicate whether there is an active connection.

New easyE4 base devices will come with the Auto IP setting configured by default. In order to configure the settings differently on the EASY-E4-...-12...C1(P), use the menu structure and go to *System Options\Ethernet*

The submenu is only provided in English

Table 83: Ethernet configuration on device

Table 84: *Main menu*

```
STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM-OPT.
PROGRAM
```

Table 85: *System options*

```
SECURITY
SYSTEM
MENU LANGUAGE
DELETE PROGR.
NET
ETHERNET
UPDATE
```

Table 86: *System options\Ethernet*

```
ADDRESS MODE
IP ADDRESS
SUBNET MASK
GATEWAY ADDRESS
DNS SERVER
```

▶ Use the cursor buttons to enter the device's IP address.

Table 87: *System Options\Ethernet\IP Address*

```
IP-ADDRESS
000.000.000.000
```

▶ Select the network setting you want.

Table 88: *System Options\Ethernet\Address mode*

```
AUTO IP        ✓
DHCP
STATIC IP
```

**See also**

→ Section "Establishing an Ethernet connection and transferring a program", page 100

# 7.7 Update

easyE4 expansion devices are equipped with a new operating system in this submenu. A firmware update is possible for devices of version 1.10 or higher.

Updates for the operating system are made available by Eaton Industries GmbH, Bonn in the Download Center - Software under Firmware Updates as *.zip files.

Download Center - Software
http://www.eaton.eu/software/Firmware Updates/easy
http://www.eaton.eu/software/OS Updates/easy

Observe the documents belonging to the update in the download center.

▶ Unzip the required operating system file matching the easyE4 expansion device "*.FW" on the microSD memory card.

The easyE4 expansion device must be connected to the base device with the EASY-E4-CONNECT1 plug connector.
The number of the easyE4 expansion is determined based on the position after the base devices, starting with 1 from the left. The maximum number 11 can be assigned to an expansion in the assembly block.
An update must be carried out separately for each expansion device.

Table 89: Updating expansion devices

Table 90: *Main menu*

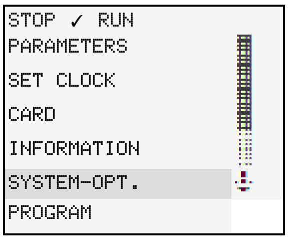



Table 91: *System options*

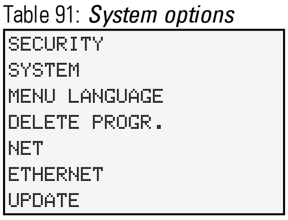
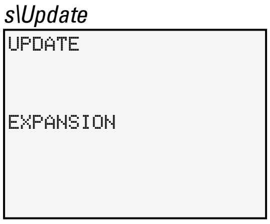


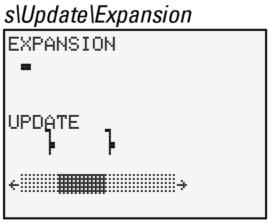

Table 92: *System options\Update*




▶ First, select the number of the easyE4 expansion in the block; 1 to 11 are possible.

▶ Select the corresponding operating system file.

Table 93: *System options\Update\Expansion*




The easyE4 base devices can only be updated via the microSD memory card.

**See also**

→ Section "Firmware update base device", page 1
→ Section "Firmware update expansion device", page 1

# 7.8 Switch languages

The menus are available in several languages.

### 7.8.1 Selecting a menu language on a base device with a display

- Go to the main menu.
- Go to SYSTEM OPTIONS\MENU LANGUAGE.
- Select one of the available languages.
- Confirm with the OK button.
- Exit the menu with the ESC button.

Table 94: *System Options\Menu Language*

| | |
|---|---|
| ENGLISH | |
| DEUTSCH | ✓ |
| FRANCAIS | |
| ESPAÑOL | |
| ITALIANO | |
| NEDERLANDS | |
| POLSKI | |
| ČESKÝ | |
| PORTUGUÊS | |
| РУССКИЙ | |
| TÜRKÇE | |
| ROMÂNĂ | |
| MAGYAR | |

The language will change when you exit the menu.

### 7.8.2 Setting the menu language in the easySoft 7

Only possible with easySoft 7.

You can change the user interface language in easySoft 7.

- Click on the Options menu.
- Select the Languages menu option.
- Click on the language you want.

Before the change can be applied, the software will need to be closed and restarted.

→ Dutch is only available as a menu language on the base device.

## 7.9 Setting the startup behavior

The startup mode defines how the easyE4 device will respond when the supply voltage is applied.

**EASY-E4-...-12...CX1(P)**

Devices without a display automatically start in RUN mode.

After power up, the easyE4 device switches directly to RUN mode if it contains a valid program.
If, on the other hand, there is no program in the easyE4 device, it will remain in STOP mode.

If the device is connected via Ethernet, it can be configured.
A *.e70 program can be loaded via a memory card.

**EASY-E4-...-12...C1(P)**

The startup behavior of devices with a display can be set.

With the SYSTEM OPTIONS/SYSTEM/RUN START menu option on the device or in the program in easySoft 7 by using the RUN start option.
This option will be stored on the device together with the program.

→ Section "Overview of switch-on behavior", page 98

**Startup behavior**

The startup behavior is an important aid during the commissioning phase.
The circuit diagram which EASY-E4-...-12...C1(P) contains is not yet fully wired up or the system or machine is in a state which EASY-E4-... is not permitted to control.

If voltage is applied to the easyE4 device in this case, it should not be possible to drive the outputs, i.e., it is not allowed for the outputs to be set immediately when the easyE4 is switched on.

### 7.9.1 Enabling / disabling the RUN START option

Only possible on base devices with a display.

#### 7.9.1.1 Configuration on base devices with a display

In order to configure it, the program must be stopped.

STOP ✓ RUN

Operating mode changes may be protected with a password.

- Go to the main menu.
- Go to SYSTEM OPTIONS\SYSTEM.
- Select the RUN START menu option.
- Press the OK button to enable and disable the option.

| Display | Status | |
|---|---|---|
| RUN START ✓ | Active | The program will start as soon as the device is switched on (the device will switch to RUN mode). |
| RUN MODE | Disabled | The program will need to be started separately (the device will remain in STOP mode). |

→ The RUN START option will be enabled by default on the EASY-E4-..., as well as after a factory reset.

**What will happen if the program is deleted**

The startup mode setting is a device function and will be retained even if the circuit diagram is deleted.

**Upload/download to memory card or PLC**

The setting will be retained when a valid program is transferred.

### 7.9.2 Enabling / disabling the CARD START option

The mode for starting up with a memory card is intended for applications in which it is necessary for it to be possible to change programs easily and quickly by changing the memory card.

If the program on the memory card is different to the program in the easyE4 device, the program from the card will be loaded on power up first of all and then started in RUN mode. If the programs only contain different function block SETPOINT values (constants), the program on the memory card is not loaded.
The program is retained in the device and is started. If there is no circuit diagram on the memory card the device will remain in STOP mode. For a detailed description of the effect of this option, see → "Switching on the device with the microSD memory card", page 1.

Default setting CARD START inactive

#### 7.9.2.1 Configuration on base devices with a display

In order to be able to do this, the program must be in STOP mode. If it is not, the device will point this out.

- ▶ Go to the main menu.
- ▶ Go to SYSTEM OPTIONS\SYSTEM.
- ▶ Select the CARD START menu option.
- ▶ Press the OK button to enable and disable the option.

If there is a checkmark ✓ next to the menu option, the program will be loaded from the memory card and applied as soon as the easyE4 device is switched on.
If there is no check mark, the current program will be kept.

#### 7.9.2.2 Configuring the easySoft 7

You can enable and disable this option in easySoft 7.

- ▶ Select the device you want from the catalog in the Project view
- ▶ Click on the System settings tab.

Go to the Memory card / device ID section and find the checkbox for Card start.

- ▶ ☑ To turn on, enable the checkbox by clicking on it.
- ▶ ☐ To turn off, disable the checkbox by clicking on it.

**See also**

→ Section "Configuring the microSD card and device ID", page 586

# 7.10 Debounce

easyE4 is factory set to evaluate input signals with an input delay, the so-called debounce function. This ensures that any contact bouncing of switches and push-buttons is masked out.

There are certain applications in which detecting very brief input signals is necessary. In order to ensure that this will happen in those cases, you can disable the input delay if necessary.

### 7.10.1 Configuring input debouncing on a base device with a display

- ▶ Go to the main menu.
- ▶ Go to SYSTEM OPTIONS\SYSTEM.
- ▶ Select the DEBOUNCE menu option.
- ▶ Press the OK button to enable and disable the option.

If there is a checkmark ✓ next to the menu option, input debouncing will be enabled. If there is no check mark, it will be disabled instead.

### 7.10.2 Configuring input debouncing in easySoft 7

You can enable and disable the input delay in easySoft 7.

- ▶ Select the device you want from the catalog in the Project view
- ▶ Click on the System settings tab.

Go to the System settings section and look for the Debounce checkbox.

- ▶ ☑ To turn on, enable the checkbox by clicking on it.
- ▶ ☐ To turn off, disable the checkbox by clicking on it.

## 7.11 Download comments

Only possible with easySoft 7.

All the comments created in easySoft 7 can be loaded onto the device together with the project.

- Select the device you want from the catalog in the Project view
- Click on the System settings tab.

Go to the Comments section and look for the Download comments checkbox.

- ☑ To turn on, enable the checkbox by clicking on it.
- ☐ To turn off, disable the checkbox by clicking on it.

# 7.12 P buttons

"P buttons" are the cursor buttons on easyE4 devices with a display and keypad.

When working with EASY-E4-...-12...C1(P) devices, you can use the buttons as a contact in your circuit diagram.

→ In order to prevent accidental operation, the buttons will not be automatically enabled.

## 7.12.1 Configuring the P buttons on a base device with a display

In order to configure it, the program must be stopped.

STOP ✓ RUN

Operating mode changes may be protected with a password.

▶ Go to the main menu.

▶ Go to SYSTEM OPTIONS\SYSTEM.

▶ Select the P BUTTON menu option.

▶ Press the OK button to enable and disable the option.

If there is a checkmark ✓ next to the menu option, input debouncing will be enabled. If there is no check mark, it will be disabled instead.

## 7.12.2 Configuring the P buttons in easySoft 7

You can enable and disable the P buttons in easySoft 7.

▶ Select the device you want from the catalog in the Project view

▶ Click on the System settings tab.

Go to the System settings section and look for the checkbox for P buttons and an input field.

▶ ☑ To turn on, enable the checkbox by clicking on it.

▶ ☐ To turn off, disable the checkbox by clicking on it.

### Max. cycle time [ms]

This setting can be used to define the maximum cycle time you want. The default setting is 1000 ms, and the value range is 0 to 1000 ms. The device will switch to STOP mode as soon as a program cycle exceeds the configured maximum cycle time.

▶ Enter the maximum cycle time in [ms] into the input field.

If you do not enter a value into the input field, the default setting will be used instead.

## 7.13 Define program name

Only possible with easySoft 7.

You can name your program in easySoft 7.

- ▶ Select the device you want from the catalog in the Project view
- ▶ Click on the System settings tab.

Go to the Program name section and look for the input field.

- ▶ Enter the name you want into the text field so that it will be applied to the program.

## 7.14 Retention function

Only possible with easySoft 7.

It is a requirement of system and machine controllers for operating states or ACTUAL values to have retentive settings. What this means is that the values will be retained until the next time the ACTUAL value is overwritten.

There are two input fields (for the start and end values of the retention range) each for markers and for the following function blocks.

*Project view/ System settings tab*

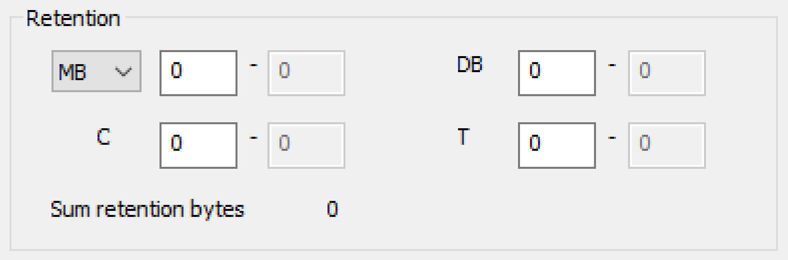



Figure 259: Screenshot of Retention section that can be found in the System settings tab in the Project view (including sample values)

Value range for the function blocks, instances that can be stored retentively:

- C - Counter relay : 01…32
- CH - High-speed counter : 01…04
- CI - Incremental counter : 01…02
- DB - Data function block : 01…32
- T - Timing relay : 01…32

For more information, please refer to the description for the relevant function block.

Marker value ranges:

- MB : 1 …512
- MW : 1…512
- MD : 1…256

The values from the input field will be automatically converted to MB marker bytes.

**Retention bytes**

The entire retentive marker range for an easyE4 must not exceed 400 bytes. The total of the retention bytes for the main program and user function blocks (UF) will be displayed in the Project view, in the System settings tab. If the retentive marker range exceeds 400 bytes, a red negative number will be shown in the Free field in order to indicate this.

**Retain retention during transfer**

Retentive ACTUAL values on the device are deleted:

- By any program change in the circuit diagram or function block diagram followed by its transfer to the device.
- When the program is deleted with the *Communication/Program/Configuration/Delete device* button
- By any change of the retentive range via *Project view/System settings/Retention*.
- By any changes to the parameters for the remote markers of a visualization device.
- When deleting the device from the workspace of the project view.

The following exception applies to retentive markers:

☑ Marker contents

If this option is enabled, the contents of the existing retentive marker range will be retained when the program is transferred. The ACTUAL marker values are retained.
In order for this to work, however, the marker range defined as being retentive must remain unchanged.

☑ Function block contents

If this option is enabled, the contents of the existing retentive operand range will be retained when the program is transferred.
In order for this to work, however, the function block defined as being retentive must remain unchanged.

### 7.14.1 Retention in the easySoft 7

You can configure the retention function both for markers and for function block contents in easySoft 7.

▶ Select the device you want from the catalog in the Project view

▶ Click on the System settings tab.

Go to the

- Retain retention during transfer
  section and look for the checkbox for Marker contents and the checkbox for Function block contents
- Retention
- Retention bytes

▶ ☑ To turn on, enable the checkbox by clicking on it.

▶ ☐ To turn off, disable the checkbox by clicking on it.

To configure the corresponding retention as necessary, enable the Marker contents and/or Function block contents checkbox.

Define the ranges that should remain retentive by selecting them and entering the corresponding values.

→ These ranges should be used exclusively for values that are required in order to be able to start the system back up after a restart. Please keep potential unforeseen and/or undesirable consequences in mind!

Retention bytes will show the amount of memory needed as you enter the values you want.

▶ Check whether there is enough memory.

# 7.15 Security – password protection

Configuring password settings and password-protected areas is only possible on easyE4 devices with a display or must alternatively be configured in easySoft 7.

Password protection can be used to lock access to various areas.

→ At least one area must be protected.
In the default setting the circuit diagram is selected.

## 7.15.1 Configuring the password on a base device with a display

### Defining password-protected areas

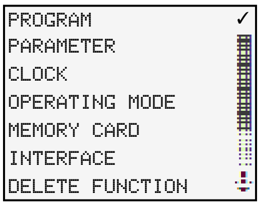
To define the areas that should be protected with a password, follow the steps below:

- Go to the main menu.
- Go to SYSTEM OPTIONS\SECURITY\AREA.
- Select the desired range
- Press the OK button to enable and disable the option.

If there is a checkmark ✓ for the area next to the scrollbar, this means that access to the corresponding area will be protected with a password prompt.

If there is no checkmark, it will be possible to access the area freely.

Table 95:
*System options\Security\Area*




The submenu shows the device areas that can be protected.

| | |
|---|---|
| PROGRAM | The password is applied to the PROGRAMS as well as function blocks that are not enabled. This area also prevents the transfer of a circuit diagram from and to the memory card. |
| PARAMETER | The PARAMETERS menu is protected. |
| CLOCK | Date and time are protected with the password. |
| OPERATING MODE | It is not possible to change the operating mode from RUN to STOP and vice versa using the operating buttons of the device. |
| MEMORY CARD | Access to the microSD memory card will be protected. |

| | |
|---|---|
| INTERFACE | Protects against access to the Ethernet interface of this device. Data exchange via the net is not affected.<br>→ Take into account the restricting effect of a protected interface if you have to reset the easyE4 device. |
| DELETE FUNCTION | When this function is not activated, the question “DELETE PROG?” will appear if the password entry is entered incorrectly four times. This question does not appear if you protect the this area. However, it is no longer possible to make changes in protected areas if you forget the password. |

→ At least one of the following areas must be protected: Program, Parameters, Clock, Operating Mode, or Memory Card. If you do not select any of these areas, "Program" will be selected automatically.
The PROGRAM area will be selected by default when using the device's factory settings.

**Assigning a password**

- Go to the main menu.
- Go to *SYSTEM OPTIONS\SECURITY*.
- Select the PASSWORD menu option.

Table 96:
*System options\Security*

```
PASSWORD
RANGE
```

You can use any numbers or letters for the six-character password. Special characters and umlauts are not permitted.

```
PASSWORD:   0XXXXX
ABCDEFGHIJ KLM
NOPQRSTUVC-XYZ
         0
1234567890
```

Figure 260: Password assignment

The first password character will flash.

- Select the first letter or number for the password.
- Confirm the entry with the OK pushbutton.
- Repeat these steps for the remaining characters in the password.

You can cancel at any time with the ESC button.

**Enabling the password:**

- Place the cursor anywhere inside the password.
- Press the OK button.

The password submenu will be displayed.

```
CHANGE PW
ACTIVATE PW✓
```

Figure 261: Password submenu

- Select the ACTIVATE PW menu option.
- Confirm the password with the OK pushbutton.

The password will be enabled in order to → Section "Defining password-protected areas", page 581.

**Changing a password**

- Press the OK button on the easyE4 to open the main menu.
- Go to SYSTEM OPTIONS\SECURITY\PASSWORD.

If a password has been assigned, the submenu for the password will be displayed.

- Select the CHANGE PW menu option.
- Enter the password.

The submenu for changing the password will be displayed.

```
NEW PASSWORD:
PASSWORD:  0 XXXXX
ABCDEFGHIJ KLM
NOPQRSTUVC-XYZ
          O
1234567890
```

Figure 262: Submenu for changing the password

To assign a new password, follow the same steps outlined in → Section "Assigning a password", page 582

**Removing password protection**

To disable password protection, assign a password of <000000> .

#### 7.15.1.1 What happens if you forget your password or enter the wrong password?

When you enter the wrong password, there will be a short period during which you will be locked out. You can try and enter the password again once this period elapses.

→ If the DELETE FUNCTION area is password-protected, you will be able to enter a password as many times as you want.

Starting from the fifth wrong attempt, the base device will display a delete prompt.

- ESC button: Abort, circuit diagram, data or password are not deleted.
- OK button: Circuit diagram, data and password are deleted.

If you no longer know the exact password, you can press OK to unlock the protected easyE4 device.
The saved program and all function relay parameters will be lost.

### 7.15.2 Configuring the password in easySoft 7

When working on a project, you can set a password for the <program>.e70 and select the areas you want to protect in easySoft 7.

- Select the device you want from the catalog in the Project view.
- Click on the Register Security.

Go to the Password entry section. This section will feature a checkbox for each area that can be protected with the password, as well as an input field for the password.

*Project view\Security*


Figure 263: Assigning a program password

- To turn on, enable the checkbox by clicking on it.
- To turn off, disable the checkbox by clicking on it.

You can use any numbers or letters for the six-character password. Special characters and umlauts are not permitted.

- Enter the password into the input field.
- Confirm the password by entering it again.

If you enable the checkbox, the password will be shown in cleartext.

When you save the project, the password will be enabled for the <program>.e70.

The Delete password button will reset the password

## 7.16 Configuring the microSD card and device ID

Only possible with easySoft 7.

- Select the device you want from the catalog in the Project view
- Click on the System settings tab.

Go to the Memory card / device ID section and look for the Card start checkbox, the Allow overwriting via card checkbox, and the numeric input field.

- ☑ To turn on, enable the checkbox by clicking on it.
- ☐ To turn off, disable the checkbox by clicking on it.

If ☑ Card start, the device will access the microSD when it is switched on.

If ☑ Allow overwriting via card, it will be possible for the program that is found on the microSD card to overwrite the program stored on the easyE4.

You can enter a six-digit number as a Program/device ID into the input field.

→ This ID ensures that a program will be transfered on the easyE4 device only if the IDs match.

The system will use the device ID and program ID you entered in order to check whether it is permissible to transfer the selected program to the corresponding base device.

**See also**

→ Section "Transferring programs from and to the microSD memory card", page 187
→ Chapter "1 -Memory card", page 1
→ Section "Device ID", page 564
easySoft 7 Help, Communication view

# 7.17 Time and Date setting

easyE4 devices feature a real-time clock (RTC) with a date and time functionality. This real-time clock forms the basis for all the time-based operations controlled with the easyE4.
When combined with the HW, HY or WT, YT function blocks, this real-time clock makes it possible to implement the functionality of a weekly timer and year time switch.

The AC manufacturer function block can be used to implement a sunrise time and sunset time functionality.

### 7.17.1 Time and date on a base device with a display

- Go to the main menu.
- Go to SET CLOCK.
- Select the DATE & TIME menu option.

Table 97: *Set Clock-\Time/Date*

```
   DD-MM-YYYY
FR 13.08.2018
   12:03:04
```

Select the display format you want in the first line.

- Use the ˄ ˅ cursor buttons to scroll through the available formats.
- Select the format you want.

| | |
|---|---|
| DD-MM-YYYY | |
| DD/MM/YYYY | Day.Month.Year |
| DD.MM.YYYY | |
| MM/DD/.YYYY | Month.Day.Year |
| YYYY-MM-DD | Year.Month.Day |
| YYYY.MM.DD | |

The display will change accordingly.

- Use the ‹ › cursor buttons to jump to the individual input positions in the date and time format.
- Set the values by means of the cursor keys ˄ ˅.
- Confirm the entry with the OK pushbutton.

There are additional configuration options available in the SET CLOCK menu.

### DST setting DST

- Go to the main menu.
- Go to SET CLOCK.
- Select the SUMMER TIME menu option.

Table 98: *Set clock\Daylight saving time*

```
NO              ✓
DST
US
RULE
```

The following will be available for selection: None, CEST,US, and Rule. A checkmark ✓ will indicate which setting is currently selected.

If you select "None," no daylight saving time schedule will be applied. "CEST" will apply the Central European Summer Time schedule, while "US" will apply the daylight saving time schedule used in the United States. Finally, selecting "Rule" will enable you to define your own custom schedule.

Table 99: *Set Clock\Daylight Saving\Rule*

```
SUMMERTIME START
SUMMERTIME END
```

- In "Rule,"
  select when you want daylight saving time to start and when you want it to end.
  The easyE4 will apply your settings and will automatically change the clock on the dates you selected.

### Setting radio clock

Alternatively, you can also have the system synchronize its clock with a radio time signal. If this functionality is enabled, the real-time clock on the device will be overwritten as soon as a suitable radio time signal is received.

- Go to the main menu.
- Go to SET CLOCK.
- Select the RADIO CLOCK menu option.
- To enable it, use the ⮝ ⮟ cursor buttons to select the YES option.
- Use the ⮜ ⮞ cursor buttons to select the input you want.
- Use the ⮝ ⮟ cursor buttons to define the value you want.
- Use the same steps to enter the offset from the radio time signal time.
  The unit for this offset is minutes, while each individual increment is 5 minutes.

Table 100: *Set Clock\Radio Clock*

```
RADIO
CLOCK
ACTIVE      : YES
INPUT       : I001
OFFSET      : +000'
```

### Setting up the astronomical clock

You can also set the real-time clock using the astronomical clock. This astronomical clock calculates both sunrise and sunset times based on latitude and longitude.

The settings in this submenu will apply globally to all 32 possible instances of the → Section "AC - Astronomic clock ", page 272 function block in the user program.

- Go to the main menu.
- Go to SET CLOCK.
- Select the ASTRON. CLOCK menu option.
- Use the ⌃ ⌄ ‹ › cursor buttons to select a digit in the coordinate input line.
- Use the ⌃ ⌄ cursor buttons to define the value you want.
- Repeat the steps above to enter the offset between the time zone and UTC. The unit for this offset is minutes, while each individual increment is 5 minutes.

→ LAT: Latitude coordinate
LNG: Longitude coordinate
(±) is entered with N-North/S-South and E-East/W-West in the first input character.
Format: (±) ddd.ddddd, in decimal degrees

- Pressing the › button will switch the input line to DMS with degrees, minutes, and seconds.

Table 101: *Set clock\astron. clock*

```
ASTRON.       CLOCK
LAT      N000. 0000000
LNG      E000. 0000000
OFFSET:       +000'
```

The input entered on the easyE4 device will be overwritten every time a program is transferred. This means that in order to always have the coordinates available on the device, the coordinate information must be stored in easySoft 7 for the program. To do this, you can transfer the modified program to easySoft 7 and save it there if you want to apply this information to the location data in the project.

**Example**

Settings for time zone in Bonn (UTC+1 hour) in decimal degrees

Table 102: *Set clock\astron. clock*

```
ASTRO        CLOCK
LAT    N050. 734012
LON    E007. 082808
OFFSET       : +060'
```

and DMS

Table 103: *Set clock\astron. clock*

```
ASTRO.        CLOCK
LAT     N050° 44'02"
LON     E007° 04'58"
DIFFERENCE    : +060'
```

### 7.17.2 Setting time and date in the easySoft 7

You can enable and disable the time in easySoft 7.

▶ Select the device you want from the catalog in the Project view.

▶ Click on the Register Clock.

Go to the SNTP synchronization section and look for the checkbox for Synchronize clock via SNTP.

▶ ☑ To turn on, enable the checkbox by clicking on it.

▶ ☐ To turn off, disable the checkbox by clicking on it.

▶ Select whether you want to enter a URL or the IP address for the SNTP server.

▶ Enter the address into the corresponding field.

Alternatively, you can synchronize the clock with a radio time signal.

Go to the Radio clock synchronization section and look for the Synchronize clock via radio (DCF77) checkbox.

▶ ☑ To turn on, enable the checkbox by clicking on it.

▶ ☐ To turn off, disable the checkbox by clicking on it.

▶ Use the drop-down menus to select the input you want to use and the time offset, in [min], at the location where the easyE4 is being used.

Finally, you can use the Daylight saving time section to configure the corresponding settings as needed.

▶ ◉ Enable the
None, CEST, US setting
or

The Rule setting if you want to define your own custom schedule.

- Click on the Modify time zone... button to specify the time zone where you are located.

Go to the Time zone section to select the time zone corresponding to the location where the easyE4 device is being used.

Clicking on the Modify time zone... button will open the Time zone and geographic coordinates dialog box.

Use the City drop-down menu to select one of the available cities.

You can click on the New button to add a new option based on geographic coordinates. The new option will be saved on the local system.

**NET-GROUP**

If you are using a NET GROUP, you can select a easyE4 device from the group that will serve as the time reference for all the other devices in the NET GROUP.

Go to the NET synchronization section and look for the checkbox for Synchronize clock via NET.

- ☑ To turn on, enable the checkbox by clicking on it.
- ☐ To turn off, disable the checkbox by clicking on it.
- Select the NET-ID (station number) corresponding to the device in the NET GROUP that will serve as the time reference for the other devices.

**Application**

Say that only one easyE4 device in the NET group features a radio clock and that you want this device to serve as the time reference for all the other devices in the group. In this case, you would enter the NET-ID of the device here.

**See also**

Timer modules

→ "HW - 7-day time switch (Hour Week)", page 218
→ "HY - Year time switch (Hora Year)", page 228
→ "WT - Weekly timer (WeekTable)", page 268
→ "YT - Year time switch (Year Table)", page 261
→ "AC - Astronomic clock ", page 272

# 8. easyE4 Inside

## 8.1 Program execution

When using the LD or FBD programming language, the program will be executed as follows:

- The program will start by reading the hardware's input states and writing them to the image table register. After this, it will run through network 01 in its entirety and not only process all function blocks and logic circuitry, but also write the state of all mapping assignments (Q, M, etc., and function blocks) to the image table register. It will then run through the next network (if any networks are jumped over, they will not be run through). Once the last network has been run through, the resulting output states will be transmitted to the hardware The cycle will then start again.

In the Programming language ST

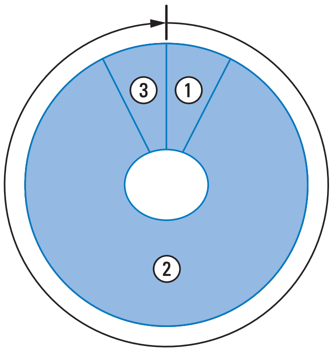
- The program will start by reading the hardware's input states and writing them to the image table register. After this, it will execute the statement and instruction list from top to bottom and modify the image table register every time there is a mapping assignment (if any statements or instructions are jumped over, they will not be executed). The cycle will then start again.

When using the EDP (easy Device Programming) programming language

- This programming language is the same one that can be use for programming directly on the base device. The way in which this program will be executed is identical to the way programs are executed on the existing easy500, easy700, and easy800 devices.

In conventional control systems, a relay or contactor control processes all the rungs in parallel. The speed with which a contactor switches is thus dependent on the components used, and ranges from 15 to 40 ms for relay pick-up and drop-out.

Table 104: Cycle timeeasyE4




| ① Read process image | ② Process circuit diagram and function block diagram | ③ Write process image |
|---|---|---|
| Local inputs<br>Function block inputs<br>Markers<br>Diagnostics information | | Local outputs<br>Function block outputs<br>Markers<br>NET stations outputs |

NET sta-
tions
inputs

During this time, the easyE4 device passes through six segments in succession.

If the operands of the inputs and outputs are addressed in the easyE4, the signal states of the digital inputs/outputs are not scanned but a memory range in the device system memory is accessed. This memory range is called the process image. The process image is divided into two sections: the process image for the inputs and the process image for the outputs.

**Segment 1 - 4**

The easyE4 device evaluates the contact fields within the first four segments. The evaluation starts in the first segment in circuit diagram line 1 and continues from top to bottom until circuit diagram line n is reached.

The easyE4 device then moves to the next contact segment and continues to evaluate from top to bottom until it has reached the last contact in the fourth segment. During this process it checks whether contacts are switched in parallel or in series and saves the switching states of all contact fields.

**Segment 5**

In the fifth segment the easyE4 device assigns all coils in one pass, from the circuit diagram line 1 - n, with the new switch states from the process image of the outputs.

**Segment 6**

In the sixth segment which is outside of the circuit diagram, the function blocks present in the function block list are evaluated.

The easyE4 device uses this sixth segment in order to:

- process the existing function blocks. The output data of a function block is immediately up-to-date after it has been processed. The function blocks are processed by the easyE4 device in the order of the function block list (→ *FUNCTION BLOCKS menu*).

The following requirements must be fulfilled when using particular function blocks:

- Contact the “outside world”
  Output relays Q 01 to Q… are switched and inputs I 1
  to I… are re-read.
- to exchange NET data if this easyE4 device receives new read data or provides new send data ( on ).
- to copy all new switching states to the process image.

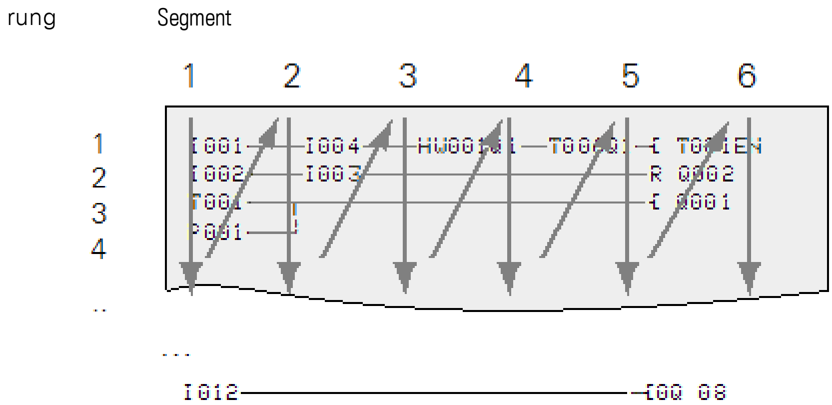



Figure 264: How the EDP evaluates circuit diagrams and function blocks

## 8.2 Transfering an existing circuit diagram

Existing easy.e60 programs can be imported with easySoft 7.

When importing existing programs / projects, you will be able to select either the EDP or LD programming language:

EDP programs will be imported in their entirety, and their flow structure will be compatible with previous devices.

If the program / project is instead imported into LD, the first mapping assignment will be to an intermediate marker. Once the last mapping assignment to an intermediate marker is completed, the intermediate markers will be mapped to the actual M, Q, etc. function block operands. This ensures that the program will keep the same flow structure as the previous devices.

easySoft 7 will output an import log that specifies how the inputs, outputs, and markers have been rewired.

→ If a project featuring an easyE4 features MFD-CP8/10 stations as well, the MFD devices will be shown as "other" NET stations.
easySoft 7 will use the previous devices and the operands used as a basis in order to optimize the easyE4 hardware and the new <xyz>.e70 program.

# 8.3 Device information

Device information is provided in the

*Information* menu both for service purposes and in order to make it possible to identify the device's performance characteristics.

Following data is displayed:

The submenu is only provided in English

ACTUAL CONFIG - Shows the device configuration

- NET GROUP: The NET group number, on a single line, e.g., 00
- NET-ID: The device's station number, on a single line, e.g., 00
- MAC ADDRESS: (MAC address of the device), two display lines e.g. 0022C712343E
- DEVICE NAME: e.g.:EASYE4-12UC1 Assigned DNS device name for the ETHERNET network → chapter "8 System settings", page 561
- IP-ADDRESS: xxx.xxx.xxx.xxx
- SUBNET MASK: xxx.xxx.xxx.xxx
- GATEWAY ADDRESS: xxx.xxx.xxx.xxx
- DNS SERVER: xxx.xxx.xxx.xxx
- WEB SERVER (Enabled / Disabled)
- HTTP PORT
- MODBUS TCP (Enabled / Disabled)

SYSTEM - Shows the operating system version

- E4- : Part number
- B : 0068(build version)
- OS : 1.00(Version)
- CRC : 60268(checksum)

## 8.4 NET network

The NET functionality via Ethernet was created in order to simplify communications between easyE4 base devices while simultaneously ensuring that it would be possible to import existing easy800 projects.

A NET group can be made up of up to eight easyE4 base devices. Within the group, the easyE4 base devices can communicate with each other. If, however, you want devices to be able to communicate across groups, you will need to use a coordinator device that allows the easyE4 base devices from the various groups to communicate with each other via Modbus.

Ten NET groups (groups 1 through 10) can be run on a single Ethernet network at one time (this is the equivalent of 80 easyE4 base devices).

| Ethernet network | | | | | | |
|---|---|---|---|---|---|---|
| ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ |
| Module | Module | Module | Stations… | Module | Module | Module |
| NET-ID 1 | NET-ID… | NET-ID 8 | NET-ID… | NET-ID 1 | NET-ID… | NET-ID 8 |
| | NET group 1 | | NET group… | NET group 10 | | |

The following list shows the operands that can be used within a group by every device:

- (n = NET-ID 1 .. 8)
- n SN 01 - 32 [Bit]
- n RN 01 - 32 [Bit]
- PT 01 - 32 (PUT) [double word]
- GT 01 – 32 (GET) double word]
- n N 01 - 512[ Bit]
- n NB 01 - 64 [Byte]
- n NW 01 - 32 [Byte]
- n ND 01 - 16 [double word]
- Synchronize clock (settings)

**Examples**

Station 1 sending a bit to station 2

| NET-ID1 | | NET-ID 2 |
|---|---|---|
| 2 SN 15 | → | 1 RN 015 |

Station 3 sending a double word to station 8 via PT16

| NET-ID1 | | NET-ID 2 |
|---|---|---|
| pt16 | → | GT 01<br>Parameter<br>NET-ID 1<br>PT 16 |

Station 4 sending a network marker [bit and word] to all stations.

| NET-ID4 | | NET-ID 2 | NET-ID 5 | NET-ID 7 |
|---|---|---|---|---|
| N 125 | → | 4 N 125 | 4 N 125 | 4 N 125 |
| NW30 | → | 4 NW 30 | 4 NW 30 | 4 NW 30 |

This basic principle applies to all network markers in all data formats:

→ Network markers overlap in the various data formats

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| N1-8 | N9-16 | N17-24 | N25-32 | N33-40 | N41-48 | N49-56 | N57-64 |
| NB1 | NB2 | NB3 | NB4 | NB5 | NB6 | NB7 | NB8 |
| NW1 | | NW2 | | NW3 | | NW4 | |
| ND1 | | | | ND2 | | | |
| N65-72 | N73-80 | N81-88 | N89-96 | N97-104 | N105-112 | N113-120 | N121-128 |
| NB9 | NB10 | NB11 | NB12 | NB13 | NB14 | NB15 | NB16 |
| NW5 | | NW6 | | NW7 | | NW8 | |
| ND3 | | | | ND4 | | | |

etc.

### Life signs NET stations

In order to make it possible for all NET stations within a group to be able to know whether NET stations important to them are still communicating, each station cyclically sends a heartbeat every second (1 s). If a heartbeat is not received, the corresponding error bit ID01 – 08 will be set to "1" until a heartbeat is detected.

### Remote Run

If this flag is set, the NET stations in a group with NET-ID 02 through 08 will follow the current operating mode of the NET station with NET-ID 1 (RUN or STOP).

### Bus Delay

The bus delay defines the delay that a station on the NET will use when sending its data to other stations.

This bus delay must be adjusted in line with the number of stations and the values being transmitted. If it is too short, there will be data collisions and the Ethernet will be limited to transmitting NET communications exclusively.

The value range for the bus delay is 10 ms to 255 ms.

The following rule of thumb applies:

- Case A: When using PUT/GET and network markers:
  - Bus delay in ms = (number of NET stations - 1) * 4 * 2 + 6

- Case B: If only network markers are being used:
  - Bus delay in ms = (number of NET stations - 1) * 2 * 2 + 6

The following table can be used as a convenient guide for configuring the setting:

| Number of modules: | Delay with put/get ms | Delay without put/get ms |
|---|---|---|
| 2 | 14 | 10 |
| 3 | 22 | 14 |
| 4 | 30 | 18 |
| 5 | 38 | 22 |
| 6 | 46 | 26 |
| 7 | 54 | 30 |
| 8 | 62 | 34 |

→ If you are no longer able to connect to the NET stations via Ethernet with easySoft 7, set the bus delay as high as possible for your application. To do this, you will need to disconnect each device from the Ethernet and use easySoft 7 to change the bus delay point by point.

## 8.5 Operating states easyE4

easyE4 devices feature various operating states.

**Switched off**- no supply voltage available

**Powered up**

- If there is no program on the base device, the base device will remain in the STOP operating mode and it will not be possible to execute any programs.
- If a program has been loaded onto the device, the base device will remain in the STOP operating mode until it switches to RUN. When the base device is in STOP mode, the program will not be executed. Connected expansion devices will communicate with the base device as long as there are no configuration errors, and all the outputs on all devices will have a state of 0 (OFF). In addition, it will be possible to communicate with easySoft 7 via Ethernet.
- The base device can be switched to the RUN operating mode with the menu or through easySoft 7. In this case, the program will be executed and the outputs will be switched on and off as per the program logic. Existing communication services such as NET, Modbus, web servers, etc. will be running, and it will be possible to use them.

# 8.6 Device easyE4 time responses

### 8.6.1 Time behavior of the inputs and outputs

The reaction time, which is measured from the reading in of a digital input signal to the setting of the associated output, is determined by the timing characteristics of the easyE4 inputs and outputs as well as the size and design of the circuit diagram.

#### Input delay (debounce)

The time required from reading the inputs to switching the contacts (setting the outputs) in the circuit diagram can be increased on the easyE4 base device using an input delay, the so-called I-DEBOUNCE, please refer to → Section "Debounce", page 574

This function is useful, for example, in order to ensure a clean switching signal despite contact bounce.

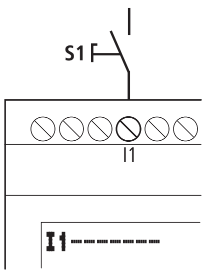



Figure 265: easyE4 input assigned a switch

EASY-E4-DC-... devices and EASY-E4-AC-... devices operate with physically different input voltages and therefore differ in the length and evaluation of debounce times.

### 8.6.2 Base device timing

#### 8.6.2.1 Delay time for operation with DC power supply

**Delay time with debounce activated**

When debounce is activated, the delay time for DC signals is 20 ms.

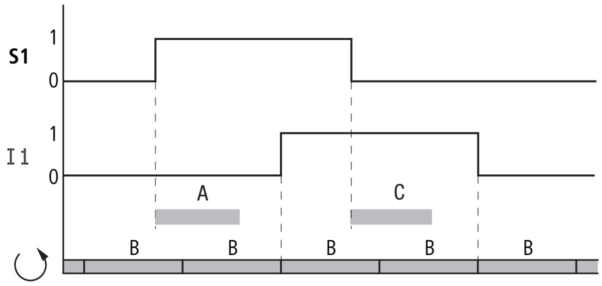



Figure 266: Delay times for evaluating an DC input signal and an activated debounce

The times for A and C depend on the specific device.

For more information, please refer to the device data sheet → Section "Technical data", page 699

An input signal S1 must therefore be 1 on the input terminal for at least 20 ms before the switch contact will change from 0 to 1 (A). If applicable, this time must also include the cycle time (B) since a easyE4 device does not detect the signal until the start of a cycle.

When the DC signal drops from 1 to 0 with debounce active, the same delay time (C) of 20 ms applies. For this the input signal S1 must be 0 at the input terminal.

### Delay time with debounce deactivated

When I-DEBOUNCE is deactivated, the delay time (A) for DC signals on the input for easyE4 base devices is reduced.

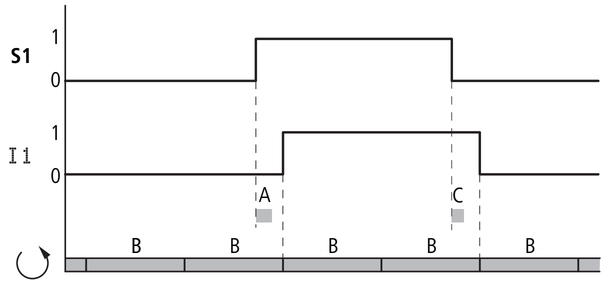



Figure 267: Switching behavior with debounce deactivated

The times for A and C depend on the specific device.

For more information, please refer to the device data sheet → Section "Technical data", page 699

→ When debounce is deactivated ensure that input signals are free of noise. The easyE4 device responds to very short signals.

→ To allow reliable recognition and processing of the input signal in the user program, it must be applied stably for a certain duration, the length of which depends on the circuit diagram processing cycle time.

#### 8.6.2.2 Delay time for operation with AC power supply

In the case of AC inputs, the easyE4 device will read the input signal every period with a scan cycle of $t_{SC}$.
The scan cycle depends on the supply frequency.

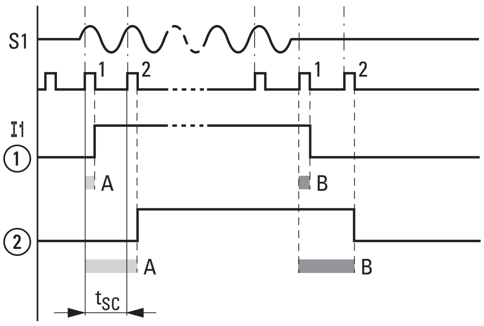



Figure 268: Delay times for evaluating an AC input signal

① without I-DEBOUNCE and

② for an activated I-DEBOUNCE

**Delay time with debounce activated**

If the DEBOUNCE function is activated, the easyE4 device checks each period whether a positive half-wave is present at an input terminal during two successive scan cycles $t_{SC1}$ (1st and 2nd scan pulse at A). If the easyE4 device registers two positive half waves in succession, it switches the appropriate input (contact) internally from 0 to 1.

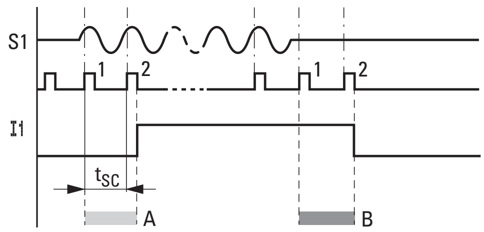



Figure 269: Switching behavior of AC input signal with DEBOUNCE activated

Accordingly, the typical input delay caused by the debounce option is at least 40 ms (50 Hz). The cycle time needs to be added to this, since a easyE4 device will not detect the signal until the start of a cycle. Conversely, the input is switched off if the easyE4 device does not detect any half-cycles twice in succession (1st and 2nd pulse at B).

- Switch-on delay (normally):
  - I1 … I8: 45 ms (38 ms)
- Off-delay (normally):
  - I1 … I8: 45 ms (38 ms)

The corresponding values for 60 Hz are given in brackets.

### Delay time with debounce deactivated

If the debounce option is disabled, the delay time will decrease. If a positive half wave is detected, the easyE4 device will directly switch the corresponding input (contact) from 0 to 1 (A) internally.

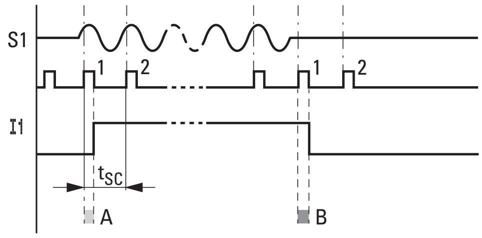



Figure 270: Switching behavior of AC input signal with DEBOUNCE deactivated

If a positive half-cycle is not detected the easyE4 switches off the contact (B).

- Switch-on delay (normally):
  - I1 … I8: 25 ms (21 ms)
- Off-delay (normally):
  - I1 … I8: 25 ms (21 ms)

The appropriate values for 60 Hz are given in brackets.

→ For more information on how to change the delay times, please refer to → Section "Time behavior of the inputs and outputs", page 603

### 8.6.3 Timing characteristics of expansion devices

You can use the EASY-E4-CONNECT1 connector to connect the easyE4 base device to up to 11 expansions and assemble them into a single device block. This connector will not only establish the mechanical connection between the devices, but also the electrical connection – easyConnect.

When the outputs and inputs on expansion devices are written to and read respectively via easyConnect, the corresponding process will not be synchronized with the program cycle. In fact, if the easyConnect cycle is more than twice as fast as the program cycle, the inputs and outputs will be refreshed with every program cycle.

If, on the other hand, the easyConnect cycle is slower than half the program cycle, the inputs and outputs may be refreshed after two program cycles instead.

The easyConnect cycle time will be between 10 ms and 15 ms depending on the specific configuration.

#### 8.6.3.1 Delay time for AC expansion devices

EASY-E4-AC-8RE1(P) AC expansion devices behave like the AC base devices.

EASY-E4-AC-16RE1(P) AC expansions support multiple phases, resulting in an additional delay.

- Switch-on delay (normally):
  - I1…I8: 39 ms (32 ms)
- Off-delay (normally):
  - I1…I8: 39 ms (32 ms)

The appropriate values for 60 Hz are given in brackets.

For more information, please refer to the device data sheet → Section "Technical data", page 699

# 9. Operating system diagnostic messages

The easyE4 devices output information on their own operating state via ID (operands) diagnostics contacts. This information can be evaluated in the circuit diagram and shown in the status display 2 on the display.

Diagnostic operands are used in order to obtain information regarding operating states in the program. (this functionality can only be used when the base device is in RUN mode). If the event in question has occurred, the operands will have a state of **1**.

| Operand | Event |
|---|---|
| ID01 | There are more than two devices in this NET group and the NET is active.<br>The NET station 1 is not present: |
| ID02 | There are more than two devices in this NET group and the NET is active.<br>The NET station 2 is not present: |
| ID03 | There are more than two devices in this NET group and the NET is active.<br>The NET station 3 is not present: |
| ID04 | There are more than two devices in this NET group and the NET is active.<br>The NET station 4 is not present: |
| ID05 | There are more than two devices in this NET group and the NET is active.<br>The NET station 5 is not present: |
| ID06 | There are more than two devices in this NET group and the NET is active.<br>The NET station 6 is not present: |
| ID07 | There are more than two devices in this NET group and the NET is active.<br>The NET station 7 is not present: |
| ID08 | There are more than two devices in this NET group and the NET is active.<br>The NET station 8 is not present: |
| ID09 | The DCF77 radio time signal has been enabled in the program. A radio signal is not being detected at the selected input. |
| ID10 | This diagnostic bit is set if one of the following time synchronization operations could not be completed successfully:<br>• "Synchronize clock via NET"<br>• "SNTP synchronisation"<br>• Date and time<br>• DCF77 radio clock<br>Using the SC function block will not result in this error message or in it being cleared. |
| ID11 | If the device cannot communicate via Ethernet. |
| ID12 | Arithmetic blocks have their own error output that is used to signal whether there is a numeric underflow/overflow, e.g., as a result of attempting to divide by zero. When using the ST programming language, this diagnostic operand will additionally be set if such an error occurs. |
| ID13 | If the base device is being operated with one or more expansion devices, this diagnostic operand will indicate whether required devices are disconnected from the easyConnect bus or are not being detected, e.g., in the event of a power failure on an expansion device. |
| ID14 | Transistor outputs in the base device are experiencing an overload or a short circuit; the outputs will be switched off and then checked again after 30 seconds. |
| ID15 | – |
| ID16 | – |

## 9. Operating system diagnostic messages

| Operand | Event |
|---|---|
| ID17 | – |
| ID18 | – |
| ID19 | There is an interrupt overload. One or more interrupt function blocks are being used, and the interrupt function block sequence is overloading the easyE4's controller. Not all interrupt function blocks can be run correctly. |

Additional diagnostic messages for the expansion devices can be assigned to diagnostic operands ID25 through ID96 based on the relevant device characteristics.

**Transistor output example**

Transistor outputs on expansion devices EASY-E4-DC-8TE1(P), EASY-E4-DC-16TE1(P)

When there is a short circuit or overload at an output, the DIAG diagnostic message can be applied to a diagnostic operand. This means that when the event occurs, the operands will assume a state of **1**.

**Example Analog expansion device**

| Name on device | Event |
|---|---|
| DIAG | General diagnostic indicating that a diagnostic event is present |
| DIAG 1 | Current input overloaded (current greater than 23 mA), excessive voltage |
| DIAG 2 | Analog output overloaded, excessive current, load too small<br>Open wire at at least one current input (I < 4 mA) |
| DIAG 3 | The measuring range has been physically exceeded at an input |
| DIAG 4 | The measuring range has been physically fallen below at an input, e.g., the current is < 4 mA for a measuring range of 4–20 mA. |

**Temperature expansion example**

| Name on device | Event |
|---|---|
| DIAG | General diagnostic indicating that a diagnostic event is present |
| DIAG 1 | Configured measuring range exceeded at at least one temperature input, or connection cable discontinuity |
| DIAG 2 | Configured measuring range fallen below at at least one temperature input, or a short-circuit has occurred |

## 9.1 Transistor outputs (overload / short-circuit)

The base and expansion devices' transistor outputs are thermally protected against overloads and short circuits. If the temperature inside the four transistor modules is too high, the outputs will be switched off. If the temperature returns to the operating range and the outputs are driven, the transistors will switch back on.

The overload / short circuit fault scenario can be detected for the base device with operand ID14.

ID14 = **1**, error

Expansion devices feature a "DIAG" output that you can assign to operands ID25 through ID96 for each device.

## 9.2 Diagnostics buffer

Only possible with easySoft 7.

During online operation, the diagnostic buffer will be shown in the Communication view. Additional information on this in the easySoft 7 Help

## 9.3 LED status messages on the device

For diagnostic purposes, base devices without a display feature two LEDs, while expansion devices feature one LED. The light patterns on these LEDs are accordingly used to indicate the corresponding device's status.

LED POW/RUN base device

The POW/RUN LED indicates the state of the POW power supply as well as the STOP or RUN state.

| | |
|---|---|
| Off | Malfunction or no supply voltage |
| Green, continuous light | Supply voltage OK, RUN mode |
| Green,<br>Flashing, 1 Hz | Supply voltage OK, STOP mode |
| Green,<br>Flashing, 4 Hz | Fault at one of the expansions,<br>between the easyE4 device and the EASY-E4-CONNECT1 connector |

### LED ETHERNET/NET (base device only)

| | |
|---|---|
| Off | No Ethernet cable connected; supply voltage off<br>The port is not enabled; the easyE4 device does not have an IP address |
| Yellow, continuous light | Ethernet cable connected |
| Green, continuous light | There is an IP address, but the NET has not been configured |
| Red, continuous light | Ethernet conflict or error, e.g.: duplicate IP address, address collision |
| Green, flashing,<br>2 flashes, pause, etc. | NET data flow working; one or more NET stations missing |
| Green, flashing,<br>1 flash, pause, etc. | NET data flow working; all NET stations working |

### LED POW/RUN status expansion unit

| | |
|---|---|
| Off | Malfunction or no supply voltage |
| Green, continuous light | Supply voltage OK, address assigned, expansion bus working correctly |
| Green,<br>Flashing, 1 Hz | Supply voltage OK, no data exchange with base device |
| Green,<br>Flashing, 3 Hz | Supply voltage OK, no data exchange with base device,<br>diagnostic bit will be set, device not working |
| Green,<br>Flashing, 10 Hz | Device waiting for firmware update |
| Green,<br>Flashing, 0.5 Hz | Firmware update in progress |

# 10. Connection to other devices

New easyE4 base devices will come with the Auto IP setting configured by default. In order to configure the settings differently on the EASY-E4-...-12...C1(P), use the menu structure and go to *System Options\Ethernet*

Table 105: Ethernet addresses on the device

Table 106: *Main menu*

| |
|---|
| STOP ✓ RUN |
| PARAMETERS |
| SET CLOCK |
| CARD |
| INFORMATION |
| SYSTEM-OPT. |
| PROGRAM |

Table 107: *System options*

| |
|---|
| SECURITY |
| SYSTEM |
| MENU LANGUAGE |
| DELETE PROGR. |
| NET |
| ETHERNET |
| UPDATE |

Table 108: *System options\Ethernet*

| |
|---|
| ADDRESS MODE |
| IP ADDRESS |
| SUBNET MASK |
| GATEWAY ADDRESS |
| DNS SERVER |

Table 109: *System Options\Ethernet\IP Address*

| |
|---|
| IP-ADDRESS |
| 000.000.000.000 |

▶ Use the cursor buttons to enter the device's IP address.

Table 110: *System Options\Ethernet\Address mode*

| | |
|---|---|
| AUTO IP | ✓ |
| DHCP | |
| STATIC IP | |

▶ Select the network setting you want.

Prerequisites that must be met in order to be able to access an easyE4 control relays:

- The PC must have an Ethernet port that is free and has been configured
- The PC's Ethernet port must be configured for auto-IP.
- The easyE4 control relays must be connected to the PC with a standard Ethernet cable featuring an RJ45 connector

**CAUTION**
**INTERFERENCES**
The values specified in the technical data, as well as the device's electromagnetic compatibility (EMC), cannot be guaranteed if the following are used: unsuitable cables, improperly assembled and terminated cables, and/or wiring that does not conform to the applicable standards.
Only use cables assembled and terminated by professionals.
The cables being used must be assembled and terminated as required by the port/interface description in this document.
When wiring the devices, follow all instructions regarding how to wire the corresponding port/interface.
All general Directives and standards must be complied with.

Only possible with easySoft 7.

## Compatibility rules for going ONLINE

As soon as a connection is established between easySoft 7 and a device (i.e., as soon as you are ONLINE), easySoft 7 will check the extent to which the physical device configuration matches the device selection in the project view.

Certain differences are allowed. In fact, if the physical device matches the device model in the project view but with slightly different characteristics, the devices will be categorized as being compatible. Within this context, devices are compatible in the following cases:

- Device model with display and device model without display
- Device model with screw terminals and device model with push in terminals

If there are differences, the devices will be shown in color in the Project view.

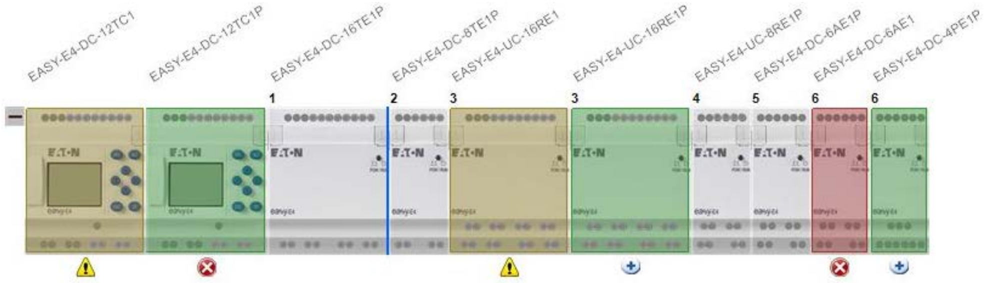



Figure 271: ONLINE Project view with devices colored differently based on their compatibility

Within this context, the following colors are used:

None — The physical device matches the device in the Project view e.g EASY-E4-DC-16TE1P .

Green — Expansions / devices found online that are not found in the configuration

The physical device is not found in the project view.

→ If the device number of the device to the left is the same, this indicates that the physical device was found instead of the configured device to its left.

Example: EASY-E4-DC-12TC1P is physically present, but EASY-E4-DC-12TC1 is configured at the corresponding spot in the Project view.
Example: EASY-E4-DC-4PE1P is physically present, but EASY-E4-DC-6AE1 is configured at the corresponding spot in the Project view.

Yellow — Replaced online with compatible expansions / devices

Example: EASY-E4-DC-12TC1 is configured in the project; the physically present device is an EASY-E4-DC-12TC1P

Red — Expansions / devices that are missing online and that are only found in the configuration

Either the device configured in the Project view is not physically present or it is not compatible with the configured device.
Example: EASY-E4-DC-6AE1 is configured in the Project view, but an EASY-E4-DC-4PE1 is physically present

If a device is not recognized in the Communication view, this indicates that an older version of easySoft 7 is being used and that the physical device is not found in the device catalog. In this case, you will need to install a newer software version.

The plausibility check will output any relevant compatibility errors and/or warnings based on the compatibility rules.

## Establish connection to device

Only possible with easySoft 7.

In easySoft 7, the connection to the device always has to be established in the Communication view.

By default, the easyE4 base device will be set to Auto IP and have a NET ID of 0.

Prerequisites that must be met in order to be able to access an easyE4 control relays:

- The PC must have an Ethernet port that is free and has been configured
- The PC's Ethernet port must be configured for auto-IP.
- The computer and the device must be connected to each other with an Ethernet cable – please refer to → "Connecting the Ethernet cable", page 76

▶ Open easySoft 7 and click on the Communication button.

▶ Now expand the Connection section in order to show the corresponding buttons.

The connection to the device will be offline.

▶ Under IP profiles, click on the Search… button.

Opens the Search for devices dialog box.

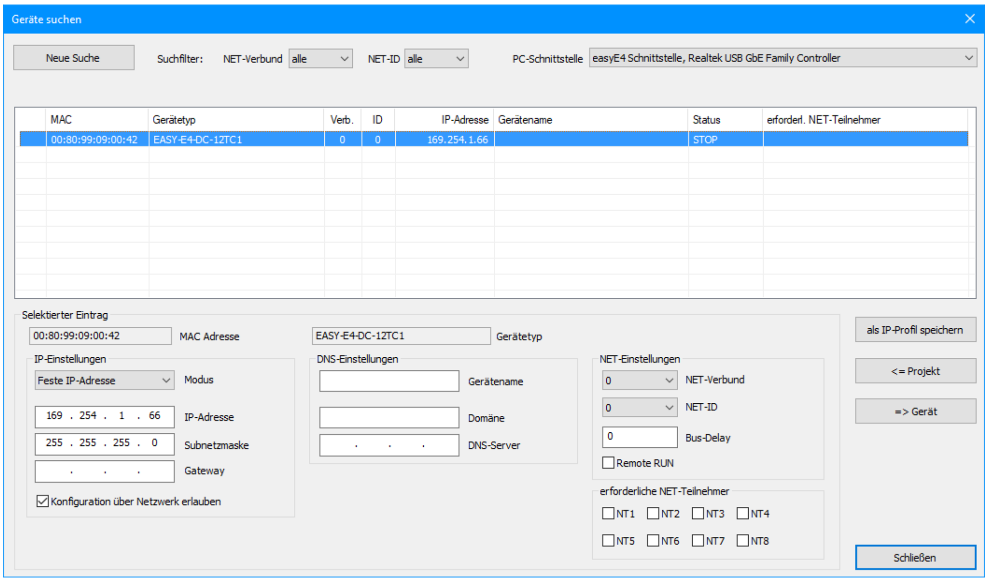
▶ Check the selected PC interface (Ethernet) for your computer in the PC interface drop-down menu.

▶ Select the "all" search filters in the NET group and NET-ID drop-down menus.

▶ Click on New search

Your PC interface will search for all reachable easyE4 control relays. Any devices that are found will be shown in the table as follows:

The Selected item section will show all the project parameters for the easyE4 base device.



- Select the row with the device to which you want to establish a connection.
- Click on the Save as IP profile button.
- Click on Close to close the dialog box

The IP profile will appear in the drop-down menu under "Interface."

- Use the Interface drop-down menu to select the IP profile you just saved.
- Use the Device drop-down menu to select the "Local" option.
  (New devices will not have a program, and accordingly will not have a NET ID either.)
- Click on the Online button to establish a connection.
- If a device is locked with a password, the Password dialog box will appear so that you can unlock it. Simply enter the corresponding password and confirm it.

The connection to the device will be established and the status line will show "ONLINE."

**Details for the table in the Search for devices dialog box**

| Column | Description |
|---|---|
| First column | Errors and Alarms |
| ? | Inconsistent entries on the device |
| ! | There are at least duplicate NET IDs |
| x | The device cannot be configured, since the "Enable configuration via network" option is disabled. |
| • | Indicates that there is currently a connection between the computer and the device. Accordingly, changes cannot be made to the device's IP settings. |
| MAC | The easyE4 base device's MAC address (fixed) |
| Device Type | (fixed) |
| Grp. | NET group (if any) |
| ID | The easyE4 base device's NET ID (if any) |
| IP Address | The easyE4 base device's IP address (as per the device's Ethernet settings) |
| DeviceName | If there is no device name in the data record that is currently selected, a new connection profile will be automatically created with the device's current IP address.<br>If there is a device name, the user will be able to select whether the new profile should be generated based on the current IP address or on the device name. If changes have already been made to the current selected data record but have not yet been transferred to the device, the attempt to generate a new profile will be aborted with the following message:<br><br>"Please transfer the modified configuration to the device first. Otherwise, obsolete parameters will be saved in the new IP profile." |
| Status | The easyE4 base device's operating status: (RUN/ STOP) |
| Required NET card | If the device has a program and is being operated in a NET group or the devices already have the relevant NET settings configured |

### Possible messages in the Search for devices dialog box

The Search for devices dialog box may show the following messages while a connection is being established:

| Message | Remedy |
|---|---|
| The configuration cannot be modified in the RUN device status! | Only relevant if you want to make a change in the "Selected item" section:<br>▶ Use the device menu to bring the device to the STOP operating status. |
| Please transfer the modified configuration to the device first. Otherwise, obsolete parameters will be saved in the new IP profile. | If you have made a change in the Selected item section (such as changing the device name), then you will first need to transfer the project to the device in order to avoid an inconsistency between the project in easySoft 7 and the project on the device.<br>▶ Click on the =>Device button in the Search for devices dialog box.<br>▶ Then click on the Save as IP profile button. |
| The device configuration is locked and you cannot make changes to it! | The device configuration in the Selected item section cannot be modified.<br>▶ Go to the *Project view/Ethernet tab* and enable the Enable configuration via network option.<br>▶ Go to the *Communication view/Connection section* and click on the Online button.<br>▶ Click on the PC => Device button in the Program / Configuration section. |

| Message | Remedy |
|---|---|
| | The project will be transferred to the device.<br>▶ Click on Offline.<br>▶ Click on Search under IP devices.<br>You will now be able to make changes in the Selected item section in the Search for devices dialog box. |
| No devices matching the selected search filter setting were found. | ▶ Check that the module rack<br>▶ Check to make sure that the correct PC interface, NET group, and NET ID are selected.<br>▶ Check to make sure that the IP addresses of the easyE4 base device and computer fall within the same number group. For more information, please refer to → "Basic information on assigning IP addresses", page 100. |

## Terminating the connection to the device

▶ In order to terminate the online connection, click on the Offline button in the Connection section.

The connection to the device will be terminated. The status line will show "OFFLINE."

## Setting up a connection to multiple devices on the NET

Before the first time a connection is established, easyE4 devices in a NET group will not know which NET ID and which parameters they should use in order to establish a connection. There are three ways to set up this connection:

1. → "Connection parameters and program on the device", page 622: Loading the program, with the NET ID and Ethernet settings, onto each device.
2. → "Connection parameters on device", page 623: Using the Search for devices dialog box to load the NET ID and Ethernet settings onto each device.
3. Device menu directly on the device: Configuring the NET ID and Ethernet settings directly on each device.

### Connection parameters and program on the device

If you have created a project with multiple easyE4 devices, the parameters for establishing a connection with every easyE4 device should be configured using the corresponding settings in the *Project view/Ethernet tab*. A program must have been created for the easyE4 device.

In order to transfer these settings to the easyE4 device in the NET group, follow the steps below:

▶ In the Project view work pane, select the first device in the project.

▶ Search for the devices in the NET group,
select the device that should correspond to the first device in the project from the list of found devices, and
go ONLINE, → Section "Establish connection to device", page 618

▶ In the Connection section, click on the PC => Device button.
The Select a NET station dialog box will appear.

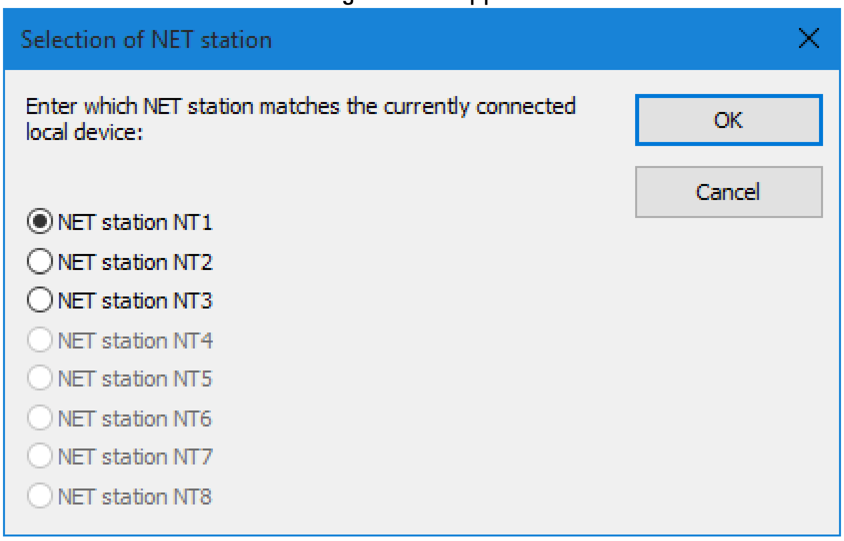



Figure 272: Selection of NET station

▶ Select the NET station. All the NET stations in the project will be available.

The program and all project settings (i.e., the NET ID and Ethernet settings) for the selected NET station will be loaded onto the easyE4 device.

- In the Project view work pane, select the next easyE4 device in the NET group.
- Search for the devices in the NET group,
  select the next device that should correspond to the next device in the project from the list of found devices, and
  go ONLINE, → Section "Establish connection to device", page 618.
- In the Connection section, click on the PC => Device button.
- Select the NET station.

Repeat these steps for every device that you want to configure in the project.

*Project view*

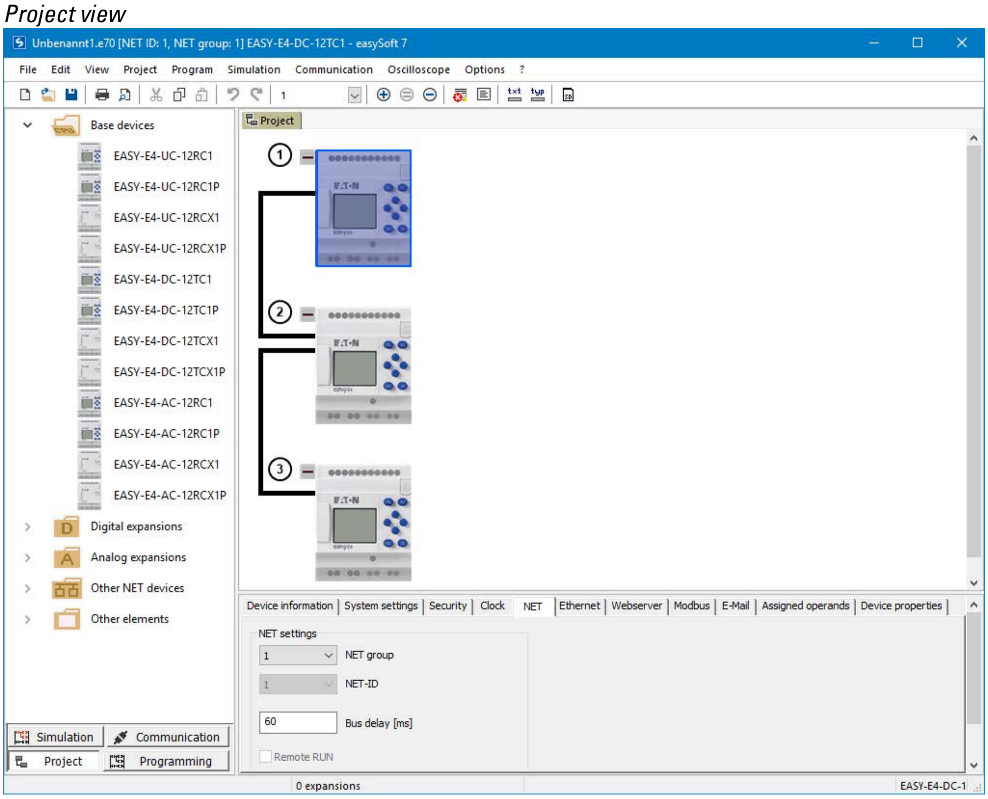



Figure 273: NET configuration with project and program

### Connection parameters on device

The computer can use easySoft 7 to establish a connection without a project or program and download the parameters for establishing a connection to each easyE4 device.

However, the ☑ Enable configuration via network option must be enabled on the device. This is only possible if the project has been loaded, with the option enabled, onto the device at least once – please refer to Enable configuration via network New.

In order to transfer these settings to the easyE4 device in the NET group, follow the steps below:

- Search for the devices in the NET group and
  select the device that should correspond to the first device in the project from the list of found devices → Section "Establish connection to device", page 618
- Configure the parameters you want for the device in the Selected item section under the list
  (the system settings for Ethernet and NET; please refer to → Section "System settings", page 561).
- Click on the => Device button.

The parameters for establishing the connection (i.e., the Ethernet settings) will be loaded onto the easyE4 device.

- Select the next device that should correspond to the second device in the project from the list of found devices → Section "Establish connection to device", page 618
- Configure the parameters you want for the device in the Selected item section under the list
  (the system settings for Ethernet and NET; please refer to → Section "System settings", page 561).
- Click on the => Device button.

Repeat these steps for every device that you want to configure in the project.

## Taking the Ethernet and NET configuration from the device

- Search for device → Section "Establish connection to device", page 618
- Select the device you want from the list of devices found.
- Click the <= Project button.
- Select the NET station you want in the "Select a NET station" dialog box.
- Confirm your selection with OK.

The NET station you selected in easySoft 7 will receive the parameters for establishing a connection from the device. To check them, go to the *Project view/Ethernet tab* and select the NET station.

Repeat these steps for every device that you want to configure.

## 10.1 Setting up a NET

### NET - a group

A NET is a group that is made up of up to eight stations and that uses a special protocol for the device series in order to allow for communications via Ethernet connections.

The name "NET" refers to Ethernet/UDP-based communications between the easyE4 devices. It was designed specifically with the needs of straightforward transfers between easyE4 devices in mind. Within a NET, every device has read access to the NET operands of any other device in its group. In addition to this, data can be transferred both cyclically and acyclically.

Please note that nodes from different groups cannot communicate directly with each other.

### Between groups

If, however, you want devices to be able to communicate across groups, you will need to use a coordinator device that controls the corresponding communication with Modbus TCP.

A total of ten NET groups (groups 1 through 10) can be run on a single Ethernet network at one time.

| Ethernet network | | | | | | |
|---|---|---|---|---|---|---|
| ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ |
| Module | Module | Module | Stations… | Module | Module | Module |
| NET-ID 1 | NET-ID… | NET-ID 8 | NET-ID… | NET-ID 1 | NET-ID… | NET-ID 8 |
| NET group 1 | | | NET group… | NET group 10 | | |

NET uses UDP protocols that send unconfirmed broadcast frames; accordingly, the devices in the NET group must be located on the same subnet. Connections through a router are not allowed, as broadcast frames will normally be unable to pass through a router.

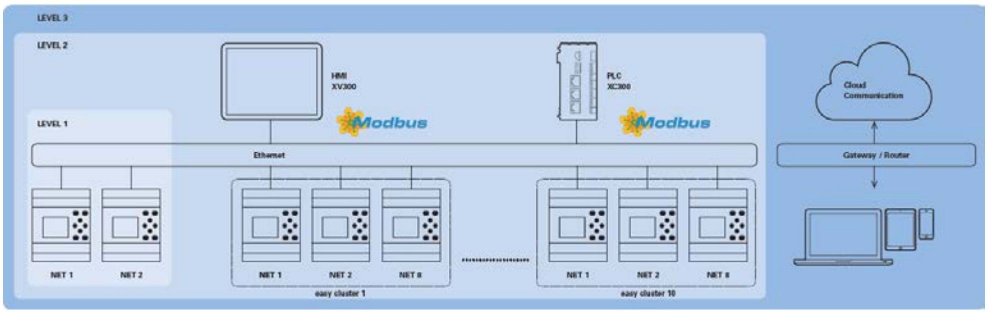



Figure 274: NET diagram

All easyE4 base devices feature an Ethernet port that can be simultaneously used for all communications, e.g., web server, Modbus TCP, e-mail, and programming the easyE4.

# 10. Connection to other devices
## 10.1 Setting up a NET

In order to be able to run a NET group, there must be an Ethernet connection between the devices or device and PC.

### 10.1.1 Access on the NET

There can be a max. of 8 easyE4 devices in a single NET group.

Access is based on various NET operands and function blocks.

1. Network markers (N, NB, NW, ND) (cyclical access)
   Every single device in the group is allowed read access to the network markers of the other devices in the group. In addition, each device has write and read access to its own network markers. This makes it possible for each device to provide up to 512 bits of data to the other stations in the group.
2. RN and SN bit markers (cyclical access)
   The RN and SN operands make it possible to directly access the state of the operands of another device on the NET. These operands are used to send and receive Boolean values. Each station in the group has 32 RN (Receive NET) and 32 SN (Sent NET) bit markers available.
3. Transmitting double words with function blocks (acyclical access)
   Each easyE4 device in the group has 32 PUT (PT) manufacturer function blocks and 32 GET (GT) function blocks available for sending and receiving analog values in an event-driven manner.
4. NET synchronization
   The device blocks in the NET group can be synchronized – please refer to
   → Section "NET-GROUP", page 591

#### Compatibility with easyNET

The easyNET devices in the easy800 series use their own custom CAN-specific transmission. Accordingly, devices from the easy800 and easyE4 series cannot be physically connected to each other.

Existing .e60 programs can be migrated to .e70 programs for the easyE4 series. When you do this, the easy800, devices that are used with the Remote I/O operating mode will be converted to local expansions.

### 10.1.2 Communication via NET

A NET group can be made up of up to eight easyE4 base devices.
Within the group, the easyE4 base devices can communicate with each other.

If, however, you want devices to be able to communicate across groups, you will need to use a coordinator device that controls the corresponding communication with Modbus TCP.

A total of ten NET groups (groups 1 through 10) can be run on a single Ethernet network at one time. This is the equivalent of a maximum of 80 easyE4 base devices that can communicate with each other.

The following list shows the operands that can be used within a group by every device:

- (n = NET-ID 1 .. 8)
- n SN 01 - 32 [Bit]
- n RN 01 - 32 [Bit]
- PT 01 - 32 (PUT) [double word]
- GT 01 – 32 (GET) double word]
- n N 01 - 512 [Bit]
- n NB 01 - 64 [Byte]
- n NW 01 - 32 [Byte]
- n ND 01 - 16 [double word]
- Synchronize clock (settings)

**Examples**

Station 1 sending a bit to station 2

| NET-ID1 | | NET-ID 2 |
|---|---|---|
| 2 SN 15 | → | 1 RN 015 |

Station 3 sending a double word to station 8 via PT16

| NET-ID1 | | NET-ID 2 |
|---|---|---|
| pt16 | → | GT 01<br>Parameter<br>NET-ID 1<br>PT 16 |

Station 4 sending a network marker [bit and word] to all stations.

| NET-ID4 | | NET-ID 2 | NET-ID 5 | NET-ID 7 |
|---|---|---|---|---|
| N 125 | → | 4 N 125 | 4 N 125 | 4 N 125 |
| NW30 | → | 4 NW 30 | 4 NW 30 | 4 NW 30 |

This basic principle applies to all network markers in all data formats.

Network markers overlap in the various data formats:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| N1-8 | N9-16 | N17-24 | N25-32 | N33-40 | N41-48 | N49-56 | N57-64 |
| NB1 | NB2 | NB3 | NB4 | NB5 | NB6 | NB7 | NB8 |
| NW1 | | NW2 | | NW3 | | NW4 | |
| ND1 | | | | ND2 | | | |
| | | | | | | | |
| N65-72 | N73-80 | N81-88 | N89-96 | N97-104 | N105-112 | N113-120 | N121-128 |
| NB9 | NB10 | NB11 | NB12 | NB13 | NB14 | NB15 | NB16 |
| NW5 | | NW6 | | NW7 | | NW8 | |
| ND3 | | | | ND4 | | | |

etc.

#### NET station heartbeat

In order to make it possible for all NET stations within a group to be able to know whether NET stations important to them are still communicating, each station cyclically sends a heartbeat every second (1 s).
If a heartbeat is not received, the corresponding error bit ID01 – 08 will be set to "1" until a heartbeat is detected.

### 10.1.3 NET settings

Prerequisites

The Ethernet settings must have already been configured.
In offline mode, setting up the configuration in easySoft 7 under the Ethernet tab is enough, → Section "Establishing an Ethernet connection and transferring a program", page 100

A NET ID needs to be assigned to every easyE4 base device and to every station added to the project as an other NET station.

*Project view*

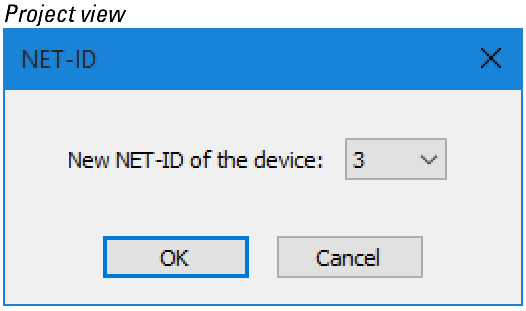



Figure 275: NET-ID dialog box used to assign a NET ID when adding a new base device

→ After you add a new station to the project, you will need to download all easyE4 programs for the NET group again.

**Loading programs onto multiple NET stations**

To conveniently load programs for multiple stations on the NET with a single operation, follow the steps below:

Prerequisites

- All the devices must be in the group physically.
- Every single device must have an assigned NET ID.

▶ If the project that is open features multiple NET stations, establish online communications with the NET-ID1 NET station.

▶ Make sure that <Device NT1> is selected instead of the usual <local> in the Communication view/Connection section/Device drop-down menu.

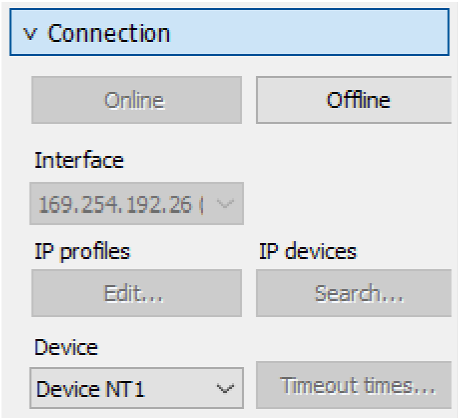
▶ Click on the PC -> Device button.

The Selection of NET stations dialog box will appear.

▶ Enable all the NET stations for which you want the new program to be loaded.

▶ Confirm your selection with OK.

The programs for all the selected NET stations will be loaded onto the devices.




*Project view*

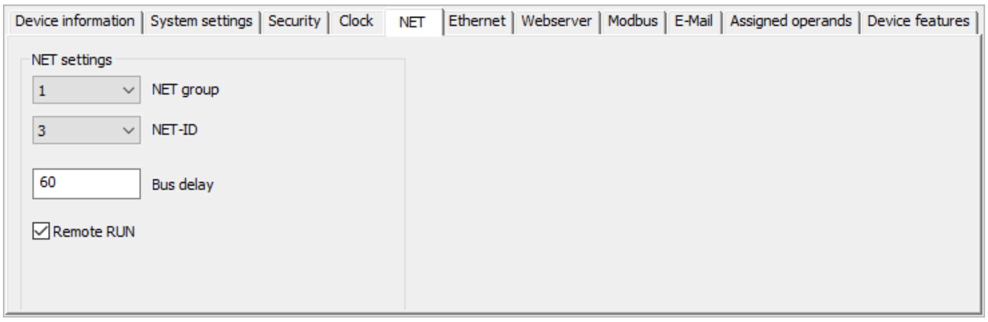



Figure 276: NET tab for the selected base device in the NET group

### NET-GROUP

Used to select the group for the selected base device.

| | |
|---|---|
| 0 | Base device running in standalone mode with the relevant I/O expansions (if any),<br>no NET group |
| 1-10 | Possible NET-GROUP |

### NET-ID

Used to assign a group device number from the NET GROUP to the selected base device.

| | |
|---|---|
| 0 | Base device running in standalone mode with the relevant I/O expansions (if any) |
| 1-8 | Available device IDs in the NET-GROUP |

### Remote RUN

If this field is enabled, the NET stations of a group with NET-IDs 02 through 08 will take their current RUN or STOP operating mode from the NET station with NET-ID 1.

### Bus delay

The bus delay is used to define the time after which a station on the NET will send its data to other stations.

This bus delay needs to be adjusted as appropriate for the number of stations and the values being transmitted. Please note than an excessively short bus delay will result in data collisions.

The permissible value range for the bus delay is 10 ms to 255 ms.

Cyclical data will be sent every 10 ms or when there is a data change, but not before the bus delay has elapsed. Using the default value of 60 ms will normally be sufficient to prevent transmission overloads.

You can use the following formula:

- Case A: When using PUT/GET and network markers:
  Bus delay in ms = (number of NET stations - 1) * 4 * 2 + 6
- Case B: When using network markers exclusively:
  Bus delay in ms = (number of NET stations - 1) * 2 * 2 + 6

The following table can be used as a convenient guide for configuring the setting:

| Number of modules: | Delay with PUT/GET in ms | Delay without PUT/GET in ms |
|---|---|---|
| 2 | 14 | 10 |
| 3 | 22 | 14 |
| 4 | 30 | 18 |
| 5 | 38 | 22 |
| 6 | 46 | 26 |
| 7 | 54 | 30 |
| 8 | 62 | 34 |

→ If you are no longer able to connect to the NET stations via Ethernet with easySoft 7, set the bus delay as high as possible for your application.
To do this, you will need to disconnect each device from the Ethernet and use easySoft 7 to change the bus delay point by point.

**See also**

→ Section "GT - Get values from NET", page 428
→ Section "PT - Put values to NET", page 432
→ Section "SC - Synchronizing clock via NET", page 436
→ Section "Establishing an Ethernet connection and transferring a program", page 100

# 10.2 Modbus TCP

Only possible with easySoft 7.

## 10.2.1 General

Modbus TCP is a simple communication protocol that uses a client-server architecture in order to make it possible for measuring and control systems (server) to communicate with higher-level control systems (client) and vice versa. Since the protocol is based on TCP/IP and Ethernet, any device that supports the Internet protocol suite and features an Ethernet port can implement it.

During communications, data is written as a payload in TCP/IP packets and transmitted this way.

Please note that Modbus TCP only assumes the functionality of a lower-level server, and not the higher-level client functionality.

Modbus TCP ensures communications with devices:

- That do not necessarily have to be part of the easyE4 family of products
- That are not found in a NET group
  or
- That do not implement NET.

The most important functions include, but are not limited to:

- Communications at the control level
- Transmitting analog and digital values to higher-level control systems
- Platform-independent communication
- Communication with devices that are not part of the easyE4 series

With firmware version 1.12 and higher, easyE4 can run two Modbus TCP clients. This makes it possible, for instance, to implement communication with a touch display and a gateway at the same time.

If you are using more than one Modbus TCP client, make sure to set a response time that is as short as possible. Change it to 60 ms!

### 10.2.2 Programming communication with Modbus TCP

In order to program these communications, you will need at least one system that fulfills the functionality of a Modbus TCP client and that is able to send commands to the lower-level server.

Since the easyE4 control relays can work with various Modbus TCP clients available on the market, only standard-compliant Modbus TCP functions are supported.
In other words, functions that are defined in a standardized manner in the Modbus standard and are accordingly implemented at the protocol level in a standardized manner by all Modbus TCP stations. For more information, please refer to the MODBUS MESSAGING ON TCP/IP IMPLEMENTATION GUIDE V1.0b published by the Modbus Organization.

**Connection:**

In order for the Modbus TCP server to work, the following ports must be opened:

- Modbus TCP: Service—Port 502

Port 502 is normally set by default. If it is not, make sure to set it when establishing a connection.

Ports that may need to be opened depending on the functionality being used:

- DNS: UDP/TCP port 53 (only if using DNS)
- DHCP: UDP port 67 for server / UDP Port 68 for clients (only if using DHCP)

The Modbus TCP server implementation on the easyE4 will provide the following standard functions:

| Function description | | Function Code |
|---|---|---|
| Read Coils | Used to read outputs | 0x01 |
| Read Discrete Inputs | Used to read inputs | 0x02 |
| Read Holding Registers | Used to read output registers | 0x03 |
| Read Input Registers | Used to read input registers | 0x04 |
| Write Single Registers | Used to write to a register | 0x06 |
| Write Multiple Registers | Used to write to multiple registers | 0x10 |

There are two basic protocol data units (PDUs) for each of the function descriptions above:

1. Request PDU (the Modbus TCP server must receive this PDU)
   a. Byte 0 contains the function code, which is used to identify the function that is desired
   b. The remaining bytes are function-specific
2. Response PDU (the Modbus TCP server must send this PDU)

a. Byte 0 contains the request's function code
b. The remaining bytes are function-specific

If an error occurs, the Modbus TCP server will send an error message

- Error-Frame
  a. Byte 0 always contains the request's error code (0x80 + function code)
  b. Byte 1 contains the exception code (error-specific)

Following is a description of the request & response function codes for each of the function descriptions in the table above:

#### 10.2.2.1 Read Coils 0x01:

This function reads the entered number of bit outputs starting from a specified starting address and then returns the result in bytes (8 outputs per byte)

Table 111: Request-PDU

| | | |
|---|---|---|
| Function Code | 1 byte | 0x01 ;Read Coils |
| Start address | 2 byte | Must always be 1 less than the starting output you want (zero-based) |
| Number of outputs | 2 byte | 1 to 2000 (0x7D0) |

Response to the request being received

1. The starting address is analyzed (distributed among bytes 1 and 2)
   a. Byte 1 = Hi; Byte 2 = Lo
2. The number of outputs is analyzed (distributed among bytes 3 and 4)
   a. Byte 3 = Hi; Byte 4 = Lo
3. The output states are read
   a. From the start of the (starting address) to (starting address + number of outputs)

Table 112: Response-PDU

| | | |
|---|---|---|
| Function Code | 1 byte | 0x01 ;Read Coils |
| Bye Count | 1 byte | N |
| Output values | n * 1 byte | Value |

n= Number of outputs read / 8

Preparation for sending the response

1. The read bits are encoded in bytes

(1 but per output state; 1=ON, 0=OFF)

2. The LSB of the first byte, i.e., bit 0, contains the state of the output that is addressed first in the request. The other outputs follow in ascending order.

3. If a byte is not used fully, the unused bits will be padded with 0's.

Once the response is encoded, it is sent.

#### 10.2.2.2 Read Discrete Inputs 0x02:

This function reads a specified number of bit inputs starting from a specified starting address and then returns the result in bytes (8 inputs per byte)

Table 113: Request-PDU

| | | |
|---|---|---|
| Function Code | 1 byte | 0x02 ;Read Discrete Inputs |
| Start address | 2 byte | Must always be 1 less than the starting input you want (zero-based) |
| Number of outputs | 2 byte | 1 to 2000 (0x7D0) |

Response to the request being received

1. The starting address is analyzed (distributed among bytes 1 and 2)
   a. Byte 1 = Hi; Byte 2 = Lo
2. The number of inputs is analyzed (distributed among bytes 3 and 4)
   a. Byte 3 = Hi; Byte 4 = Lo
3. The bit input states are read
   a. From the start of the (starting address) to (starting address + number of bit inputs)

Table 114: Response-PDU

| | | |
|---|---|---|
| Function Code | 1 byte | 0x02 ;Read Discrete Inputs |
| Bye Count | 1 byte | N |
| Output values | n* 1 Byte | Value |

n= Number of inputs read / 8

Preparation for sending the response

1. The read bits are encoded in bytes
   Bit per input state; 1=ON, 0=OFF)
2. The LSB of the first byte, i.e., bit 0, contains the state of the input that is addressed first in the request. The other inputs follow in ascending order.
3. If a byte is not used fully, the unused bits will be padded with 0's.

Once the response is encoded, it is sent.

#### 10.2.2.3 Read Holding Registers 0x03:

Function 0x03 reads internal registers (e.g., marker words in the easyE4) word by word.

Table 115: Request-PDU

| | | |
|---|---|---|
| Function Code | 1 byte | 0x03 ;Read Holding Registers |
| Start address | 2 byte | Must always be 1 less than the starting input you want (zero-based) |
| Number of registers | 2 byte | 1 to 125 (0x7D) |

Response to the request being received

1. The starting address is analyzed (distributed among bytes 1 and 2)
   a. Byte 1 = Hi; Byte 2 = Lo
2. The number of registers is analyzed (distributed among bytes 3 and 4)
   a. Byte 3 = Hi; Byte 4 = Lo
3. The data words are read from the start of the (starting address) to (starting address + number of registers)
   A register corresponds, e.g., to one marker word

Table 116: Response-PDU

| | | |
|---|---|---|
| Function Code | 1 byte | 0x03 ;Read Holding Registers |
| Bye Count | 1 byte | A value of<br>= 2 * n must always be entered here |
| Register values | n* 2 byte | Value |

n= Number of registers read

Preparation for sending the response

1. The registers read (marker words) are mapped to two bytes per register
2. There are a high byte and a low byte for each register (marker word)

**Example**

- Register word Hi0x02
- Register word Lo0x2B
- Content of marker word 0x022B

3. The LSB within the byte is bit 0

Once the response is encoded, it is sent.

#### 10.2.2.4 Read Input Registers 0x04:

Function 0x04 reads the analog inputs' registers word by word.

The Modbus client treats two bytes as one input register.

This means that in order to read an analog input with 32 bits, two consecutive input registers must be read.

Table 117: Request-PDU

| | | |
|---|---|---|
| Function Code | 1 byte | 0x04 ;Read Input Registers |
| Start address | 2 byte | Must always be 1 less than the starting input you want (zero-based) |
| Number of input registers | 2 byte | 1 to 125 (0x7D) |

Response to the request being received

1. The starting address is analyzed (distributed among bytes 1 and 2)
   b. Byte 1 = Hi; Byte 2 = Lo
2. The number of registers is analyzed (distributed among bytes 3 and 4)
   b. Byte 3 = Hi; Byte 4 = Lo
3. The input registers are read from the start of the (starting address) to (starting address + number of input registers)
   (An input register corresponds to two bytes)

Table 118: Response-PDU

| | | |
|---|---|---|
| Function Code | 1 byte | 0x04 ;Read Input Registers |
| Bye Count | 1 byte | A value of<br>= 2 * N must always be entered here |
| Register values | n* 2 byte | Value |
| n= Number of input registers read | | |

Preparation for sending the response

1. The input registers read are mapped to two bytes per input register
2. There are a high byte and a low byte for each input register
   a. The first byte = Hi; the second byte = Lo
   b. Example:
      - Register word Hi0x00
      - Register word Lo0x0A
      - Content of marker word 0x000A
3. The LSB within the byte is bit 0

Once the response is encoded, it is sent.

#### 10.2.2.5 Write Single Registers 0x06:

This function writes 16 bites to a register ((NET) marker word in the easy)

Table 119: Request-PDU

| | | |
|---|---|---|
| Function Code | 1 byte | 0x06 ;Write single Registers |
| Target address | 2 byte | Must always be 1 less than the MW being written to (if you want MW1 to be written to, there must be a 0 here) |
| Register value | 2 byte | Value to be written |

Response to the request being received

1. The target address is analyzed (distributed among bytes 1 and 2)
   a. Byte 1 = Hi; Byte 2 = Lo
2. The value being written is analyzed (distributed among bytes 3 and 4)
   a. Byte 3 = Hi; Byte 4 = Lo
3. The value is written to the target register ((NET) marker word)

Response-PDU

If the value is written successfully, the request will be echoed once as a response (→ Section "Write Single Registers 0x06:", page 638 Request PDU)

In other words, the response is identical to the corresponding request and is used for confirmation purposes only.

#### 10.2.2.6 Write Multiple Registers 0x10:

This function writes n * 16 bits to N registers ((NET) marker words in the easyE4)

Table 120: Request-PDU

| | | |
|---|---|---|
| Function Code | 1 byte | 0x10 ;Write Multiple Registers |
| Start address | 2 byte | Must always be 1 less than the starting marker word (if you want MW1 to be written to, there must be a 0 here) |
| Number of registers | 2 byte | 1-123 (0x0001 to 0x007B) |
| Bye Count | 1 byte | 2 * N |
| Register values being written (marker words) | n * 2 byte | Values being written |

n = Number of registers being written to

Response to the request being received

1. The starting address is analyzed (distributed among bytes 1 and 2)
   a. Byte 1 = Hi; Byte 2 = Lo
2. The number of registers is analyzed (distributed among bytes 3 and 4)
   a. Byte 3 = Hi; Byte 4 = Lo

3. The number of bytes is analyzed
4. The marker words are written to the target registers

If the values are written successfully, a response is sent.

This response will contain the function code, the starting address, and the number of registers from the request
(→ Section "Write Multiple Registers 0x10:", page 638 Request PDU)

Table 121: Response-PDU

| | | |
|---|---|---|
| Function Code | 1 byte | 0x10 ;Write Multiple Registers |
| Start address | 2 byte | Same value as in request |
| Number of registers | 2 byte | Number of registers written to (the value should match the request) |

### 10.2.3 Modbus TCP error handling

**Read Coils 0x01:**

In the event of an error, Modbus TCP will send an error frame.

| | | |
|---|---|---|
| Fault Code | 1 byte | 0x81 ; Read Coils |
| Exception Code | 1 byte | 02 or 03 or 04 |

Exception Code 02 = Invalid address, i.e.:

- 0 (the user keeps specifying addresses as one-based addresses)
- Undefined* (please refer to the "Modbus map" table) or
- Un-enabled*

Exception Code 03 = The number of outputs is not >= 0x0001 and <= 0x07D0

Exception Code 04 = (Error in server) n.a.**

*For an error message, it is sufficient if only one of the requested addresses is not unlocked or is invalid.

**Data in the image table is protected from other modules by semaphores; as of this writing, no known criterion for a "read coil" error in the server.

**Read Discrete Inputs 0x02:**

In the event of an error, Modbus TCP will send an error frame.

| | | |
|---|---|---|
| Fault Code | 1 byte | 0x82 ; Read Discrete Input |
| Exception Code | 1 byte | 02 or 03 or 04 |

Exception Code 02 = Starting address is invalid, i.e.:

- 0 (the user keeps specifying addresses as one-based addresses)
- Undefined* (please refer to the "Modbus map" table) or

- Un-enabled*

Exception Code 03 = The number of inputs is not >= 0x0001 and <= 0x07D0

Exception Code 04 = (Error in server) n.a.**

*For an error message, it is sufficient if only one of the requested addresses is not unlocked or is invalid.

**A read operation always returns consistent data from the image table, since this data is protected from other modules by semaphores. In other words, as of this writing, there is no known criterion for a "read discrete inputs" error in the server.

#### Read Holding Registers 0x03:

In the event of an error, Modbus TCP will send an error frame.

| | | |
|---|---|---|
| Fault Code | 1 byte | 0x83 ; Read Holding Registers |
| Exception Code | 1 byte | 02 or 03 or 04 |

Exception Code 02 = Starting address is invalid, i.e.:

- 0 (the user keeps specifying addresses as one-based addresses)
- Undefined* (please refer to the "Modbus map" table) or
- Un-enabled*

Exception Code 03 = The number of inputs is not >= 0x0001 and <= 0x07D0

Exception Code 04 = (Error in server) n.a.**

If no analog I/O is physically present (e.g., analog I/O smart modules missing or faulty), the image table (values = 0) will still be delivered to the client. No check, no error message.

*For an error message, it is sufficient if only one of the requested addresses is not unlocked or is invalid.

**A read operation always returns consistent data from the image table, since this data is protected from other modules by semaphores. In other words, as of this writing, there is no known criterion for a "read holding registers" error in the server.

#### Read Input Registers 0x04:

In the event of an error, Modbus TCP will send an error frame.

| | | |
|---|---|---|
| Fault Code | 1 byte | 0x84 ; Read Input Registers |
| Exception Code | 1 byte | 02 or 03 or 04 |

Exception Code 02 = Starting address is invalid, i.e.:

- 0 (the user keeps specifying addresses as one-based addresses)
- Undefined* (please refer to the "Modbus map" table) or
- Un-enabled*

Exception Code 03 = The number of inputs is not >= 0x0001 and <= 0x07D0

Exception Code 04 = (Error in server) n.a.**

If no analog I/O is physically present (e.g., analog I/O smart modules missing or faulty), the image table (values = 0) will still be delivered to the client. No check, no error message.

*For an error message, it is sufficient if only one of the requested addresses is not unlocked or is invalid.

**A read operation always returns consistent data from the image table, since this data is protected from other modules by semaphores. In other words, as of this writing, there is no known criterion for a "read input registers" error in the server.

**Write Single Register 0x06:**

In the event of an error, Modbus TCP will send an error frame.

| | | |
|---|---|---|
| Fault Code | 1 byte | 0x90 ;Write Single Register |
| Exception Code | 1 byte | 02 or 03 or 04 |

Exception Code 02 = Target address is invalid, i.e.,

- 0 (the user keeps specifying addresses as one-based addresses)
- Undefined* (please refer to table → Section "Modbus map", page 643) or
- Un-enabled*

*For an error message, it is sufficient if only one of the requested addresses is not unlocked or is invalid.

Exception Code 04 = Error when attempting to write to the register (marker word)**

**A write operation can always write consistent data to the image table, since this data is protected from other modules by semaphores. In other words, as of this writing, there is no known criterion for a "Write Single Register" error in the server.

Values are only allowed to be written if all required addresses are valid and unlocked.

**Write Multiple Registers 0x10:**

In the event of an error, Modbus TCP will send an error frame.

| | | |
|---|---|---|
| Fault Code | 1 byte | 0x86 ;Write Multiple Registers |
| Exception Code | 1 byte | 02 or 03 or 04 |

Exception Code 02 = Target address is invalid, i.e.,

- 0 (the user keeps specifying addresses as one-based addresses)
- Undefined*(please refer to table → Section "Modbus map", page 643) or
- Un-enabled*

Exception Code 03 = The number of registers is not >= 0x0001 and <= 0x007B

OR

Number of bytes != Number of registers x 2

Exception Code 04 = Error when attempting to write to the registers**

Values are only allowed to be written if all required addresses are valid and unlocked.

*For an error message, it is sufficient if only one of the requested addresses is not unlocked or is invalid.

**A write operation can always write consistent data to the image table, since this data is protected from other modules by semaphores. In other words, as of this writing, there is no known criterion for a "write multiple registers" error in the server.

**Unknown function:**

If the client requests an unsupported function, the Modbus TCP server must return the following error frame after receiving the request:

| | | |
|---|---|---|
| Fault Code | 1 byte | 0x80 + Function code |
| Exception Code | 1 byte | 01 |

This will indicate to the client that the desired function is not supported by the server.

### 10.2.4 Modbus map

Table 122: How the Modbus registers and read data for the Modbus server are mapped easyE4 control relays

| Function code Modbus | Modbus reg. # | Operand | Description | Remark |
|---|---|---|---|---|
| 0x01 (Read Coil) Max. 512 coils at once, 8 coils are grouped into a byte | 1 | Q1 | Bit output 1 | Local outputs base device |
| | … | … | … | |
| | 4 | Q4 | Bit output 4 | |
| | | | | |
| | 17 | Q17 | Expansion Bit output | Local outputs expansion |
| | … | … | … | |
| | 128 | Q128 | Expansion Bit output | |
| | | | | |
| | 1001 | M1 | Marker bit 1 | |
| | ... | … | … | |
| | 1512 | M512 | Marker bit 512 | |
| | | | | |
| | 2001 | N1 | NET marker bit 1 | Only the local NET marker bits are returned |
| | … | … | … | |
| | 2512 | N512 | loc. NET marker bit 512 | |
| 0x03 (Read Holding Register) Max. 125 registers at once, 1 register = 2 byte/1word | 1 | QA1 | 16-bit analog output 1 | Local analog outputs base device |
| | … | … | … | |
| | 4 | QA4 | 16-bit analog output 4 | |
| | | | | |
| | 5 | QA5 | Expansion 16-bit analog output | Local analog outputs expansion |
| | … | … | … | |
| | 48 | QA48 | Expansion 16-bit analog output | |
| | | | | |
| | 1001 | MW1 | Marker word 1 | |
| | … | … | … | |
| | 1512 | MW512 | Marker word 512 | |
| | | | | |
| | 2001 | NW1 | loc. Net marker word 1 | Only local NET marker words will be returned, i.e., no double words, no bytes, and no markers from the other stations will be returned |
| | … | … | … | |
| | 2032 | NW32 | NET marker word 32 | |
| | | | | |
| | 5000 | | RTC (second) | 5000: Seconds; 5001: Minutes; 5002: Hours; 5003: Day; 5004: Month; 5005: Year |
| | …. | … | … | |
| | 5005 | | RTC (year) | |

| Function code Modbus | Modbus reg. # | Operand | Description | Remark |
|---|---|---|---|---|
| 0x02 (Read discrete input) Max. 512 inputs at once, 8 inputs are grouped into a byte | 1 | I1 | Bit input 1 | Local inputs base device |
| | … | … | … | |
| | 8 | I8 | Bit input 8 | |
| | | | | |
| | 17 | I17 | Expansion bit input | local inputs expansion |
| | … | … | … | |
| | 128 | I128 | Expansion bit input | |
| | | | | |
| | 1001 | M1 | Marker bit 1 | |
| | … | … | … | |
| | 1512 | M512 | Marker bit 512 | |
| | | | | |
| | 2001 | N1 | NET marker bit 1 | Only the local NET marker bits and no marker bits of the other stations are returned |
| | … | … | … | |
| | 2512 | N512 | NET marker bit 512 | |
| | | | | |
| | 3001 | ID1 | Diagnostics bit 1 | Diagnostics for base device |
| | … | … | … | |
| | 3024 | ID24 | Diagnostics bit 24 | |
| | | | | |
| | 3125 | ID25 | Diagnostics bit 25 | Diagnostics extension |
| | … | … | … | |
| | 3196 | ID96 | Diagnostics bit 96 | |
| 0x04 (Read Input Register) Max. 125 registers at once, 1 register = 2 bytes / 1 word | 1 | IA1 | 16-Bit analog input 1 | Local analog inputs base device |
| | … | … | … | |
| | 4 | IA4 | 16-Bit analog input 8 | |
| | | | | |
| | 5 | IA5 | 16-Bit analog input | Local analog inputs expansion |
| | … | … | … | |
| | 48 | IA48 | 16-Bit analog input | |
| | | | | |
| | 1001 | MW1 | Marker word 1 | |
| | … | … | … | |
| | 1512 | MW512 | Marker word 512 | |
| | | | | |
| | 2001 | NW1 | NET marker word 1 | Only the local NET marker words will be returned, i.e., accessing NET markers from other nodes is not possible. NET marker bytes and NET marker double words can be calculated based on the NET marker words. |
| | … | … | … | |
| | 2032 | NW32 | NET marker word 32 | |

If a function code is used on Modbus registers that are not listed (gray), a value of 0 or an exception code will be returned.

Table 123: How the Modbus registers and write data for the easyE4 Modbus server are mapped

| Function codeModbus | Modbus reg.# | Operand | Description | Remark |
| --- | --- | --- | --- | --- |
| 0x06 (Write Single Register) and 0x10 (Write Multiple Register) | | | | |
| | 1001 | MW1 | Marker word 1 | |
| | … | … | … | |
| | 1512 | MW512 | Marker word 512 | |
| | | | | |
| | 2001 | NW1 | NET marker word 1 | |
| | … | … | … | |
| | 2032 | NW32 | NET marker word 32 | |

→ Please note that little-endian is used when converting bytes to words in an easyE4. If you want to implement Modbus communications with big-endian, you will need to make the necessary adjustments.

**See also**

## 10.3 Setting up a web server

Only possible with easySoft 7.

The web server is intended to make it even more convenient for users to use an easyE4 control relay. This web server makes it possible to use a web client (i.e., a web browser) to access the device as though it were being accessed directly on the easyE4 base device. In other words, this means that the web service offers an additional interface for communication that is much like an additional HMI for the easyE4 device. In addition to this, the web client is also perfectly compatible with mobile devices.

The device status can be ready directly on the display on EASY-E4-...-12...C1(P) devices – please refer to → Section "Status display on easyE4 control relays with display and keypad", page 94.
EASY-E4-...-12...CX1(P) devices without a display can also be read using the web server function.

The web server only has a limited amount of computing time available. This ensures that easyE4 will not be negatively affected when it comes to running programs.

The web server needs to be configured in easySoft 7 by opening the Project view and then going to the Webserver tab.

### Webserver tab

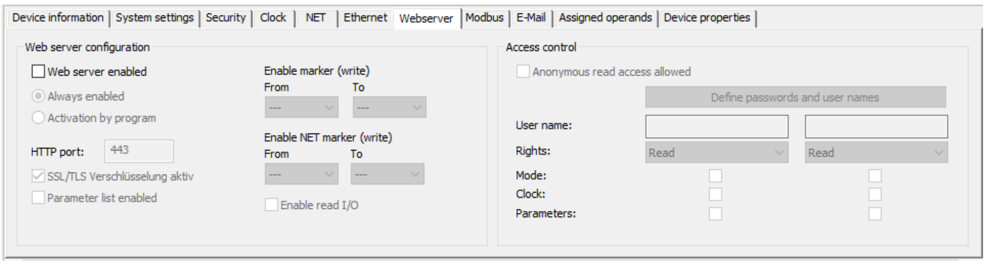



Figure 277: Project view Webserver tab

### Web server configuration

| | |
|---|---|
| Web server enabled | When this option is enabled, the **Web server passwords and user names** dialog box will appear so that you can set up the corresponding users – please refer to → Section "Setting up users", page 648<br>If the option is disabled, all settings, passwords, and usernames will be reset. |
| Always enabled | As soon as the project is loaded onto the easyE4 base device, the web server will be active every time the device is turned on. |
| Activation by program | If the web server is supposed to be activated by a specific program, all AL alarm function blocks will be read before the web server is started. The parameter set for these alarm function blocks includes the **Web server act-** |

| | |
|---|---|
| | ive as long as there is a state of 1 at input EN option, and the web server will only be started if this option is enabled for at least one alarm function block and the condition is met (i.e., function block input EN = 1). This means that if this option is not enabled for any of the alarm function blocks, the web server will never be started. |
| HTTP : Port | HTTPS default port 443 will be set for encrypted connections, but you can use any port you want. Please keep the TCP/UDP ports assigned to protocols (specified by IANA) in mind. |
| ☑ SSL/TLS encryption enabled | If this option is enabled, communications between the web server and the web client will always be encrypted. |
| ☐ Parameter list enabled | If this option is enabled, the Parameter list menu option will be shown in the web client catalog. In this case, it will be able to put together a custom parameter list with operands in the web client. This will make monitoring and controlling the relevant operands significantly easier. |
| Enable marker (write)<br>copy from up to | This is where the marker range for access through the web client is enabled. The enabled range will apply to the administrator and to all defined users equally. |

## Access control

| | |
|---|---|
| ☑ Anonymous read access allowed | If this option is enabled, everyone will be granted read access to the easyE4 base device. As soon as the web client starts, the contents will be shown without requiring the user to log in. |
| Define passwords and user names | Clicking on this button will open<br>→ "Web server passwords and user names dialog box", page 649 |
| User-name | If additional users have been set up in addition to the administrator, they will be shown here. |
| Entitlements | Shows the Read or Read and write permissions for the user. |

The following options are equivalent to the settings in *Project view /Security tab/Area password input*, see Password input:

| | |
|---|---|
| ☑ Mode | If this option is enabled with a check mark, the relevant user will be able to change the RUN/STOP operating mode for the easyE4 base device using the web client's menu bar. The administrator definitely has the permission. (the administrator always has the required permissions for this). |
| ☑ Clock | If this option is enabled with a check mark, the device time of the device clock can be modified from the web client. This function may be helpful during commissioning.<br>However, if in *Project view/Clock* the option<br>Synchronize clock via SNTP or<br>Synchronize clock via wireless (DCF77) is enabled, |

| | |
|---|---|
| | the device will get its device time as a client from an SNTP server or from a wireless clock (DCF77). In the process, the time modified via the web client is overwritten again. |
| ☑ Parameters | If this option is enabled, the relevant user will be able to use the *Catalog display* menu option in the web client in order to access the PARAMETERS menu in the remote display and, once there, configure function block inputs and outputs. In addition, the user will be able to write the function block inputs and outputs that are grouped together in a custom manner in the parameter list menu option in the web client. → If this option is not being displayed, check to make sure that firmware version 1.10 or higher is selected in the *Project view/System settings tab*. |

**Default administrator permissions**

- Only the administrator can operate the remote display
- Changing the STOP/RUN operating mode
- Writing markers, provided that they have been enabled in the Web server configuration section.
- Reading diagnostics

### 10.3.1 Configuring the web server function in easySoft 7

You can configure the web server functions you want for each device in the project in easySoft 7. To configure the web server functions for a device, follow the steps below:

▶ Select the device you want from the catalog in the Project view.

▶ Click on the Webserver tab.

**Setting up users**

Under the tab, there is the Webserver configuration section that can be used to enable and configure the web server functionality, as well as an Access control section.

▶ Click on the ☑ Web server enabled checkbox to enable the option.

As soon as you enable the web server function, the Web server passwords and user names dialog box will appear. In order to be able to access the easyE4 base device through a web client later on, an administrator must be able to log in to the easyE4 base device. – this will require a password.

→ Please note the security requirements for the password:
At least eight ASCII characters with at least one uppercase character, one lowercase character, one number, and one special character.

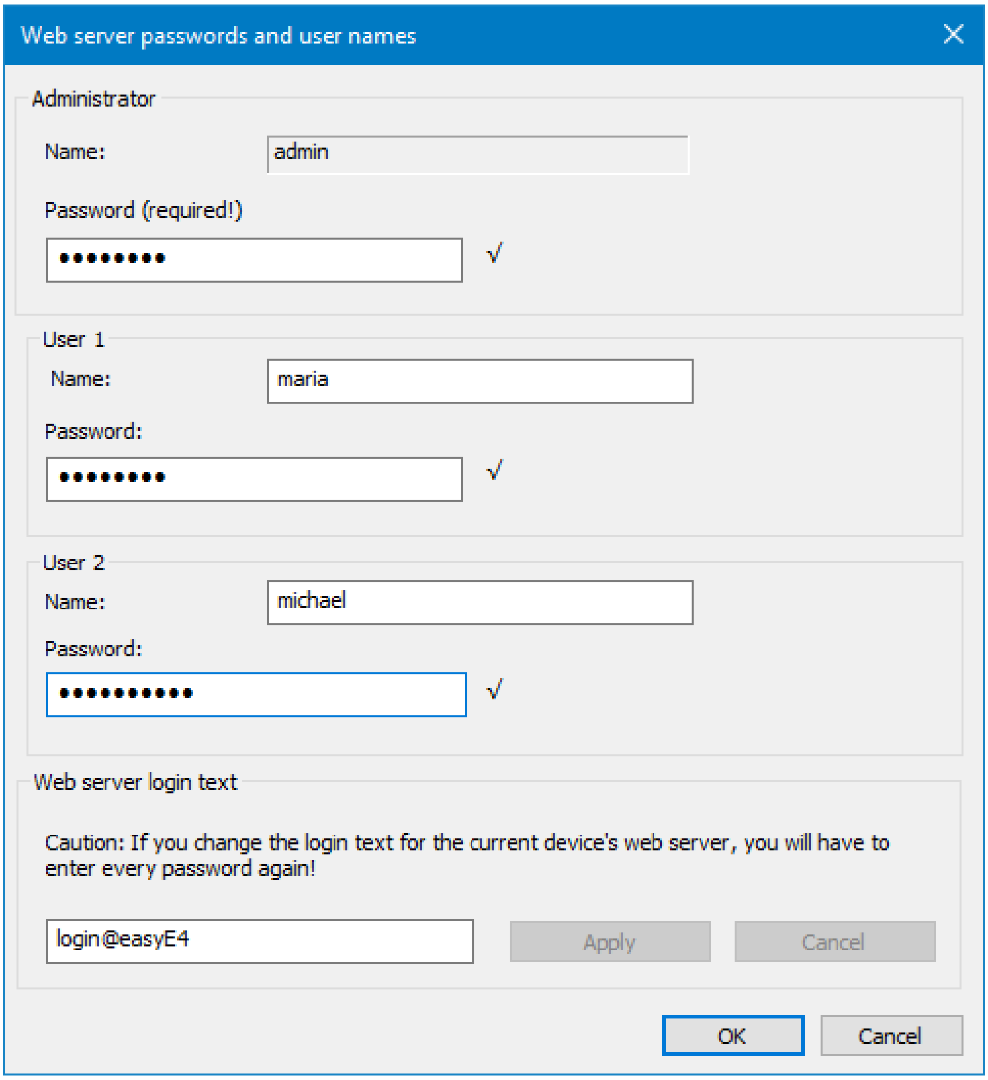



Figure 278: Web server passwords and user names dialog box

▶ Enter a password for the administrator.

You will then be able to create up to two users.

▶ Enter a username into the corresponding text field.

▶ Enter a password into the corresponding text field.

### Setting the web server login text

If there are multiple easyE4 base devices on the Ethernet network, you can assign a different web server login text to each device. This web server login text will then appear in the login dialog box for the web client, where it can be used to make sure that the device to which a connection is being established is actually the desired device.

▶ Enter a web server login text of your choice for the easyE4 base device or keep the default <login@easyE4> login text in the text field.

→ Please note that every time you change the web server login text and apply the change with the Apply button, you will need to set up all users again.

When you confirm with the OK button, the users will be created and you will be taken back to the Webserver tab.

**Configuring the settings in the Webserver tab**

**Web server configuration**

- Now select whether you want the web server to be Always enabled
  or
- if Activation by program should instead be required.
  If the web server is supposed to be activated by a specific program, all AL alarm function blocks will be read before the web server is started. The parameter set for these alarm function blocks includes the Web server active as long as there is a state of 1 at input EN option, and the web server will only be started if this option is enabled for at least one alarm function block and the condition is met (i.e., function block input EN = 1). This means that if this option is not enabled for any of the alarm function blocks, the web server will never be started.
- Set the HTTP port you want.
  80 will be set as the default "HTTP port".
  If you change this port, please make sure that the HTTP port is set to the same value both on the web browser and on the easyE4 device.

Now define the ranges that can be written to via the web server with the From and To drop-down menus. .

- Select the range for Enable marker (write) .
  The enabled marker range applies to the administrator and to all created users equally.

**Access inhibit**

- Enable this option if you want anonymous read access to be allowed.
  If this option is enabled, everyone will have read access to the easyE4 base device. As soon as the web client starts, the contents will be shown without requiring the user to log in.
- The User name field will have a maximum of two users, with these users being the ones you previously set up in the Setting up users step. You can use the drop-down menus underneath these users to configure their access permissions: Read or Read and write.
- A user will be able to change the RUN/STOP operating mode through the web client if this option is enabled for them (please note that the administrator will always have write permissions for the operating mode).

▶ If you want to change a user or their password, simply click on the button to → "Web server passwords and user names dialog box", page 1

The settings will take effect as soon as you store the project on the easyE4 base device.

**See also**

→ Section "Web client", page 652
→ Section "AL - Alarm function block", page 440

## 10.4 Web client

The web client can only be started if the web server function has already been configured and the password for the administrator or for a different user is known. The following web browsers are supported:

- Internet Explorer 11 or higher,
- Chrome,
- Safari,
- MS Edge,
- Opera,
- Brave,
- Firefox.

We recommend using Chrome, because the web client was optimized for this browser.

The web client was developed according to Responsive Design principles, allowing a good remote display for each display device such as screen, laptop, tablet and smart phone as well.

→ Please note that each access to the easyE4 base device increases the security risk from the outside.
For this reason, please note EATON's recommendations on product safety.
Only provided in English.

Product Cybersecurity, Secure Hardening Guideline MZ049001EN

*Webclient/ Start.*

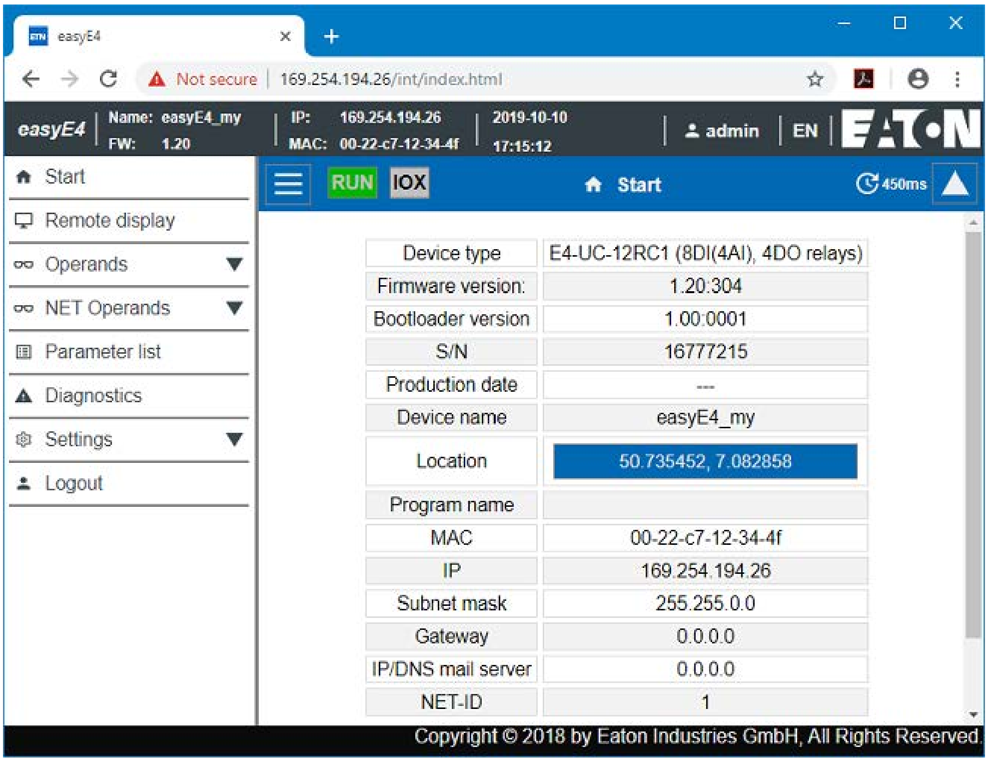



Figure 279: Web client, running

Depending on the protocol used, we recommend to only access a certain number of client programs at the same time on a easyE4 base device:

- https: 2 Client programs
- http: ≤ 4 Client programs

The term client programs refers to web client or JSON API. Otherwise, the wait time for the updated display may increase disproportionately.

**Starting the web client**

To start the web client, follow the steps below:

▶ Open your web browser.

▶ You may need to accept the IP address for easyE4 in the browser settings for the proxy server.

▶ We recommend using an encrypted IP connection with the HTTPS port. Accordingly, enter the following into the address bar:
"https://" "IP address of easyE4 base device", e.g., <https://192.168.0.2>.
If you configured an HTTPS port other than port 443 or an HTTP port other than port 80 when configuring the web server function, include the HTTPS port accordingly, e.g., <https://192.168.0.2:90>.
The following dialog appears:

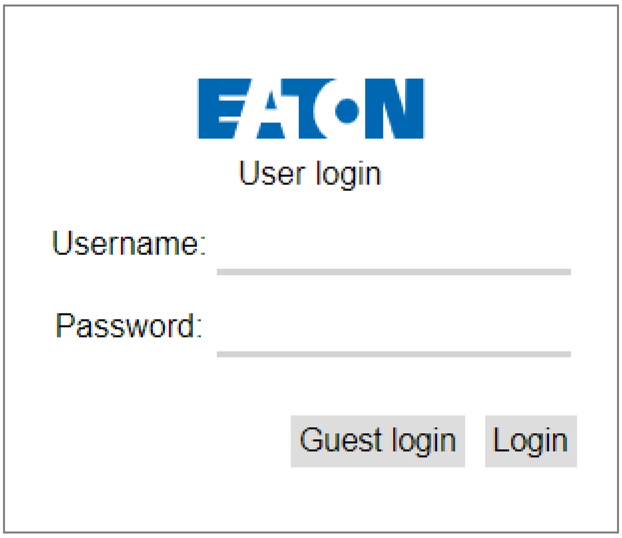



Figure 280: Web client login dialog box

- ▶ If you want to access the easyE4 base device as an administrator, enter <admin> as the username, as well as the corresponding password, into the dialog box.
- ▶ If you instead want to access the easyE4 base device as a user, enter the username and password that you set when when configuring the web server function into the dialog box.
- ▶ Confirm by clicking on the Login button.
- ▶ If you would like to log on as a guest, confirm your entry by clicking on the Guest Login button.
  This requires that the option *Anonymous read access permitted* is enabled with the check mark in the Project view/ Tab web server/ Access protection area.

The web client will start and you will be able to access the easyE4 base device. The specifics of this access will depend on the web server function configuration set up in the *Project view/Webserver/Access control* section.

**Log on as guest**

This requires that the option *Anonymous read access permitted* is enabled with the check mark in the Project view/ Tab web server/ Access protection area.

- ▶ Do not enter a user name. Instead, simply click on the Guest login button.

The web client starts and you only have read access to the easyE4base device.

**Using the web client**

The web client is subdivided into three areas: A menu bar, a left pane, and a work pane.

**How it looks in the web client**

Generally, the following colors indicate the editability of fields:

- Gray: Read access only
- Blue: Read/write access

As a rule, the digital operands will be indicated with the following colors:

- M1 : Operand =0, read access only
- M1 : Operand =0, read and write access
- M1 : Operand =1 is set, read access only
- M1 : Operand =1 is set, read and write access

Show comments – Comments that were configured in easySoft 7 can be shown or hidden in the web client.

Clicking on the input field in the workspace moves the viewing window so that the input field you clicked on is shown in the center. Please refer to → "PreventInputScroll", page 669 as well.

**Menu bar**

The menu bar contains non-editable and editable information. Editable information can also be edited in the easySoft 7 in the web client and in the device, depending on the access permission issued in the easySoft 7. The content of the menu bar is explained below with a definition of the editing options:

| Menu bar 1 | Description | easySoft 7 | Webclient | Device |
|---|---|---|---|---|
| easyE4 (NT1) | Device (station) | x | – | – |
| Name: Easy2 | Device name | x1) | x | – |
| IP: 192.168.0.2 | IP address of the device, | x1) | x | x |
| 2019-03-13 | Current device date | x2) | x | x |
| FW: 1.10 | Firmware version of the device | – | – | – |
| MAC: 00-22-c7-12-0d-31 | The device's MAC address | – | – | – |
| 15:45:09 | Current device time | x2) | x | x |
| Admin | Display of the logged on user | – | x | – |
| \| DE \| | Selecting the language for the web client, DE for example; There are thirteen languages available: DE, EN, IT, ES, PL, FR, for example | – | x | – |

– Information that cannot be edited
1) see also Project view
2) see also Clock

The language selection for the web client may be different to the language selection in the device. Since the language selection is stored exclusively in the browser, each web client can show the device content in a different language.

| Menu bar 2 | Description | easySoft 7 | Webclient | Device |
|---|---|---|---|---|
| | Show/hide catalog | – | x | – |
| RUN | Button used to select the easyE4 operating state : green RUN, red STOP | x | x | x |

| Menu bar 2 | Description | easySoft 7 | Webclient | Device |
|---|---|---|---|---|
| IOX | Displays the status of the easyConnect bus (IO eXtension)<br>IOX - highlighted in gray: No expansion devices are connected or fault at the easyConnect bus.<br>Possible causes:<br>• Configuration not OK<br>• Expansion device defective<br>• Expansion device has no supply voltage<br>• Communication error to an expansion device<br>IOX - highlighted in green: easyConnect bus in operation | – | – | – |
| Start | Show selection in the catalog | – | x | – |
| | | | | |
| | Show/hide menu bar | – | x | – |

### Catalog

| Menu bar 2 | Description |
|---|---|
| Start | Web client start menu with the most important information regarding the connected device. |
| Remote display | The remote display will be shown in the work pane (only the administrator has access to this remote display). This display is operated exactly the same way as the easyE4 base device itself. |
| Operands | Operands can be modified. The administrator always has write permissions for local operands. Users can also be assigned these permissions. However, access to the marker range through the web client must first be enabled in easySoft 7 and, if necessary, read permissions for I/O must be assigned as well – please refer to → "Enable marker (write)", page 647 as well. |
| NET operands | NET operands can be modified. The administrator always has write permissions for his own NET markers. However, access to the NET marker range through the web client must first be enabled in easySoft 7 – please refer to → "Enable marker (write)", page 647 as well.<br>Other users will be able to modify operands if write permissions have been set up for them in the access control settings – please refer to → "Access control", page 647 |
| Parameter list | The user can compose a list of operands that he would like to observe and/or edit. |
| Diagnostics | Shows the diagnostic messages that are currently present – please refer to → "Operating system diagnostic messages", page 609 as well. |
| Settings | The user can make **General settings** for the device, **Network settings**, **Email settings** and Settings for the **web client** on his own. |
| Logout | Logs off a logged on user. |

### Update operands

The web client queries all data in the easyE4 base devices cyclically at a specific interval. This interval is referred to as the "web client cycle time" and is configurable. The default value is 450 ms. The data is stored temporarily in the web client's storage area. The displayed operands in the web client are no older than one second.

→ As soon as the data is a few seconds old (), the loading circle will appear.

Depending on the protocol used, we recommend to only access a certain number of client programs at the same time on a easyE4 base device:

- https: 2 Client programs
- http: ≤ 4 Client programs

The term client programs refers to web client or JSON API. Otherwise, the wait time for the updated display may increase disproportionately.

### Update web client

The web client is an integral part of the firmware. To update a web client, the latest firmware must be installed on an SD card. The SD card must be inserted in the device. The index.html file is started as a web client.

## Remote display

The keypad can be operated in the web client display exactly in the same way as on the device itself. We recommend switching into the special menu using the key combination ALT+SHIFT instead of using the common actuation on the device with the ALT key. Alternatively, you can control the keypad with mouse clicks.

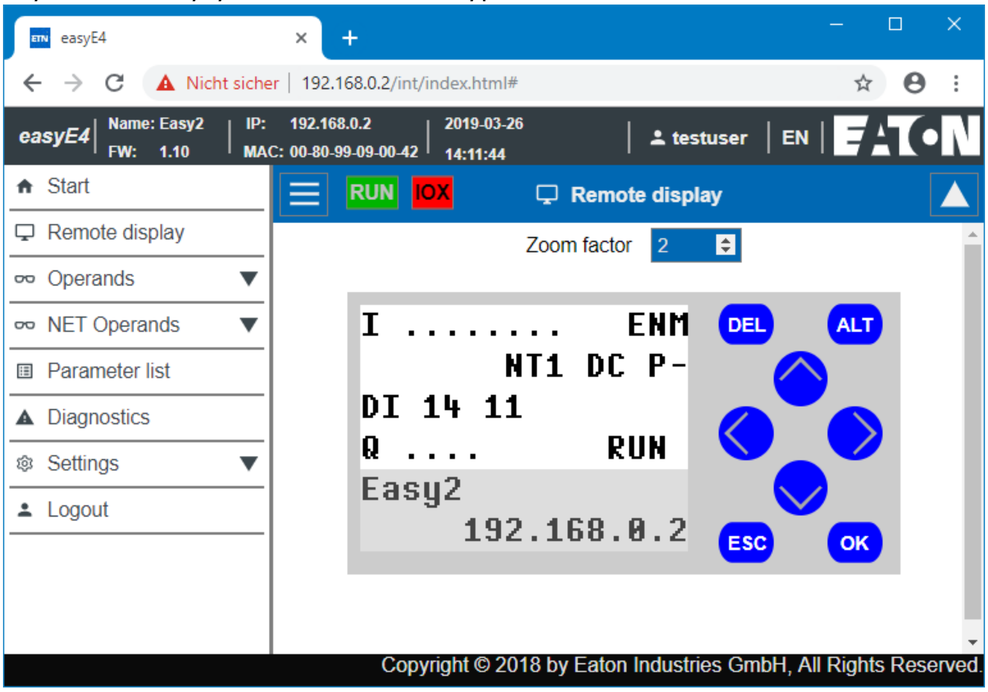



Figure 281: Device display

### Zoom increment

There is an option to zoom in 0.25 (25%) increments. By default, the zoom range is set to 2 and has a value range from 0.25 to 15.75.

The zoom level will be stored locally in the web client even after closing the session.

## Operands

In the operating range, operands show the states of the local bit and value operands of the device, see → "Overview of operands for numeric formats", page 202.

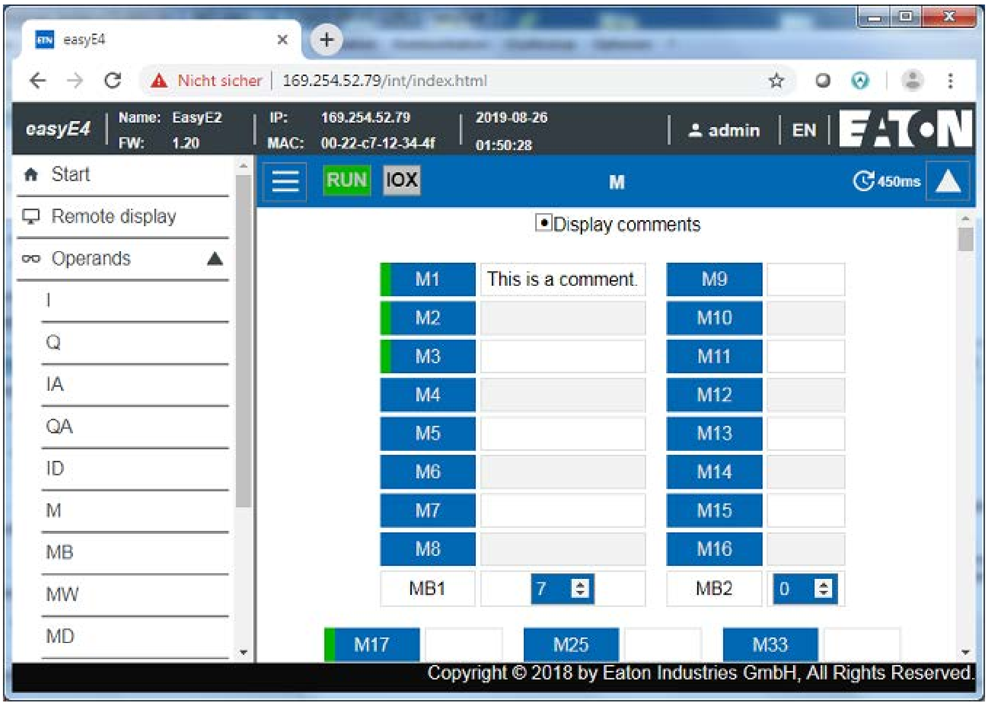



Figure 282: Operands

## NET operands

In the operating range, NET operands display the states of the local bit and value operands of the device or the NET bit and value operands of the other NET subscribers, see also → "Overview of operands for numeric formats", page 202.

The NET bit and value operands of the other NET subscribers are selected using the Select NET ID button. The drop-down menu will only show the NET IDs of the devices that are actually found on the NET. The web client only allows writing onto the NET operands of the local device. The NET operands of other NET subscribers are read only.

You can click on the NETWebClient button to connect to the web server for the NET station that is selected with the Select NET ID button. This will open a second web client without requiring for the IP address to be entered. After logging in, the NET station will become the local device for the web client, making it possible to write to the corresponding NET operands.

→ In order to emphasize which device the web client is currently connected to and which operands are being shown, we recommend assigning device names, e.g., "EasyE2".

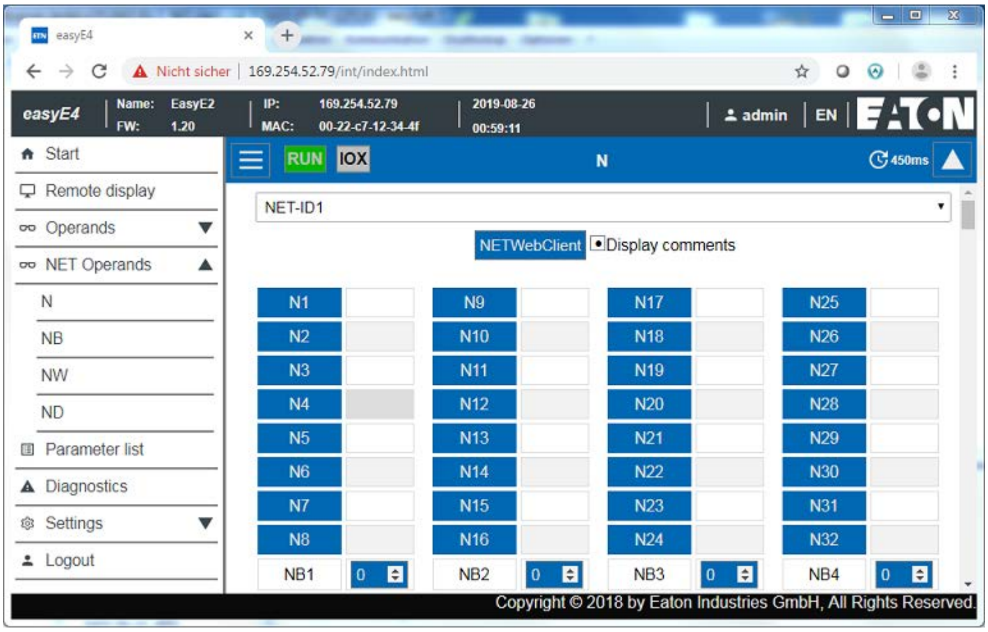



Figure 283: NET operands

## Parameter list

The project in the device must allow access in order to display this menu item. To allow access, enable the Parameter list enabled option in the *Project view/Webserver tab* (refer to → " Parameter list enabled", page 647 as well) or enable it in *Catalog settings/Web client/Separate operands* in the web client (please refer to → "Parameter list", page 668 as well).

The web client has the option of putting together a custom view of the easyE4 base device's and its expansions' operands. This view is defined in the parameter list. The parameter list can be made up of all available operands, i.e., EASY-E4-... operands, I/O expansion for easyE4 control relays operands, NET operands, and function block operands. User function blocks UF are excluded. The parameter list is stored in the browser's local memory and not in EASY-E4-.... The parameter list is retained when visiting the site again in the browser.

Each web client has its own parameter list.

→ If a parameter list, domain name, or device name is very long, the corresponding request will be broken down into multiple small queries and will require multiple cycle times.

The parameter list can be exported and imported. This allows it to be transferred from one browser, PC, web client, or mobile device to another.

The parameter list can contain a maximum of 18658 different entries. In order to prevent requests for the easyE4 base device from being unnecessarily long, however, make sure to keep the parameter list as short as possible.

In addition, parameter list operands that corresponding to function block inputs or outputs will be highlighted with a red box:

| | |
|---|---|
| FB.A01.F1 | Shows that the selected operand in the parameter list is not being used in the easyE4 base device's program. The value is set to "0". |

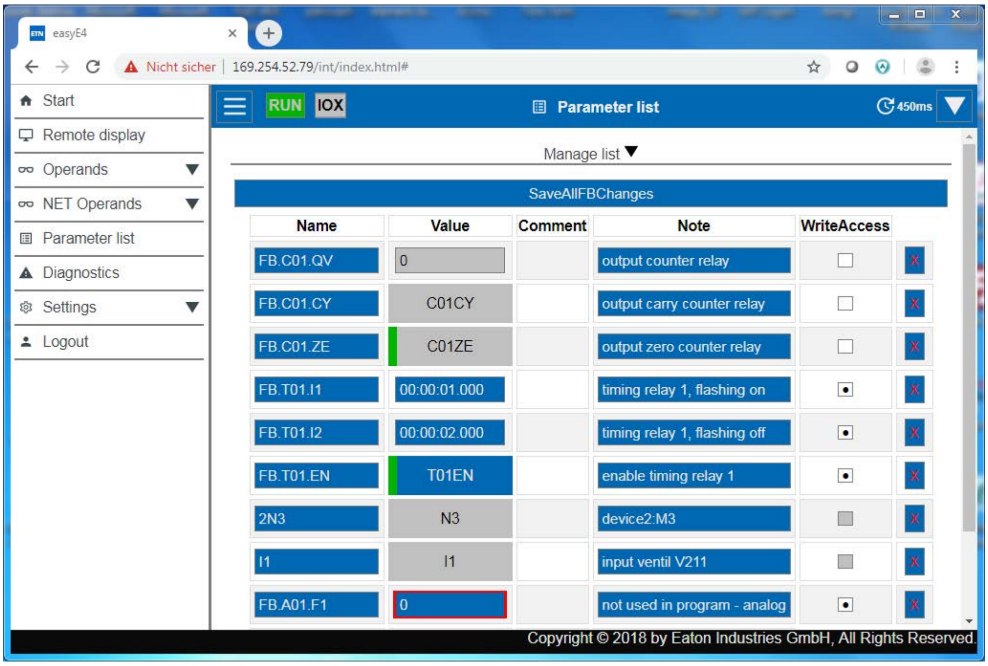
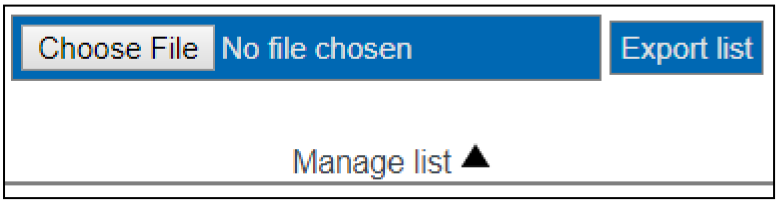



Figure 284: Separate operands

| Column | Description |
|---|---|
| Name | Any random operand can be entered in the Name column. The context-sensitive search supports the input by displaying all of the operands supported by easySoft 7 that contain the entered text anywhere in the operand or in the comment.<br>The proposed text can be assumed by performing the following steps:<br>• Use the arrow keys ↑ and ↓ to navigate through the proposed entries<br>• Make a selection with a click of the mouse and hit Enter. |
| Value | Depending on the device's operating status, the states of the selected operands are displayed in the workspace.<br>In the case of digital operands, the name of the operand will be shown. For a status of 1, there will additionally be a green bar in the field, e.g., T01EN. For a status of 0, no bar will be shown.<br>In the case of analog operands, the operand's current value will be shown.<br>Specifically in the case of function block inputs and outputs, a red box will be shown if the operand is not being used in the program on the device. The operand's value will then be set to "0", e.g.,<br>FB.A01.F1<br>0 |
| Comment | A comment is displayed for each operand stored in the program on the device. |
| Note | A comment can be entered that is only stored in the browser. Notes are exported and imported along with the parameter list. |
| Write access | This option is available exclusively to the administrator.<br>The administrator can enable and disable write access for all writable operands in the parameter list in order to set up permissions for another person. To do this, the parameter list needs to be exported and then imported into the browser being used by other people. |

#### Manage list




| Column | Description |
| --- | --- |
| Pick File | You can import a previously exported JSON file that contains the **parameter list**. |
| None selected | As soon as a **parameter list** is loaded, the filename will be shown here. |
| Export List | The file "OwnsOps.json" is saved. Depending on the browser settings, the file is saved in the folder provided for the download. After this, you can archive the file, open it with a text editor, or provide it to other people so that they can import it. |

#### Save temporary changes permanently

Clicking on the SaveAllFBChanges button will result in all the changes made in all web clients since the last time the easyE4 base device was started being retentively copied to the device.

Only digital constant, analog constant, and time constant values will be copied.

→ Changes that come from other web clients and that have been made across several sessions will also be copied.

This means that the modified constants will be immediately available on the device and will be retained the next time the device is started.

## Diagnostics

Diagnostics displays which diagnostics operands are set and their meaning. In the web client, the values listed in the Bit column are equal to the diagnostics operands. For more information on diagnostics options, see also → "Operating system diagnostic messages", page 609.

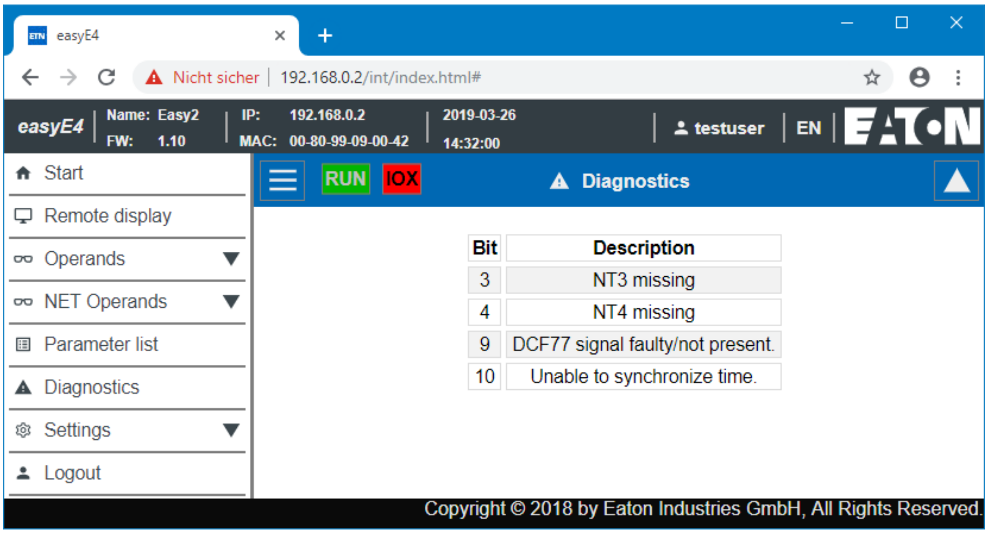



Figure 285: Diagnostics

### Settings

The fields highlighted in blue can be edited: Only the admin can modify the settings. The following settings can be displayed:

- General settings
- Network settings
- E-mail settings
- API key (is only displayed for the admin)
- Web client (is only displayed for the admin)

#### General settings

The admin can modify the device name, device date and device time. Any changes in the web client must be confirmed after a query. The modified data is not transferred to the device until then. The standard user has read only access to the General settings.

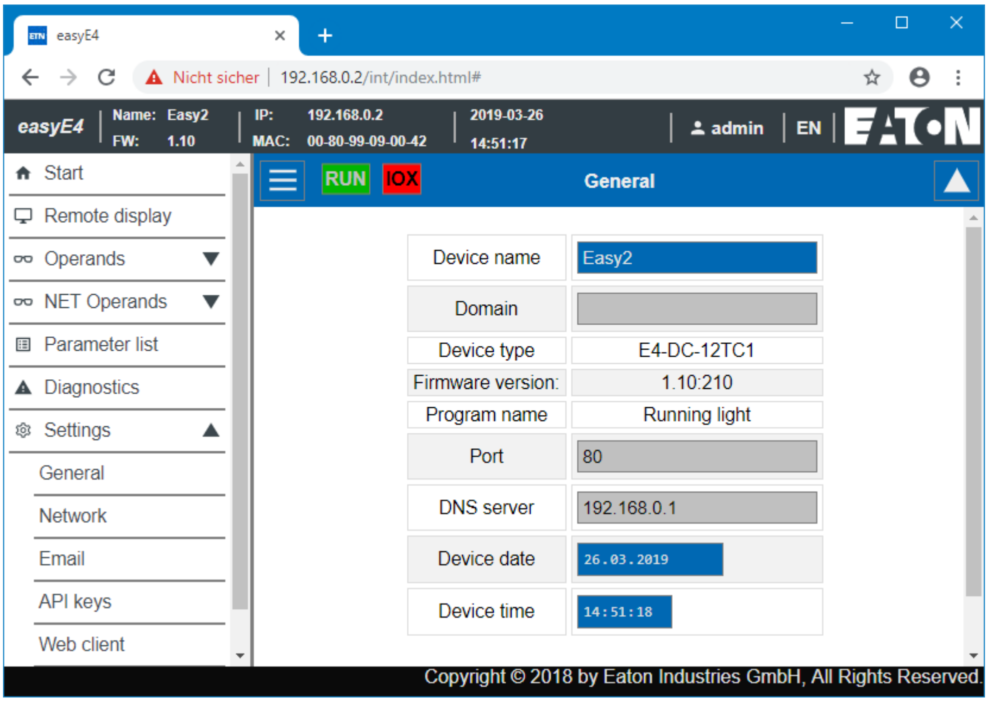



Figure 286: Web client - General settings

#### Network settings

The admin can modify the network settings, make changes to the IP address, subnet mask and the gateway's IP address. Any changes in the web client must be confirmed after a query. The modified data is not transferred to the device until then. The standard user has read only access to the Network settings.

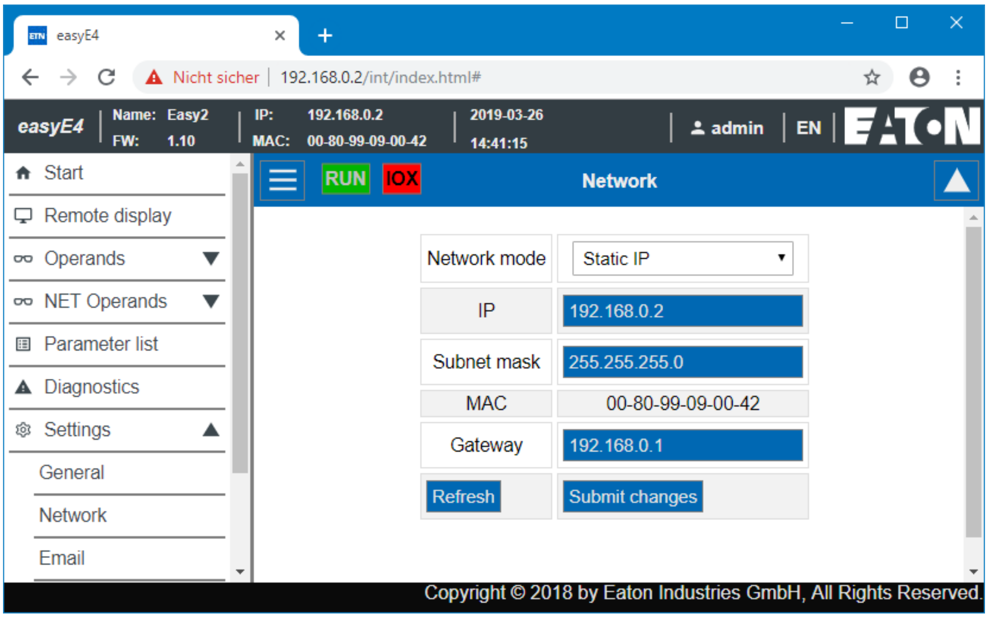



Figure 287: Web client - Network settings

### E-mail settings

The admin can modify the mail server's email settings. These are the same parameters that are configured in easySoft 7 *Project view/Email tab/Mail server settings area*, see also Email tab. These are the IP address or the DNS name of the mail server, mail server domains, encrypted mail server connection, login name or user and login password of the mail server user and the mail server port. All changes in the web client must be confirmed after a query. The modified file is then transferred to the device. The standard user has read only access to the email settings.

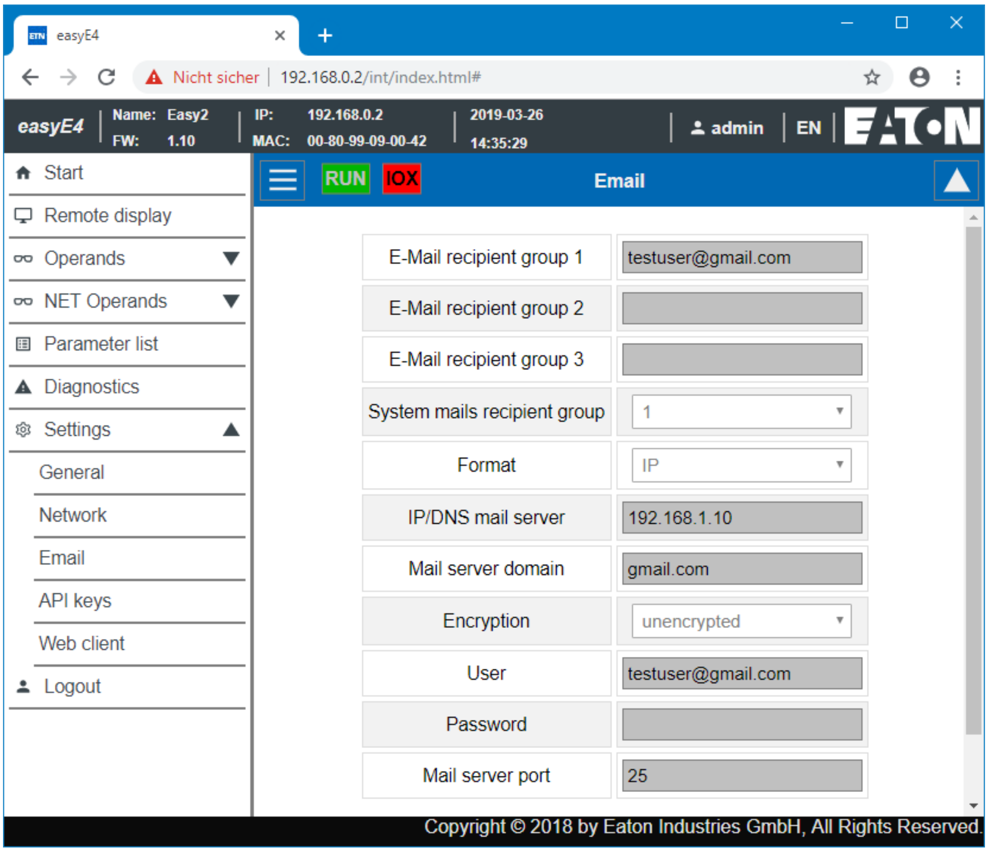



Figure 288: Web client - E-mail settings

### API key

Only the administrator can create an API key. An API key can be created for any user in the web client's workspace.

The web server offers the option for an application programming interface JSON API. Any program can access and edit the easyE4 data from this interface, for example the program of an enterprise software. easySoft 7 is not required. The API can be used in all high-level languages that provide a library for HTTP GetRequests, for example Java Script, Python, VBa, C++.

A software that wants to access the application programming interface can authenticate itself in two different ways:

1. With the web client's user name and password
   <User name web client>:<Password user name web client>@<IP address device>.api/…
   Example: testuser:$myPasswd@192.168.0.2.api/get…
2. API key
   <API key>@<IP address device>.api/…
   Example: FTZKVUGUBGLIUIHGIGIZZTIUFFZKUFTABC@192.168.0.2.api/get…

The JSON API is described in a separate document – please refer to www.eaton.eu-/easy-jsonapi.

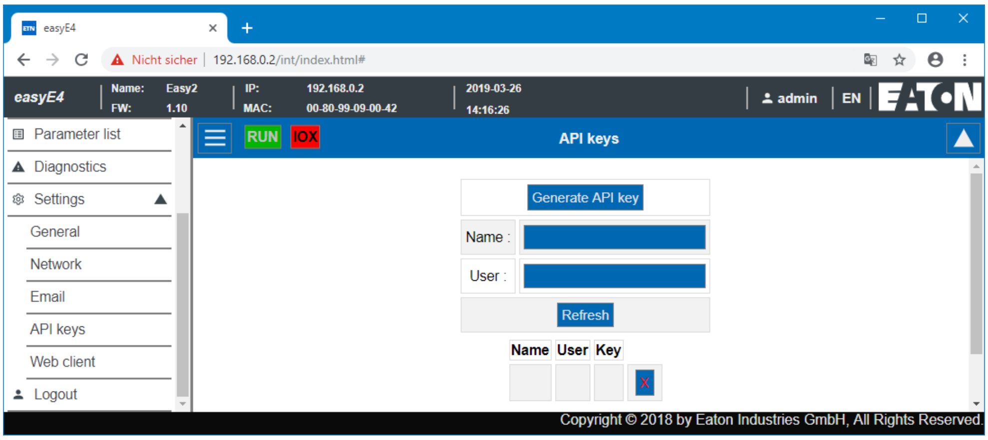



Figure 289: API key

### Web Client

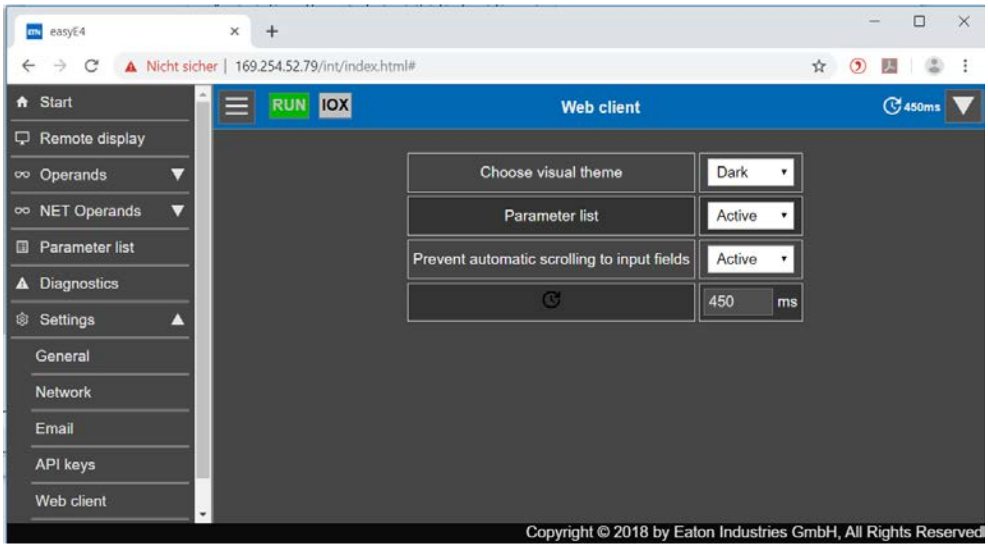



Figure 290: Web Client

### Select display topic

- White - The web client's interface is presented bright.
- Dark - The web client's interface is presented in dark gray.

### Parameter list

- Active
  If this option is set to Enabled, creating a parameter list will be allowed. The menu item Separate operands can be accessed in the web client's catalog. This option is equivalent to the option Parameter list active in *Project view/Web server tab*, see also → " Parameter list enabled", page 647.
- Inactive
  If this option is set to Disabled, creating a parameter list will not be allowed. The menu item Separate operands is not displayed in the web client's catalog. This

option is equivalent to the option Parameter list active in *Project view/Web server tab*, see also → " Parameter list enabled", page 647.

**PreventInputScroll**

- Active
  If the cursor is placed inside an input field in the web client, the display will not scroll, and the way the fields are being displayed will remain unchanged.
- Inactive
  Default setting. If the cursor is placed inside an input field in the web client, the display will scroll automatically so that the input field is shown in the center.

### Web client cycle time

The web client's cycle time is the time between two requests for the device used to update local data. In the following screen refresh cycle, the modified data is shown in the web client (please note that the cycle time for the web client and the screen refresh cycle time and independent from each other). The value range for the web client's cycle time is 250 ms to 30000 ms, with the default value being 450 ms.

The web client's cycle time can be shortened if data needs to be shown in the web client faster than the default value and the program is able to do this with its cycle time.

→ Please note that reducing the web client's cycle time can place an excessive load on the easyE4 device and block device responses under certain circumstances.

Changes in the web client will not affect the settings in the project. However, they will be saved in the browser and retained even after the session is closed.

**See also**

→ Section "Setting up a web server", page 646
→ Section "AL - Alarm function block", page 440

## 10.5 E-mail function

Only possible with easySoft 7.

The e-mail function can be used to have the easyE4 control relay send a message to up to three different recipient groups.

Precondition:
In order for the e-mail function to work, there must be an Ethernet connection between the easyE4 control relay and an server.
This server must either provide the mail server itself or the connection to the mail server that will be used.

The e-mail message will be triggered:

- If an error occurs in the NET group (all devices located on the same network as the easyE4)
- The controller's operating state changes
  or
- The program is deleted

In addition, e-mails can be sent to recipients if an alarm function block has been configured in the relevant program.

Since the easyE4 control relay cannot send any messages itself, the e-mail function is the ideal way to ensure that the defined people will promptly receive a notification when required.
These notifications will be triggered automatically if there is an active connection between the easyE4 and a mail server and the latter has been configured accordingly.

In addition, the e-mail functionality comes with the advantage of traceability. This traceability can be viewed much the same way as data logging.
The following will be saved:

- When an error occurred
- When the operating state changed
- When programs were deleted

## Register e-mail

All settings required to send an email are made in *Project view/Email tab*. The email time stamp incorporates the time zone configured for the device location, please refer to Project view.

*Project view/E-Mail tab*

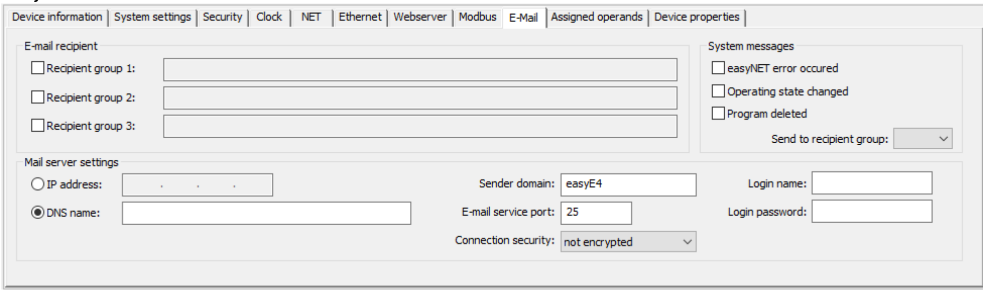



Figure 291: Register e-mail

### E-mail recipient

You can enter up to three recipient groups.

A recipient group can contain one recipient or several recipients separated with a semi-colon.
The definition for each recipient group can have a maximum length of 254 bytes. There is an activation bit for each recipient group. When an email is triggered (e.g., by the alarm function block), emails are only sent to the recipient groups that are activated.

### Mail server settings

You will need to enter the connection data for the mail server in the Mail server settings section. If the settings do not match, it will not be possible to send the easyE4 system messages. The mail server can either be defined with an IP address or with the DNS name (preferably).

- DNS name (64 bytes) or IP address of the mail server;
  Enter the mail server's full name, e.g., "smtp.gmail.com"
  DHCP mode or a DNS server is required in order to use DNS names. The DNS server encrypts the DNS name of the mail server and links it to the correct IP address. Thus, the DNS server establishes the connection to the mail server. In this case, the DNS server's IP address must be defined in *Project view/Ethernet tab*.
- Sender domain (64 bytes); "easyE4" by default
  Enter the hostname or the domain for the easyE4 device itself as the sender domain. This information will be used to log in to the mail server.
- Email service port of the SMTP server;
  The service port will depend on the selected security option. If an external provider is used for the email service, the service port must be queried from the relevant provider.

For example, Gmail uses port 587 for the connection security STARTTLS and port 465 for SSL/TLS.

- Connection security:
  - not encrypted
  - STARTTLS
  - SSL/TLS (most common connection security protocol)

The DNS name, mail server domains and service port are defined by the email provider.
Often times, you will be able to find the entire domain name by running a simple Internet search by <SMTP server> followed by the mail server; for example Yahoo, Googlemail, gmx.

An email account must definitely be set up with the mail server. If easyE4 should transfer the email via a public network, an email account must be set up with a provider. The login data must be entered in the following fields for the email account:

- Login name (32 Byte)
- Login password (32 Bytes)

A check mark next to the field Login password shows that you repeated the password correctly.

**System Messages**

You can define for which events easyE4 is to send emails in the system messages area.

☐ easyNet error occured

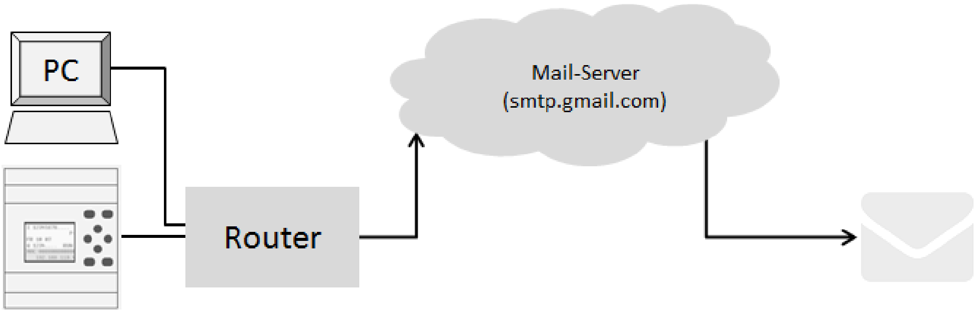
☐ Operating state changed

☐ Program deleted

Sending to the recipient group
With this ID the recipient group is selected to which easyE4 should send emails if one of the defined trigger events occurs.

## Example: Sending an email with easyE4 if there is an operating mode change

In the following example, an easyE4 base device should send an email if the operating mode changes.




IP: 192.168.0.2 IP: 192.168.0.1

Email account with Gmail:
Address: alarmEasyE4@gmail.com
Password:: • • • • • • •
SMTP (Simple Mail Transfer Protocol)

STARTTLS Port:587

Recipient:
testuser1@eaton.com

Prerequisites
You have set up an email account with a provider and you know the port for the STARTTLS connection security.

To complete this example, follow the steps below:

**Settings in the email tab**

You can configure the e-mail function as necessary in easySoft 7.

- ▶ Open a new project.
- ▶ Select the device you want from the catalog in the *Project view*.
- ▶ Click on the E-Mail tab.

There are three sections under the tab: E-mail recipient, Mail server settings, and System messages.

- ▶ Activate a recipient group for editing, such as <Recipient group1>.
- ▶ Enter the recipient's e-mail address, <testuser1@eaton.com>, for example.

In the System messages section, select the events that should trigger an e-mail to the recipient group.

- ▶ Enable the operating mode option with a check mark.
- ▶ In the Send to recipient group drop-down menu, select the recipient group to which the selected messages should be sent, e.g. <1>.

You will need to enter the connection data for the mail server in the Mail server settings section. In this example, the mail server is a Gmail server smtp.gmail.com.

- Start by selecting whether you will be entering an IP address or a DNS name. The DNS name that you want to activate is entered in this example.
- Enter the DNS name in the field <smtp.gmail.com>.
- Confirm or change the sender domain for the easyE4 base device.
- Enter the email service port; for example, Gmail uses port 587 for the connection security STARTTLS and port 465 for SSL/TLS.
- Select the connection security, for example STARTTLS.
- In the field Login Name, enter the address of your email account through which easyE4 should send the email.
- In the field Login Password, enter the password for your email account through which easyE4 should send the email.
  A check mark next to the field Login password shows that you repeated the password correctly.
- DHCP mode or a DNS server is required in order to use DNS names. The DNS server encrypts the DNS name of the mail server and links it to the correct IP address. Thus, the DNS server establishes the connection to the mail server.

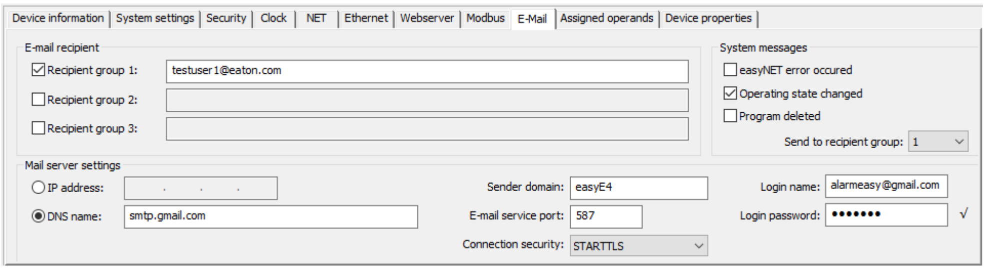



Figure 292: Email tab with the settings from the example

Upper and lower case do not play a role when naming email accounts.

**Ethernet tab settings**

First, enter the parameters for communicating with the device.

Because the mail server is entered with the DNS name in the example, the DHCP mode or a DNS server is required to establish a connection to the mail server.

- Switch to *Project view/Ethernet tab*.
- In the selection field mode, select the option Fixed IP address .
- Enter the IP address of the easyE4 base device, for example 192.169.0.2.
- Enter the subnet mask, such as 255.255.255.0.
- Enter the IP address of the router in the Gateway field. It establishes the connection between the easyE4 and the public network.
- Set the check mark to activate the option Enable configuration via network. This will allow you to modify the IP settings in the Search for devices window in the Communication view during testing.
- Enter a name for the easyE4 base device in the field Device name, such as <myEasyE4>. The Device name is noted in the email.

- Enter the IP address of the router in the DNS server field. In this example, the DNS server is equivalent to the router, because it establishes the connection to the public network and from the device perspective, will establish the connection to the DNS server. The DNS server encrypts the DNS name of the mail server and links it to the correct IP address.

→ Make sure that the IP addresses for the PC, easyE4 and the router are in the same number range.
You may need to adjust the system settings of your PC.

*Project view/Ethernet tab*

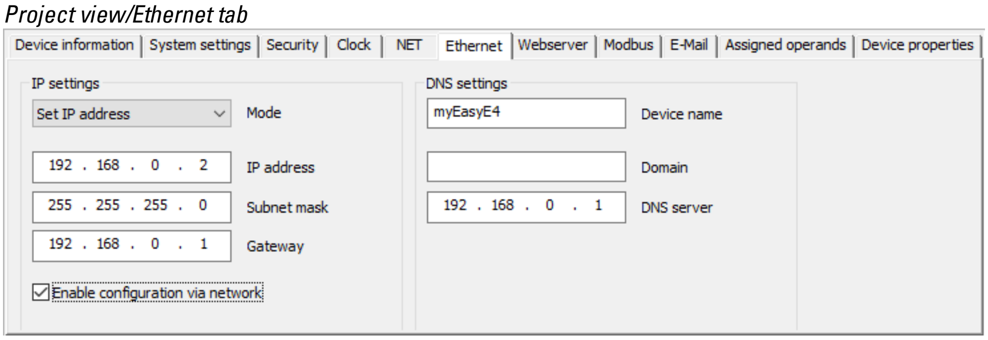



Figure 293: Ethernet tab with settings from the example

### Programming

Before you can load your project onto the easyE4 base device, you must first create a small program. Otherwise, the plausibility check will report an error.

- Switch to *Programming view*.
- Select the programming language, preferably FUP or KOP.
- Drag a N/O to the work pane, such as I01.
- Drag a contactor to the work pane, such as Q01.

### Establish a connection to easyE4 and load the program onto easyE4

- Switch to *View communication*.
- In the range, select the IP address of the easyE4 base device, such as 192.168.0.2.
- Click on the Online button.

When the device is online, the illustration of the easyE4 on the work pane changes.

- Press the PC-> device button to load the program to the device.
- Switch the Status Display On using the command sequence *Communication menu bar/ Status display on*.
- Click the RUN button to start the program.

### Trigger the event and send the email.

- Click the RUN button to start the program and to change the operating mode of the device.

▶ Check the incoming email folder to see whether an email is received shortly after this; such as testuser1@eaton.com.

Example of an email:

```
From: myEasyE4@local <alarmeasye4@gmail.com>
To: testuser1@eaton.com
CC:
Subject: [EXTERNAL] Device: myEasyE4- Enter RUN
```

```
Device : myEasyE4
Time : 2019-02-01 14:52:55
IP : 192.168.0.12
State : STOP

Message reason: Enter RUN
```

Figure 294: Email example when the operating mode changes

## Example: Sending an email with alarm function block AL

You now expand the previous example → chapter "10 E-mail function", page 670 by adding an alarm function block AL.

After pressing the P button P1 on the easyE4 base device, the easyE4 should send an email.

Precondition:
You have created a project using the example Sending email with easyE4 when the operating mode changes.

To complete this example, follow the steps below:

**Program and configure the alarm function block**

- Make sure that the project from the example Sending email with easyE4 when the operating mode changes is open.
- Switch into View programming.
- Select the alarm function block AL from the catalog and left click it to drag it to the workspace.
- Select the alarm function block AL N/O from the catalog left click it to drag it to the workspace at input T_ of the AL01 function block.
- In the Contact tab in the selection list, select operand P- device key.
- Make sure that number 1-< is selected in the selection list.
- Left click on the alarm function block AL01. The tab Alarm function block parameters opens.
- In the field Subject, enter a text that describes the trigger event.
- In the field Message text enter any text with a maximum of 160 characters.
- Make sure that the ID of the desired recipient group is entered in the selection field Recipient assignment. Which recipient is assigned to which recipient group is defined in *Project view/Email tab*.

*Programming view/AL01*

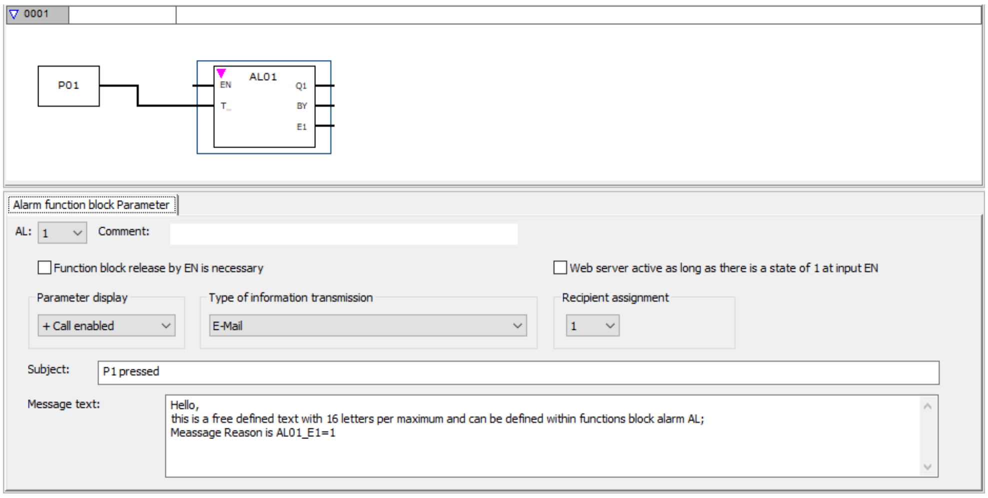



Figure 295: Alarm function block tab with parameters from the example and FUP program with alarm function block and P button P01

### Activating P buttons

- Switch into the system settings tab.
- Enable the P buttons option with a check mark. By doing so, you allow the program to read the condition of the P buttons on the device.

### Transfer program

- Save the project.
- Switch to Communication view and press the Online button.
- Stop the device by clicking on the *Program/configuration/*STOP.
- By clicking on *Program/configuration/*PC->device, you load the program onto the device.
- Start the device by clicking on the *Program/configuration/*RUN.
- To check whether the P button works properly, set the Status indicator ON from the *Communication menu bar/ Status indicator ON*.

### Trigger the event and send the email.

- Press the P button P1 < on the device to trigger the event.
- Check the incoming email folder to see whether an email is received shortly after this; such as testuser1@eaton.com.

Example of an email:

```
From: myEasyE4@local <alarmeasye4@gmail.com>
To: testuser1@eaton.com
CC:
Subject: [EXTERNAL] P1 pressed
```

```
Hello,
this is a free defined text with 160 letters per
maximum and can be defined within functions block
alarm AL; Message Reason is AL01_E1=1
```

Figure 296: Example email when triggered by alarm function block AL01

**See also**

→ Section "AL - Alarm function block", page 440

## 10.6 Convenient visualization for easyE4

For simple control applications, the easyE4 relay has the option of running the visualization through the display on the base device.
However, you can also use an XV-102-A0-35TQRB-1E4 touch display as an operating terminal for much more convenient project visualization.
This 3.5“ display is a cost-effective solution for distributed visualization applications.


Figure 297: Visualization on operating terminal

For more information on how to connect the controller, please refer to the tutorials and the various documents → Section "Further information ", page 709.

For additional product information, as well as access to the software demo version, please visit the product page.

- http://www.eaton.eu/easy
- http://www.eaton.eu/galileo

Data is exchanged between the devices using the GALILEO 10 internal tag import format (*.itf). easySoft 7 supports this export format for Modbus TCP..

# 11. Faults

This section provides troubleshooting information for your easyE4 in case it does not behave as expected.

| Fault | Cause | Remedy |
|---|---|---|
| The base device will not boot up | There is no supply voltage | Check the input wiring.<br>Switch on the device |
| The display stays or turns dark. | The backlight is deactivated. | Turn on the backlight; please refer to the text function block description or check the corresponding function in the program with easySoft 7. |

If a easyE4 device does not behave as expected, the following tips can help you in rectifying any possible problems. If a program does not function as expected, in spite of a thorough simulation in easySoft 7, the power flow display in the EASY-E4-...-12...C1(P) device enables you to test the logic operations of the circuit diagram.

Only qualified persons should test electrical voltages while the easyE4 device is in operation.

## 11.1 Messages from the operating system

| Messages on the LCD display | | Explanation | Remedy |
|---|---|---|---|
| No display | | Power supply interrupted | Restore power |
| | | LCD is faulty | easyE4 replace |
| Temporary display | | | |
| | TEST: EEPROM | Only when switched on for the first time | - |
| | TEST: CLOCK | | |
| | UPDATE ERROR | The operating system file (*.FW) does not match the selected easyE4 expansion device. | Select the operating system file "*.FW" corresponding to the expansion device on the microSD |
| Continuous display | | | |
| | ERROR: EEPROM | The memory for storing the retentive values or the easyE4 circuit diagram memory is faulty. | easyE4 replace |
| | ERROR: CLOCK | Clock error | easyE4 replace |

## 11.2 Possible situations when creating programs

| Situations when creating a program | Explanation | Remedy |
|---|---|---|
| Cannot enter contact or relay in program | easyE4 device is in RUN mode | Select STOP mode |
| Time switch switches at wrong times | Time or time switch parameters not correct | Check time and parameters |
| Message when using a memory card PROG INVALID | Memory card in easyE4 device contains no circuit diagram | Change easyE4 device type or the circuit diagram on the memory card |
| | Circuit diagram on the memory card uses contacts/ relays that the easyE4 device does not recognize | |
| Power flow display does not show changes to the rungs | easyE4 device is in STOP mode | Select RUN mode |
| | Association/ connection not fulfilled | Check and modify circuit diagram and parameter sets |
| | Relay does not activate coil | |
| | Incorrect parameter values/time | |
| | Analog value comparison is incorrect<br>Time value of timing relay is incorrect<br>Function of timing relay is incorrect | |
| Relay Q or M does not pick up | Relay coil has been wired up several times | Check coil field entries |
| Input not detected | Loose terminal contact | Check installation instructions, check external wiring |
| | No voltage to switch/button | |
| | Wire break | |
| | easyE4 device input is faulty | Replace easyE4 device |
| Relay output Q does not switch and activate the load | easyE4 device in STOP mode | Select RUN mode |
| | No voltage at relay contact | Check installation instructions, check external wiring |
| | easyE4 device power supply interrupted | |
| | easyE4 device circuit diagram does not activate relay output | |
| | Wire break | |
| | easyE4 device relay is faulty | Replace easyE4 device |

# 11.3 Event

| Event | Explanation | Remedy |
|---|---|---|
| The ACTUAL values are not being stored retentively. | Retention has not been switched on. | Switch on retention in the SYSTEM menu. |
| The RETENTION… menu is not displayed in the SYSTEM menu. | easyE4 device is in RUN operating mode. | Select STOP mode |
| The retentive data is cleared when there is a mode change from RUN to STOP. | This behavior only occurs when using the PW02 (pulse width modulation) function block in easyE4. | Avoid using the PW02 function block. |
| When the device is switched on, the easyE4 switches to STOP mode | No circuit diagram in the easyE4 device | Load, input circuit diagram |
| | RUN START is deactivated at easyE4. | Activating RUN mode in Menu SYSTEM OPTIONS. |
| The contacts of the BC (data block comparison) and BT (block transfer) function blocks flash in the power flow display | The display of the easyE4 is being updated to intermediate states too frequently although the contacts are operating properly | Ignore this section of the power flow display. |
| The display is not showing anything | No supply voltage | Switch on the power supply |
| | easyE4 device faulty | Press the OK button. If no menu appears, replace the easyE4 device. |
| | Text displayed with too many spaces | Enter text or do not activate text output |

## 11.4 Functionality of the NET faulty

*NOTICE*
You can make a visual check the functionality of the NET by means of the NET LED and in the circuit diagram via diagnostics bit ID01-ID08.

Checking the functionality of the NET using the NET LED

| Status of the NET LED | Description |
|---|---|
| Off | NET not operational, fault in configuration |
| Continuous light | NET station fault - possible causes:<br>• Net is initialized and at least one station has not been detected. Check the plug-in connections.<br>• You have modified the NET ID or baud rate for at least one station after the con-figuration has been completed. Change the configuration.<br>• You have deleted the program on a NET station and thus also its NET configuration. Reconfigure the NET via station 1.<br>• You have expanded an existing NET station and replaced it with a new device which cannot be assigned parameters. |
| Flashing | NET operating fault-free |

# 12. Maintenance

## 12.1 Cleaning and maintenance

The easyE4 are maintenance-free.
However, the following work may need to be carried out:

- Cleaning the easyE4 when soiled.

When soiled:

**CAUTION**
**POINTY, SHARP OBJECTS AND CORROSIVE LIQUIDS**
When cleaning the device:

- Do not use any pointy or sharp objects (e.g., knives).
- Do not use aggressive or abrasive cleaning products or solvents.

Make sure that no liquids get into the device (short-circuit hazard) and that the device is not damaged in any way.

▶ Clean the device with a clean, soft, damp cloth.

## 12.2 Repairs

Contact your local supplier or technical support for repairs.

**CAUTION**
**DESTRUCTION**
The easyE4 should only be opened by the manufacturer or by an authorized center. Operate the device until only with the enclosure fully closed and sealed.

Use the original packaging to ship the device.

# 12.3 Storage, transport and disposal

## 12.3.1 Storage and transport

**CAUTION**
**UV LIGHT**
Plastics will become brittle when exposed to UV light. This artificial aging will reduce the easyE4 unit's lifespan. Protect the device from direct sunlight and other sources of UV radiation.

**CAUTION**
**SHORT-CIRCUIT HAZARD**
If the device is or has been exposed to environmental fluctuations (ambient temperature, air humidity), condensation may form on or inside it. As long as this condensation is present, there will be a short-circuit hazard. Do not switch on the device when it has condensation in or on it.
If the device has condensation in or on it, or if the panel has been exposed to environmental fluctuations, let the panel settle into the existing ambient temperature before switching it on.

The ambient conditions must be met when transporting and storing the easyE4.

The ambient air temperature for storage and transportation must not exceed the maximum specified limit:

| Ambient climatic conditions | |
|---|---|
| Air pressure (in operation) | 795 - 1080 hPa |
| | Max. 2000 m above sea level |
| Temperature | |
| Operation | - 25 – +55 °C (-13 – +131 °F)<br>The display is readable between ϑ -5°C (-23°F) ≦ T ≦ 50°C (122°F). |
| Storage / Transport | - 40 – +70 °C (-40 – +158 °F) |
| Humidity | Relative humidity 5 - 95 % |
| Condensation | Prevent condensation by means of suitable measures |

→ Before commissioning
If storing/transporting the device in cold weather conditions or in such a way that it will be exposed to extreme differences in temperature, make sure that no condensation forms on or inside the device.
If there is condensation in or on the device, do not switch on the device until it is completely dry.

Use the original packaging to ship the device.

The easyE4 is sturdily built, but the components inside it are sensitive to excessively strong vibrations and/or mechanical shock.

Accordingly, make sure to protect the easyE4 from mechanical loads that exceed the scope of the unit's intended use.

The device should only be transported in its original packaging after being packed properly.

### 12.3.2 Disposal

**Important!**

Dispose of recyclables as required by your local recycling regulations.

easyE4 no longer being used must be professionally disposed of as per local standards or returned to the manufacturer or relevant sales department.

Materials used in the packaging

| Packaging | Material |
|---|---|
| Outer packaging | Cardboard |
| Inner packaging | Cardboard<br>Plastic bag: polyethylene (PE) |

# Appendix


A.1 Dimension and weight specifications ........ 694

A.2 Approvals and declarations ........ 698

A.3 Technical data ........ 699

A.3.1 Data sheets ........ 699

A.4 Required memory for function blocks ........ 705

A.5 Further information ........ 709

A.6 Sample Projects ........ 711

A.6.1 easyE4_Lauflicht_EDP.e70 application example ........ 711

Alphabetical index ........ 714

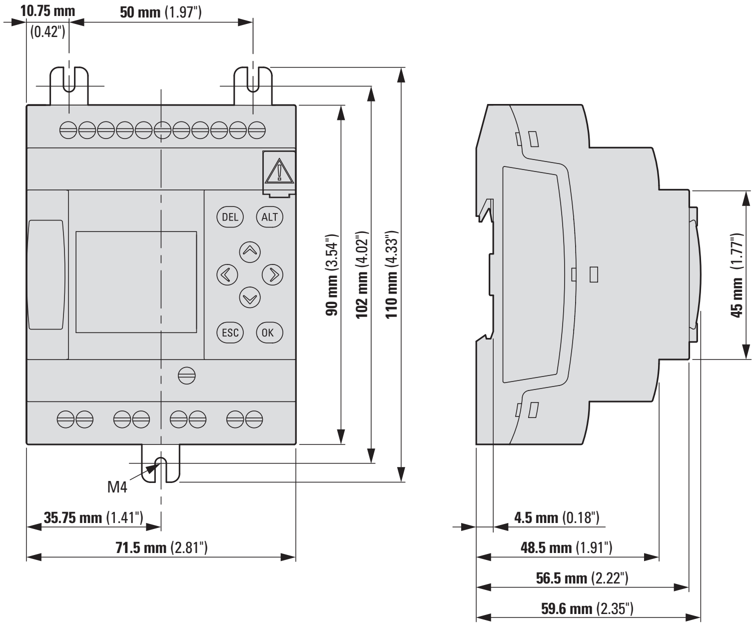
List of Figures ........ 726

Glossary ........ 737

# A.1 Dimension and weight specifications

*Base devices with 4 space unit front dimension*

EASY-E4-UC-12RC1(P), EASY-E4-DC-12TC1(P), EASY-E4-AC-12RC1(P)




EASY-E4-UC-12RCX1(P), EASY-E4-DC-12TCX1(P), EASY-E4-AC-12RCX1(P)

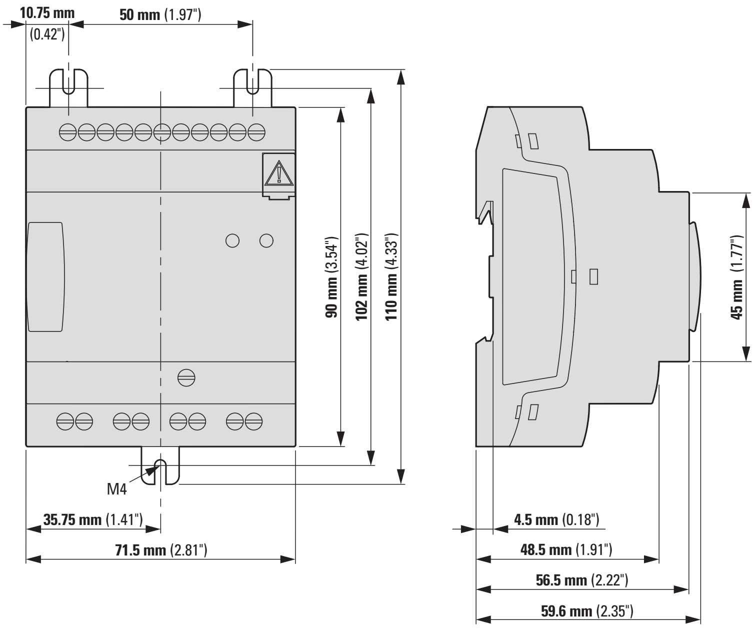



Figure 298: Dimensions in mm (Inch) base devices

| | |
|---|---|
| Width x Height x Depth | 71.5 mm x 90 mm x 58 mm (2.81" x 3.54" x 2.28") |

| | |
|---|---|
| (without plug) | |
| Weight | Refer to the data sheet for the device; varies between 139 g and 230 g depending on the specific model |

*Expansion devices with 4 space unit front dimension*

EASY-E4-UC-16RE1(P),EASY-E4-DC-16TE1(P), EASY-E4-AC-16RE1(P),

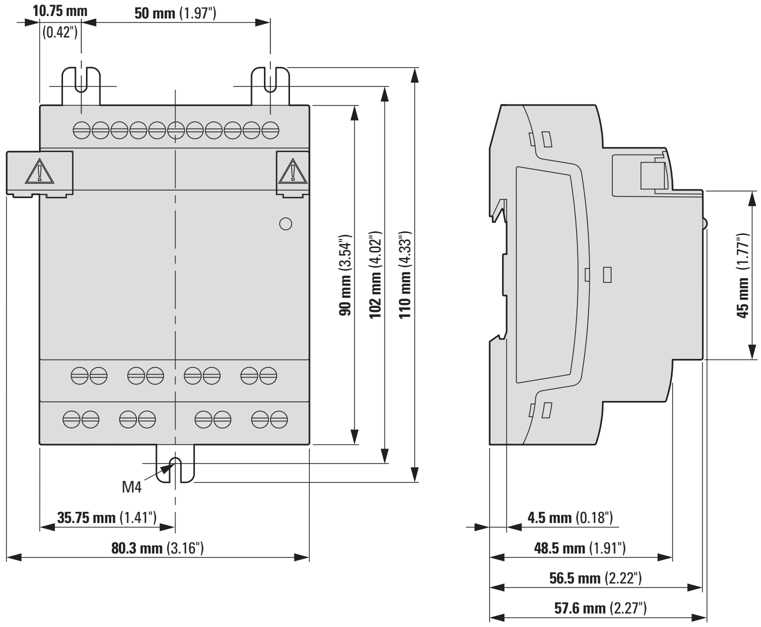



Figure 299: Dimensions in mm (Inch) extensions 4SU

| | |
|---|---|
| Width x Height x Depth (without plug) | 71.5 mm x 90 mm x 58 mm (2.81" x 3.54" x 2.28") |
| Weight | Refer to the data sheet for the device; varies between 139 g and 230 g depending on the specific model |

*Expansion devices with 2 space unit front dimension*

EASY-E4-UC-8RE1(P), EASY-E4-DC-8TE1(P), EASY-E4-DC-6AE1(P), EASY-E4-AC-8RE1(P)

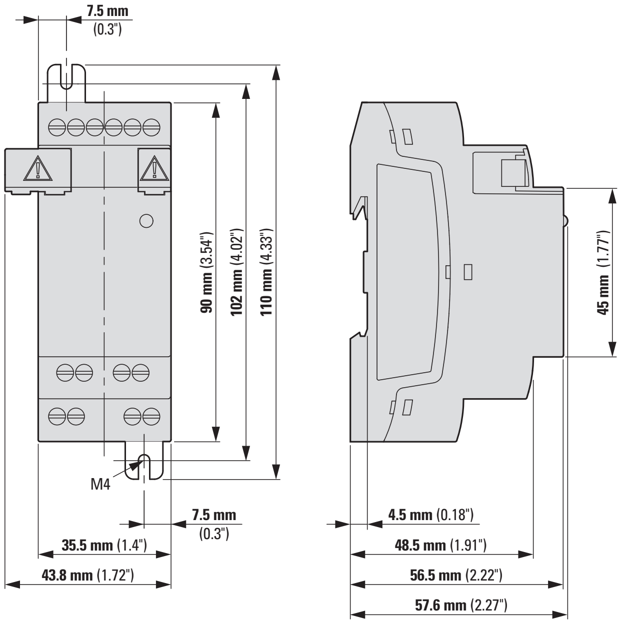



Figure 300: Dimensions in mm (Inch) extensions 2SU

EASY-E4-DC-4PE1(P)

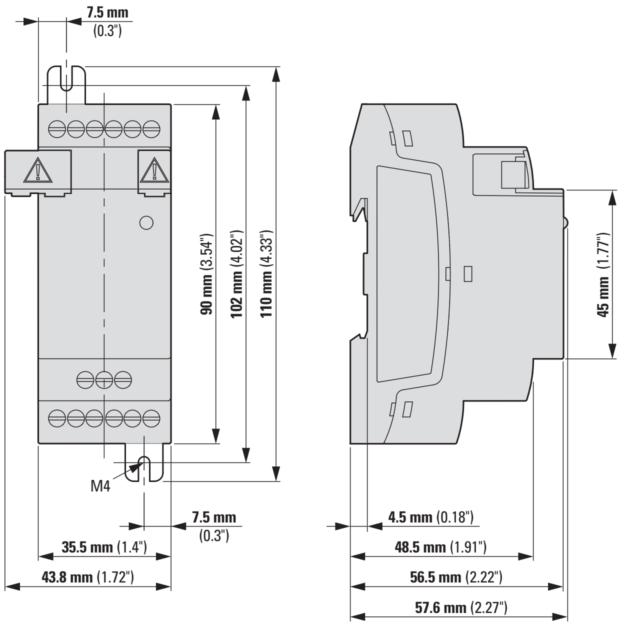



Figure 301: Dimensions in mm (Inch)

| | |
|---|---|
| Width x Height x Depth (without plug) | 35.5 mm x 90 mm x 58 mm (1.4" x 3.54" x 2.28") |
| Weight | Refer to the data sheet for the device; varies between 79 g and 232 g depending on the specific model |

## A.2 Approvals and declarations

The following specifications apply to all easyE4 devices.

| Approvals and declarations | |
|---|---|
| cUL | easyE4 approval present |
| CE | easyE4 units comply with all applicable European Union (EU) Directives and feature the CE marking. |
| NEMA | easyE4 device complies with the applicable guidelines in North America |
| Marine approval (shipping classification) | Application for easyE4 approval submitted |

Marine approvals applied for:

| Base devices | from Version |
|---|---|
| EASY-E4-UC-12RC1 | 02 |
| EASY-E4-UC-12RCX1 | 02 |
| EASY-E4-DC-12TC1 | 02 |
| EASY-E4-DC-12TCX1 | 02 |
| EASY-E4-AC-12RC1 | 01 |
| EASY-E4-AC-12RCX1 | 01 |
| EASY-E4-...-12...C1P | 00 |
| EASY-E4-...-12...CX1P | 00 |

| I/O expansions | from Version |
|---|---|
| EASY-E4-UC-8RE1 | 03 |
| EASY-E4-UC-16RE1 | 03 |
| EASY-E4-DC-4PE1 | 01 |
| EASY-E4-DC-6AE1 | 03 |
| EASY-E4-DC-8TE1 | 03 |
| EASY-E4-DC-16TE1 | 03 |
| EASY-E4-AC-8RE1 | 01 |
| EASY-E4-AC-16RE1 | 01 |
| EASY-E4-...-...E1P | 00 |

→ Base and expansion devices with a lower version number than specified in the above table do not have a Marine approbation. For devices without Marine approbation the maximum contact discharge is 4 kV.

| Applied standards and directives | | |
|---|---|---|
| EMC (relevant for CE) | | 2004/108/EEC 2014/30/EU |
| | IEC/EN 61000-6-2 | Interference immunity for industrial environments |
| | IEC/EN 61000-6-3 | |
| Security | | |
| | IEC/EN 61010 | Safety requirements for electrical equipment for measurement, control and laboratory use |
| Product standards | | |
| | EN 50178_x | Electronic equipment for use in power installations |
| | IEC/EN 61131-2 | Programmable controllers: Equipment requirements and tests |
| Mechanical shock resistance | IEC/EN 60068-2-27 | 15g /11ms |
| Vibration | IEC/EN 60068-2-6 | Displacement amplitude: 5–9 Hz: 3.5 mm; 9–60 Hz: 0.15 mm<br>Acceleration amplitude: 60–150 Hz: 2 g |
| Environmental tests | IEC/EN 60068-2-30 | |

# A.3 Technical data

## A.3.1 Data sheets

The current specifications for the device can be found in the corresponding data sheet at www.eaton.eu/ecat

### A.3.1.1 Base devices

With screw terminal connection

| | |
|---|---|
| 197211 - EASY-E4-UC-12RC1 | 197212 - EASY-E4-UC-12RCX1 |
| 197213 - EASY-E4-DC-12TC1 | 197214 - EASY-E4-DC-12TCX1 |
| 197215 - EASY-E4-AC-12-RC1 | 197216 - EASY-E4-AC-12RCX1 |

With push in connection

| | |
|---|---|
| 197504 - EASY-E4-UC-12RC1P | 197505 - EASY-E4-UC-12RCX1P |
| 197506 - EASY-E4-DC-12TC1P | 197507 - EASY-E4-DC-12TCX1P |
| 197508 - EASY-E4-AC-12RC1P | 97509 - EASY-E4-AC-12RCX1P |

### A.3.1.2 Expansions

With screw terminal connection

| **with relay outputs** | **with transistor outputs** |
|---|---|
| 197217 - EASY-E4-UC-8RE1 | 197219 - EASY-E4-DC-8TE1 |
| 197218 - EASY-E4-UC-16RE1 | 197220 - EASY-E4-DC-16TE1 |
| 197221 - EASY-E4-AC-8RE1 | |
| 197222 - EASY-E4-AC-16RE1 | |

| **with analog inputs** | **with temperature inputs** |
|---|---|
| 197223 - EASY-E4-DC-6AE1 | 197224 - EASY-E4-DC-4PE1 |

With push in connection

| | |
|---|---|
| 197510 - EASY-E4-UC-8RE1P | 197512 - EASY-E4-DC-8TE1P |
| 197511 - EASY-E4-UC-16RE1P | 197513 - EASY-E4-DC-16TE1P |
| 197514 - EASY-E4-AC-8RE1P | |
| 197515 - EASY-E4-AC-16RE1P | |

| **with analog inputs** | **with temperature inputs** |
|---|---|
| 197516 - EASY-E4-DC-6AE1P | 197517 - EASY-E4-DC-4PE1P |

### Accessory devices

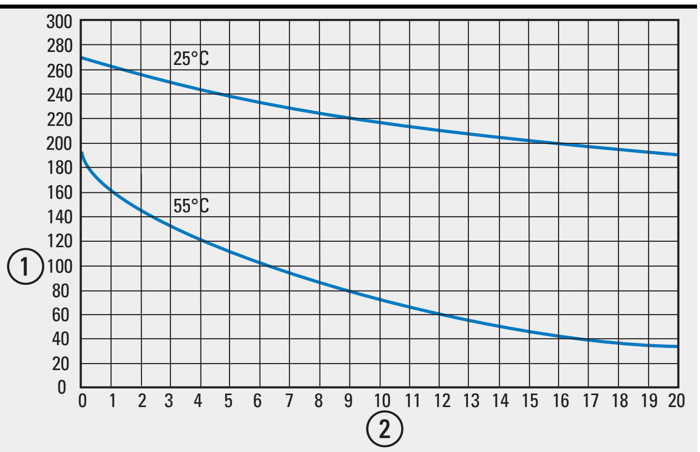
| Cat No. and type | Description |
|---|---|
| 198513 XV-102-AO-35TQRB-1E4 | Touch display for easyE4, 24 VDC, 3.5 Zoll, TFTcolor, Ethernet |
| 198514 XV100-BOX-E4-DC1 | Starter package consisting of EASY-E4-DC-12TC1, XV-102-AO-35TQRB-1E4 touch display, Ethernet switch, three patch cables, and license for easySoft 7 |
| 198515 XV100-BOX-E4-UC1 | Starter package consisting of XV-102-AO-35TQRB-1E4, EASY-E4-UC-12RC1, Ethernet switch, three patch cables, and license for easySoft 7 |
| 191087 MEMORY-SUD-A1 | microSD<br>2 GB memory card with adapter, I Grade, without an operating system |
| 197226 EASYSOFT-SWLIC | Programming software license easySoft 7 |
| 061360 ZB4-101-GF1 | ZB4-101-GF1 Device foot for screw mounting |
| 197225 EASY-E4-CONNECT1 | EASY-E4-CONNECT1 spare parts package<br>Consists of three (3) connectors and three (3) end covers for the easyE4 series between the control relay and input/output expansions |
| 272484 - TR-G2/24 | Transformer, 230 V, 12/24 V, 2/1 A |

Following are some of the technical specifications from the various data sheets. This information should help you get an overview of common properties and compare different individual devices.

| EASY-E4- | UC-12RC1 (P) | UC-12RCX1 (P) | DC-12TC1 (P) | DC-12TCX1 (P) | AC-12RC1 (P) | AC-12RCX1 (P) |
|---|---|---|---|---|---|---|
| Basic function | Control relay,<br>expandable with I/O expansions from the easyE4 series;<br>Ethernet port for connecting to a network<br>Real-time clock | | | | | |
| Display with keypad | Monochrome<br>6 x 16 lines | - | Monochrome<br>6 x 16 lines | - | Monochrome<br>6 x 16 lines | - |
| Specifications for connection to supply voltage | 12/24 V DC or<br>24 V AC | | 24 V DC | | 100 - 240 V AC or **DC (cULus 100 - 110 V DC)** | |
| Input points | Digital: 8,<br>of which 4 can be used as analog inputs | | | | | |
| Mounting | Tophat rail IEC/EN 60715 (35 mm) or screw fixing with fixing brackets ZB4-101-GF1 (accessories) | | | | | |
| Protection Style | IP 20 | | | | | |

| Ambient climatic conditions | |
|---|---|
| Air pressure (in operation) | 795 - 1080 hPa |
| | Max. 2000 m above sea level |
| Temperature | |
| Operation | - 25 – +55 °C (-13 – +131 °F)<br>The display is readable between ϑ -5°C (-23°F) ≦ T ≦ 50°C (122°F). |
| Storage / Transport | - 40 – +70 °C (-40 – +158 °F) |
| Humidity | Relative humidity 5 - 95 % |
| Condensation | Prevent condensation by means of suitable measures |

| Ethernet interface | on the basic device |
|---|---|
| Connection | RJ45 plug, 8-pin |
| Wire type | CAT5 |

| | |
|---|---|
| Back-up<br>of real-time clock | <br>① Backup time (hours) with fully charged double layer capacitor<br>② Service life (years) |
| Accuracy<br>of the real-time clock | typ. ± 2 s/day (± 0.2 h/year)<br>depending on ambient air temperature fluctuations of up to<br>± 5 s/day (± 0.5 h/year) are possible |

→ The supercapacitor's full charge will be reached if the easyE4 device is connected to the power supply for 24 hours.

### Electromagnetic compatibility (EMC)

| | | |
|---|---|---|
| Overvoltage category/degree of pollution | | III/2 |
| Electrostatic discharge (ESD) | | according to IEC EN 61000-4-2 |
| Air discharge | | 8 kV |
| Contact discharge | Version | |
| EASY-E4-UC-12RC1 | 01 | 4 kV |
| | from 02 | 6 kV |
| EASY-E4-UC-12RCX1 | 01 | 4 kV |
| | from 02 | 6 kV |
| EASY-E4-DC-12TC1 | 01 | 4 kV |
| | from 02 | 6 kV |
| EASY-E4-DC-12TCX1 | 01 | 4 kV |
| | from 02 | 6 kV |
| EASY-E4-AC-12RC1 | from 01 | 6 kV |
| EASY-E4-AC-12RCX1 | from 01 | 6 kV |
| EASY-E4-UC-8RE1 | 01 | 4 kV |
| | 02 | 4 kV |
| | 03 | 6 kV |
| EASY-E4-UC-16RE1 | 01 | 4 kV |
| | 02 | 4 kV |
| | from 03 | 6 kV |
| EASY-E4-DC-4PE1 | from 01 | 6 kV |
| EASY-E4-DC-6AE1 | 01 | 4 kV |
| | 02 | 4 kV |
| | from 03 | 6 kV |
| EASY-E4-DC-8TE1 | 01 | 4 kV |
| | 02 | 4 kV |
| | from 03 | 6 kV |
| EASY-E4-DC-16TE1 | 01 | 4 kV |
| | 02 | 4 kV |
| | from 03 | 6 kV |
| EASY-E4-AC-8RE1 | from 01 | 6 kV |
| EASY-E4-AC-16RE1 | from 01 | 6 kV |

→ The contact discharge value for all EASY-E4-..-....1P devices with push in terminals is 6 kV.

| | | |
|---|---|---|
| Electromagnetic fields (RFI) | according to IEC EN 61000-4-3 | 0.8 - 1.0 GHz: 10 V/m<br>1.4 - 2 GHz: 3 V/m<br>2.0 - 2.7 GHz: 1 V/m |
| Radio interference suppression | In accordance with EN 61000-6-3 | Class B |
| Burst | according to IEC/EN 61000-4-4 | Supply cables: 2 kV<br>Signal cables: 2 kV |
| Power pulses (Surge) | according to IEC/EN 61000- | 1 kV (supply cables, symmetrical) |

| | | |
|---|---|---|
| | 4-5 | 2 kV (supply cables, asymmetrical) |
| Radiated RFI | according to IEC/EN 61000-4-6 | 10 V |

## A.4 Required memory for function blocks

The memory required for unconnected function blocks is the same for all programming languages. In order to measure this memory when using the EDP programming language, drag each function block to the function block diagram individually and read the difference regarding the remaining memory space in the status line.

Each function block will reserve the memory space listed below when unconnected. In addition, text function block D has extensive static operating parameters that require additional memory space.

Table 124: Memory Required FB

| Function blocks | Instance 1 | Instance 2 |
|---|---|---|
| A | 68 | 68 |
| AC | 68 | 68 |
| AL | 540 | 290 |
| AR | 40 | 40 |
| AV | 60 | 60 |
| BC | 48 | 48 |
| BT | 48 | 48 |
| BV | 40 | 40 |
| C | 52 | 52 |
| CF | 48 | 48 |
| CH | 52 | 52 |
| CI | 52 | 52 |
| CP | 32 | 32 |
| D | 76 | 56 |
| DB | 36 | 36 |
| DC | 120 | 120 |
| DL | 92 | |
| FT | 56 | 56 |
| HW | 68 | 68 |
| HY | 68 | 68 |
| IC | – | – |
| IE | – | – |
| IT | – | – |
| JC | 20 | 20 |
| LS | 64 | 64 |
| MM | 48 | 48 |
| MR | 20 | 20 |
| MX | 96 | 96 |
| NC | 32 | 32 |
| OT | 64 | 64 |
| PM | 56 | 48 |
| PO | 96 | 96 |

| Function blocks | Instance 1 | Instance 2 |
|---|---|---|
| PW | 48 | 48 |
| RC | 76 | – |
| RE | 112 | 104 |
| SR(BIT) | 96 | 96 |
| SR(DWORD) | 96 | 96 |
| ST | 24 | – |
| T | 60 | 60 |
| TB | 112 | 112 |
| TC | 76 | 76 |
| VC | 48 | 48 |
| WT | 84 | 84 |
| YT | 96 | 96 |

**Required memory when connecting function blocks, using CP, T, D as an example**

In order to estimate the required memory for a connected function block, you can assume a required memory space of eight bytes for each connected function block input and function block output. This applies regardless of whether the function block inputs and/or outputs are digital or analog and of whether the connection involves MB marker bits or MD marker double words.

easySoft 7 will attempt to optimize the code, and in certain cases, less than eight bytes will be needed. Please note that connections between function blocks also require memory.

The following information was determined using the FBD programming language.

Table 125: Memory Required FB CP

| CP - Comparator | Connected to | Memory Required |
|---|---|---|
| **Function block inputs/outputs** | **Operand** | **bytes** |
| CP (not connected) | | 36 |
| EN | DI | 8 |
| I1 | AI | 4 |
| I2 | AI | 8 |
| LT | TH | 8 |
| EQ | TH | 8 |
| GT | TH | 4 |
| TOTAL | | 76 |

Table 126: Memory Required FB T

| T – Timing relay | Connected to | Memory Required |
|---|---|---|
| **Function block inputs/outputs** | **Operand** | **bytes** |
| T (not connected) | | 56 |
| EN | DI | 8 |
| RE | DI | 4 |
| ST | AI | 8 |
| I1 | AI | 12 |
| I2 | AI | 12 |
| Q1 | TH | 4 |
| QV | TH | 8 |
| TOTAL | | 112 |

The D text display function block heavily depends on the configured display and input elements and their texts. Every display and input element requires memory itself. Moreover, the texts available for the element also require memory. Identical texts in a display or input element do not require additional memory, however.

Table 127: Memory required by D text display function block

| D - Text display | Memory Required |
|---|---|
| **Function block inputs/outputs** | **bytes** |
| **Indication elements** | |
| D (not connected) | 108 |
| Value display | 12 |
| Bargraph | 20 |
| Static text (without text)<br>+ per text with 16 characters | 12<br>40 1) |
| Running text (without text)<br>+ per 2 characters | 8<br>4 |
| Rolling text (without text)<br>+ per text with 16 characters | 16<br>40 1) |
| Message text (without text)<br>+ per text with 16 characters | 24<br>40 1) |
| Date and time display | 12 |
| DZ Weekday | 8 |
| Timing relay value display | 12 |
| **Entry elements** | |
| Value entry | 12 |
| Latching pushbutton | 12 |
| Message text selection (without text)<br>+ per text with 16 characters | 28<br>40 1) |

| D - Text display | Memory Required |
|---|---|
| **Function block inputs/outputs** | **bytes** |
| Date and time entry | 8 |
| Timing relay value entry | 8 |
| 1) Potentially less memory required if optimization is possible | |

# A.5 Further information

### Documents

For more information on additional devices and modules, please refer to the following documentation:

| | | |
|---|---|---|
| PDF | Installation instructions Base devices | IL050020ZU |
| PDF | Installation leaflet for I/O expansions | IL050021ZU |
| PDF | Installation instructions Fixing bracket | IL05009005Z |

### Download Center, Eaton Online Catalog

Enter "easy" into the search box and the catalog will take you directly to the corresponding product group in the Automation, Control and visualization section.

http://www.eaton.eu/doc

http://www.eaton.eu/ecat

### Product information

For up-to-date information, please consult the product page on the Internet.

http://www.eaton.eu/easy

### Application examples

Support has provided a number of applications that are available for download as ZIP files from the Software Download Center.

Download Center - Software
http://www.eaton.eu/software/Anwendungsbeispiele/easy/Deutsch
http://www.eaton.eu/software/Application Samples/easy/English

These examples come with a task description, the circuit diagram, and the easySoft 7 project (in the EDP and LD programming languages as of this writing).

### Tutorials

For helpful videos that explain how to use specific functions, please visit the product page at http://www.eaton.eu/easy.

### Product training

The Eaton Experience Center Training (EEC) has a series of training courses available for the easyE4. For more information, as well as to download the workshop catalog, please visit:

http://www.eaton.eu/training

### Community

easyForum is an additional source of help that can be found on the Internet at:
http://www.easy-forum.net

### Cybersecurity

Eaton recommends implementing measures for protecting against cyberattacks.

→ Eaton cybersecurity
https://www.eaton.com/us/en-us/company/news-insights/cybersecurity.html

https://www.eaton.com/cybersecurity

Secure Hardening Guideline MZ049001EN

# A.6 Sample Projects

To get a quick look at the possibilities offered by the easyE4 series, please visit the corresponding product page on the Internet (the page includes a series of application examples and tutorials).

**Application examples**

Support has provided a number of applications that are available for download as ZIP files from the Software Download Center.

Download Center - Software
http://www.eaton.eu/software/Anwendungsbeispiele/easy/Deutsch
http://www.eaton.eu/software/Application Samples/easy/English

These examples come with a task description, the circuit diagram, and the easySoft 7 project (in the EDP and LD programming languages as of this writing).

**Tutorials**

For helpful videos that explain how to use specific functions, please visit the product page at http://www.eaton.eu/easy.

If you do not have an Internet connection available, you can try out one of the application examples here if you have already installed easySoft 7:

→ The application examples created by Eaton can only be transferred to the easyE4 device if a license has been added for easySoft 7.

## A.6.1 easyE4_Lauflicht_EDP.e70 application example

**Task definition**

Say that you want to use the easyE4 to switch on four lamps in sequence and then switch them off accordingly.
Starting from the first lamp all the way to the fourth lamp, and then vice versa from the fourth lamp all the way to the first lamp, and so on. The system can be switched on and off with main switch S1.

The selector switch S2 defines whether the chaser light is to be activated permanently or only at the set times (daily from 6 PM to 10 PM).

Three different speeds can be set for the chaser light:

- Switch S3 > Fast chaser light speed (0.30 sec.)
- Switch S4 > Medium chaser light speed (0.60 sec)
- Switches S3 S4 simultaneously > Slow chaser light speed (1 sec)

### Wiring topic

1. Inputs:

| | |
|---|---|
| I1 | Main switch S1 (system ON / OFF) |
| I2 | Selector switch S2 (time switch ON/ OFF |
| I3 | Switch S3 (chaser light speed) |
| I4 | Switch S4 (chaser light speed) |

2. Outputs:

| | |
|---|---|
| Q1 | Lamp H1 |
| Q2 | Lamp H2 |
| Q3 | Lamp H3 |
| Q4 | Lamp H4 |

3. Parameters:

| | |
|---|---|
| T1 | Fast pulse speed (0.30 sec) |
| T2 | Medium pulse speed (0.60 sec) |
| T3 | Slow pulse speed (1 sec) |
| C1-C4 Number of pulses | Number of pulses |
| H1 Chaser light on times | Chaser light on times |

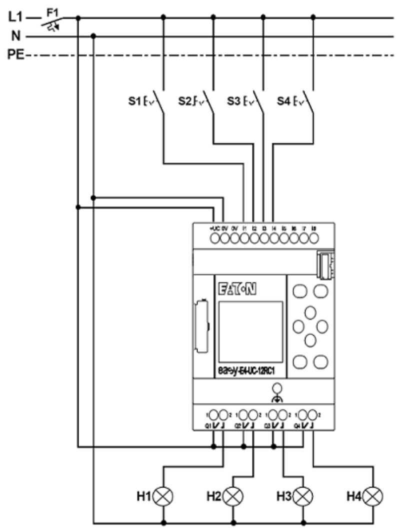



Figure 302: Circuit diagram for easyE4 chaser light

# Alphabetical index


**2**

24 V pulses ..... 364

**7**

7-day time switch ..... 218

**A**

A - Analog value comparison visualization device ..... 306-307
AC ..... 272
AC base devices ..... 41
AC device notes ..... 41
AC expansions ..... 41
Acceleration sequence
PO - pulse output ..... 366
Access control ..... 647
Accessory devices ..... 29
Activation P buttons ..... 452
Actuators ..... 345
ADD ..... 312
Adding ..... 313
Rung ..... 176
Aeration and de-aeration ..... 45
AL - Alarm function block ..... 440
Alarm ..... 451
Alarm function block ..... 440
Ambient climatic conditions ..... 45, 690, 701
Analog Signals ..... 40
Analog value comparison ..... 306
AND ..... 444
API key ..... 667
Approvals ..... 698
AR ..... 312
Arithmetic ..... 312
Assigning IP addresses ..... 100
Assigning variables, function block input ..... 195
Associated variable ..... 461
Astronomic clock ..... 272
Auxiliary relay ..... 202
AV ..... 317
Continuous mode ..... 317
One-time mode ..... 317
AV continuous mode ..... 317
AV one-time mode ..... 317
Available operands
User function blocks ..... 548
Average ..... 317

**B**

Backward jumps ..... 182
Bar graph ..... 462
BCD ..... 497
BCD-coded decimal value ..... 497
Beschreibung ..... 19
Bestimmungsgemäße Verwendung ..... 19
BIP ..... 353
Block transfer ..... 395
Blocks ..... 281, 299
Blue ..... 654
BOOL definition ..... 200
Boolean operation ..... 444
BOOT.TXT ..... 109, 113
Brake sequence ..... 368
Brand names
Product names ..... 2
Break-Down Service ..... 2
BT - Block transfer ..... 395

Bus delay ....566, 631
BV .... 444

### C

C .... 281
C - Counter relay for 800 control relays, Visualization devices .... 282
C - Counter relay for visualization devices .... 281
Cable protection .... 56
Canceling, circuit diagram input .... 178
CARD START .... 572
Carry ....283, 295, 302
CF .... 287
CH - Frequency counter .... 287
CH - High-speed counter for the control relays 800 and the visualization devices .... 294
CH - High-speed counter for the visualization devices .... 293
Changing input values, on function blocks .... 196
Channel number
  MX - data multiplexer .... 406
Characteristic curve .... 339
Characteristic map .... 339
CI .... 299
CI - Incremental Counters for the Visualization Devices .... 299
CI - Incremental counters for visualization devices .... 299
CI - Incremental Counters for Visualization Devices .... 299
Circuit diagram .... 161
  Checking .... 180
  Elements .... 163
  Saving .... 177
Circuit diagram creation, Troubleshooting .... 683
Cleaning .... 688
Clock .... 228
  HW - 7-day time switch .... 218
Coil
  Connect .... 175
  Definition .... 165
  Deleting .... 174
  Entering, modifying .... 173
  Field .... 161
  Function, overview .... 165
  Negation .... 168
  Searching .... 178
Colors in Communication view .... 616
Comment
  User function block .... 541
Commissioning .... 87
Communication
  View .... 615
Community .... 710
Company information .... 2
Comparison of analog and setpoint values .... 306
Comparison of variables and constants .... 326
Compatibility rules .... 616
Conditional jump .... 486
Connection
  Deleting .... 176
  Representation in circuit diagram display .... 162
  To the device .... 618
Connection rules for operands .... 201
Connections
  External .... 72
Constants
  Assigning, function block input .... 195
Contact
  Changing, N/O – N/C .... 172
  Connect .... 175
  Cursor buttons .... 179

Definition ... 164
Deleting ... 174
Entering, modifying ... 171
Fields ... 161
Name ... 171
Number ... 171
Searching ... 178
Contactor function ... 166
Continuous operation ... 322
Copy
Marker contents ... 397
Copy-protected ... 2
Copyright ... 2
Count direction ... 282, 294
Counter ... 238, 281, 287, 293, 299
CI - incremental counter ... 281
Counter input ... 293, 299
C - Counter relay visualization device ... 281
CF - Frequency counter ... 288
Counter relay ... 281
CP ... 326
CP - Comparators for 800 control relays/visualization devices ... 326
CP - Comparators for visualization devices ... 327
Create log files ... 480
Creating, editing an operand list ... 668
Cybersecurity ... 710
Cycle pulse
Falling edge ... 169
Rising edge ... 168
Cycle time ... 345, 359, 503, 576


**D**


D - Text display
Bar graph ... 462
Date and time display ... 468
Date and time entry ... 473
Display and input elements ... 461
Latching button ... 472
Message text ... 465
Message text selection ... 473
Rolling text ... 464
Running text ... 463
Static text ... 462
Timing relay value display ... 469
Timing relay value entry ... 473
Value display ... 461
Value entry ... 470
D - Text display (Display) ... 448, 458
Damage ... 46
Data block ... 388, 395
Data function block ... 401
Data logger ... 475
Data multiplexer ... 406
Data Types ... 200
Date and time display ... 468
Date and time entry ... 473
Date setting ... 587
DB ... 401
Q1 (Boolean function block output) ... 401
DC ... 351
Debounce ... 574
Debounce activated ... 604
Debounce deactivated ... 605
Deceleration ramp ... 367
Decimal numbers ... 497
Declarations ... 698
Default colors display ... 452
Default IP address ... 618
Default NET ID ... 618
Defining password-protected areas ... 581
Delay time ... 604

Delay time AC voltage ... 606
Delay value ... 358
DELETE
Function blocks ... 197
Deleting
Operands at function block inputs/outputs ... 196
Rung ... 176
Derivative term ... 352
Destination address ... 395
Destination range ... 389, 396
Detecting or changing bit patterns ... 444
Determining counter frequency ... 287
Device
Changing language ... 91
Device ID ... 564
Device reset ... 116
Device versions ... 22, 24
Devices shown in color ... 616
Dimensions ... 694
Directives ... 698
Display
Default colors ... 452
Elements ... 448
Display and input elements ... 461
Display elements ... 448
Display messages ... 682
Display priority ... 451
Disposal
Recycling ... 691
DIV ... 312
Dividing ... 313
DL - Data logger ... 475
Download ... 454
Download Center ... 709
DST ... 588
DWORD definition ... 200

**E**

E-mail ... 670
E1 ... 347
easyConnect ... 656
easyNET - NET - Compatibility ... 626
ecat ... 709
Ejecting the memory card ... 124
Electrical isolation ... 51
Electromagnetic compatibility ... 703
EN ... 345
EQ ... 308, 327, 388
Equal ... 306, 327
Equipment supplied ... 46
Error
Rectification, on event ... 684
Establishing an Ethernet connection ... 100
Ethernet ... 75, 568
Configuration ... 622
Ethernet network ... 96
Evaluating a falling edge ... 169
Evaluating a rising edge ... 168
Execution time for an interrupt ... 506, 519, 525

**F**

Factory communication parameters ... 618
Factory IP address setting ... 618
Factory NET ID setting ... 618
Faults ... 681
Features ... 20
Firmware update ... 118
Firmware update expansion device ... 121
Firmware version ... 118, 538
Firmware version 1.12 ... 632
Fixed IP address ... 100
Frequency change
PO - Pulse output ... 367

FT 330, 358
Function 20
Function block editor 194
Function block list 193
Function blocks
  Adding to a circuit diagram for the first time 191
  Assign operands, output 196
  Assigning operands, input 195
  Check 198
  Configuration editor 194
  Definition 163
  Delete 197
  List 193
Further reading 709

G

Gain factor 307
General 293, 299
GET 428
GO TO other rungs 177
Gray 654
Greater than 306, 312, 327
Green user function block 541
GT 307, 327, 428
GT - "GET" Network Visualization device, 800 control relay 428
Guest logon 654

H

Hardware counter 293
Hardware inputs 293
Hardware output 345
Hazards
  Device-specific 36
High-speed counter 293
High-speed counter functions 287
High-Speed Counters 299
HW - 7-day time switch 218
HY 228
Hysteresis 306

I

IC - 506
IE 519
Impulse relay 166
Incremental Encoder 299
Initial commissioning 87
Initialization mode 398
Input cables
  Length 39
Input delay 574
Installation 43
Installation position
  SD card 44
  Selection 44
Integer value 497
Integral term 352
Interface
  User function block 538
Interfaces
  Ethernet 75
Interrupt 519, 525
Interrupt function block 519, 525
  Counter-controlled 506
  Time-controlled 525
Interrupt load 512, 529
Inversion
  Contactor function 168
Inverting
  Contact 172
IOX 656
IP address, fixed 100

IP addresses 100
IT - Interrupt function block 525

J

JC - Conditional jump 486
Jog mode 366
JSON API 653, 657
Jump 486
Jumps 181

K

K
MX - data multiplexer 406
Key to part numbers 703
Know-how protection
User function block 539
KP 358

L

Label 30
Language 454
Language changeover 453
Languages 453
Changing on the device 91
Latching button 472
Latching relay 167
LB - Jump label 491
LED
Checking of the NET 685
LED ETHERNET 89, 613
LED POW/RUN 88-89, 613
Less than 306, 327
LI 354
Lifespan
Backlight 87
LIFO 423
Limits of PW pulse width modulation function block 347
Load torque
PO - Pulse output 366
Loading circle 657
Loading programs onto multiple NET stations 629
Location of use 44
Login guest 654
Lower and upper limits 384
LS 330
LT 327
A - Analog value comparator for visualization device 306

M

Maintenance 687-688
Manipulated variable 352
Manipulated variable SV 345-346
Mark-to-space ratio 293, 300
Marker 493
Definition 202
Initialize MB, MW + MD 398
Marker range accessible with offset 395
Retention 213
Value range 202
Marker range 508, 520, 526
Markers
Assigning, function block input 195
Copy MB, MW + MD 397
Memory card 124
Memory display, circuit diagram 162
Message
PROG INVALID 683
Message text 465
Message text selection 473
microSD 124

Minimum on duration ... 345
Minimum on duration = Minimum off duration ... 347
Minimum period duration ... 347
Missing parts ... 46
MM- Min-/Max function ... 335
Modbus TCP ... 632
Modifying
    Connections ... 175
    Contacts and coils ... 170
Motion control ... 364
Mounting ... 47
MR - Masterreset ... 493
MUL ... 312
Multiplying ... 313
MX - Data multiplexer ... 406

**N**

N/C contact ... 164
    Inverting ... 172
N/O contact ... 164
    Inverting ... 172
Name
    User function block ... 536
Nameplate ... 30
NC - Numerical converter ... 497
Negation, coil ... 168
NET ... 566, 625
    -ID ... 184
    Configuration ... 622
    Operands ... 183
NET-GROUP ... 566, 630
NET-ID ... 566, 630
NET - Definition ... 625
NET settings ... 629
NET station heartbeat ... 628
Network function blocks ... 432
Network Function Blocks ... 428
Network operation ... 96
NO ... 388, 395
Normal mode ... 372
Normalized variables of the PID controller ... 351
NOT ... 445
Number of pulses
    PO - Pulse output ... 366
Numeric formats ... 202
Numerical converter ... 497
Numerical converter mode ... 497

**O**

Offset ... 306, 395
On the CH High-Speed Counter ... 293
On the CI Incremental Encoder ... 299
ONLINE ... 616
Online Catalog ... 709
Operand connection rules ... 201
Operand overview ... 200
Operand table ... 204
Operands ... 200
    Assigning ... 195
    Assigning, function block output ... 196
    Deleting at function block inputs/outputs ... 196
Operating frequency ... 365-366
Operating hours counter ... 238
Operating hours counter for visualization device ... 238
Operating mode ... 131, 330, 353, 358, 497, 503
    Timing relay ... 248
Operating sequence
    Pulse output ... 366
Operating states ... 602
Operating system V1.00 update ... 118
Operation
    Proper ... 35

Original Operating Instructions .................... 2
OT .................................................. 238
OT - Operating hours counter for visualization device .................................... 238
Overview of operands .......................... 202

## P

P buttons ..................................... 179, 448
Package contents ............................... 46
Parameter setting .............................. 450
Parameters
Enabling/disabling access .................... 193
Part number ..................................... 28
Password
Assigning ....................................... 582
Changing ........................................ 583
Enabling ........................................ 583
Forgotten ....................................... 583
Performance map ............................... 339
Period duration ................................. 345
Period duration to minimum ON duration ratio ... 348
PID controller .................................. 351
Operating mode ................................ 351
PID Controller
Scan time ....................................... 351
Plausibility check ............................... 551
PO
Jog mode ....................................... 374
Normal mode .................................. 372
Pulse output .................................... 364
PO - Pulse output ............................... 364
Ports ............................................. 72
PreventInputScroll ............................. 669
Product information ............................ 709
Product training ................................ 709
Program cycle time ............................. 354
Program name .................................. 577
Programming method
User function block ............................ 536
Proportional actuators ......................... 345
Proportional Factor ............................ 358
Proportional gain ............................... 358
Proportional gain Kp ........................... 353
Proportional term ............................... 352
PT ................................................ 432
PT1 signal smoothing filter ................... 358
Pulse duration .................................. 345
Pulse output .................................... 345
PO - Pulse output ............................. 364
Pulse sequence ................................. 345
Pulse Shape of Counter Signals ............... 299
Pulse width ..................................... 345
Pulse width modulation ........................ 345
PUT .............................................. 432
PW ............................................... 345

## Q

Q01/Q02 ......................................... 345
Q1 (boolean function block output) ........... 238
PT - "PUT" Network ............................ 432
SC - Synchronizing time via network ......... 436
Q1 (Boolean function block output) ...........306, 428

## R

Rate time Tv .................................... 353
RC - Real-time clock ........................... 242
RE ................................................ 410
Read log files ................................... 481
Reading values from the network ............. 428
Real-time clock ................................. 242
Synchronizing via network .................... 436
Recipe ........................................... 410

Recovery time ..... 358
Reference data block ..... 388
Reference value table ..... 339
Relay
  Coil function ..... 165
Relays
  Definition ..... 163
Remote operation ..... 97
Remote RUN ..... 567, 630
Removing password protection ..... 583
Repairs ..... 689
Required function block memory ..... 705
Required memory for function blocks ..... 705
Reset ..... 116, 353, 358, 497
  VC - value limitation ..... 384
Reset time Tn ..... 353
Reset, coil function ..... 167
Resolution ..... 345
Restart ..... 482
Retention ..... 401, 539, 578
Retention with &function relays ..... 281
Retention with Function &Relays ..... 299
Retention with function relays ..... 287
Retentive markers ..... 213
Rolling text ..... 464
Rolling time ..... 451
RUN ..... 131
RUN START ..... 572
Rung
  Adding/Deleting ..... 176
  Change ..... 177
  Deleting ..... 176
Rungs ..... 162
Running text ..... 463


**S**


Safety ..... 33
Saving, Circuit diagram ..... 177
SC ..... 436
Scan time ..... 354
SD card ..... 73
Search for devices ..... 102
Searching, contacts and coils ..... 178
Sending e-mail ..... 440
Separate web client operands ..... 661
Series production ..... 117
Service ..... 31
Set cycle time ..... 503
Set, coil function ..... 167
Setpoint ..... 351
Setting a boot program ..... 109, 113
Setting a starting program ..... 109, 113
Setting daylight saving time ..... 590-591
Setting the web server login text ..... 649
Setting up a web server ..... 646
Setting up users ..... 648
Settings web client ..... 665
SH ..... 384
Shift register ..... 415
Shifting bits forward/backward ..... 415
Shifting double words forward/backward ..... 415
Showing easyE4 operands in the web client ..... 661
Signal smoothing ..... 358
SL ..... 384
SNTP synchronization ..... 590-591
Source address ..... 395
Source range ..... 390, 396
Special considerations for AC devices ..... 41
Splash screen ..... 126
Splash screen display duration ..... 565
SR ..... 415

ST ... 503
Standards ... 698
Start frequency ... 366
Start ramp ... 367
Starting program ... 124
Starting the web client ... 652
Startup mode ... 571
Static text ... 462
Status display ... 94
Std. pack ... 46
Step response ... 358
Stepper motor ... 364
STOP ... 131
Storage ... 690
SUB ... 312
Subtracting ... 313
Support ... 31
Switching contact –> please refer to Contact ... 164
Switching duration
- T - Timing relay logic relay ... 246

Synchronize date via NET ... 436
Synchronizing NET stations ... 436
System requirements ... 79
System settings ... 646


**T**


T-Timing relay ... 246
- Flashing ... 246
- On-delayed ... 246
- Retention ... 259
- Single pulse switching ... 248
- Stop ... 246
- Switch off time ... 246
- Timing and counter relay example ... 557
- Trigger input ... 246

T – Timing relay logic relay ... 249
Tab
- Webserver ... 646

Table function ... 423
TB ... 423
TC - Three step controller ... 379
Technical data ... 699
Temperature measuring ... 67
Terminal capacity ... 54
Testing, circuits via the P buttons ... 179
Text display ... 448, 450
Text display editor ... 458
- Static text ... 462

TG ... 358
Threshold value switch ... 306
Time ... 218, 228, 436
- SC - Synchronizing time via network ... 436

Time reference value
- T - Time relay/logic relay ... 248

Time response ... 603
Time response;Base devices ... 604
Time setting ... 587
Time value ... 249
- T-Timing relay logic relay ... 249

Timer ... 218
- HY - Year time switch ... 228

Timing and counter relay example ... 557
Timing characteristics;Expansion devices ... 608
Timing relay ... 246
- Operating mode ... 248

Timing relay value display ... 469
Timing relay value entry ... 473
TN ... 351
Transit damage ... 46
Transport ... 690
Trigger input (trigger coil)
- "PT - PUT" Network ... 432

Troubleshooting 681
- During circuit diagram creation 683

Two's complement 444

## U

UF - User function block 533
UNP 353
Up/down counters 281
Update 389, 569
Update web client 657
Updating data 657
Updating firmware 117-118
Upper limit 330
Upper threshold value 238
User function block 533
- Calling in the main program 543
- Configuring 536
- Creating 534
- Exporting 550
- Importing 552
- In an ST main program 547
- Printing 556
- Programming 541
- Replacing 553
- Saving 548

User function block green 544
User function block storage location 548
User function block yellow 544
User function blocks
- Comparing 554

Using the web client 654

## V

Value display 461
Value entry 470
Value limitation 384
Value range 313
Value range, marker 202
Value scaling 330
VC - Value limitation 384
version
- User function block 536

Versions 22, 28
View
- Communication 615

Visualization 680
Visualization device 238, 281

## W

Web client
- Login guest 654
- Operand list 668
- Separate operands 661
- Settings 665
- Update the operands 657
- Updating data 657

Web client cycle time 669
Web operand list 668
Web server configuration 646
Window discriminator 384
Wirin
- Grid 161

Wiring arrow 175
WORD definition 200
WT - Weekly timer 268

## X

XOR 445

## Y

Year time switch 228
Yellow user function block 541

YT - Year time switch .................................. 261

## Z

Zoom function .......................................... 658
Zoom increment ...................................... 658

## List of Figures

Figure 1: Device model with EASY-E4-...-12...C1(P) display and button controls or with EASY-E4-...-12...CX1(P) LED display for diagnostics 22
Figure 2: Device model with 4SU 24
Figure 3: Device models in 2SU 24
Figure 4: Mounting distance min. 3 cm 48
Figure 5: Assembling a base device with expansions 49
Figure 6: Installation on IEC/EN 60715 mounting rail 50
Figure 7: Inserting a fastening bracket. 52
Figure 8: Screw mounting configuration for a device 52
Figure 9: Remove adjacent connectors 53
Figure 10: Dismantling 53
Figure 11: Connecting the power supply for base devices 56
Figure 12: Connecting the power supply for expansions 56
Figure 13: Connecting the digital inputs on base devices 58
Figure 14: Connecting the digital inputs on expansions 58
Figure 15: Connect digital counter inputs 60
Figure 16: Connecting the analog inputs on base devices 61
Figure 17: Connecting relay outputs 62
Figure 18: Connecting base device transistor outputs 63
Figure 19: Connecting expansion device transistor outputs 63
Figure 20: Inductive load with suppressor circuit 64
Figure 21: Device parameters tab, using the EASY-E4-DC-6AE1 as an example 65
Figure 22: Connecting analog inputs EASY-E4-DC-6AE1(P) 65
Figure 23: Connecting analog outputs EASY-E4-DC-6AE1(P) 66
Figure 24: Connecting analog inputs EASY-E4-DC-4PE1(P) 67
Figure 25: Expansion parameter tab, using the EASY-E4-DC-4PE1 as an example 68
Figure 26: Slot for microSD 72
Figure 27: Ethernet port on base device 72
Figure 28: Inserting a memory card 73
Figure 29: Removing the memory card 74

Figure 30: Removing the memory card 74
Figure 31: RJ-45 socket, 8-pole 75
Figure 32: Connecting the Ethernet cable 76
Figure 33: Removing the Ethernet cable 77
Figure 34: Removing the Ethernet cable 77
Figure 35: license product certificate 79
Figure 36: Input screen for the license product certificate No. 80
Figure 37: License dialog box 81
Figure 38: Options in ? menu 82
Figure 39: Step 1 83
Figure 40: Step 2 License agreement 84
Figure 41: Step 3 License key 84
Figure 42: Step 4 Destination folder 84
Figure 43: Step 4.1 Changing the destination folder 85
Figure 44: Step 4.2 Creating your own destination folder 85
Figure 45: Step 6 Starting the installation 85
Figure 46: Step 7 Progress display 86
Figure 47: Step 8 Finishing 86
Figure 48: easySoft 7 icon depending on the screen resolution 86
Figure 49: LED status indication 88
Figure 50: Example of status display on display 90
Figure 51: Main menu in English 91
Figure 52: Menu path in English 91
Figure 53: Start displays for easyE4 base device in English 94
Figure 54: Example of status display on display 95
Figure 55: Startup procedure with device initialization 99
Figure 56: Establish Ethernet connection 102
Figure 57: Search for devices with an IP address 103
Figure 58: Saving the found device's IP profile 103
Figure 59: Selecting the easyE4 device's IP address 104
Figure 60: Connection to the easyE4 device established and program transferred 104
Figure 61: Offline dialog box for memory card 107
Figure 62: microSD memory card drive with PROGRAM folder contains 109

BOOT.TXT and compiled test.prg program

Figure 63: Offline dialog box for memory card 111

Figure 64: microSD memory card drive with PROGRAM folder contains BOOT.TXT and compiled test.prg program 113

Figure 65: microSD memory card content when using bootloader version 1.01 118

Figure 66: boot.bmp 126

Figure 67: Storing the boot.bmp file 126

Figure 68: Display and keypad 127

Figure 69: Example of status display on display 127

Figure 70: Empty circuit diagram 145

Figure 71: Fields in the circuit diagram 146

Figure 72: Lighting control circuit 147

Figure 73: Circuit diagram with inputs I01, I02 and output Q1 147

Figure 74: Completed circuit diagram 149

Figure 75: SAVE menu option in the status line 149

Figure 76: Power flow display 1 150

Figure 77: Power flow display 2 150

Figure 78: Display with zoom, power flow 151

Figure 79: Display with zoom, power flow interrupted 151

Figure 80: Sample program open 154

Figure 81: Card setup dialog box 155

Figure 82: File selection dialog box 156

Figure 83: The program was transferred to the memory card. 157

Figure 84: Ethernet connection on PC 159

Figure 85: Circuit diagram display 161

Figure 86: Contactor function signal diagram 166

Figure 87: Impulse relay signal diagram 166

Figure 88: Set and Reset signal diagram 167

Figure 89: Simultaneous triggering of Q 01 167

Figure 90: Inverse contactor function signal diagram 168

Figure 91: Signal diagram of cycle pulse with rising edge 168

Figure 92: Signal diagram of cycle pulse with negative edge 169

Figure 93: Circuit diagram with inputs 170

Figure 94: Contact legend 171

Figure 95: Change contact I 03 from N/O to N/C ....172
Figure 96: Relay coil "Output Q" .... 173
Figure 97: Relay coil for timing relay function block with control coil .... 173
Figure 98: Relay coil of a NET station ....173
Figure 99: Circuit diagram with five contacts, invalid ....175
Figure 100: Circuit diagram with M marker relay ....175
Figure 101: Inserting a new rung ....176
Figure 102: The cursor buttons are wired in the circuit diagram as contacts P 01 to P 04. ....179
Figure 103: Switch Q1 via I1, I2, Í, or Ú ....179
Figure 104: I5 switches to cursor buttons. ....179
Figure 105: Paralleling link ....180
Figure 106: Power flow display .... 180
Figure 107: 1 slave ....186
Figure 108: 2 slave ....186
Figure 109: Explanation of the function block list .... 194
Figure 110: Manufacturer function block display in the function block editor .. 194
Figure 111: ....201
Figure 112: Signal diagram ....222
Figure 113: Tab with parameters in the Programming view ....222
Figure 114: Signal diagram ....223
Figure 115: Tab with parameters in the Programming view ....223
Figure 116: Signal diagram ....224
Figure 117: Tab with parameters in the Programming view ....224
Figure 118: Signal diagram ....225
Figure 119: Tab with parameters in the Programming view Settings Time overlap ....225
Figure 120: Signal diagram ....226
Figure 121: Tab with parameters in the Programming view 24 hours setting ..226
Figure 122: Tab with parameters in the Programming view ....227
Figure 123: Year time switch parameters tab with and example in which a year range is being selected ....232
Figure 124: Entry screen in the programming software ....234
Figure 125: Entry screen in the programming software ....234

Figure 126: Entry screen in the programming software 235
Figure 127: Entry screen in the programming software 235
Figure 128: Entry screen in the programming software 236
Figure 129: Entry screen in the programming software 236
Figure 130: Entry screen in the programming software 237
Figure 131: Signal diagram of timing relay, on-delayed (with and without random switching) 251
Figure 132: Signal diagram of timing relay, on-delayed (with and without random switching) 252
Figure 133: Signal diagram of timing relay, off-delayed 253
Figure 134: Signal diagram of timing relay, off-delayed 254
Figure 135: Operational diagrams timing relay, on and off delayed 1 255
Figure 136: Signal diagram timing relay, single pulse 1 256
Figure 137: Signal diagram timing relay, single pulse 2 257
Figure 138: Signal diagram timing relay, single pulse 1 257
Figure 139: Wiring the function block coils 259
Figure 140: Wiring of the function block contact 259
Figure 141: Year time switch (new) parameters tab for YT function block with example showing all four modes 264
Figure 142: Entry screen in the programming software 265
Figure 143: Entry screen in the programming software 265
Figure 144: Entry screen in the programming software 266
Figure 145: Entry screen in the programming software 266
Figure 146: Entry screen in the programming software 266
Figure 147: Weekly timer (new) parameters tab with an example 270
Figure 148: Sunrise and sunset in Bonn 276
Figure 149: Sunrise and sunset in Drevja 277
Figure 150: Offset; O1=-2; O2=2; Q1=1 will switch on 2 hours before sunrise and off 2 hours after sunset 277
Figure 151: No offset; O1=0; O2=0; Q1=1 between sunrise and sunset 278
Figure 152: Offset; O1=1; O2= -1; Q1 will switch on 1 hour after sunrise and off 1 hour before sunset 278
Figure 153: Offset; O1=-2; O2=2; Q1=1 will switch on 2 hours before sunrise and off 2 hours after sunset 278
Figure 154: Offset; O1=-2; O2=-2; Q1=1 will switch on 2 hours before sunrise 279

and off 2 hours before sunset
Figure 155: Q1 does not switch off during the summer months 279
Figure 156: Q1 does not switch on during the winter months 280
Figure 157: Signal diagram of counter relay 285
Figure 158: Signal diagram of frequency counter 292
Figure 159: Signal diagram High-speed counter 297
Figure 160: CI function block counting up; QV=QV+4 300
Figure 161: CI function block counting down; QV=QV-4 300
Figure 162: Signal diagram High-speed incremental value counter 304
Figure 163: Analog comparator signal diagram 310
Figure 164: Parameters on the display 311
Figure 165: Wiring the contacts 315
Figure 166: Parameters on the device display 315
Figure 167: Sample curve for hourly temperature measurement, over 7 days 323
Figure 168: Wiring the contacts 328
Figure 169: Parameters on the display 329
Figure 170: Figure: Scaling the input values - reducing 330
Figure 171: Scaling the input values - increasing 330
Figure 172: Mathematical interrelationship 331
Figure 173: Example of a characteristic curve for the PM function block 343
Figure 174: Wiring the function block coils 356
Figure 175: Wiring of the function block contact 356
Figure 176: Parameters on the device display 357
Figure 177: Response of the FT function block 359
Figure 178: Wiring the function block coils 362
Figure 179: Parameters shown on display 362
Figure 180: Typical pulse profile for a stepper motor in normal mode 366
Figure 181: Signal diagram for PO pulse output with specified number of pulses for I1 - possible normal mode phases 373
Figure 182: Signal diagram for jog mode with specified number of steps P1 375
Figure 183: Signal diagram for jog mode with specified jog frequency, P1 after deceleration phase reached 376
Figure 184: Signal diagram for jog mode with specified jog frequency, P1 not reached after deceleration phase 377

Figure 185: Three-step controller schematic diagram .......... 379
Figure 186: Timing diagram for three-step controller .......... 379
Figure 187: Signal diagram for three-step controller .......... 382
Figure 188: Figure: Restriction of the input values to the specified limits. .......... 384
Figure 189: Wiring the enable coil .......... 393
Figure 190: Wiring the contacts .......... 393
Figure 191: Parameters on the display .......... 393
Figure 192: Parameters on the display .......... 399
Figure 193: Wiring the trigger coil .......... 400
Figure 194: Wiring the contacts .......... 400
Figure 195: Signal diagram of data function block .......... 404
Figure 196: Wiring the trigger coil .......... 404
Figure 197: Wiring of the function block contact .......... 404
Figure 198: Parameters on the display .......... 404
Figure 199: Shift register SR..: Forwards operation in BIT operating mode .......... 416
Figure 200: Shift register SR..: Backwards operation in DW operating mode .......... 417
Figure 201: Circuit diagram with EDP programming language for user example 2 .......... 421
Figure 202: Parameters on the device display .......... 421
Figure 203: Text display parameters tab for text display function block in the Programming view .......... 451
Figure 204: Default colors tab for text display .......... 453
Figure 205: Text display function block, language tab .......... 453
Figure 206: Signal diagram Text display .......... 455
Figure 207: Signal diagram for text display with text function blocks with an identical priority of 3 .......... 455
Figure 208: Text display editor with static text in the first line .......... 459
Figure 209: Character table Special characters .......... 460
Figure 210: Example showing an exact value message text .......... 465
Figure 211: Value range message text example .......... 467
Figure 212: Workspace with function block and device button .......... 480
Figure 213: Data logger tab with set parameters for the programming view .......... 481
Figure 214: Card content after logging has ended .......... 483
Figure 215: Activated function block in the function block status display .......... 487

Figure 216: Wiring the function block coils ........ 495
Figure 217: Wiring of the function block contact ........ 495
Figure 218: Parameters on the device display ........ 496
Figure 219: Wiring the function block coils ........ 502
Figure 220: Setting of the parameters ........ 502
Figure 221: Input and output states being passed between the main program and interrupt program ........ 507
Figure 222: easySoft 7 Main program Pulse counter with external direction ........ 514
Figure 223: easySoft 7 Interrupt program Pulse counter with external direction ........ 514
Figure 224: easySoft 7 Main program, two counter inputs ........ 515
Figure 225: easySoft 7 Interrupt program, two counter inputs ........ 515
Figure 226: easySoft 7 Main program Incremental counter ........ 516
Figure 227: easySoft 7 Interrupt program Incremental counter ........ 517
Figure 228: easySoft 7 Main program Frequency measurement ........ 518
Figure 229: easySoft 7 Interrupt program Frequency measurement ........ 518
Figure 230: Input and output states being passed between the main program and interrupt program ........ 520
Figure 231: easySoft 7 Main program Slope ........ 524
Figure 232: easySoft 7 Interrupt program Slope ........ 524
Figure 233: Input and output states being passed between the main program and interrupt program ........ 526
Figure 234: easySoft 7 Main program Time-controlled ........ 532
Figure 235: easySoft 7 Interrupt program Time-controlled ........ 532
Figure 236: Create user function block ........ 535
Figure 237: Configure user function block ........ 538
Figure 238: Screenshot of Retention section that can be found in the System settings tab in the Project view (including sample values) ........ 539
Figure 239: Programming view for UF Blinker1 user function block ........ 541
Figure 240: User function block comments being shown in the tab ........ 543
Figure 241: UF Blinker1 user function block used in the main program ........ 544
Figure 242: Inputs/outputs wiring ........ 545
Figure 243: Contact tab ........ 546
Figure 244: Analog contact tab ........ 546
Figure 245: Coil tab ........ 546

Figure 246: Analog coil tab ........ 547
Figure 247: Delete user function blocks dialog box ........ 554
Figure 248: Location of user function block user for comparison dialog box ... 554
Figure 249: UF user function block ........ 555
Figure 250: Import user function block ........ 555
Figure 251: Hardwiring with relays ........ 557
Figure 252: Wiring with EASY-E4-UC-..., for example ........ 557
Figure 253: Wiring of counter and timing relay ........ 557
Figure 254: Enter parameter C01 ........ 558
Figure 255: Enter ParameterT01 ........ 558
Figure 256: Testing the circuit diagram ........ 559
Figure 257: Testing the circuit diagram +10 ........ 559
Figure 258: Doubling the flashing frequency ........ 559
Figure 259: Screenshot of Retention section that can be found in the System settings tab in the Project view (including sample values) ........ 578
Figure 260: Password assignment ........ 582
Figure 261: Password submenu ........ 583
Figure 262: Submenu for changing the password ........ 583
Figure 263: Assigning a program password ........ 584
Figure 264: How the EDP evaluates circuit diagrams and function blocks ..... 596
Figure 265: easyE4 input assigned a switch ........ 603
Figure 266: Delay times for evaluating an DC input signal and an activated debounce ........ 604
Figure 267: Switching behavior with debounce deactivated ........ 605
Figure 268: Delay times for evaluating an AC input signal without I-DEBOUNCE and for an activated I-DEBOUNCE ........ 606
Figure 269: Switching behavior of AC input signal with DEBOUNCE activated 606
Figure 270: Switching behavior of AC input signal with DEBOUNCE deactivated ........ 607
Figure 271: ONLINE Project view with devices colored differently based on their compatibility ........ 616
Figure 272: Selection of NET station ........ 622
Figure 273: NET configuration with project and program ........ 623
Figure 274: NET diagram ........ 625
Figure 275: NET-ID dialog box used to assign a NET ID when adding a new 629

base device
Figure 276: NET tab for the selected base device in the NET group 630
Figure 277: Project view Webserver tab 646
Figure 278: Web server passwords and user names dialog box 649
Figure 279: Web client, running 653
Figure 280: Web client login dialog box 654
Figure 281: Device display 658
Figure 282: Operands 659
Figure 283: NET operands 660
Figure 284: Separate operands 662
Figure 285: Diagnostics 664
Figure 286: Web client - General settings 665
Figure 287: Web client - Network settings 666
Figure 288: Web client - E-mail settings 667
Figure 289: API key 668
Figure 290: Web Client 668
Figure 291: Register e-mail 671
Figure 292: Email tab with the settings from the example 674
Figure 293: Ethernet tab with settings from the example 675
Figure 294: Email example when the operating mode changes 676
Figure 295: Alarm function block tab with parameters from the example and FUP program with alarm function block and P button P01 678
Figure 296: Example email when triggered by alarm function block AL01 679
Figure 297: Visualization on operating terminal 680
Figure 298: Dimensions in mm (Inch) base devices 694
Figure 299: Dimensions in mm (Inch) extensions 4SU 695
Figure 300: Dimensions in mm (Inch) extensions 2SU 696
Figure 301: Dimensions in mm (Inch) 696
Figure 302: Circuit diagram for easyE4 chaser light 712

# Glossary

**Client**
The term "client" refers to an application that requests specific services from a server.

*****

***.bmp**
Pixel-based file format for two-dimensional raster graphics

***.csv**
Comma-Separated Values (Character-Separated Values) Data format for text

***.DLL**
Dynamic link library

***.itf**
Internal Tag Import Format

***.jpg**
Pixel-based file format for the JPEG (Joint Photographics Expert Group) image file format The JPEG format does not support transparency

***.png**
PNG (Portable Network Graphics) image file format for graphics and video software, The PNG file format supports transparency with its alpha channel

***.tiff**
Vector-based image file format for graphics and video software, The TIFF format supports transparency, as well as images using 8-bit channels (grayscale, RGB, CMYK, etc.)

***.uf7**
User function block file format

***.zip**
ZIP file format used to compress and archive files

**A**

**Address reference**
The term "address reference" refers to the data packet's start address.

**Alpha channel**
Transparency information for PNG images Used to specify the degree of transparency for each pixel

**Application**
Short for "application software," a computer program that performs a function useful to the user.

**B**

**B**
Build

**Bitmap**
Image file in the BMP raster graphics image file format.

**Boot**
Booting up, starting (up) - automatic process that takes place after the device is switched on, and in which a simple program in ROM memory starts a more complex program.

**C**

**CBA**
Communication Board Adapter

**CEST**
Central European Summer Time

**CIDR**
ClasslessInterDomainRouting

**CIS**
Card Information Structure

**Command sequence**
: Path information List of the commands that the device operator must tap in succession in order to get to the location described; for example: Start\Project Overview\\Variables folder.

**Communication**
: The transfer of data between the panel and the PLC, controller, or peripheral connected to it.

**CRC**
: Operating system checksum

---

## D

**Data Set Ready**
: The transmitter is ready to send data.

**Data Terminal Ready**
: The receiver is ready to receive data.

**DCF77**
: German long-wave time signal, Frankfurt frequency 77

**DHCP**
: Dynamic Host Configuration Protocol

**DHCP (used to obtain an IP address automatically)**
: You can enable this setting if you do not want to configure every single individual computer within a network, provided there is a DHCP server on the network. When this setting is enabled, the computer will get information such as an IP address, subnet mask, gateway, and DNS from the DHCP server. In most cases, the router used on a network will also feature a DHCP server.

**Direction URL**
: Uniform Resource Locator

**DNS**
: Domain Name System

**DNS (Domain Name Server)**
: When you enter an address such as www.intel.com into a browser or FTP client, your computer will first need to ask a server for the IP address behind the name in order to actually be able to reach the address. The server that provides this information is known as a "domain name server." Every single Internet provider provides this service, and most providers have a secondary DNS in case their primary DNS fails. and most providers have a secondary DNS in case their primary DNS fails.DNS records are the IP addresses for these servers.

**DSR**
: Data Set Ready

**DST**
: Daylight Saving Time

**DTR**
: Data Terminal Ready

---

## E

**easyConnect**
: Data connection for easyE4 with each other via bus connector plug EASY-E4-CONNECT1

**EDP**
: Easy Device Programming - programming method

---

## F

**FAT**
: File Allocation Table

**FB**
: function block

**FBD**
: Function block diagram - programming method

**File Allocation Table**
: FATs are used to define filesystems.

**Firewall**

Firewalls are used to prevent outside attempts to access IP addresses on a private network. In other words, they are used to protect internal data. When configured correctly, they can also be used to set up rules or lists that prevent specific URLs from being requested, e.g., when they are in violation of company policy. A firewall's main task is to use the information in a packet (the source and destination IP addresses, as well as the port) to decide whether the packet should be rejected or allowed to pass. This also prevents packets not meant for the network from subjecting the network to an unnecessary load, as well as packets meant for the private network from reaching the Internet.

**FTP**

File Transfer Protocol

## G

**Gateway**

Gateway When two computers on different networks want to communicate with each other, the networks need to be connected with a router. For example, surfing on the Internet requires for packets to be routed from the Internet to the network and vice versa. By using a subnet mask, a computer can know whether the receiver can be found on its network or whether it is located outside of it. If it is located outside the network, the computer will send a packet to the router specified with the gateway IP address.

## H

**Human-machine interface**

Human Machine Interface

## I

**IL**

Installation instructions

**IP Address**

IP addresses are 32 bits (4 bytes) long and are used to uniquely identify networks, subnetworks, and individual computers that work with the TCP/IP protocol. A distinction is drawn between private address spaces for local networks (intranet) and public addresses (Internet).

**IR**

Infrared

## L

**LAN**

Local Area Network

**LD**

Ladder diagram - programming method

**LSB**

Last Significant Bit

## M

**MDI**

Multi Document Interface

**Menu bar**

Menu ribbon that can be expanded and collapsed and that provides the various available commands

**MESZ**

Central European Summer Time

**MN**

Manual - Operation manual

**Modulo**

From the Latin modulo, i.e., "a small measure"

## O

**Object**

Static or dynamic element used for engineering purposes. Static objects are located in the view's background and do not change at runtime. In contrast, dynamic objects are located in the view's foreground, and their appearance can change as a result of data changes.

**Operating system**

A group of programs that control and manage the processes in a computer and its connected devices.

**OS**

Operation System

## P

**PCMCIA**

Personal Computer Memory Card International Association (PCMCIA)

**Peer to Peer (P2P)**

Peer-to-peer is a term used for computers that are connected to each other in an architecture in which both computers can assume the role of server and client.

**PELV (protective extra low voltage)**

Protective low voltage that provides protection against electric shock. It refers to how machines are electrically installed – one side of the circuit or a point on the PELV circuit's power source needs to be connected to the protective bonding circuit.

**Personal computer**

A personal computer is made up of a central processing unit, RAM, external data storage devices, an operating system, and application programs, and is connected to peripheral devices (monitor, printer). PCs can be stationary or portable.

**Personal Computer Memory Card International Association (PCMCIA)**

PCMCIA Card, also known as PC Card, is a standard for the expansion cards for portable computers used in Eaton touch panels. PCMCIA cards are energy-efficient and support hot plugging, i.e., they can be replaced during ongoing operation. Plug and play is supported thanks to the fact that all the properties required in order to automatically configure the driver are stored on the card's CIS.

**PID controller**

Proportional–Integral–Derivative Controller

**PLC**

Programmable logic controller The controller or peripheral that is connected to the HMI.

**PLC(S)**

Programmable logic controller The controller or peripheral that is connected to the HMI.

**Polling**

Cyclical reading of the PLC's addressed variables

**Port**

Ports can be seen as virtual mailboxes for data packets. A computer can communicate with other computers on 65536 different ports.

**Projected capacitive touch**

A display designed for high precision, user friendliness, and durability. It is designed to bring the controls that have now become prevalent in consumer electronics to machines, with advantages such as a gesture-based user interface, two-finger multi-touch depending on the application software being used, intuitive operation that enables operators to start working right away, and the fact that no calibration is required

**PU**

Polyurethane

## R

**Registers**
Subpages in a dialog box or object

**Retention**
Refers to the ability of operands to retain their value (memory contents) in the event of a loss of voltage

**ROM (read-only memory)**
Non-volatile read-only memory

**Router**
Routers are devices used to forward ("route") requests from a network to the Internet (or to another network). Routers provide a measure of security for private networks, as nodes outside of the network will be unable to determine which specific computer requested the data. This is because all the computers on the private network will appear under the same IP address on the Internet.

**RTC**
Real Time Clock

**RxD**
Receive cable for received data

## S

**SD card**
Secure Digital memory cards are non-volatile, rewritable flash data storage devices that are used with Eaton and are commonly referred to as microSD cards. Data written to these cards is stored in a non-volatile manner that does not require any additional (secondary) power.

**SELV (safety extra low voltage)**
Circuit in which no dangerous voltage occurs even in the event of a single fault.

**Server:**
The term "server" is usually used to refer to computers that provide services on a network. Admittedly, however, this definition is not very precise. More specifically, servers are applications on a computer that are responsible for providing or processing data. In fact, every computer can provide such services. Servers are not active in and of themselves. They wait until they are addressed by a client, after which they perform the corresponding tasks. Each server application provides its service on the network via a specific port.

**Slot**
Refers to a slot for a memory card

**SNTP**
Simple Network Time Protocol

**SSL/TLS**
Secure Sockets Layer/ Transport Layer Security

**ST**
Structured Text - programming method

**Stroke**
A hub is a device used to connect various network devices together. Hubs broadcast all data to all connected devices (devices connected with a patch cable).

**Subnet mask**
A subnet mask is an IP address "filter." It has the same syntax as an IP address. This mask defines which computers can transfer data between themselves within a network. This also means that subnet masks define the maximum size of the corresponding subnetworks.

**Switch**
Switches are networking devices that are more advanced than hubs. One of the main features that sets them apart from the latter is the fact that they are more "intelligent" and forward data packets much more efficiently by sending them only to the devices that need to receive them. Multiple data packets can pass through a

switch at the same time. Among other things, this means that switches have a significantly higher total bandwidth (throughput) than hubs. Moreover, switches learn which stations are connected to which ports, meaning that additional data transfers will not result in any ports being subjected to unnecessary loads, i.e., that data will only be forwarded to the port connected to the intended destination. With the exception of their higher price, switches are superior to hubs in every way.

**System character set**
: Font type and size used to output system messages.

## T

**TE**
: Space units

**Toolbar**
: The toolbar provides all important functions so that they can be accessed directly. All the buttons in a toolbar can also be found as menu options in the menu.

**Transfer parameters**
: Baud rate, data bit, start bit, stop bit, and parity

**TxD**
: Transmit cable for transmitted data

## U

**UNC**
: Unified Coarse Thread

**User**
: Operator using the device on which the user interface created with Galileo is running.

**UTC**
: Universal Time Coordinated

## W

**widescreen**
: Widescreen format

**Windows**
: Dialog boxes, prompts, etc. that open while the application is running and remain on the current program page Synonyms: dialogue box, dialog These windows are shown by the application in various situations in order to obtain specific input or confirmations from the user. Dialog boxes expect input from the user, while prompts are shown to get the user's confirmation for specific messages.

**Windows Embedded Compact 7 pro**
: A component-based, multi-functional real-time operating system designed to meet the needs of even the most demanding industrial applications. The operating system will unpack itself into the device's RAM memory every time the device is started, making it possible to de-energize the device without the need for a UPS or other similar precautions. As the successor to WinCE 6, Windows Embedded Compact 7 features a large selection of standard technologies, communication servers/clients, and web services. • Fast boot time • Multi-touch support (two-finger) • Long-term availability

**WINS**
: Windows Internet Name Service, Name resolution service within Microsoft networks. In order for this service to be used, there must be a WINS server. If there is no WINS server, names will be resolved using broadcasts and other mechanisms. A fixed name can be assigned to an IP address in WINS so that a computer will continue to be recognized even if its IP address changes.

Eaton’s electrical business is a global leader with expertise in power distribution and circuit protection; backup power protection; control and automation; lighting and security; structural solutions and wiring devices; solutions for harsh and hazardous environments; and engineering services. Eaton develops innovations that not only provide energy for the important things across industries worldwide, but also help customers tackle the most critical electrical power management challenges out there.
The Eaton Corporation is a diversified power management company with 2017 sales of 20.4 billion dollars. We provide energy-efficient solutions that help our customers effectively manage electrical, hydraulic, and mechanical power more efficiently, safely, and sustainably. Eaton is dedicated to improving people's quality of life and the environment through the use of power management technologies and services. Eaton has approximately 96,000 employees and sells products to customers in more than 175 countries. For more information, please visit Eaton.com.

Eaton Addresses Worldwide:
www.eaton.eu

Email: automation@eaton.com
Internet: www.eaton.eu/easy

Eaton Industries GmbH,
Hein-Moeller-Straße 7-11,
D-53115 Bonn


10/19 MN050009 EN

EATON
Powering Business Worldwide