

## easyE4



Powering Business Worldwide

## **Company information**

All brand and product names are trademarks or registered trademarks of their respective owners.

### **Service**

For service and support, please contact your local sales team.

Contact info. [Eaton.com/contact](https://Eaton.com/contact)

Service page: [Eaton.com/aftersales](https://Eaton.com/aftersales)

### **Original Operating Instructions**

is the German-language edition of this document

Publication date

04/25 MN050009EN Edition 8.31, Build 240

Copyright

© 2018 Eaton Industries GmbH, 53105 Bonn

All rights, including those of translation, reserved.

No part of this manual may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, micro-filming, recording or otherwise, without the prior written permission of Eaton Industries GmbH, Bonn.

Subject to alteration.



**DANGER!**

Hazardous 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 switching off 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 switching off 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

	<b>easyE4 manual</b>	<b>1</b>
	Company information	2
	Before starting with the installation	3
	Table of Contents	1
0.1	About this manual	15
0.1.1	List of revisions	16
0.1.2	Target group	17
0.1.3	Legal disclaimer	18
0.1.4	Short designations	19
0.1.5	Writing conventions	20
0.1.5.1	Warning labels	20
0.1.5.2	Additional information for use	21
<b>1.</b>	<b>Description of easyE4 control relay</b>	<b>23</b>
1.1	Use as intended	23
1.2	Function	24
1.3	Device models - versions and part nos.	27
1.3.1	Base device versions	27
1.3.2	Expansion versions	29
1.3.3	Overview of available easyE4 devices	31
1.4	What the different parts of the part number mean	33
1.5	Accessory devices	34
1.6	Nameplate	36
1.7	Support	36
1.8	easySoft 8 programming software	37
1.8.1	System requirements	38
1.9	Safety regulations	39
1.9.1	Basics	39
1.9.2	Mandatory requirements, personnel requirements	39
1.9.2.1	Occupational safety	39
1.9.2.2	Personnel qualifications	39
1.9.2.3	Device documentation	40

1.9.2.4	Installation, maintenance, and disposal .....	40
1.9.2.5	Prerequisites for proper operation .....	41
1.9.3	Device-specific hazards .....	42
1.10	Engineering .....	46
1.10.1	Length of Signal input cables .....	46
1.10.1.1	Digital inputs .....	46
1.10.1.2	Analog inputs .....	47
1.10.2	Length of signal analog output lines .....	47
1.10.3	Notes for connection of EASY-E4-AC-... devices .....	48
1.10.3.1	Connect digital AC inputs .....	48
1.10.4	Analog Signals .....	51
1.10.5	Notes on connecting an easy communication module .....	52
<b>2.</b>	<b>Installation .....</b>	<b>53</b>
2.1	Prerequisites for the location of use .....	54
2.1.1	Installation position .....	54
2.1.1.1	Temperatures .....	54
2.1.1.2	Aeration and de-aeration .....	55
2.2	Unpacking and checking the equipment supplied .....	56
2.3	Mounting .....	58
2.3.1	Mounting easyE4 control relays .....	58
2.3.1.1	Installation on mounting rail .....	62
2.3.1.2	Screw mounting .....	64
2.3.1.3	Dismounting of a device .....	65
2.4	Connection terminals .....	66
2.4.1	Screw terminals .....	66
2.4.2	Push-in terminals .....	67
2.4.3	Connecting the power supply .....	68
2.4.3.1	Special notes on connecting EASY-E4-AC-... devices .....	70
2.4.4	Connect digital inputs .....	71
2.4.4.1	Special considerations for EASY-E4-AC-... expansions .....	72
2.4.4.2	Connect digital counter inputs .....	73
2.4.5	Connecting analog inputs .....	74
2.4.6	Connecting relay outputs .....	75

2.4.7	Connecting transistor outputs .....	76
2.4.7.1	Transistor output behavior in the event of a short circuit/overload .....	77
2.4.7.2	Connecting outputs in parallel .....	77
2.4.8	Connecting analog I/O expansion devices .....	78
2.4.9	Connecting analog inputs with temperature measuring on expansion devices .....	80
2.4.10	Terminal configurations for individual devices .....	83
2.5	External connections on the base device .....	87
2.5.1	External connection layouts .....	87
2.5.2	Memory card .....	88
2.5.3	Ethernet .....	90
2.5.3.1	Connecting the Ethernet cable .....	91
2.5.3.2	Removing the Ethernet cable .....	92
2.6	Programming software license .....	93
2.6.1	Licensing .....	94
2.6.2	Adding a license key later on .....	96
2.6.3	Software updates and hardware changes .....	97
2.6.4	easyE4 root certificate .....	97
2.6.5	Installation instructions .....	98
<b>3.</b>	<b>Commissioning .....</b>	<b>105</b>
3.1	Initial commissioning .....	105
3.2	Daily operation .....	106
3.3	Switch on .....	106
3.3.1	Startup behavior of easyE4 control relay with LED indicators ..	106
3.3.2	Startup behavior of control relay easyE4 with a display and keypad .....	108
3.3.3	Startup behavior of base devices with connected expansion devices .....	110
3.3.4	Status display on control relay easyE4 with display and keypad .....	111
3.3.5	Commissioning the Ethernet network .....	113
3.3.6	Remote operation .....	114
3.4	Overview of switch-on behavior .....	115

3.5	Establishing an Ethernet connection and transferring a program or visualization project .....	117
3.5.1	Basic information on assigning IP addresses .....	117
3.6	Automatic booting of the memory card .....	124
3.6.1	Preparing the card in the device for booting with easySoft 8 ...	125
3.6.2	Preparing the card in the easyE4 device for booting with easySoft 8 .....	129
3.6.3	Preparing the card for booting on the easyE4 device itself .....	133
3.7	Reset with memory card - reset device to its delivery condition .....	135
3.8	Updating firmware .....	136
3.8.1	Firmware Update base device .....	138
3.8.2	Updating the firmware on expansion devices .....	141
3.8.3	Updating the firmware on easy communication modules .....	143
3.9	Functions of the microSD memory card .....	146
3.9.1	Ejecting the microSD memory card .....	146
3.10	Setting a splash screen for the EASY-E4-...-12...C1(P) display ..	147
3.11	Set system parameters using a memory card - e4settings.ini ..	148
<b>4.</b>	<b>Operation .....</b>	<b>153</b>
4.1	Base device with display and buttons .....	153
4.1.1	LCD Display .....	153
4.1.1.1	Display color backlight .....	154
4.1.2	Keyboard .....	154
4.1.3	Selecting menus and entering values .....	155
4.1.4	Cursor display .....	156
4.1.5	Entering of values .....	156
4.2	Operating modes of the easyE4 .....	157
4.2.1	RUN mode .....	157
4.2.2	STOP mode .....	157
4.3	Operation of the menu selection and value entry .....	159
4.3.1	How to navigate the device menus .....	159
4.3.2	Operating principle in the circuit diagram and function block editor .....	159
4.3.3	Selecting a device menu .....	160

4.4	Overview of the menus on the device .....	161
4.4.1	Main menu .....	161
4.4.2	STOP RUN operating mode menu .....	161
4.4.3	Menu Parameter .....	162
4.4.4	Set clock menu .....	163
4.4.5	Menu Card .....	164
4.4.6	MenuInformation .....	165
4.4.7	System options menu .....	166
4.4.8	Program menu .....	168
4.5	Your first EDP program .....	170
4.5.1	Draw a wiring diagram .....	172
4.5.2	Testing the circuit diagram .....	176
4.5.3	Control options in RUN mode .....	177
4.5.4	Delete Program .....	179
4.6	Transfer program to the easyE4 device .....	180
4.6.1	Transfer with a microSD memory card .....	180
4.6.2	Establishing an Ethernet connection .....	185
<b>5.</b>	<b>Programming on the device .....</b>	<b>187</b>
5.1	Program .....	187
5.2	Circuit diagram display .....	187
5.3	Circuit diagram elements .....	189
5.3.1	Function blocks .....	189
5.3.2	Relays .....	189
5.3.3	Contacts .....	190
5.3.4	Coils .....	191
5.4	Working with contacts and coils .....	196
5.4.1	Entering and modifying contacts .....	197
5.4.2	Changing an N/O contact to an N/C contact .....	198
5.4.3	Entering and modifying coils .....	199
5.4.4	Deleting contacts and coils .....	200
5.4.5	Creating and modifying connections .....	201
5.4.6	Deleting connections .....	202
5.4.7	Adding a rung .....	202

5.4.8	Deleting a rung .....	202
5.4.9	Got to a rung .....	203
5.4.10	Saving the circuit diagram .....	203
5.4.11	Exiting the circuit diagram without saving .....	204
5.4.12	Searching for contacts and coils .....	204
5.4.13	Switching with the Cursor Buttons .....	205
5.4.14	Checking the circuit diagram .....	206
5.4.15	Jumps .....	207
5.4.16	Wiring NET operands in the circuit diagram .....	209
5.5	Transferring programs from and to a microSD memory card ...	213
5.5.1	Configuration on a base device with a display .....	214
5.5.1.1	PROGRAM submenu .....	215
5.6	Working with function blocks .....	217
5.6.1	Adding function blocks to the circuit diagram for the first time	217
5.6.2	Function block list .....	219
5.6.3	Configuring parameters in the function block editor .....	220
5.6.4	PARAMETERS menu .....	223
5.6.5	Deleting function blocks .....	224
5.7	Using operands in a program .....	226
5.7.1	Elementary data types .....	226
5.7.2	Permissible operands at a glance .....	227
5.7.3	Connection rules for operands .....	228
5.7.4	Overview of operands Numeric formats .....	229
5.7.5	Timer constant .....	230
5.7.6	Organizing marker ranges .....	234
5.7.7	Operand table .....	236
5.7.8	Retentive markers .....	239
5.7.9	Internal marker ranges in function blocks .....	239
<b>6.</b>	<b>Function blocks .....</b>	<b>241</b>
6.1	Manufacturer function blocks .....	244
6.1.1	Timer modules .....	244
6.1.1.1	HW - Weekly timer (Hour Week) .....	244
6.1.1.2	HY - Year time switch (Hora Year) .....	254

6.1.1.3	OT - Operating hours counter .....	264
6.1.1.4	RC - Real-time clock .....	269
6.1.1.5	T - Timing relay .....	272
6.1.1.6	YT - Year time switch (Year Table) .....	284
6.1.1.7	WT - Weekly timer (WeekTable) .....	292
6.1.1.8	AC - Astronomic clock .....	296
6.1.2	Counter Function Blocks .....	305
6.1.2.1	C - Counter Relay .....	305
6.1.2.2	CF - Frequency counter .....	311
6.1.2.3	CH - High-speed counter .....	317
6.1.2.4	CI - Incremental Counter .....	323
6.1.3	Arithmetic and analog function blocks .....	330
6.1.3.1	A - Analog value comparator .....	330
6.1.3.2	AR - Arithmetic .....	336
6.1.3.3	AV - Average .....	342
6.1.3.4	CP – Comparator .....	350
6.1.3.5	LS - Value scaling .....	354
6.1.3.6	MM - Min-/Max function .....	359
6.1.3.7	PM - Performance map .....	362
6.1.3.8	PW - Pulse width modulation .....	368
6.1.4	Open-loop and closed-loop function blocks .....	375
6.1.4.1	DC - PID controller .....	375
6.1.4.2	FT - PT1-Signal smoothing filter .....	382
6.1.4.3	PO - Pulse output .....	387
6.1.4.4	TC - Three step controller .....	402
6.1.4.5	VC - Value limitation .....	407
6.1.5	Data and register function blocks .....	411
6.1.5.1	BC - Block comparison .....	411
6.1.5.2	BT - Block transfer .....	419
6.1.5.3	DB - Data function block .....	425
6.1.5.4	ED - EdgeDetector .....	430
6.1.5.5	FF - Flip-flop .....	434
6.1.5.6	MX - Data multiplexer .....	437
6.1.5.7	RE - Recipe records .....	441

6.1.5.8	SR - Shift register .....	447
6.1.5.9	TB - Table function .....	455
6.1.6	NET Function Blocks .....	460
6.1.6.1	GT - Get values from NET .....	460
6.1.6.2	PT - Put values to NET .....	464
6.1.6.3	SC - Synchronizing clock via NET .....	468
6.1.7	Other function blocks .....	472
6.1.7.1	AL - Alarm function block .....	472
6.1.7.2	BV - Boolean operation .....	477
6.1.7.3	D - Text display .....	481
6.1.7.4	D - Text display editor .....	491
6.1.7.5	DL - Data logger .....	510
6.1.7.6	JC - Conditional jump .....	524
6.1.7.7	LB - Jump label .....	529
6.1.7.8	MC - Acyclical Modbus TCP request .....	531
6.1.7.9	MR - Master Reset .....	543
6.1.7.10	MU - Acyclical Modbus RTU request .....	547
6.1.7.11	NC - Numerical converter .....	563
6.1.7.12	ST - Set cycle time .....	569
6.2	Interrupt function blocks .....	572
6.2.1	IC - Counter-controlled interrupt .....	572
6.2.1.1	General .....	572
6.2.1.2	Operating principle .....	573
6.2.1.3	The function block and its parameters .....	574
6.2.1.4	Other .....	577
6.2.2	IE - Edge-controlled interrupt .....	583
6.2.2.1	General .....	583
6.2.2.2	Operating principle .....	584
6.2.2.3	The function block and its parameters .....	585
6.2.2.4	Other .....	587
6.2.3	IT - Time-controlled interrupt function block .....	589
6.2.3.1	General .....	589
6.2.3.2	Operating principle .....	589
6.2.3.3	The function block and its parameters .....	591



6.2.3.4	Other .....	594
6.3	UF - User function block .....	597
6.3.1	General .....	597
6.3.1.1	General information on user function blocks .....	598
6.3.2	Creating a user function block .....	598
6.3.3	Configuring a user function bl .....	602
6.3.4	Programming a user function block .....	608
6.3.4.1	Programming view tabs .....	608
6.3.5	Adding comments to user function blocks .....	610
6.3.6	Calling a user function block in the main program .....	611
6.3.6.1	User function blocks in an ST main program .....	614
6.3.7	Opening a project with an existing user function block .....	616
6.3.8	Saving a user function block .....	617
6.3.8.1	Operands available for user function blocks .....	620
6.3.9	Exporting a user function block .....	622
6.3.9.1	Plausibility check .....	622
6.3.10	Importing a user function block .....	624
6.3.11	Replacing a user function block .....	625
6.3.12	Deleting a user function block .....	627
6.3.13	Comparing user function blocks .....	629
6.3.14	Printing a user function block .....	630
6.4	Timing and counter relay example .....	631
<b>7.</b>	<b>System settings .....</b>	<b>634</b>
7.1	System options - Base device with display and buttons .....	635
7.2	Display .....	636
7.3	Device ID .....	636
7.4	Splash screen .....	637
7.5	NET .....	638
7.6	Ethernet .....	640
7.7	Update .....	642
7.8	Switch languages .....	644
7.9	Setting the startup behavior .....	645
7.9.1	Enabling / disabling the RUN START option .....	646

7.9.1.1	Configuration on a base device with a display .....	646
7.9.2	Enabling / disabling the CARD START option .....	646
7.9.2.1	Configuration on a base device with a display .....	647
7.9.2.2	Configuration in easySoft 8 .....	647
7.10	Debounce .....	648
7.10.1	Configuring input debouncing on a base device with a display .....	648
7.10.2	Configuring input debouncing in easySoft 8 .....	648
7.11	P buttons .....	649
7.11.1	Configuring the P buttons on a base device with a display .....	649
7.11.2	Configuring the P buttons in easySoft 8 .....	649
7.12	Define program name .....	650
7.13	Retention function .....	651
7.13.1	Retention in easySoft 8 .....	653
7.14	Security – password protection .....	654
7.14.1	Configuring the password on a base device with a display .....	654
7.14.1.1	What happens if you forget your password or enter the wrong password? .....	657
7.15	Configuring the microSD card and device ID .....	658
7.16	Time and Date setting .....	659
<b>8.</b>	<b>How easyE4 works internally .....</b>	<b>664</b>
8.1	Program execution .....	664
8.2	Transferring an existing circuit diagram .....	667
8.3	Device information .....	668
8.4	NET network .....	669
8.5	Operating states easyE4 .....	672
8.6	Controlling the backlight with operands .....	673
8.6.1	Backlight intensity .....	673
8.6.2	Background color .....	673
8.7	Device easyE4 time responses .....	676
8.7.1	Time behavior of the inputs and outputs .....	676
8.7.2	Base device timing .....	677
8.7.2.1	Delay time for operation with DC power supply .....	677
8.7.2.2	Delay time for operation with AC power supply .....	679
8.7.3	Timing characteristics of expansion devices .....	681

8.7.3.1	Delay time for AC expansion devices .....	682
<b>9.</b>	<b>Operating system diagnostic messages .....</b>	<b>684</b>
9.1	Diagnostic messages easy communication module .....	686
9.2	Transistor outputs (overload / short-circuit) .....	687
9.3	Diagnostics and diagnostic buffer .....	687
9.4	LED status messages on the device .....	688
<b>10.</b>	<b>Communication Connection to other devices easyE4 .....</b>	<b>690</b>
10.1	Secure communications with easyProtocol V2 .....	691
10.2	Secure communications via HTTPS (encrypted) .....	693
10.3	Windows 7 operating systems and easyProtocol V1 .....	694
10.4	Windows 7 operating systems and easySoft 8 – pay attention to the project size .....	695
10.5	easyProtocol V1 .....	696
10.6	Compatibility rules for going online .....	698
10.7	Establish a connection to the device .....	700
10.8	Terminating the connection to the device .....	704
10.9	Setting up a connection to multiple devices on the NET .....	705
10.10	Taking the Ethernet and NET configuration from the device ....	708
10.11	Secure communication with certificates .....	709
10.11.1	What is the Eaton easyE4 root certificate for? .....	709
10.11.2	When is the Eaton easyE4 root certificate needed? .....	710
10.11.3	What should I do if I am unable to establish a connection due to a certificate error? .....	710
10.11.4	How does a certificate request work? .....	711
10.11.5	Installing the Eaton easyE4 root certificate automatically at the same time as easySoft 8 .....	712
10.11.6	Installing the Eaton easyE4 root certificate separately .....	713
10.11.7	How can I check to make sure that the Eaton easyE4 root cer- tificate has been successfully installed on my PC/tablet/cell phone? .....	717
10.12	Setting up a NET group .....	721
10.12.1	Access on the NET .....	722
10.12.2	Communication via NET .....	723
10.12.3	NET settings .....	725

10.13	Setting up a Web Server .....	728
10.13.1	Webserver tab .....	728
10.13.2	Configuring the web server function in easySoft 8 .....	732
10.13.2.1	Setting up users .....	732
10.13.2.2	Setting the web server login text .....	733
10.13.2.3	Configuring the web server's startup behavior .....	733
10.13.2.4	Configuring the settings in the Webserver tab .....	734
10.14	Web Client .....	736
10.14.1	Web Client start .....	738
10.14.2	Using the Web Client .....	740
10.14.2.1	Menu bar Menu bar .....	740
10.14.2.2	List .....	742
10.14.3	Updating operands .....	743
10.14.3.1	Updating the Web Client .....	743
10.14.4	Display .....	744
10.14.5	Operands .....	745
10.14.6	NET Operands .....	746
10.14.7	Parameter list .....	747
10.14.8	Diagnostics .....	750
10.14.9	Settings .....	751
10.14.9.1	General settings .....	751
10.14.9.2	Network settings .....	751
10.14.9.3	E-mail settings .....	752
10.14.9.4	API key .....	753
10.14.9.5	System update .....	754
10.14.9.6	Web Client .....	755
10.15	Setting up the e-mail function .....	758
10.15.1	E-mail tab .....	759
10.16	easy communication modules .....	769
10.16.1	easyE4 as a SmartWire-DT coordinator .....	770
10.16.1.1	SmartWire-DT - The System .....	770
10.16.1.2	EASY-COM-SWD-... easy communication module .....	772
10.16.1.3	LED status messages on the EASY-COM-SWD-... com- munication module .....	778

10.16.2	easyE4 Communication via Modbus RTU .....	782
10.16.2.1	EASY-COM-RTU-... easy communication module .....	784
10.16.2.2	LED status messages on EASY-COM-RTU-... communication module .....	789
10.17	Connecting to the AWS Cloud .....	792
10.17.1	AWS - access .....	796
10.17.1.1	Enabling data transfers .....	797
10.17.2	Creating accounts for Amazon Web Services (AWS) .....	802
10.17.3	AWS IoT Core – registering an easyE4 .....	808
10.17.3.1	Method 1: using the easyE4 Wizard .....	808
10.17.3.2	Method 2: Using a Python script .....	812
10.17.4	MQTT test client .....	815
10.17.5	Update via AWS IoT Jobs .....	815
10.17.5.1	Updating firmware through the cloud .....	816
10.17.5.2	Updating easyE4 programs through the cloud .....	818
10.17.5.3	Creating a job .....	820
10.17.5.4	S3 bucket .....	824
10.17.5.5	Setting an update ID for cloud connections .....	826
10.17.6	AWS diagnostics .....	828
10.17.6.1	Messages for cloud computing diagnostics via easyE4 .....	828
10.18	Modbus TCP .....	830
10.18.1	easyE4 as a Modbus TCP client .....	832
10.18.2	easyE4 as a Modbus TCP server .....	844
10.18.2.1	Programming communication with Modbus TCP .....	844
10.18.2.2	Modbus TCP error handling .....	851
10.19	Convenient visualization for easyE4 .....	856
10.19.1	easyE Remote Touch Display .....	856
10.19.2	HMI Touchdisplays .....	859
<b>11.</b>	<b>Faults .....</b>	<b>860</b>
11.1	Messages from the operating system .....	861
11.2	Possible situations when creating programs .....	862
11.3	Event .....	863
11.4	Functionality of the NET faulty .....	864
11.5	Issues related to the microSD memory card .....	865

<b>12.</b>	<b>Maintenance</b>	<b>868</b>
12.1	Cleaning and maintenance	868
12.2	Repairs	868
12.3	Storage, transport and disposal	869
12.3.1	Storage and transport	869
12.3.2	Disposal	870
	<b>Appendix</b>	<b>872</b>
A.1	Dimensions	873
A.2	Approvals and declarations	878
A.3	easyE4 compatibility overview	881
A.4	Parts of an easyE4 project file (*.e80)	882
A.5	Technical data	883
A.5.1	Data sheets	883
A.5.2	Overview of select characteristics	885
A.6	Required memory for function blocks	889
A.7	Additional information for use	893
A.7.1	Documents	893
A.7.1.1	Instruction leaflets	893
A.7.1.2	Manuals	893
A.7.1.3	Documents for SmartWire-DT communication system	893
A.7.2	Download Center, Eaton Online Catalog	894
A.7.3	Product information	894
A.7.4	Product training	894
A.7.5	Community	894
A.7.6	Cybersecurity	894
A.7.7	Internet links	895
A.8	Sample Projects	896
	Alphabetical index	898
	List of Figures	912
	Glossary	926

### 0.1 About this manual

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

The 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 integral part of the software, the easySoft 8 Help system summarizes the sections that are relevant to understanding how the corresponding programming works.

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.



[Eaton.com/documentation](https://www.eaton.com/documentation)

Please send any comments, recommendations, or suggestions regarding this document to: [DocumentationEGBonn@eaton.com](mailto:DocumentationEGBonn@eaton.com)

## 0.1 About this manual

### 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 1st edition		New edition	✓		
11/2018	A3 A5 24	Real-time clock characteristic curve Sample program Catalog No. MEMORY-SUD-A1		✓	
1/2019	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, microSD card		✓	
10/2019	ff	Device versions with push-in terminals	✓	✓	
11/2019 3rd edition	ff	cULus modification for EASY-E4-AC-...		✓	
09/2020 4th edition	ff	Added easy communication module EASY-COM-SWD-..., Modbus TCP, additional touch displays	✓	✓	
11/2021 5th edition	ff	Added EASY-COM-RTU-... easy communication module, links to *.com, removed part number for connectors	✓	✓	✓
07/2022 6th edition	ff	Adaptations for basic devices hardware version 08	✓	✓	
06/2023 7th edition (DE only)	ff	Bugfix package for easySoft V8.01 and configuration with e4setting.ini file	✓	✓	
02/2024 8th edition	ff	Extension with visualization view / easyE RTD Advanced	✓	✓	
04/2024 Edition 8.1	ff	Corrections		✓	
07/2024 Edition 8.2	ff	Expanded to include a connection to the AWS cloud service and WebEditor function with easySoft V8.2x	✓	✓	
12/2024 Edition 8.3	ff	Functional expansions; expansion in AWS cloud service with easySoft V8.3x		✓	



### 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 8.
- Want to test or commission a developed application
- Maintain an application with easySoft 8.
- 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 About this manual

### 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 8 programming software

are subject to the following safety instructions and operating guidelines:

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 easySoft 8, 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 terminal type screw terminals
EASY-E4-...-12...C1P	Type with push-in terminals
EASY-E4-...-12...CX1(P)	Base devices from the product family with diagnostic LEDs
EASY-E4-...-12...CX1	Type with terminal type screw terminals
EASY-E4-...-12...CX1P	Type with push-in terminals
EASY-E4-...-...E1(P)	All input and output expansions as devices in the product family
EASY-E4-...-...E1	Type with terminal type screw terminals
EASY-E4-...-...E1P	Type with push-in terminals
easySoft 8	Programming software for easyE4 devices
EASY-COM-...	easy communication modules for easyE4 devices



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

0.1 About this manual


0.1.5 Writing conventions

Tab. 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 8
Menu path\submenu\... \item	Used for paths to views and dialog boxes in easySoft 8
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

**ATTENTION**  
Warns about the possibility of material damage.

Prohibited use



**Prohibited uses, actions, etc.**  
Prohibition signs interdict actions or the use of certain objects

Requirements



**Requirement**  
Mandatory signs require a certain behavior

Notes




► Indicates instructions to be followed



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


0.1.5.2 Additional information for use

Documents (such as manuals) are listed after the  icon together with the corresponding name and Eaton number.



Publication title

For identifying the Eaton publication code

Links to external Internet addresses. They will be shown after the  icon.



Destination address without http(s)://www.

Links are shown in [blue](#).

## 0.1 About this manual

## **1. Description of easyE4 control relay**

### **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.



**Requirement**

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. Description of easyE4 control relay

### 1.2 Function

### 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.

#### 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 firmware; 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 8

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 8 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



## 1. Description of easyE4 control relay

### 1.2 Function

or falling edge-triggered relays or as latching relays.

- ...

## 1. Description of easyE4 control relay

### 1.2 Function

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 241

If you wish to wire a easyE4 device via your PC, i.e. create a circuit diagram, use the easySoft 8

→ Section "easySoft 8 programming software", page 37.

If you want to connect an easyE4 device to a visualization, use Eaton touch displays

→ Section "Convenient visualization for easyE4", page 856

If you want to use the functionality of the easyE4 series to directly implement a controller in a communication system, use an easy communication module

→ Section "easy communication modules ", page 769.

The easyE4 can log operating states with its data logging feature and run a simple analysis of events. In addition, it makes diagnostics easier with status information for all communication nodes and expansion modules.

With added cloud connectivity and visualization functions, the easyE4 can be used to take advantage of the Internet of Things and the numerous benefits provided by digitalization. The easyE4 supports Node-RED and JSON {json}. Moreover, the hardware security module means that information can be stored in conformity with the latest cybersecurity requirements in order to ensure secure communications from the device with a certificate.

### **1.3 Device models - versions and part nos.**

All easyE4 devices come with 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.

EASY-COM-... easy communication modules can be used with an easyE4 base device starting with generation 05.

#### **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

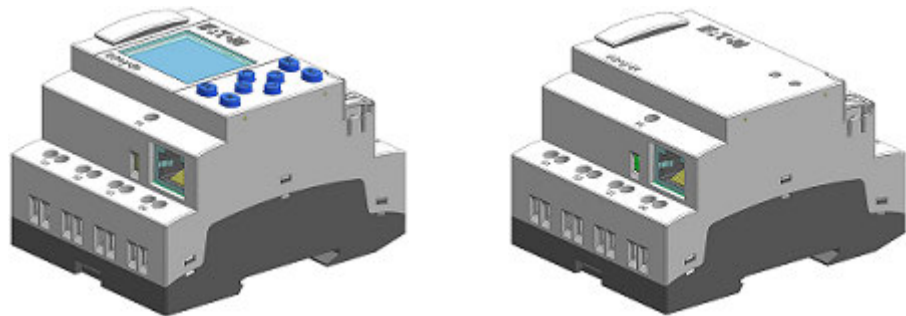
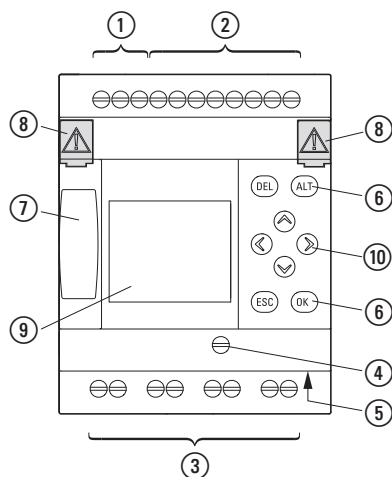


Fig. 1: Device model with EASY-E4-...-12...C1(P) display and button controls or with EASY-E4-...-12...CX1 (P) LED display for diagnostics

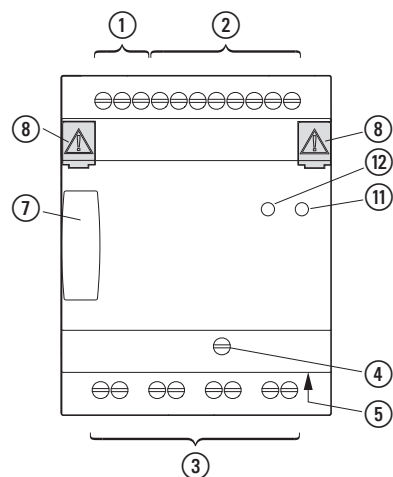
## 1. Description of easyE4 control relay

### 1.3 Device models - versions and part nos.

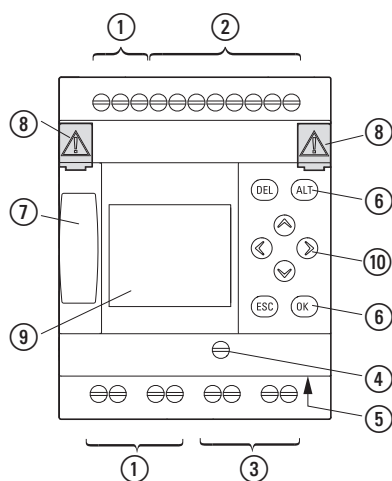
EASY-E4-UC-12RC1(P),  
EASY-E4-AC-12RC1(P)



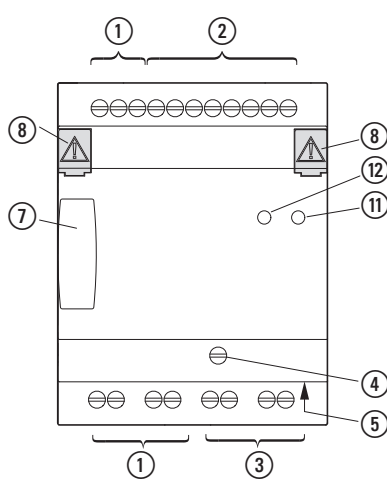
EASY-E4-UC-12RCX1(P),  
EASY-E4-AC-12RCX1(P)



EASY-E4-DC-12TC1(P)



EASY-E4-DC-12TCX1(P)



① Power Supply

② Input points

③ Outputs

④ Ethernet connection  
Functional earth

⑤ Ethernet socket

⑥ Pushbuttons

⑦ Slot for  
microSD memory card

⑧ Covering cap

⑨ Display

⑩ Cursor buttons

⑪ LED POW/RUN

⑫ LED ETHERNET/NET

## 1. Description of easyE4 control relay

### 1.3 Device models - versions and part nos.

#### 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)

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)

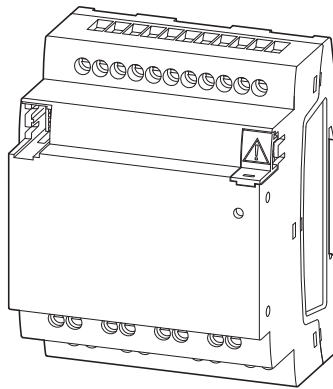


Fig. 2: Device models in 4SU

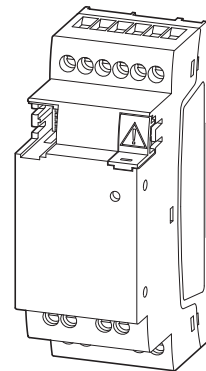
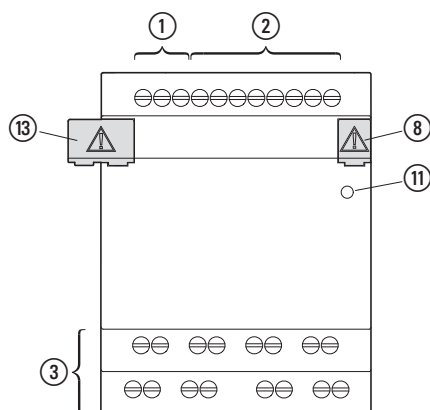


Fig. 3: Device models in 2SU

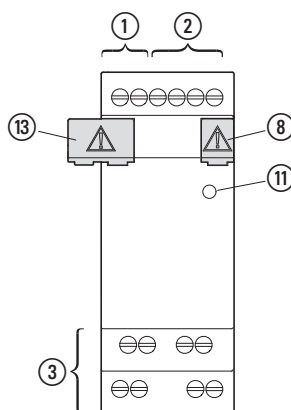
# 1. Description of easyE4 control relay

## 1.3 Device models - versions and part nos.

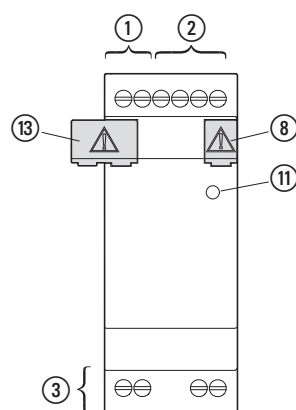
EASY-E4-...-16...



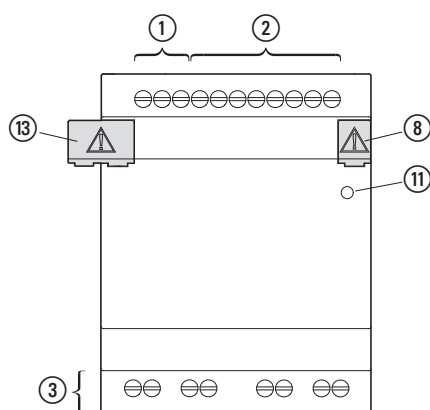
EASY-E4-...-8...



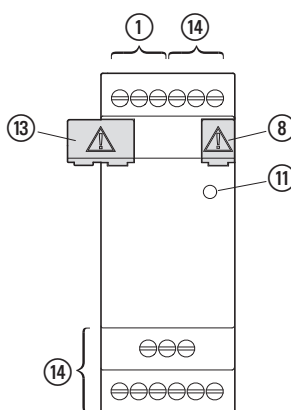
EASY-E4-DC-8TE1(P)



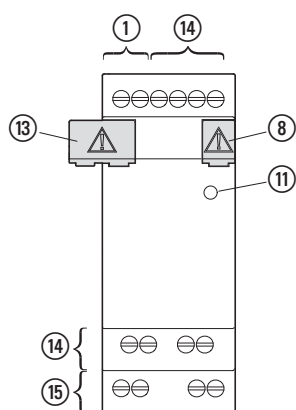
EASY-E4-DC-16TE1(P)



EASY-E4-DC-4PE1(P)



EASY-E4-DC-6AE1(P)



- |                       |                      |
|-----------------------|----------------------|
| 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 |                      |



For optional EASY-COM-... module versions, please refer to  
→ Chapter "1 easy communication modules ", page 769

# 1. Description of easyE4 control relay

## 1.3 Device models - versions and part nos.

### 1.3.3 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.

#### easyE4 control relays

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

Catalog No. and type	Description
197211 - EASY-E4-UC-12RC1 197504 - EASY-E4-UC-12RC1P	Base device with display; 12/24 V <sub>DC</sub> , 24 V <sub>AC</sub> ; digital inputs: 8, of which 4 can be used as analog inputs; digital outputs: 4 relay outputs
197212 - EASY-E4-UC-12RCX1 197505 - EASY-E4-UC-12RCX1P	Base device with diagnostic LED; 12/24 V <sub>DC</sub> , 24 V <sub>AC</sub> ; digital inputs: 8, of which 4 can be used as analog inputs; digital outputs: 4 relay outputs
197213 - EASY-E4-DC-12TC1 197506 - EASY-E4-DC-12TC1P	Base device with display; 24 V <sub>DC</sub> ; digital inputs: 8, of which 4 can be used as analog inputs; digital outputs: 4 transistor outputs
197214 - EASY-E4-DC-12TCX1 197507 - EASY-E4-DC-12TCX1P	Base device with diagnostic LED; 24 V <sub>DC</sub> ; digital inputs: 8, of which 4 can be used as analog inputs; digital outputs: 4 transistor outputs
197215 - EASY-E4-AC-12RC1 197508 - EASY-E4-AC-12RC1P	Base device with display; 100 – 240 V <sub>AC</sub> , 100 – 240 V <sub>DC</sub> (cULus 100 – 110 V DC); digital inputs: 8; digital outputs: 4 relay outputs
197216 - EASY-E4-AC-12RCX1 197509 - EASY-E4-AC-12RCX1P	Base device with diagnostic LED; 100 – 240 V <sub>AC</sub> , 100 – 240 V <sub>DC</sub> (cULus 100 – 110 V DC); digital inputs: 8; digital outputs: 4 relay outputs

## 1. Description of easyE4 control relay

### 1.3 Device models - versions and part nos.

#### I/O expansion for easyE4 control relays

- With screw terminals EASY-E4-...-...E1 or push-in terminals EASY-E4-...-...E1P

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

#### easy communication modules for easyE4 control relays

- Featuring screw terminals EASY-COM-...1

Catalog No. and type	Description
199452 - EASY-COM-SWD-C1	For using the easyE4 control relay as a SmartWire-DT coordinator in a SmartWire-DT line
199453 - EASY-COM-RTU-M1	For using the easyE4 control relay with Modbus RTU



## 1. Description of easyE4 control relay

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

#### 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.

Tab. 2: Key to part numbers

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

## 1. Description of easyE4 control relay

### 1.5 Accessory devices

### 1.5 Accessory devices

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

#### **NOTICE**

Only use original accessories.



Order accessories through your supplier or through the EATON online catalog

Example:

Cat No. and type	Description
<a href="#">198513 XV-102-A0-35TQRB-1E4</a>	Touch display for easyE4, 3.5 inches, 24 V <sub>DC</sub> , TFTcolor, QVGA 320 x 240 pixels, Ethernet
<a href="#">199734 XV-102-A3-57TVRB-1E4</a>	Touch display for easyE4, 5.7 inches, 24 V <sub>DC</sub> , TFTcolor, VGA 640 x 480 pixels, Ethernet
<a href="#">199740 EASY-RTD-DC-43-03B1-00</a>	easy Remote Touch Display, 4.3 inches, easyE RTD Standard 24 V <sub>DC</sub> , TFTcolor, 480x272 px, Res., Ethernet, RS485
<a href="#">EP-401057 EASY-RTD-DC-43-03B2-00</a>	easyE Remote Touch Display, easyE RTD Advanced 4.3 inches 24 V <sub>DC</sub> , FTcolor, 480x272 px, Res., Ethernet, RS485
<a href="#">191087 MEMORY-SUD-A1</a>	microSD 2 GB memory card with adapter, I Grade, without an operating system
<a href="#">197226 EASYSOFT-SWLIC</a>	Programming software license easySoft 8
<a href="#">061360 ZB4-101-GF1</a>	Device foot for screw mounting
<a href="#">197225 EASY-E4-CONNECT1</a>	Spare parts package for expansion modules, consisting of three (3) connectors and three (3) end covers
<a href="#">199513 EASY-E4-CONNECT-COM1</a>	Spare parts package for communication modules, consisting of three (3) connectors and three (3) end covers
<a href="#">229424 EASY200-POW</a>	Switched-mode power supply unit, 100-240 V <sub>AC</sub> / 24 V <sub>DC</sub> / 12 V <sub>DC</sub> , 0.35 A / 0.02 A, single-phase, controlled
<a href="#">212319 EASY400-POW</a>	Switched-mode power supply unit, 100-240 V <sub>AC</sub> / 24 V <sub>DC</sub> , 1.25 A, single-phase, controlled
<a href="#">272484 TR-G2/24</a>	Transformer, 230 V, 12/24 V, 2/1 A
<a href="#">199711 XN-332-5ETH-UMS</a>	Industrial standalone switch as slice module, 5 ports, 100 Mbit/s
<a href="#">EP-401058 EASY-E4-BOX-SKF-4TE</a>	Hinged inspection window for 4RU
<a href="#">EP-401059 EASY-E4-BOX-SKF-6TE</a>	Hinged inspection window for 6RU

## 1. Description of easyE4 control relay

### 1.5 Accessory devices

#### Starter packages

We have various packages with limited availability that are intended to make it easier to get started with control applications.

Cat No. and type	Starter kit consists of:
198514 XV100-BOX-E4-DC1	EASY-E4-DC-12TC1 control relay, XV-102-AO-35TQRB-1E4 touch display , Ethernet switch, three patch cables to connect the devices to a PC, and one EASYSOFT-SWLIC license.
198515 XV100-BOX-E4-UC1	EASY-E4-UC-12RC1 control relay, XV-102-AO-35TQRB-1E4 touch display , Ethernet switch, three patch cables to connect the devices to a PC, and one EASYSOFT-SWLIC license.
197227 EASY-BOX-E4-UC1	EASY-E4-UC-12RC1 control relay, one patch cable to connect the control relay to the Ethernet port, and one EASYSOFT-SWLIC license.
197228 EASY-BOX-E4-DC1	EASY-E4-DC-12TC1 control relay, one patch cable to connect the control relay to the Ethernet port, and one EASYSOFT-SWLIC license.
197229 EASY-BOX-E4-AC1	EASY-E4-AC-12RC1 control relay, one patch cable to connect the control relay to the Ethernet port, and one EASYSOFT-SWLIC license.
199507 EASY-BOX-E4-UC-SWD1	control relay EASY-E4-UC-12RC1, as well as EASY-COM-SWD-C1 and a EASYSOFT-SWLIC license.
199508 EASY-BOX-E4-UCX-SWD1	control relay EASY-E4-UC-12RCX1, as well as EASY-COM-SWD-C1 and a EASYSOFT-SWLIC license.
199509 EASY-BOX-E4-DC-SWD1	control relay EASY-E4-DC-12TC1, as well as EASY-COM-SWD-C1 and a EASYSOFT-SWLIC license.
199510 EASY-BOX-E4-DCX-SWD1	control relay EASY-E4-DC-12TCX1, as well as EASY-COM-SWD-C1 and a EASYSOFT-SWLIC license.
199511 EASY-BOX-E4-AC-SWD1	control relay EASY-E4-AC-12RC1, as well as EASY-COM-SWD-C1 and a EASYSOFT-SWLIC license.
199512 EASY-BOX-E4-ACX-SWD1	control relay EASY-E4-AC-12RCX1, as well as EASY-COM-SWD-C1 and a EASYSOFT-SWLIC license.

## 1. Description of easyE4 control relay

### 1.6 Nameplate

### 1.6 Nameplate

The device can be identified by checking the nameplate on its side.

This nameplate includes the following information:

- Manufacturer
- Generation (hardware revision)
- 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:

- [EPAS code](#) (digital nameplate)
- 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 firmware version
- If applicable, build No., version easySoft 8

## **1.8 easySoft 8 programming software**

easyE4 control relays are designed to be programmed with the easySoft 8 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 8 also allows you 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
- Create the visualization project file for the easyE RTD Advanced easy Remote Touch Display.
- Integrate the easyE4 into the IoT.

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

The easySoft 8 help is an integral part of easySoft 8 and is designed to help you use the programming software.

There are four programming languages available:

- easy Device Programming (EDP)
- Ladder diagram (LD)
- Function block diagram (FBD)
- Structured Text (ST)

### **Installing multiple easySoft 8 versions**

If you are using easySoft version 7.40 or higher, you can install multiple different easySoft versions on your computer at the same time (e.g., 8.00, 7.40, and 7.32 or lower).

If, for example, version 7.40 is already installed and you also want to install version 7.41, you do not need to uninstall the former first. When you install 7.41, 7.40 will be uninstalled as part of the installation routine. Likewise, when updating from 7.30 to 7.32, you do not need to uninstall 7.30 before you install 7.32.

When running these minor version update installations, only the new program files will be replaced.

## 1. Description of easyE4 control relay

### 1.8 easySoft 8 programming software

#### Tutorials

For helpful videos that explain how to use specific functions, please visit the product page at [Eaton.com/easy-tutorial](https://Eaton.com/easy-tutorial).

#### 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

[Eaton.com/software/Anwendungsbeispiele/easy/Deutsch](https://Eaton.com/software/Anwendungsbeispiele/easy/Deutsch)

[Eaton.com/software/Application Samples/easy/English](https://Eaton.com/software/Application Samples/easy/English)

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

#### 1.8.1 System requirements

##### Hardware

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

##### Software

One of the following operating systems:

- Windows 10 (32 + 64 Bit)
- Windows 11 (64 Bit)

## 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. Description of easyE4 control relay

### 1.9 Safety regulations

#### 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 lifecycle 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

 [Eaton.com/documentation](https://www.eaton.com/documentation)

 [Eaton.com/easy](https://www.eaton.com/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](https://www.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 18.

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. Description of easyE4 control relay

### 1.9 Safety regulations

#### 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.

## 1. Description of easyE4 control relay

### 1.9 Safety regulations



#### **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  $\geq 1.5 \text{ mm}^2$ , length  $\leq 350 \text{ mm}$

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.

## 1. Description of easyE4 control relay

### 1.9 Safety regulations



#### **CAUTION DATA LOSS**

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

► Insert the microSD card only when the easyE4 is de-energized.

Do not write to microSD cards constantly:

- microSD 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 microSD card only when the easyE4 is de-energized.

► Before switching off the device, make sure that there are no programs writing to the microSD 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. Do not expose the device to direct thermal radiation from heating appliances.



#### **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.

## 1. Description of easyE4 control relay

### 1.9 Safety regulations



#### 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. Description of easyE4 control relay

### 1.10 Engineering

### 1.10 Engineering

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

#### 1.10.1 Length of Signal input cables

##### 1.10.1.1 Digital inputs

Severe interference can cause a signal 1 on the inputs without a proper signal being applied.

Therefore, observe the following maximum cable lengths without additional circuitry:

Basic device	Digital inputs	Qty.		Max. cable length in m
EASY-E4-UC-12...	24 V DC	8	Of which can be used	100 (unshielded)
EASY-E4-DC-12...			4 (I5, I6, I7, I8) as analog inputs	30 screened
			4 (I1, I2, I3, I4) as frequency counter or rapid counter inputs	20 (screened)
			2 (I1 + I2, I3 + I4) as incremental counter	20 (screened)
EASY-E4-UC-12...	12 V DC	8		100 (unshielded)
EASY-E4-UC-12...	24 V AC	8		40 (unshielded)
EASY-E4-AC-12...	115/230 V AC	8	(I1 - I6)	40 (unshielded)
			(I7, I8)	100 (unshielded)

I/O expansion	Digital inputs	Qty.	Cable length in m
EASY-E4-DC-16TE1(P)	24 V DC	8	100 (unshielded)
EASY-E4-UC-16RE1(P)		4	100 (unshielded)
EASY-E4-DC-8TE1(P)	12 V DC	8	100 (unshielded)
EASY-E4-UC-8RE1(P)		4	100 (unshielded)
EASY-E4-UC-16RE1(P)	24 V AC	8	40 (unshielded)
EASY-E4-UC-8RE1(P)		4	40 (unshielded)
EASY-E4-AC-16RE1(P)	115/230 V AC	8	40 (unshielded)
EASY-E4-AC-8RE1(P)		4	40 (unshielded)

### **1.10.1.2 Analog inputs**

On the EASY-E4-DC-6AE1(P) expansion, 4 analog input signals are available with a maximum line length of 10 m (shielded).

On the EASY-E4-DC-4PE1(P) expansion with temperature recording, 4 analog input signals are available with a maximum line length of 30 m (unshielded).

### **1.10.2 Length of signal analog output lines**

On the EASY-E4-DC-6AE1(P) expansion, 2 analog output signals are available with a line length of less than 10 m (shielded).

## 1. Description of easyE4 control relay

### 1.10 Engineering

#### 1.10.3 Notes for connection of EASY-E4-AC-... devices

##### 1.10.3.1 Connect digital AC inputs



#### CAUTION

Connect the inputs on EASY-E4-AC-... devices according to the VDE, IEC, UL and CSA safety requirements. The same phase conductor to which the device power supply is connected should be used for the supply of the inputs. EASY-E4-... Otherwise, the device will not detect the switching level or may be destroyed by overvoltage.

For the inputs I5-I8 of the EASY-E4-AC-16RE1(P) expansion devices, one of the other two phases can also be used.

During wiring, make sure to meet all cable protection requirements. → Section "Cable protection", page 68.

#### Input signal voltage range

- OFF signal: 0 to 40 V
- ON signal: 79 V to 264 V

#### Input current

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

**In addition, the following applies to I1–I6 on base devices and for the AC extensions:**

On longer wires, you connect a diode (e.g. 1N4007, 1A) with minimum 1000 V reverse voltage in series to the device input. Make sure that the diode points to the input, i.e. that the cathode of the diode is connected to the input, otherwise the device will not recognize state 1.

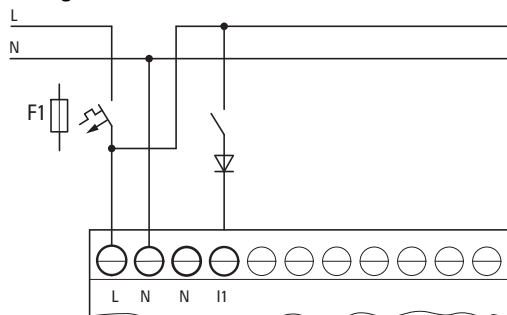


Fig. 4: AC input with suppression diode easyE4 AC

Alternatively, ballast M22-XLED-T (item no. 231079) can be used as a diode.



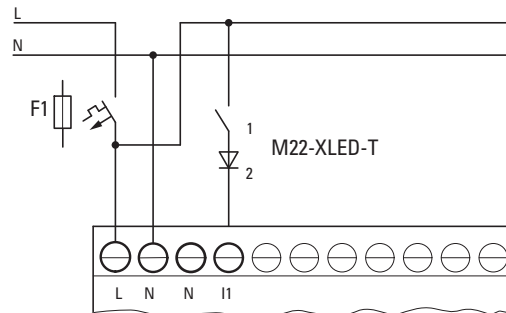


Fig. 5: AC input with ballast M22-XLED-T

### Switching the I7/I8 inputs on AC base devices

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



#### **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.

For all inputs - except for the high current inputs I7, I8 on the base device - the following applies:

The following input circuit can be used in order to prevent interference and also when using two-wire proximity switches:

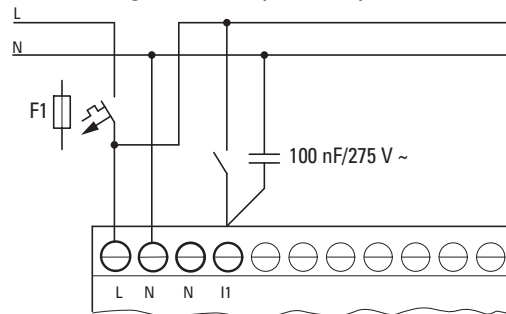


Fig. 6: Increased input current with X2 safety capacitor

- When an X2 safety condenser is in the circuit, the decay time of the input extends, from 100 nF by 75 (45) ms at 230V (115V).
- The current is increased by 6 mA at 230V/50 Hz and by 4 mA at 115V/60 Hz.

To limit the engagement current, you can include a resistor in series in the circuit.

## 1. Description of easyE4 control relay

### 1.10 Engineering

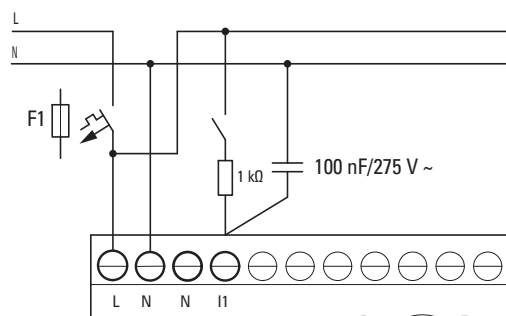


Fig. 7: Limitation of the input current through resistors

As an alternative to the condenser, you can use the M22-XLED230-T ballast (item no. 231080). It includes a 150 nF condenser in series with a 2k resistor and this increases the current by 9.9 mA at 230 V/50 Hz and by 6.5 mA at 115 V/60 Hz. The decay time of the input increases by 140 (70) ms at 230 (115) V.

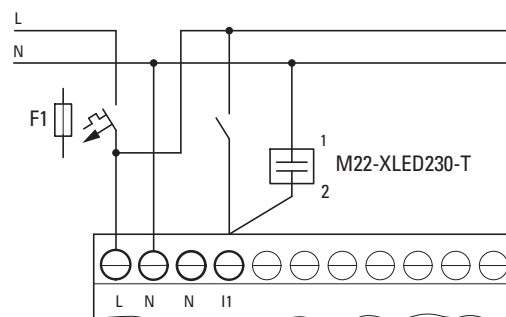


Fig. 8: An increase in input current with M22-XLED230-T

For the M22-XLED-T as well as for the M22-XLED230-T, adapter clip M22-TC (item no.216398) can be used for installation on a DIN rail.

#### 1.10.4 Analog Signals



##### **DANGER**

Analog signals are more sensitive to interference than digital signals, which is why 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 the signal wires as short as possible.  
→ Section "Length of Signal input cables", page 46
- ▶ 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. Description of easyE4 control relay

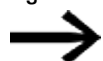
### 1.10 Engineering

#### 1.10.5 Notes on connecting an easy communication module

EASY-COM-... easy communication modules can be used with an easyE4 base device starting with generation 05.

(ID on the nameplate, → page 36)

The easy communication module needs to be connected to the left side of the easyE4 base device, while an I/O expansion for easyE4 control relays is connected to the right side.



It may be necessary to update the firmware on the easyE4 base device before use.



Only available on firmware version 1.30 or higher.

Each easyE4 base device can only support one easy communication module.

The easy communication modules are configured in easySoft 8.

#### Special details regarding SmartWire-DT

easySoft 8 offers planning and ordering help that can be used to configure the SmartWire-DT line.

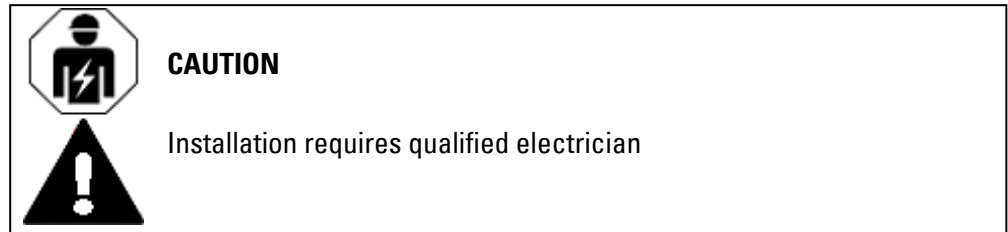
This SmartWire-DT planning and ordering help will assist you with selecting and configuring the SmartWire-DT modules on the SmartWire-DT line. The program has the current consumption specifications for all SmartWire-DT modules. During planning, it will automatically calculate and display the corresponding system's current consumption.

To set up a SmartWire-DT line and install and run easyE4 as a SmartWire-DT coordinator, you will need to be familiar with the basic contents in the SmartWire-DT documents.



The inputs and outputs on an SWD line are available alongside the inputs and outputs for I/O expansion for control relay easyE4. The limitation lies in the number of operands used in the \*.e80 project.

## 2. Installation

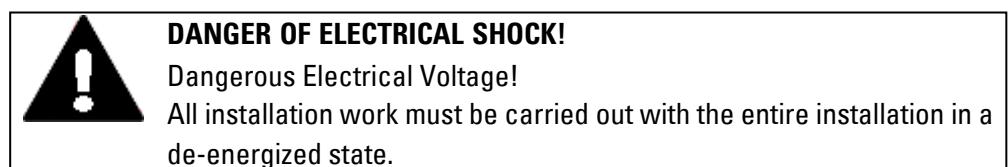


easyE4 devices must be installed and wired exclusively by an electrician or a person who

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. Assemble the base device and the easy communication module together into a block (optional)
4. Connecting the power supply
5. Connecting inputs
6. Connecting outputs
7. Connect to Ethernet



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

## 2. Installation

### 2.1 Prerequisites for the location of use

- 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 36 and

the specifications in the → Section "Technical data", page 883 for the individual devices → page 883



#### **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.

## 2. Installation

### 2.1 Prerequisites for the location of use

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) The display is readable between $\theta$ -5°C (-23°F) $\leq T \leq$ 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 free space around the easyE4 measures: a, b, c  $\geq$  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:  $\theta$  -25°C (-13°F)  $\leq T \leq$  55°C (131°F)  
The display (option) will be readable between  $\theta$  -5°C (-23°F)  $\leq T \leq$  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. Installation

### 2.2 Unpacking and checking the equipment supplied

#### 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:

Tab. 3: easyE4 control relay std. pack

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

Tab. 4: easyE4 control relay I/O expansion std. pack

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

Tab. 5: easy communication module std. pack EASY-COM-SWD-...

Unit	Description
1 x	EASY-COM-SWD-C1(P)
1 x	Bus connector plug
1 x	Installation instructions IL050024ZU

Tab. 6: easy communication module std. pack EASY-COM-RTU-...

Unit	Description
1 x	EASY-COM-RTU-M1(P)
1 x	Bus connector plug
1 x	Installation instructions IL050035ZU

The easyE4 series 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.



### 2.2 Unpacking and checking the equipment supplied

#### **Missing parts or damage**

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

## 2. Installation

### 2.3 Mounting

### 2.3 Mounting

#### *ATTENTION*

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/ad-ded.

- ▶ Check to make sure that the installation clearances are being observed  
→ Section " Installation position", page 54
- ▶ 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 the easyE4 control relay in a control panel, service distribution board, or enclosure so that the power supply and terminal connections cannot be touched directly during operation.

The easyE4 control relay 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.

## 2. Installation

### 2.3 Mounting

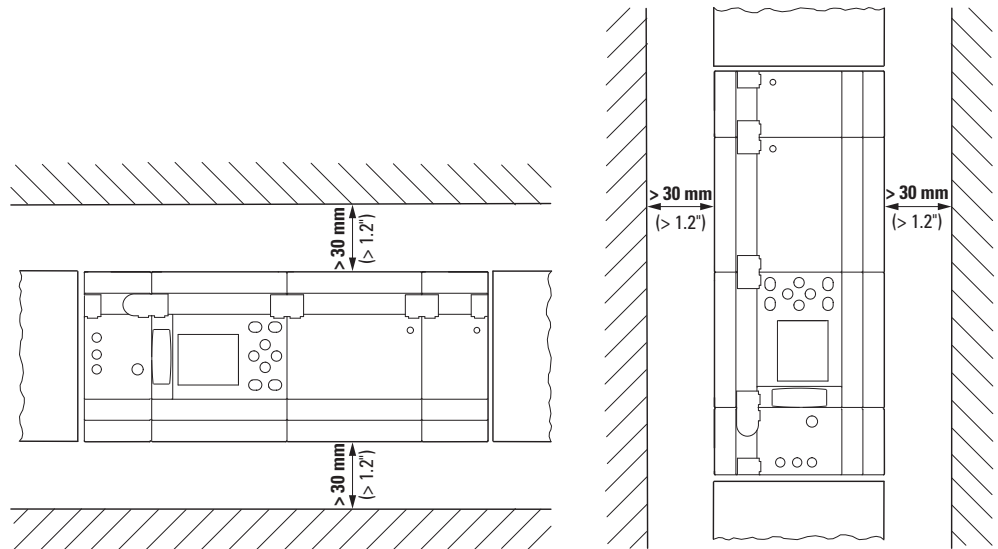


Fig. 9: Min. clearance of 3 cm

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

## 2. Installation

### 2.3 Mounting

#### I/O expansion for easyE4 control relays

Local expansion devices are connected directly to the right of the basic unit.

You can use the 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 matching 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 a connector.

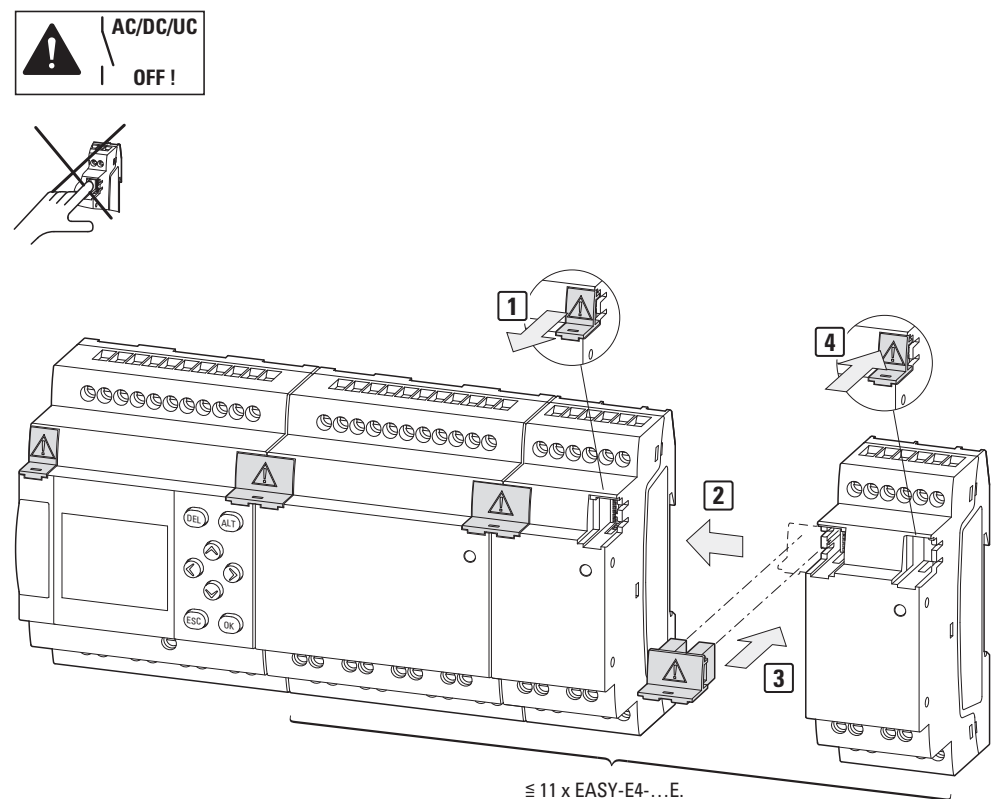


Fig. 10: Assembling a base device with expansions

#### easy communication modules for easyE4 control relays

The easy communication module needs to be placed directly next to the base device, on the microSD side.

You can use the connector to connect the easyE4 base device to an easy communication module and assemble them into a single device block.

EASY-COM-... comes with an matching connector as standard.

By using the easy communication module, you can:

- Connect easyE4 base devices from generation 05 or higher directly to a communication system

The easy communication module 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 easy communication module with the connector.

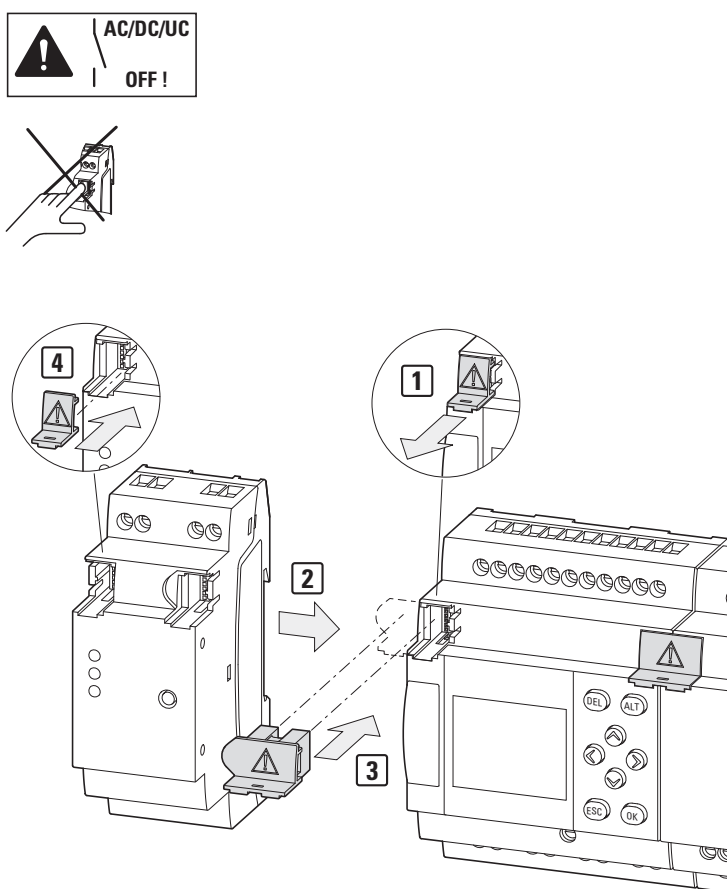


Fig. 11: Assembling the base device with an easy communication module as example EASY-COM-SWD-C1

## 2. Installation

### 2.3 Mounting

#### 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.

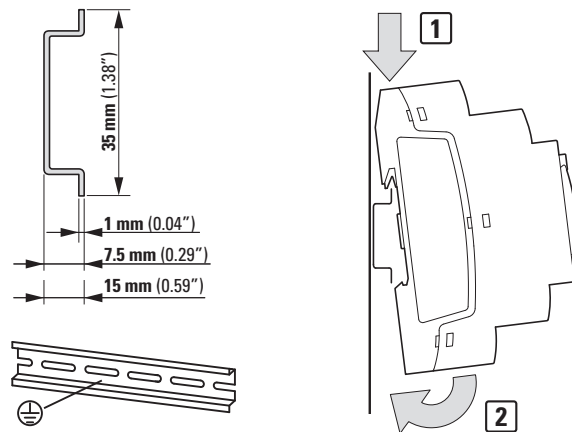


Fig. 12: Installation on IEC/EN 60715 mounting rail

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 matching connector.
5. Repeat for all additional expansions – up to 11 EASY-E4-...-...E1(P)

#### **Installing the easy communication module (optional)**

1. Position the easy communication module to the left of the base device against the mounting rail's upper lip in an inclined position.
2. Slide the easy communication module 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 easy communication module with the matching connector.

#### **Finish assembly**

1. Take the base device's end cover and install it on the right side of the last expansion.
2. Take the base device's end cover and install it on the left side of the easy communication module.

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

- Basic isolation, 400 V<sub>AC</sub> (+10%).
- Safe isolation, 240 V<sub>AC</sub> (+10%).

The base device, expansion device, and easy communication module can be powered with different power supplies.

## 2. Installation

### 2.3 Mounting

#### 2.3.1.2 Screw mounting

Fixing brackets ZB4-101-GF1 that can be inserted on the rear of the easyE4 device are required for screw mounting.

These feet are available as an accessory – please refer to → Section "Accessory devices", page 34.

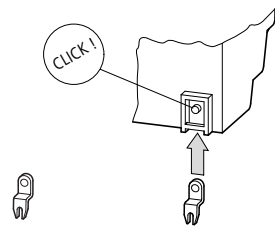


Fig. 13: Inserting a fastening bracket.

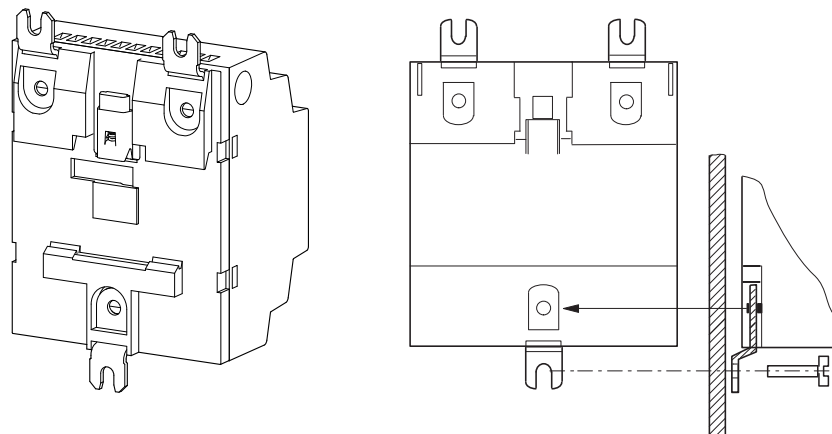


Fig. 14: Example: Screw mounting for a 4U device



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 and/or an easy communication module, remove the connectors.

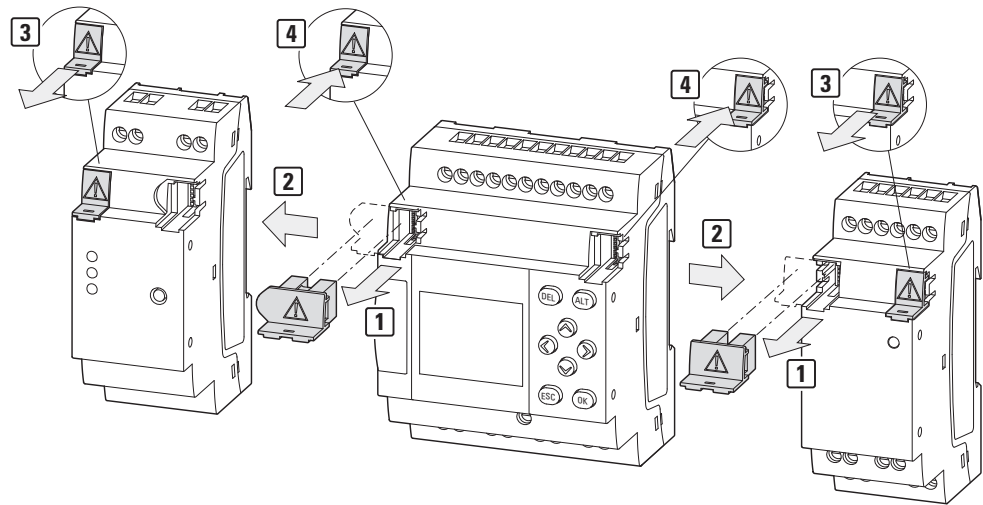


Fig. 15: Remove adjacent connectors

- ▶ Remove the device from the mounting rail

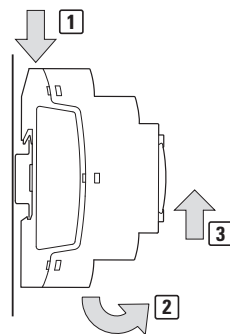


Fig. 16: Dismantling

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

2. Installation

2.4 Connection terminals

2.4 Connection terminals

All devices are available with two terminal types.

The specific terminal type can be found at the end of the part No.→ page 33

You will need a flat-blade screwdriver for installation:

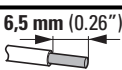





- with terminal type screw terminals  
Slot-head screwdriver, width 0.8 mm x 3.5 mm.
- with push-in terminals  
Slot-head screwdriver, width 0.4 mm x 2.5 mm.

2.4.1 Screw terminals

EASY-E4-...-12...C1, EASY-E4-...-12...CX1, EASY-E4-...-...E1, and EASY-COM-...-.1 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.

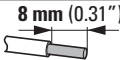
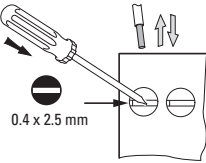


		<b>Terminal capacity in mm²</b>	
	solid	0.2 up to 4	
	Flexible	0.2 to 2.5	
	Conductor cross section AWG	min 22 - max 12	
	Solid cable with ferrule	0.2 up to 2.5	
	Flexible with ferrule		

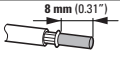
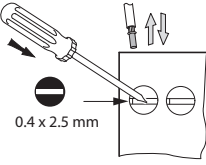


### 2.4.2 Push-in terminals

EASY-E4-...-12...C1P, EASY-E4-...-12...CX1P, EASY-E4-...-...E1P and EASY-COM-...-.1P 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.

		<b>Terminal capacity in mm<sup>2</sup></b>	
	solid	0.2 up to 2.5	
	Flexible		
Conductor cross section AWG		min 24 - max 14	

		<b>Terminal capacity in mm<sup>2</sup></b>	
	Solid cable with ferrule	0.25 up to 1.5	
	Flexible with ferrule		

## 2. Installation

### 2.4 Connection terminals

#### 2.4.3 Connecting the power supply

##### Cable protection

###### *NOTICE*

Make sure to meet all relevant line protection requirements!

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, expansion device(s), and easy communication module that takes into account the amount - a maximum of 11 and the terminal - with UC, DC, and AC power.

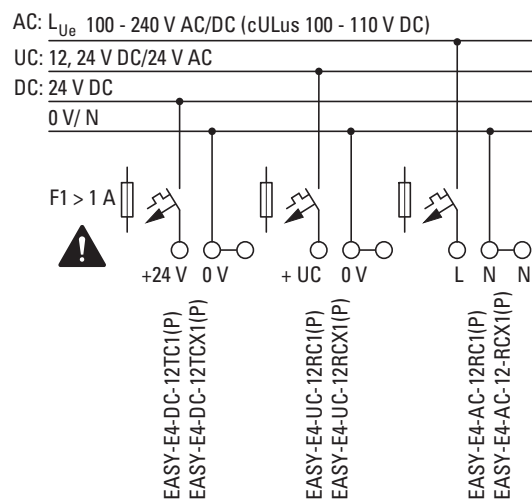


Fig. 17: Connecting the power supply for base devices

## 2. Installation

### 2.4 Connection terminals

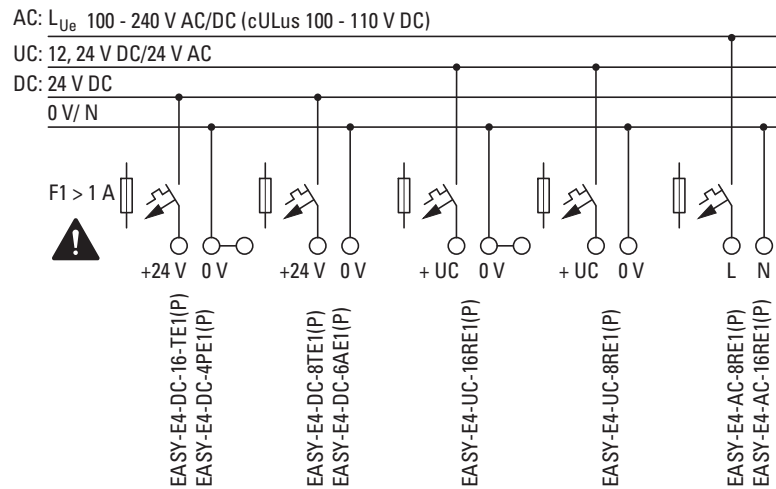


Fig. 18: Connecting the power supply for expansions



The way in which the power supply should be connected to the easy communication module is described in the corresponding section:  
 EASY-COM-SWD-...

→ Section "Connecting the power supply through POW/AUX", page 773  
 EASY-COM-RTU-... → Section "Connecting the power supply", page 787

#### 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 883

## 2. Installation

### 2.4 Connection terminals

#### 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.

##### **See also**

→ Section "Notes for connection of EASY-E4-AC-... devices", page 48

#### 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.

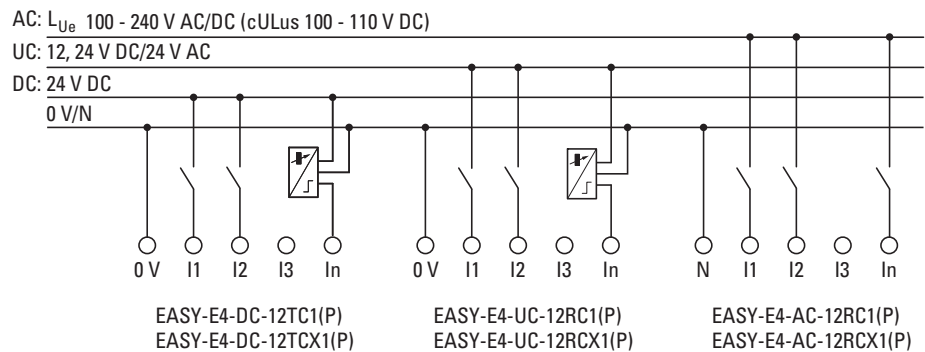


Fig. 19: Connecting the digital inputs on base devices

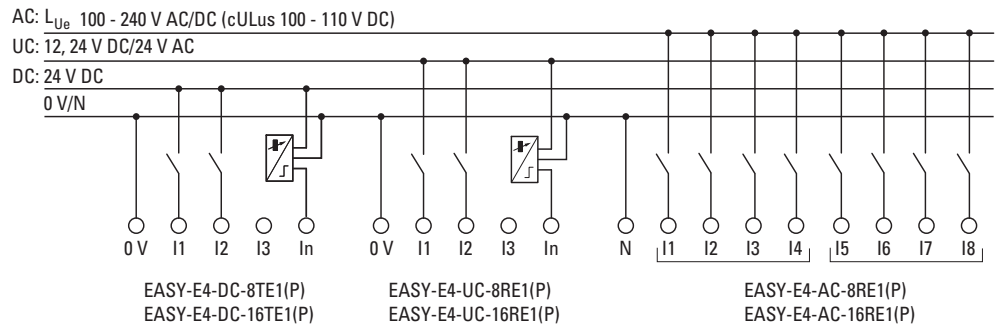


Fig. 20: Connecting the digital inputs on expansions

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.

#### See also

→ Section "Connect digital AC inputs", page 48

## 2. Installation

### 2.4 Connection terminals

#### 2.4.4.1 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.

Tab. 7: AC phase assignment

		EASY-E4-AC-12RC1(P), EASY-E4-AC-12RC1, EASY-E4-AC-8RE1(P)	EASY-E4-AC-16RE1(P)	
L <sub>Ue</sub>	N <sub>Ue</sub>	I1-I8	I1-I4	I5 - I 8
L1	N	L1	L1	L1
L1		L1	L1	L2
L1		L1	L1	L3
L2	N	L2	L2	L2
L2		L2	L2	L1
L2		L2	L2	L3
L3	N	L3	L3	L3
L3		L3	L3	L1
L3		L3	L3	L2

#### Example showing how to read the table

L <sub>Ue</sub>	N <sub>Ue</sub>	I1-I8	I1-I4	I5-I8
L1	N	L1	L1	L1
L1		L1	L1	L2
L1		L1	L1	L3
L2	N	L2	L2	L2
L2		L2	L2	L1
L2		L2	L2	L3
L3	N	L3	L3	L3
L3		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.



### 2.4.4.2 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

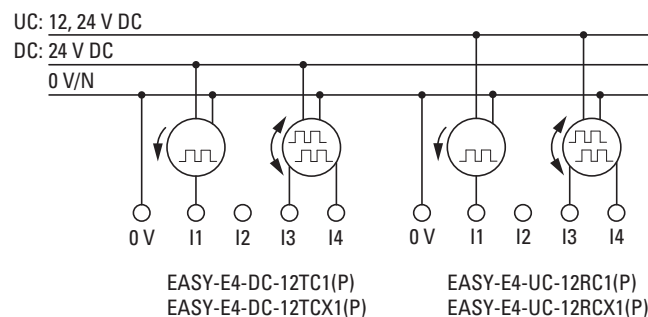


Fig. 21: Connect digital counter inputs



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. Installation

### 2.4 Connection terminals

#### 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 analog inputs' input impedance is 13.3 k $\Omega$ .

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.

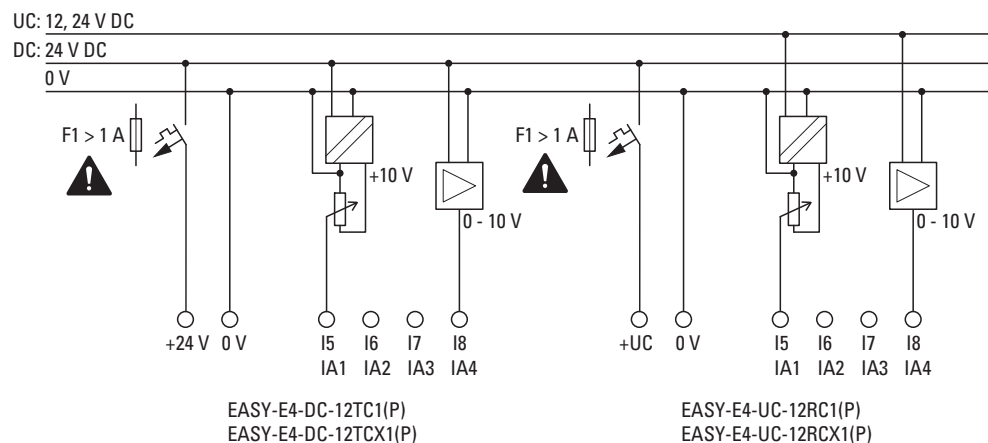


Fig. 22: Connecting the analog inputs on base devices



SETPOINT encoder:

Use a potentiometer with a resistance  $\leq 1$  k $\Omega$ , e.g., 1 k $\Omega$ , 0.25 W.



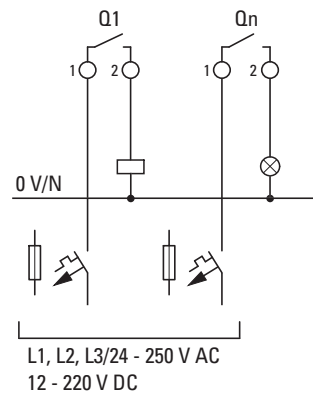
#### **DANGER**

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  $\rightarrow$  Section "Analog Signals", page 51

### 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(P) EASY-E4-UC-8RE1(P)  
EASY-E4-UC-12RCX1(P) EASY-E4-UC-16RE1(P)  
EASY-E4-AC-12RC1(P) EASY-E4-AC-8RE1(P)  
EASY-E4-AC-12RCX1(P) EASY-E4-AC-16RE1(P)

Fig. 23: Connecting relay outputs



#### **DANGER**

Make sure to observe the technical data for the relays.

Do not exceed the upper voltage limit of 250 V<sub>AC</sub> 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. Installation

### 2.4 Connection terminals

#### 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.

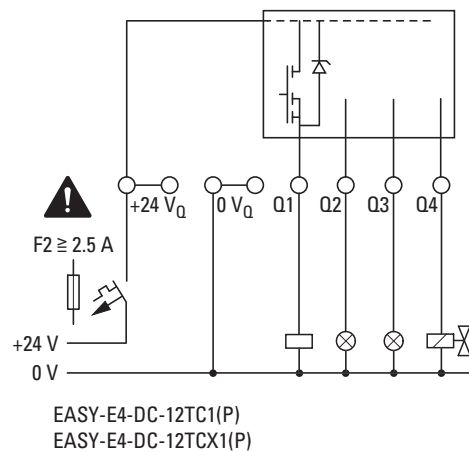


Fig. 24: Connecting base device transistor outputs

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.

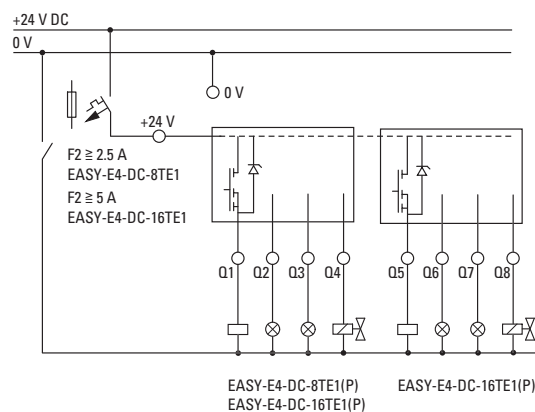


Fig. 25: Connecting expansion device transistor outputs



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 this and prevent electronic components from overheating in the worst-case scenario.



Depending on the actual inductive load ( $I$ ,  $L$ ):

If the +24 V<sub>DC</sub> 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.

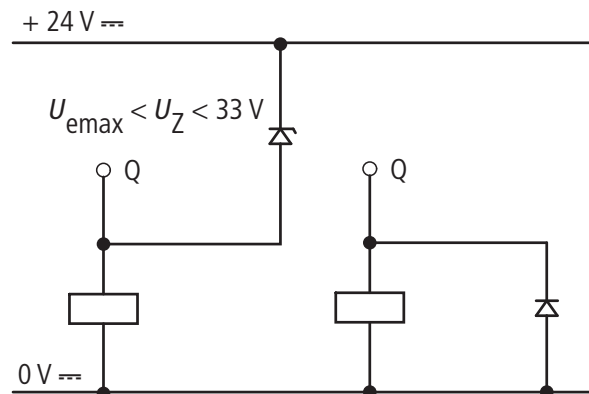


Fig. 26: Inductive load with suppressor circuit

#### 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 simultaneously 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. Installation

### 2.4 Connection terminals

#### 2.4.8 Connecting analog I/O expansion devices

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 8 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 bits	0 - 4095
4 – 20 mA	12 bits	819 - 4095
0 – 20 mA	12 bits	0 - 4095

For all analog inputs, you can configure noise suppression, averaging, and an update rate in easySoft 8.

*Project view*

Expansion information | **Expansion parameter** | Assigned operands

**EASY-E4-DC-6AE1 - (Expansion, 4 IA, 2 QA, screw terminals)**

**Parameter for IA1**  
 Measuring range: 0...10 V  
 Scaling: None (0..4095)

**Parameter for IA2**  
 Measuring range: 0...20 mA  
 Scaling: 0,1 mA (0..200)

**Parameter for IA3**  
 Measuring range: 0...20 mA  
 Scaling: 10 µA (0..2000)

**Parameter for IA4**  
 Measuring range: 0...10 V  
 Scaling: 10 mV (0..1000)

**Parameter for QA1**  
 Output range: 0...10 V  
 Scaling: None (0..4095)

**Parameter for QA2**  
 Output range: 0...10 V  
 Scaling: 10 mV (0..1000)

**Settings for all inputs**  
 Averaging: None  
 Update rate: 50 ms

Fig. 27: Device parameters tab, using the EASY-E4-DC-6AE1 as an example

## 2. Installation

### 2.4 Connection terminals

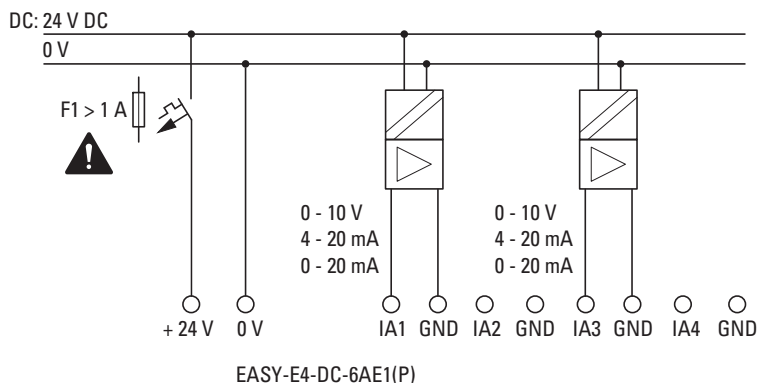


Fig. 28: Connecting analog inputs EASY-E4-DC-6AE1(P)

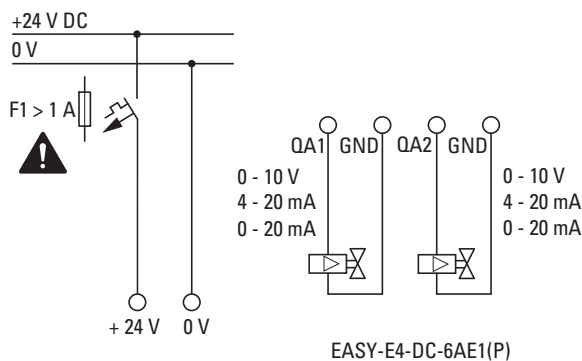


Fig. 29: Connecting analog outputs EASY-E4-DC-6AE1(P)



#### DANGER

Analog signals are more sensitive to interference than digital signals, which is why 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 51

In addition to the specifications in the data sheet, the following applies to EASY-E4-DC-6AE1(P):

Input impedance		
	Voltage:	12,122 kΩ
	Current:	≤ 300 Ω
Voltage output:	Max. current:	10 mA (load resistance ≥ 1000 Ω)
Current output:	Load resistance	≤ 600 Ω

## 2. Installation

### 2.4 Connection terminals

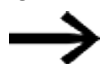
#### 2.4.9 Connecting analog inputs with temperature measuring on expansion devices

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 to 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.

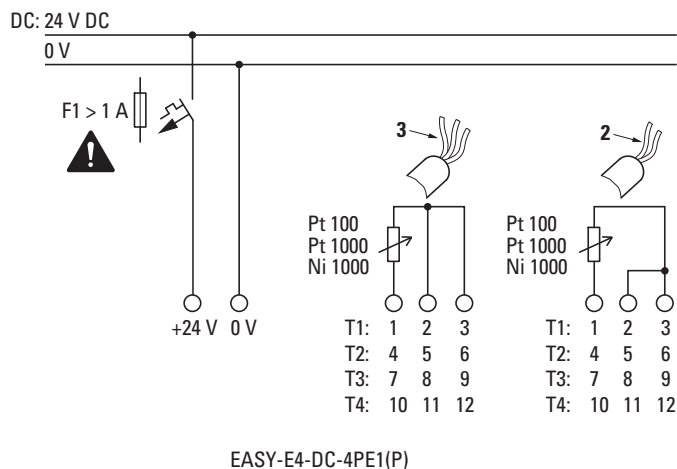


Fig. 30: Connecting analog inputs EASY-E4-DC-4PE1(P)

**DANGER**

Analog signals are more sensitive to interference than digital signals, so 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 51

easySoft 8 is required in order to be able to configure the connected RTD sensors.



#### Project view

Expansion information | Expansion parameter | Assigned operands

**EASY-E4-DC-4PE1 - (Input/output expansion, 24 V DC, 4AI, 4 analog temperature inputs, screw terminal)**

**Module settings**

Optional expansion: ☒

**Settings for individual inputs**

T1: Pt100 (-100°C...800°C / -148°F...1472°F) ▼

T2: Pt100 (-100°C...200°C / -148°F...392°F) ▼

T3: Ni1000 (-50°C...100°C / -58°F...212°F) ▼

T4: Not defined ▼

**Settings for all inputs**

Scaling: °C (view: 1/10 °) ▼

Averaging: None ▼

Fig. 31: 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 range °C
1	Pt100 / Pt1000	-100 – +200 (-148 – +392°F)
2	Pt100 / Pt1000	-100 – +400 (-148 – +752°F)
3	Pt100 / Pt1000	-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

## 2. Installation

### 2.4 Connection terminals

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:

- None (no formation of mean values)
- Weak (mean value formation over 4 measuring cycles)
- Moderate (mean value formation over 8 measuring cycles)
- Strong (mean value formation over 16 measuring cycles)



There is a description of the formation of mean values implemented in the AV function block → Section "Temperature averaging example", page 348

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.

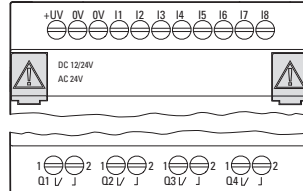
Designation	Event
DIAG	General diagnostic indicating that a diagnostic event is present
DIAG 1	Configured measuring range exceeded at at least one temperature input, or connection cable discontinuity
DIAG 2	Configured measuring range fallen below at at least one temperature input, or a short-circuit 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

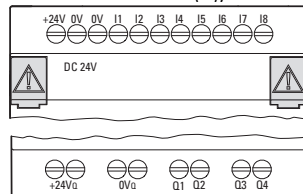
#### Base devices

##### EASY-E4-UC-12RC1(P), EASY-E4-UC-12RCX1(P)



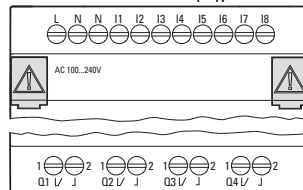
Power Supply	+UC	0V	0V								
Input				I1	I2	I3	I4	I5	I6	I7	I8
Output				Q1/1	Q1/2	Q2/1	Q2/2	Q3/1	Q3/2	Q4/1	Q4/2

##### 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

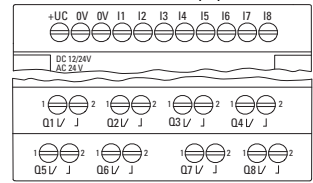
2. Installation

2.4 Connection terminals

Expansions

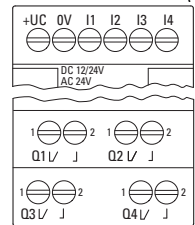
UC input expansions with relay outputs

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

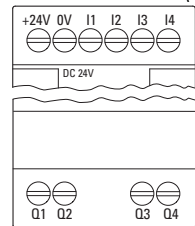
EASY-E4-UC-8RE1(P)



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)

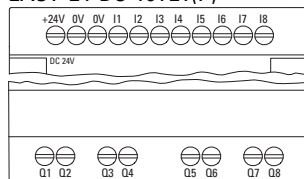


Power Supply	+24V	0V				
Input			I1	I2	I3	I4
Output			Q1	Q2	Q3	Q4

## 2. Installation

### 2.4 Connection terminals

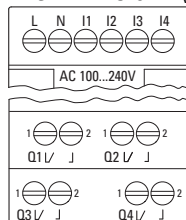
#### EASY-E4-DC-16TE1(P)



Power Supply	+24V	0V	0V									
Input				I1	I2	I3	I4	I5	I6	I7	I8	
Output				Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	

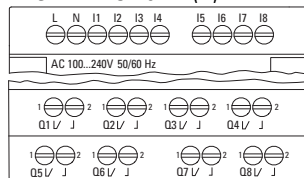
#### AC input expansions with relay outputs

#### EASY-E4-AC-8RE1(P)



Power Supply	L	N				
Input			I1	I2	I3	I4
Output			Q1/1	Q1/2	Q2/1	Q2/2
Output			Q3/1	Q3/2	Q4/1	Q4/2

#### EASY-E4-AC-16RE1(P)



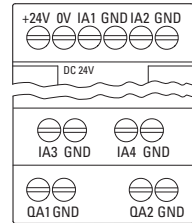
Power Supply	L	N													
Input			I1	I2	I3	I4	I5	I6	I7	I8					
Output			Q1/1	Q1/2	Q2/1	Q2/2	Q3/1	Q3/2	Q4/1	Q4/2					
Output			Q5/1	Q5/2	Q6/1	Q6/2	Q7/1	Q7/2	Q8/1	Q8/2					

2. Installation

2.4 Connection terminals

Analog input expansion

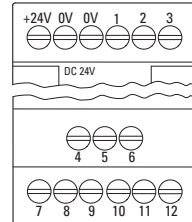
EASY-E4-DC-6AE1(P)



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

easy communication modules for easyE4 control relays

- ➔ The terminal layout for the optional EASY-COM-SWD-... module is described in the easyE4 as a SmartWire-DT coordinator coordinator section,  
→ Section "Terminal layout", page 774
- ➔ The terminal layout for the optional EASY-COM-RTU-... module is described in the easyE4 Communication via Modbus RTU section,  
→ Section "Terminal layout", page 785

## 2.5 External connections on the base device

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

### 2.5.1 External connection layouts

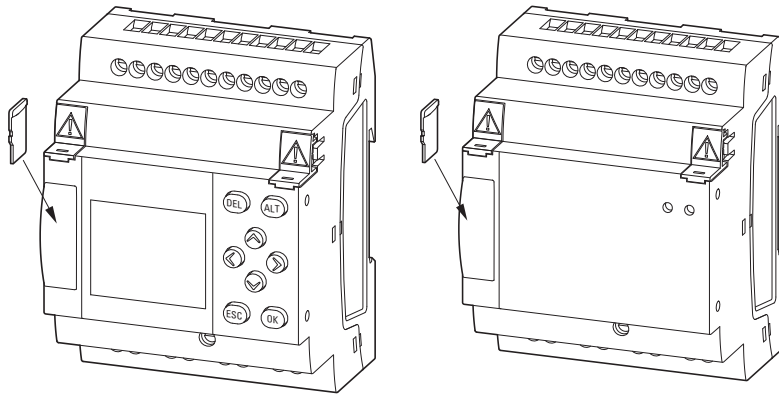


Fig. 32: Slot for microSD

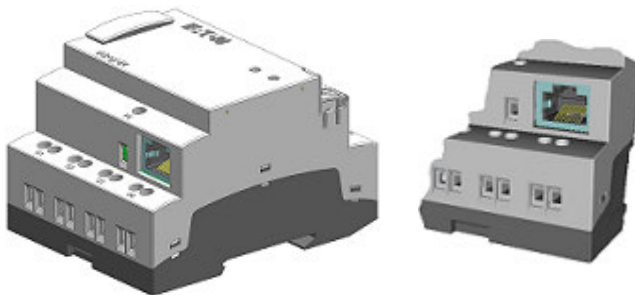


Fig. 33: Ethernet port on base device

## 2. Installation

### 2.5 External connections on the base device

#### 2.5.2 Memory card

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



Do not install or remove microSD memory card while the easyE4 is switched on.

#### 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.

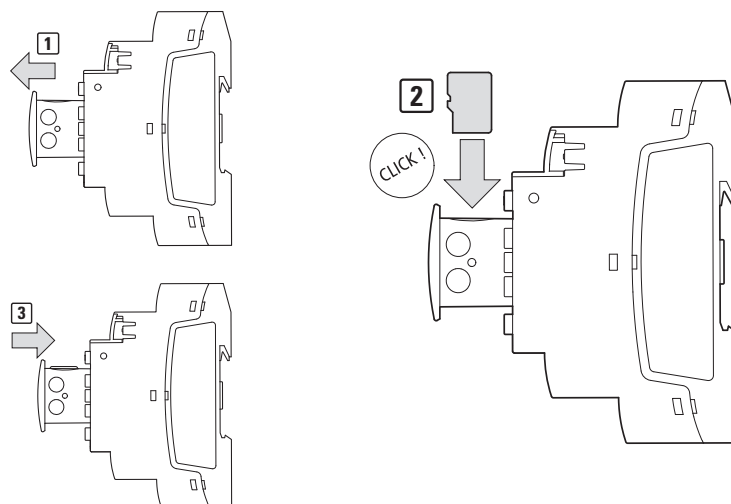


Fig. 34: Inserting a memory card



## 2. Installation

### 2.5 External connections on the base device

#### Removing 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

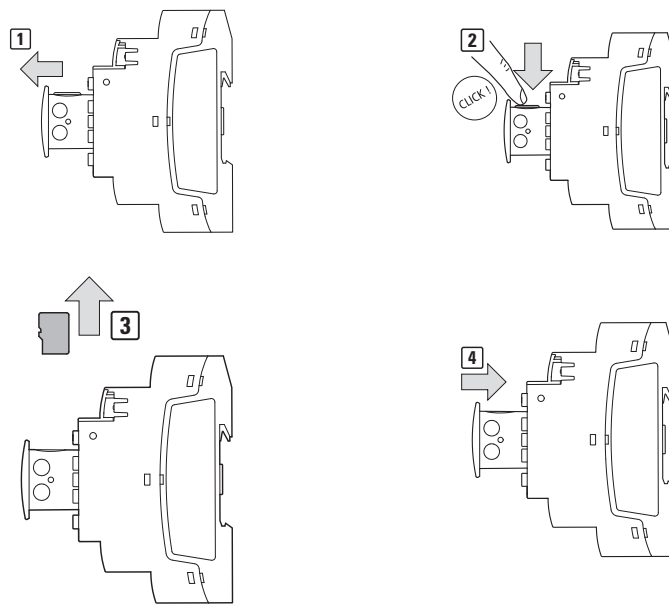


Fig. 35: Removing the memory card

## 2. Installation

### 2.5 External connections on the base device

#### 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.

The easyE4's connectivity is ensured with the aforementioned Ethernet port. Not only does this apply to connectivity directly through easyNET and Modbus TCP, but also to connectivity implemented with communication modules through Modbus RTU and SmartWire-DT, as well as any web connection.



In order to successfully use a web connection, the easyE4's Ethernet connection must always be on.



Fig. 36: 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 640

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 66

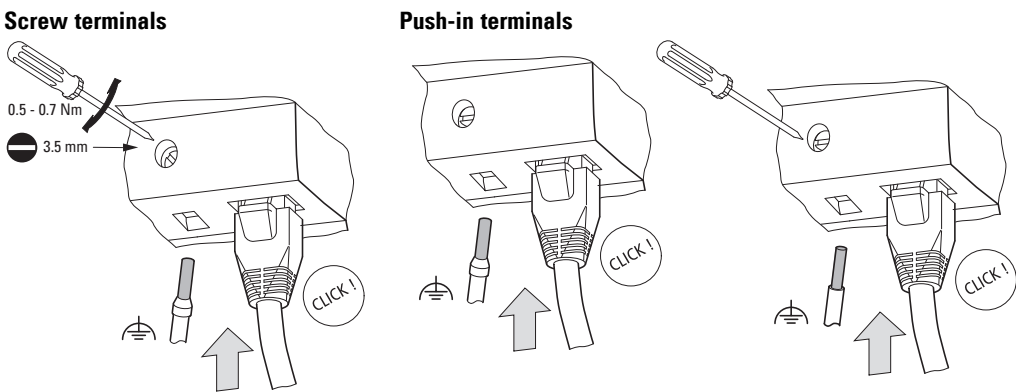
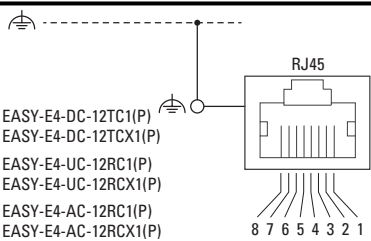


Fig. 37: Connecting the Ethernet cable

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

- ▶ Connect the functional earth
- ▶ Plug in the Ethernet cable

## 2. Installation

### 2.5 External connections on the base device

#### 2.5.3.2 Removing the Ethernet cable

with terminal type screw terminals

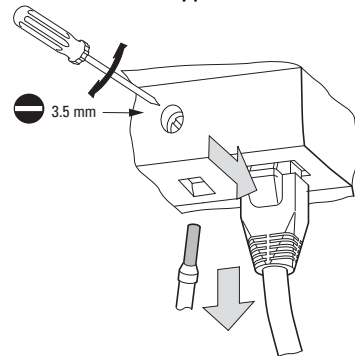


Fig. 38: Removing the Ethernet cable

with push-in terminals

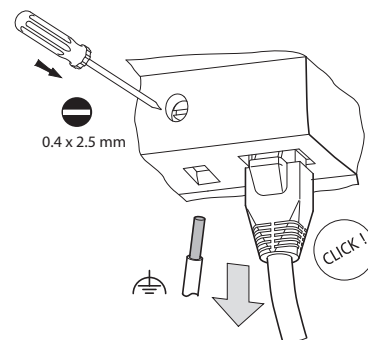


Fig. 39: Removing the Ethernet cable

## 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 versions of easySoft 7 or higher.

The programming software easySoft 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 8 programming software license through your supplier or through the EATON online catalog EASYSOFT-SWLIC, catalog no. 197226.

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. This license key is also valid for all higher versions of easySoft.

### Installation requirements

- An easySoft 7 version or higher
- 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 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. Installation

### 2.6 Programming software license

#### 2.6.1 Licensing

When you purchase the EASYSOFT-SWLIC you acquire a license product certificate for easySoft 8.

This license product certificate bears a 36-digit certificate number.

With this certificate number, you can obtain your 24-digit licence key online.

➔ During the installation process, you will be asked for the 24-digit license key for your easySoft 8.

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.



Fig. 40: license product certificate

#### Getting a license key

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

 [Eaton.com/license](https://Eaton.com/license)

Licensing

Please enter the certificate no. of your license document.

Certificate

Fig. 41: 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.

## 2. Installation

### 2.6 Programming software license

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. Installation

### 2.6 Programming software license

#### 2.6.2 Adding a license key later on

If you installed the demo version of easySoft 8, you can add a valid license key later on in order to unlock the full version.

- ▶ Go to easySoft8 the *? menu* and click on  License.

A dialog box for entering the license key will appear.

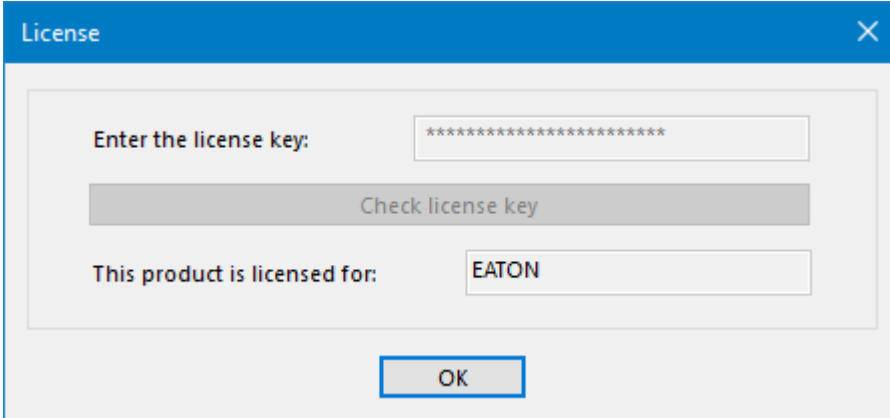
A screenshot of a Windows-style dialog box titled "License" with a blue header bar and a close button (X) in the top right corner. The dialog box has a light gray background. Inside, there is a section with a label "Enter the license key:" followed by a text input field containing 24 asterisks. Below this is a gray button labeled "Check license key". Underneath the button is another label "This product is licensed for:" followed by a text input field containing the word "EATON". At the bottom center of the dialog box is an "OK" button.

Fig. 42: License dialog box

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



#### 2.6.3 Software updates and hardware changes

Once you have licensed the easySoft 8 programming software, you can download the latest 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 8 can check whether there are any updates for the version installed. This requires for the PC to have an active Internet connection.

*Menu?*

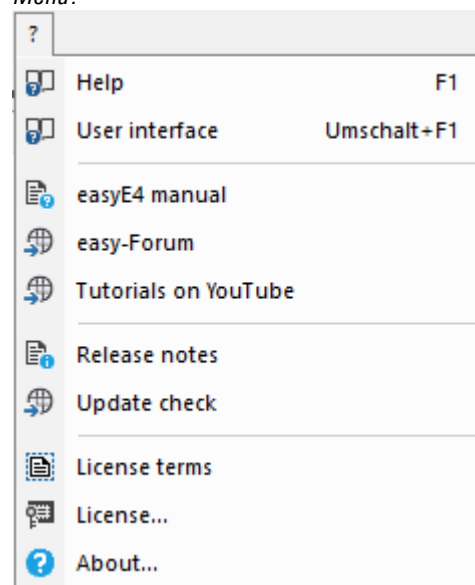


Fig. 43: Options in ? menu

#### 2.6.4 easyE4 root certificate

The easyE4 root certificate together with the C:\Program Files (x86)\Common Files\Eaton\easyRootCA in the target folder is installed from easySoft 8 programming software.

Retrospective installation of this certificate is possible. A user who is not installing the easyE4 Root certificate during the easySoft 8 installation process at this stage can proceed with installation of the certificate at a later date.

##### See also

→ Section "Secure communication with certificates", page 709

[System requirements](#)

## 2. Installation

### 2.6 Programming software license

#### 2.6.5 Installation instructions

Before starting with the installation, close all open applications.

To install easySoft 8, you will need to have local admin privileges for your system.

##### Download

- Download the full version of the easySoft 8 program from the Software Download Center.
- Select the "Software" category, then the easySoft 8 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.

The InstallShield wizard features a maintenance mode that you can use to modify, repair, and uninstall existing software when reinstalling. It also makes it possible to select individual components.

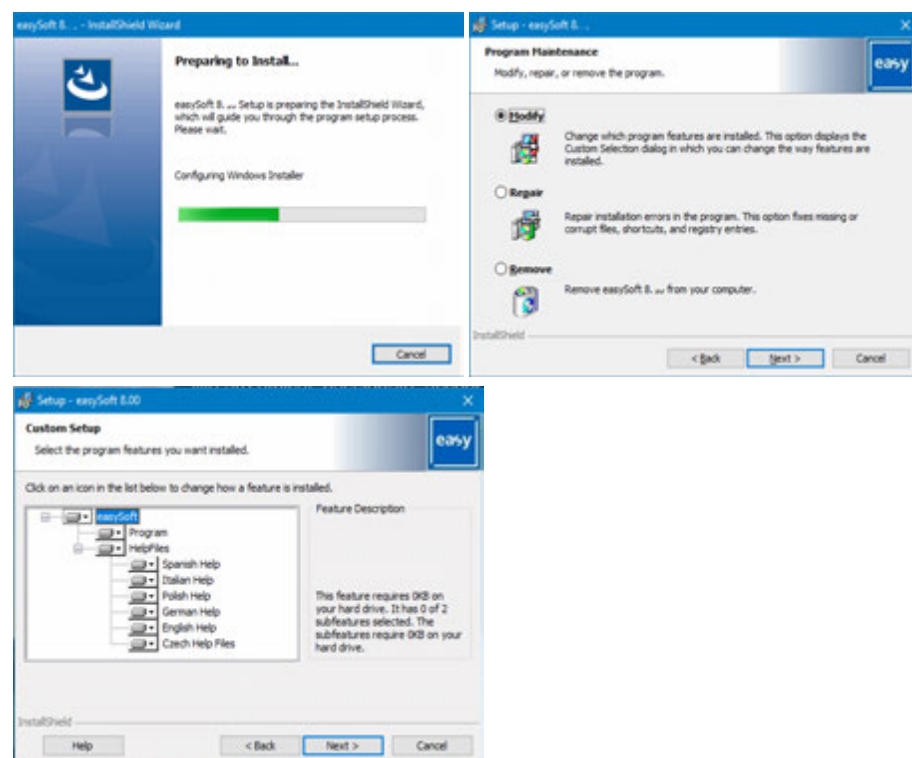


Fig. 44: InstallShield Wizard

## 2. Installation

### 2.6 Programming software license

#### Installing the software for the first time



During the installation process, you will be asked for the 24-digit license key for your easySoft 8. 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.

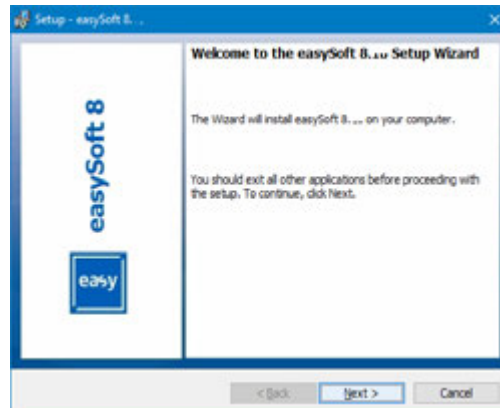


Fig. 45: Step 1

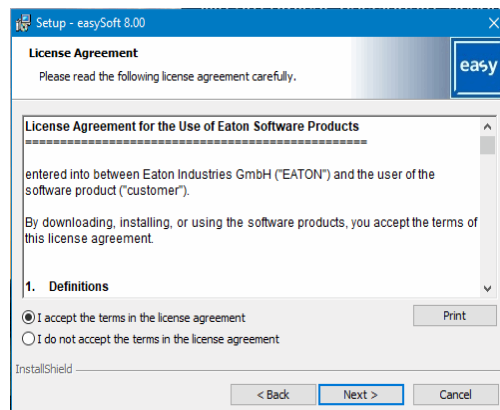


Fig. 46: Step 2 License agreement

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

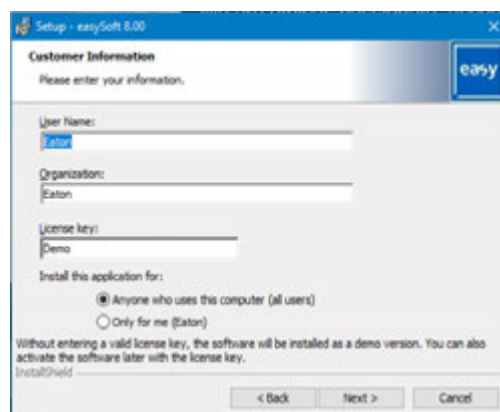


Fig. 47: Step 3 License key

## 2. Installation

### 2.6 Programming software license

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 96.

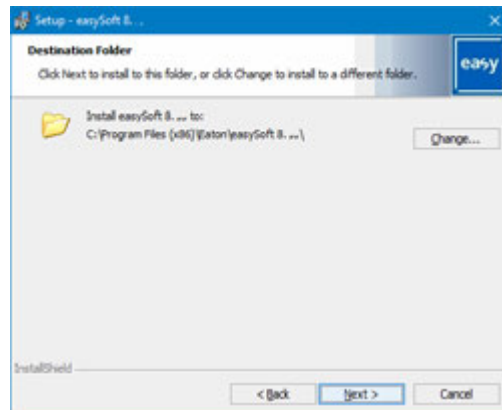


Fig. 48: Step 4 Destination folder

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 8 programming software to be installed.

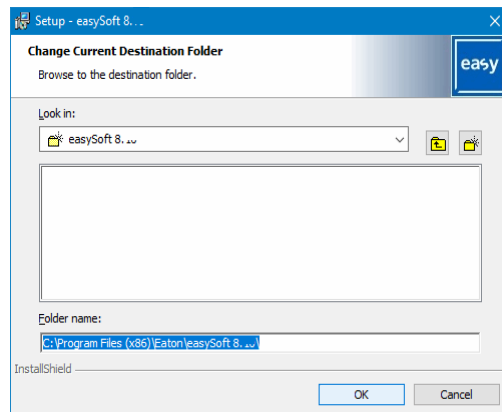


Fig. 49: Step 4.1 Changing the destination folder

## 2. Installation

### 2.6 Programming software license

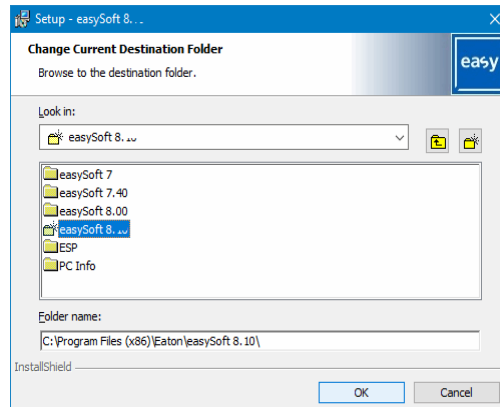


Fig. 50: Step 4.2 Creating your own destination folder

You can now select the installation options you want.

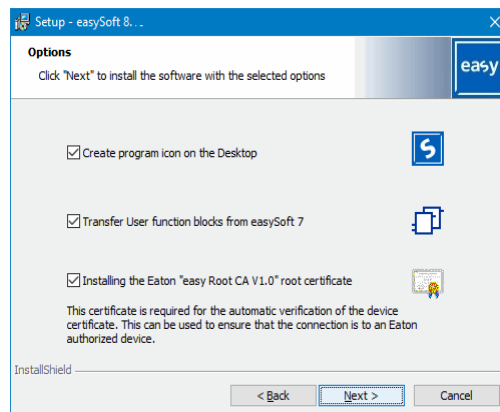


Fig. 51: Step 5 Selecting options

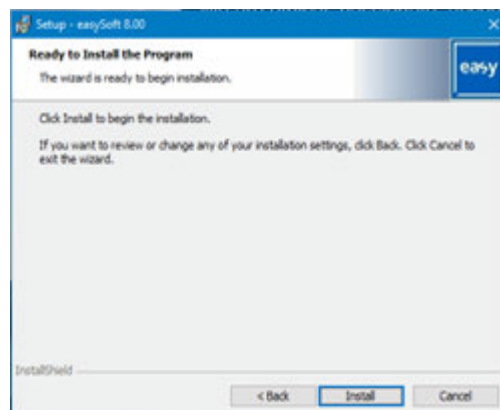


Fig. 52: Step 6 Starting the installation

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

## 2. Installation

### 2.6 Programming software license

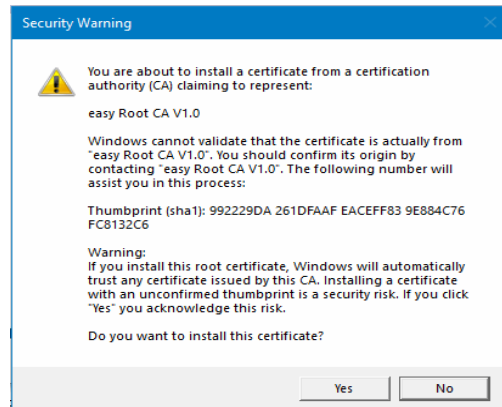


Fig. 53: Step 7 Confirmation prompt

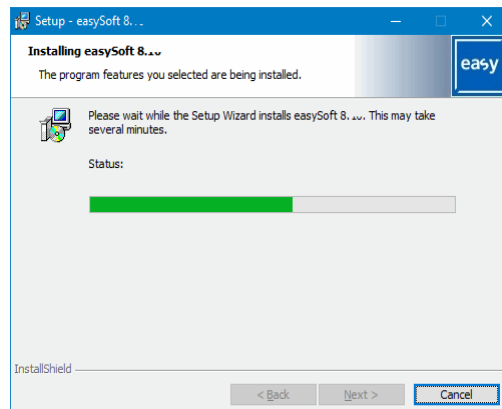


Fig. 54: Step 7 Progress display

Messages regarding the installation will appear.  
Click OK on them.

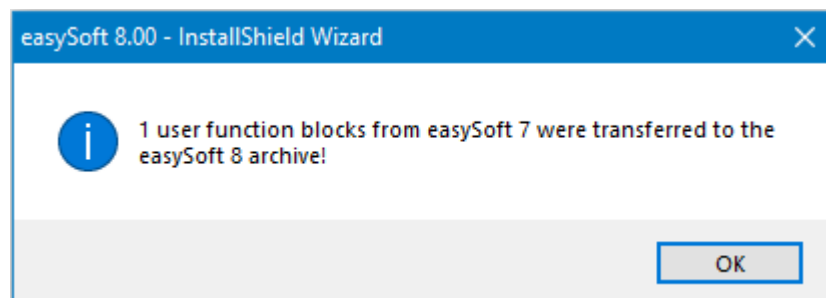


Fig. 55: Step 7.1 Messages



If there are any previously existing user function blocks in the C:\ProgramData\Eaton\easySoft 8\UserFBs folder, they will not be overwritten. A message saying that these user function blocks exist already will appear.

## 2. Installation

### 2.6 Programming software license

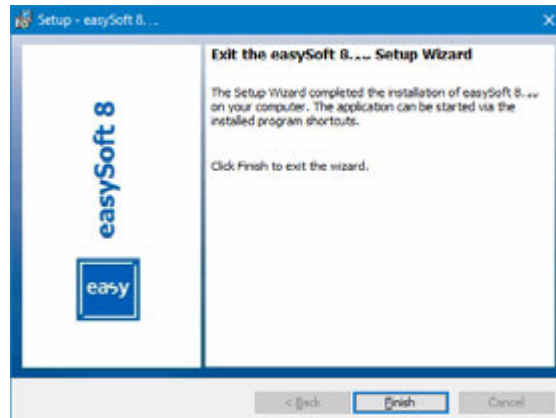


Fig. 56: Step 8 Finishing

The easySoft 8 icon will be added to your Desktop during the installation process.

- Click on the easySoft 8 icon to open easySoft 8.



Fig. 57: easySoft 8 icon depending on the screen resolution or position

## **2. Installation**

### **2.6 Programming software license**



## 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. Do not expose the device to direct thermal radiation from heating appliances.

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 8 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 8.

### 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 644
- ▶ Install the required software package easySoft 8.
- ▶ Transfer program to the easyE4 device
- ➔ The commissioning process for the optional EASY-COM-SWD-... module is described in the easyE4 as a SmartWire-DT coordinator section  
→ Section "Configuring the SWD line", page 777
- ➔ The optional EASY-COM-RTU-... module can only be commissioned with easySoft 8  
→ Section "easyE4 Communication via Modbus RTU", page 782

### 3. Commissioning

#### 3.2 Daily operation

### 3.2 Daily operation

Once the easyE4 device 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 setting can be configured in the device menu.

➔ 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 860

### 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 relay with LED indicators

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

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

If there is an executable program on the control relay easyE4, 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:



- POW/RUN LED or POW/RUN/Status LED
- ETHERNET/NET LED (base device only)

Fig. 58: 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, Flashing, 1 Hz	Supply voltage OK, STOP mode
Green, Flashing, 4 Hz	Fault at one of the expansions, between the easyE4 device and the connector

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

Off	No Ethernet cable connected; supply voltage off 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, 2 flashes, pause, etc.	NET data flow working; one or more NET stations missing
Green, flashing, 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, Flashing, 1 Hz	Supply voltage OK, no data exchange with base device
Green, Flashing, 3 Hz	Supply voltage OK, no data exchange with base device, diagnostic bit will be set, device not working
Green, Flashing, 10 Hz	Device waiting for firmware update
Green, Flashing, 0.5 Hz	Firmware update active



LED signals for the optional EASY-COM-SWD-... module → Section "LED status messages on the EASY-COM-SWD-... communication module", page 778



LED signals for the optional EASY-COM-RTU-... module → Section "LED status messages on EASY-COM-RTU-... communication module", page 789

### 3. Commissioning

#### 3.3 Switch on

##### 3.3.2 Startup behavior of control relay easyE4 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 control relay easyE4, 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 control relay easyE4, 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.

```
I 1..4..78 EOF
NT1 P      DC P-
MO 13:08   ST
Q 1..4     RUN
Device name
167.67.3.1
```

Fig. 59: Example of status display on display

##### 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.

### 3. Commissioning

#### 3.3 Switch on

##### Main menu

```
STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMAT
SYSTEM-OPTIONS
PROGRAM
```

Fig. 60: 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

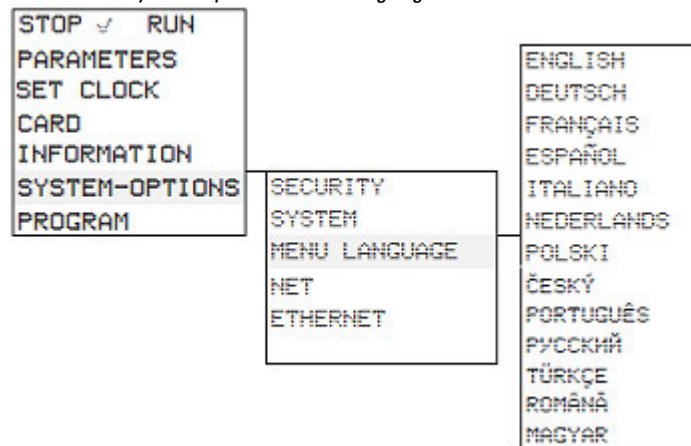


Fig. 61: Menu path in English

- ▶ 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. Commissioning

### 3.3 Switch on

#### 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 8).
- ▶ If there is no program on the base device, load the program you want (with the memory card or easySoft 8) 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 control relay easyE4 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	I 12345678 EOK	1 2 3 4 5 6 7 8
2	RE I NT1 DC P-	ID 1-8: . . . . .
3	WD hh:mm ST	ID 9-16: . . . . .
4	Q 1234 STOP	ID 17-24: . . . . .
5	Device name	
6	IP-Adresse	S T O P

Fig. 62: 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; no Ethernet cable connected; supply voltage off 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
1x <b>ALT</b>	DD-MM-YYY	Displays the device date with the configured format
	ST	Configured startup behavior for the device; nothing displayed – automatic starting is possible

3. Commissioning

3.3 Switch on

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

<b>Status indicator 2</b>	
	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.

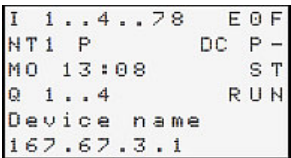


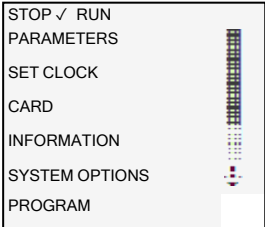
Fig. 63: Example of status display on display

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.

Tab. 8: Main menu



See also

→ Chapter "3 Operation", page 153



### 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 91

easySoft 8 communications make it possible to search for connected easyE4 that are switched on and establish communications with them.

#### 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 VLAN 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



[Eaton.com/cybersecurity](https://Eaton.com/cybersecurity)

#### **See also**

→ Section "Establishing an Ethernet connection", page 185

→ "Establishing an Ethernet connection and transferring a program or visualization project", page 117

## 3. Commissioning

### 3.3 Switch on

#### 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 728
- Section "Modbus TCP", page 830
- Section "Setting up a NET group", page 721
- Section "easyE4 as a SmartWire-DT coordinator", page 770
- Section "easyE4 Communication via Modbus RTU", page 782

### **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.

### 3. Commissioning

#### 3.4 Overview of switch-on behavior

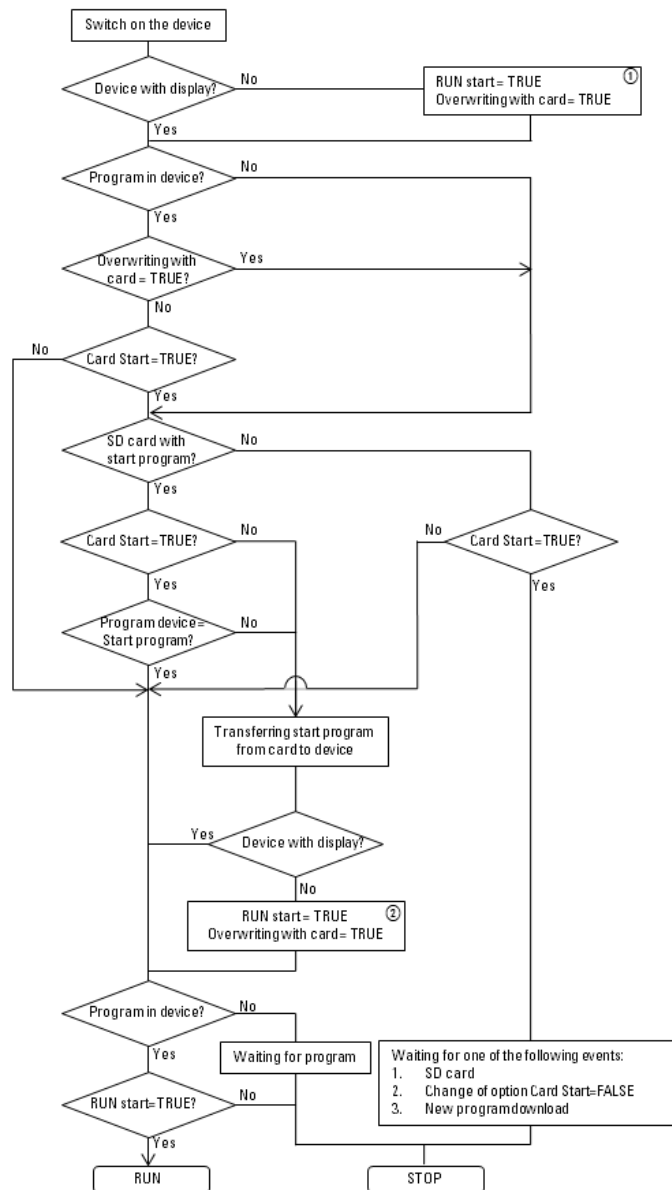


Fig. 64: Startup procedure with device initialization

- ① RUN start: The device should be able to start without easySoft 8  
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 or visualization project****3.5 Establishing an Ethernet connection and transferring a program or visualization project**

To enable access to an easyE4 base device or a easyE RTD Advanced visualization device via programming, an Ethernet connection is available.

**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. You can use any standard switch and Ethernet cable with RJ45 connectors.

You can also use an Ethernet connection to program the individual devices.

**3.5.1 Basic information on assigning IP addresses**

For communication between easyE4 base devices and easyE RTD Advanced visualization devices on an Ethernet network, Internet Protocols (IP) Version 4 IPv4 addresses are used.

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.

### 3. Commissioning

#### 3.5 Establishing an Ethernet connection and transferring a program or visualization project

Tab. 9: Sample addresses for PC

PC	Decimal	Binary	
IP address	192.168.178.100	11000000 10101000 10110010 01100100	
Subnet Mask	255.255.254.0	11111111 11111111 11111110 00000000	AND
Network section	192.168.178.192	11000000 10101000 10110010 00000000	

Tab. 10: Possible easy4 or easyE RTD Advanced IP addresses

easyE4/ easyE RTD Advanced	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.

## 3.5 Establishing an Ethernet connection and transferring a program or visualization project

### Establishing an Ethernet connection

Requirements for access to an easyE4 control relay or a easyE RTD Advanced visualization device:

- The PC must have an Ethernet port that is free and has been configured
  - The Ethernet port on the PC must be on the same subnet as the easyE4 base device and the easyE RTD Advanced visualization device.
  - The devices can be connected to the PC using any commercially available Ethernet cable with RJ45 connector.
  - An Ethernet address was assigned to the easyE4 base device or to the easyE RTD Advanced visualization device, entered either using DHCP, AUTO-IP or manually.
- On easyE4 base devices with a display, make a note of the IP address of the easyE4 base device from the menu path on the *INFORMATION\ACTUAL CONFIG* device and scroll to the IP ADDRESS entry.  
In the case of easyE RTD Advanced visualization devices, check in the device menu.

Continuing as described below is only possible with easySoft 8.

- Open the Communication view in the easySoft 8 programming software.

### Communication view

*Communication\Connection view*

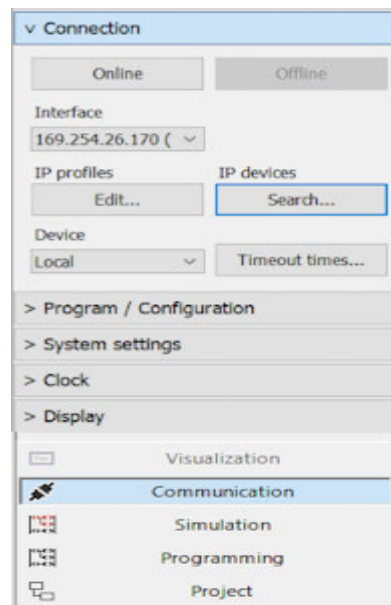


Fig. 65: Establishing an Ethernet connection

### 3. Commissioning

#### 3.5 Establishing an Ethernet connection and transferring a program or visualization project

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

##### Search for devices dialog box

Fig. 66: Search for devices with an IP address

With an existing Ethernet connection, the easyE4 base device and/or the easyE RTD Advanced visualization device is found and gets recorded with its parameters.

- ▶ For each easyE4 base device and/or easyE RTD Advanced visualization device found, use the button to store the IP profile as an IP profile.

##### Search for devices dialog box

Fig. 67: Saving the found device's IP profile

A corresponding message appears, stating that the IP address of the easyE4 base device and the easyE RTD Advanced visualization device have been set up as a new profile.

- ▶ Close the Search for devices dialog box.



#### 3.5 Establishing an Ethernet connection and transferring a program or visualization project

##### Download/Upload – used to transfer a program, visualization file, or Web-Visu Changes in the Interface drop-down menu

The IP address of the easyE4 base device is stored under port together with the easyE RTD Advanced visualization device.

If a connection to several devices was established before this, correspondingly more entries are available. In this instance, select the IP address of the easyE4 base device needed or of the easyE RTD Advanced visualization device at port.

*Communication\Connection view*

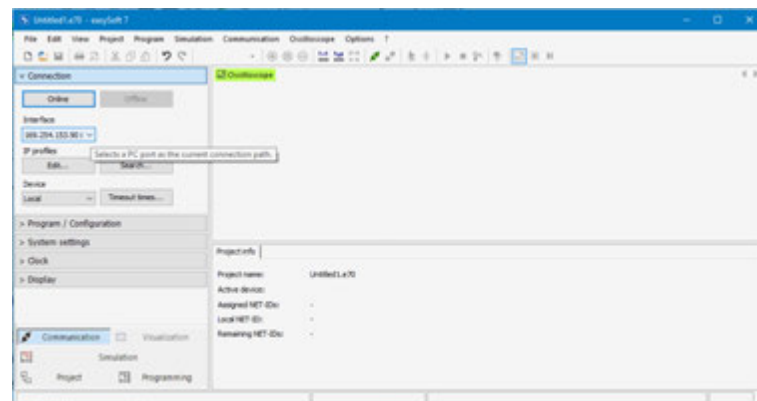


Fig. 68: Selecting the easyE4 device's IP address

- ▶ Click on the **Online** button to establish a connection between your PC and the easyE4 base device or the easyE RTD Advanced visualization 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.  
For easyE RTD Advanced visualization devices, you must enter the Admin password.

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**.

- ▶ Click on **PC => device** in the Program area to transfer your program or visualization.  
easySoft 8 transfers the part of the project relevant to the device.

### 3. Commissioning

#### 3.5 Establishing an Ethernet connection and transferring a program or visualization project

Communication\Connection view

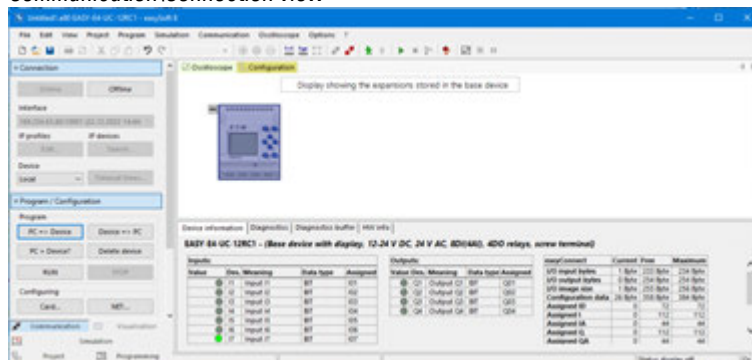


Fig. 69: Connection to the easyE4 device established and program transferred



For additional help working with easySoft 8, read the various help topics in the easySoft 8 help. To access it, press the **F1** key on your keyboard.

#### What gets transferred with each download

If there is a NET group, easySoft 8 can be used to establish a connection to the first NET station and the download for multiple devices can then be initiated at the same time.

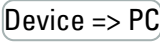
If multiple easyE RTD Advanced visualization devices are involved, a connection must be established to each visualization device one after the other, and the visualization project must be transferred to each respective visualization device.

- **easyE4 base device**  
In the download, the program will be transferred to the selected easyE4 base device together with all settings from the Project view relevant to the device.
- **Web-Visu**  
The visualization for the easyE4 base device that was created with the Web Editor will be transferred together with the program.
- **easyE RTD Advanced visualization device**  
In the download, all information for the visualization will be transferred to the selected easyE RTD Advanced visualization device together with all settings from the Project view relevant to the device.  
Information indicating which easyE4 base devices are involved in the visualization will also be transferred. This makes it possible, with an upload, to create a reconstruction of the project in easySoft 8.

Among other things, during this download, the settings from *Project view/Ethernet tab* are transferred. Depending upon how these are set, it is possible to change the behavior of the Ethernet connection directly after the download. This can lead to disconnection of the device. If another connection is to be established, the aforementioned steps need to be repeated.

## 3.5 Establishing an Ethernet connection and transferring a program or visualization project

### Upload the programs and visualization project files

To rebuild a project, you can upload the \*.e80 program from the easyE4 base device, the visualization project, and the configuration of the devices back to easySoft 8 by clicking on . To do this, you can connect to the first NET station and start the upload for multiple easyE4 base devices simultaneously. To complete this information, you will then have to connect to the relevant easyE RTD Advanced visualization device and upload the visualization project and the device configuration to the PC.

### See also

- Section "Connecting the Ethernet cable", page 91
- Section "Establishing an Ethernet connection", page 185

### 3. Commissioning

#### 3.6 Automatic booting of the memory card

#### 3.6 Automatic booting of the memory 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, it will check whether the Allow overwriting via card option is  enabled.

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 115.

##### 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 8  
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 8  
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 8 is not needed in this case.

### 3.6.1 Preparing the card in the device for booting with easySoft 8

Only possible with easySoft 8.

#### Prerequisites

- Licensed easySoft 8 version on the PC

- ▶ Insert the microSD memory card into a card reader on your PC.
- ▶ Open easySoft 8 and open the project you want to transfer, e.g., <test.e80>.
- ▶ 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.

3. Commissioning

3.6 Automatic booting of the memory card

Transfer program

Project/Card... menu option

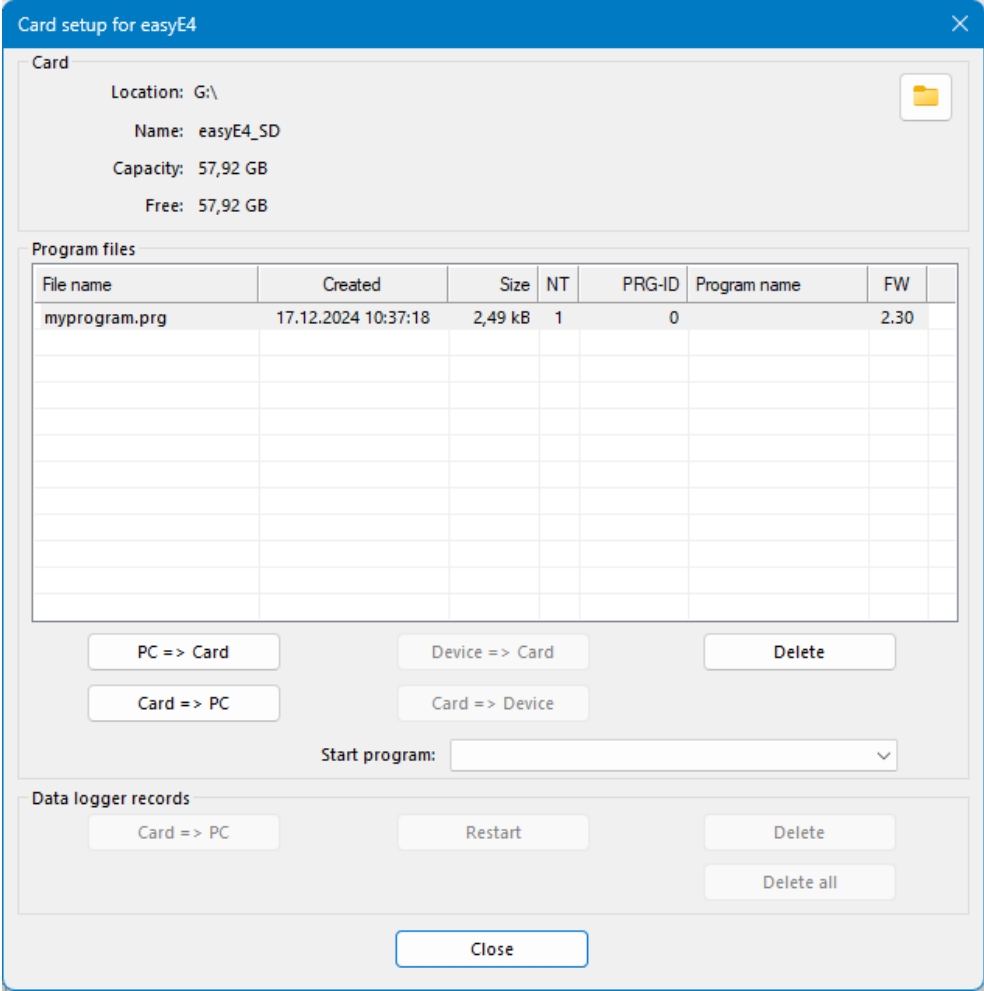
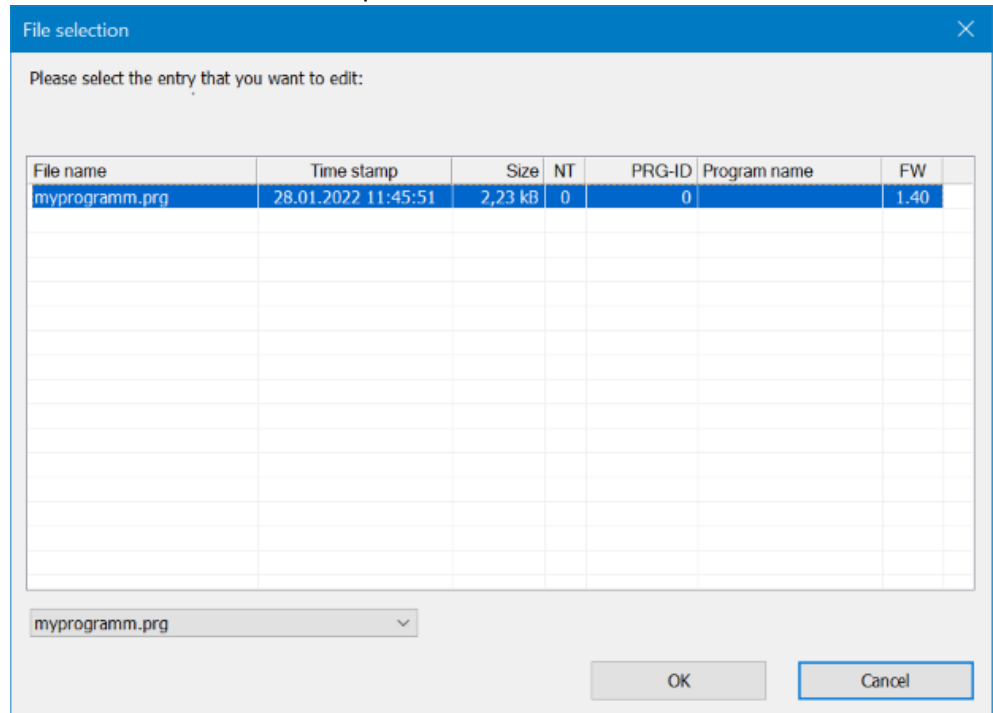


Fig. 70: Offline dialog box for memory card

- ▶ 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 \*.e80 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 is then run. If the plausibility check is completed successfully, the prompt will appear after the start program.

### 3. Commissioning

#### 3.6 Automatic booting of the memory card

##### 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 .e80 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.

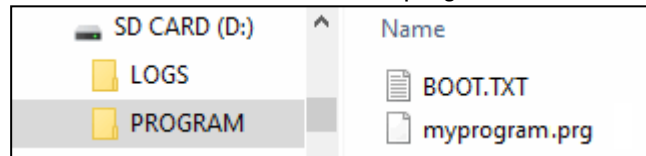


Fig. 71: microSD memory card drive with PROGRAM folder contains BOOT.TXT and compiled test.prg program

The card is now prepared with all the prerequisites for booting. You can now use automatic booting for the card.



### 3.6.2 Preparing the card in the easyE4 device for booting with easySoft 8

#### Prerequisites

- Licensed easySoft 8 version on the PC
- ▶ Insert the card into the device while the latter is de-energized.
  - ▶ Switch on the supply voltage.
  - ▶ Open easySoft 8 and open the project you want to transfer, e.g., <myprogram.e80>.
  - ▶ 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
  - ▶ 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 *Communication view/System settings*.
  - ▶ Go to the *Communication view/Program/Configuration* section and click on the Card... button.

The Card setup dialog box will appear.





### 3. Commissioning

#### 3.6 Automatic booting of the memory card

##### 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 .e80 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.



Fig. 73: 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 for the card.

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

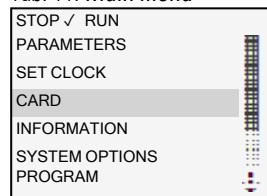
#### Prerequisites

- 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.

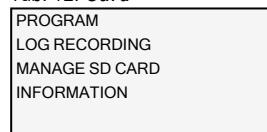
- ▶ Insert the memory card while the device is de-energized.
- ▶ Switch on the supply voltage.
- ▶ Go to the main menu.
- ▶ Open the CARD menu option.

Tab. 11: *Main menu*



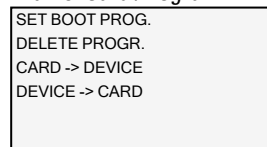
- ▶ Go to PROGRAM

Tab. 12: *Card*



- ▶ Go to START PROGRAM

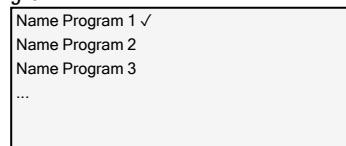
Tab. 13: *Card\Program*



- ▶ 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.

Tab. 14: *Card\Program\Start program*



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

### 3. Commissioning

#### 3.6 Automatic booting of the memory card

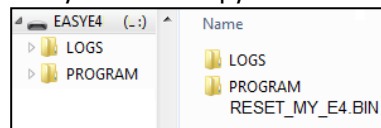
- ▶ Switch off the power supply.

The card is now prepared with all the prerequisites for booting. You can now use automatic booting for the card.

**3.7 Reset with memory card - reset device to its delivery condition****3.7 Reset with memory card - reset device to its delivery condition**

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

The firmware of individual devices can be updated. This procedure varies depending on the hardware generation of the devices.

easyE4 devices can be updated from version 1.00.



A device that was updated to version 1.10 or higher can no longer be reset to version 1.00.



A device with firmware version 2.xx cannot be rolled back to version 1.xx.



Firmware version 2.25 and higher cannot be loaded directly onto devices with firmware version 2.02 or 2.00.

Instead, these devices must first be updated to firmware version 2.10.

The base devices in generation 05 and higher feature a connection to an easy communication module:

- EASY-COM-SWD-..., can be updated starting with version 1.30 and higher.
- EASY-COM-RTU-..., can be updated starting with version 1.40 and higher.

The base devices from the generation 08 have secure communication with easyProtocol V2. They perform much better than the previous generation because they have a bigger program memory and can communicate faster. The easyE4 base devices as of this version deliver a TLS device certificate that is based on the easyE4 root certificate.

Up to generation 08, the procedure for base devices differs from the procedure for expansion devices or communication modules.

To update the firmware, you will need to use a microSD memory card.

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



Devices from generation 02 up to generation 06 can only be updated with firmware <V2.00.



Devices belonging to generation 08 or higher can only be updated with firmware V2.00 or higher.



easyE4 base devices belonging to generation 09 with firmware version 2.30 or higher can be updated through the Web Client or through the AWS Cloud (in addition to updates made with microSD memory cards).

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.



### 3. Commissioning

#### 3.8 Updating firmware

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

A configuration file is not needed for expansion devices and easy communication modules.



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.

#### **See also**

→ "easyE4 compatibility overview", page 881

3. Commissioning

3.8 Updating firmware

3.8.1 Firmware Update base device

All base devices can be updated with newer firmware.

Firmware versions 2.xx can be updated on all devices belonging to generation 08 and higher.

Firmware versions 1.xx can be updated on all devices belonging to generation 01 through 06.

To identify the generation to which your easyE4 device belongs, please refer to the nameplate.

To identify the firmware version that is installed on the base device, establish online communication with the easyE4 base device and, in *easySoft 8Communication view/HW Info tab* shown.

In the case of easyE4 base devices with a display, the firmware version can be seen in the *Information\System* device menu → Section "MenuInformation", page 165

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.

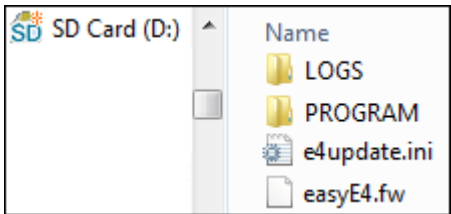


Fig. 74: microSD memory card content when using bootloader version 1.01

The specific files that will be unzipped and that are required for the firmware update will depend on the bootloader version 1.xx on the 01, ...,06 base device, as shown below:

Unzipped files	Boot loader version	
	1.00	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	0	An update from the microSD memory card will always be run.
1	1	

- ➔ 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 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.

### 3. Commissioning

#### 3.8 Updating firmware

- 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.

#### **Update of base device from generation 08**

The firmware update for EASY-E4-...-12...C1(P) easyE4 base devices from generation 08 can be booted from the device menu and also from the "e4settings.ini" configuration file on the microSD memory card.

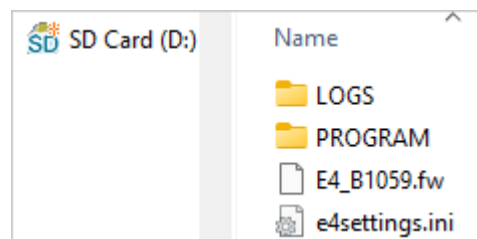


Fig. 75: microSD memory card content when using bootloader version 2.00

This requires the unpacked firmware file "E4\_V2xx.FW" to be on the microSD memory card.

You can reach the device menu through

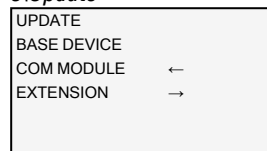
- a base device with display, or
- Display\Display + Buttons in the easySoft 8 Communication view or
- from a remote display with the web server.

An individual update needs to be conducted for each base device.

To update a base device with display, proceed as follows:

- ▶ Go to the main menu.
- ▶ Open the menu path SYSTEM OPTIONS\UPDATE\BASE DEVICE.

Tab. 15: *System options\Update*



- ▶ 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.

On completion of the update, the display jumps back to the SYSTEM OPTIONS\UPDATE\BASE DEVICE menu.

With the "e4settings.ini" configuration file, certain system parameters can be pre-configured, → Section "Set system parameters using a memory card - e4settings.ini", page 148.

#### 3.8.2 Updating the firmware on expansion devices

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. in

easySoft 8 the *Communication view/HW Info tab* during online communication .

You can reach the device menu through

- a base device with display, or
- Display\Display + Buttons in the easySoft 8 Communication view or
- from a remote display with the web server.

An update must be run separately for each expansion device.

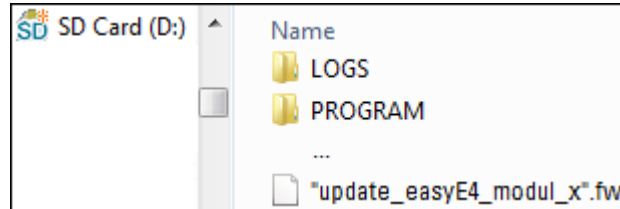
Just as with updates for base devices, this requires for the required unzipped "\*.FW" firmware file to be stored on the microSD memory card.

- ▶ Load the firmware you want on your computer.
- ▶ Connect an empty microSD memory card (FAT format) to your computer.

### 3. Commissioning

#### 3.8 Updating firmware

- 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 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**.

Tab. 16: *System option-  
s\Update*

UPDATE	
BASE DEVICE	
COM MODULE	←
EXTENSION	→

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

Tab. 17: *System options\Update\Expansion*

```
EXTENSION  
<1-11>  
  
UPDATE  
<Filename on SD>  
$fff|iiiiiii|ffffffC
```

- ▶ 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 8.

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.

### 3.8.3 Updating the firmware on easy communication modules

easy communication modules must be updated through the device menu of an easyE4 base device.

### 3. Commissioning

#### 3.8 Updating firmware

To find out which firmware version is found on the device, check the Communication view/HW Info tab during online communication. in easySoft 8, the *Communication view/HW Info tab* during online communication .

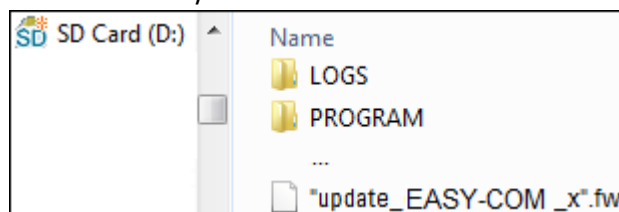
You can reach the device menu through

- a base device with display, or
- Display\Display + Buttons in the easySoft 8 Communication view or
- from a remote display with the web server.

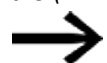
An update must be run separately for each easy communication module.

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 file that matches the easy communication module (.FW).



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

To update the firmware, the easy communication module must be connected to the base device by a plug connector.

The number of the easy communication module is indicated by 01.

An individual update must be performed for each easy communication module.

#### Updating EASY-COM-... using a base device with a display


To update an easy communication module using a base device with a display, follow the steps below:

- ▶ Go to the main menu.
- ▶ Open the SYSTEM OPTIONEN\UPDATE\KOMM.-MODUL menu path.

Tab. 18: *System option-  
s\Update*

UPDATE	
BASE DEVICE	
COM MODULE	←
EXTENSION	→



- ▶ Press the **OK** button.
- ▶ Use the cursor button P4  to skip the number of the easy communication module.

Tab. 19: *System option-*

*s\Update\COMM.-MODULE*

COM MODULE <01>  UPDATE <Filename on SD> çEEE iiii EEEEEEÇ
---

- ▶ Use keys < > to select the corresponding firmware file, e.g., "eComSWD\_B0023.fw".  
 ➔ Please note that the name of the firmware file must not exceed 14 characters.
- ▶ 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 is completed, the display will jump back to the status display.

- ➔ You can only view the hardware information (HW info), i.e., which firmware version is on the easy communication module, via easySoft 8.  
 To do this, go to the Communication view and connect to your easyE4 block. In the workspace Configuration, the FW version is displayed in the HW info.

### 3. Commissioning

#### 3.9 Functions of the microSD memory card

#### 3.9 Functions of the microSD memory 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 format, Speed Class 2 or higher).

Firmware version 2.00 and higher also supports SDXC memory cards with a capacity of up to 256 GB.



SDXC memory cards with large capacities are often formatted to exFAT. Before using them, these SDXC memory cards need to be formatted to FAT32 with the easyE4 or a PC.

**See also**

→ Section "", page 214



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



Note on device safety when relevant changes are made:  
A confirmation prompt will appear in the device menu. The program will not be deleted until you select Yes and press **OK** as a confirmation.



Do not install or remove microSD memory card while the easyE4 is switched on.

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
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 510

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 8, in the Project view.

The DL - Data logger function block can be used for logging data and states.

##### 3.9.1 Ejecting the microSD memory card

As an alternative to removing the memory card from the device, you can eject it with easySoft 8.

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

## 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*



Fig. 76: boot.bmp

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

*microSD memory card in the PC*

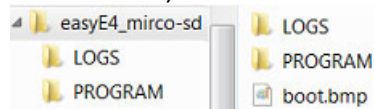


Fig. 77: Storing the boot.bmp file

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.

### 3. Commissioning

#### 3.11 Set system parameters using a memory card - e4settings.ini

#### 3.11 Set system parameters using a memory card - e4settings.ini

A few system settings on the base device can be defined from generation 08 with the help of the "e4settings.ini" file and can be transferred to the device by the microSD without using easySoft 8 or the device menu. These system settings are not part of a user program.

The "e4settings.ini" file is saved in the same directory (ROOT) as the one for a firmware update.

Creation and editing of this "e4settings.ini" file must be in the form of an ASCII text file. You can do this with any text editor on the PC.

The syntax for value assignment can be taken from the example of the ini file, see → Section ""e4settings.ini" example for data content from generation 08", page 152



Compliance with this syntax is mandatory.  
The values can be adapted.

The following system settings for the display and update behavior can be set in the "e4settings.ini" file:

- Display language
- Brightness display 1 and 2
- Timeout for brightness changeover
- Color setting
- Start time for the graphic screen
- Firmware update behavior

To transfer the parameters of the "e4settings.ini" file to the base device, proceed as follows:

- ▶ Switch off the easyE4 base device.
- ▶ Plug the microSD memory card with the "e4settings.ini" file into the microSD card holder, then slide the holder into the device.
- ▶ Switch on the easyE4 base device.

The parameters have then been adopted by the base device.



If the entered value is not plausible,  
the previously set value for the base device is retained.



Any parameters not required do not need to be set.  
The sequence of parameters is not defined.

## 3.11 Set system parameters using a memory card - e4settings.ini

**Display Language - Display Language**

Used to set the device menu language, → Section "Switch languages", page 644

Tab. 20: *Display Language*

*guage*

0	ENGLISH
1	DEUTSCH
2	FRANCAIS
3	ESPAÑOL
4	ITALIANO
5	NEDERLANDS
6	POLSKI
7	ČESKÝ
8	PORTUGUÊS
9	РУССКИЙ
10	TÜRKÇE
11	ROMÂNĂ
12	MAGYAR
13	SRPSKI
14	HRVATSKI
15	SLOVENŠČINA

**Brightness1, Brightness2 - Display brightness 1 and 2**

The two brightness levels 1 and 2 can be set as multiples of 10, i.e. they can be edited in increments of 10.

The value range is between 0 and 100 (%). An intermediate value is rounded up to the next multiple of ten

Brightness1 Display brightness during operation on the device, see → Section "Display", page 636  
Default value: 100

Brightness2 Brightness for sleep mode  
Default value: 50  
Value: 0 will cause the display to be switched off in sleep mode

### 3. Commissioning

#### 3.11 Set system parameters using a memory card - e4settings.ini

##### Timeout Brightness - Timeout for brightness changeover

Indication of time in seconds when the display changes to rest mode if no operation takes place on the easyE4 device.

The changeover time between this display brightness 1 and 2 must be indicated in seconds in accordance with the following table.

Tab. 21: *Timeout*

*Brightness*

0	s
10	s
30	s
60	s (1 min)
120	s (2 min)
300	s (5 min)
600	s (10 min)
900	s (15 min)

If an intermediate value is indicated in seconds, the value is rounded up to the nearest second in accordance with the table.

Example: If 2 seconds are indicated in the \*.ini file, the value is rounded up to 10 seconds.

## 3.11 Set system parameters using a memory card - e4settings.ini

**Color - color setting**

With this indication of an index, the color scheme of the display showing menu and title entries, warning and error messages as well as input boxes and the cursor color can be defined as one of the 16 predefined color combinations.

The relevant color settings here are those for remote operation of the easyE4, e.g. on the easyE RTD, in the easySoft 8 or the web server.

The following table lists the two dominant colors in the color scheme for text and background design from the related color index value:

0	Black / white (default)
1	white / black
2	Black / white (alternative)
3	White / black (alternative)
4	Black / white (Alternative2)
5	White / black (Alternative2)
6	Grey blue / light blue
7	White / dark blue
8	Dark brown / light brown
9	Light brown / dark brown
10	Dark green / light green
11	Light green / dark green
12	Dark red / light red
13	Light red / dark red
14	Dark purple / light purple
15	Black / white (Alternative3)

Color index for the two main colors for text and background design

Colors Index		Warning Text Background	Error Text Background	Inputs Text Background	Navigation Text Background	Headings Text Background	Cursor background Text Input
0	Text						
1	Text						
2	Text						
3	Text						
4	Text						
5	Text						
6	Text						
7	Text						
8	Text						
9	Text						
10	Text						
11	Text						
12	Text						
13	Text						
14	Text						
15	Text						
16	Text						

Fig. 78: Color scheme from the index during remote operation of the easyE4

### 3. Commissioning

#### 3.11 Set system parameters using a memory card - e4settings.ini

##### Timeout start graphics - Start time for the graphic screen

Display time for the boot.bmp graphic before the status display appears on screen.

The display time of the start graphic can be configured within the limits of 0 to 10 seconds ( $0 \leq x \leq 10$ ).

→ Section "Setting a splash screen for the EASY-E4-...-12...C1(P) display", page 147

##### Firmware update - behavior

An alternative way of starting the firmware update on the base device via the configuration file if no display is present, and/or if no access is possible to the device menu. See → Section "Updating firmware", page 136

Two parameters are required for this:

- updatefw** Targeted update of firmware involving the setting of permitted values: 0 or 1.  
If the value is not set to 1, no FW update takes place but the other system settings are adopted.
- updatefile:** Targeted selection of a specific firmware update <file name>.fw.  
The corresponding \*.fw file must be in the ROOT directory of the microSD memory card.  
To learn which firmware updates match which hardware generation, see → Section "Updating firmware", page 136

"e4settings.ini" example for data content from generation 08

##### sample e4settings.ini file

```
Display Language=0
Brightness1=80
Brightness2=70
Timeout Brightness=30
Timeout start graphics=1
updatefw=1
updatefile:E4_V200.fw
Color=3
```

##### See also

- "Overview of switch-on behavior", page 115
- "System settings", page 634



## 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 relay with LED indicators", page 106

### 4.1 Base device with display and buttons



Fig. 79: Display and keypad

#### 4.1.1 LCD Display

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

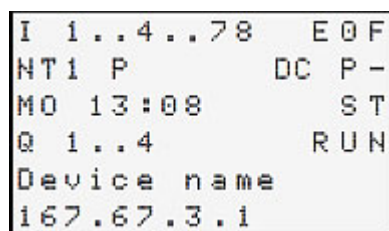


Fig. 80: Example of status display on display

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. Operation

### 4.1 Base device with display and buttons

#### 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 check-mark ✓ 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 636





#### 4.1.2 Keyboard


<b>DEL</b>	Deleting in the circuit diagram
<b>ALT</b>	Special functions in circuit diagram, Status display
Cursor buttons	Move cursor Select menu items, Change numbers, contacts and values
<b>ESC</b>	Back, Cancel
<b>OK</b>	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 	Calling the System menu
	Return to the previous menu level Cancel entries since last 
	Move to the next menu level Call menu item Activate, change, save entries

Cursor buttons	Change menu item, Change the value, 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  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  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. Operation

4.1 Base device with display and buttons












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  button to switch to <b>Entry</b> mode so that you can enter or change a value at the current cursor position. Press the  button in <b>Entry</b> mode to restore to the last changes of an entry.
I	Press the  button to switch to <b>Connect</b> mode for wiring contacts and relays. Press the  button again to return to <b>Move</b> .
	Press the  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
	Cancel, retain previous value
	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-up of real-time clock", page 886

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 645

### 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.

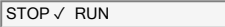
4. Operation

4.2 Operating modes of the easyE4

The main menu on the easyE4 display is used to switch between operating modes, i.e., from RUN to STOP and vice versa,→ Section "STOP RUN operating mode menu", page 161

→ 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.

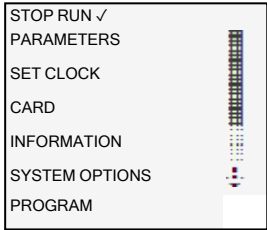
→ In order to configure it, the program must be stopped.



Operating mode changes may be protected with a password.

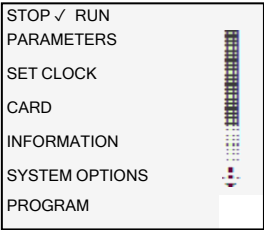
Ongoing operation

Tab. 22: Main menu



To work on the easyE4









Tab. 23: Main menu













## 4.3 Operation of the menu selection and value entry

## 4.3 Operation of the menu selection and value entry

## 4.3.1 How to navigate the device menus

	Select; confirm value
	Cancel; go back
	Delete
	Depending on the starting point: - Toggle display - Jump to the start or end of the menu - 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
	Delete rung, contact, relay or empty rung in the circuit diagram
	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
	Undo setting from last OK, current display, leave menu
	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. Operation

### 4.3 Operation of the menu selection and value entry

#### 4.3.3 Selecting a device menu

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

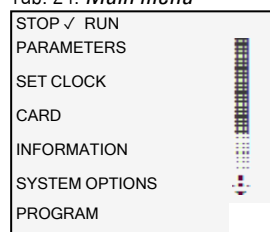
► Press the **OK** button.

The main menu will appear.

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

Tab. 24: *Main menu*



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



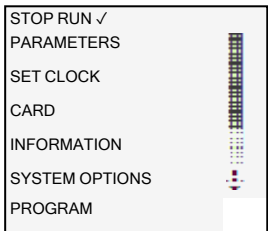
4.4 Overview of the menus on the device

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

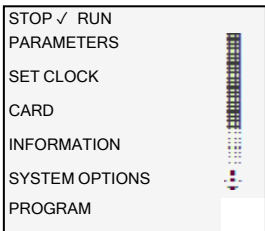
Ongoing operation

Tab. 25: *Main menu*



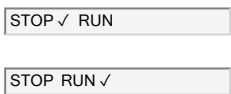
To work on the easyE4

Tab. 26: *Main menu*



4.4.2 STOP RUN operating mode menu

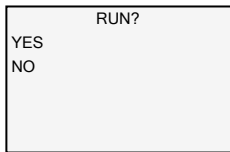
This submenu can be used to switch between operating modes.



Tab. 27: *STOP*



Tab. 28: *RUN*



See also

→ Section "Operating modes of the easyE4", page 157

4. Operation

4.4 Overview of the menus on the device

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

Tab. 29: *Parameter*

STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM OPTIONS
PROGRAM

Tab. 30: *Parameter*

T 01	Ü	S	+
C 02			
L:1			STOP

Tab. 31: *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.

Tab. 32: *Timer module example*

T 01	Ü	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

Tab. 33: *Set clock*

STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM OPTIONS
PROGRAM

Tab. 34: *Set clock*

DATE/TIME
DST
RADIO CLOCK
ASTRO CLOCK

Tab. 35: *Set Clock-*

<i>Date&amp;Time</i>
DD-MM-YYYY
FR 2018/08/13
12:03:04

Tab. 36: *Set clock\Daylight saving*

NONE	✓
MESZ	
US	
RULE	

Tab. 37: *Set Clock\Radio Clock*

RADIO CLOCK
ACTIVE : YES
INPUT : 1001
OFFSET : +000'

Tab. 38: *Set clock\astron. clock*

ASTRO	CLOCK
LAT N089.	9990000
LON E000.	0000000
OFFSET :	+000'

#### See also

→ Section "Time and Date setting", page 659

## 4. Operation

### 4.4 Overview of the menus on the device

#### 4.4.5 Menu Card

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

Tab. 39: *Main menu*

STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM OPTIONS
PROGRAM

Tab. 40: *Card*

PROGRAM
LOG RECORDING
MANAGE SD CARD
INFORMATION

Tab. 41: *Card\Program*

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

Tab. 42: *Card\Log recording*

START NEW LOG
DELETE OLD LOG
DELETE ACT.

Tab. 43: *Cards\Manage card*

FORMAT
RELEASE CARD

Tab. 44: *Card\Information*

EXISTING:	YES
FORMATTED:	YES
SIZE	xxxMB
FREE	xxxMB

#### See also

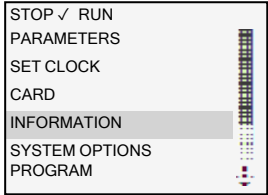
- Section "Functions of the microSD memory card", page 146
- Section "Transferring programs from and to a microSD memory card", page 213
- Section "Configuring the microSD card and device ID", page 658

4.4.6 MenuInformation

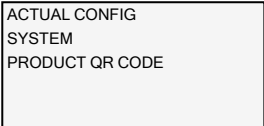
Shows the current status of the easyE4 device.

Opens additional menus,  
The submenu is only provided in English

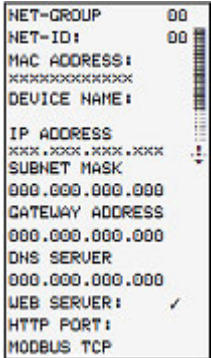
Tab. 45: Main menu



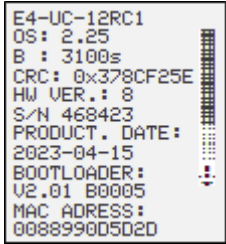
Tab. 46: Information



Information\Actual Config



Information\System



Shows the versions on the easyE4  
Specification of the part number  
OS: Firmware version  
B: Firmware build version  
CRC: Cyclic redundancy check result  
HW VER.: Hardware version/revision  
S/N Serial number  
Product Date: Date of production  
Bootloader: Version  
MAC address

Information\Product QR Code



Shows the QR code for the easyE4  
and, with ⓘ, the serial number and security code in  
cleartext

To update the devices

- Section "Updating firmware", page 136
- Section "Device information", page 668
- "easyE4 compatibility overview", page 881

4. Operation

4.4 Overview of the menus on the device

4.4.7 System options menu

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

Opens additional menus

Tab. 47: *Main menu*

STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM OPTIONS
PROGRAM

Tab. 48: *System options*

SECURITY
SYSTEM
MENU LANGUAGE
DELETE PROGR.
NET
ETHERNET
UPDATE

Tab. 49:

<i>System options\Security</i>
PASSWORD
RANGE

Tab. 50: *System option-  
slSystem*

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

Tab. 51: *System Option-  
slMenu Language*

ENGLISH	
DEUTSCH	✓
FRANCAIS	
ESPAÑOL	
ITALIANO	
NEDERLANDS	
POLSKI	
ČESKÝ	
PORTUGUÊS	
РУССКИЙ	
TÜRKÇE	
ROMÂNĂ	
MAGYAR	
SRPSKI	
HRVATSKI	
SLOVENŠČINA	

Tab. 52: *System option-  
slDelete progr.*

DELETE PROGRAM?
YES
NO

Deletes the program in  
the easyE4 device

Tab. 53: *System options\Net*

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

The submenu is only  
provided in English

## 4. Operation

### 4.4 Overview of the menus on the device

The submenu is only provided in English easyE RTD available in OS Version 1.25 and higher  
Email test from OS version 2.0

Tab. 54: *System option-  
slEthernet*

ADDRESS MODE
IP ADDRESS
SUBNET MASK
GATEWAY ADDRESS
DNS SERVER
easyE RTD
Email Test

Available in OS Version 1.10 and higher

Tab. 55: *System option-  
slUpdate*

UPDATE	
BASE DEVICE	
COM MODULE	←
EXTENSION	→

#### See also

- Section "System settings", page 634
- Section "Security – password protection", page 654
- Section "Setting up a NET group", page 721
- Section "Setting up a Web Server", page 728
- Section "Modbus TCP", page 830
- Section "Setting up the e-mail function", page 758
- Section "Convenient visualization for easyE4", page 856
- Section "Functions of the microSD memory card", page 146

4. Operation

4.4 Overview of the menus on the device

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.

Opens another menu

Tab. 56: *Main menu*

STOP ✓ RUN	
PARAMETERS	
SET CLOCK	
CARD	
INFORMATION	
SYSTEM OPTIONS	
PROGRAM	

Tab. 57: *Programs*

CIRCUIT DIAGRAM
FUNCTION BLOCKS

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

I001	—	I002
Q001	—	HY01Q1
L: 1 C:1 40352		

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

Tab. 58: *Programs\Function Blocks*

T 01	Ü	S	+
C 02			
L:1 STOP			

Tab. 59: *Timer module example*



T 01	Ü	S	+
>I1	000	800	
>I2	009	200	
QV>	000	000	
..			



## 4. Operation

### 4.4 Overview of the menus on the device

#### 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.

Tab. 60: *Programs\Circuit diagram*



Tab. 61: *Programs\Function blocks*



## 4. Operation

### 4.5 Your first EDP program

### 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.

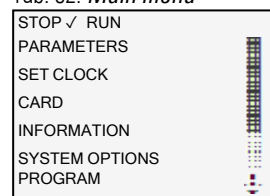
The easyE4 program will take care of the entire wiring for these components.

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



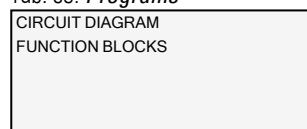
Use easySoft 8 to create your own program

Tab. 62: *Main menu*



Opens another menu

Tab. 63: *Programs*



#### 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*

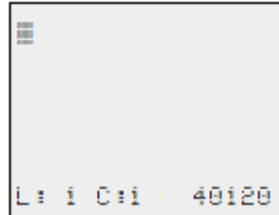


Fig. 81: Empty circuit diagram

The last line shows the cursor's position:

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

## 4. Operation

### 4.5 Your first EDP program

#### 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*



Fig. 82: 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.

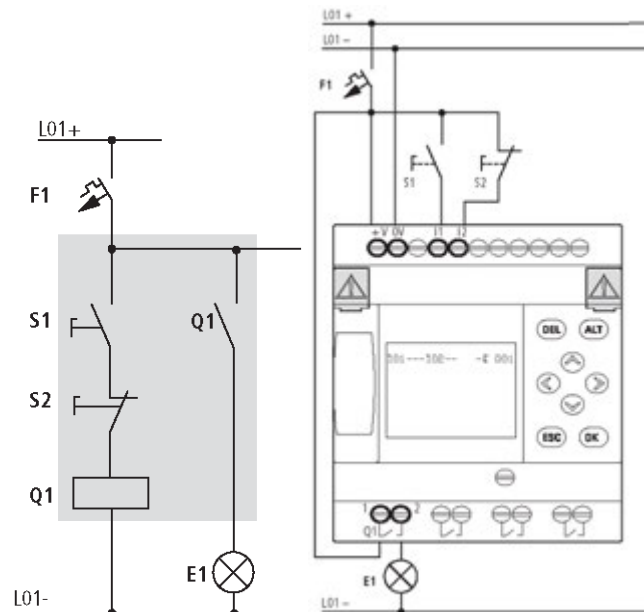


Fig. 83: Lighting control circuit

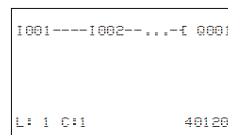



Fig. 84: Circuit diagram with inputs I01, I02 and output Q1

► 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 easyE4 device'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  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  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.

## 4. Operation

### 4.5 Your first EDP program

- ▶ 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

An 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

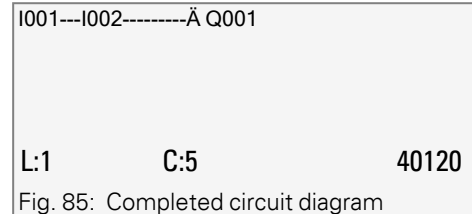


Fig. 85: Completed circuit diagram

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

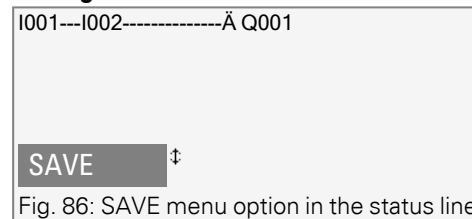


Fig. 86: SAVE menu option in the status line

- ▶ 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. Operation

### 4.5 Your first EDP program

#### 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.

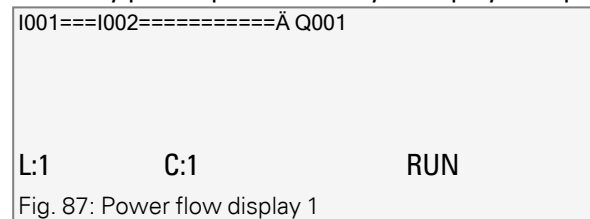


Fig. 87: Power flow display 1

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.

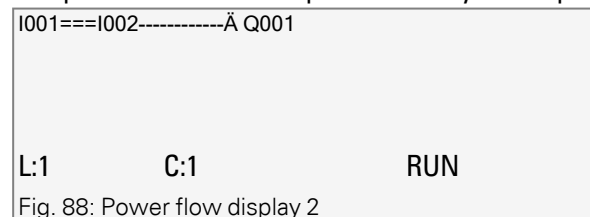


Fig. 88: Power flow display 2

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.

## 4. Operation

### 4.5 Your first EDP program

#### 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.

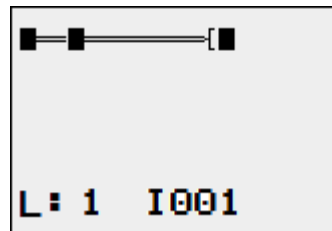


Fig. 89: Display with zoom, power flow

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.

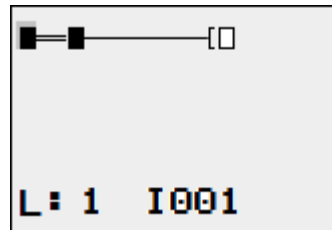


Fig. 90: Display with zoom, power flow interrupted

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

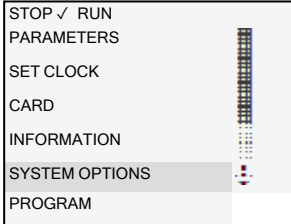
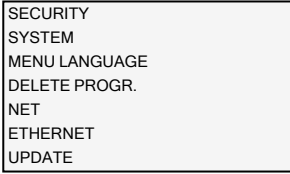
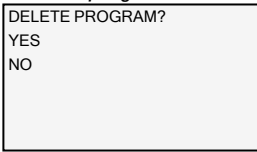
The system settings and operating parameters are reset to the default settings and also a possible NET parameterization.

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

<p>Tab. 64: <i>Main menu</i></p> 	<p>Tab. 65: <i>System options</i></p> 	<p>Tab. 66: <i>System option-Delete progr.</i></p> 
--	--	--

- 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. Operation

### 4.6 Transfer program to the easyE4 device

#### 4.6 Transfer program to the easyE4 device

There are two options for directly transferring a finished .prg file 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 256 GB., → Section "Installation instructions", page 98.
- A PC with easySoft 8 programming software installed on it, → Section "Installation instructions", page 98

- ▶ Insert the microSD memory card into a drive on your PC (with a suitable adapter if necessary).
- ▶ Open the easySoft 8 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 896



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

[Eaton.com/software/Anwendungsbeispiele/easy/Deutsch](https://www.eaton.com/software/Anwendungsbeispiele/easy/Deutsch)

[Eaton.com/software/Application Samples/easy/English](https://www.eaton.com/software/Application Samples/easy/English)

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

## 4. Operation

### 4.6 Transfer program to the easyE4 device

- Click on the Project\  Card/USB drive... menu option.

View Project easySoft 8

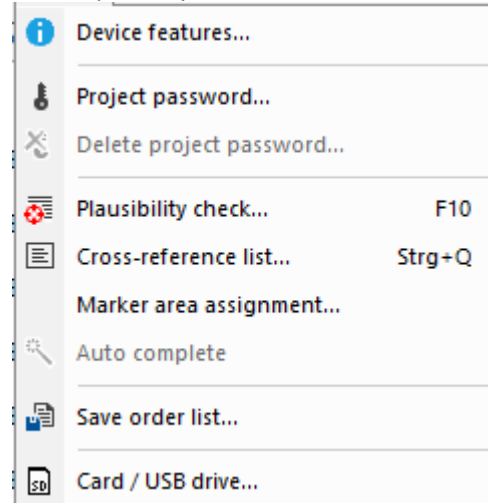

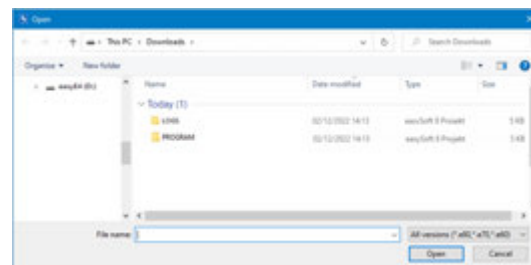


Fig. 91: Sample program open

In the Card setup dialog box that appears, click on the  icon to select the card root directory that easySoft 8 needs in order to create the LOGS and PROGRAM folders.

- Select the drive where the memory card is located

Exit the dialog box by clicking on **Select Folder**.



4. Operation

4.6 Transfer program to the easyE4 device

The Card setup for easyE4 dialog box will appear.

*easySoft 8 Project View\Project\Card / USB drive...*

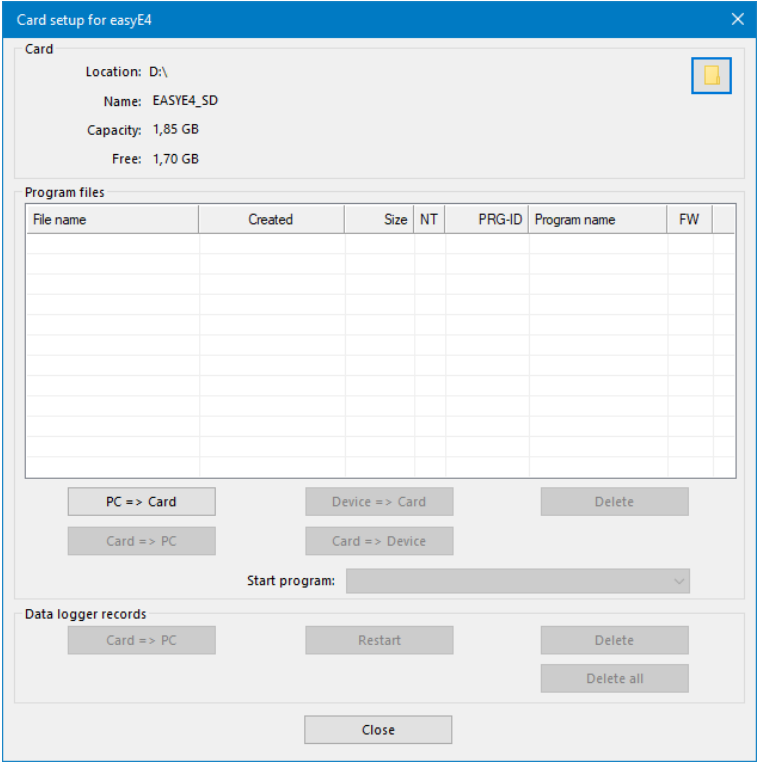
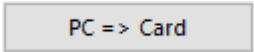


Fig. 92: Card setup dialog box

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.



## 4. Operation

### 4.6 Transfer program to the easyE4 device

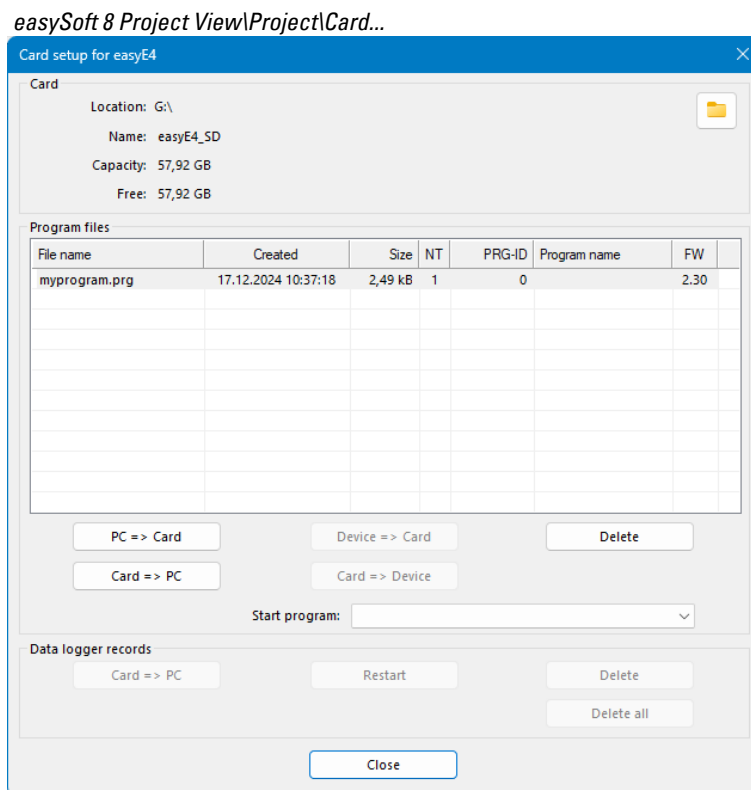


Fig. 94: The program was transferred to the memory card.

- ▶ 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 88

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 213



4.6.2 Establishing an Ethernet connection

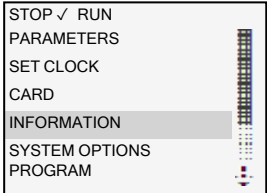
Establishing a connection between a PC and an easyE4 base device

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).

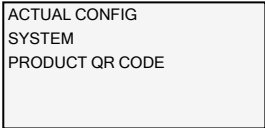
The PC's and the easyE4 base device's IP addresses must fall within the same range, i.e., the addresses' first two or three numbers must be the same, while the last number must be different and not equal to 0.

- Check the easyE4 device's IP address.
- To do this, open the *INFORMATION\ACTUAL CONFIG* menu and scroll to IP ADDRESS.

Tab. 67: Main menu



Tab. 68: Information



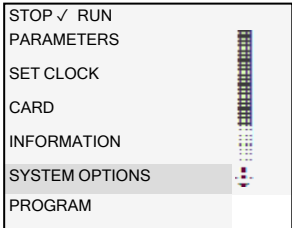
The submenu is only provided in English

Information\Actual Config

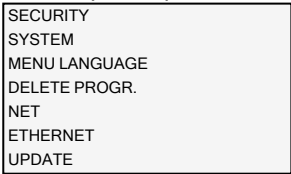


- If an IP address has not been assigned yet, enter one.
- To do this, open the *SYSTEM OPTIONS\ETHERNET\IP Address* menu.

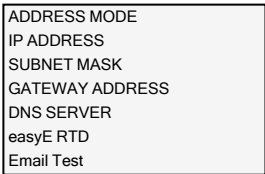
Tab. 69: Main menu



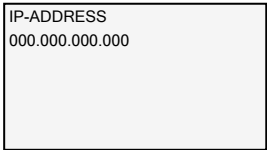
Tab. 70: System options



Tab. 71: System option-s\Ethernet



Tab. 72: System Option-s\Ethernet\IP Address



- Use the cursor buttons to enter the device's IP address.

## 4. Operation

### 4.6 Transfer program to the easyE4 device

Tab. 73: *System Option-  
s\Ethernet\Address mode*

AUTO-IP	✓
DHCP	
Static IP	

► Select the network setting you want.

Tab. 74: *System option-  
s\Ethernet\Email test*

Email Test?
YES
NO

► Possibility to check the e-mail function.  
→ Section "Setting up the e-mail function",  
page 758

► 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

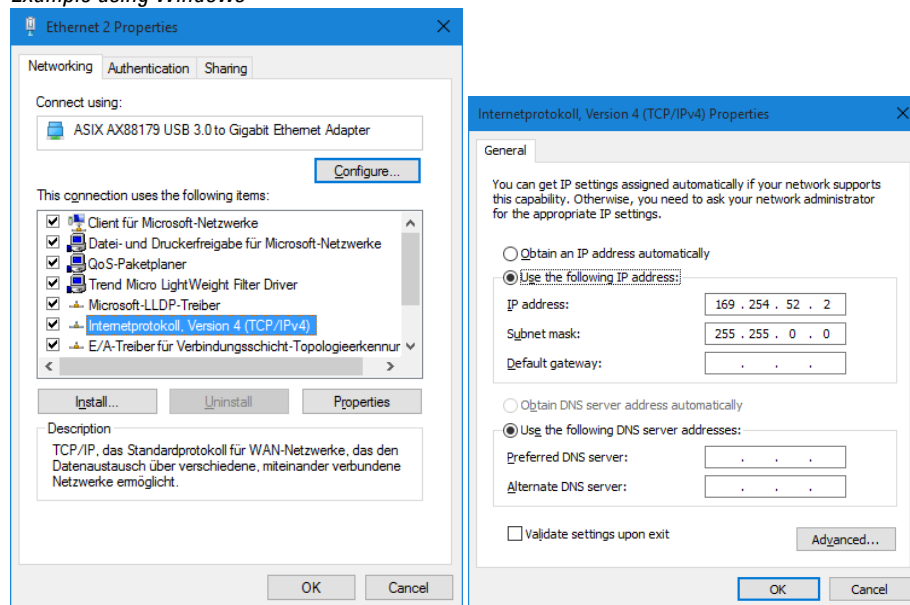


Fig. 95: Ethernet connection on PC

You can now connect to your easySoft 8 device with the easyE4 programming software.

#### See also

→ Section "Establishing an Ethernet connection and transferring a program or visualization project", page 117

## 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 .e80. However, please note that this file extension will not be shown on the display.



The programs themselves can be created very easily with easySoft 8 and then transferred to the easyE4 device. easySoft V8 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.

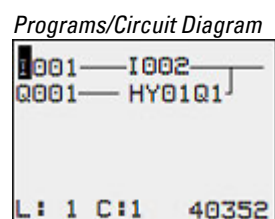


Fig. 96: Circuit diagram display

In the easyE4 circuit diagram, contacts and coils of relays are connected up 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

## 5. Programming on the device

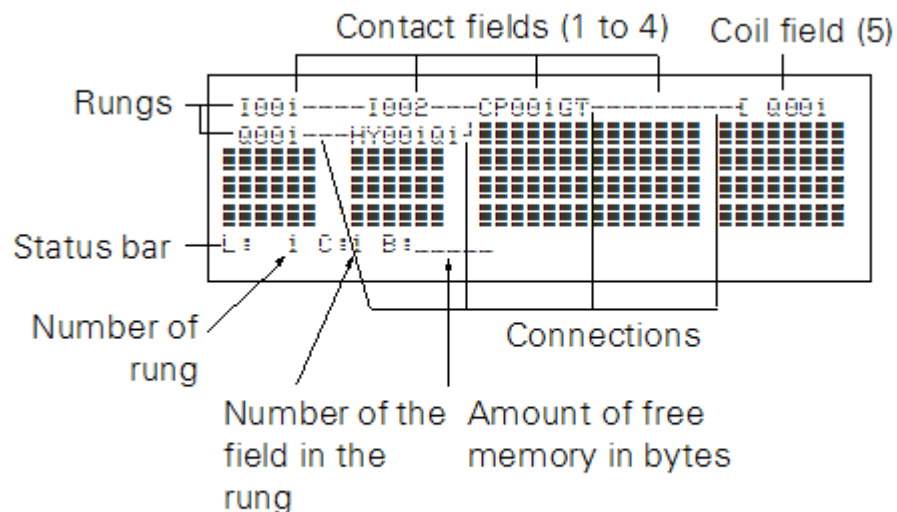
### 5.2 Circuit diagram display

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 display 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.

### **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 217

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. Programming on the device



### 5.3 Circuit diagram elements

#### 5.3.3 Contacts

You modify the current flow with the contacts in 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.

Tab. 75: 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 241

### 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:

Tab. 76: Coil function

Display	Coil function	Example	→ Page
	Contactor function	Q01,  D02,  S04,  :01,  M07,...	→ page 192
	Impulse relay function	Q03,  M04,  D08,  S07,  :01,...	→ page 192
S	Set	SQ08, SM02, SD03, SS04..	→ page 193
R	Reset	RQ04, RM05, RD07, RS03..	→ page 193
	Contactor function with negated result	Q04,  M96..	→ page 194
	Cycle pulse on rising edge	M01..	→ page 194
	Cycle pulse on falling edge	M42..	→ page 195



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, 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 217

## 5. Programming on the device

### 5.3 Circuit diagram elements

#### Coil with contactor function

The output signal follows the input signal directly, the relay operates as a contactor.

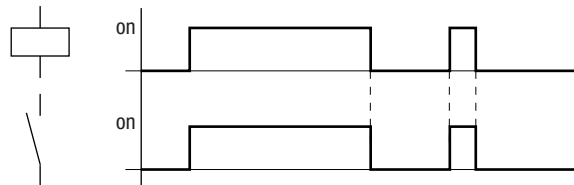


Fig. 97: Contactor function signal diagram

#### 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.

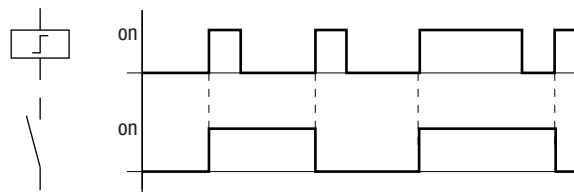


Fig. 98: Impulse relay signal diagram

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 651



### "Set" S and "Reset" R 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.

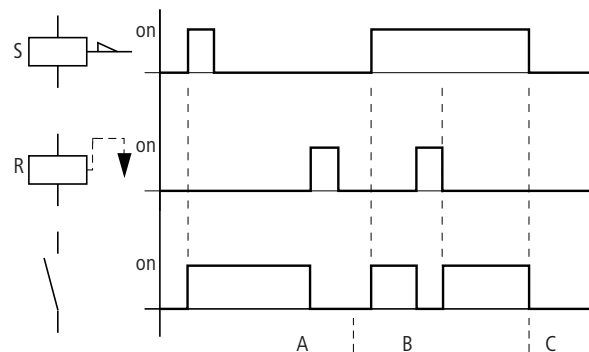


Fig. 99: Set and Reset signal diagram

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.)

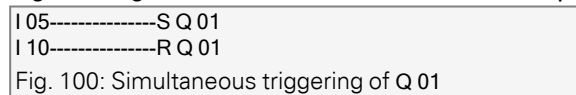


Fig. 100: Simultaneous triggering of Q 01

The above example shows the Reset coil with priority when the Set and Reset coil are triggered at the same time.

## 5. Programming on the device

### 5.3 Circuit diagram elements

#### Coil negation (inverse contactor function) ]

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.

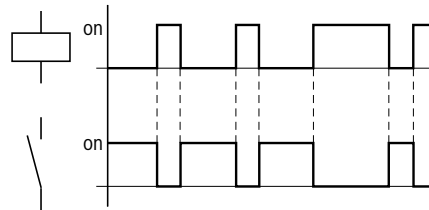


Fig. 101: Inverse contactor function signal diagram

#### 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.

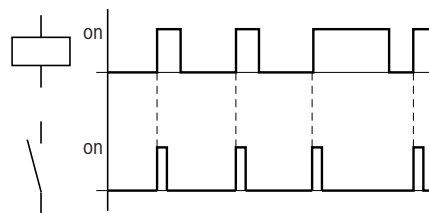


Fig. 102: Signal diagram of cycle pulse with rising edge

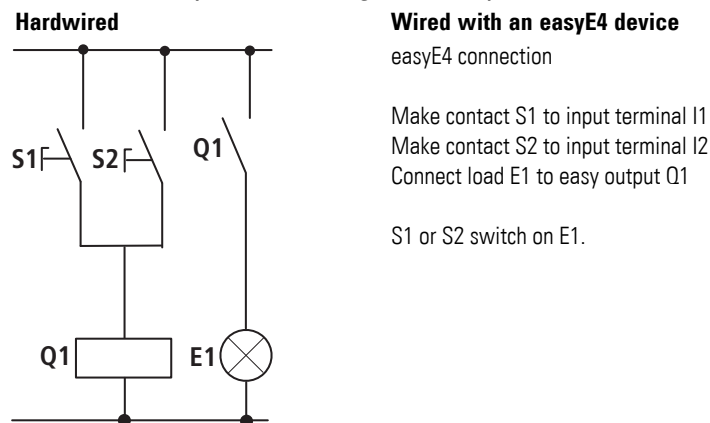


## 5. Programming on the device

### 5.4 Working with contacts and coils

#### 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.



#### easyE4 circuit diagram:

I 001-----Q 001  
I 002--k  
Fig. 104: 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. Programming on the device

### 5.4 Working with contacts and coils

#### 5.4.1 Entering and modifying contacts

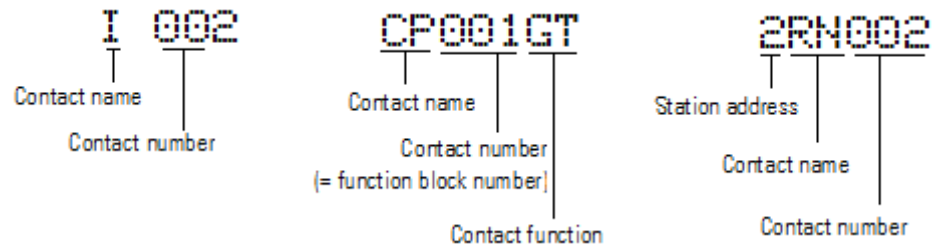


Fig. 105: Contact legend

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 241.

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. Programming on the device

### 5.4 Working with contacts and coils

#### 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.

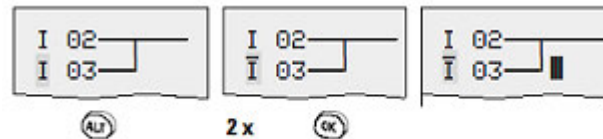


Fig. 106: Change contact I 03 from N/O to N/C

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.

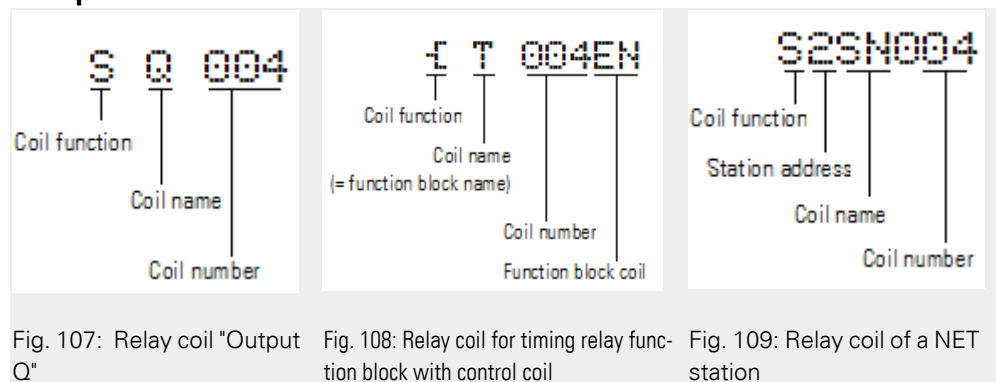
### 5.4.3 Entering and modifying coils

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



A complete list of all contacts and coils,  
→ Section "Function blocks", page 241

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  $\bar{A}$  Q 001 for entry in an empty field.

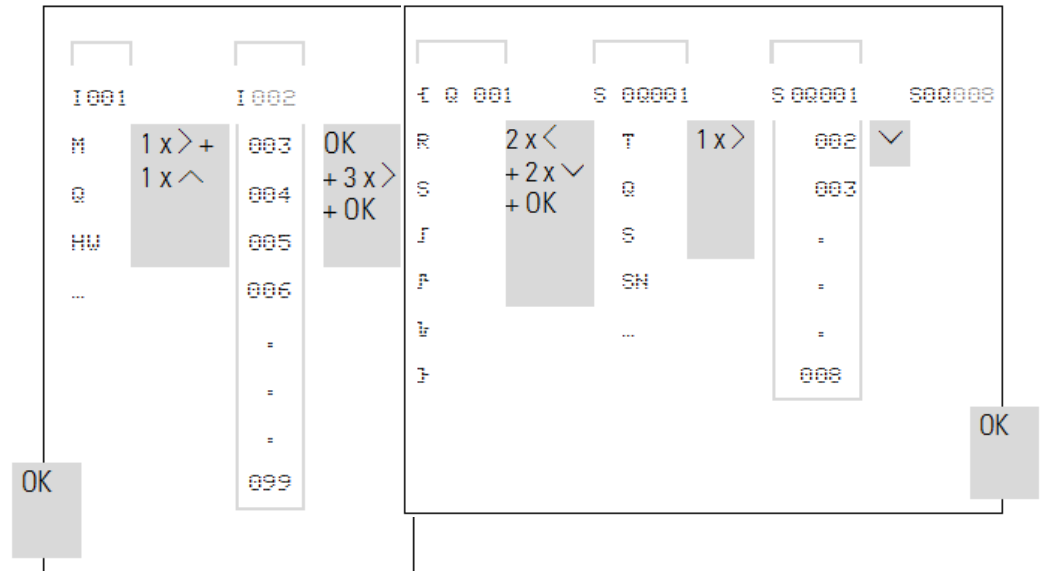
- ▶ Use the ⤴ ⤵ ⤶ ⤷ buttons to move the cursor to a free contact or coil field.
- ▶ Press the **OK** button to switch to Entry mode.
- ▶ Use ⤴ ⤶ to select the position you wish to change and press the **OK** to move to the next position  
(a selected position is shown in light grey in the following figure).
- ▶ 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. Programming on the device

### 5.4 Working with contacts and coils

In the contact field to change I 01 to I 02 In the coil field change Q 001 to S Q 008



#### 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.

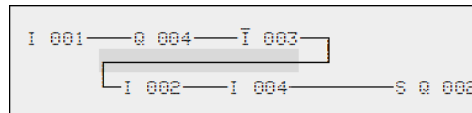


Fig. 110: Circuit diagram with five contacts, invalid

When using more than four contacts in series, use one of the 96 M or 128 M markers.

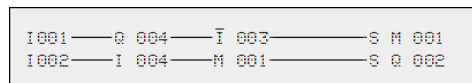


Fig. 111: Circuit diagram with M marker relay

## 5. Programming on the device

### 5.4 Working with contacts and coils

#### 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.

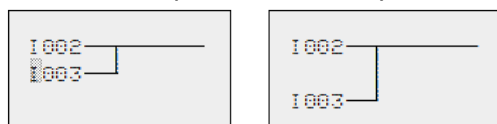


Fig. 112: Inserting a new rung

#### 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.

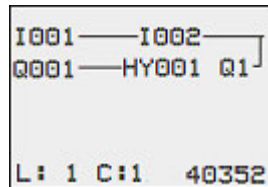
#### 5.4.9 Got to a rung

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. Programming on the device

### 5.4 Working with contacts and coils

#### 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.

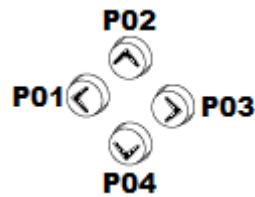


Fig. 113: The cursor buttons are wired in the circuit diagram as contacts P 01 to P 04.

#### 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 ÍÚ.

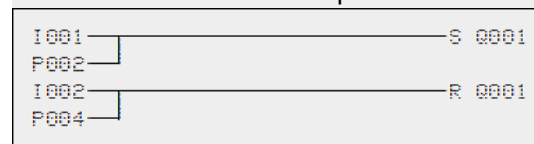


Fig. 114: Switch Q1 via I1, I2, Í, or Ú

#### 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 M 01.

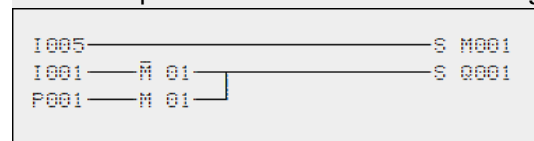


Fig. 115: I5 switches to cursor buttons.



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.



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 529 and → "JC - Conditional jump", page 524.

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.

## 5. Programming on the device

### 5.4 Working with contacts and coils



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"

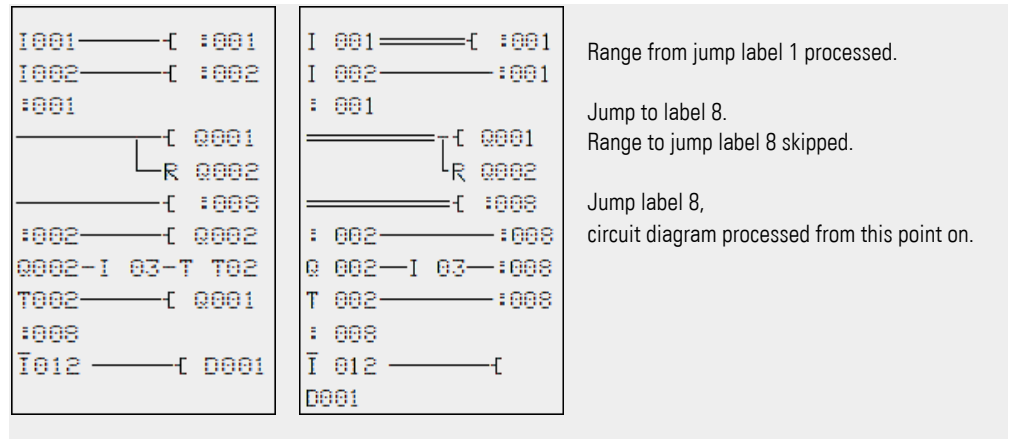
Circuit Diagram:

Power flow display: I001 selected:



## 5. Programming on the device

### 5.4 Working with contacts and coils



#### 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

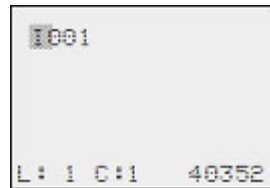
You must have selected an I.., Q.., R.., RN.., or SN.. operand in the circuit diagram and be in input mode.

This mode is displayed by a flashing operand.

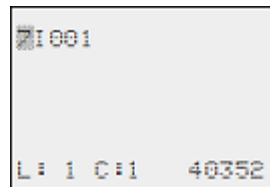
- Use the cursor button to move the cursor to the position to the left of the operand. A flashing zero appears as the initial value.

## 5. Programming on the device

### 5.4 Working with contacts and coils



- ▶ Use the  $\bar{I}$  or  $\bar{Q}$  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 217

→ Section "Function blocks", page 241

### 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:

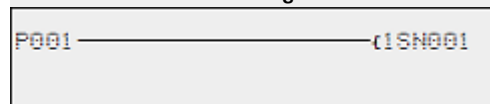
- on the sending station and receiving station use the same operand number for each SN/RN pair to be formed.
- in the circuit diagram of the sending station you set for the SN operand (coil) the station number (NET ID) of the receiving station.
- in the circuit diagram of the receiving station you set for the RN operand (contact) the station number (NET ID) of the sending station.

### SN-RN example

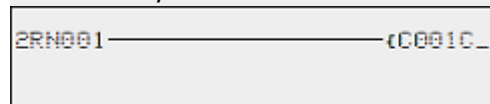
NET station 2 sends the status of the P button P01 via SN1 to NET station 1.



The relevant circuit diagram is as follows:



On NET station 1 the status of P1 is associated via RN01 as a count pulse for the counter relay C01.



## 5. Programming on the device

### 5.4 Working with contacts and coils

#### 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 684

More information about the function blocks: → Section "Function blocks", page 241

#### NET marker

N., nB., nW., nD...

Every station that the NET marker describes can read any of the other stations.

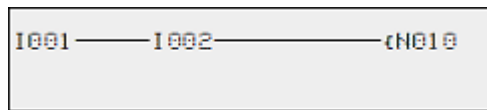


Fig. 118: 1 slave

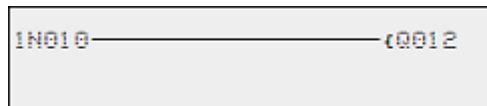


Fig. 119: 2 slave

## **5.5 Transferring programs from and to a 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: → Section "Functions of the microSD memory card", page 146

Programs are normally transferred from easySoft 8 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 of the memory card", page 124

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 8 if the latter is connected to the easyE4.



Do not install or remove microSD memory card while the easyE4 is switched on.

5. Programming on the device

5.5 Transferring programs from and to a microSD memory card

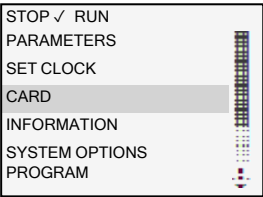
5.5.1 Configuration on a base device with a display

The program needs to be transferred using the Card menu option.

In order to be able to configure this option, the program must be in STOP mode. If it is not, the device will point this out.

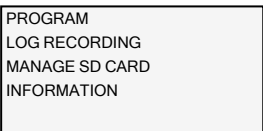
- ▶ Go to the main menu.
- ▶ Open the CARD menu option.

Tab. 77: Main menu



The device menu for the memory card will appear with additional menu options.

Tab. 78: Card



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 the microSD memory cards to FAT32 and eject them.
INFORMATION	Provides information on the card size and the amount of free space left

## 5. Programming on the device

### 5.5 Transferring programs from and to a microSD memory card

#### 5.5.1.1 PROGRAM submenu

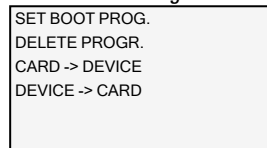
##### Requirement:

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:

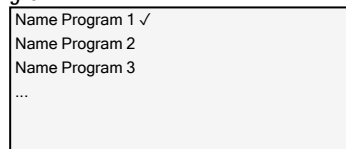
Tab. 79: *Card\Program*



##### SET BOOT PROG.

If you select this submenu, a list with the names of all the programs stored on the memory card will appear.

Tab. 80: *Card\Program\Start program*



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 in the device menu. The program will not be deleted until you select Yes and press **OK** as a confirmation.

## 5. Programming on the device

### 5.5 Transferring programs from and to a microSD memory card

#### 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 in the device menu. 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 "Functions of the microSD memory card", page 146



## 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 8). 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 8 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.

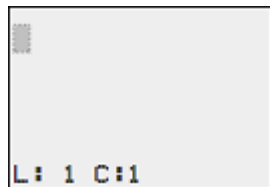
To add a function block to a circuit diagram for the first time, follow the steps below:

- Switch to the circuit diagram display by selecting *Main menu -> PROGRAMS -> CIRCUIT DIAGRAM*.

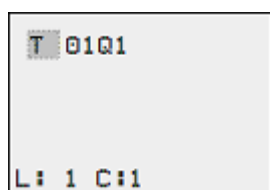
## 5. Programming on the device

### 5.6 Working with function blocks

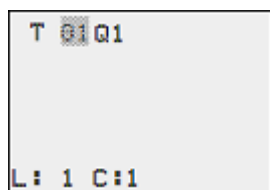
- ▶ Use the ⬆ ⬇ ⬅ ➡ buttons to move the cursor to a free contact or coil field.
- ▶ Press the **OK** button to switch to Entry mode.



- ▶ Then use the ⬆ ⬇ cursor buttons to select the function block you want (e.g., a timing relay with the T identifier).

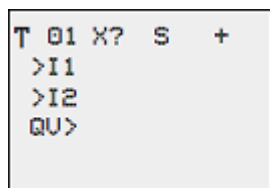


- ▶ While the function block identifier is flashing, press the **OK** button or the ➡ button to move to the function block number
- ▶ Press the **OK** button.

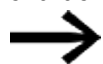


The display switches to the function block editor. Here you normally define all function block parameters. As you have reached the function block editor via the circuit diagram in this case, you can only set the basic parameters.

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.

## 5. Programming on the device

### 5.6 Working with function blocks

- ▶ 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).

5. Programming on the device

5.6 Working with function blocks

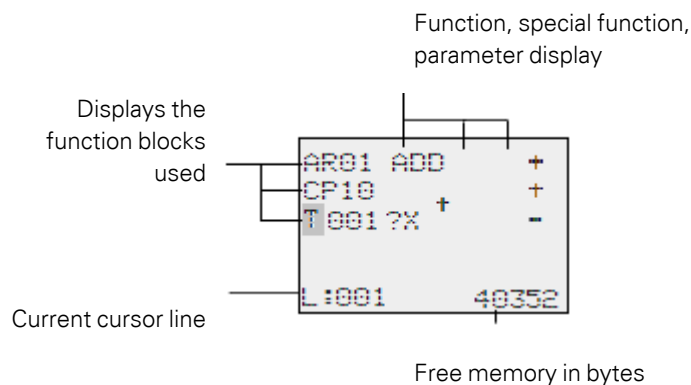
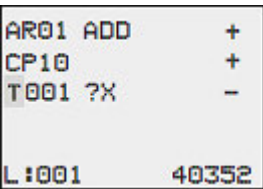


Fig. 120: Explanation of the function block list

- 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.

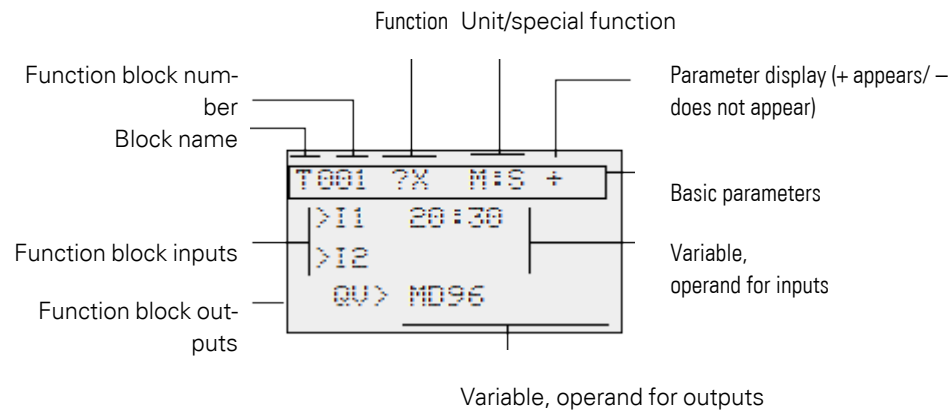


Fig. 121: Manufacturer function block display in the function block editor

#### 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:
  - ▶  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.
- ▶  button: Saves the constant immediately
- ▶ Press the  button to leave the parameter display.

 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.

## 5. Programming on the device

### 5.6 Working with function blocks



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.

```
T001 ?X M:S +  
>I1 ==:30  
>I2  
  
QV> MD96
```

The operand is deleted.

```
T001 ?X M:S +  
>I1 ==  
>I2  
  
QV> MD96
```

#### 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 processing 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.



#### 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 221

## 5. Programming on the device

### 5.6 Working with function blocks

#### 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.

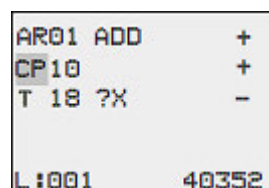
- ▶ Switch to the circuit diagram display by selecting  
*Main menu -> PROGRAMS -> CIRCUIT DIAGRAM.*
- ▶ Move the cursor in the circuit diagram to all the contact fields and the coil field in which the function block to be deleted is used and press the **DEL** button each time.

#### 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.



AR01 ADD	+
CP10	+
T 18 ?X	-

L:001 40352

- ▶ 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 651.  
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 creating 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. Programming on the device

### 5.7 Using operands in a program

### 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.

#### 5.7.1 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

### 5.7.2 Permissible operands at a glance

Tab. 81: Permissible operands

Operand	Explanation	Data width:	Data type
I	Input	1 bit	BOOL
Q	Output	1 bit	BOOL
p <sup>2)</sup>	P buttons	1 bit	BOOL
ID	diagnostic alarm	1 bit	BOOL
IA	Analog Input	32 bits	DINT
QA	Analog output	32 bits	DINT
M	Markers	1 bit	BOOL
MB	Marker byte	8 bits	BYTE
MW	Marker word	16 bits	WORD
MD	Marker double word	32 bits	DINT
LE <sup>2)</sup>	LED output	1 bit	BOOL
RN <sup>1,2)</sup>	Input bit via NET (receive)	1 bit	BOOL
SN <sup>1,2)</sup>	Output bit via NET (send)	1 bit	BOOL
N	Network marker	1 bit	BOOL
NB	Network marker byte	8 bits	BYTE
NW	Network marker word	16 bits	WORD
ND	Network marker double word	32 bits	DINT

1) Not available for visualization elements

2) Not available for cloud connection

Used for	Operand range
Local bit operands	I1...I16 <sup>1)</sup> I17...I128 Q1...Q16 <sup>1)</sup> Q17...Q128 P1...P8 M1...M512 (EDP: M1...M128) ID1...ID24 <sup>1)</sup> ID25...ID96 LE1...LE3
Local value operands	IA1...IA4 <sup>1)</sup> IA5...IA48 QA1...QA4 <sup>1)</sup> QA5...QA48 MB1...MB512 MW1...MW512 MD1...MD256
N operands bit	N1...N512 (EDP: N1...N128) xRN1...xRN32 <sup>2)</sup> xSN1...xSN32 <sup>2)</sup>
N operands value	NB1...NB64 NW1...NW32 ND1...ND16

1) base device permanently assigned

2) Not available for visualization elements

## 5. Programming on the device

### 5.7 Using operands in a program

#### 5.7.3 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:

Operands	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 - Backlight output	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
Analog inputs IA	x	x
Analog output QA	x	x
Numeric output from another QV FB	x	x

#### 5.7.4 Overview of operands 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

```
0000000000000000000000000000000010000010010bin = 412hex = 1042dez  
11111110110111001111010001000111bin = FEDCF447hex = -19073977dez
```

## 5. Programming on the device

### 5.7 Using operands in a program

#### 5.7.5 Timer constant

Timer constants are used at the function block inputs for the T and AC function blocks.

The value range of timer constants depends on the time range of the function block for which they are being used.

The moment you drag & drop the timer constant from the left pane and onto the work pane and connect it to the corresponding function block input, the timer constant will be assigned the same time range as the function block and will display a default value of 0 with the corresponding resolution.

If, for instance, the time range for the function block is configured as S - 000.000 with a resolution of 5 ms, the timer constant will be shown with a default value of 0.000 s.

#### **Quickly entering values with the keyboard**

You can enter values for timer constants using the keyboard. Please note that the values you enter must fall within the configured time range.

To enter timer constant values with the keyboard, follow the steps below:

- ▶ Select the timer constant by clicking on it.
- ▶ Enter a value with the keyboard, e.g., <9>.
- ▶ Press the Enter key to confirm – the value will be applied as the timer constant value.
- ▶ To cancel instead, simply press the ESC key.

Values that fall outside the set resolution will be rounded down automatically.

For instance, entering a value of <9> for a time constant with a time range of S - 000.000 and a resolution of 5 ms will result in the value being rounded down to 5 ms.

## 5. Programming on the device

### 5.7 Using operands in a program

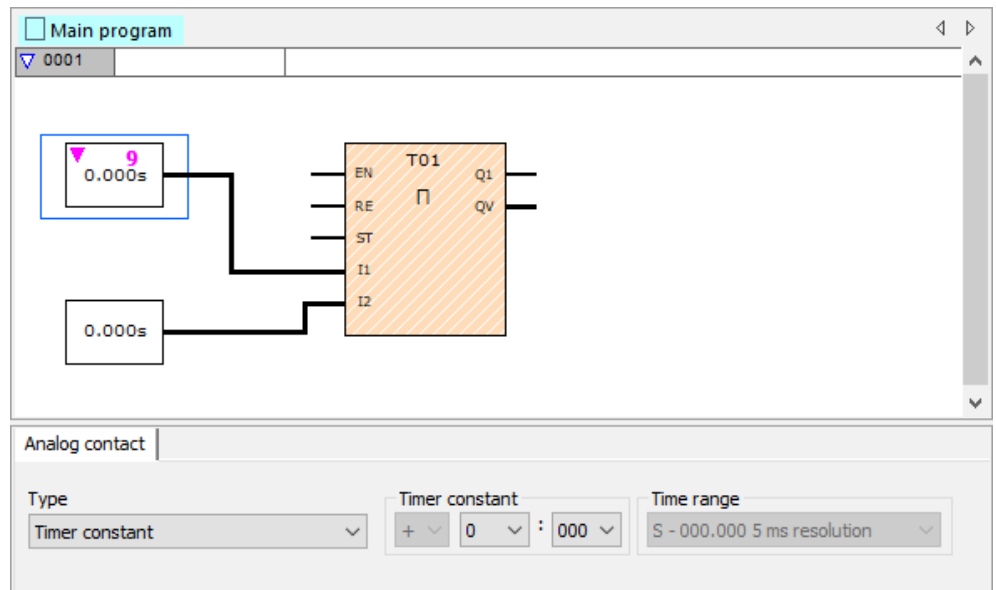


Fig. 122: Programming view: Selected time constant on function block input I1 and unconfirmed value of <9> entered with the keyboard

If you are using different time range configurations, the values will need to be entered differently. You will have to enter <t#> with the keyboard before the timer constant.

Example: For a time range of M:S - 00:00 and a resolution of 1 s, entering <t#5m10s>.

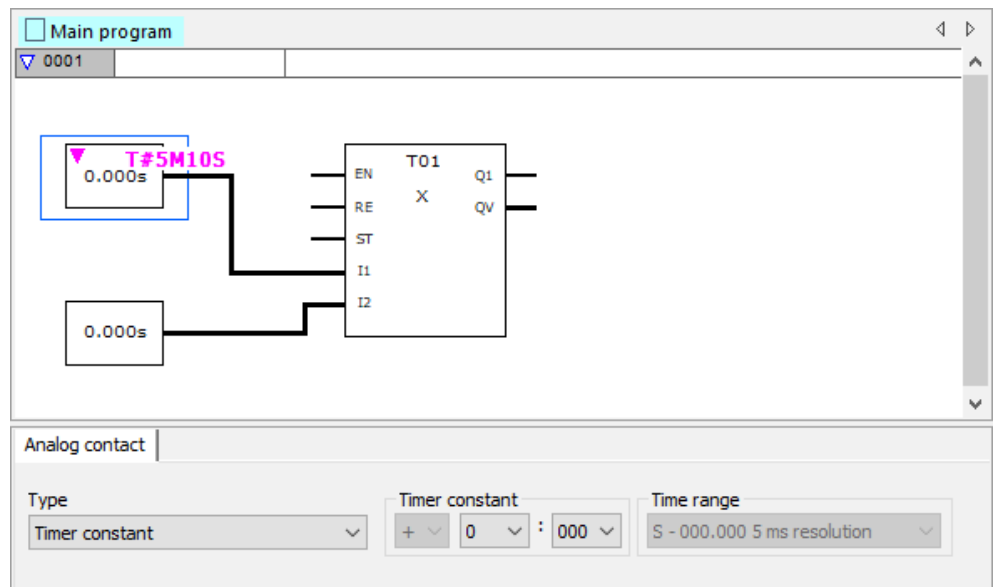


Fig. 123: Programming view: Selected timer constant on function block input I1 and unconfirmed value of <t#5m10s> entered with the keyboard

## 5. Programming on the device

### 5.7 Using operands in a program

Example: For a time range of H:M - 00:00 and a resolution of 1 min., entering <t#3h25m>.

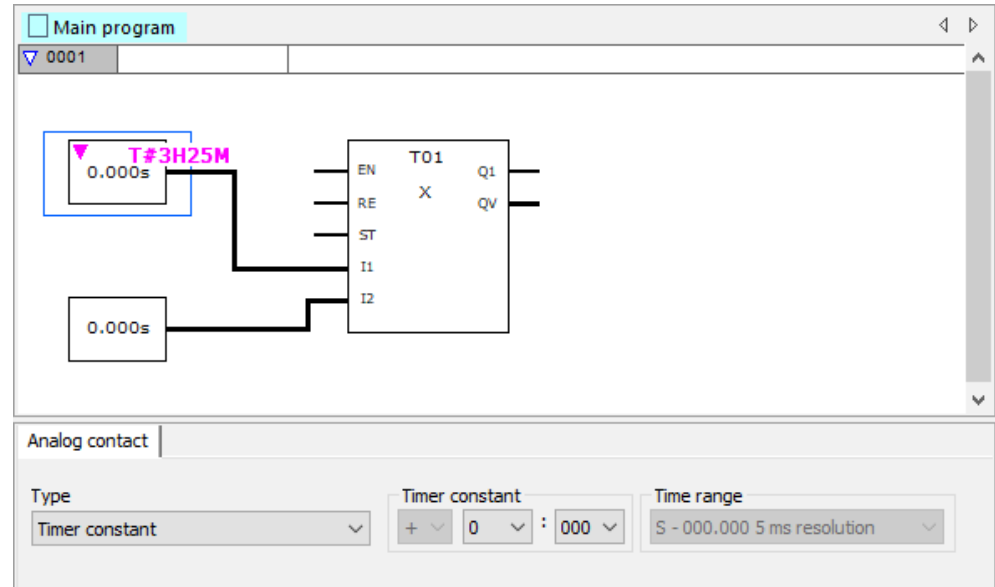


Fig. 124: Programming view: Selected timer constant on function block input I1 and unconfirmed value of <t#3h25m> entered with the keyboard

Negative times are allowed, but only for timer constants connected to the input of AC function blocks. In this case, you can enter values within the following range: -12h00m to +12h00m.



## 5. Programming on the device

### 5.7 Using operands in a program

#### Permissible time ranges for timer constants (as an input value for T or AC function blocks)

You can set the following time ranges for the function blocks:

Time range	Function block T	Function block AC
S - 000.000 5 ms resolution	✓	–
M:S - 00:00 1 s resolution	✓	–
H:M - 00:00 1 min. resolution	✓	✓

#### Changes to timer constants when the time range for a function block changes

If the time range for a function block changes, the time ranges of all timer constants connected to the function block will change as well. The values of these timer constants will then be changed accordingly, with an important limitation being the fact that the changed values will not be allowed to exceed or fall below the new time ranges. If any data or accuracy is lost, a message to this effect will be shown.

Example:

Say that the time range for a function block T is changed from H:M - 00:00 and a resolution of 1 min. to M:S - 00:00 and a resolution of 1 s.

How would the following timer constant values change?

H:M - 00:00 1 min. resolution	M:S - 00:00 1 s resolution	Note
70h 00m	0m 00s	The conversion results in 4200 minutes. However, this exceeds the max. time range limit of 99 minutes for the timer constants -> message
1h 02m	62m 00s	✓
1h 39m	99m 00s	✓
1h 40m	40m 00s	The conversion results in 100 minutes. However, this exceeds the max. time range limit of 99 minutes for the timer constants -> message

## 5. Programming on the device

### 5.7 Using operands in a program

#### 5.7.6 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 operand 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

➔ Use the following command in order to filter out multiple marker assignments. To do this, click on the *Project/Marker area assignment...* menu option

### Marker range mapping

The marker range map shows which markers are read and written to. Above all, this map shows which markers are affected by write access operations that will result in write conflicts.

➔ Before finishing a project, make sure to always check the marker range map.

If any write conflicts are shown, open the cross-reference list and use the help function to find out where the duplicate mapping is – .

In the following example, marker bytes 1 to 8 are read by a recipe function block. In addition, there is a write conflict for marker word 1.

*Project/Marker area assignment menu option*

The screenshot shows the 'Marker area assignment' window. It contains a table with columns for marker ranges (M), marker bytes (MB), marker words (MW), and marker double words (MD). The table is color-coded: green for 'Read', orange for 'Write', yellow for 'Read / Write', grey for 'Not fully usable', and red for 'Writing conflict'. A legend on the right explains these colors. The table shows that marker bytes 1 to 8 are read (green), and marker word 1 has a write conflict (red). The 'Information' column provides details for each entry, such as 'MB1.R; MW1: Writing conflict; M Not'.

	M	MB	MW	MD	Information
1	1 ... 8	1	1	1	MB1.R; MW1: Writing conflict; M Not
2	9 ... 16	2	2	2	MB2.R; ; M Not fully usable
3	17 ... 24	3	3	3	MB3.R
4	25 ... 32	4	4	4	MB4.R
5	33 ... 40	5	5	5	MB5.R
6	41 ... 48	6	6	6	MB6.R
7	49 ... 56	7	7	7	MB7.R
8	57 ... 64	8	8	8	MB8.R
9	65 ... 72	9	9	9	
10	73 ... 80	10	10	10	
11	81 ... 88	11	11	11	
12	89 ... 96	12	12	12	
13	97 ... 104	13	13	13	
14	105 ... 112	14	14	14	
15	113 ... 120	15	15	15	
16	121 ... 128	16	16	16	
17	129 ... 136	17	17	17	
18	137 ... 144	18	18	18	
19	145 ... 152	19	19	19	
20	153 ... 160	20	20	20	
21	161 ... 168	21	21	21	
22	169 ... 176	22	22	22	
23	177 ... 184	23	23	23	
24	185 ... 192	24	24	24	
25	193 ... 200	25	25	25	
26	201 ... 208	26	26	26	

Fig. 125: Marker range map with write conflict for MW1

The operand table below shows the relationships between marker bits, bytes, words, and double words in a different way.

## 5. Programming on the device

### 5.7 Using operands in a program

#### 5.7.7 Operand table

The operand table must be read as follows:

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	64...57	56...49	48...41	40...33	32...25	24...17	16...9	8...1
Byte	8	7	6	5	4	3	2	1
Word	4	3	2	1				
DWord	2	1						
Bit	128...121	120...113	112...105	104...97	96...89	88...81	80...73	72...65
Byte	16	15	14	13	12	11	10	9
Word	8	7	6	5				
DWord	4	3						
Bit	192...185	184...177	176...169	168...161	160...153	152...145	144...137	136...129
Byte	24	23	22	21	20	19	18	17
Word	12	11	10	9				
DWord	6	5						
Bit	256...249	248...241	240...233	232...225	224...217	216...209	208...201	200...193
Byte	32	31	30	29	28	27	26	25
Word	16	15	14	13				
DWord	8	7						
Bit	320...313	312...305	304...297	296...289	288...281	280...273	272...265	264...257
Byte	40	39	38	37	36	35	34	33
Word	20	19	18	17				
DWord	10	9						
Bit	384...377	376...369	368...361	360...353	352...345	344...337	336...329	328...321
Byte	48	47	46	45	44	43	42	41
Word	24	23	22	21				
DWord	12	11						
Bit	448...441	440...433	432...425	424...417	416...409	408...401	400...393	392...385
Byte	56	55	54	53	52	51	50	49
Word	28	27	26	25				
DWord	14	13						
Bit	512...505	504...497	496...489	488...481	480...473	472...465	464...457	456...449
Byte	64	63	62	61	60	59	58	57
Word	32	31	30	29				
DWord	16	15						

## 5. Programming on the device

### 5.7 Using operands in a program

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	
DWord		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	
DWord		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	
DWord		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	
DWord		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	
DWord		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	
DWord		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	
DWord		96				95				94				93		
Byte	400	399	398	397	396	395	394	393	392	391	390	389	388	387	386	385
Word	200		199		198		197		196		195		194		193	
DWord		100				99				98				97		
Byte	416	415	414	413	412	411	410	409	408	407	406	405	404	403	402	401
Word	208		207		206		205		204		203		202		201	
DWord		104				103				102				101		
Byte	432	431	430	429	428	427	426	425	424	423	422	421	420	419	418	417
Word	216		215		214		213		212		211		210		209	
DWord		108				107				106				105		
Byte	448	447	446	445	444	443	442	441	440	439	438	437	436	435	434	433
Word	224		223		222		221		220		219		218		217	
DWord		112				111				110				109		
Byte	464	463	462	461	460	459	458	457	456	455	454	453	452	451	450	449
Word	232		231		230		229		228		227		226		225	
DWord		116				115				114				113		

## 5. Programming on the device

### 5.7 Using operands in a program

Byte	480	479	478	477	476	475	474	473	472	471	470	469	468	467	466	465
Word	240		239		238		237		236		235		234		233	
DWord		120				119				118				117		
Byte	496	495	494	493	492	491	490	489	488	487	486	485	484	483	482	481
Word	248		247		246		245		244		243		242		241	
DWord		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		
Word	384		383		382		381		380		379		378		377	
DWord		192				191				190				189		
Word	392		391		390		389		388		387		386		385	
DWord		196				195				194				193		
Word	400		399		398		397		396		395		394		393	
DWord		200				199				198				197		
Word	408		407		406		405		404		403		402		401	
DWord		204				203				202				201		
Word	416		415		414		413		412		411		410		409	
DWord		208				207				206				205		
Word	424		423		422		421		420		419		418		417	
DWord		212				211				210				209		
Word	432		431		430		429		428		427		426		425	
DWord		216				215				214				213		
Word	440		439		438		437		436		435		434		433	
DWord		220				219				218				217		
Word	448		447		446		445		444		443		442		441	
DWord		224				223				222				221		
Word	456		455		454		453		452		451		450		449	
DWord		228				227				226				225		
Word	464		463		462		461		460		459		458		457	
DWord		232				231				230				229		
Word	472		471		470		469		468		467		466		465	
DWord		236				235				234				233		
Word	480		479		478		477		476		475		474		473	
DWord		240				239				238				237		
Word	488		487		486		485		484		483		482		481	
DWord		244				243				242				241		
Word	496		495		494		493		492		491		490		489	
DWord		248				247				246				245		
Word	504		503		502		501		500		499		498		497	
DWord		252				251				250				249		
Word	512		511		510		509		508		507		506		505	
DWord		256				255				254				253		

### 5.7.8 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 651.

### 5.7.9 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:

function block	Marker range	
UF	16 marker double words	→ "UF - User function block", page 597
IE	32 marker bits	→ "IE - Edge-controlled interrupt", page 583
IC		→ "IC - Counter-controlled interrupt", page 572
IT		→ "IT - Time-controlled interrupt function block", page 589





## 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.

Following is a description of each individual function block that goes over the number of instances allowed; how the function block works; and which inputs, outputs, and operating modes it has.

### Function block value ranges

Each description gives the value ranges for the analog inputs and outputs of each function block. Analog function block inputs and outputs are linked to operands or constants of data type DWORD, meaning that operands can theoretically be assigned values of -2147483648 to +2147483647, although value processing is limited to a more reasonable value range. Assigning higher values will set the operands to the relevant maximum or minimum value for the corresponding value range.

### Manufacturer function blocks

Manufacturer function blocks are available in easySoft 8 and directly on the device  
Timer modules

HW - Weekly timer (Hour Week)	→ page 244
HY - Year time switch (Hora Year)	→ page 292
OT - Operating hours counter	→ page 264
RC - Real-time clock	→ page 269
T - Timing relay	→ page 272
WT - Weekly timer (WeekTable)	→ page 292
YT - Year time switch (Year Table)	→ page 284
AC - Astronomic clock	→ page 296

### Counter Function Blocks

C - Counter Relay	→ page 305
CF - Frequency counter	→ page 311
CH - High-speed counter	→ page 317
CI - Incremental Counter	→ page 323

## 6. Function blocks

### Arithmetic and analog function blocks

A - Analog value comparator	→ page 330
AR - Arithmetic	→ page 336
AV - Average	→ page 342
CP – Comparator	→ page 350
LS - Value scaling	→ page 354
MM - Min-/Max function	→ page 359
PM - Performance map	→ page 362
PW - Pulse width modulation	→ page 368

### Open-loop and closed-loop function blocks

DC - PID controller	→ page 375
FT - PT1-Signal smoothing filter	→ page 382
PO - Pulse output	→ page 387
TC - Three step controller	→ page 402
VC - Value limitation	→ page 407

### Data and register function blocks

BC - Block comparison	→ page 411
BT - Block transfer	→ page 419
DB - Data function block	→ page 425
ED - EdgeDetector	→ page 430
FF - Flip-flop	→ page 434
MX - Data multiplexer	→ page 437
RE - Recipe records	→ page 441
SR - Shift register	→ page 447
TB - Table function	→ page 455

### NET Function Blocks

GT - Get values from NET	→ page 460
PT - Put values to NET	→ page 464
SC - Synchronizing clock via NET	→ page 468

### Other function blocks

AL - Alarm function block	→ page 472
BV - Boolean operation	→ page 477
D - Text display	→ page 481
D - Text display editor	→ page 491
DL - Data logger	→ page 510
JC - Conditional jump	→ page 524
MC - Acyclical Modbus TCP request	→ page 531
MR - Master Reset	→ page 543
MU - Acyclical Modbus RTU request	→ page 547
NC - Numerical converter	→ page 563
ST - Set cycle time	→ page 569

### **Interrupt function blocks**

Interrupt function blocks are available only in easySoft 8

- IC - Counter-controlled interrupt → page 572
- IE - Edge-controlled interrupt → page 583
- IT - Time-controlled interrupt function block → page 589

### **User function block – used to create custom function blocks**

User function blocks are only available in easySoft 8.

- UF - User function block → page 597

6. Function blocks  
6.1 Manufacturer function blocks

6.1 Manufacturer function blocks

6.1.1 Timer modules

6.1.1.1 HW - Weekly timer (Hour Week)

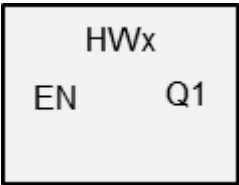
easyE4 devices feature a real-time clock with a date and time functionality. When combined with the HW, HY or WT, YT function blocks, this real-time clock makes it possible to implement the functionality of a weekly timer and year time switch.

→ Section "Time and Date setting", page 659

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 = Thu, 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 <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND - NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> 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 <sup>2)</sup> NET station n	x
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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 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 + 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.

Tab. 82: 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. 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. 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. 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-up of real-time clock", page 886



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.

6. Function blocks

6.1 Manufacturer function blocks

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.

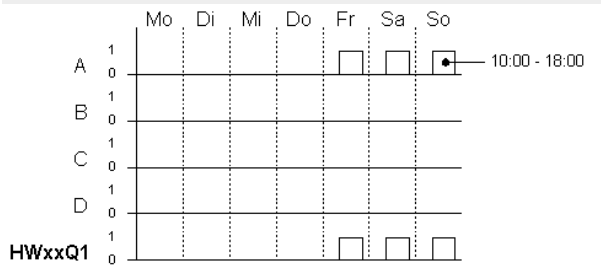


Fig. 126: Signal diagram

The HW time switch must be assigned the following parameters:

Fig. 127: Programming view Weekly timer parameters tab



## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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.

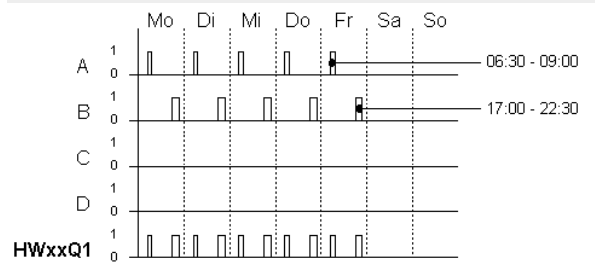


Fig. 128: Signal diagram

The HW time switch must be assigned the following parameters:

The screenshot shows the 'Weekly timer parameters' programming view. It includes a 'HW' dropdown set to '1' and a 'Comment' field. A checkbox 'Function block release by EN is necessary' is unchecked. Below are four channel configuration panels (A, B, C, D). Channel A is configured with Day: Mon, DY1: Mon, DY2: Fri, ON: 06:30, and OFF: 09:00. Channel B is configured with Day: Mon, DY1: Mon, DY2: Fri, ON: 17:00, and OFF: 22:30. Channels C and D are configured with Day: --, DY1: --, DY2: --, ON: --:--, and OFF: --:--.

Fig. 129: Programming view Weekly timer parameters tab

6. Function blocks

6.1 Manufacturer function blocks

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.

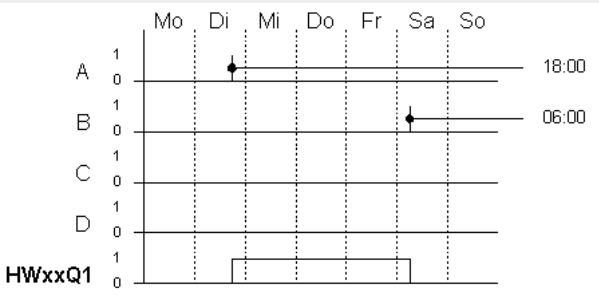


Fig. 130: Signal diagram

The HW time switch must be assigned the following parameters:

Weekly timer parameters

HW: 1 Comment:

☒ Function block release by EN is necessary

Channel	Day	DY1	DY2	ON: Hour Minute	OFF: Hour Minute
Channel A	Tue	--	--	18 00	-- --
Channel B	Sat	--	--	-- --	6 00
Channel C	--	--	--	-- --	-- --
Channel D	--	--	--	-- --	-- --

Fig. 131: Programming view Weekly timer parameters tab

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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.

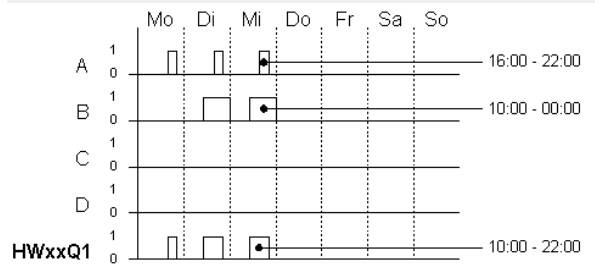
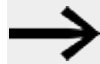


Fig. 132: Signal diagram



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:

Weekly timer parameters

HW: 1 Comment:

☒ Function block release by EN is necessary

Channel	Day	ON: Hour	ON: Minute	OFF: Hour	OFF: Minute
Channel A	Mon	16	00	22	00
Channel B	Tue	10	00	--	--
Channel B	Wed	--	--	--	--
Channel C	Tue	--	--	--	--
Channel C	Wed	--	--	--	--
Channel D	Tue	--	--	--	--
Channel D	Wed	--	--	--	--

Parameter display: + Call enabled

Fig. 133: Programming view Weekly timer parameters tab Settings Time overlap

6. Function blocks

6.1 Manufacturer function blocks

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.

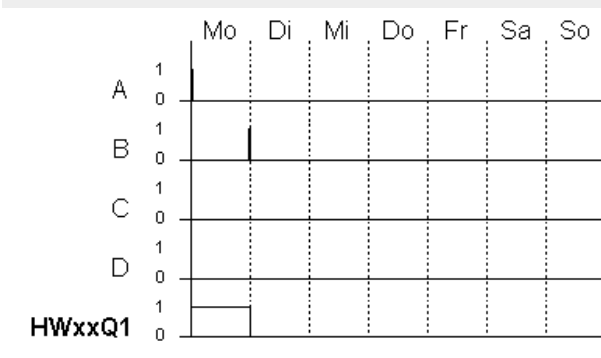


Fig. 134: Signal diagram

The HW time switch must be assigned the following parameters:

Weekly timer parameters

HW: 1 Comment:

☒ Function block release by EN is necessary

Channel A	Channel B	Channel C	Channel D
Day: Mon	Day: Tue	Day: --	Day: --
DY1: Mon	DY1: Tue	DY1: --	DY1: --
DY2: --	DY2: --	DY2: --	DY2: --
ON: 00:00	ON: --:--	ON: --:--	ON: --:--
OFF: --:--	OFF: 00:00	OFF: --:--	OFF: --:--
Parameter display	Parameter display	Parameter display	Parameter display
+ Call enabled	+ Call enabled	+ Call enabled	+ Call enabled

Fig. 135: Programming view Weekly timer parameters tab - 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:

Weekly timer parameters

HW: 1 Comment:

☒ Function block release by EN is necessary

Channel A	Channel B	Channel C	Channel D
Day	Day	Day	Day
DY1: Mon	DY1: Tue	DY1: --	DY1: --
DY2: --	DY2: --	DY2: --	DY2: --
ON: 22 00	ON: -- --	ON: -- --	ON: -- --
OFF: -- --	OFF: 6 00	OFF: -- --	OFF: -- --
Parameter display	Parameter display	Parameter display	Parameter display
+ Call enabled	+ Call enabled	+ Call enabled	+ Call enabled

Fig. 136: Programming view Weekly timer parameters tab

#### See also

- Section "HY - Year time switch (Hora Year)", page 254
- Section "OT - Operating hours counter ", page 264
- Section "RC - Real-time clock", page 269
- Section "T - Timing relay", page 272
- Section "WT - Weekly timer (WeekTable)", page 292
- Section "YT - Year time switch (Year Table)", page 284
- Section "AC - Astronomic clock ", page 296

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 659

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 8.

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.



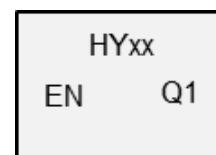
The parameters for the switch-on and switch-off times for recurring intervals are configured in one single channel for each.

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. However, the time switch relays will no longer switch. The contacts are kept open when de-energized.

Information on the battery back-up time are provided on → Section "Back-up of real-time clock", page 886



#### **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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> 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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

##### Function block outputs

	Description	Note
<b>(bit)</b>		
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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

Parameter set	Description	Note
<input checked="" type="checkbox"/> 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 + 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 8 Programming view by clicking on it, a table with the various parameters will appear under the tab.

## 6. Function blocks

### 6.1 Manufacturer function blocks

Year time switch parameters

HY: 1 Comment:

☐ Function block release by EN is necessary

Channel A

Day Month Year

ON: -- -- 20

Day Month Year

OFF: -- -- 27

Parameter display

+ Call enabled

Channel B

Day Month Year

ON: -- -- --

Day Month Year

OFF: -- -- --

Parameter display

+ Call enabled

Channel C

Day Month Year

ON: -- -- --

Day Month Year

OFF: -- -- --

Parameter display

+ Call enabled

Channel D

Day Month Year

ON: -- -- --

Day Month Year

OFF: -- -- --

Parameter display

+ Call enabled

Fig. 137: Year time switch parameters tab with an 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 260 .



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 260 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:

## 6. Function blocks

### 6.1 Manufacturer function blocks

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 260 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 260 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 263 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.

6. Function blocks

6.1 Manufacturer function blocks

Other

**Retention** - The function block does not recognize retentive data.

**Examples HY - Year time switch in easySoft 8**

**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*

Fig. 138: 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*

Fig. 139: 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

The HY year time switch must be assigned the following parameters:

*Programming view/HY01/Year time switch parameters tab*

Year time switch parameters

HY: 1 Comment:

☐ Function block release by EN is necessary

Channel A

Day Month Year

ON: 1 -- --

Day Month Year

OFF: 28 -- --

Parameter display

+ Call enabled

Channel B

Day Month Year

ON: -- -- --

Day Month Year

OFF: -- -- --

Parameter display

+ Call enabled

Channel C

Day Month Year

ON: -- -- --

Day Month Year

OFF: -- -- --

Parameter display

+ Call enabled

Channel D

Day Month Year

ON: -- -- --

Day Month Year

OFF: -- -- --

Parameter display

+ Call enabled

Fig. 140: 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*

Year time switch parameters

HY: 1 Comment:

☐ Function block release by EN is necessary

Channel A

Day Month Year

ON: 25 12 --

Day Month Year

OFF: 27 12 --

Parameter display

+ Call enabled

Channel B

Day Month Year

ON: -- -- --

Day Month Year

OFF: -- -- --

Parameter display

+ Call enabled

Channel C

Day Month Year

ON: -- -- --

Day Month Year

OFF: -- -- --

Parameter display

+ Call enabled

Channel D

Day Month Year

ON: -- -- --

Day Month Year

OFF: -- -- --

Parameter display

+ Call enabled

Fig. 141: Entry screen in the programming software

6. Function blocks

6.1 Manufacturer function blocks

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

Fig. 142: 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

Fig. 143: 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

Fig. 144: 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

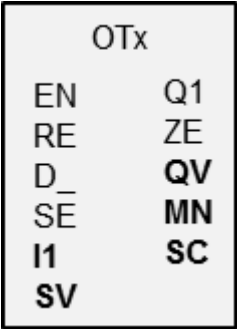
- Section "HW - Weekly timer (Hour Week)", page 244
- Section "OT - Operating hours counter ", page 264
- Section "RC - Real-time clock", page 269
- Section "T - Timing relay", page 272
- Section "WT - Weekly timer (WeekTable)", page 292
- Section "YT - Year time switch (Year Table)", page 284
- Section "AC - Astronomic clock ", page 296

6. Function blocks  
6.1 Manufacturer function blocks

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 rising 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
<b>(bit)</b>		
EN	1: Activates the function block.	
RE	Reset 1: Resets the actual counter value back to zero.	
D_	Count direction 1: down counting 0: up counting	Integer value range: 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	
<b>(DWord)</b>		
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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> NET station n	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

	Description	Note
<b>(bit)</b>		
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 1: If operating hours counter = 0	
<b>(DWord)</b>		
QV	Actual operating hours counter value Displayed in hours	Integer 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET


## **6. Function blocks**

### **6.1 Manufacturer function blocks**

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Parameter set

Configuration/time range	Description	Note
<input checked="" type="checkbox"/> 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 	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

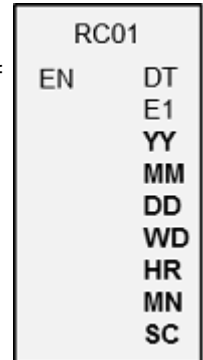
- Section "AC - Astronomic clock ", page 296
- Section "HW - Weekly timer (Hour Week)", page 244
- Section "HY - Year time switch (Hora Year)", page 254
- Section "RC - Real-time clock", page 269
- Section "T - Timing relay", page 272
- Section "WT - Weekly timer (WeekTable)", page 292
- Section "YT - Year time switch (Year Table)", page 284

#### 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 that 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 <input checked="" type="checkbox"/> 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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

	Description	Note
<b>(bit)</b>		
DT	0: The output value is standard time 1: The output value is daylight saving time	
E1	Error 0: error-free operation 1: The value is invalid, since it is before the device's initial date	
<b>(DWord)</b>		
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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

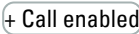
You can assign the following operands to the function block outputs that are bit outputs:

Assigning operands	Bit outputs
M – Markers	x
SN – Output bit via NET (send) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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 	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 "AC - Astronomic clock ", page 296
- Section "HW - Weekly timer (Hour Week)", page 244
- Section "HY - Year time switch (Hora Year)", page 254
- Section "OT - Operating hours counter ", page 264
- Section "T - Timing relay", page 272
- Section "WT - Weekly timer (WeekTable)", page 292
- Section "YT - Year time switch (Year Table)", page 284

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 6.1.1.5 T - Timing relay

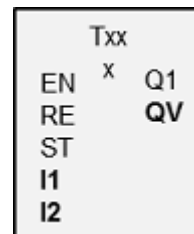
##### General

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, you can configure three time ranges: Seconds:Milliseconds, Minutes:Seconds, Hours:Minutes

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
<b>(bit)</b>		
EN	1: Activates the function block. 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. The only operating mode in which detecting a rising edge is enough is the <b>Single pulse</b> mode. In this case, the function block will be activated for one cycle and started for this mode.	
RE	Reset 1: Resets the timing relay to a value of zero (reset coil)	
ST	Stop coil 1: stops the timing relay. The started time will cease to time out whilst the ST is set to 1. The stopped time will continue to time out if the signal is reset to 0. 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.	
<b>(DWord)</b>		
I1	Time setpoint 1	Integer value range: S: 1...999995 ms, resolution 5 ms M:S: 1... 5999 s, Resolution 1 s 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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Operating mode

This parameter defines the switch function of the timing relay.

Device parameters	easySoft 8 Operating mode	Note
X	On-delayed	
?X	On-delayed with random time	
â	Off-delayed	
?â	Off-delayed with random time	
Xâ	On/off delayed	There are two time setpoints to be configured.
?X#	On-/off-delayed with random time	With random time, 2 time setpoints
û	Single pulse	Normalizes input signals of different pulse lengths to a fixed pulse length at the switch contact of the timing relay.
Û	Flashing Time values: S1=Pulse time, S2= Pause time;	Time values Two time reference values need to be configured. I1=Pulse time, I2= Pause time; Synchronous flashing: I1 = I2 Mark-to-space ratio = 1:1  Asynchronous flashing: I1 ≠ I2 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
<b>(bit)</b>		
Q1	Switching contact	
<b>(DWord)</b>		
QV	Elapsed actual time in RUN mode	Integer value range: 0 to max. 99990 in time range: seconds; milliseconds; hours depending on configured time range.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Parameter set

Configuration/time range	Description	Note
S	<b>Seconds:Milliseconds</b> Configurable as a constant: 00.005 to 999.995 (s.ms)	Resolution: 5 ms
M : S	<b>Minutes:Seconds</b> Configurable as a constant: 00:01 to 99:59 (min:s)	Resolution: 1 s
H : M	<b>Hours:Minutes</b> Configurable as a constant: 00:01 to 99:59 (h:min)	Resolution: 1 min
Parameter display + 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	00:100	01:40	01:40
300	00:300	05:00	05:00
500	00:500	08:20	08:20
4095 (Maximum)	04:095	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,



Note: 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 relay switches randomly within the setpoint value range.

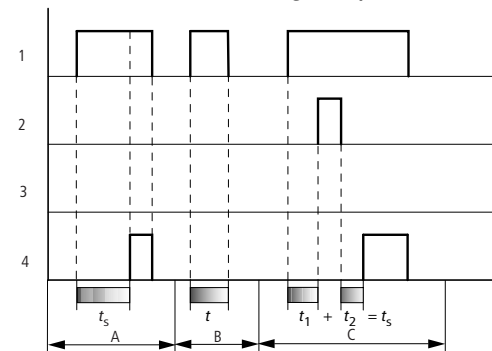


Fig. 145: Signal diagram of timing relay, on-delayed (with and without random switching)

1: Trigger coil T..EN

2: Stop coil T..ST

3: Reset coil T..RE

4: Switching contact (N/O contact) T...Q1

$t_s$ : 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

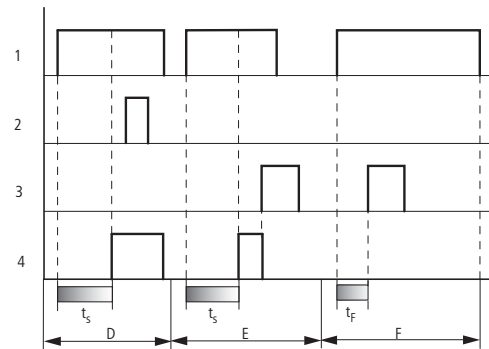


Fig. 146: Signal diagram of timing relay, on-delayed (with and without random switching)

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 relay 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

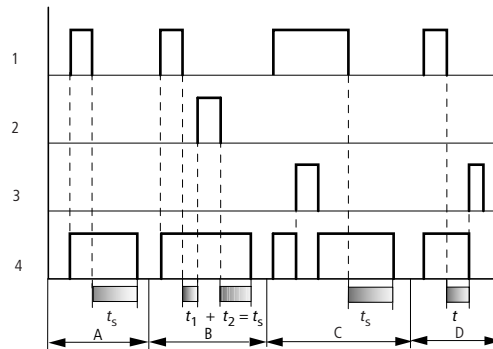


Fig. 147: Signal diagram of timing relay, off-delayed (with/without random switching, with/without re-triggering)

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 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.

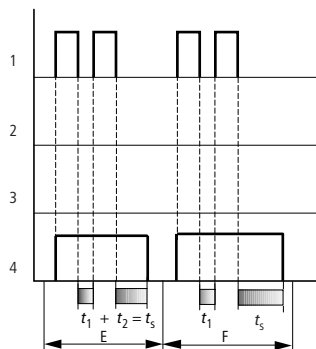


Fig. 148: Signal diagram of timing relay, off-delayed (with/without random switching, with/without re-triggering)

Range E: The trigger coil drops out twice.

The set time  $t_s$  consists of  $t_1$  plus  $t_2$  (switch function not retriggerable).

Range F: The trigger coil drops out twice. The actual time  $t_1$  is cleared and the set time  $t_s$  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

## 6. Function blocks

### 6.1 Manufacturer function blocks

The contact of the timing relay switches randomly within the setpoint value range.

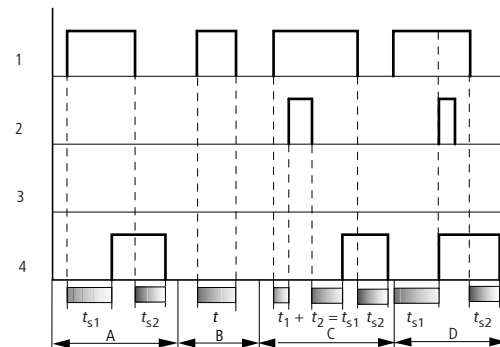


Fig. 149: Operational diagrams timing relay, on and off delayed

1: Trigger coil T..EN

2: Stop coil T..ST

3: Reset coil T..RE

4: Switching contact (N/O contact) T...Q1

$t_{s1}$ : Pick-up time

$t_{s2}$ : 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

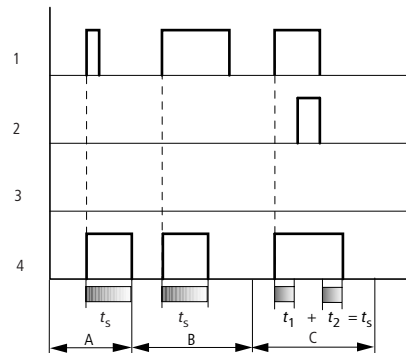


Fig. 150: Signal diagram timing relay, single pulse 1

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.

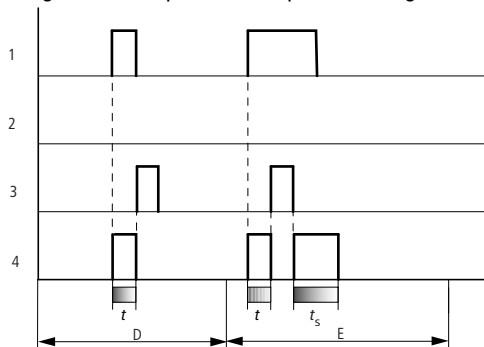


Fig. 151: Signal diagram timing relay, single pulse 2

• 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..

6. Function blocks  
6.1 Manufacturer function blocks

How the timing relay works with the flashing operating mode, synchronous and asynchronous

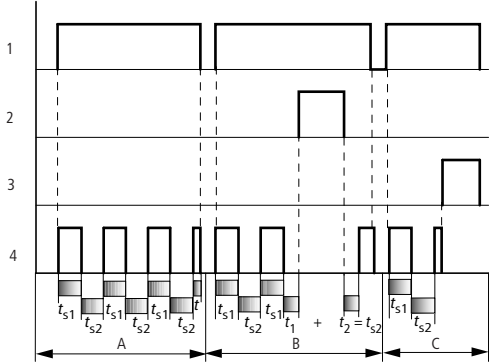


Fig. 152: Signal diagram timing relay, single pulse

- 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. The 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) If the counter actual value is greater than the maximum permissible setpoint of the configured time range, the setpoint will be limited to this maximum value. Example: You have configured the time range M: S and the counter actual value is 31333. The device limits 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-----Ä T 02EN  
M 42-----Ä T 02RE  
M 43-----Ä T 02ST
```

Fig. 153: 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, another marker activates the stop coil.

```
T 02Q1-----Ä Q 01
```

Fig. 154: Wiring of the function block contact

The signal from the function block will go directly to the device output.

#### See also

- Section "AC - Astronomic clock ", page 296
- Section "HW - Weekly timer (Hour Week)", page 244
- Section "HY - Year time switch (Hora Year)", page 254
- Section "OT - Operating hours counter ", page 264
- Section "RC - Real-time clock", page 269
- Section "WT - Weekly timer (WeekTable)", page 292
- Section "YT - Year time switch (Year Table)", page 284

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 659

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 - Yearly timer function block.

Base devices provide 32 yearly timer YT01...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

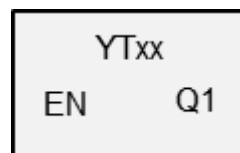
Variable holidays other than Easter cannot be selected.

##### 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.



If channels overlap, the OFF signal will override the ON signal of a different channel.



#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> 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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
<b>(bit)</b>		
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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

Parameter set	Description	Note
<input checked="" type="checkbox"/> 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 - H	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	Constants can be edited on the	

## 6. Function blocks

### 6.1 Manufacturer function blocks

Parameter set	Description	Note
+ Call enabled	device, as can function block parameters when using the EDP programming language.	
Simulation possible		

#### Parameterization

If you select the function block in the easySoft 8 Programming view by clicking on it, a table with the various parameters will appear under the tab.

Active	Channel	Mode	ON (00:00 o'clock)	OFF (24:00 o'clock)	Parameter display
<input checked="" type="checkbox"/>	A	Fixed date	01.01.2010	03.01.2020	+ Call enabled
<input checked="" type="checkbox"/>	B	Fixed date for every year	01.01.	31.12.	+ Call enabled
<input checked="" type="checkbox"/>	C	Weekday rule	Go to First Sunday of January	02 On time day(s)	+ Call enabled
<input checked="" type="checkbox"/>	D	Easter rule	Easter Sunday	Easter Sunday	+ Call enabled
<input type="checkbox"/>	E	---	---	---	---
<input type="checkbox"/>	F	---	---	---	---
<input type="checkbox"/>	G	---	---	---	---
<input type="checkbox"/>	H	---	---	---	---

Fig. 155: 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Other

**Retention** - The function block does not recognize retentive data.

#### Signal overriding

If channels overlap, the OFF signal will override the ON signal of a different channel.  
Examples with year time switch overlapping

##### Example 1:

If channel A is configured from 01/01 (ON) to 05/01 (OFF) and channel B is configured in such a way that it will switch on 03/01 (ON) and 03/01 (OFF), this means that there is an overlap on March 1st.

The result: The function block's output will be activated from 01/01 to 03/01 and deactivated for the rest of the year, since an OFF signal will come from channel B starting on March 1st.

##### Example 2:

If channel A is configured from January 1st (ON) to January 1st (OFF) (refer to the "Select year range" example) and channel B is configured from 01/02 (ON) to 01/02 (OFF), the output of the YT function block will be activated on 01/01 and deactivated for the rest of the year on 01/02 (provided that not additional channels have been configured), since an OFF signal will come from channel B starting on January 2nd.



#### Example with overlapping time ranges

If there are overlapping switching intervals, the function block may switch off output Q1 earlier if the switching-off moment for a different channel takes place before the configured OFF date. In the following example, Q1 is switched on every first Monday of January and then switched off on the following Wednesday. However, if the first Monday of a year falls on 01/01, the weekday rule for channel A will be overwritten by channel B and Q1 will be switched off on Tuesday already.

#### Programming view/YT01/Year time switch parameters tab

Year time switch (new) - Parameter

YT:  Comment:

☐ Function block release by EN is necessary

Active	Channel	Mode	ON (00:00 o'clock)	OFF (24:00 o'clock)	Parameter display
<input checked="" type="checkbox"/>	A	Weekday rule	Go to First Monday of January	03 On time day(s)	- Call disabled
<input checked="" type="checkbox"/>	B	Fixed date for every year	01.01.	02.01.	- Call disabled
<input type="checkbox"/>	C	---	---	---	---
<input type="checkbox"/>	D	---	---	---	---
<input type="checkbox"/>	E	---	---	---	---
<input type="checkbox"/>	F	---	---	---	---
<input type="checkbox"/>	G	---	---	---	---
<input type="checkbox"/>	H	---	---	---	---

Fig. 156: Entry screen in the programming software

#### Examples YT - Year time switch in easySoft 8

##### 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 parameters tab

Year time switch (new) parameters

YT:  Comment:

☐ Function block release by EN is necessary

Active	Channel	Mode	ON (00:00 o'clock)	OFF (24:00 o'clock)	Parameter display
<input checked="" type="checkbox"/>	A	Fixed date	01.01.2020	31.12.2027	+ Call enabled
<input type="checkbox"/>	B	---	---	---	---
<input type="checkbox"/>	C	---	---	---	---
<input type="checkbox"/>	D	---	---	---	---
<input type="checkbox"/>	E	---	---	---	---
<input type="checkbox"/>	F	---	---	---	---
<input type="checkbox"/>	G	---	---	---	---
<input type="checkbox"/>	H	---	---	---	---

Fig. 157: Entry screen in the programming software

6. Function blocks

6.1 Manufacturer function blocks

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 parameters tab

Year time switch (new) parameters

YT: 1 Comment:

☐ Function block release by EN is necessary

Active	Channel	Mode	ON (00:00 o'clock)	OFF (24:00 o'clock)	Parameter display
<input checked="" type="checkbox"/>	A	Fixed date for every year	01.03.	31.10.	+ Call enabled
<input type="checkbox"/>	B	---			---
<input type="checkbox"/>	C	---			---
<input type="checkbox"/>	D	---			---
<input type="checkbox"/>	E	---			---
<input type="checkbox"/>	F	---			---
<input type="checkbox"/>	G	---			---
<input type="checkbox"/>	H	---			---

Fig. 158: 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 parameters tab

Year time switch (new) parameters

YT: 1 Comment:

☐ Function block release by EN is necessary

Active	Channel	Mode	ON (00:00 o'clock)	OFF (24:00 o'clock)	Parameter display
<input checked="" type="checkbox"/>	A	Fixed date for every year	05.12.	27.12.	+ Call enabled
<input type="checkbox"/>	B	---			---
<input type="checkbox"/>	C	---			---
<input type="checkbox"/>	D	---			---
<input type="checkbox"/>	E	---			---
<input type="checkbox"/>	F	---			---
<input type="checkbox"/>	G	---			---
<input type="checkbox"/>	H	---			---

Fig. 159: 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 parameters tab

Year time switch (new) parameters

YT: 1 Comment:

☐ Function block release by EN is necessary

Active	Channel	Mode	ON (00:00 o'clock)	OFF (24:00 o'clock)	Parameter display
<input checked="" type="checkbox"/>	A	Fixed date for every year	01.05.	01.11.	+ Call enabled
<input type="checkbox"/>	B	---	---	---	---
<input type="checkbox"/>	C	---	---	---	---
<input type="checkbox"/>	D	---	---	---	---
<input type="checkbox"/>	E	---	---	---	---
<input type="checkbox"/>	F	---	---	---	---
<input type="checkbox"/>	G	---	---	---	---
<input type="checkbox"/>	H	---	---	---	---

Fig. 160: 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 parameters tab

Year time switch (new) parameters

YT: 1 Comment:

☐ Function block release by EN is necessary

Active	Channel	Mode	ON (00:00 o'clock)	OFF (24:00 o'clock)	Parameter display
<input checked="" type="checkbox"/>	A	Fixed date for every year	09.06.	16.06.	+ Call enabled
<input checked="" type="checkbox"/>	B	Fixed date for every year	09.07.	16.07.	+ Call enabled
<input checked="" type="checkbox"/>	C	Fixed date for every year	09.08.	16.08.	+ Call enabled
<input checked="" type="checkbox"/>	D	Fixed date for every year	09.09.	16.09.	+ Call enabled
<input checked="" type="checkbox"/>	E	Fixed date for every year	09.10.	16.10.	+ Call enabled
<input type="checkbox"/>	F	---	---	---	---
<input type="checkbox"/>	G	---	---	---	---
<input type="checkbox"/>	H	---	---	---	---

Fig. 161: Entry screen in the programming software

#### See also

- Section "HW - Weekly timer (Hour Week)", page 244
- Section "HY - Year time switch (Hora Year)", page 254
- Section "OT - Operating hours counter ", page 264
- Section "RC - Real-time clock", page 269
- Section "T - Timing relay", page 272
- Section "WT - Weekly timer (WeekTable)", page 292
- Section "AC - Astronomic clock ", page 296

6. Function blocks

6.1 Manufacturer function blocks

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 659

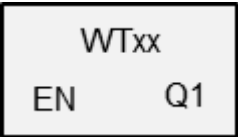
The AC manufacturer function block, Astronomic clock, can be used to program switching operations based on sunrise and sunset times. In order for this to work properly, the settings for the device clock and the device location's time zone and geographic coordinates must be correctly selected in this tab.

General

This function block is an enhanced version of the existing HW - Weekly timer function block.

Base devices provide 32 weekly timer WT01...WT32 (WeekTable). 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 <input checked="" type="checkbox"/> 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

Parameter set	Description	Note
<input checked="" type="checkbox"/> 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 - H	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 + 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 8 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
<input checked="" type="checkbox"/>	A	✓	✓	✓	✓	✓			12:00	ON	- Call disabled
<input checked="" type="checkbox"/>	B	✓	✓	✓	✓	✓			18:00	OFF	- Call disabled
<input type="checkbox"/>	C								--:--	---	---
<input type="checkbox"/>	D								--:--	---	---
<input type="checkbox"/>	E								--:--	---	---
<input type="checkbox"/>	F								--:--	---	---
<input type="checkbox"/>	G								--:--	---	---
<input type="checkbox"/>	H								--:--	---	---

Fig. 162: Weekly timer (new) parameters tab with an example

## 6. Function blocks

### 6.1 Manufacturer function blocks

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, while channel B switches off.

#### See also

- Section "HW - Weekly timer (Hour Week)", page 244
- Section "HY - Year time switch (Hora Year)", page 254
- Section "OT - Operating hours counter ", page 264
- Section "RC - Real-time clock", page 269
- Section "T - Timing relay", page 272
- Section "YT - Year time switch (Year Table)", page 284
- Section "AC - Astronomic clock ", page 296

6. Function blocks

6.1 Manufacturer function blocks

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 8, make sure that you are using firmware version 1.10 or higher for the project.

General

The astronomic clock function block makes it possible to control your system with absolute precision, e.g., after sunrise and sunset times.

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.

ACxx	
EN	Q1
O1	E1
O2	T1
	T2
	T3
	T4

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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		
01	Offset for calculating the sunrise time, in minutes	Integer value range: -720...+720
02	Offset for calculating the sunset time, in minutes	Value range when using a timer constant: -12h 00m...+12h 00m

##### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
<b>(bit)</b>		
Q1	1: During the time between sunrise and sunset	
E1	Error 1: If the device location's latitude exceeds the value range – please refer to <i>Project view/C-lock tab</i>  or if O1, O2 exceeds the value range.	Integer value range starting from the prime meridian:  Longitude -180...+180 (W...O)  Latitude -89.899...+89.899 (S...N) (-89°54'...+89°54')
<b>(DWord)</b>		
T1	Hours in the switch-on time calculated on the basis of the calculated sunrise and the value at O1	Integer value range: 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: 0...59
T3	Hours in the switch-off time calculated on the basis of the calculated sunset and the value at O2	Integer value range: 0...23
T4	Minutes in the switch-off time calculated on the basis of the calculated sunset and the value at O2	Integer 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> 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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Bit outputs
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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		

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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

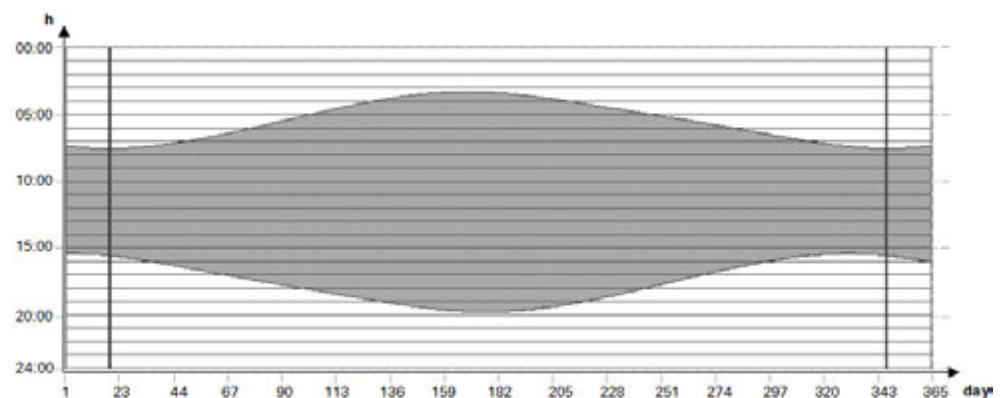


Fig. 163: Sunrise and sunset in Bonn

#### Drevja in Norway

The following is the geodata for the Drevja location in Norway:

- Latitude: 65.9780775
- Longitude: 13.2348074

The sun does not set during the summer months (day 165 to 180).

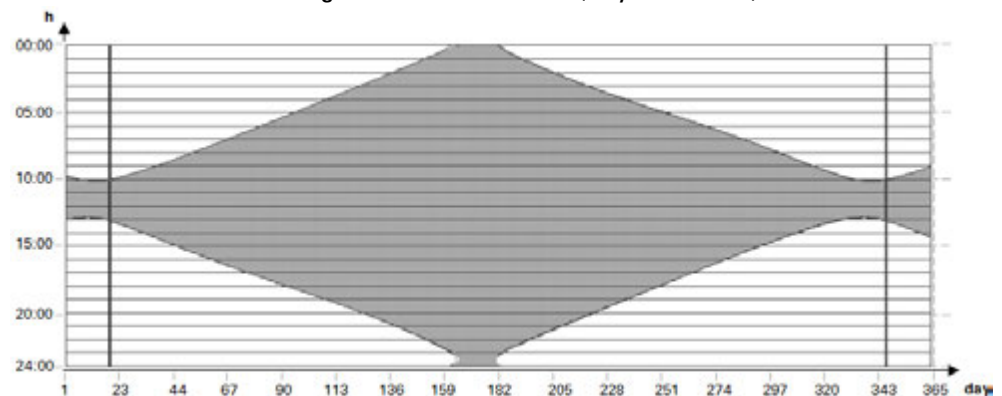


Fig. 164: Sunrise and sunset in Drevja

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Douala in Cameroon

The following is the geodata for the Douala location in Cameroon:

- Latitude: 4.0047314
- Longitude: 9.7329299

The sunrise and sunset times remain the same throughout the whole year with little variation.

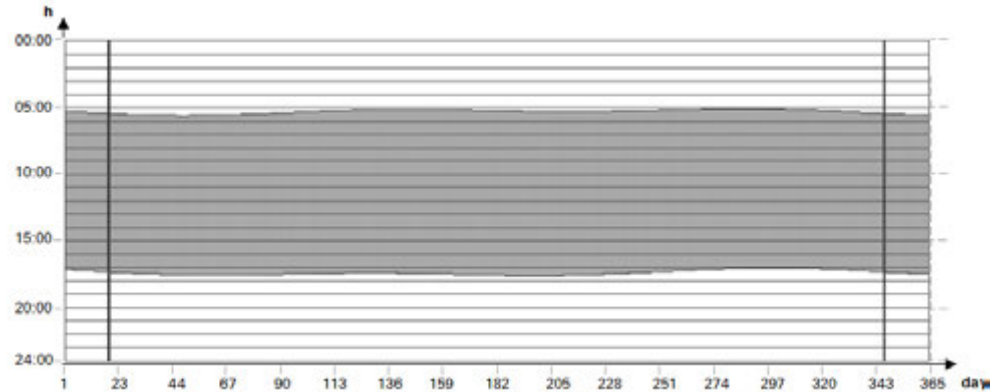


Fig. 165: Offset; O1=-2; O2=2; Q1=1 will switch on 2 hours before sunrise and off 2 hours after sunset

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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

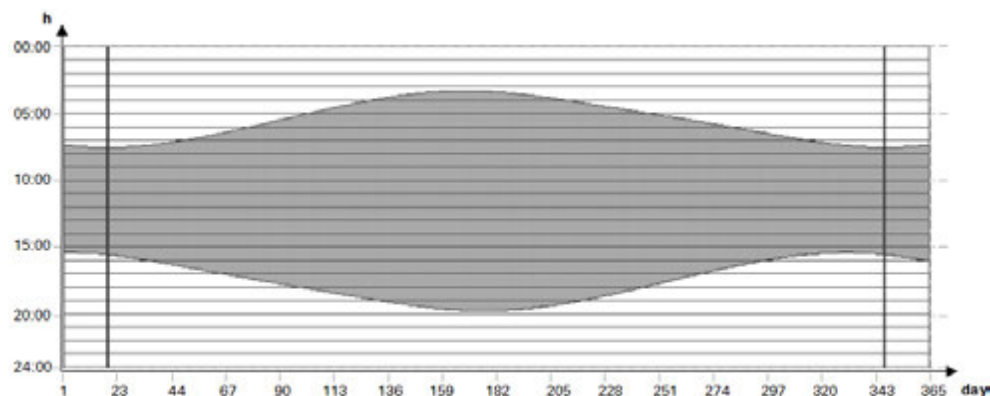


Fig. 166: No offset; O1=0; O2=0; Q1=1 between sunrise and sunset

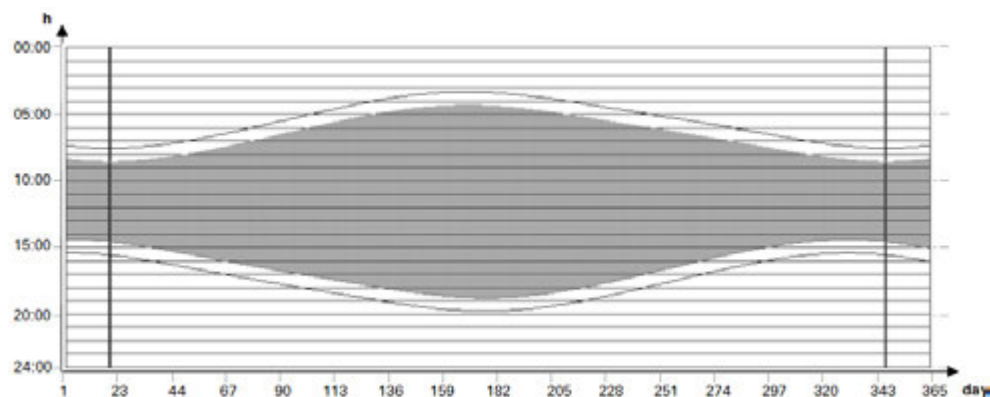


Fig. 167: Offset; O1=1; O2=-1; Q1 will switch on 1 hour after sunrise and off 1 hour before sunset

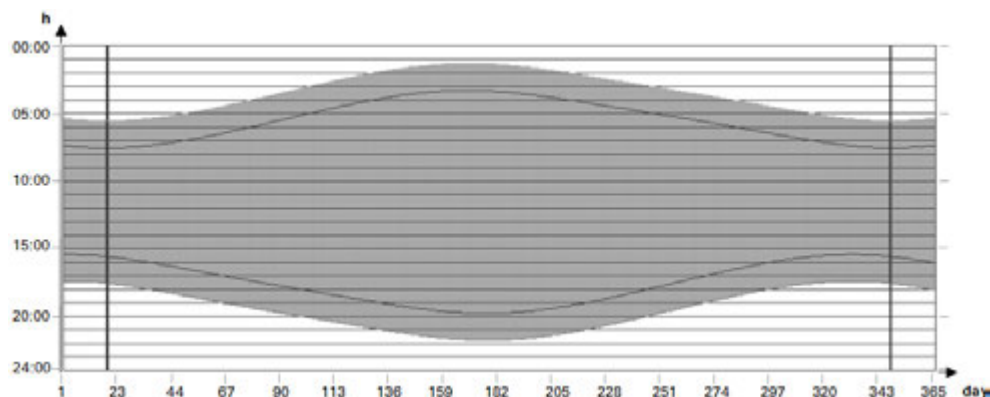


Fig. 168: Offset; O1=-2; O2=2; Q1=1 will switch on 2 hours before sunrise and off 2 hours after sunset

## 6. Function blocks

### 6.1 Manufacturer function blocks

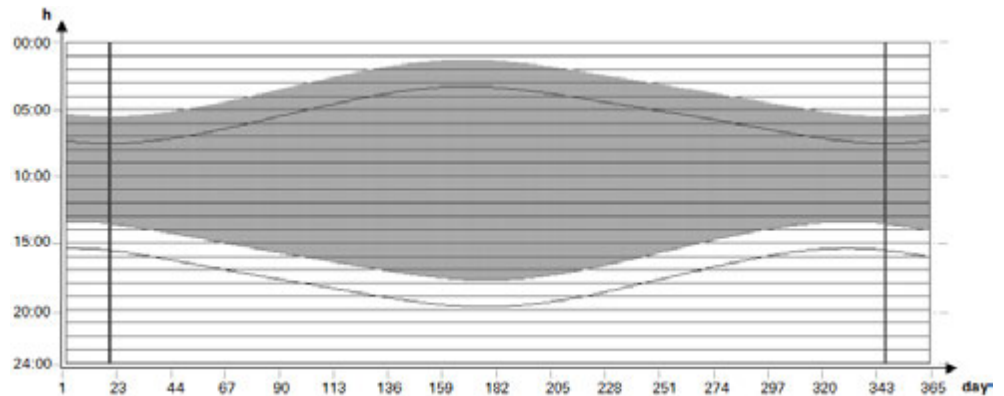


Fig. 169: Offset; O1=-2; O2=-2; Q1=1 will switch on 2 hours before sunrise and off 2 hours before sunset

#### Overlap between switch-on and switch-off times

The following geodata applies to the examples below:

- Latitude: 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.

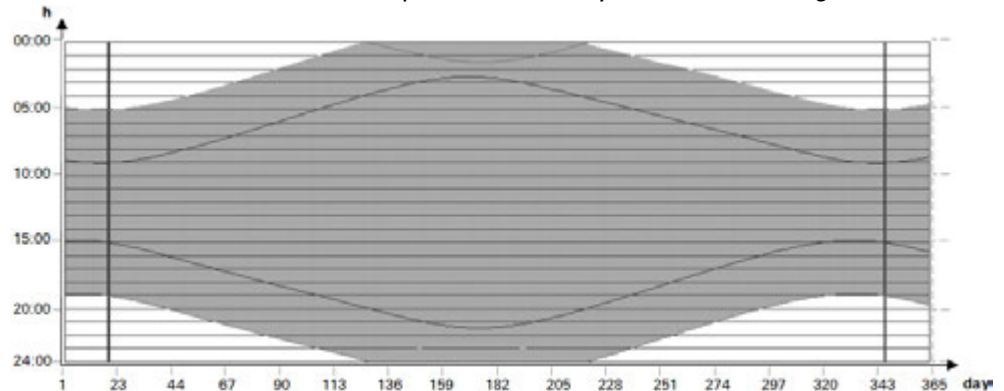


Fig. 170: Q1 does not switch off during the summer months

#### Switch-off time before switch-on time

The following geodata applies to the examples below:

- Latitude: 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 to always remain off during these months.

## 6. Function blocks

### 6.1 Manufacturer function blocks

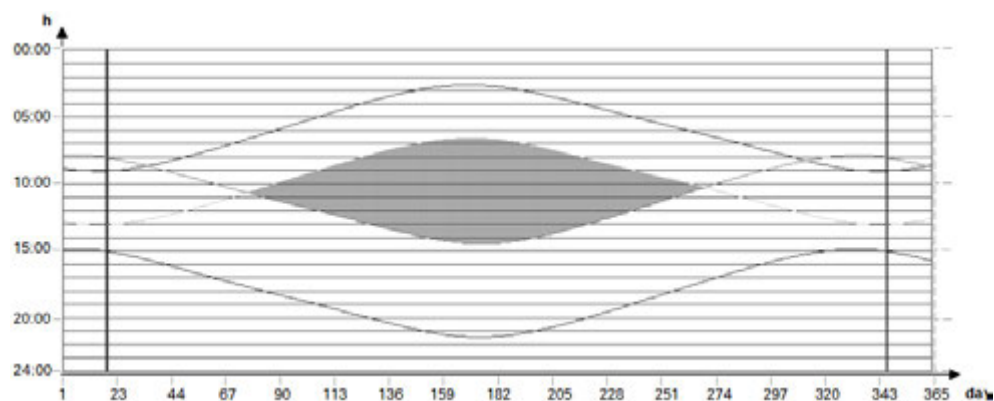


Fig. 171: Q1 does not switch on during the winter months

#### See also

- Section "HW - Weekly timer (Hour Week)", page 244
- Section "HY - Year time switch (Hora Year)", page 254
- Section "OT - Operating hours counter ", page 264
- Section "RC - Real-time clock", page 269
- Section "T - Timing relay", page 272
- Section "WT - Weekly timer (WeekTable)", page 292
- Section "YT - Year time switch (Year Table)", page 284



## 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 317.

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.

Cxx	
EN	OF
C_	FB
D_	CY
SE	ZE
RE	<b>QV</b>
<b>SH</b>	
<b>SL</b>	
<b>SV</b>	

#### 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.

#### **ATTENTION**

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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
C_	Counter input, counts with every rising edge	
D_	Count direction 0: up counting 1: down counting	
SE	The starting value at SV is applied whenever there is a rising edge at this input	
RE	Reset 1: QV=0	Reset the counter to zero
<b>(DWord)</b>		
SH	Upper Threshold Value	Integer value range: -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. 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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

	Description	Note
<b>(bit)</b>		
OF	Overflow 1: if $QV \geq SH$	OF=1, if the actual value QV is greater than or equal to the upper threshold value;
FB	Fall below 1: if $QV \leq SL$	FB=1, if the actual value QV is less than or equal to the lower threshold value;
CY	Carry 1: if $QV > \text{value range}$	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 1: if $QV = 0$	
<b>(DWord)</b>		
QV	Current counter value in RUN mode	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 <sup>2)</sup> NET station n	x
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Bit outputs
bit via NET (send)	
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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 diagrams

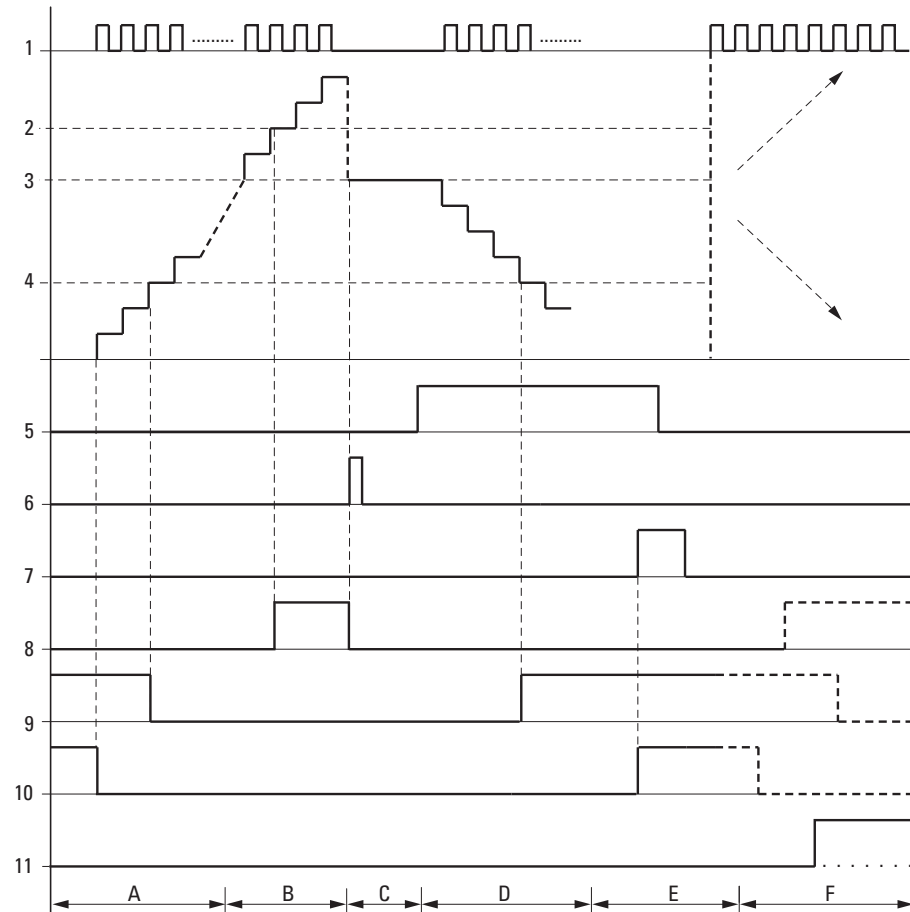


Fig. 172: Signal diagram of counter relay

#### 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

- 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 *easySoft 8Project view/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.

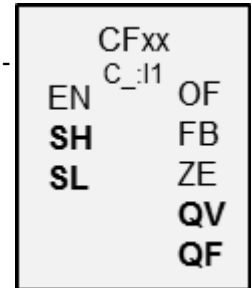
#### See also

- Section "Timing and counter relay example", page 631
- Section "CF - Frequency counter", page 311
- Section "CH - High-speed counter", page 317
- Section "CI - Incremental Counter", page 323

### 6.1.2.2 CF - Frequency counter

#### 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.

$$F = QF \cdot 0.1$$

The value range cannot be exceeded as the maximum measured value is less than the value range.

The CF01...CF04 frequency counters are not dependent on the cycle time.

The minimum counter frequency is 0 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



Square wave count pulses are required to ensure a 1:1 mark to space ratio.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### *ATTENTION*

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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		
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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> NET station n	x
ID: Diagnostic alarm	x
LE - Output backlight	x



## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
P device buttons	x
I - Bit input	x
Q - Bit output	x
Q - Bit output of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

	Description	Note
<b>(bit)</b>		
OF	Overflow 1: if $QV \geq SH$	
FB	Fall Below 1: if $QV \leq SL$	
ZE	Zero 1: if $QV = 0$	
<b>(DWord)</b>		
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 0...50 000. The following formula applies: $10\ 000 = 1\text{ kHz}$ . The measurable frequency range is 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Bit outputs
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> 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 inter-val	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														
<input checked="" type="checkbox"/> 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.															
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>	Measuring interval	Maximum value at QV	0.1s	500	0.5s	2 500	1.0s	5 000	2.0s	10 000	5.0s	25 000	10.0s	50 000	The longer the measuring interval, the smaller the frequency being measured can be.
Measuring interval	Maximum value at QV															
0.1s	500															
0.5s	2 500															
1.0s	5 000															
2.0s	10 000															
5.0s	25 000															
10.0s	50 000															
Simulation possible																

## Other

### Signal diagram

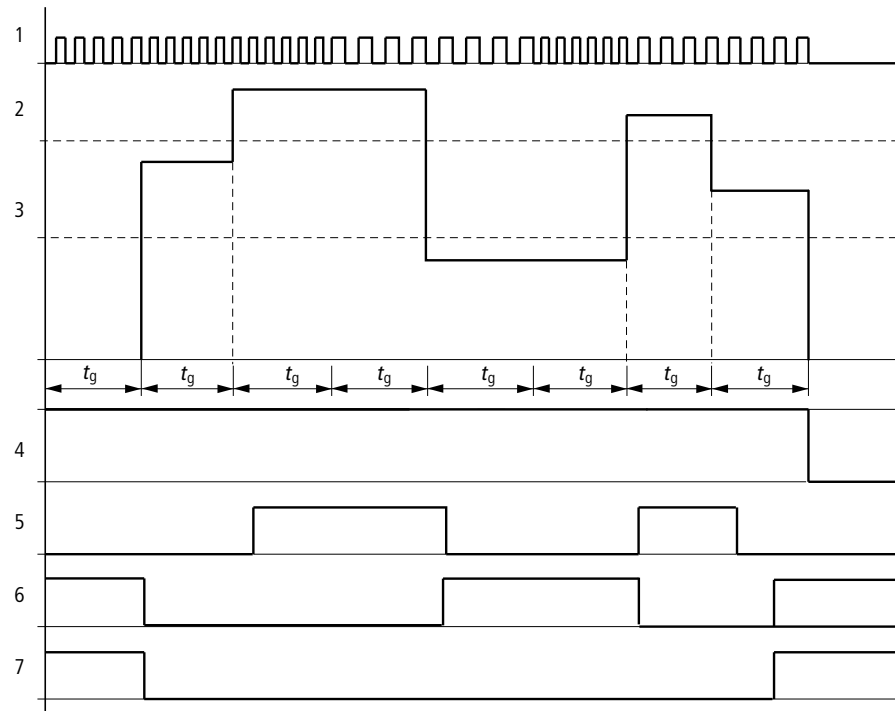


Fig. 173: Signal diagram of frequency counter

- 1: One of the device inputs I01 to I04
- 2: Upper threshold value SH
- 3: Lower threshold value SL
- 4: Enable CF..EN
- 5: Function block output (N/O) OF: Upper threshold value exceeded
- 6: Function block output (N/O) FB: Lower threshold value undershot.
- 7: Function block output (N/O) ZE: If actual value equal to zero
- 8.  $t_g$ : gate time (= measuring interval) for the frequency measurement

The first measurements are made after the function block input EN enable signal has been activated. After the gate time has expired, the value is output at the block outputs QV and converted to QF. The contacts OF, FB and ZE are set in accordance with the measured frequency. If the 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.

## **6. Function blocks**

### **6.1 Manufacturer function blocks**

#### **See also**

- Section "C - Counter Relay", page 305
- Section "CH - High-speed counter", page 317
- Section "CI - Incremental Counter", page 323
- Section "Timing and counter relay example", page 631

### 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.

CHxx C_:1	
EN	OF
D_	FB
SE	CY
RE	ZE
SH	QV
SL	
SV	

#### 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

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### *ATTENTION*

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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
D_	Count direction 0: up counting 1: down counting	
SE	The starting value at SV is applied whenever there is a rising edge at this input	
RE	Reset 1: QV=0	
<b>(DWord)</b>		
SH	Upper Threshold Value	Integer value range: -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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

	Description	Note
<b>(bit)</b>		
OF	Overflow 1: if $QV \geq SH$	OF=1, if the actual value is greater than or equal to the upper threshold value;
FB	Fall below 1: if $QV \leq SL$	FB=1, if the actual value is less than or equal to the lower threshold value;
CY	Carry 1: if $QV > \text{value range}$	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 1: if $QV = 0$	
<b>(DWord)</b>		
QV	Current counter value in RUN mode	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 <sup>2)</sup> NET station n	x
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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

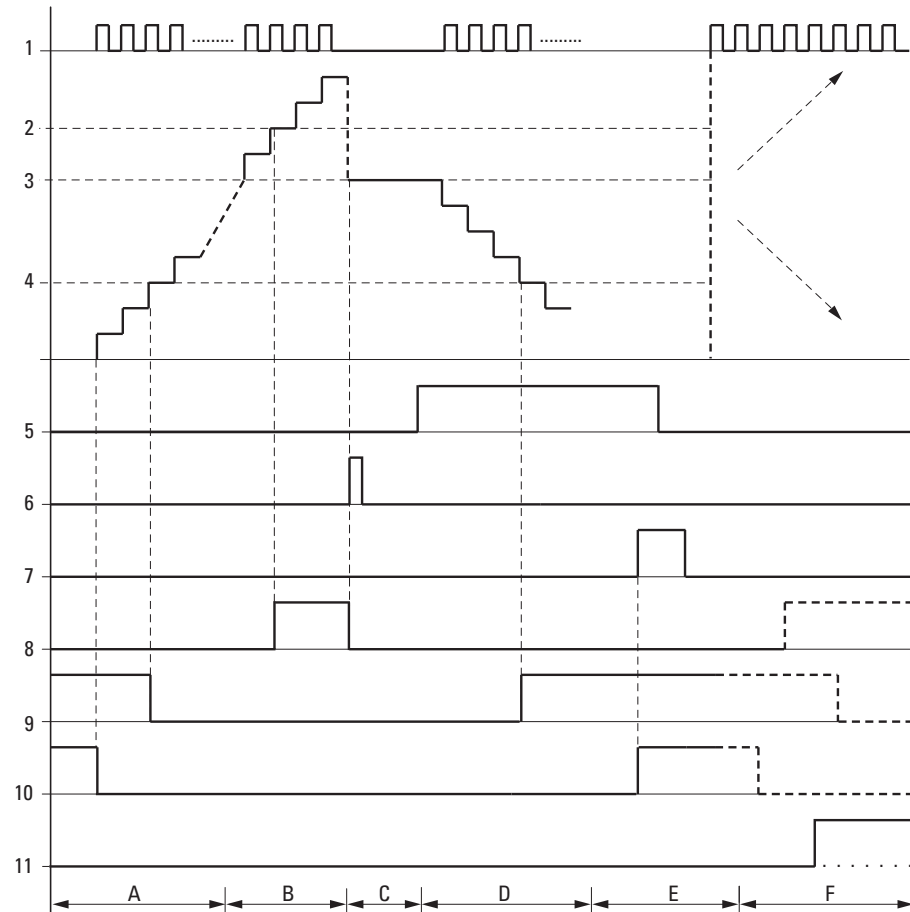


Fig. 174: Signal diagram High-speed counter

#### 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

## 6. Function blocks

### 6.1 Manufacturer function blocks

- 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 *easySoft 8Project view/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 "C - Counter Relay", page 305
- Section "CF - Frequency counter", page 311
- Section "CI - Incremental Counter", page 323
- Section "Timing and counter relay example", page 631

#### 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.

CI0x	
A:Iy B:I(y+1)	
EN	OF
SE	FB
RE	CY
SH	ZE
SL	QV
SV	

#### 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

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.

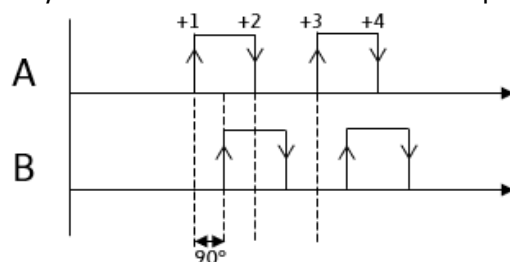


Fig. 175: CI function block counting up;  $QV=QV+4$

#### 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 B followed by a rising edge at channel A. The same applies to the falling edges in sequence at channel B and channel A. The counter relay's result will be decremented and output at output QV.

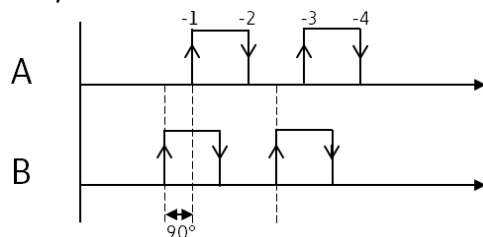


Fig. 176: CI function block counting down;  $QV=QV-4$

#### ATTENTION

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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> 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 1: QV=0	
<b>(DWord)</b>		
SH	Upper Threshold Value	Integer value range: -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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> NET station n	x
ID: Diagnostic alarm	x
LE - Output backlight	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
P device buttons	x
I - Bit input	x
Q - Bit output	x
Q - Bit output of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

	Description	Note
<b>(bit)</b>		
OF	Overflow 1: if $QV \geq SH$	OF=1, if the actual value is greater than or equal to the upper threshold value;
FB	Fall below 1: if $QV \leq SL$	FB=1, if the actual value is less than or equal to the lower threshold value;
CY	Carry 1: if $QV > \text{value range}$	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 1: if $QV = 0$	
<b>(DWord)</b>		
QV	Current counter value in RUN mode	The pulses at channel A and channel B are counted. 2 pulses are counted per counting period. Example: 2 pulses at channel A and 2 pulses at channel B; value at CI..QV = 4

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

Parameter set	Description	Note
<input checked="" type="checkbox"/> 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		

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Other

#### Signal diagram

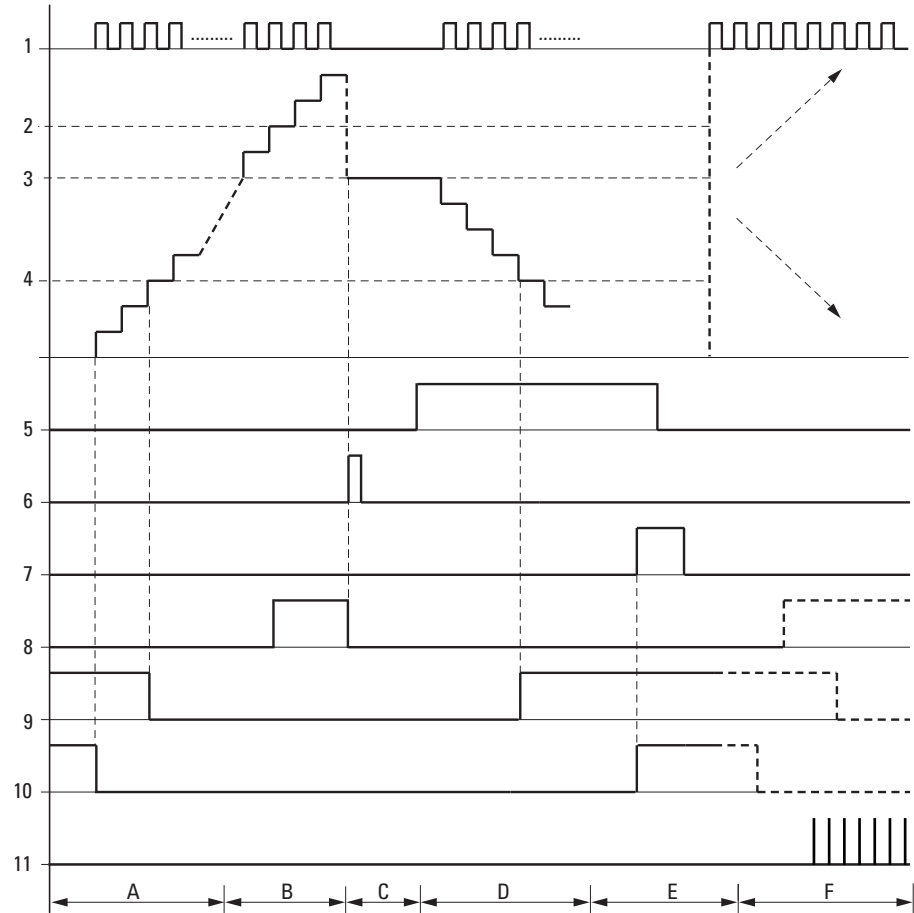


Fig. 177: Signal diagram High-speed incremental value counter

#### 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 range is exceeded.



- 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 *easySoft 8Project view/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 "C - Counter Relay", page 305
- Section "CF - Frequency counter", page 311
- Section "CH - High-speed counter", page 317
- Section "Timing and counter relay example", page 631

6. Function blocks

6.1 Manufacturer function blocks

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.

Axx	
EN	Q1
I1	CY
I2	
F1	
F2	
OS	
HY	

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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		
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) Default value = 1	
F2	Gain factor for I2 (I2 = F2 * value) Default value = 1	
OS	Offset for the value at I1, $I1_{OS} = OS + \text{actual value at I1};$	
HY	Switching hysteresis for value at I2. 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. $I2_{HY} = \text{Actual value at I2} + HY,$ $I2_{HY} = \text{Actual value at I2} - HY);$	

### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Operating modes

	Description	Note
LT: less than (I1 < I2)	Less than (I1 < I2)	
LE: less than/equal to (I1 <= I2)	Less than or equal to (I1 <= I2)	
EQ: equal to (I1 = I2)	Equal to (I1 = I2)	
GE: greater than/ equal to (I1 >= I2)	Greater than or 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 (e.g. I1 < I2 with LT mode active)	
CY	$-2^{31} \leq I1 * F1 + OS \leq (2^{31} - 1) \Rightarrow CY = 0$ $-2^{31} \leq I2 * F2 + HY \leq (2^{31} - 1) \Rightarrow CY = 0$ $-2^{31} \leq I2 * F2 - HY \leq (2^{31} - 1) \Rightarrow CY = 0$ <p>Status 1 if the above permissible value range of the function block is exceeded.</p>	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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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		

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Other

**Retention** - The function block does not recognize retentive data.

#### Signal diagrams

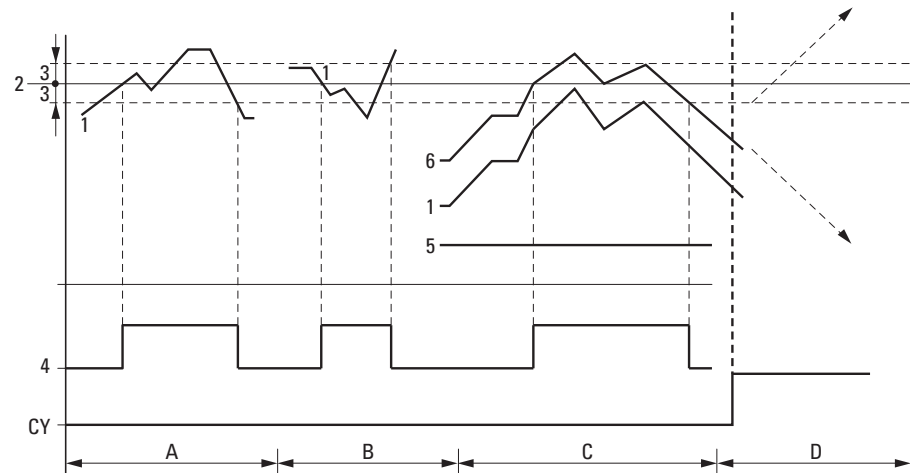


Fig. 178: Analog comparator signal diagram

#### 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.

#### Example of a analog value comparator function block when using the EDP programming language

```
I01----A01Q1-----A 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
```

Fig. 179: 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) 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 336
- Section "AV - Average", page 342
- Section "CP – Comparator", page 350
- Section "LS - Value scaling", page 354
- Section "MM - Min-/Max function", page 359
- Section "PW - Pulse width modulation", page 368
- Section "PM - Performance map ", page 362

6. Function blocks

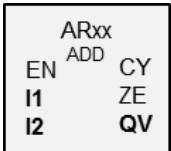
6.1 Manufacturer function blocks

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 alarm function blocks, AR01 through AR32. These function blocks can be used to carry out the four basic arithmetic operations: addition, subtraction, multiplication, and division. Firmware version 2.30 and higher additionally supports exponentiation and the modulo function (remainder).



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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		
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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Operating modes

	Description	Note
ADD – Adder	Adding (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 – Subtractor	Subtracting (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	Multiply (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	Divide (I1 : I2 = QV)	1024: 0 = QV contains the last permissible value as an overflow occurred. The carry bit AR..CY is set to 1. 10 : 100 = 0
MOD - Modulo	Modulo (remainder) (I1 mod I2 = QV)	I1 mod 0 = 0 The carry bit gets state "0".
POW - Exponentiation block	Exponent (I1 ^ I2) = QV)	100^10000 = QV contains the last permissible value, since an overflow occurred. The carry bit gets state "0".

#### Function block outputs

	Description	Note
<b>(bit)</b>		
CY	Status 1 if the above value range is exceeded.	Output signals overflow or division by 0
ZE	Status 1 if the value of the function block output QV (the calculation result) equals zero	Output signals when the result is 0
<b>(DWord)</b>		
QV	Current counter value in RUN mode	Integer value range: -2,147,483,648 to +2,147,483,647

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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		

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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

$$12 \times 12 = 144$$

1000042 x 2401 = Last valid value of this arithmetic operation, due to overflow (Carry) correct value = 2401100842 AR..CY = 1

$$-1000 \times 10 = -10000$$

#### 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-----Ä Q 01
I 02---AR02ZE-----S Q 02
```

Fig. 180: 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>
```

Fig. 181: 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. Integer value range: -2,147,483,648 to +2,147,483,647
>I2	Second value; Integer value range: -2,147,483,648 to +2,147,483,647
>QV	Supplies the calculation result. Integer value range: -2,147,483,648 to +2,147,483,647

#### Examples for Modulo

$$6 \bmod 2 = 0$$

$$9 \bmod 4 = 1$$

$$13 \bmod 0 = 0$$

#### POW examples

$$6 ^ 2 = 36$$

$$10 ^ 0 = 1$$

$$-9 ^ 3 = -729$$

#### See also

- Section "A - Analog value comparator", page 330
- Section "AV - Average", page 342
- Section "CP – Comparator", page 350
- Section "LS - Value scaling", page 354
- Section "MM - Min-/Max function", page 359
- Section "PM - Performance map ", page 362
- Section "PW - Pulse width modulation", page 368

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 8, 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 large deviations that occur rarely. 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.

AVxx ONE	
EN	RY
T_ RE	E1 QV
I1	QN
NO	

##### 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.

##### 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.

##### Operating mode 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 tem-

## 6. Function blocks

### 6.1 Manufacturer function blocks

perature 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
T_	Trigger input 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.	
<b>(DWord)</b>		
I1	Input value	Integer value range: -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: 0...+2 147 483 647

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> NET station n	x



## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	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

<sup>2)</sup> Only on projects with  $\geq 2$  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 mode	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
<b>(bit)</b>		
RY	1: The average value calculation has been completed, since the specified number of input values to be included has been reached.	
E1	Error 1: If the value range for I1 or NO is exceeded.	
<b>(DWord)</b>		
QV	The current average value	Integer value range: -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: 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block outputs that are bit outputs:

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Bit outputs
M – Markers	x
SN – Output bit via NET (send) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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. 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 + 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.

### 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:

$$\text{One-time mode average value } CMA(n) = \text{ROUND} [ CMA_{n-1} + (I1_n - CMA_{n-1}) / (n+1) ]$$

CMA(n) = Currently calculated simple moving average

n = 1...NO

I1<sub>n</sub> = Value at function block input I1; e.g., temperature value

### Continuous operation

In continuous mode, the smoothing factor is calculated first.

$$\text{Smoothing factor } SF = 2 / (NO+1)$$

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:

$$\text{Continuous operation average value } EMA(n) = \text{ROUND} [ EMA_{n-1} + SF * (I1_n - EMA_{n-1}) ]$$

EMA(n) = Currently calculated exponential moving average

n = 1...NO

SF = Smoothing factor; value between 0 and 1

I1<sub>n</sub> = Value at function block input I1; e.g., temperature value

## 6. Function blocks

### 6.1 Manufacturer function blocks

**Temperature averaging example**

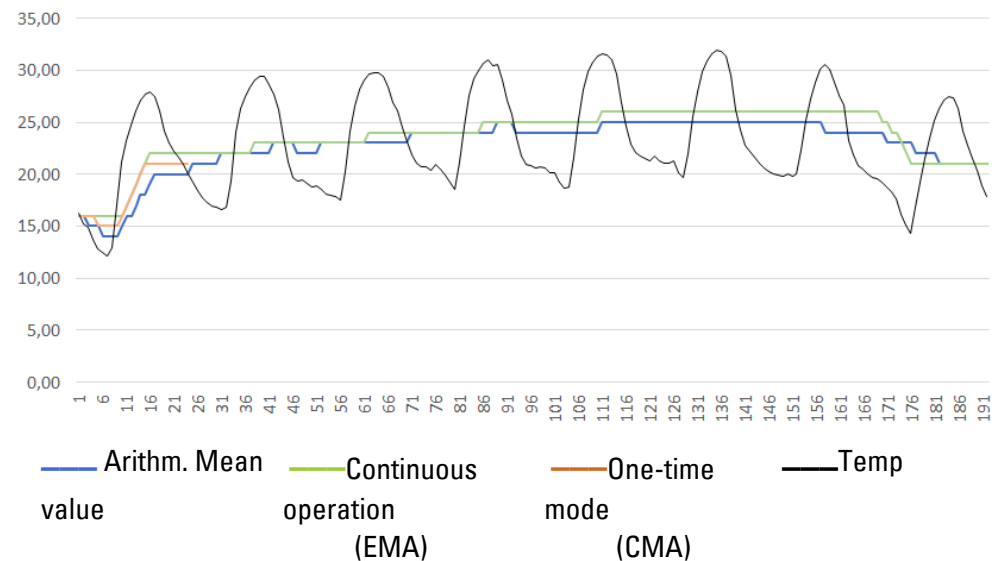


Fig. 182: Sample curve for hourly temperature measurement, over 7 days

#### One-time mode

In the example, the average value in one-time mode is calculated for the 24th value / CMA (23) as follows:

$$\text{CMA}(23) = \text{ROUND} \left[ \text{CMA}(22) + \frac{\text{I1}(23) - \text{CMA}(22)}{23 + 1} \right]$$

$$\text{CMA}(23) = \text{ROUND} [ 1889 + (2004 - 1889)/24 ] = \text{ROUND} [1893,792] = 1894$$

#### Continuous operation

The smoothing factor in the example is  $\text{SF} = 2/(24+1) = 0.08$ .

In the example, the average value in continuous mode is calculated for the 24th value:

$$\text{EMA}(23) = \text{ROUND} [ \text{EMA}(22) + 0,08 * (\text{I1}(23) - \text{EMA}(22)) ]$$

$$\text{EMA}(23) = \text{ROUND} [ 2035 + 0,08 * (2004 - 2035) ]$$

$$\text{EMA}(23) = \text{ROUND} [2032,52] = 2033$$

## 6. Function blocks

### 6.1 Manufacturer function blocks

Tab. 83: 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	–
...		...	...	...	...	–

#### See also

- Section "A - Analog value comparator", page 330
- Section "AR - Arithmetic", page 336
- Section "CP – Comparator", page 350
- Section "LS - Value scaling", page 354
- Section "MM - Min-/Max function", page 359
- Section "PM - Performance map ", page 362
- Section "PW - Pulse width modulation", page 368

## 6. Function blocks

### 6.1 Manufacturer function blocks

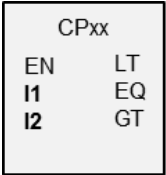
#### 6.1.3.4 CP – Comparator

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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		
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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

	Description	Note
<b>(bit)</b>		
LT	Less Than 1: if I1 < I2	
EQ	Equal 1: if I1 = I2	
GT	Greater Than 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 <sup>2)</sup> NET station n	x
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Bit outputs
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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-----Ä M 21
CP12LT-----Ä M 22
CP12GT-----u R M 21
               h R M 22
```

Fig. 183: 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  
Fig. 184: 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 Integer value range: -2,147,483,648 to +2,147,483,647
>I2	Comparison value; I2 is compared with I1 Integer value range: -2,147,483,648 to +2,147,483,647

#### See also

- Section "A - Analog value comparator", page 330
- Section "AR - Arithmetic", page 336
- Section "AV - Average", page 342
- Section "LS - Value scaling", page 354
- Section "MM - Min-/Max function", page 359
- Section "PM - Performance map ", page 362
- Section "PW - Pulse width modulation", page 368

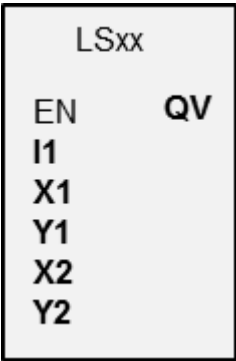
6. Function blocks  
6.1 Manufacturer function blocks

6.1.3.5 LS - Value scaling

General

easyE4 base devices provide 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 for the conversion of values for 0...20 mA in 4...20 mA.



Operating principle

"EN = 1" starts the function block.  
"EN = 0" initiates a reset in which the output **QV** is reset to 0.

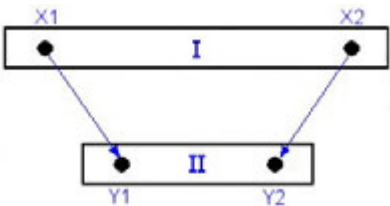


Fig. 185: Figure: Scaling the input values - reducing

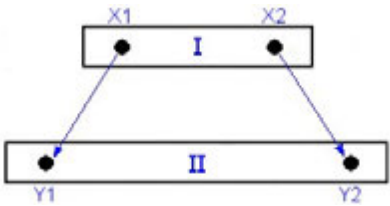


Fig. 186: Scaling the input values - increasing

- ① Source range
- ② Destination range

## 6. Function blocks

### 6.1 Manufacturer function blocks

The mathematical relationship is follows:

$$Y = m * X + Y_0$$

$$m = \frac{Y_2 - Y_1}{X_2 - X_1} \quad Y_0 = \frac{X_2 * Y_1 - X_1 * Y_2}{X_2 - X_1}$$

m = Gradient

Y<sub>0</sub> = Y offset when X = 0

X<sub>1</sub>, Y<sub>1</sub> = First value pair

X<sub>2</sub>, Y<sub>2</sub> = Second value pair

g = Straight line with positive gradient

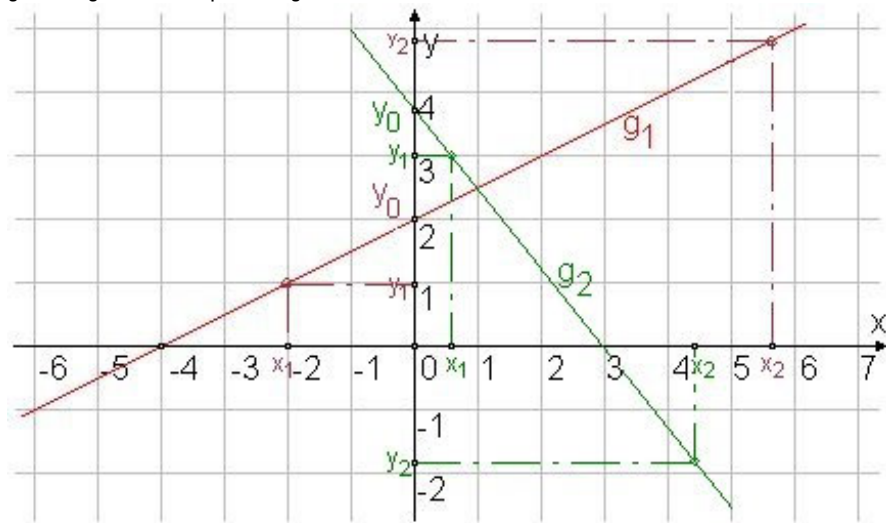


Fig. 187: Mathematical interrelationship

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		<b>(DWord)</b>
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	Second scale; data point 1	
X2	First scale; data point 2	
Y2	Second scale; data point 2	

##### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND - NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
(DWord)		
QV	Delivers the scaled input value	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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> 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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x


<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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	

## 6. Function blocks

### 6.1 Manufacturer function blocks

	Description	Note
	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

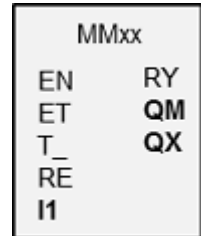
- Section "AR - Arithmetic", page 336
- Section "AV - Average", page 342
- Section "CP – Comparator", page 350
- Section "MM - Min-/Max function", page 359
- Section "PW - Pulse width modulation", page 368
- Section "PM - Performance map ", page 362

### 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
<b>(bit)</b>		
EN	1: Activates the function block.	
ET	Enable trigger 0: Calculates Min/Max every time the function block is called; trigger input T_ is deactivated 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 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	
<b>(DWord)</b>		
I1	Analog value used for the Min/Max comparison	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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Function block outputs

	Description	Note
<b>(bit)</b>		
RY	Event message signaling that a new minimum or maximum value has been entered	This message will only be displayed for one cycle
<b>(DWord)</b>		
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 <sup>2)</sup>	x



## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Value outputs
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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.

#### See also

- Section "AR - Arithmetic", page 336
- Section "AV - Average", page 342
- Section "CP – Comparator", page 350
- Section "LS - Value scaling", page 354
- Section "PW - Pulse width modulation", page 368
- Section "PM - Performance map", page 362

## 6. Function blocks

### 6.1 Manufacturer function blocks

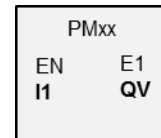
#### 6.1.3.7 PM - Performance map

Only available on easySoft Version 7.10 or higher.

If this function block is not being shown in the leftmost pane in easySoft 8, 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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		
I1	Input value	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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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. 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
<b>(bit)</b>		
E1	Error 1: If QV exceeds the value range	
<b>(DWord)</b>		
QV	Output value determined from the reference value table based on input value I1.	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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block outputs that are bit outputs:

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Bit outputs
M – Markers	x
SN – Output bit via NET (send) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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. 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 + Call enabled	Constants can be edited on the device, as can function block parameters when using the EDP programming language.	
Simulation possible		

6. Function blocks

6.1 Manufacturer function blocks

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.

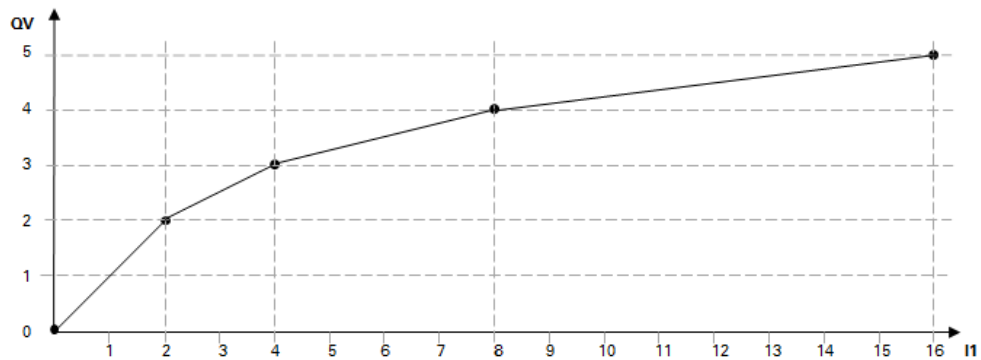


Fig. 188: Example of a characteristic curve for the PM function block

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 Next higher value: 2 Next lower value: 0 Nearest value: 2
3	Interpolating: 3 Next higher value: 3 Next lower value: 2 Nearest value: 3
5	Interpolating: 4 Next higher value: 4 Next lower value: 3

## 6. Function blocks

### 6.1 Manufacturer function blocks

Value at I1	Value at QV as a function of the operating mode
	Nearest value: 3
8	Interpolating: 4 Next higher value: 4 Next lower value: 4 Nearest value: 4
27	Interpolating: 11 Next higher value: 12 Next lower value: 10 Nearest value: 10

#### See also

- Section "A - Analog value comparator", page 330
- Section "AR - Arithmetic", page 336
- Section "AV - Average", page 342
- Section "CP – Comparator", page 350
- Section "LS - Value scaling", page 354
- Section "MM - Min-/Max function", page 359
- Section "PW - Pulse width modulation", page 368

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 6.1.3.8 PW - Pulse width modulation

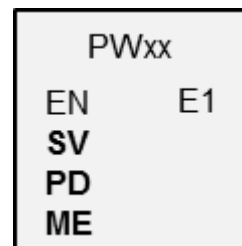
##### General

easyE4 base devices provide two function blocks for pulse width modulation PW01...PW02.

The potential application area for PW function blocks consists of any application in which an actuator system cannot be driven with an analog system, but only digitally with ON and OFF states instead.

PW function modules are used primarily to drive easyE4 devices with transistor outputs.

However, PW function blocks can also be used with easyE4 devices featuring relay outputs. Due to the relay's ON and OFF times, a longer period duration and a longer minimum on duration should be selected than for easyE4 devices with transistor outputs.



##### Operating principle

The period duration of the signal stays constant. You define the period duration at the PD input. The PW function block generates a rectangular function signal with an on and off duration. This on duration is proportional to the manipulated variable at input SV.

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 shouldn't 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
<b>(bit)</b>		
EN	1: Activates the function block. 0: Output Q1 or Q2 switches to a state of 0.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		
SV	Manipulated variable Value range: 0 to 4095 (12 bit) of this value range corresponds to 0...100% of the period duration.	Value range: 0 ... 4095  No signals are output at Q1 and Q2 if SV=0 or pulse width < ME, and the output concerned will stay at 0.
PD	Period duration [ms] When the value is 0, no pulses are output at Q1 and Q2. The minimum period duration for a easyE4 device with transistor outputs is 5 ms. (the resulting max. frequency is 200 Hz).	Value range: 0...65535
ME	Minimum On duration [ms] = Minimum off duration In the case of electronic load relays, a minimum on duration of 0 can be configured.  The shortest on and off durations or pulse width for devices with transistor outputs are 0.1 ms. These times are determined by the electronics to a very large extent.  For easyE4 device relay outputs and for contactor control systems, a minimum on duration of 300 ms is recommended.	Value range: 0...65535  Computationally, the value range is limited: 0 to 32767; otherwise, there will be a fault scenario, since pulse width < ME or off duration < ME.

#### 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 scaling since the DC..QV covers the same value range between 0 and 4095.

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, (OFF). The status of the PW..E1 contact must be observed.

## 6. Function blocks

### 6.1 Manufacturer function blocks

If the off duration of the pulse at the output is shorter than the minimum off duration, Q1 or Q2 will remain in a state of "1" (ON). The status of the PW..E1 contact must be observed.

#### Parameter limits for period duration and minimum ON duration

Tab. 84: Parameter limit values for period duration and minimum on duration

	Period duration [ms]	Minimum On duration [ms]	Note
<b>Basic device</b>			
EASY-E4-UC-... EASY-E4-DC-... EASY-E4-AC-...	Min. 5 Max. 65535	min. 0.1 <sup>1)</sup> Max. 65535	<b>Period duration</b> When the value is 0, no pulses are output at Q1 and Q2. <b>minimum contact closing time</b> Can be selected within the valid limits

1) For devices with transistor outputs

#### Minimum period duration PD

The minimum period duration is 5 ms.

#### Minimum on duration ME = Minimum off duration

If the calculated on duration pulse width is shorter than the minimum on duration ME, a pulse will not be output at Q1/Q2.

If the calculated off duration is shorter than the minimum off duration ME, output Q1/Q2 will remain on.

The following applies when driving contactors: Select the shortest possible minimum contact closing time ME, but make sure it is still longer than the contactor switching time (300 ms, for example). Select the longest possible period duration in order to reduce contactor wear. One possible use is a heater controller.

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 (PD/ME) determines which percentage of manipulated variables have no effect.

Because of this, the minimum on duration must be as short as possible and the period duration as long as possible so that PD/ME will be as large as possible.

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

	Description	Note
(bit)		
E1	Error output 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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.

### Sample project configuration

PD=40000 ms; ME=300 ms

Tab. 85: Effect of various SV values on pulse width for a specific period duration

Value SV	Period duration PD [ms]	Duty factor Pulse width PW [ms]	Off duration [ms]
0	40000	0	0
5	40000	0 (ME)	40000 (ME)
35	40000	342	39648
1000	40000	9768	30232
1400	40000	13675	27325
2048	40000	20005	19995
3218	40000	31433	8567
3768	40000	36805	3195
4093	40000	40000 (ME)	0 (ME)

1) For devices with transistor outputs

$$PW = [SV/4095] \cdot PD$$

PW = Pulse width (duty factor)

SV = Manipulated variable value

PD = Period duration

## 6. Function blocks

### 6.1 Manufacturer function blocks

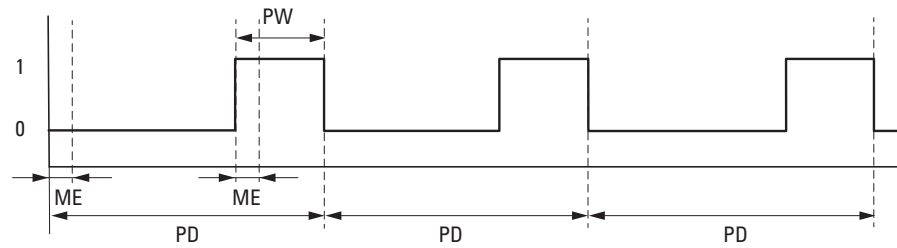


Fig. 189: PW pulse at the function block output when SV = 1400, ME = 93 ms, PD=1000 ms

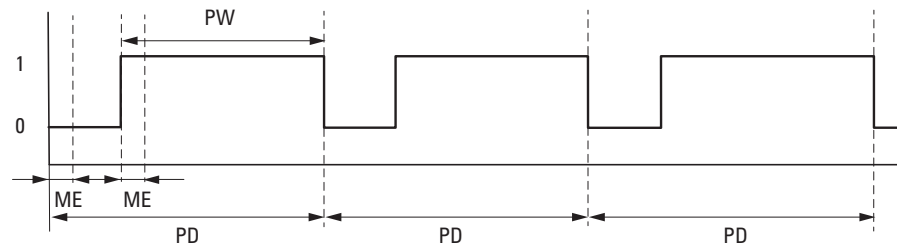


Fig. 190: PW pulse at the function block output when SV = 3218, ME = 93 ms, PD=1000 ms

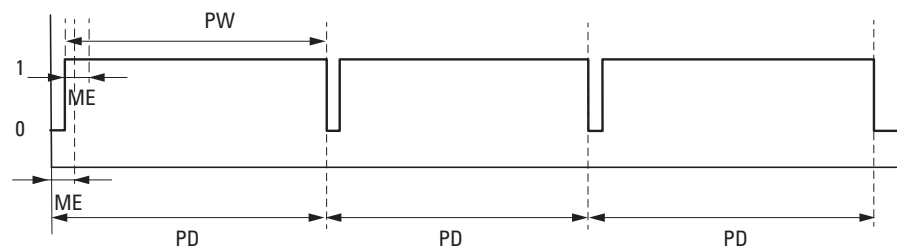


Fig. 191: A constant signal is signaled at the function block output when SV = 3768, ME = 93 ms, PD=1000 ms; E1 = 1

PD: Pulse duration

PW: Pulse width

ME: Minimum on duration, Minimum off duration

#### See also

- Section "AR - Arithmetic", page 336
- Section "AV - Average", page 342
- Section "CP – Comparator", page 350
- Section "LS - Value scaling", page 354
- Section "MM - Min-/Max function", page 359
- Section "PM - Performance map ", page 362

## 6.1.4 Open-loop and closed-loop function blocks

### 6.1.4.1 DC - PID controller

#### General

easyE4 base devices provide 32 PID controller function blocks DC01...DC32.

DCxx	
EN	UNP
EP	LI
EI	QV
ED	QP
SE	QI
I1	QD
I2	
KP	
TN	
TV	
TC	
MV	

#### 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

## 6. Function blocks

### 6.1 Manufacturer function blocks

the actual value ( $X_i$ ) at a specified scan time. The equation used by the device for the proportional component is as follows:

$$Y_P(t) = K_p * [X_s(t) - X_i(t)]$$

$K_p$  = Proportional gain

$X_s(t)$  = SETPOINT value at scan time  $t$

$X_i(t)$  = ACTUAL value at scan time  $t$

#### Integral component

The integral component  $Y_I$  is proportional to the sum of the control difference over time. The equation used by the device for the integral component is as follows:

$$Y_I(t) = K_p * T_c / T_n * [X_s(t) - X_i(t)] + Y_I(t-1)$$

$K_p$  = Proportional gain

$T_c$  = Scan time

$T_n$  = Reset time (also known as integration time).

$X_s(t)$  = Setpoint with scan time  $t$

$X_i(t)$  = Actual value with scan time  $t$

$Y_I(t-1)$  = Value of the integral component of the manipulated variable with scan time  $t - 1$

#### Differential component

The differential component  $Y_D$  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:

$$Y_D(t) = K_p * T_v / T_c * (X_i(t-1) - X_i(t))$$

$K_p$  = Proportional gain

$T_c$  = Scan time

$T_v$  = Rate time of the control system (also called the differential time)

$X_i(t)$  = Actual value with scan time  $t$

$X_i(t-1)$  = 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 function block input  $EN$  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.



## 6. Function blocks

### 6.1 Manufacturer function blocks

Example: If only function block input EP and 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 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 function block input is activated or the value at the MV input is changed. If 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 MV will be output to QV as the value zero.

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	
EP	1: Activates the proportional term	
EI	1: Activates the integral term	
ED	1: Activates the derivative term	
SE	1: Accept manual manipulated variable	
<b>(DWord)</b>		
I1	Setpoint	Value range: -32768 ... +32767
I2	Actual value	Value range: -32768 ... +32767
KP	Proportional gain Kp [%]	Value range: 0 ... 65535 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. If the value 0 is specified, the scan time is determined by the program cycle time.	

## 6. Function blocks

### 6.1 Manufacturer function blocks

	Description	Note
MV	Manual manipulated variable	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 569.



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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Operating mode

	Description	Note
<b>Operating mode</b>		
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
<b>(bit)</b>		
LI	1: if the value range of the medium-voltage is exceeded.	
<b>(DWord)</b>		
QV	Manipulated variable	Integer value range for operating mode UNP: 0...+4095 (12 Bit) for operating mode BIP: -4096...+4095 (13 Bit)
QP	Proportional component of the manipulated variable Can be used for diagnostic purposes	
QI	Integral component of the manipulated variable Can be used for diagnostic purposes	
QD	Differential component of the manipulated variable 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 <sup>2)</sup>	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Value outputs
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> 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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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 PID controller with the programming method EDP

```

M 51-----u-Ä DC02EN
           d-Ä DC02EP
           d-Ä DC02EI
           v-Ä DC02ED
M 52-----Ä DC02SE

```

Fig. 192: Wiring the function block coils

The function block coils are activated by markers.

```

DC02LI-----Ä S M 96

```

Fig. 193: 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>
```

Fig. 194: 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, in %; Example: The value 1500 is processed in the function block as 15.
>TN	Reset Time Tn: 0... 65535, in 100 ms; Example: The value 250 is processed in the function block as 25 s.
>TV	Rate time TV: 0...65535, 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: • unipolar: 0...4095 • bipolar: -4096...+4095

#### See also

- Section "FT - PT1-Signal smoothing filter ", page 382
- Section "TC - Three step controller", page 402
- Section "VC - Value limitation ", page 407
- Section "BC - Block comparison", page 411
- Section "BV - Boolean operation", page 477
- Section "PO - Pulse output", page 387

## 6. Function blocks

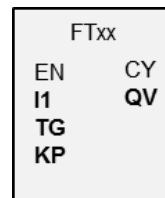
### 6.1 Manufacturer function blocks

#### 6.1.4.2 FT - PT1-Signal smoothing filter

##### General

easyE4 base 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 the output QV is reset to 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  $t = T_g$  the normalized output value is 0.63 QV/QVmax.

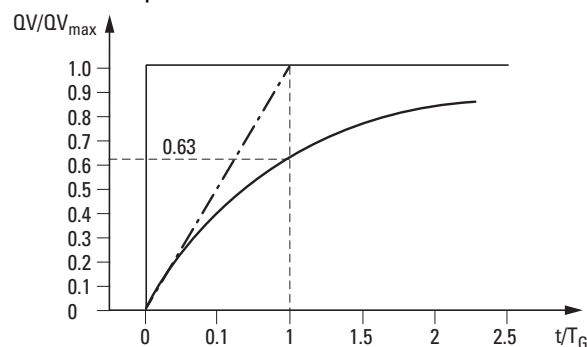


Fig. 195: Response of the FT function block

—— Output value of the FT-PT1 signal smoothing filter function block, Tangent

— · — · Tangent

The output value is based on the following equation:

$$Y(t) = [T_A/T_G] \cdot [K_P \cdot (X(t) - Y(t-1))] + Y(t-1)$$

$Y(t)$  = 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 $T_A$
$T_G \leq 1000 \text{ ms}$	$T_A = 10 \text{ ms}$
$T_G > 1000 \text{ ms}$	$T_A = T_G/100$

### Cycle time to scan time

The ratio of cycle time  $t_{cyc}$  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):  $T_A = 10 \cdot t_{cyc}$ . The sample time is set indirectly with the value for recovery time  $T_G$  (please refer to the table above).

The following therefore applies:  $t_{cyc} \ll T_A$ .

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  $n = 1, 2, 3, \dots$

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

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	
<b>(DWord)</b>		
I1	Input value	Value range: -32768...+32767
TG	Recovery time TG [0.1 s]	Value range: 0...65535 The value 10 corresponds to a recovery time of 1000 ms.
KP	Proportional gain Kp [%] Value range: 0 ... 65535	Value range: 0...65535 The value 100 corresponds to a KP (factor) of 1. 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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND - NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET



#### Function block outputs

	Description	Note
<b>(bit)</b>		
CY	Carry 1: If output value QV falls outside the valid value range.	Value range: -32768...+32767
<b>(DWord)</b>		
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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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		

6. Function blocks

6.1 Manufacturer function blocks

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-----Ä FT01EN

Fig. 196: 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>

Fig. 197: 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 <sub>G</sub>	Recovery time: 0... 65535 resolved in 100 ms Example: The value 250 is processed in the function block as 25 s.
>K <sub>p</sub>	Proportional gain: 0... 65535 in %; Example: With KP=1500 the function block performs the calculation with K <sub>p</sub> =15
QV>	Output value: -32768 ... +32767, smoothed

See also

- Section "BC - Block comparison", page 411
- Section "BV - Boolean operation", page 477
- Section "PO - Pulse output", page 387
- Section "TC - Three step controller", page 402
- Section "VC - Value limitation ", page 407

### 6.1.4.3 PO - Pulse output

#### General

The DC versions of easyE4 base devices provide 2 pulse output function blocks, P001 through P002. These function blocks make it possible to quickly output 24 V pulses in order to drive stepper motors. Pulse output P001 is hardwired to device output Q1 and P002 to device output Q2.



Only easyE4 transistor versions support PO pulse output function blocks.



If you use a PO function block with its hardwired device output Q1 or Q2, do 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.

POxx	
EN	AC
S_	E1
BR	QV
TP	QF
I1	
FS	
FO	
RF	
BF	
P1	
PF	



#### 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 EN function block input is actuated. After you have parameterized the function block, you can actuate the function block input S\_. This will start normal operation. Alternatively, you can also actuate the function block input 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) I1 for normal mode and P1 for jog mode at the permanently assigned high-speed device output Q1 or Q2.

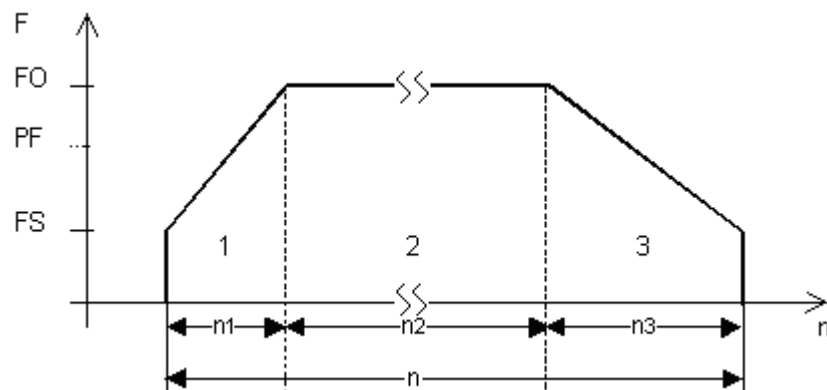


Fig. 198: Typical pulse profile for a stepper motor in normal mode

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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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.

$$n_{RRF} = \frac{(FO - FS)}{RF} * 1000$$

$$n_{RBF} = \frac{(FO - FS)}{BF} * 1000$$

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]

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
EN	Enable of the function block on status 1. The operations <b>Start positioning job (S_)</b> or <b>Jog mode (TP)</b> can be executed when the function block is enabled. Disable the function block on status 0. The function block <b>Reset</b> is carried out when there is falling edge from 1 to 0.	<b>Caution!</b> In normal operation always stop a positioning job via the BR input. In this case, the step sequence frequency will be reduced according to the braking ramp and the motor will brake gently. 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. An active positioning job will be signaled with AC =1.	As a prerequisite, braking must not be activated (BR=0). When a positioning job is activated, the acceleration, operating and braking phases are executed in succession. 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	Braking Aborts the ongoing positioning job when there is a rising edge.	As a prerequisite, jog mode must not be activated (TP=0). After the positioning job is aborted, the function block now runs the braking phase, i.e. a delayed motor stop is executed. The bit output AC is only set to 0 when the braking phase has been completed.  <b>The S_ function block input is not evaluated during the braking phase.</b>
TP	Activates the Jog mode on status 1 The on duration TP = 1 determines the type of Jog mode.	Two operating mode are available in Jog mode for diagnostics and testing. <b>1. Positioning with a preset number of steps</b> TP on duration ≤ 0.5 seconds The motor moves by the number of steps defined at P1. <b>2. Positioning with a specified jog frequency - manual mode</b> TP on duration > 0.5 seconds The motor will be accelerated to the jog frequency set at PF.  <b>The BR function block input is not evaluated during jog mode.</b>
<b>(DWord)</b>		
I1	Number of pulses	For the number of pulses, enter the total number of pulses for the entire sequence, consisting of the three individual acceleration, operation, and braking phases. Integer value range:

## 6. Function blocks

### 6.1 Manufacturer function blocks

	Description	Note
		0 – +2 147 483 647
FS	Start Frequency	Integer value range: 0...5000 Hz
FO	Operating frequency	Integer value range: 0...5000 Hz
RF	Frequency change in the acceleration phase [mHz/step]	Integer value range: 0...65 535 Value for changing the frequency during the acceleration in 0.001Hz per step. Example: 0 = No frequency change 100 = Frequency increased by 0.1 Hz per step
BF	Frequency change in the braking phase [mHz/step]	Integer value range: 0...65 535 Value for changing the frequency during braking in 0.001Hz per step. Example: 0 = No frequency change 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. Integer value range: 0...65 535
PF	Jog frequency	Integer value range: 0...5000 Hz



## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
<b>(bit)</b>		
AC	1: Signals an ongoing positioning job 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.  0: Signals that there is no ongoing positioning job.	
E1	Error output 1: in the event of incorrect parameters, such as: - FO < FS (operating frequency < start frequency) - PF < FS (jog frequency < start frequency)	No positioning commands if the function block detects incorrect parameters. 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.
<b>(DWord)</b>		
QV	Actual number of steps completed	Integer value range: 0...+2 147 483 647
QF	Actual output frequency	Integer value range: 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> 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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

Parameter set	Description	Note
<input checked="" type="checkbox"/> 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.

#### 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

- ▶ 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.
- ➔ The S\_ function block input is not evaluated during the braking phase.

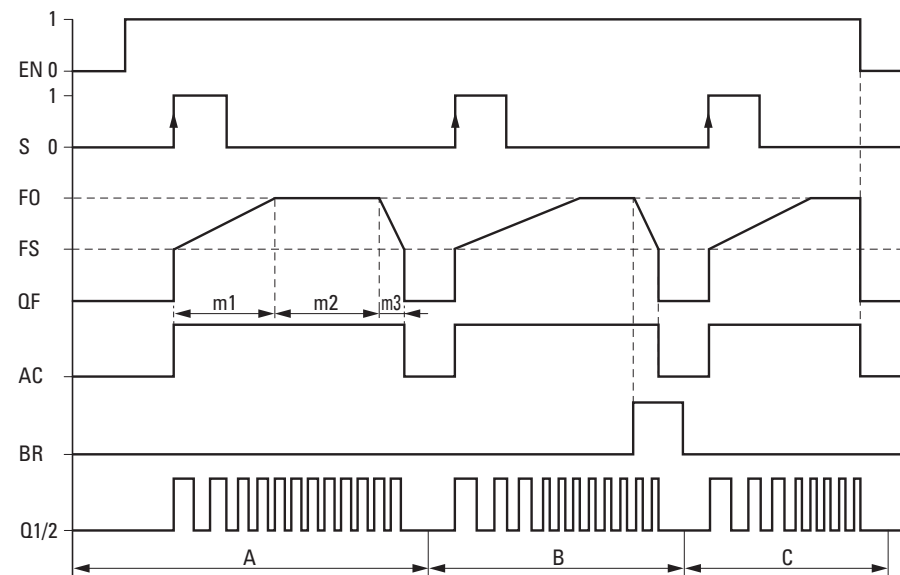


Fig. 199: Signal diagram for PO pulse output with specified number of pulses for I1 - possible normal mode phases

EN: Function block input for enable signal

S: Function block input for starting the pulse sequence

F0: Operating frequency,

FS: start frequency,

QF: Current output frequency

m1 = Acceleration phase, m2 = Operating phase, m3 = Braking phase

AC: Positioning job active

BR: Function block input for stopping the 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 function block input BR initiates the braking phase and reduces the frequency of the pulse sequence.
- Range C: Turning off function block input EN causes the device output to immediately turn off the pulse sequence.

#### 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  $TP \leq 0.5$  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 (0 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Positioning with a specified number of steps P1 (defined distance)

##### TP on time $\leq 0.5$ seconds

When using this operating mode, set the distance using the number of steps P1.

► Switch the TP input to 1 for a duration  $\leq 0.5$  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.

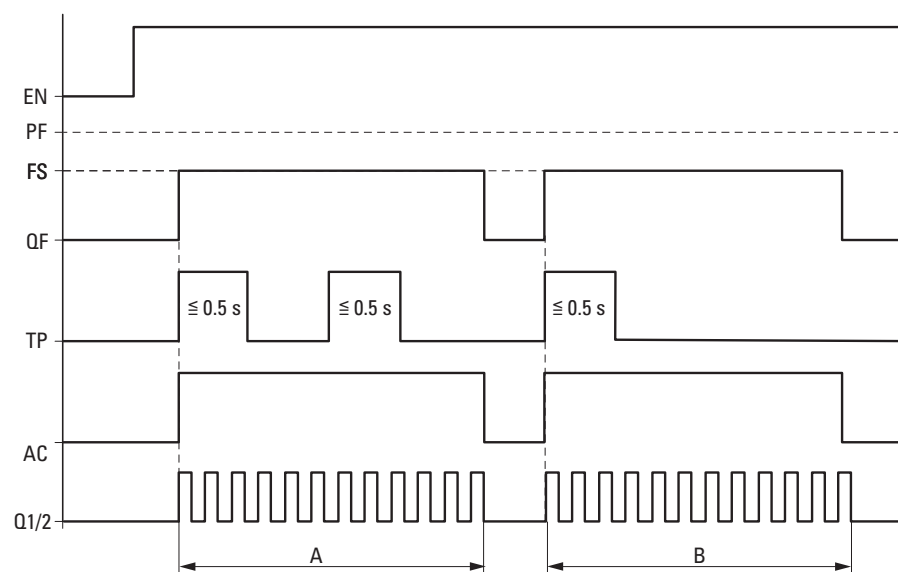


Fig. 200: Signal diagram for jog mode with specified number of steps P1

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  $\leq 0.5$  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.

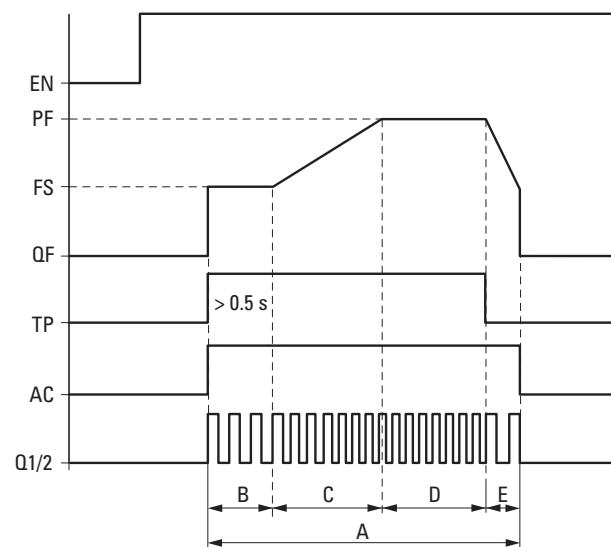


Fig. 201: Signal diagram for jog mode with specified jog frequency, P1 after deceleration phase reached

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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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.

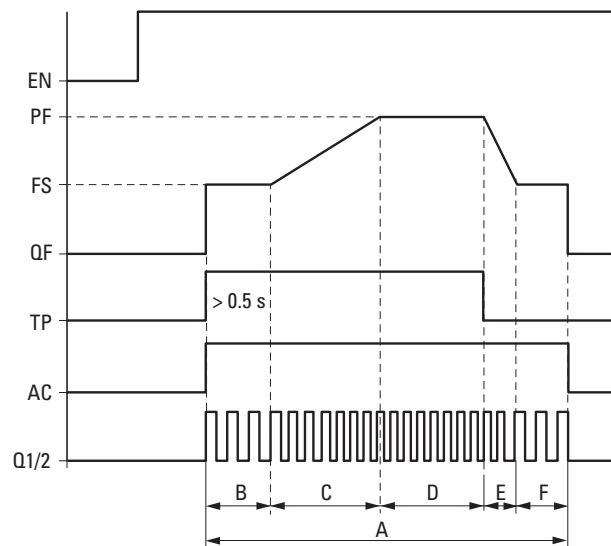


Fig. 202: Signal diagram for jog mode with specified jog frequency, P1 not reached after deceleration phase

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.



The BR function block input is not evaluated during jog mode.

### **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 "BC - Block comparison", page 411
- Section "BV - Boolean operation", page 477
- Section "FT - PT1-Signal smoothing filter ", page 382
- Section "TC - Three step controller", page 402
- Section "VC - Value limitation ", page 407

## 6. Function blocks

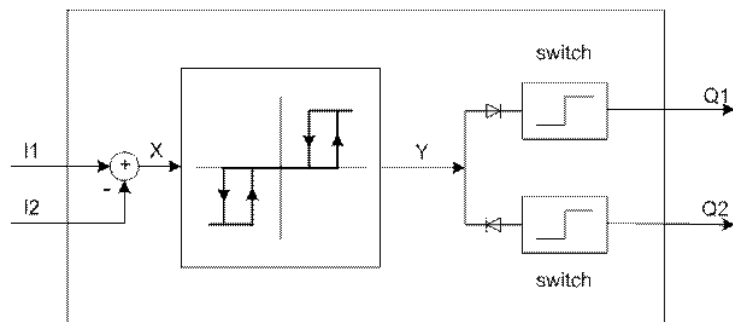
### 6.1 Manufacturer function blocks

#### 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.



TCxx	
EN	Q1
I1	Q2
I2	
H1	
H2	
XH	
TC	

Fig. 203: Three-step controller schematic diagram

I1: Setpoint

I2: Actual value

##### Operating principle

The following timing diagram shows how the three-step controller behaves.

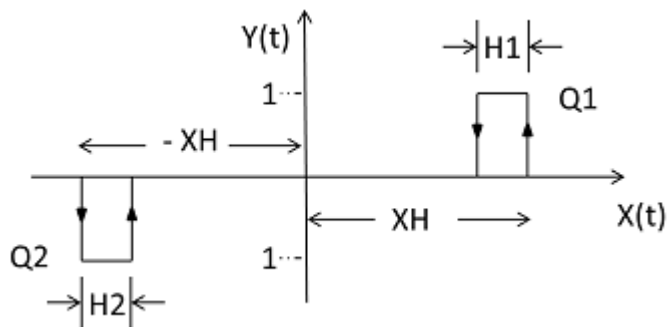


Fig. 204: Timing diagram for three-step controller

XH/ -XH: Distance X from switching point

H1: Hysteresis 1 for XH

H2: Hysteresis 2 for -XH

Y(t): Operating points for Q1/ Q2

## 6. Function blocks

### 6.1 Manufacturer function blocks

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 Q1, or neither one can be switched on at any one time.

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	
<b>(DWord)</b>		
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 Contact distance
TC	Cycle time	0...65535 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.

Operands	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:

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
	EN
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET	x
SN _ Output bit via NET (send)	x
N - Network marker bit	x
nN - NET station n marker	x
ID: Diagnostic alarm	x
LE - Output backlight	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:

Operands	Bit outputs
	Q1, Q2
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET	x
SN _ Output bit via NET (send)	x
N - Network marker bit	x
nN	x
ID: Diagnostic alarm	x
LE - display brightness indicators	x
I Bit input	x
Q Bit output from another FB	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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.

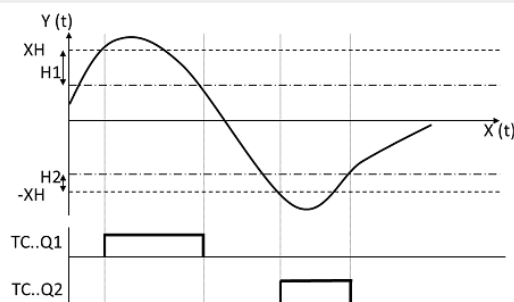


Fig. 205: Signal diagram for three-step controller

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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.

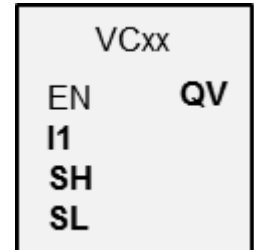
#### See also

- Section "FT - PT1-Signal smoothing filter ", page 382
- Section "VC - Value limitation ", page 407
- Section "BC - Block comparison", page 411
- Section "BV - Boolean operation", page 477
- Section "PO - Pulse output", page 387

#### 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.

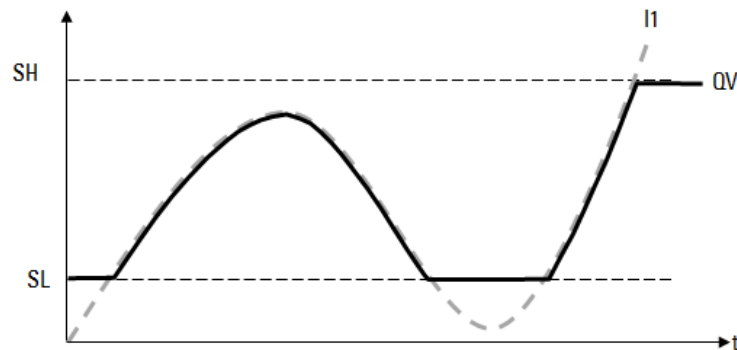


Fig. 206: Figure: Restriction of the input values to the specified limits.

SL: Lower limit  
SH: Upper limit

I1: Input function at I1  
QV: Bounded output function at QV

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		
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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET



## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
(DWord)		
QV	Outputs the value at input I1 within the set limits.	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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> 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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

Parameter set	Description	Note
<input checked="" type="checkbox"/> 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		

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Other

**Retention** - The function block does not recognize retentive data.

#### See also

- Section "DC - PID controller", page 375
- Section "FT - PT1-Signal smoothing filter ", page 382
- Section "TC - Three step controller", page 402
- Section "BC - Block comparison", page 411
- Section "BV - Boolean operation", page 477
- Section "PO - Pulse output", page 387

## 6.1.5 Data and register function blocks

### 6.1.5.1 BC - Block comparison

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, refer to section → Section "Organizing marker ranges", page 234.

BCxx	
EN	EQ
I1	E1
I2	E2
NO	E3

#### 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. In order to ensure that the marker ranges being compared do not overlap, the selected value should not exceed 512 for NO (number of elements).



The marker ranges being compared must not overlap!

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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 8 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

During the configuration stage already, you can ensure that marker ranges will be mapped correctly by clicking on the *Project/Marker area assignment...* menu option.

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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		
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 in bytes to be compared	Integer value range 1...+1024 bytes

##### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> NET station n	x
ID: Diagnostic alarm	x
LE - Output backlight	x
P device buttons	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
I - Bit input	x
Q - Bit output	x
Q - Bit output of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

	Description	Note
(bit)		
EQ	1: if the data ranges are identical. 0: if the data ranges are not identical.	
E1	Error output 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 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 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 1: if the data ranges are identical. 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## **6. Function blocks**

### **6.1 Manufacturer function blocks**

## 6. Function blocks

### 6.1 Manufacturer function blocks

Parameter set		
Parameter set	Description	Note
<input checked="" type="checkbox"/> 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 <div>+ Call enabled</div>	Constants can be edited on the device, as can function block parameters when using the EDP programming language.	
Simulation possible		



## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Other

**Retention** - The function block does not recognize retentive data.

#### Example of a data block comparator BC function block when using the FBD programming language

In this example, the goal is to compare two marker ranges with each other for NO=5 marker bytes. The start addresses will be determined at runtime by the values in MB01 and MB02.

For this example, a constant value of <9> is written to MB01 and a constant value of <19> is written to MB02. Since the offset is counted starting from marker byte MB01, this results in

marker ranges MB10-MB15 and MB20-MB25 being compared.

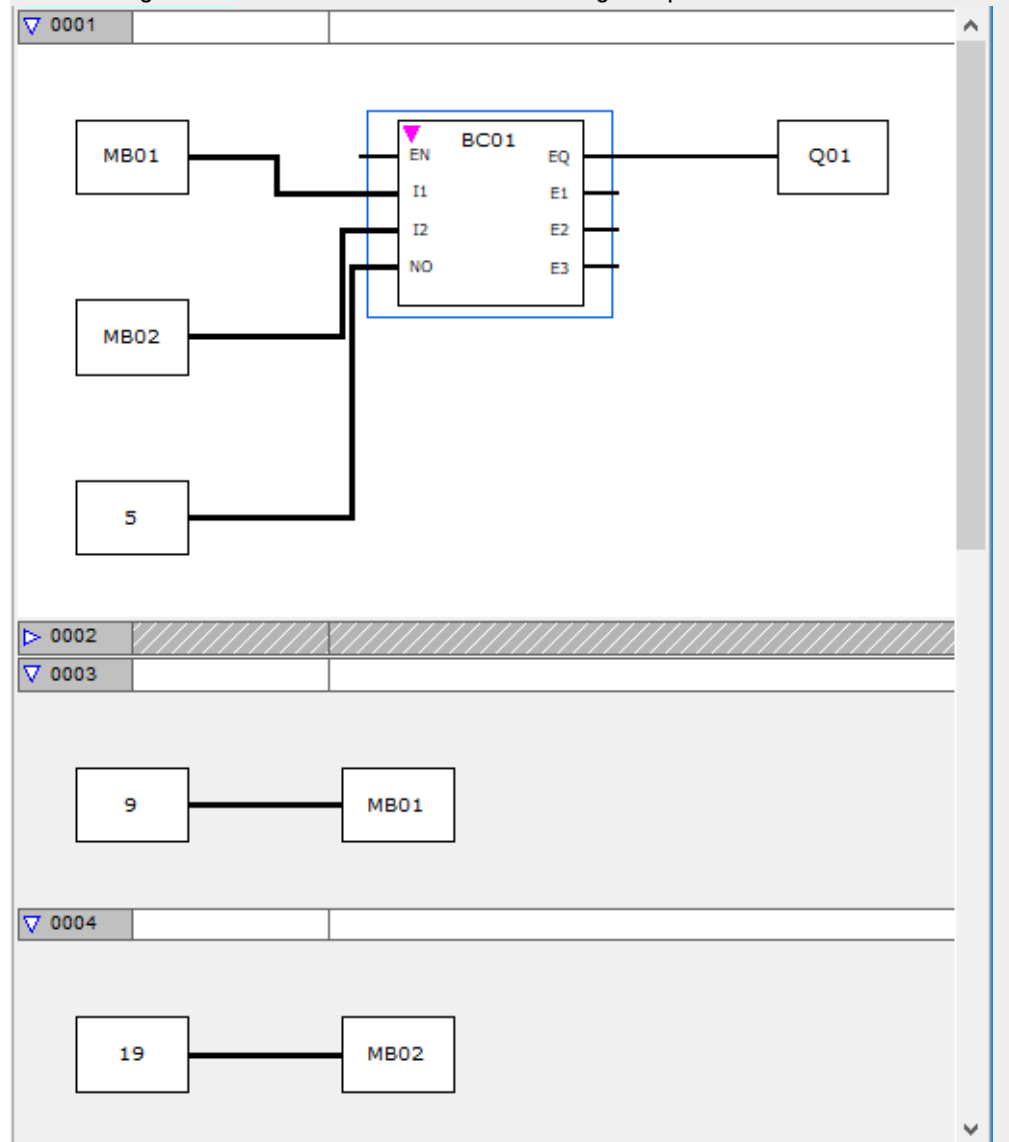


Fig. 207: \*.e80 project with circuit diagram BC in FBD

6. Function blocks

6.1 Manufacturer function blocks

Example of a data block comparator function block when using the EDP programming language

I 05-----Ä BC11EN  
Fig. 208: Wiring the enable coil

BC11E1o  
BC11E2s  
BC11E3j-----Ä M 48  
BC11EQ-BC11EN-----Ä M 49  
Fig. 209: Wiring the contacts

Example of a function block BC 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  
Fig. 210: 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 "BT - Block transfer", page 419
- Section "DB - Data function block", page 425
- "ED - EdgeDetector", page 430
- "FF - Flip-flop ", page 434
- Section "MX - Data multiplexer", page 437
- Section "RE - Recipe records ", page 441
- Section "SR - Shift register", page 447
- Section "TB - Table function", page 455

### 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.

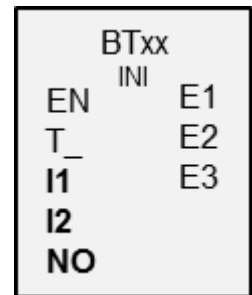
#### General

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 bitwise.



#### 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 address 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Offset calculation for addressing marker words

$$\text{Offset} = \text{MW} (x-1) * 2$$

#### Offset calculation for addressing marker double words

$$\text{Offset} = \text{MD} (x-1) * 4$$

#### 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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
T_	Triggering the transfer on a rising edge	
<b>(DWord)</b>		
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 Operating mode INI: 1...+1024 Byte Operating mode CPY: 1...+1024 Byte

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  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).

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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
<b>(bit)</b>		
E1	Error output 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 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 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 <sup>2)</sup>	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Value outputs
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> 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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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
```

Fig. 211: Parameters on the display

## 6. Function blocks

### 6.1 Manufacturer function blocks

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
> N0	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.

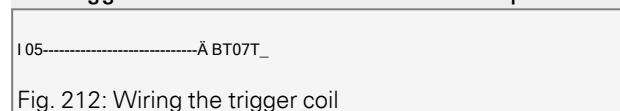


Fig. 212: Wiring the trigger coil

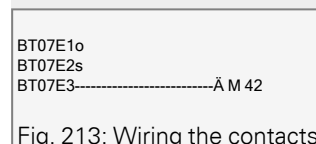


Fig. 213: Wiring the contacts

The messages of the function block are sent as a group message to a marker M42.

#### See also

- "BC - Block comparison", page 411
- Section "DB - Data function block", page 425
- "ED - EdgeDetector", page 430
- "FF - Flip-flop ", page 434
- Section "MX - Data multiplexer", page 437
- Section "RE - Recipe records ", page 441
- Section "SR - Shift register", page 447
- Section "TB - Table function", page 455

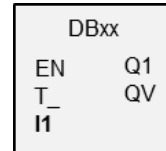


### 6.1.5.3 DB - Data function block

#### 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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> 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.	
<b>(DWord)</b>		
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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND - NET markers <sup>2)</sup> NET station n	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Value inputs
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x
<sup>1)</sup> Only on function blocks T, AC	
<sup>2)</sup> Only on projects with $\geq 2$ base devices on NET	

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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
<sup>2)</sup> Only on projects with $\geq 2$ base devices on NET	

#### Function block outputs

	Description	Note
<b>(bit)</b>		
Q1	1: transfer confirmation, if function block output T_ is 1.	
<b>(DWord)</b>		
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: -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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x
<sup>2)</sup> Only on projects with $\geq 2$ base devices on NET	

You can assign the following operands to the function block outputs that are bit outputs:

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Bit outputs
M – Markers	x
SN – Output bit via NET (send) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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.
<input checked="" type="checkbox"/> No edge evaluation of T bit input		
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

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.

6. Function blocks

6.1 Manufacturer function blocks

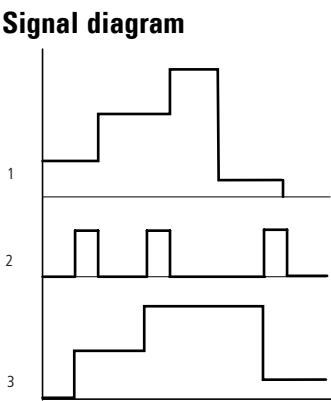


Fig. 214: Signal diagram of data function block

- 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.

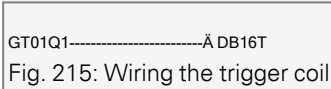


Fig. 215: Wiring the trigger coil

The output of the data function block DB16Q1 is assigned to the input D02 EN of the text display function block.

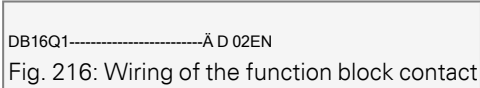


Fig. 216: Wiring of the function block contact

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.

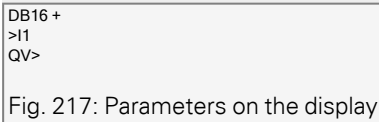


Fig. 217: Parameters on the display

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 Integer value range: -2,147,483,648 to +2,147,483,647
>I2	Outputs the value of DB..I1 when triggered. Integer value range: -2,147,483,648 to +2,147,483,647

**See also**

- "BC - Block comparison", page 411
- Section "BT - Block transfer", page 419
- "ED - EdgeDetector", page 430
- "FF - Flip-flop ", page 434
- Section "MX - Data multiplexer", page 437
- Section "RE - Recipe records ", page 441
- Section "SR - Shift register", page 447
- Section "TB - Table function", page 455
- Section "Organizing marker ranges", page 234
- Section "Retention function", page 651

6. Function blocks

6.1 Manufacturer function blocks

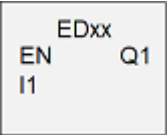
6.1.5.4 ED - EdgeDetector

General

The EdgeDetector function block is available in easySoft 8.25 and higher.

This function block can be used to detect a rising or falling edge.

easyE4 base devices provide 64 EdgeDetector function blocks, ED01 through ED64.



This function block uses its output to indicate, for one single cycle, whether it has detected a rising or falling edge at its input.

If you turn on the ENC option, the enable input will have to be set in order for edges to be detected.

Operating principle

Bit output Q1 is refreshed every cycle based on the state of the I1 change and the function block operating mode.

Output Q1 will be set to a value of one if one of the following conditions is met:

- In operating mode R\_TRIG, output Q1 will be set to 1 for a single cycle if the state of I1 changes from 0 to 1.
- In operating mode F\_TRIG, output Q1 will be set to 1 for a single cycle if the state of I1 changes from 1 to 0.

Output Q1 will be set to a value of zero if the following condition is met:

- No edges have been detected in the current cycle.

The function block and its parameters

Function block inputs

(bit)	Description	Note
EN	1: Activates the function block.	
I1	Input 1: Input bit for which a rising or falling edge must be detected. In R_Trig mode, the output will be set if there is a rising edge. In F_Trig mode, the output will be set if there is a falling edge.	

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Assigning operands

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN _ Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Operating mode

Operating mode	Description	Note
R_TRIG	rising edge detection Detector for rising edges Output Q1 will be set to 1 for a single cycle, if the state of I1 changes from 0 to 1.	Default settings
F_TRIG	falling edge detection Detector for falling edges Output Q1 will be set to 1 for a single cycle, if the state of I1 changes from 1 to 0.	

#### Function block outputs

	Description	Note
(bit)		
Q1	1: Set for a single cycle if an edge is detected at input I1 in conformity with the corresponding operating mode.	

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Assigning operands

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 <sup>2)</sup> via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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 an EdgeDetector function block when using the ST programming language

```
;ED01 (
    EN := I02,
    I1 := I01,
    Q1 =>
);
```

Input I02 turns on the function block enable signal.

Depending on the operating mode, a rising or falling edge at input I01 will result in output ED01Q1 being set to 1 for a single cycle

If output Q1 of the function block is connected, e.g., to Q01, the following applies:



In R\_trig mode, a rising edge at input I01 will cause a state of 1 to be output to output Q01 via Q1.

In F\_trig mode, a falling edge at input I01 will cause a state of 1 to be output to output Q01 via Q1.

#### Other

```
Q01 := Q01 AND ( NOT I01 );
```

This is an ST statement that reproduces the function of the ED function block in R\_TRIG mode.

```
Q01 := Q01 OR ( I01 );
```

This is an ST statement that reproduces the function of the ED function block in F\_TRIG mode.

#### See also

- "BC - Block comparison", page 411
- Section "BT - Block transfer", page 419
- Section "DB - Data function block", page 425
- "FF - Flip-flop ", page 434
- Section "MX - Data multiplexer", page 437
- Section "RE - Recipe records ", page 441
- Section "SR - Shift register", page 447
- Section "TB - Table function", page 455
- Section "Organizing marker ranges", page 234

6. Function blocks

6.1 Manufacturer function blocks

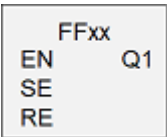
6.1.5.5 FF - Flip-flop

General

The flip-flopfunction block, also known as a bistable multivibrator, makes it possible to store the state of a single bit.

The FF flip-flop function block is a level-triggered SR or RS flip-flop and is available in easySoft 8.25 and higher.

easyE4 base devices provide 64 flip-flop function blocks, FF01 through FF64.



The operating mode is used to define which input will be the dominant one if both inputs are set at the same time, i.e., whether SET or RESET will prevail.

If you turn on the ENC option, the enable input will have to be set in order for edges to be detected.

Operating principle

A rising edge at SE will set output Q1. A rising edge at RE will clear output Q1.

Set dominant: If both inputs are set, output Q1 will be set.

Reset dominant: If both inputs are set, output Q1 will be cleared.

As a result of its bistability, the flip-flop FB can store a data volume of one bit for an unlimited amount of time. However, in contrast to non-volatile memory, this requires for the FB to be continuously powered.

The value that the flip-flop FB is currently storing can be retrieved at output Q1.

A rising edge at SE will cause the function block to store a state of 1. Meanwhile, a rising edge at RE will cause the function block to store a state of 0.

- Set dominant:  
If both inputs are set at the same time, a state of 1 will be stored in the function block.
- Reset dominant:  
If both inputs are set at the same time, a state of 0 will be stored in the function block.

The function block and its parameters

Function block inputs		
(bit)	Description	Note
EN	1: Activates the function block.	
SE	Set: Set input	
RE	Reset: Reset input	

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Assigning operands

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN _ Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Operating mode

Operating mode	Description	Note
SR	Set-dominant If both inputs are set at the same time, output Q1 will be set to a state of 1.	Default settings
RS	Reset-dominant If both inputs are set at the same time, output Q1 will be set to a state of 0.	

#### Function block outputs

(bit)	Description	Note
Q1	1: The flip-flop function block will store a state of 1 if the SE input detects a rising edge 0: The flip-flop function block will store a state of 0 if the RE input detects a rising edge 0/1: Depends on the mode when there is a rising edge at SE and RE at the same time	

#### Assigning operands

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 <sup>2)</sup> via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Parameter set

Parameter set	Description	Note
<input checked="" type="checkbox"/> 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	The operating mode can be selected EDP programming language: Function block parameters and constants can be edited on the device.	Enabled/disabled
Simulation possible		

#### Other

**Retention** - The function block does not recognize retentive data.

#### Example of a flip-flop function block when using the ST programming language

```
FF01 (
    EN := I03,
    SE := I01,
    RE := I02,
    Q1 =>
);
```

Input 1 and input 2 are assigned and switch when the condition for input 3 is met. Input I03 turns on the function block enable signal.

Rising edges at input I01 or I02 will respectively set or clear output FF01Q1

If output Q1 of the function block is connected, e.g., to Q01, the following applies:

When there is a rising edge at input I01, flip-flop FF01 will store a state of 1 and output this state to output Q01 via Q1.

When there is a rising edge at input I01, flip-flop FF01 will store a state of 0 and output this state to output Q01 via Q1.

#### See also

- "BC - Block comparison", page 411
- Section "BT - Block transfer", page 419
- Section "DB - Data function block", page 425
- "ED - EdgeDetector", page 430
- Section "MX - Data multiplexer", page 437
- Section "RE - Recipe records ", page 441
- Section "SR - Shift register", page 447
- Section "TB - Table function", page 455
- Section "Organizing marker ranges", page 234

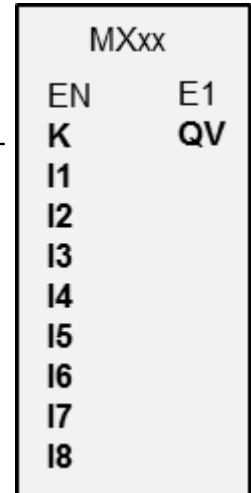
### 6.1.5.6 MX - Data multiplexer

#### 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 function block input, 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 function block input, 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						
<b>(bit)</b>								
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled						
<b>(DWord)</b>								
K	Channel number References the function block input you want (I1 through I8). <table><thead><tr><th>channel</th><th>Function Block Input</th></tr></thead><tbody><tr><td>0</td><td>I1</td></tr><tr><td>1</td><td>I2</td></tr></tbody></table>	channel	Function Block Input	0	I1	1	I2	Integer value range: 0...7
channel	Function Block Input							
0	I1							
1	I2							

## 6. Function blocks

### 6.1 Manufacturer function blocks

	Description	Note
	2 i3	
	3 I4	
	4 I5	
	5 I6	
	6 I7	
	7 I8	
I1...I8	Input value	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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
<b>(bit)</b>		
E1	Error output 1: with incorrect parameters , if 0 > K or K > 7	Output QV is reset to 0 if there is a parameter error.
<b>(DWord)</b>		
QV	Output value of the selected channel	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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> 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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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		

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Other

**Retention** - The function block does not recognize retentive data.

#### See also

- "BC - Block comparison", page 411
- Section "BT - Block transfer", page 419
- Section "DB - Data function block", page 425
- "ED - EdgeDetector", page 430
- "FF - Flip-flop ", page 434
- Section "RE - Recipe records ", page 441
- Section "SR - Shift register", page 447
- Section "TB - Table function", page 455
- Section "Organizing marker ranges", page 234



#### 6.1.5.7 RE - Recipe records

Only available on easySoft Version 7.10 or higher.

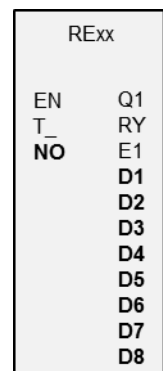
If this function block is not being shown in the leftmost pane in easySoft 8, make sure that you are using firmware version 1.10 or higher for the project.

##### 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 output per function block instance at outputs D1 through D8. Each record (recipe) consists of eight 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 8.

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

## 6. Function blocks

### 6.1 Manufacturer function blocks

NO	Data1	Data2	Data3	Data4	Data5	Data6	Data7	Data8
5	-84987	-31789	-5255	-45	768	235	66	-234
...	...	...	...	...	...	...	...	...
32	-89365	-356978	21	-13	34999	-476	35879	-637



Records cannot be modified at runtime.

#### Marker linking and input assistance

Only available on firmware version 2.00 or higher.

Only available on easySoft 8 or higher.

The records in a recipe can be linked with markers. As soon as you click on the **Default...** button, an input assistance dialog box will appear so that you can select eight consecutive values, marker bytes, marker words, or marker double words. You can also select the start of the operand number. After this, the record will be added to the recipe as per your selection. You can then edit the record, meaning that a mix of marker types and values is also possible for a record.

The linked markers will be listed accordingly in the cross-reference list.

They will also be entered accordingly into the marker range mapping – please refer to → "Marker range mapping", page 235 as well.

		D1	D2	D3	D4	D5	D6	D7	D8		
1	Preset...	20	30	40	50	600	70	80	90	-	+
2	Preset...	MB1	MB2	MB3	MB4	MB5	MB6	MB7	MB8	-	+
3	Preset...	MW1	MW2	MW3	MW4	MW5	MW6	MW7	MW8	-	+
4	Preset...	MD1	MD2	MD3	MD4	MD5	MD6	MD7	MD8	-	+
5	Preset...	0	MB2	MW3	MD4	55	MB6	MD1	MB8	-	+
6	Preset...									-	+

Fig. 218: Recipe with five records; record 5 contains a mixture of values, marker bytes, marker words, and marker double words

## The function block and its parameters

### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block. 0: Resets all function block outputs.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
T_	Trigger input 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.	
<b>(DWord)</b>		
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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> NET station n	x
ID: Diagnostic alarm	x
LE - Output backlight	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
P device buttons	x
I - Bit input	x
Q - Bit output	x
Q - Bit output of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

	Description	Note
<b>(bit)</b>		
Q1	1: if T_=1; 0: if T_=0 or E1=1 or EN=0;	
RY	1: The record for the requested recipe with number NO has been loaded. 0: No recipes are loaded. 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 1: If the recipe with requested number NO does not exist or the value range of NO is exceeded. 0: As soon as there is a value at NO with which the record in a valid recipe can be addressed.	
<b>(DWord)</b>		
D1...D8	Values in the record of the recipe selected with NO.	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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Bit outputs
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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. 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 + Call enabled	Constants can be edited on the device, as can function block parameters when using the EDP programming language.	
Simulation possible		

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Other

#### Retention

Recipes are parts of the parameter set and are accordingly stored retentively as part of the project.

#### See also

- "BC - Block comparison", page 411
- Section "BT - Block transfer", page 419
- Section "DB - Data function block", page 425
- "ED - EdgeDetector", page 430
- "FF - Flip-flop ", page 434
- Section "MX - Data multiplexer", page 437
- Section "SR - Shift register", page 447
- Section "TB - Table function", page 455
- Section "Organizing marker ranges", page 234

### 6.1.5.8 SR - Shift register

#### General

easyE4 base devices provide 32 shift register function blocks, SR01 through 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.

SRxx BIT		SRxx DWORD	
EN	Q1	EN	D1
FP	Q2	FP	D2
BP	Q3	BP	D3
RE	Q4	RE	D4
FD	Q5	I1	D5
BD	Q6	I2	D6
	Q7		D7
	Q8		D8

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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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

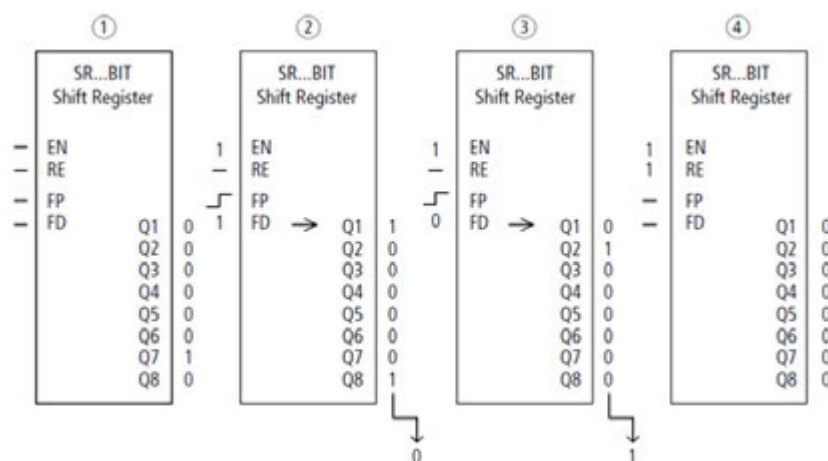


Fig. 219: Shift register SR...: Forwards operation in BIT operating mode

- 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



## 6. Function blocks

### 6.1 Manufacturer function blocks

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

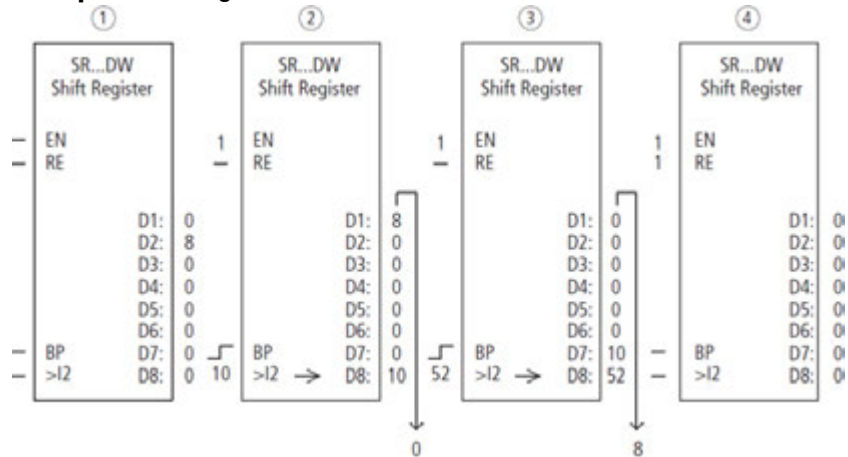


Fig. 220: Shift register SR... Backwards operation in DW operating mode

- 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
FP	Forward Pulse Single Clock input, shift register forward	
BP	Backward Pulse Clock input, shift register backward	
RE	Reset 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	
<b>(DWord)</b>		
I1	Input value for shift direction forwards	Integer value range: -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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> 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
<b>(bit)</b>		
Q1...Q8	Output of the bit register fields 1 - 8	
<b>(DWord)</b>		
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 <sup>2)</sup>	x
NET station n	

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Value outputs
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> 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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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.

#### 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.



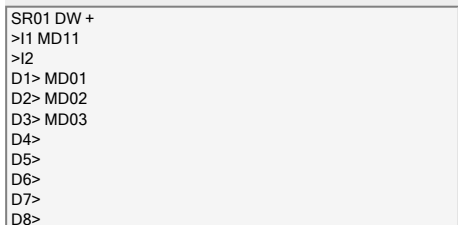
```
-----A SR01EN
M 09-----A SR01FP
```

Fig. 221: 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>
```

Fig. 222: Parameters on the device display

## 6. Function blocks

### 6.1 Manufacturer function blocks

Enter the function block settings here. The display contains the following elements:

SR01	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: Integer value range: -2,147,483,648 to +2,147,483,647
>I2	Input value DW backwards: Integer value range: -2,147,483,648 to +2,147,483,647
D1>	Register value 1 of the shift register, Integer value range: -2,147,483,648 to +2,147,483,647 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

- "BC - Block comparison", page 411
- Section "BT - Block transfer", page 419
- Section "DB - Data function block", page 425
- "ED - EdgeDetector", page 430
- "FF - Flip-flop ", page 434
- Section "MX - Data multiplexer", page 437
- Section "RE - Recipe records ", page 441
- Section "TB - Table function", page 455
- Section "Organizing marker ranges", page 234
- Section "Retention function", page 651

### 6.1.5.9 TB - Table function

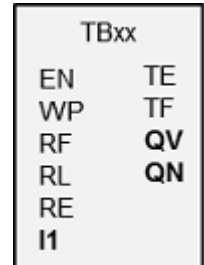
#### General

easyE4 base devices provide 32 table 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 module inputs EN and WP 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 function block input, 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 function block input 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
WP	Trigger coil 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 Read First 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 Read Last 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 Rising edge: The entire table will be cleared. Function block output QN will be set to 0.	
<b>(DWord)</b>		
I1	Input value to be transferred to the table.	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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:



## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### The function block and its parameters

##### Function block outputs

	Description	Note
<b>(bit)</b>		
TE	1: if the table is empty.	
TF	1: if the table is full.	
<b>(DWord)</b>		
QV	Read operation: The value read from the start or end of the table. 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Bit outputs
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

Configuration/time range	Description	Note
<input checked="" type="checkbox"/> 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 table function when using the EDP programming language

You must be 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

- "BC - Block comparison", page 411
- Section "BT - Block transfer", page 419
- Section "DB - Data function block", page 425
- "ED - EdgeDetector", page 430
- "FF - Flip-flop ", page 434
- Section "MX - Data multiplexer", page 437
- Section "RE - Recipe records ", page 441
- Section "SR - Shift register", page 447
- Section "Organizing marker ranges", page 234
- Section "Retention function", page 651

## 6. Function blocks

### 6.1 Manufacturer function blocks

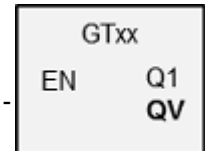
#### 6.1.6 NET Function Blocks

##### 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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> 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

## 6. Function blocks

### 6.1 Manufacturer function blocks

	Description	Note
<b>(DWord)</b>		
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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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.	
Function Block Input	NET-ID: The number of the transmitting NET station.  PT: Number of the transmit function block (e.g. PT 20) by which the transmitting NET station puts a value onto the NET.	Value range: 01...08  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 464
- Section "SC - Synchronizing clock via NET", page 468
- Section "Setting up a NET group", page 721

## 6. Function blocks

### 6.1 Manufacturer function blocks

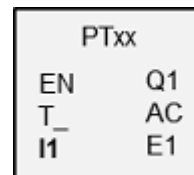
#### 6.1.6.2 PT - Put values to NET

##### 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
<b>(bit)</b>		
EN	1: Activates the function block.	
T_	Trigger coil When there is a rising edge, the function block will temporarily save the input value at I1 and pass it to the NET.	
<b>(DWord)</b>		
I1	Input value that is to be put onto the NET.	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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND - NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
<b>(bit)</b>		
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 1: If the value could not be sent and the previously set AC output changes from a state of 1 back to 0. 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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

## 6. Function blocks

### 6.1 Manufacturer function blocks

	Description	Note
		0 or 1 depending on the function block.
<input checked="" type="checkbox"/> 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. 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 <input checked="" type="checkbox"/> No edge evaluation of T bit input is enabled.	
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.

#### See also

- Section "Setting up a NET group", page 721
- Section "GT - Get values from NET", page 460
- Section "SC - Synchronizing clock via NET", page 468

## 6. Function blocks

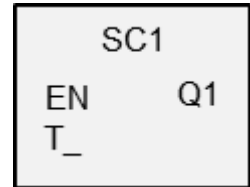
### 6.1 Manufacturer function blocks

#### 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
<b>(bit)</b>		
EN	1: Activates the function block.	
T_	Trigger coil 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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> 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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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 <input type="checkbox"/> 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 "Setting up a NET group", page 721
- Section "GT - Get values from NET", page 460
- Section "PT - Put values to NET", page 464

6. Function blocks

6.1 Manufacturer function blocks

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.

Only available on firmware version 2.00 or higher.

Operand values can also be sent with a text message by placing the special character \$ before and after each operand, e.g. \$MW01\$. The following operands are supported: I, Q, IA, QA, M, MB, MW, MD, N, NB, NW.

A maximum of 128 operand values can be transmitted for all alarm function blocks in the project being used.

ALxx	
EN	Q1
T_	BY
	E1



#### 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 → "Setting up the e-mail function", page 758.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> 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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
<b>(bit)</b>		
Q1	1: If function block input EN = 1.	
E1	Error output 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 1: The most recent send job is still in progress. 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

Parameter set	Description	Note
<input checked="" type="checkbox"/> 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.
<input checked="" type="checkbox"/> Web server active as long as there is a state of 1 at input EN	The web server will be selectively turned on and off based on AL_EN. In order for this to work, the web	Turning off the web server saves processing time

## 6. Function blocks

### 6.1 Manufacturer function blocks

Parameter set	Description	Note
	server must not be permanently enabled – please refer to → " Activation by program ", page 729	
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. 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. A maximum of 128 operand values can be transmitted for all alarm function blocks in the project being used.	Example: The value of analog input IA01 should be sent in the text:  \$IA01\$
Simulation possible		

#### Other

**Retention** - The function block does not recognize retentive data.

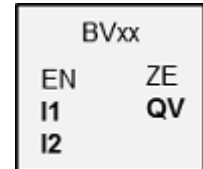
- Section "Setting up the e-mail function", page 758
- Section "BV - Boolean operation", page 477
- Section "D - Text display", page 481
- Section "D - Text display editor", page 491
- Section "DL - Data logger", page 510
- Section "JC - Conditional jump", page 524
- Section "LB - Jump label", page 529
- Section "MC - Acyclical Modbus TCP request", page 531
- Section "MR - Master Reset ", page 543
- Section "NC - Numerical converter", page 563
- Section "ST - Set cycle time", page 569

### 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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		
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. <b>Example</b> -10 (dec) = 10000000 00000000 00000000 00001010 (bin) Two's complement = 11111111 11111111 11111111 11110110 (bin) = FFFFFFF6 (hex) Bit 32 is retained at 1 as a sign bit.
I2	Second value	

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Value inputs
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x
<sup>1)</sup> Only on function blocks T, AC	
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET	

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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
<sup>2)</sup> 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
<b>(bit)</b>		
ZE	Zero 1: if the value of the function block output QV (the operation result) equals zero	
<b>(DWord)</b>		
QV	Result of the operation	

#### Assigning operands

You can assign the following operands to the function block outputs that are numeric outputs:

## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Value outputs
MB, MD, MW – Markers	x
NB, NW, ND – NET markers <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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 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

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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

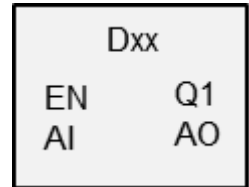
- Section "AL - Alarm function block", page 472
- Section "D - Text display", page 481
- Section "D - Text display editor", page 491
- Section "DL - Data logger", page 510
- Section "JC - Conditional jump", page 524
- Section "LB - Jump label", page 529
- Section "MC - Acyclical Modbus TCP request", page 531
- Section "MR - Master Reset ", page 543
- Section "NC - Numerical converter", page 563
- Section "ST - Set cycle time", page 569



### 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 8, 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. 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
(bit)		
Q1	Returns the state of input EN.	
A0	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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> 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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
Display priority	001...032	001: highest priority 032 lowest priority
Rolling time [s]	001...030	Text display time when the priority is the same
<input checked="" type="checkbox"/> 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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.

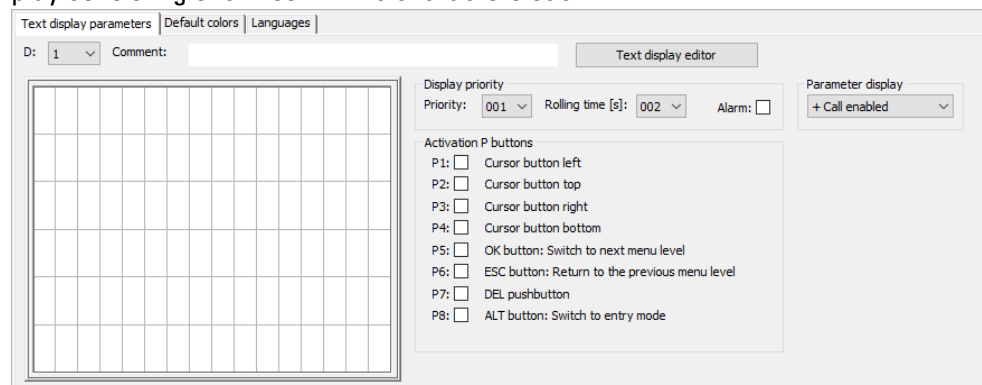


Fig. 223: Text display parameters tab for text display function block in the Programming view

#### 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: → Chapter "6 P buttons", page 649 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 back-light 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.

#### **Selecting default colors**

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.

These will be the default color settings used in the text display editor.

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.

6. Function blocks

6.1 Manufacturer function blocks

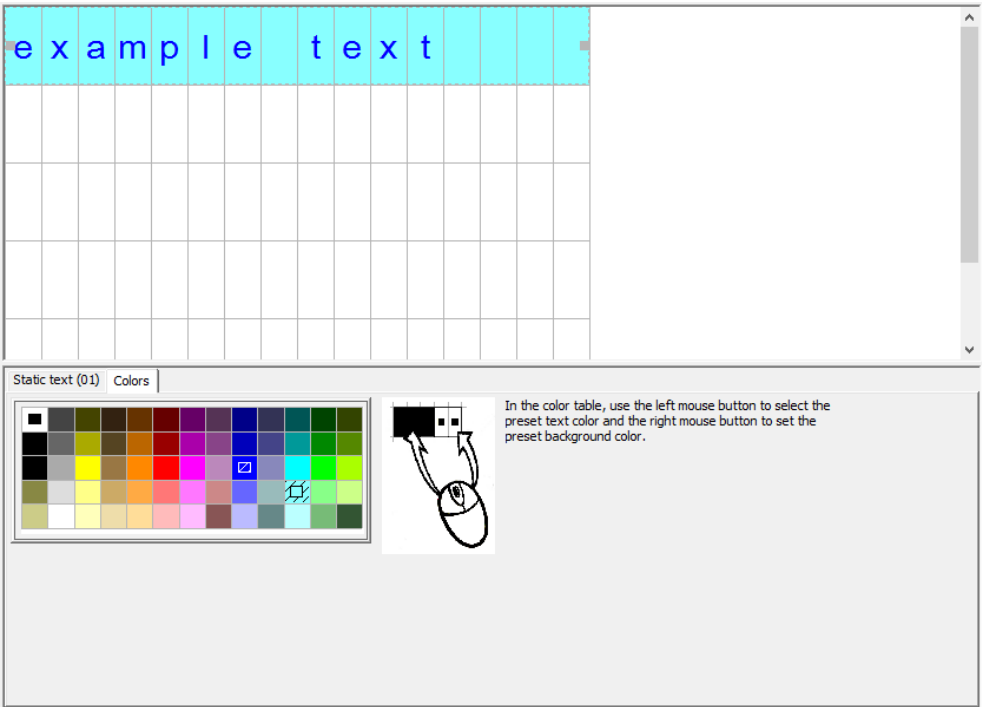


Fig. 224: Default colors tab for text display

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.

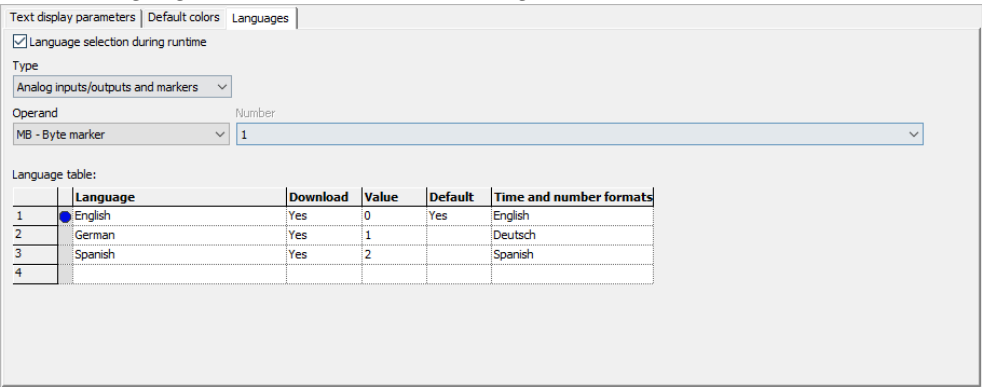


Fig. 225: Text display function block, language tab

☒ **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 <b>text display editor</b> 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 <b>Default</b> 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.

6. Function blocks

6.1 Manufacturer function blocks

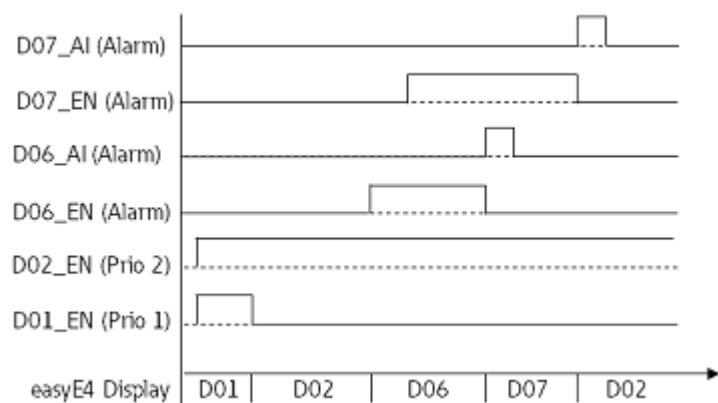


Fig. 226: Signal diagram Text display

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.

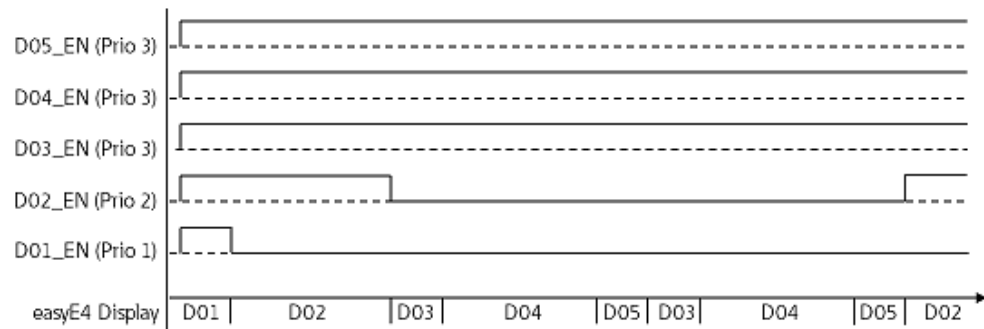


Fig. 227: Signal diagram for text display with text function blocks with an identical priority of 3

Rolling time: D03 = 1s; D04 = 3s; D05 = 1s



## 6. Function blocks

### 6.1 Manufacturer function blocks

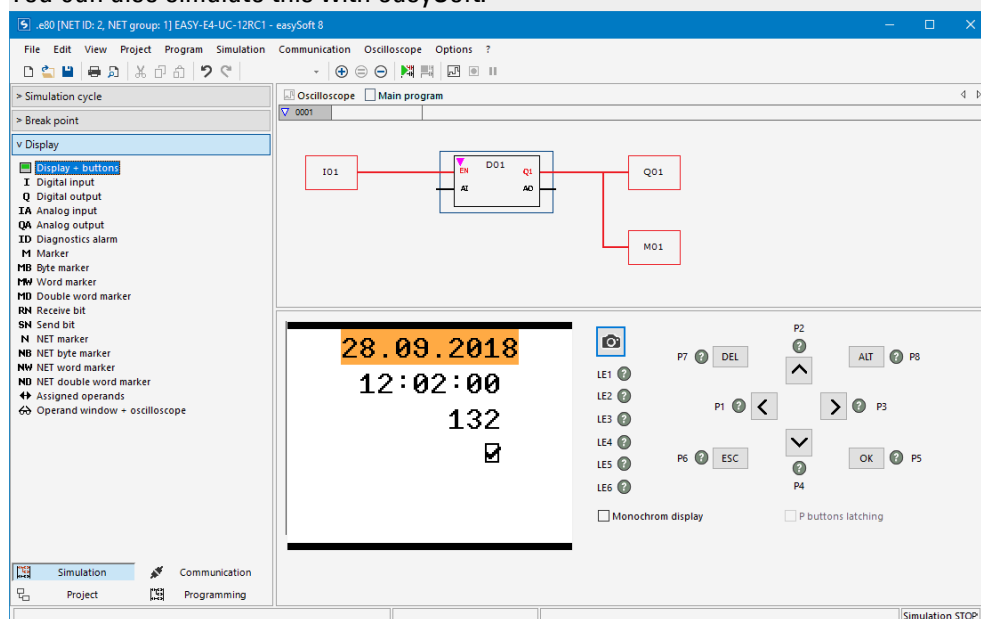
**Retention** - The function block does not recognize retentive data.

#### 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### See also

- Section "AL - Alarm function block", page 472
- Section "BV - Boolean operation", page 477
- Section "D - Text display editor", page 491
- Section "DL - Data logger", page 510
- Section "JC - Conditional jump", page 524
- Section "LB - Jump label", page 529
- Section "MC - Acyclical Modbus TCP request", page 531
- Section "MR - Master Reset ", page 543
- Section "NC - Numerical converter", page 563
- Section "ST - Set cycle time", page 569

#### **6.1.7.4 D - Text display editor**

easySoft 8 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 then 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

## 6. Function blocks

### 6.1 Manufacturer 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>*.

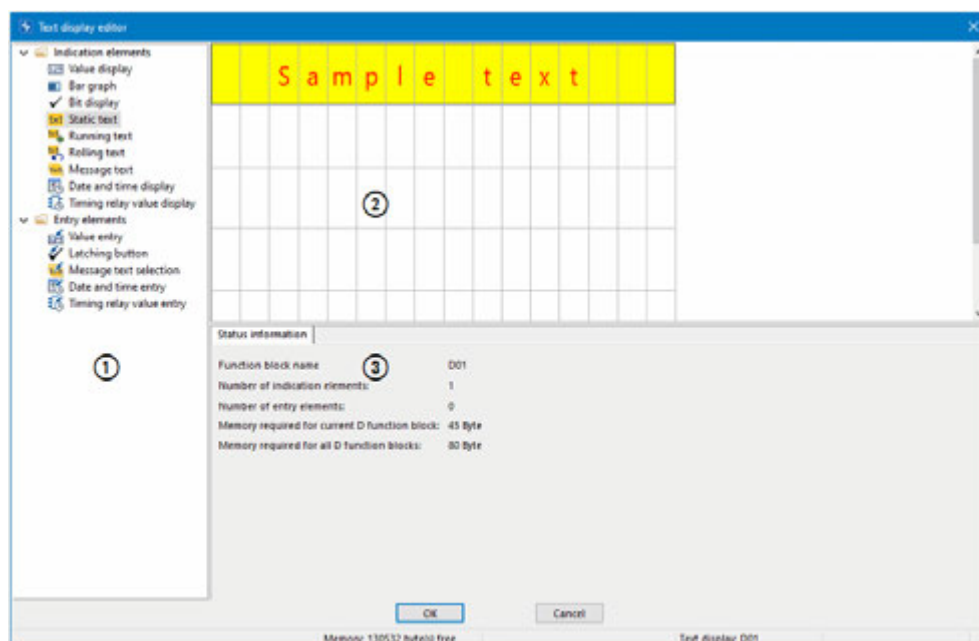


Fig. 228: Text display editor with static text in the first line

- 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 select the "inverse" display type, the colors will be swapped.

Default colors for the Text display editor are set in the *Text display function block- /Default colors tab* – please refer to → "Selecting default colors", page 485 as well.

#### 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.

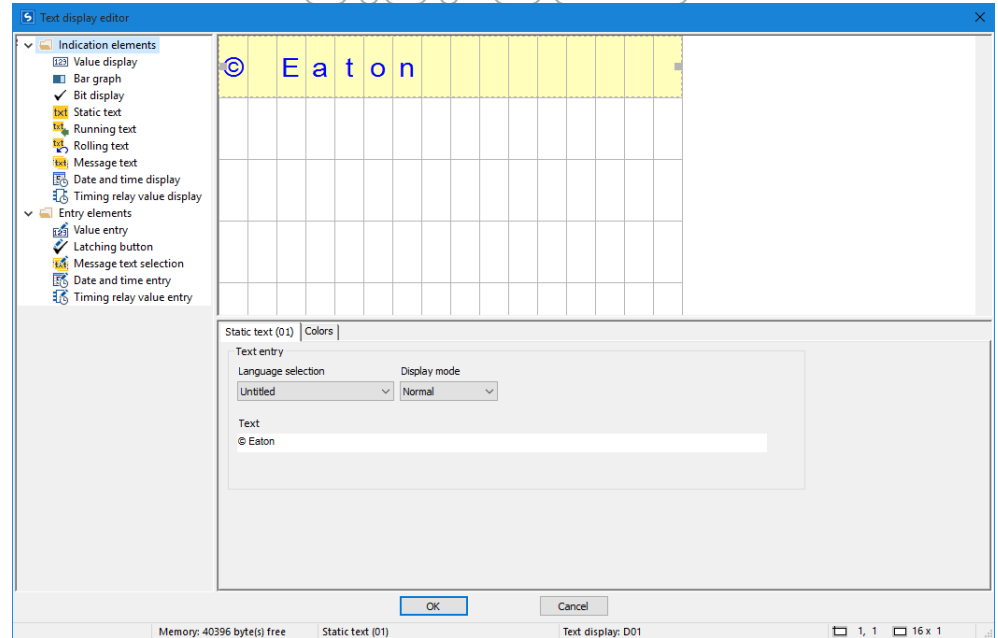


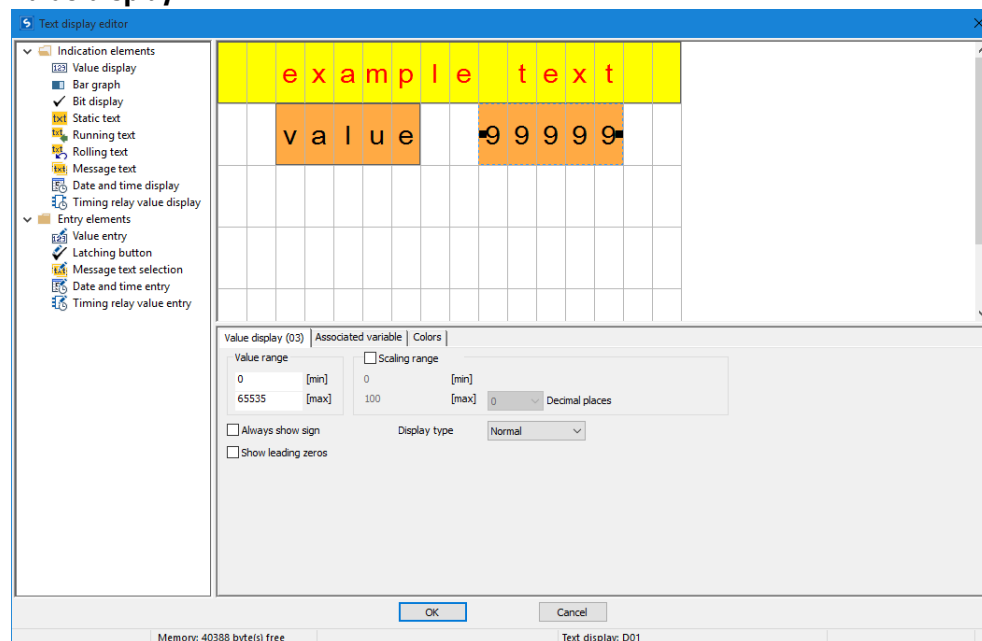
Fig. 229: Character table Special characters

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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.)

Only available on firmware version 2.00 or higher.

With firmware version 2.00 and higher, the value display can be shown with the original character size or double the original character size. To get double the original character size, move the cursor over the lower element handle and drag the handle down over to the next row. To reduce the character size to the original size, move the cursor over the lower element handle and drag the handle up so that the element only takes up one single row. Alternatively, you can use the following context menu options to change the character size: Increase character size and Decrease character size

The character width will be adjusted automatically.

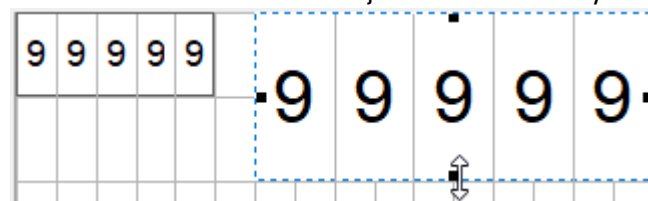


Fig. 230: Value display with original and double character sizes

## 6. Function blocks

### 6.1 Manufacturer function blocks

If multiple value display elements overlap, this will be indicated with red – characters. In addition, the plausibility check will show a corresponding error message.

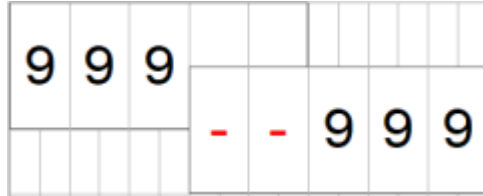


Fig. 231: Two value displays with two characters overlapping

**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.

With firmware version 2.00 and higher, you can also use up to three decimal places for the scaling (in earlier firmware versions, scaling is instead limited to a maximum of two decimal places).

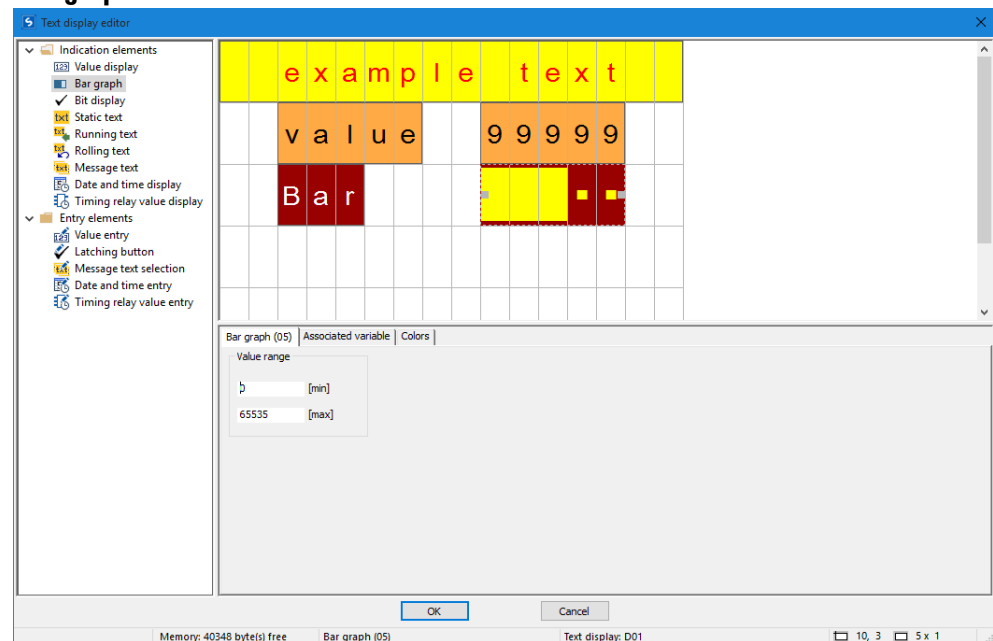
Displays can be customized with a sign and/or leading zeros.

**"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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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.

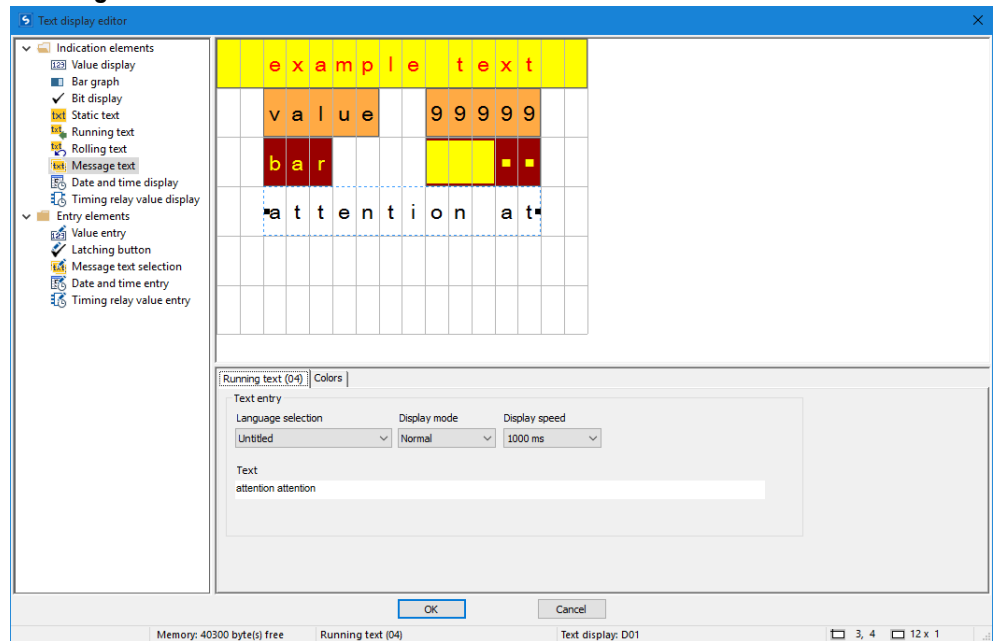
#### Static text

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.

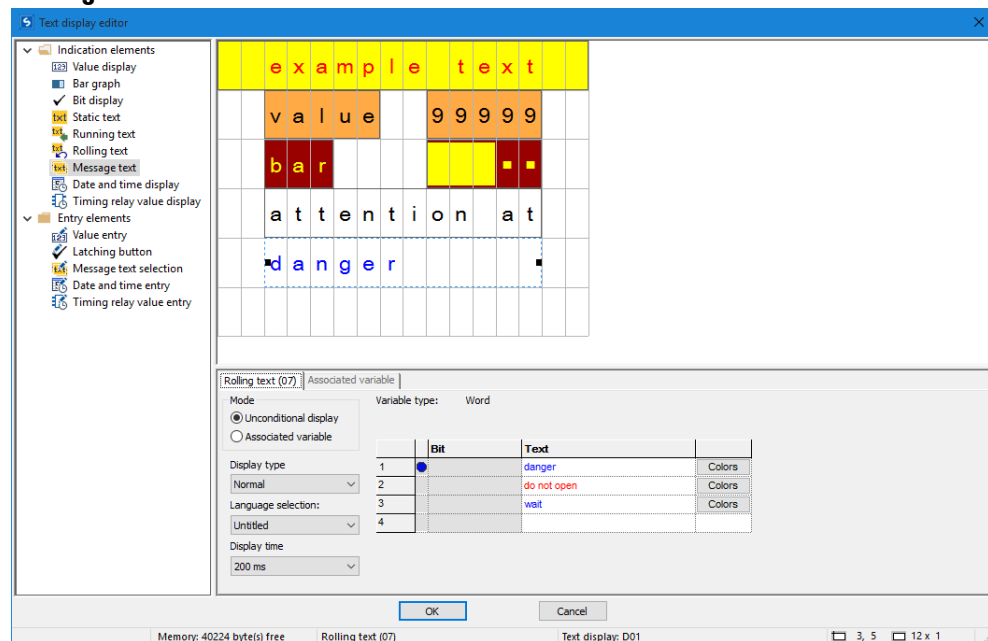
Simply select the running text display element from the list, hold down the left mouse button, and drag the element to the work pane. You can then move your cursor over one of the element handles and drag the handle until the running text element has the size you want.

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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Rolling text



The Display element Rolling Text makes it possible to display consecutive texts in a line. Various messages or even error messages can be displayed that change over in succession in a firmly specified time.

The required texts are entered in the parameter table and colors as well as display mode are selected.



This rolling function only works if at least two rows of text are present.

#### Resize mode

- Unconditional

In this operating mode, the texts are displayed in a time-controlled manner without any further condition, and after the last text entry in the parameter table, they start again with the first one. The clock speed is defined in the Display duration parameter.

- Tag

When this operating mode is selected, the text selection will depend on the application program. 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.

## **6. Function blocks**

### **6.1 Manufacturer function blocks**

During operation, if bit 2 of the operand is now set in the program, text 2 is displayed. If several bits are set in the operand, the associated texts are then shown in succession. The display is switched on for the set display duration. If no bit is set for the operand, no Rolling text is displayed.

6. Function blocks  
6.1 Manufacturer function blocks

Message text

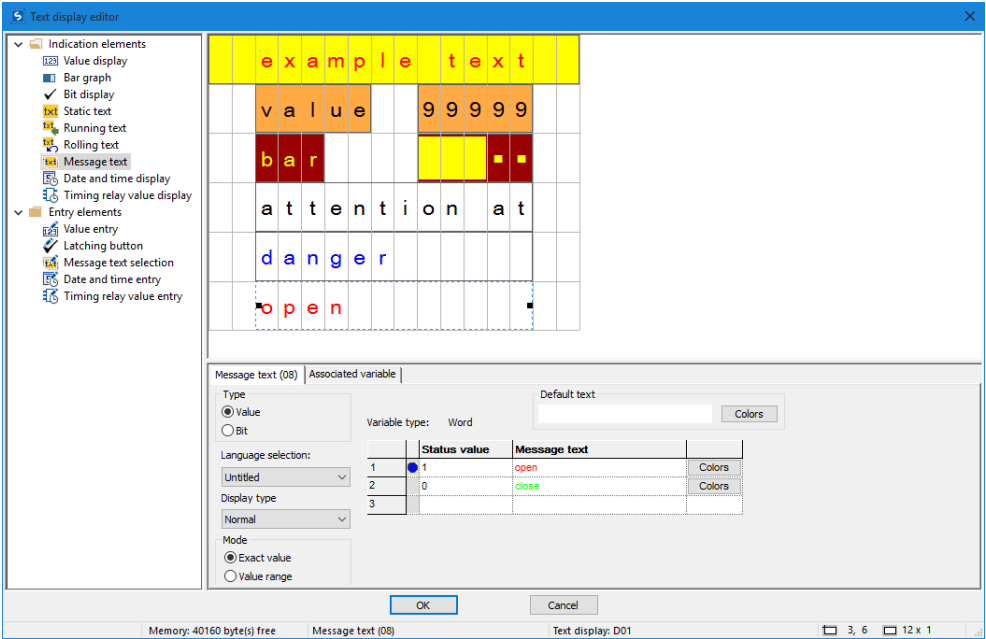


Fig. 232: Example showing an exact value message text

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 226.

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).

6. Function blocks  
6.1 Manufacturer function blocks

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

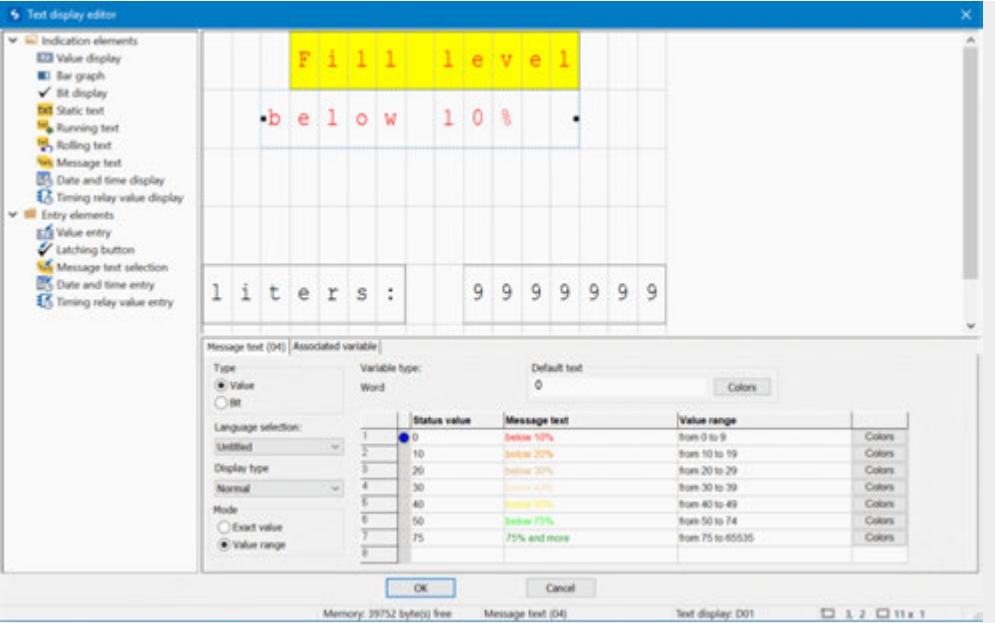


Fig. 233: Value range message text example

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%

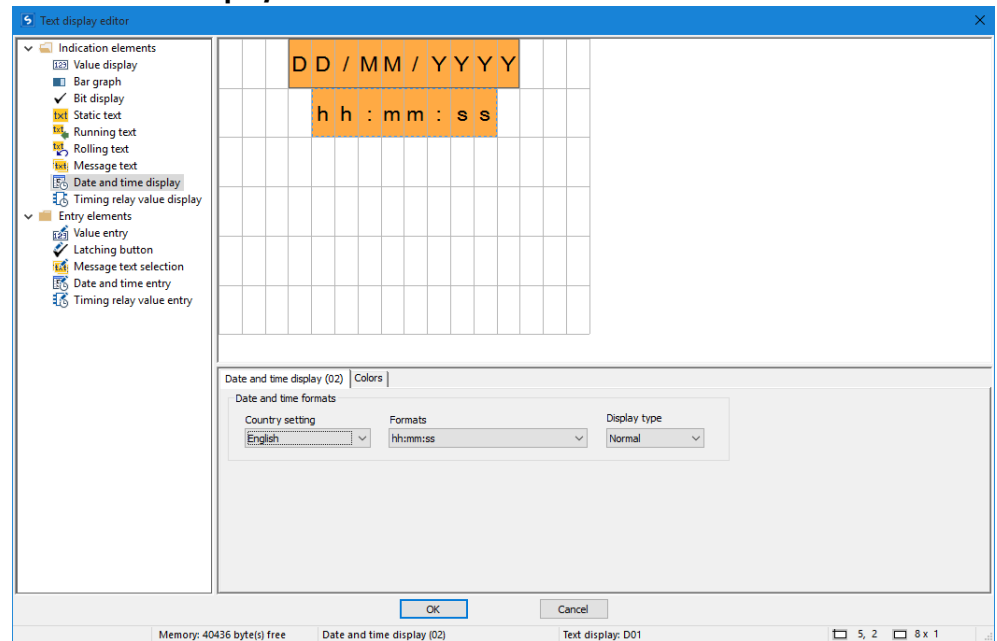
The maximum value will depend on the type of associated variable. In this particular case, it is a marker word with a value range of 0 to 65535.

In this specific example, the default text will not be shown.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Date and time display

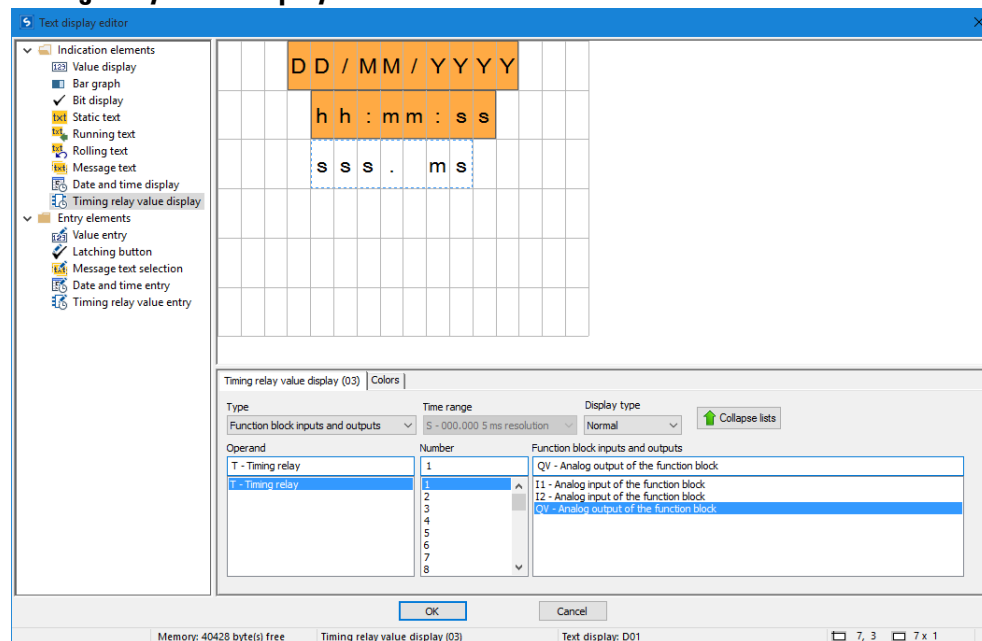


Date and time elements are available in various display formats. Drag a date and time display element to the screen and then select the format you want. The example above contains two data and time display elements configured with the same background color.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Timing relay value display



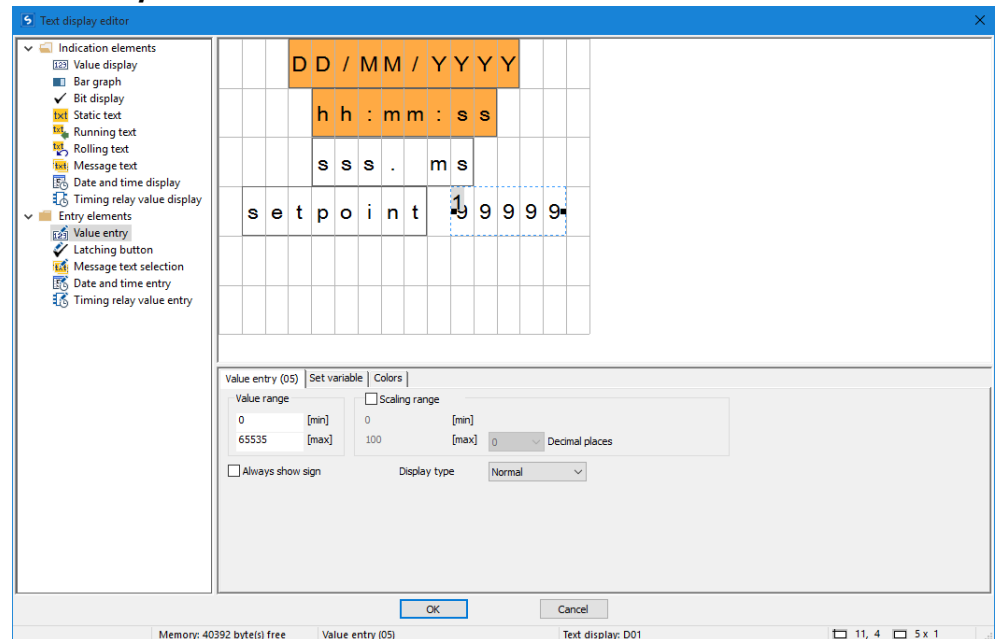
Time functions can be implemented using the timing relay value display elements. You can conveniently display the reference value or the running time value in its own display element. Please note that the number of characters, and the size of the display element accordingly, is fixed. To configure the 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.



## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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  $65535 \cdot 0.4 = 26214$  will be written to the set variable.

#### ☒ Scaling range

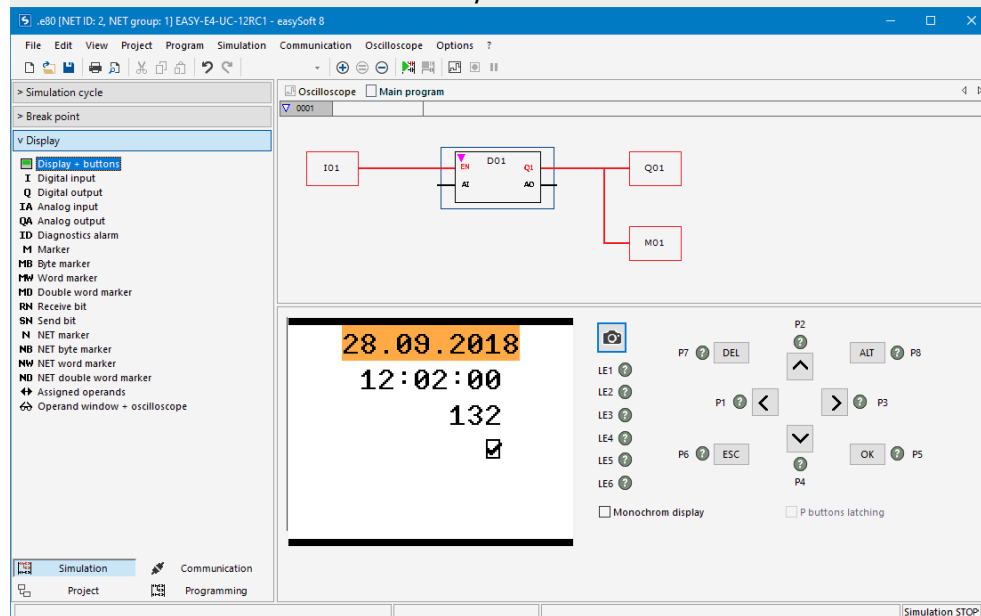
Enabling the checkbox will allow you to configure the scaling range for the value entry element. If you, for example, enter a value of <1000> into the [max] field, the value input will be limited to 4 digits, <9999>.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 8.



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

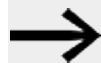
RIGHT: The next smaller decimal place will be selected or the input value to the right or underneath will be selected

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.

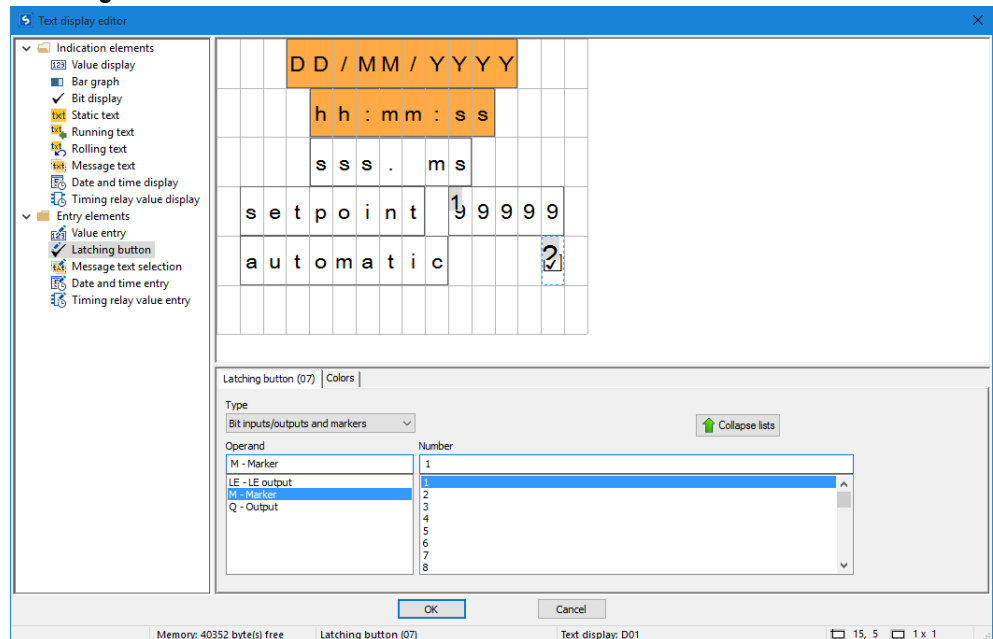
Once a new value is entered, it can be confirmed with **OK**. This will exit input mode.



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).

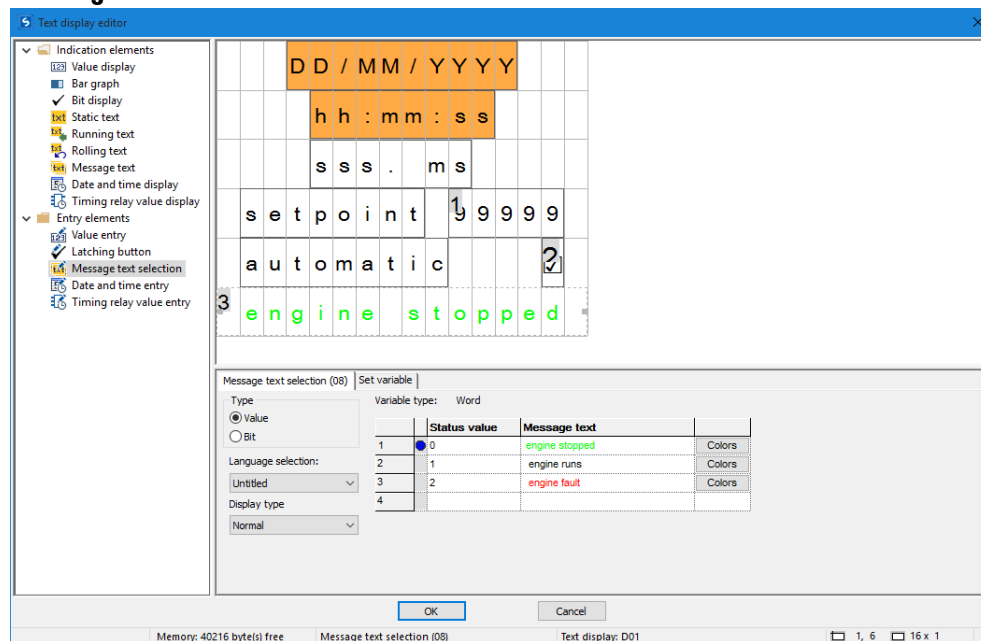
To enter input mode, press the <ALT> button at runtime or during the simulation. You will then be able to use the checkbox by pressing the **P2** or **P4** button, and the binary value will switch between 0 and 1 accordingly.

The superscript <sup>2</sup> 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 506.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 500.

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 506.

#### 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 506

In addition to the display functionality, this 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 506

In addition to the display functionality, this element allows for operator input.

**See also**

- Section "AL - Alarm function block", page 472
- Section "BV - Boolean operation", page 477
- Section "D - Text display", page 481
- Section "DL - Data logger", page 510
- Section "JC - Conditional jump", page 524
- Section "LB - Jump label", page 529
- Section "MC - Acyclical Modbus TCP request", page 531
- Section "MR - Master Reset ", page 543
- Section "NC - Numerical converter", page 563

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 6.1.7.5 DL - Data logger

##### General

easyE4 base devices provide up to eight data logger function blocks, DL01 through DL08. Firmware version 2.25 and lower only supports 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.

DL01	
EN	RY
T1	BY
T2	E1
T3	
T4	
I1	
I2	
I3	
I4	

##### 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  $\Delta I$  parameter for each function block input (I1 through I4) to specify the data change magnitude starting from which logging will be triggered.

Any byte, word, or double word operand can be connected to analog inputs I1 through I4.

All events are saved as data sets in a specified number of files. One file after another gets filled with the specified number of data sets.

There are two storage modes available for selection:

1. Ring buffer  
At the same time as the last file gets filled with the last data set, the first file with all data sets gets deleted. The next data set is written to the first file as the first data set.
2. Until number of log files is reached  
Logging stops at the same time as the last file gets filled with the last data set.

### **Starting a new log session**

Logging is restarted by the following actions for both types of storage:

- Actuating the **Start again** button in the online operating dialog Card Manager in the data logger recordings area while the easyE4 is in its STOP operating mode.
- Actuating the **Start again** button on the web server
- Inserting a new SD card without an existing folder
- Actuating the **Card => PC** button in the online operating dialog Card Manager, data logger recordings area, for downloading the current log file while easyE4 is in its RUN operating mode.
- Downloading the current log file in the Web Client *Web Client/Diagnostics/Data logger* while easyE4 is in its RUN operating mode

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
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.	
<b>(DWord)</b>		
I1	Analog value 1 for storage	
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 data sets may be lost. An important factor to consider within this context is the speed of the memory card being used. When triggered by the T1...T4 inputs function block, the large number of log entries can be controlled by evaluating the function block output BY in the program. The storage process should only get triggered if function block output is BY=0.

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x



## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

(bit)	Description	Note
RY	Ready 0: Logging is active 1: Logging inactive RY = 0 always applies to ring buffers; 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: <ul style="list-style-type: none"> <li>• n log files have been written</li> <li>• The memory card is full</li> <li>• No memory card fitted</li> <li>• Memory card faulty</li> </ul>
BY	Busy 1: Logging not possible 0: Logging possible	Possible causes: <ul style="list-style-type: none"> <li>• The card is currently being written to</li> <li>• The temporary internal buffer is full</li> </ul>
E1	Error output 1: Data loss	Possible causes: <ul style="list-style-type: none"> <li>• No memory card fitted</li> <li>• The memory card does not have enough space for another log file</li> <li>• Memory card faulty</li> <li>• Temporary internal buffer is exceeded by at least one record</li> </ul>

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 <sup>2)</sup> NET station n	x
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Parameter set

	Description	Note
DL	Specifies the DL function block instance	Up to eight instances can be used simultaneously in a single program. Each instance will be assigned an operand number from DL01 to DL08
Comment:	Can be used to enter	a comment for identifying the function block
Directory name of log session	Enter the name of the folder that contains the log files for the corresponding DL function block here, such as <MYLOG1>. A maximum of 8 characters are allowed, and they must conform to Microsoft DOS conventions. The default name is <EASYLOGn>.	In order to ensure that all DL function blocks will be able to carry out logging simultaneously without interfering with each other, each function block of type DL must be assigned a separate directory for logging in the program. A unique directory name must be assigned for the log session.
Storage mode	Ring buffer Until number of log files is reached	
Number of files per log session	A log session contains n log files	Integer value range for 1...1000
Number of data sets per log file	A log file contains n data sets	Integer value range for 1...60 000
Log when input values change	If the changes at DL_I are greater than or equal to $\Delta I > 0$ , a data set will be logged. $\Delta I = 0$ : No logging taking place.	Integer value range for $\Delta I$ : 0...65 535
<input checked="" type="checkbox"/> 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 <b>+ Call enabled</b>	Constants can be edited on the device, as can function block parameters when using the EDP programming language.	
Simulation possible		
<b>Logging setting</b>	Only available on firmware version 2.30 or higher.	
Minimum interval before a new data set is saved	The minimum interval must be shorter than the maximum wait time.	The default setting (milliseconds; 0) will disable this parameter.
Maximum wait time	The maximum wait time should be	The default setting (milliseconds; 0)

## 6. Function blocks

### 6.1 Manufacturer function blocks

	Description	Note
before a new data set is saved	longer than 100 ms in order to make sure that no logs are lost.	will disable this parameter.
Time range	Units: milliseconds, seconds, minutes, hours, days	
Value		Integer value range 0...1000

### Storage mode

You can select between Ring buffer and Until number of log files is reached :

- **Ring buffer**

All events are stored in a specified number of files. One file after another gets filled with the specified number of data sets. At the same time as the last file gets filled with the last data set, the first file is already getting prepared for the next data set and the data sets it contains get deleted. The next data set is written to the first file as the first data set. This prevents any loss of current value data.



On the ring buffer, select the number of files for each log session > 1.

Example of data logger as a ring buffer

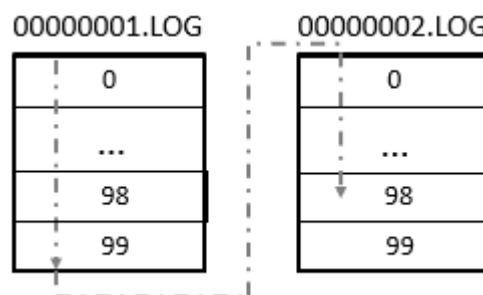
$$\begin{array}{rcl}
 ((\text{Number of files per} & * & (\text{Number of data sets} - (1 \text{ data set})) = \text{Maximum} \\
 \text{log session}) & & \text{per log file)) & \text{number of} \\
 & & & \text{data sets} \\
 & & & \text{per CSV} \\
 & & & \text{file} \\
 (2 & * & 100) & - 1 = 199
 \end{array}$$

If for example 2 files with 100 data sets are defined for a log session, up to 199 data sets can be written and then read out again.

When the 199th data set is written, the 2nd file is closed and the first one is opened for the next data set. This involves the values stored in it getting deleted. This means that the 100 oldest data records can be read out safely.

Here, the individual steps are explained:

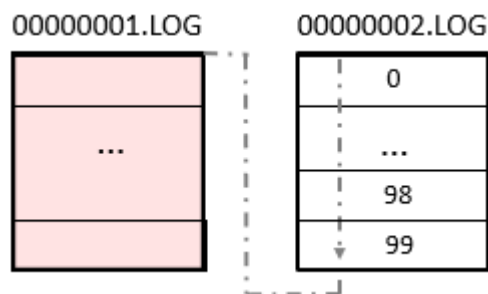
1. A maximum of 100 data sets, data set 0 to data record 99, are entered in 00000001.LOG. Data sets 0 to 98 are then entered in 00000002.LOG.



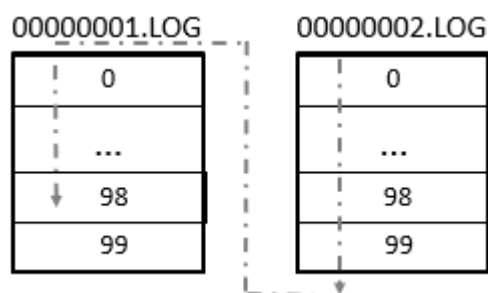
2. Data set 99 is entered in 00000002.LOG and 00000001.LOG is prepared for the next data set.

## 6. Function blocks

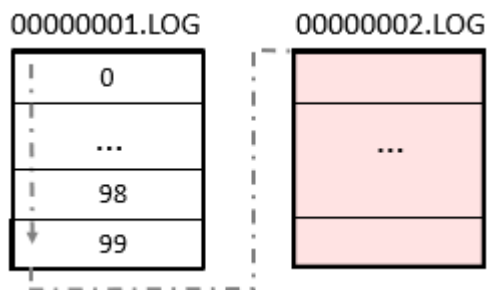
### 6.1 Manufacturer function blocks



3. The next data set is entered in 00000001.LOG. 00000002.LOG remains unchanged.



4. Writing continues in 00000001.LOG and when data set 99 is written, 00000002.LOG is prepared for the next data set.



The process then starts again with step 1.

This means that the 100 oldest data records can be read out safely. In this mode, logging will always be running, i.e., the log session will not be terminated automatically.

Also see → Section "Example of first data logger instance as a ring buffer", page 520

- **Up until number of log files is reached**

Log files will continue to be created one after the other in a single folder until the value defined in the Number of files per log session parameter is reached.

When this number is reached, logging will stop (i.e., the log session will be terminated) and output RY will be set to 1. The name of the log files will be an eight-digit name, with the number counted up starting from 00000001.log.

Data logged with the DL data logger manufacturer function block is stored on the card in a directory. This directory will be named as specified under *Programming view/Data logger - Parameter tab* in the Directory name of log session field.

The data in the log files is binary-encoded and cannot be read with standard Windows PC tools. Within easySoft 8, it can be read with the card manager, where you can see all the logs on the card and convert them to \*.csv format, merge them together, and save them to a file. This file can then be opened and edited with Excel.

The recording of binary data in various log files takes place for security reasons. If a file is corrupt or if the card was pulled out while being written to, only the data records in that file are corrupt. The previous ones are saved securely.

If a PC=>Device or PC=>Card program download is initiated, easySoft 8 will determine which operands are assigned to the DL function block inputs and store the corresponding operand comments in the corresponding log session in a file named "Comments.txt".

#### **Number of files per log session**

The desired number of files that should be logged on the microSD card in each log session is defined in this Number of files per log session parameter.

The maximum possible number is 1000.

#### **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.



Make sure that the number of data sets you select is only as large as necessary so that 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 ( $\Delta I1$  through  $\Delta I4$ ) for the four analog values at DL\_I1 through DL\_I4. Please note that all data will always be logged with each log operation.

6. Function blocks

6.1 Manufacturer function blocks

Programming view/DL01

Data logger - Parameter

DL: 1 Comment:

☐ Function block release by EN is necessary

Log session

Directory name of log session: MY\_LOG1

Storage mode: Ring buffer

Number of files per log session: 0002

Number of data sets per log file: 00100

Log when input values change

Δ I1

00044

Δ I2

00000

Δ I3

00000

Δ I4

00000

Logging Setting

Minimum distance until a new record is stored

Time range

Milliseconds

Value

0

Maximum waiting time until a new record is stored

Time range

Milliseconds

Value

0

Fig. 234: Example of first data logger instance as a ring buffer

In this example, for the ring buffer, two files with 100 data sets are defined for the log session for DL01. Up to 199 data sets can be written to the directory for log session **MY\_LOG1** and then read out again – refer to the → "Example of data logger as a ring buffer", page 517 description as well.



#### Other

**Retention** - The function block does not recognize retentive data.

#### Create log files

Data logged with the DL data logger manufacturer function block is stored on the card in a directory. This directory will be named as specified under *Programming view/Data logger - Parameter tab* in the Directory name of log session field.

The data in the log files is binary-encoded and cannot be read with standard Windows PC tools. Within easySoft 8, it can be read with the card manager, where you can see all the logs on the card and convert them to \*.csv format, merge them together, and save them to a file. This file can then be opened and edited with Excel.

The recording of binary data in various log files takes place for security reasons. If a file is corrupt or if the card was pulled out while being written to, only the data records in that file are corrupt. The previous ones are saved securely.

If a PC=>Device or PC=>Card program download is initiated, easySoft 8 will determine which operands are assigned to the DL function block inputs and store the corresponding operand comments in the corresponding log session in a file named **"Comments.txt"**.

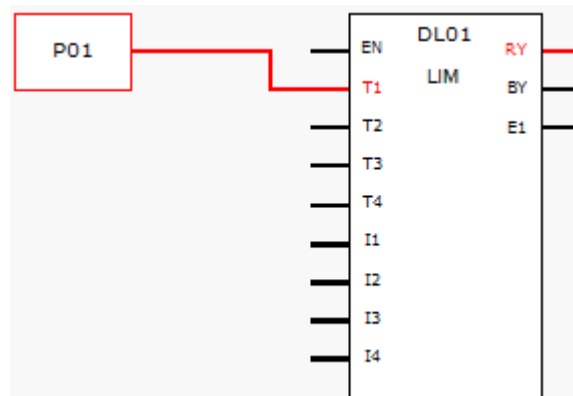


Please keep in mind that only one folder can be created for each log session, even if the number of files per log session is selected as greater than 1 and several binary files are also stored accordingly.

**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



6. Function blocks

6.1 Manufacturer function blocks

Fig. 235: Workspace with function block and device button

- ▶ Click on the function block DL and configure as shown in the following illustration.

Data logger Parameter

DL: 1 Comment:

☐ Function block release by EN is necessary

Log session

Directory name of log session: MYLOG

Storage mode: Until number of log files is reached

Number of files per log session: 003

Number of data sets per log file: 00003

Log when input values change

Δ I1 Δ I2 Δ I3 Δ I4

00000 00000 00000 00000

Fig. 236: 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.

The log files can only be read with easySoft 8.

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	2023-07-26	12:08:40	365	1	0	0	0	0	0	0	0
1	2023-07-26	12:08:40	968	1	0	0	0	0	0	0	0
2	2023-07-26	12:08:42	965	1	0	0	0	0	0	0	0
3	2023-07-26	12:08:43	677	1	0	0	0	0	0	0	0
4	2023-07-26	12:08:45	579	1	0	0	0	0	0	0	0

## 6. Function blocks

### 6.1 Manufacturer function blocks

5	2023-07-26	12:08:46	908	1	0	0	0	0	0	0	0
6	2023-07-26	12:08:51	529	1	0	0	0	0	0	0	0
7	2023-07-26	12:08:52	332	1	0	0	0	0	0	0	0
8	2023-07-26	12:08:53	367	1	0	0	0	0	0	0	0

There are 9 data sets logged in this log file. The logging of all data sets is triggered by a rising flank on digital input DL01T01. Log files do not include any information regarding the operating mode.

The log files can only be read with easySoft 8.

#### See also

- Section "AL - Alarm function block", page 472
- Section "BV - Boolean operation", page 477
- Section "D - Text display", page 481
- Section "D - Text display editor", page 491
- Section "JC - Conditional jump", page 524
- Section "LB - Jump label", page 529
- Section "MR - Master Reset ", page 543
- Section "NC - Numerical converter", page 563

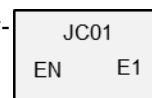
## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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.

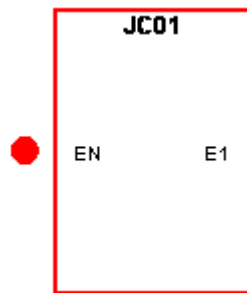


Fig. 237: Activated function block in the function block status display

### 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks



For greater clarity, position the conditional jump function block JC.. in the circuit diagram if possible directly in front of the function blocks to be skipped.

If the error output is to be evaluated, position the standard function block in the standard 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 <input checked="" type="checkbox"/> 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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> NET station n	x
ID: Diagnostic alarm	x
LE - Output backlight	x
P device buttons	x
I - Bit input	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
Q - Bit output	x
Q - Bit output of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Function block outputs

Description	Note
(bit)	
E1	Error 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

Parameter set	Description	Note
–		

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### See also

- Section "AL - Alarm function block", page 472
- Section "BV - Boolean operation", page 477
- Section "D - Text display", page 481
- Section "D - Text display editor", page 491
- Section "DL - Data logger", page 510
- Section "LB - Jump label", page 529
- Section "MR - Master Reset ", page 543
- Section "NC - Numerical converter", page 563
- Section "ST - Set cycle time", page 569



#### 6.1.7.7 LB - Jump label

##### General

This function block is only available when using the EDP (easy Device Programming) programming language. easyE4 base 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.



The JC function block is used in the circuit diagram, while the LB function block is used in the function block diagram.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### 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 472
- Section "BV - Boolean operation", page 477
- Section "D - Text display", page 481
- Section "D - Text display editor", page 491
- Section "DL - Data logger", page 510
- Section "JC - Conditional jump", page 524
- Section "MR - Master Reset ", page 543
- Section "NC - Numerical converter", page 563
- Section "ST - Set cycle time", page 569

#### 6.1.7.8 MC - Acyclical Modbus TCP request

Only available on easySoft Version 7.30 or higher.

If this function block is not being shown in the leftmost pane in easySoft 8, make sure that you are using firmware version 1.30 or higher for the project.

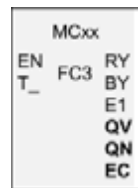
##### General

easyE4 base devices provide 32 acyclical Modbus TCP request function blocks, MC01 through MC32.

An MC function block sends exactly one acyclical request to the selected Modbus TCP server. This function block is available for all programming languages and all easyE4 base devices.



MC function blocks cannot be used within a user function block.



They are primarily used in order to request acyclical values such as temperatures or to request fixed values once when the program is starting.

##### Operating principle

An acyclical Modbus TCP request function block will send exactly one acyclical request to the selected Modbus TCP server as soon as there is a rising edge at trigger coil T\_ and function block EN=1. Function code FC3 will be set as the acyclical request by default. The data associated with the request will be read in the easyE4 base device in a defined marker range or written from there. After the data is transferred successfully, the server will respond and the RY function block output will switch to a state of 1.

The QV function block output indicates the number of elements transferred.

The following function applies to FC23:

- The QV function block output indicates the number of elements read.
- The QV function block output indicates the number of elements written. QN remains equal to 0 for other function codes.

Like with cyclical data communication, a response time can be defined here as well. As soon as the server does not respond within the specified time, function block output E1 will be set to a state of 1. Whether the registers are cleared when there is a timeout will depend on the setting configured with the option of the same name under *Project view / Cyclical data tab* when the Modbus TCP server module is selected previously – please refer to → "Cyclical data tab", page 837 as well.

If an MC function block is used in the program without Modbus modules having been configured first, the plausibility check will report errors.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
T_	Trigger input When there is a rising edge at T_, the request is sent with the function code to the Modbus TCP server.	

##### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
<b>(bit)</b>		
RY	1: Request sent and positive response received from server. 0: Request sent, but Modbus exceptions received as response from server.	
BY	BUSY 1: Waiting for server response. 0: The request has been completed.	
E1	ERROR 1: In the event of a rejection by the server or a formal error.	
<b>(DWord)</b>		
QV	Actual number of elements	Integer value range: FC1, FC2, FC5, FC15: 0...+2000 FC3, FC4, FC6, FC16, FC23: 0...+125
QN	Only relevant to function code FC23: Actual number of elements for 2nd request	Integer value range: 0...+125
EC	Fault Code	

#### 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

6. Function blocks

6.1 Manufacturer function blocks

Parameter set		
	Description	Note
Parameter set		
<input checked="" type="checkbox"/> 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. 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 + Call enabled	Constants can be edited on the device, as can function block parameters when using the EDP programming language.	
Simulation not possible		

Acyclical Modbus TCP request – Parameters tab

The Acyclical Modbus TCP request – Parameters tab is used to configure the same communication parameters as for the cyclical data in general – please refer to → "Expansion parameter tab", page 835 .

The MC function block will send its acyclical Modbus client request to the selected Modbus TCP server module. The selected function code defines whether the operation will be a read or write operation, whether one or more elements are involved, and whether the elements are of data type BIT or WORD. The function block will be run for the number of elements. It will write or read the easyE4 marker range, starting with word markers, to/from the server's Modbus TCP map, beginning with the element at index 1.

Modbus TCP acyclic request - Parameter

MC: 1 Comment:

☒ Function block release by EN is necessary

Parameter display  
+ Call enabled

Specific parameterization of the Modbus TCP acyclic request

Function code: FC3 - Read Multiple Holding Register Modbus TCP server: MS1 Unit ID: 255 Response timeout: 3000 ms

☐ 32 bit mode

1st request

Start address: 0 Number of items: 1 Marker assignment: MW01

Fig. 238: Acyclical Modbus TCP request - Parameters tab

Function code

The following function codes can be selected. FC3 will be set as the default.

## 6. Function blocks

### 6.1 Manufacturer function blocks

FC <sub>dec</sub>	Function description		Function Code <sub>hex</sub>
FC1	Read Coils	Used to read outputs	0x01
FC2	Read Discrete Inputs	Used to read inputs	0x02
FC3	Read Multiple Holding Registers	Used to read multiple input registers	0x03
FC4	Read Input Registers	Used to read input registers	0x04
FC5 <sup>1)</sup>	Write Single Coil	Used to write to exactly one output	0x05
FC6	Write Single Holding Register	Used to write to a single output register	0x06
FC15 <sup>1)</sup>	Write Multiple Coils	Used to write to multiple outputs	0x15
FC16	Write Multiple Holding Registers	Used to write to multiple output registers	0x10
FC23 <sup>1)</sup>	Read and Write Multiple Holding Registers	Used to read or write from/to multiple word output registers	0x17

1) Only available for Modbus TCP clients or Modbus RTU master in easyE4

#### Modbus TCP Client

#### Expansion parameter tab for Modbus TCP servers

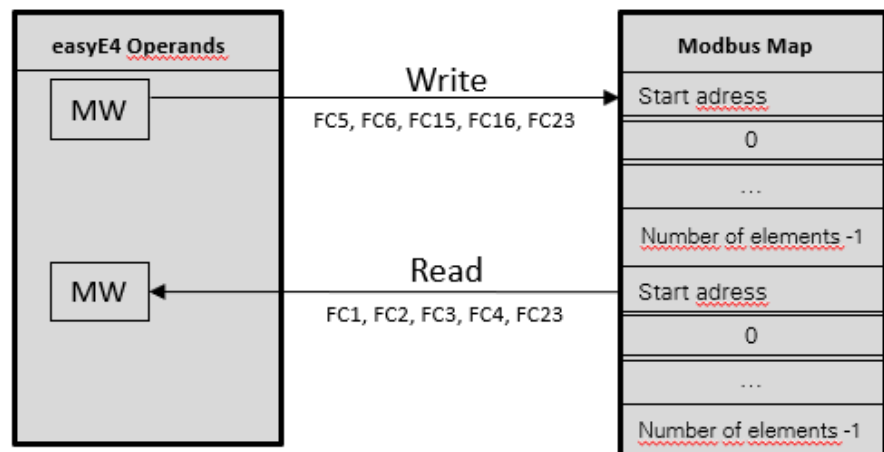


Fig. 239: Overview of how function codes are used

#### Expansion parameter tab for Modbus TCP servers

The options are MS1 through MS4. This setting is used to select the Modbus TCP server to which the request should be sent.

#### Unit ID

The value range is 1 to 255.

#### Response timeout

As soon as the server does not respond within the specified time, function block output E1 will be set to a state of 1. Whether the registers are cleared when there is a timeout will depend on the setting configured with the option of the same name

## 6. Function blocks

### 6.1 Manufacturer function blocks

under *Project view / Cyclical data tab* – please refer to → "Clear register on timeout", page 839 as well. The default value will be 3000 ms.



#### ☐ 32 bit mode

Only available on easySoft Version 7.40 or higher.

Only available on firmware version 1.40 or higher.

Otherwise this option will not be available.

Enable this option if you want register contents that are written or read with function codes FC3, FC4, FC16, FC23 to be interpreted as double words. This will cause two consecutive word registers to be combined into a double word. The number of elements per request can then be exclusively in steps of two.

This option is also important when it comes to data interpretation regarding the byte order; .

#### 1. Requirement

The parameters for the 1st request are used to define the easyE4 marker range on which the function code will be run. The range's word markers will be either written to the Modbus TCP server or read from it and stored in the easyE4's word markers.

Start address	<p>The address of the first Modbus TCP server register that should be written to or read. The value range is 0 to 65535.</p> <p>→ Keep the 0-based address system in mind. If the address range does not match the Modbus server's address range because the former starts from 0 and the latter from 1, you will need to use an offset. In this case, you will need to set the start address to the original value minus 1. Alternatively, the option <input checked="" type="checkbox"/> Auto-decrement on all addresses can be activated with a checkmark.</p>
---------------	---

No. of Elements	<p>The number of elements that will be read from the server's Modbus TCP map in the easyE4 marker range or that will be written from the easyE4 marker range to the server's Modbus TCP map. Depending on the specific function code, "element" may refer to different data formats of data type BIT or WORD.</p>
-----------------	---

Marker assignment	<p>The marker word selected in the Marker assignment field is the starting point of the marker range for which the function block will run the function codes. The elements will be written from the easyE4 marker range or read in it. The value range is 1 to 512. Make sure not to overwrite any marker range registers or sections.</p>
-------------------	---

#### 2nd write request (FC23 only)

The range for the 2nd write request will be shown exclusively for function code FC23 and must be defined under the tab.

2nd request (FC23 write)


Start address:	Number of items:	Marker assignment:
<input type="text" value="0"/>	<input type="text" value="1"/>	<input type="text" value="MW01"/>

Fig. 240: Acyclical Modbus TCP request - 2nd write request tab

6. Function blocks

6.1 Manufacturer function blocks

The parameters for the 2nd request are used to define the easyE4 marker range for which function code FC23 will be run. The range's word markers will be either written to the Modbus TCP server or read from it and stored in the easyE4's word markers.

Index 1. Element:	<div>The address of the first server Modbus TCP map register that is written to. The value range is 0 to 65535.</div> <div> Keep the 0-based address system in mind. If the address range does not match the Modbus server's address range because the former starts from 0 and the latter from 1, you will need to use an offset. In this case, you will need to set the start address to the original value minus 1. Alternatively, the option <input checked="" type="checkbox"/> Auto-decrement on all addresses can be activated with a checkmark.</div>
No. of Elements	<div>The number of elements that should be written from the easyE4 marker range to the server's Modbus TCP map.</div> <div>Depending on the specific function code, "element" may refer to various data formats.</div>
Marker assign- ment	<div>The marker word selected in the <b>Marker assignment</b> field is the starting point of the marker range for which the function block will run the function codes. The elements in the easyE4 marker range will be read.</div> <div>The value range is 1 to 512.</div> <div>Make sure not to overwrite any registers.</div>

Function block outputs

If you select the EDP programming language, the Function block outputs tab will appear as well

## 6. Function blocks

### 6.1 Manufacturer function blocks

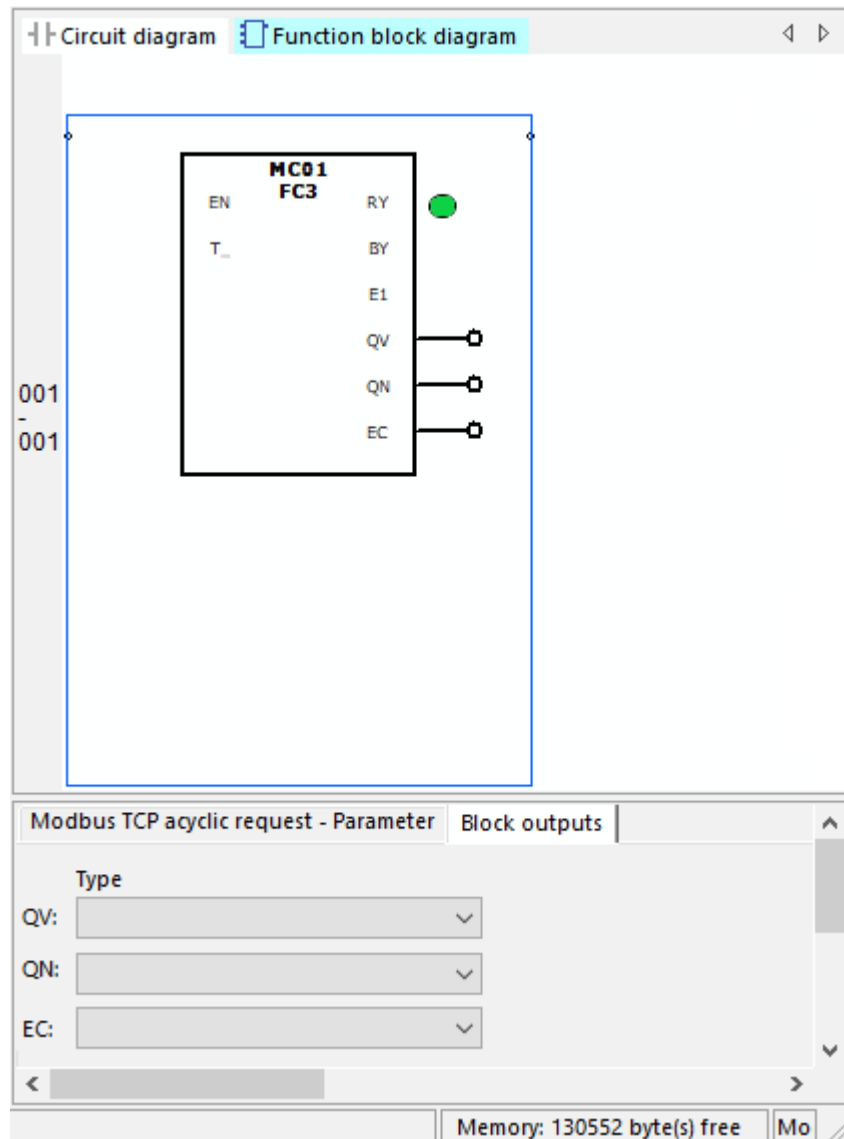


Fig. 241: Function block outputs tab

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Other

**Retention** - The function block does not recognize retentive data.

#### Signal diagram

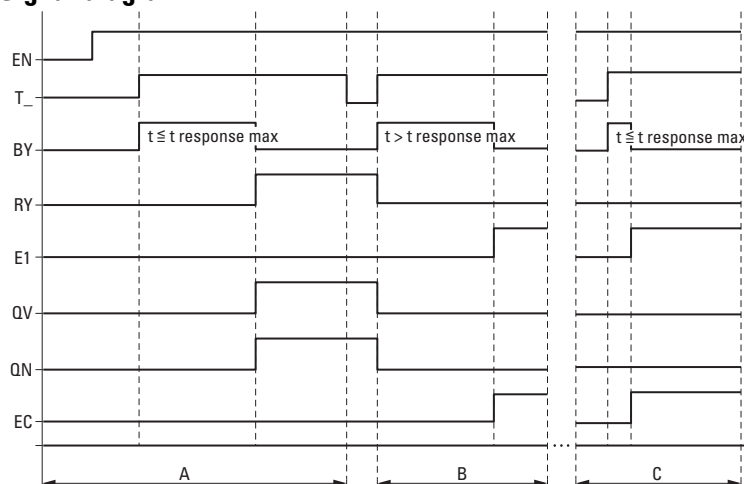


Fig. 242: Signal diagram of frequency counter

EN: Activates the function block

T\_: Trigger input; the function code is sent to the Modbus TCP server when there is a rising edge at T\_.

BY: Busy; indicates waiting for the server response and switches to a state of 0 when time  $t_{\text{response}}$  elapses.

RY: Ready; the request has been sent and the Modbus TCP client has received a response. RY=0 if EN=0

E1: Error; server rejection or formal error

QV: Actual number of elements

QN: Only for FC23: Actual number of elements in 2nd request

EC: Errorcode value

A: Normal operation; the server responds within specified time  $t_{\text{response}}$

B: Fault scenario; the server does not respond within the specified time  $t_{\text{response}}$ , potentially because the cable has been disconnected.

C: Fault scenario; the server is sending an exception code, or wrong port selected, etc.

#### Example FC23



Keep the 0-based address system in mind.

If the address range does not match the Modbus server's address range because the former starts from 0 and the latter from 1, you will need to use an offset.

In this case, you will need to set the start address to the original value minus 1.

Alternatively, the option ☒ Auto-decrement on all addresses can be activated with a checkmark.

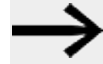
The following parameter configuration for the MC function block does the following:

#### 1. Requirement

## 6. Function blocks

### 6.1 Manufacturer function blocks

Read the server's Modbus TCP map starting from register #120 and write the content for 50 elements to the marker range starting from marker word MW10; i.e., in marker range MW10 to MW59. Elements for FC23 mean a data type of WORD.



Due to the 0-based address system, #121 must be entered in easyE4 as the 1st element index in order to read/write to the server's Modbus TCP map starting from register #120.

#### 2. Requirement

At the same time, write the content of 2 elements from the marker range starting with marker word W100 (i.e., from marker range MW100 to MW101) to the server's Modbus TCP map starting from register #200. Elements for FC23 mean a data type of WORD.



Due to the 0-based address system, #201 must be entered in easyE4 as the 1st element index in order to read/write to the server's Modbus TCP map starting from register #200.

Fig. 243: Acyclical Modbus TCP request tab

#### Example FC15



Keep the 0-based address system in mind.

If the address range does not match the Modbus server's address range because the former starts from 0 and the latter from 1, you will need to use an offset.

In this case, you will need to set the start address to the original value minus 1.

Alternatively, the option ☒ Auto-decrement on all addresses can be activated with a checkmark.

The following parameter configuration for the MC function block does the following:

#### 1. Requirement

Starting from register #21, write to the server's Modbus TCP map the content of 8 elements from the marker range starting with marker word MW10; elements for FC15 mean a data type of BIT. Write the first 8 least significant bits of MW10.

## 6. Function blocks

### 6.1 Manufacturer function blocks

➔ Due to the 0-based address system, #22 must be entered in easyE4 as the 1st element index in order to read/write to the server's Modbus TCP map starting from register #21.

Modbus TCP acyclic request - Parameter

MC: 1 Comment:

☒ Function block release by EN is necessary

Parameter display

+ Call enabled

Specific parameterization of the Modbus TCP acyclic request

Function code: FC15 - Write Multiple Coils Modbus TCP server: MS1 Unit ID: 255 Response timeout: 3000 ms

☐ 32 bit mode

1st request

Start address: 22 Number of items: 8 Marker assignment: MW10

Fig. 244: Acyclical Modbus TCP request tab

#### See also

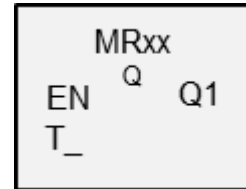
- Section "easyE4 as a Modbus TCP client", page 832
- Section "AL - Alarm function block", page 472
- Section "BV - Boolean operation", page 477
- Section "D - Text display", page 481
- Section "D - Text display editor", page 491
- Section "DL - Data logger", page 510
- Section "JC - Conditional jump", page 524
- Section "LB - Jump label", page 529
- Section "MR - Master Reset", page 543
- Section "NC - Numerical converter", page 563
- Section "ST - Set cycle time", page 569

### 6.1.7.9 MR - Master Reset

#### 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

(bit)	Description	Note
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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Operating mode

Operating mode	Description	Note
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: <ul style="list-style-type: none"> <li>• Marker range MD01...MD256</li> <li>• ND01...ND16</li> <li>• Internal markers of existing function blocks UF, IC, IE and IT</li> </ul>	
ALL = Reset both	Has an effect on the operands set at Q and M.	

#### Function block outputs

(bit)	Description	Note
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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x



## 6. Function blocks

### 6.1 Manufacturer function blocks

Assigning operands	Bit outputs
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

Parameter set	Description	Note
<input checked="" type="checkbox"/> 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 master reset function block with the programming method EDP

I 05-----Ä MR07T\_  
Fig. 245: Wiring the function block coils

The trigger coil is connected to a device input.

MR07Q1-----Ä S M42  
Fig. 246: Wiring of the function block contact

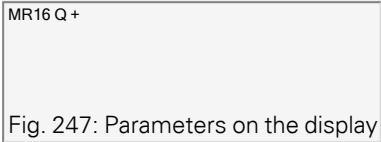
The message of the function block is sent to a marker.

6. Function blocks

6.1 Manufacturer function blocks

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.



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 472
- Section "BV - Boolean operation", page 477
- Section "D - Text display", page 481
- Section "D - Text display editor", page 491
- Section "DL - Data logger", page 510
- Section "JC - Conditional jump", page 524
- Section "LB - Jump label", page 529
- Section "MC - Acyclical Modbus TCP request", page 531
- Section "NC - Numerical converter", page 563
- Section "ST - Set cycle time", page 569

#### 6.1.7.10 MU - Acyclical Modbus RTU request

Only available on easySoft Version 7.40 or higher.

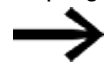
Only available on firmware version 1.40 or higher.

Otherwise this option will not be available.

##### General

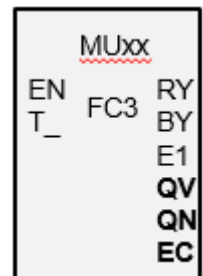
easyE4 base devices provide 32 acyclical Modbus RTU request function blocks, MU01 through MU32.

An MU function block sends exactly one acyclical request to the selected Modbus RTU slave. This function block is available for all programming languages and all easyE4 base devices.



MU function blocks cannot be used within a user function block.

They are primarily used in order to request acyclical values such as temperatures or to request fixed values once when the program is starting.



##### Operating principle

An acyclical Modbus RTU request function block will send exactly one acyclical request to the selected Modbus RTU slave as soon as there is a rising edge at trigger coil T\_ and function block EN=1. Function code FC3 will be set as the acyclical request by default. The data associated with the request will be read in the easyE4 base device in a defined marker range or written from there. After the data is transferred successfully, the slave will respond and the RY function block output will switch to a state of 1.

The QV function block output indicates the number of elements transferred.

The following function applies to FC23:

- The QV function block output indicates the number of elements read.
- The QV function block output indicates the number of elements written. QN remains equal to 0 for other function codes.

Like with cyclical data communication, a response time can be defined here as well. As soon as the slave does not respond within the specified time, function block output E1 will be set to a state of 1. Whether the registers are cleared when there is a timeout will depend on the setting configured with the option of the same name under *Project view / Cyclical data tab* when the Modbus RTU slave module is selected previously – please refer to → "Cyclical data tab", page 837 as well.

If an MU function block is used in the program without Modbus modules having been configured first, the plausibility check will report errors.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### The function block and its parameters

##### Function block inputs

	Description	Note
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
T_	Trigger input When there is a rising edge at T_, the request is sent with the function code to the Modbus TCP server.	

##### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC  
<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
<b>(bit)</b>		
RY	1: Request sent and positive response received from slave. 0: Request sent, but Modbus exceptions received as response from slave.	
BY	BUSY 1: Waiting for slave response. 0: The request has been completed.	
E1	ERROR 1: In the event of a rejection by the slave or a formal error.	
<b>(DWord)</b>		
QV	Actual number of elements	Integer value range: FC1, FC2, FC5, FC15: 0...+2000 FC3, FC4, FC6, FC16, FC23: 0...+125
QN	Only relevant to function code FC23: Actual number of elements for 2nd request	Integer value range: 0...+125
EC	Fault Code	

#### 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Parameter set

	Description	Note
<b>Parameter set</b>		
<input checked="" type="checkbox"/> 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. 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 + Call enabled	Constants can be edited on the device, as can function block parameters when using the EDP programming language.	
Simulation not possible		

#### Acyclical Modbus RTU request – Parameters tab

The Acyclical Modbus RTU request – Parameters tab is used to configure the same communication parameters as for the cyclical data in general – please refer to → "Expansion parameter tab", page 835 .

The MU function block will send its acyclical Modbus RTU request to the selected Modbus RTU slave module. The selected function code defines whether the operation will be a read or write operation, whether one or more elements are involved, and whether the elements are of data type BIT or WORD. The function block will be run for the number of elements. It will write or read the easyE4 marker range, starting with word markers, to/from the slave's Modbus RTU map, beginning with the element at index 1, see also → "Modbus RTU map", page 558

Fig. 248: Acyclical Modbus RTU request - Parameters tab

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function code

The following function codes can be selected. FC3 will be set as the default.

FC <sub>dec</sub>	Function description		Function Code <sub>hex</sub>
FC1	Read Coils	Used to read outputs	0x01
FC2	Read Discrete Inputs	Used to read inputs	0x02
FC3	Read Multiple Holding Registers	Used to read multiple input registers	0x03
FC4	Read Input Registers	Used to read input registers	0x04
FC5 <sup>1)</sup>	Write Single Coil	Used to write to exactly one output	0x05
FC6	Write Single Holding Register	Used to write to a single output register	0x06
FC15 <sup>1)</sup>	Write Multiple Coils	Used to write to multiple outputs	0x15
FC16	Write Multiple Holding Registers	Used to write to multiple output registers	0x10
FC23 <sup>1)</sup>	Read and Write Multiple Holding Registers	Used to read or write from/to multiple word output registers	0x17

1) Only available for Modbus TCP clients or Modbus RTU master in easyE4

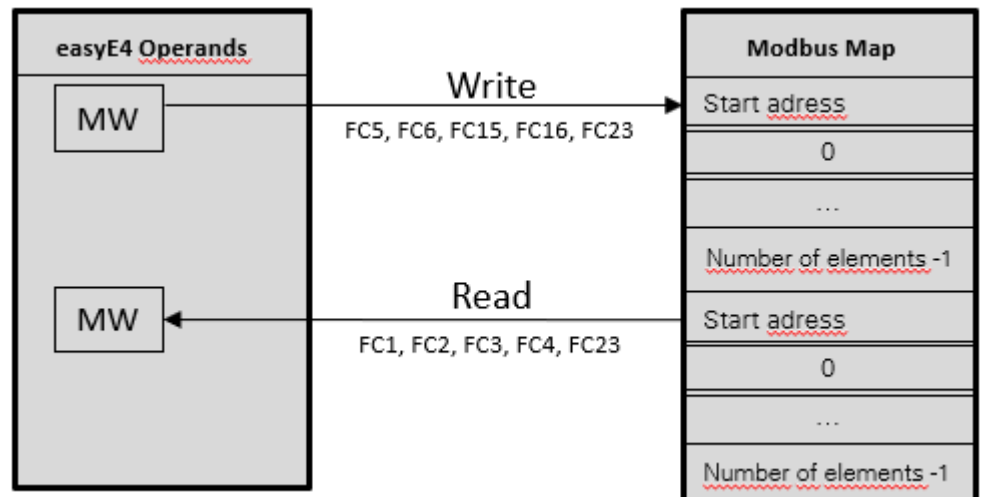


Fig. 249: Overview of how function codes are used

#### ComBUS module

C1 is preset as the Modbus RTU slave communication module to which the request should be sent.

6. Function blocks

6.1 Manufacturer function blocks

Slave ID

The value range is 0 to 255.

If a slave ID of 0 is selected, the Modbus RTU master will send the request as a broadcast to all configured Modbus RTU slaves. In this case, only function codes for write jobs can be sent (FC5, FC6, FC15, FC16). The request will be sent with default settings, i.e., with a big-endian byte order and an address offset of 1, i.e., with the Auto-decrement on all addresses option disabled.

Response timeout

As soon as the slave does not respond within the specified time, function block output E1 will be set to a state of 1. Whether the registers are cleared when there is a timeout will depend on the setting configured with the option of the same name under *Project view / Cyclical data tab* – please refer to → "Clear register on timeout", page 839 as well. The default value will be 3000 ms.

☐ 32 bit mode

Only available on easySoft Version 7.40 or higher.

Only available on firmware version 1.40 or higher.


Otherwise this option will not be available.

Enable this option if you want register contents that are written or read with function codes FC3, FC4, FC16, FC23 to be interpreted as double words. This will cause two consecutive word registers to be combined into a double word. The number of elements per request can then be exclusively in steps of two.

This option is also important when it comes to data interpretation regarding the byte order; .

1. Requirement

The parameters for the 1st request are used to define the easyE4 marker range on which the function code will be run. The range's word markers will be either written to the Modbus RTU slave or read from it and stored in the easyE4's word markers.

Start address	The address of the first Modbus RTU slave element that should be written to or read. The value range is 0 to 65535.
	<div><div>Keep the 0-based address system in mind. If the address range does not match the Modbus RTU slave's address range because the former starts from 0 and the latter from 1, you will need to use an offset. In this case, you will need to set the start address to the original value minus 1.</div></div>
No. of Elements	The number of elements that will be read from the slave's Modbus RTU map in the easyE4 marker range or that will be written from the easyE4 marker range to the slave's Modbus RTU map. Depending on the specific function code, "element" may refer to different data formats of data type BIT or WORD. If the 32-bit mode option is enabled, only a number of elements that can be divided by two will be accepted.



## 6. Function blocks

### 6.1 Manufacturer function blocks

**Marker assignment** The marker word selected in the **Marker assignment** field is the starting point of the marker range for which the function block will run the function codes. The elements will be written from the easyE4 marker range or read in it.  
The value range is 1 to 512.  
Make sure not to overwrite any marker range registers or sections.

#### 2nd write request (FC23 only)

The range for the 2nd write request will be shown exclusively for function code FC23 and must be defined under the tab.

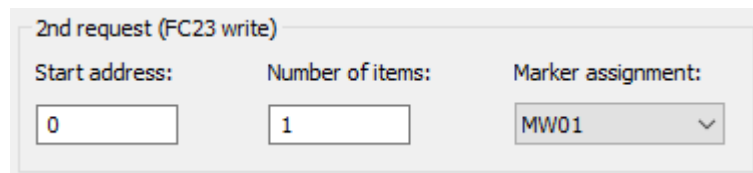


Fig. 250: Acyclical Modbus master request - 2nd write request tab

The parameters for the 2nd request are used to define the easyE4 marker range for which function code FC23 will be run. The range's word markers will be either written to the Modbus RTU slave or read from it and stored in the easyE4's word markers.

**Start address:** The address of the first slave Modbus RTU map register that is written to. The value range is 0 to 65535.



Keep the 0-based address system in mind.

If the address range does not match the Modbus RTU slave's address range because the former starts from 0 and the latter from 1, you will need to use an offset.

In this case, you will need to set the start address to the original value minus 1.

**No. of Elements** The number of elements that should be written from the easyE4 marker range to the slave's Modbus RTU map.  
Depending on the specific function code, "element" may refer to various data formats.

**Word Marker** The marker word selected in the **Word marker** field is the starting point of the marker range for which the function block will run the function codes. The elements in the easyE4 marker range will be read.  
The value range is 1 to 512.  
Make sure not to overwrite any registers.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

If you select the EDP programming language, the Function block outputs tab will appear as well

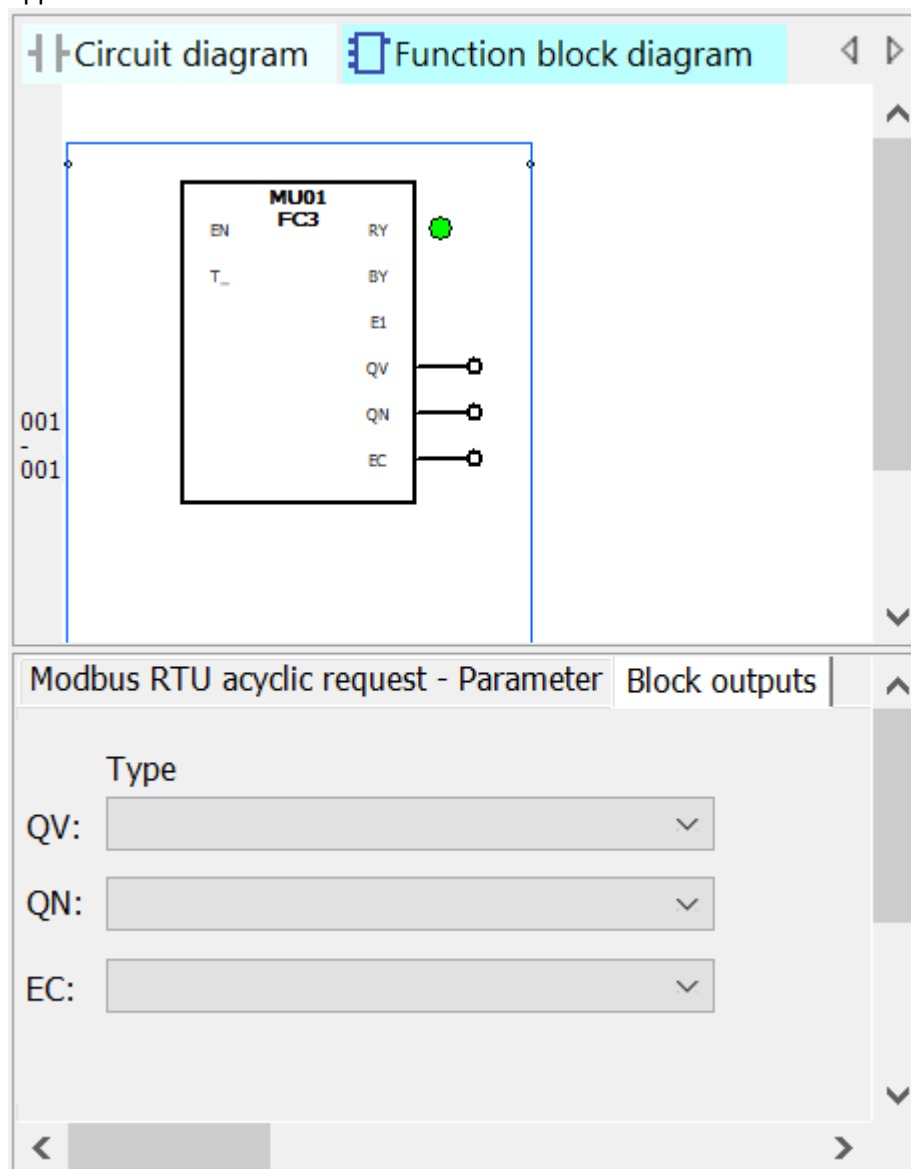


Fig. 251: Function block outputs tab

### Other

**Retention** - The function block does not recognize retentive data.

### Signal diagram

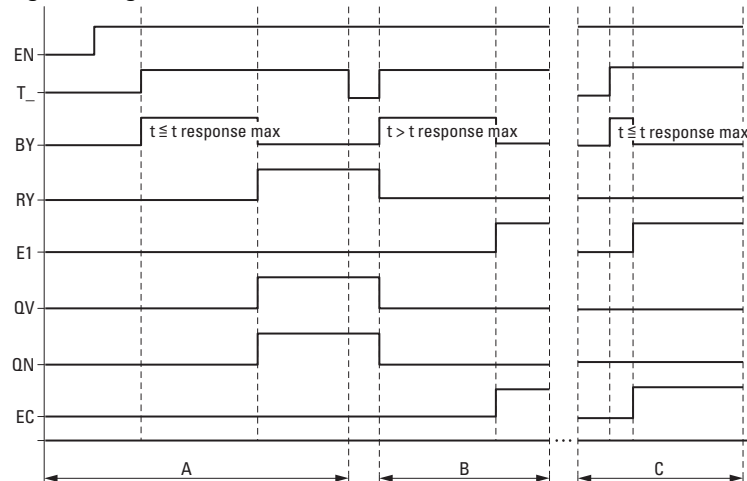


Fig. 252: Signal diagram of frequency counter

EN: Activates the function block

T\_: Trigger input; the function code is sent to the Modbus RTU slave when there is a rising edge at T\_.

BY: Busy; indicates waiting for the slave response and switches to a state of 0 when time  $t_{\text{response}}$  elapses.

RY: Ready; the request has been sent and the Modbus RTU master has received a response. RY=0 if EN=0

E1: Error; slave rejection or formal error

QV: Actual number of elements

QN: Only for FC23: Actual number of elements in 2nd request

EC: Errorcode value

A: Normal operation; the slave responds within specified time  $t_{\text{response}}$

B: Fault scenario; the slave does not respond within the specified time  $t_{\text{response}}$ , potentially because the cable has been disconnected.

C: Fault scenario; the slave is sending an exception code, or wrong port selected, etc.

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Example FC23

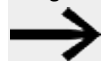


Keep the 0-based address system in mind.  
 If the address range does not match the Modbus server's address range because the former starts from 0 and the latter from 1, you will need to use an offset.  
 In this case, you will need to set the start address to the original value minus 1.  
 Alternatively, the option ☒ Auto-decrement on all addresses can be activated with a checkmark.

The following parameter configuration for the MU function block does the following:

#### 1. Requirement

Read the slave's Modbus RTU map starting from register #120 and write the content for 50 elements to the marker range starting from marker word MW10; i.e., in marker range MW10 to MW59. Elements for FC23 mean a data type of WORD.



Due to the 0-based address system, #121 must be entered in easyE4 as the 1st element index in order to read/write to the slave's Modbus RTU map starting from register #120.

#### 2. Requirement

At the same time, write the content of 2 elements from the marker range starting with marker word W100 (i.e., from marker range MW100 to MW101) to the slave's Modbus RTU map starting from register #200. Elements for FC23 mean a data type of WORD.



Due to the 0-based address system, #201 must be entered in easyE4 as the 1st element index in order to read/write to the slave's Modbus RTU map starting from register #200.

Fig. 253: Acyclical Modbus RTU request tab

#### Example FC15



Keep the 0-based address system in mind.  
If the address range does not match the Modbus server's address range because the former starts from 0 and the latter from 1, you will need to use an offset.  
In this case, you will need to set the start address to the original value minus 1.  
Alternatively, the option ☒ Auto-decrement on all addresses can be activated with a checkmark.

The following parameter configuration for the MC function block does the following:

#### 1. Requirement

Starting from register #21, write to the slave's Modbus RTU map the content of 8 elements from the marker range starting with marker word MW10; elements for FC15 mean a data type of BIT. Write the first 8 least significant bits of MW10.



Due to the 0-based address system, #22 must be entered in easyE4 as the 1st element index in order to read/write to the slave's Modbus RTU map starting from register #21.

Fig. 254: Acyclical Modbus client request tab

#### See also

- Section "AL - Alarm function block", page 472
- Section "BV - Boolean operation", page 477
- Section "D - Text display", page 481
- Section "D - Text display editor", page 491
- Section "DL - Data logger", page 510
- Section "JC - Conditional jump", page 524
- Section "LB - Jump label", page 529
- Section "MR - Master Reset ", page 543
- Section "NC - Numerical converter", page 563
- Section "ST - Set cycle time", page 569
- Section "Modbus RTU map", page 558

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Modbus RTU map

If you use a Modbus RTU communication module in slave mode for Modbus RTU communications, a Modbus RTU master will be able to access the following easyE4 base device registers with read and write operations.

Tab. 86: Modbus registers and read data for the easyE4 control relay Modbus slave

Function code Modbus	Modbus reg. #	Operand	Meaning	Remark
0x01 (Read Coil, FC1)	50001	Q1	Bit output 1	Local outputs base device
	...	...	...	
	50004	Q4	Bit output 4	
0x02 (Read Discrete Input, FC2)	50017	Q17	Expansion Bit output 17	Local outputs expansion
	...	...	...	
	50128	Q128	Expansion Bit output 128	
Max. 512 coils/discrete inputs at once				
8 coils/discrete inputs are merged into a byte	52001	I1	Bit input 1	Local inputs base device
	...	...	...	
	52008	I8	Bit input 8	
	52017	I17	Expansion bit input 17	local inputs expansion
	...	...	...	
	52128	I128	Expansion bit input 128	
	54001	ID1	Diagnostic bit 1	Diagnostics for base device
	...	...	...	
	54024	ID24	Diagnostic bit 24	
	54025	ID25	Diagnostic bit 25	Expansion diagnostics
	...	...	...	
	54096	ID96	Diagnostic bit 96	
	56001	M1	Marker bit 1	
	...	...	...	
	56512	M512	Marker bit 512	

## 6. Function blocks

### 6.1 Manufacturer function blocks

Function code Modbus	Modbus reg. #	Operand	Meaning	Remark
	58001	N1	NET marker bit 1	Only the local NET marker bits will be returned, and no marker bits from the other stations will be returned
	...	...	...	
	58512	N512	loc. NET marker bit 512	

## 6. Function blocks

### 6.1 Manufacturer function blocks

Function code Modbus	Modbus reg. #	Operand	Meaning	Remark
0x03 (Read Holding Register, FC3)	6001	QA1	32-bit analog output 1	Local analog outputs base device
	...	...	...	
	6008	QA4	32-bit analog output 4	
0x04 (Read Input Register, FC4)	6009	QA5	Expansion 32-bit analog output 5	Local analog outputs expansion
	...	...	...	
	6096	QA48	Expansion 32-bit analog output 48	
Max. 125 registers at once, 1 register = 2 byte/1- word	6501	IA1	32-Bit analog input 1	Local analog inputs base device
	...	...	...	
	6508	IA4	32-Bit analog input 4	
0x17 (Read Multiple Registers, FC23)	6509	IA5	32-Bit analog input 5	Local analog inputs expan- sion
	...	...	...	
	6596	IA48	32-Bit analog input 48	
	5000		RTC (second)	<b>RTC format</b> 5000: Seconds; 5001: Minutes; 5002: Hours; 5003: Day of the month; 5004: Month; 5005: Year
	...	...	...	
	5005		RTC (year)	
	5006		Minutes, seconds	<b>GALILEO format</b> High byte, low byte
	5007		— Hours	
	5008		Month, Day	
	5009		Year	
	7001	MW1	Marker word 1	
	...	...	...	
	7512	MW512	Marker word 512	



## 6. Function blocks

### 6.1 Manufacturer function blocks

Function code Modbus	Modbus reg. #	Operand	Meaning	Remark
	8001	NW1	loc. 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.
	....	...	...	
	8032	NW32	loc. 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.				

Tab. 87: How the Modbus registers and write data for the Modbus slave are mapped easyE4

Function code Mod- bus	Modbus reg.#	Operand	Meaning	Remark
0x05 (Write Single Coil, FC5)	56001	M1	Marker bit 1	
	...	...	...	
	56512	M512	Marker bit 512	
0x0F (Write Multiple Coils, FC15)	58001	N1	loc. NET marker bit 1	Only the local NET marker bits can be written to (marker bits for other stations cannot be written to).
	...	...	...	
	58512	N512	loc. NET marker bit 512	

## 6. Function blocks

### 6.1 Manufacturer function blocks

Function code Mod-bus	Modbus reg.#	Operand	Meaning	Remark
0x06 (Write Single Register, FC6)	5000		RTC (second)	<b>RTC format</b> 5000: Seconds; 5001: Minutes; 5002: Hours; 5003: Day of the month; 5004: Month; 5005: Year
	...	...	...	
0x10 (Write Multiple Register, FC16)	5005		RTC (year)	
	5006		Minutes, seconds	
0x17 (Write Multiple Registers, FC23)	5007		— Hours	<b>GALILEO format</b> High byte, low byte
	5008		Month Day	
	5009		Year	
	7001	MW1	Marker word 1	
	...	...	...	
	7512	MW512	Marker word 512	
				Only the local NET marker words can be written to (NET marker words for other stations cannot be written to).
	8001	NW1	loc. NET marker word 1	
	...		...	
	8032	NW32	loc. 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.



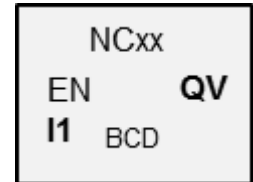
Please note that little-endian is used when converting bytes to words in an easyE4. If you want to implement Modbus communications with big-endian, you will need to make the necessary adjustments.

### 6.1.7.11 NC - Numerical converter

#### General

easyE4 base 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 versa, 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
<b>(bit)</b>		
EN	1: Activates the function block.	
<b>(DWord)</b>		
I1	Operand to be converted	Integer value range, decimal not all the way due to BCD limitation BCD: -9 999 999 ... +9 999 999 Decimal: -161 061 273 ... +161 061 273

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup> NET station n	x
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  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.	

## 6. Function blocks

### 6.1 Manufacturer function blocks

#### Function block outputs

	Description	Note
<b>(DWord)</b>		
QV	Supplies the converted value.	Integer value range Decimal: -161 061 273...+161 061 273 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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

Configuration/time range	Description	Note
<input checked="" type="checkbox"/> 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		

6. Function blocks

6.1 Manufacturer function blocks

Other

**Retention** - The function block does not recognize retentive data.

Example for BIN operating mode

For simulation purposes in easySoft 8, 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	NC	BCD	QV
(dec)	(hex)	BIN			(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 1001 1000 1001 0110 0111 1111		0000 1001 1001 1001 1001 1001 1001 1001	161 061 273
-9 999 999	FF 676 981	1111 1111 0110 0111 0110 1001 1000 0001		1111 0110 0110 0110 0110 0110 0110 0111	-161 061 273
	-10 000 000	1001 0000 0000 0000 0000 0000 0000 0000	Value range exceeded	1001 1001 1001 1001 1001 1001 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.

## 6. Function blocks

### 6.1 Manufacturer function blocks

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 8, 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		I1			QV
(dec)	(hex)	BCD		BIN	(dec)
		➔	NC	➔	
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 0000 0000 0100 1000 1001 1001		0000 0000 0000 0000 0001 0011 0010 0011	4 899
161 061 273	9 999 999	0000 1001 1001 1001 1001 1001 1001 1001		0000 0000 1001 1000 1001 0110 0111 1111	9 999 999
-161 061 273	F6 666 667	1111 0110 1001 1001 1001 1001 1001 1001		1111 1111 0110 0111 0110 1001 1000 0001	-9 999 999
161 061 274		1001 1001 1001 1001 1001 1001 1001 1001	Value range exceeded	1001 1001 1001 1001 1001 1001 1001 1001	9 999 999

➔ The most significant nibble determines the sign. For negative numbers, the two's complement will be calculated.

## 6. Function blocks

### 6.1 Manufacturer function blocks



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 give 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.

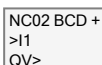
#### Example of a numerical converter function block when using the EDP programming language

Function block input NC..EN is connected directly to device terminal I5



I 05-----Ä NC01EN

Fig. 255: Wiring the function block coils



NC02 BCD +  
>I1  
QV>

Fig. 256: Setting of the parameters

#### See also

- Section "AL - Alarm function block", page 472
- Section "BV - Boolean operation", page 477
- Section "D - Text display", page 481
- Section "D - Text display editor", page 491
- Section "DL - Data logger", page 510
- Section "JC - Conditional jump", page 524
- Section "LB - Jump label", page 529
- Section "MC - Acyclical Modbus TCP request", page 531
- Section "MR - Master Reset", page 543
- Section "ST - Set cycle time", page 569



### 6.1.7.12 ST - Set cycle time

#### General

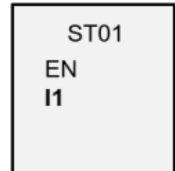
easyE4 base devices provide exactly one set cycle time function block ST01.

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
<b>(bit)</b>		
EN	1: Activates the function block.	The checkbox for the <input checked="" type="checkbox"/> Function block release by EN is necessary parameter must first be enabled
<b>(DWord)</b>		
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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x

## 6. Function blocks

### 6.1 Manufacturer function blocks

Operands	Bit inputs
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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 472
- Section "BV - Boolean operation", page 477
- Section "D - Text display", page 481
- Section "D - Text display editor", page 491
- Section "DL - Data logger", page 510
- Section "JC - Conditional jump", page 524
- Section "LB - Jump label", page 529
- Section "MC - Acyclical Modbus TCP request", page 531
- Section "MR - Master Reset ", page 543
- Section "NC - Numerical converter", page 563

6. Function blocks  
6.2 Interrupt function blocks

6.2 Interrupt function blocks

6.2.1 IC - Counter-controlled interrupt

Only possible with easySoft 8.

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.

ICxx	
C_:I1	D_:I2
EN	Q1
RE	Q2
I1	Q3
I2	Q4
I3	QV
I4	
SV	

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\_Q4 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.

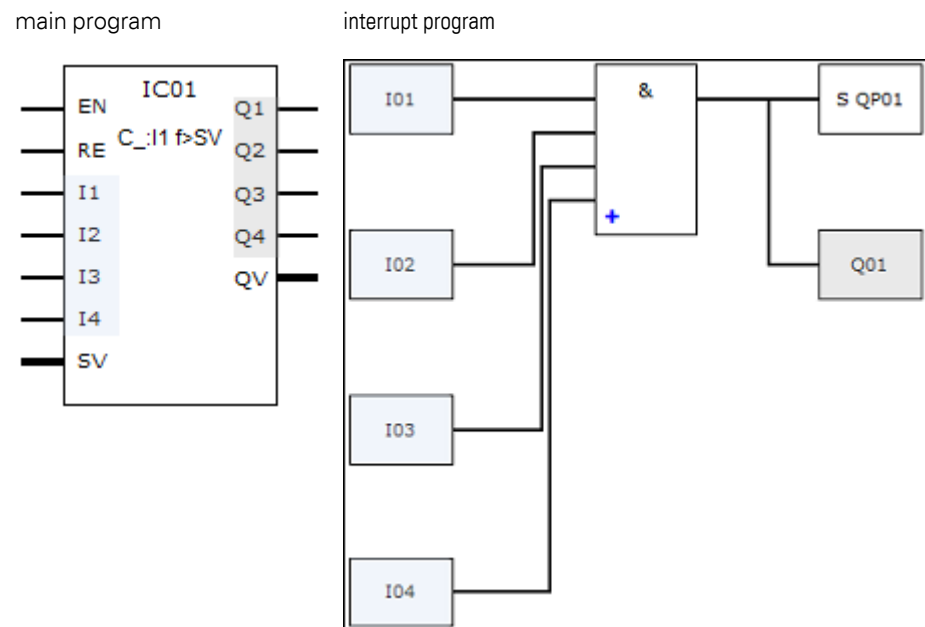


Fig. 257: Input and output states being passed between the main program and interrupt program

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.

## 6. Function blocks

### 6.2 Interrupt function blocks

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.

Function	LD	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
<b>(bit)</b>		
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		
<b>(DWord)</b>		
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.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

## 6. Function blocks

### 6.2 Interrupt function blocks

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Operating modes

(bit)	Description	Note
Counter with external direction control	<p>Pulse at device input I1 – I8, which is defined as a counter input in the parameters.</p> <p>Duration signal at device input I1 – I8, which specifies the counting direction. 0: up counting 1: down counting</p>	Maximum frequency 5kHz
Counter with 2 counter inputs	<p>Pulse at device input I1 – I8 counts up.</p> <p>Pulse at device input I1 – I8 counts down.</p>	
incremental counter	<p>Double evaluation</p> <p>With automatic up/down counting direction detection, two counter inputs</p> <p>I1...I8, counter input channel A, pulse</p> <p>I1...I8, counter input channel B, pulse</p> <p>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.</p>	
Frequency counter; $f > SV$	<p>I1...I8, timeout of the frequency reference</p> <p>Measuring interval 0.01s, 500 Hz - 5000 Hz</p>	

## 6. Function blocks

### 6.2 Interrupt function blocks

(bit)	Description	Note
	Measuring interval 0.1 s, 50 Hz - 5000 Hz Measuring interval 1.0 s 5 Hz - 5000 Hz	
Frequency counter; $f < SV$	I1 – I8, Frequency reference fallen below Measuring interval 0.01s, 500 Hz - 5000 Hz Measuring interval 0.1 s, 50 Hz - 5000 Hz 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: -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 <sup>2)</sup>	x
NET station n	



## 6. Function blocks

### 6.2 Interrupt function blocks

Assigning operands	Value outputs
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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.	
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.

#### 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 311, → "CH - High-speed counter", page 317, → "CI - Incremental Counter", page 323 as well.

For function blocks IE01 – IE08 and IC01 – IC08, device inputs IO1 – IO8 can be assigned freely.

## 6. Function blocks

### 6.2 Interrupt function blocks

For function blocks IT01 – IT08, an interrupt that is not yet in use is assigned in easySoft 8. 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 8. The maximum number of 8 interrupts is also taken into account within this context.

	Device inputs							
	I01	I02	I03	I04	I05	I06	I07	I08
<b>Interrupt source</b>								
CF01	x							
CF02		x						
CF03			x					
CF04				x				
CH01	x							
CH02		x						
CH03			x					
CH04				x				
CI01	x	x						
CI02			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.

## 6. Function blocks

### 6.2 Interrupt function blocks

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 8

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.

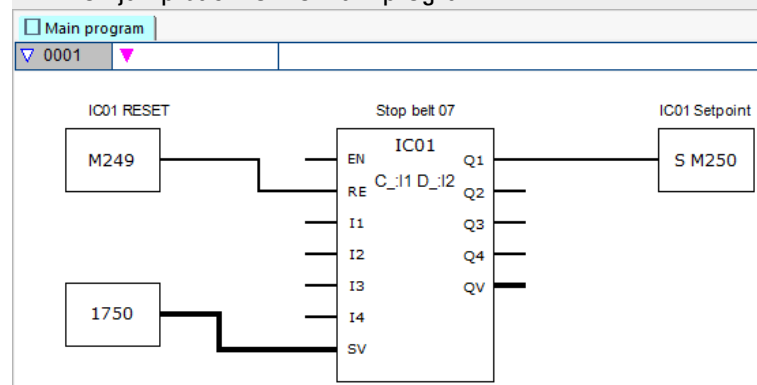


Fig. 258: easySoft 8 Main program Pulse counter with external direction

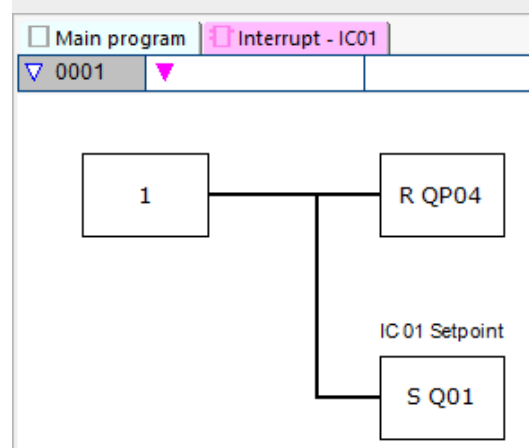


Fig. 259: easySoft 8 Interrupt program Pulse counter with external direction

6. Function blocks

6.2 Interrupt function blocks

Example with two counter inputs in easySoft 8

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.

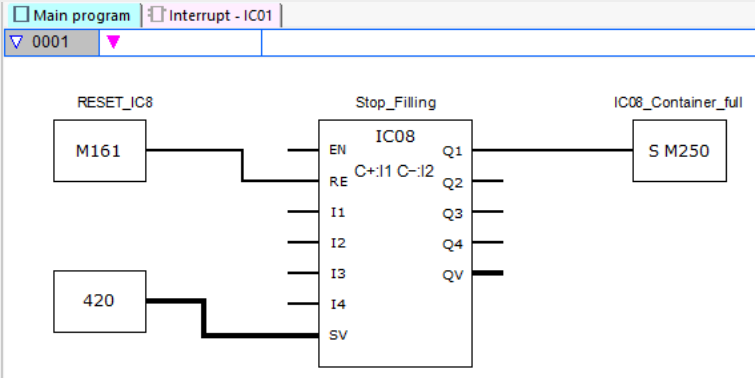


Fig. 260: easySoft 8 Main program, two counter inputs

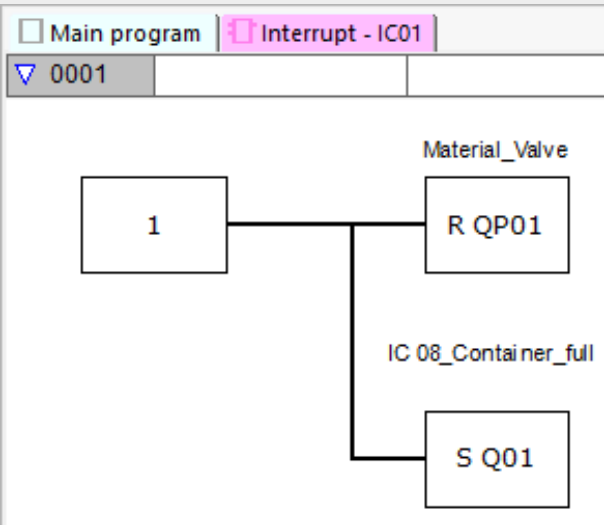


Fig. 261: easySoft 8 Interrupt program, two counter inputs

#### Example of an incremental counter in easySoft 8

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.

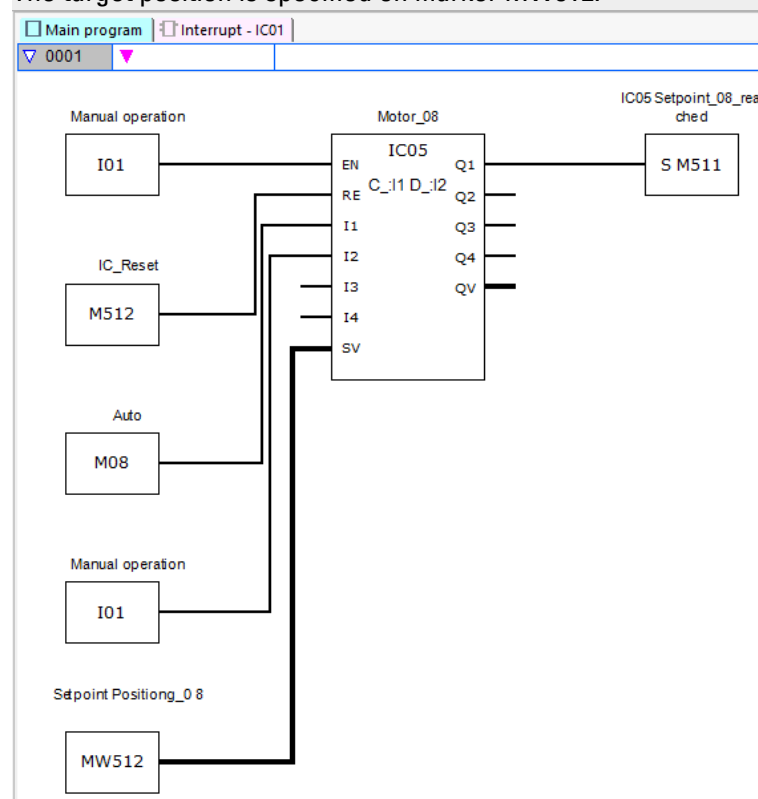


Fig. 262: easySoft 8 Main program Incremental counter

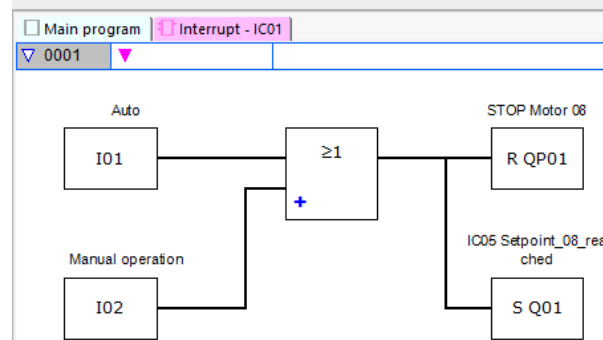


Fig. 263: easySoft 8 Interrupt program Incremental counter

## 6. Function blocks

### 6.2 Interrupt function blocks

#### Example frequency measurement in easySoft 8

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.

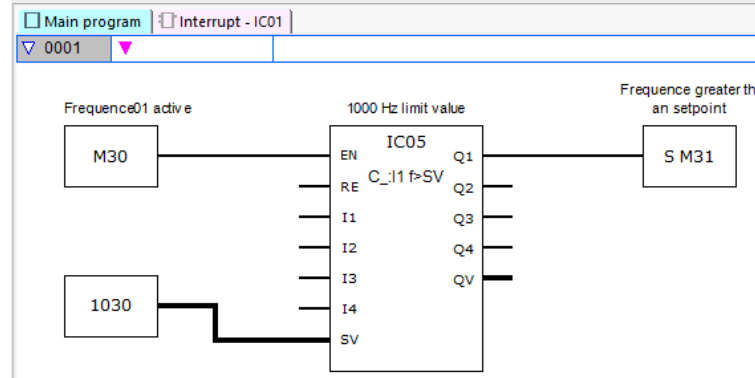


Fig. 264: easySoft 8 Main program Frequency measurement

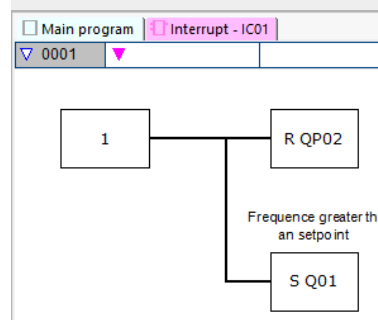


Fig. 265: easySoft 8 Interrupt program Frequency measurement

#### See also

- Section "IE - Edge-controlled interrupt", page 583
- Section "IT - Time-controlled interrupt function block", page 589

## 6.2.2 IE - Edge-controlled interrupt

Only possible with easySoft 8.

### 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 edge, falling edge, 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.

IExx P:I1	
EN	Q1
RE	Q2
I1	Q3
I2	Q4
I3	QV
I4	
TD	



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. Function blocks

### 6.2 Interrupt function blocks

#### 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\_Q4 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.

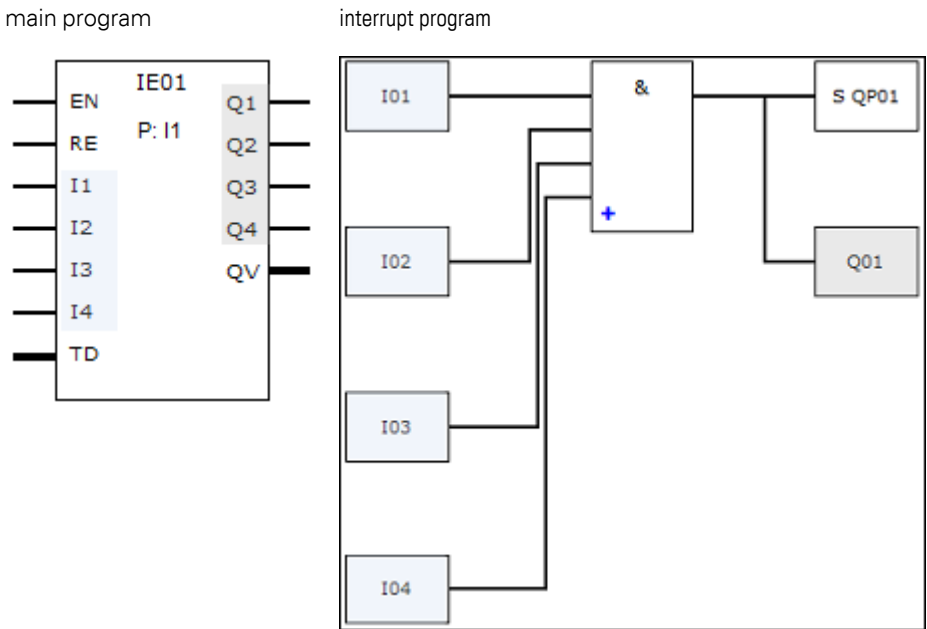


Fig. 266: Input and output states being passed between the main program and interrupt program

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.



## 6. Function blocks

### 6.2 Interrupt function blocks

Function	LD	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 <input checked="" type="checkbox"/> Function block release by EN is necessary 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 Resolution: 10 ms

##### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

## 6. Function blocks

### 6.2 Interrupt function blocks

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> 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

<sup>2)</sup> Only on projects with  $\geq 2$  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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

## 6. Function blocks

### 6.2 Interrupt function blocks

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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
<input checked="" type="checkbox"/> 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.

#### Example slope in easySoft 8

##### Rising edge operating mode

Cutting device at station 2. The interrupt is triggered with a rising edge at device input I1. In the interrupt program, device output Q01 is set as per function block inputs I1 and I2 (which can be seen on QP01) and the goods are cut. Device output Q02 is cleared (which can be seen on QP02). Function block output Q1 gets the result of the AND operator.

## 6. Function blocks

### 6.2 Interrupt function blocks

In the main program, markers M512 and M42 are provided to the function block inputs of function block IE for the next interrupt. The result of the last AND operator is provided in marker 211.

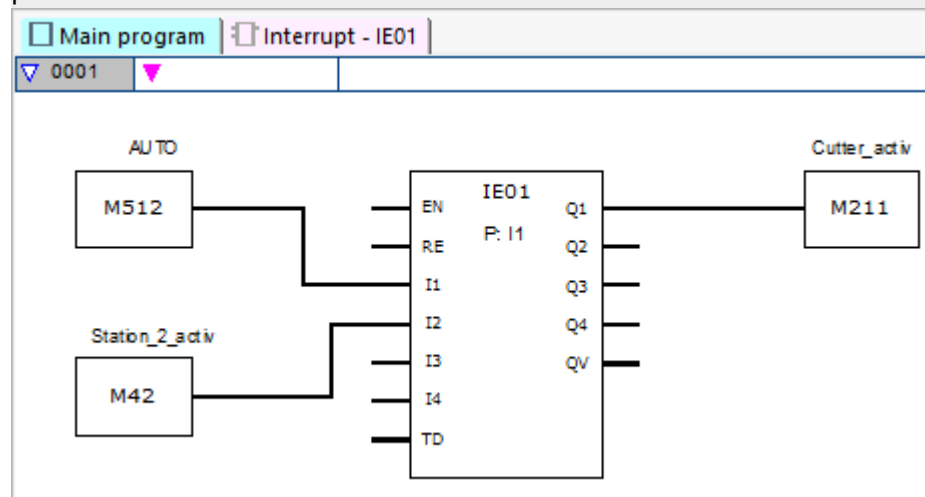


Fig. 267: easySoft 8 Main program Slope

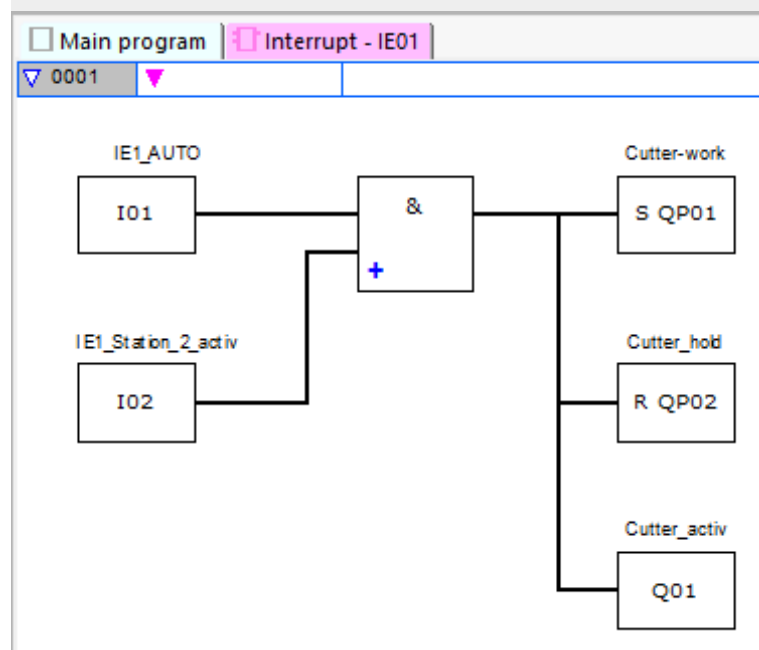


Fig. 268: easySoft 8 Interrupt program Slope

#### See also

- Section "IC - Counter-controlled interrupt", page 572
- Section "IT - Time-controlled interrupt function block", page 589

### 6.2.3 IT - Time-controlled interrupt function block

Only possible with easySoft 8.

#### 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.

ITxx	
X	
EN	Q1
RE	Q2
I1	Q3
I2	Q4
I3	QV
I4	
PD	

#### 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\_Q4 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.

6. Function blocks

6.2 Interrupt function blocks

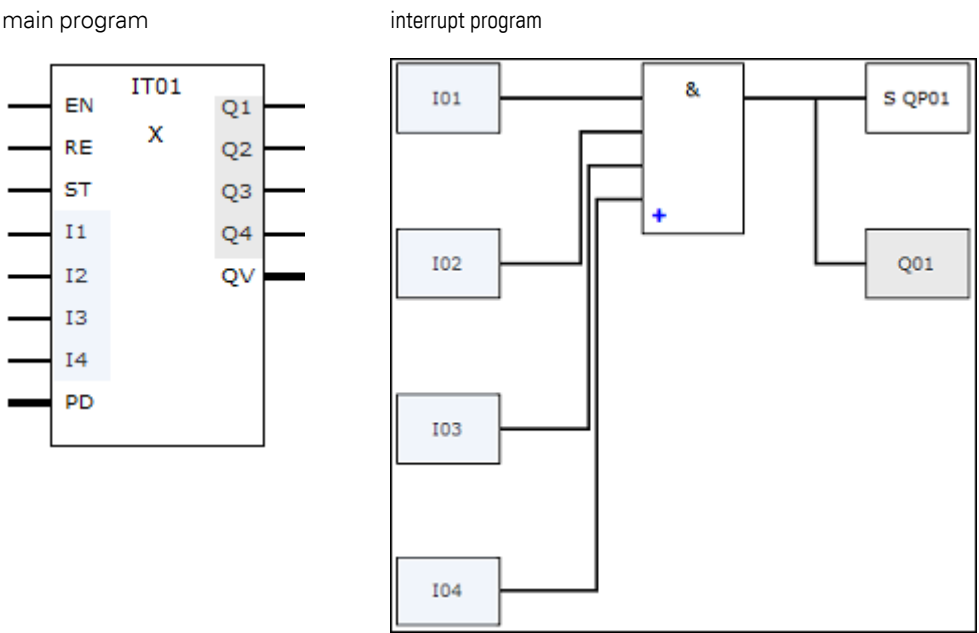


Fig. 269: Input and output states being passed between the main program and interrupt program

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.

Function	LD	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
<b>(bit)</b>		
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. 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		
<b>(DWord)</b>		
PD	Pulse pause time: Value of the delay that must elapse before the interrupt program is started.	Integer value range: 20...999 990 ms, resolution 10 ms

#### Assigning operands

You can assign the following operands to the function block inputs that are numeric inputs.

Operands	Value inputs
Constant, timer constant <sup>1)</sup>	x
MD, MW, MB - Markers	x
NB, NW, ND - NET markers <sup>2)</sup>	x
nNB, nND, nND- NET markers <sup>2)</sup>	x
NET station n	
IA - Analog input	x
QA - Analog output	x
QV - QV - Numeric output of a FB	x

<sup>1)</sup> Only on function blocks T, AC

<sup>2)</sup> Only on projects with ≥ 2 base devices on NET

You can assign the following operands to the function block inputs that are bit inputs:

Operands	Bit inputs
Constant 0, constant 1	x
M – Markers	x
RN - Input bit via NET <sup>2)</sup>	x
SN - Output bit via NET (send) <sup>2)</sup>	x
N - Net marker bit <sup>2)</sup>	x
nN - NET marker bit <sup>2)</sup> NET station n	x
ID: Diagnostic alarm	x

## 6. Function blocks

### 6.2 Interrupt function blocks

Operands	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

<sup>2)</sup> Only on projects with  $\geq 2$  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		
(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 <sup>2)</sup>	x
NET station n	
QA – Analog output	x
I – Value input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  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) <sup>2)</sup> SN - Output bit via NET (send)	x
N – Network marker bit <sup>2)</sup>	x
LE – Output backlight	x
Q – Bit output	x
I – Bit input of a FB	x

<sup>2)</sup> Only on projects with  $\geq 2$  base devices on NET

#### Parameter set

	Description	Note
Parameter display (+ 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		

## 6. Function blocks

### 6.2 Interrupt function blocks

#### 6.2.3.4 Other

**Retention** - The function block does not recognize 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 311, → "CH - High-speed counter", page 317, → "CI - Incremental Counter", page 323 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 8. 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 8. The maximum number of 8 interrupts is also taken into account within this context.

	Device inputs							
	I01	I02	I03	I04	I05	I06	I07	I08
<b>Interrupt source</b>								
CF01	x							
CF02		x						
CF03			x					
CF04				x				
CH01	x							
CH02		x						
CH03			x					
CH04				x				
CI01	x	x						
CI02			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)							

## 6. Function blocks

### 6.2 Interrupt function blocks

	Device inputs							
	I01	I02	I03	I04	I05	I06	I07	I08
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  $\mu$ s. 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 8

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.

## 6. Function blocks

### 6.2 Interrupt function blocks

Operating mode: On-delayed

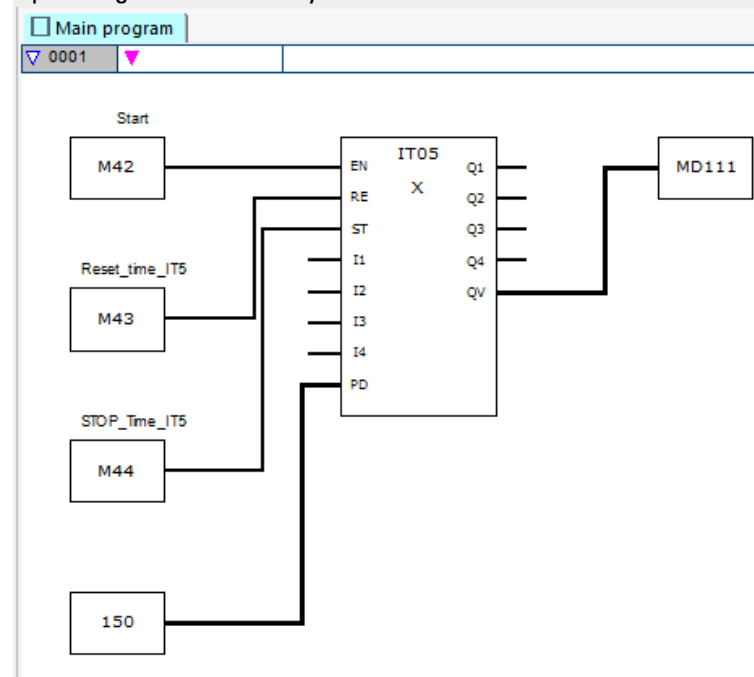


Fig. 270: easySoft 8 Main program Time-controlled

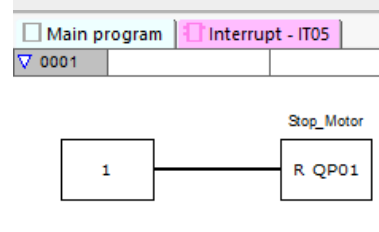


Fig. 271: easySoft 8 Interrupt program Time-controlled

#### See also

- Section "IC - Counter-controlled interrupt", page 572
- Section "IE - Edge-controlled interrupt", page 583

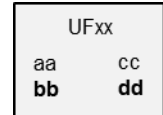
## 6.3 UF - User function block

Only possible with easySoft 8.

### 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 must not be exceeded, and that this total includes the retentive markers from the main program and the retentive markers from all user function block instances. The total number of retentive markers depends on the firmware version – please refer to → "Retention section", page 605. 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.

The maximum number of user function blocks that can be called in a single main program is 128.

## 6. Function blocks

### 6.3 UF - User function block

#### 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.

The following applies to easySoft 7:

One and only one easySoft 7 project with user function blocks can be open at any one time. Opening additional easySoft 7 projects with user function blocks is not possible.

Only available on easySoft Version 8.00 or higher.

Multiple easySoft 8 projects with user function blocks can be open simultaneously.

#### 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.

- In the programming view, 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

## 6. Function blocks

### 6.3 UF - User function block

Program/ Create user function block... menu option

Create user function block

Name  Version  .  !

For easyE4 base device From firmware version

Programming method

Interface

Bit inputs:  Bit outputs:

Value inputs:  Value outputs:

☐ Interface locked

Know-how protection

Password

Repeat password

☐ Show password

Retention

MB  -  DB  -

C  -  T  -

Sum retention bytes 0

Comment

OK Cancel

Fig. 272: 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.

Only available on easySoft Version 8.00 or higher.

## 6. Function blocks

### 6.3 UF - User function block

The user function block can then be found in *Programming view/list of operands and function blocks/User function blocks/Project*. It will be stored together with the project (user function blocks in this directory are not saved at the file level).

The following applies to easySoft 7:

The user function block can then be found in *Programming view/catalog/User function blocks/*. All user blocks from this directory are automatically saved at file level in the `\ProgramData\Eaton\easySoft 7\UserFBs` directory.

#### 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.

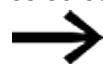
- Clicking on **OK** to close the "Create user function block" dialog box will create and save the new user function block.

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 – Light timer V1.10>`.

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.

#### Firmware version or higher

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.



## 6. Function blocks

### 6.3 UF - User function block

The drop-down list shows (and is used to select) the firmware version for which the created user function block will be used later on. It will then be possible to use the user function block for easyE4 devices with the selected firmware version or higher.

If the currently selected easyE4 device has firmware version V2.30, then V2.30 will be selected by default in the drop-down menu.



If you will be using the user function block for easyE4 devices with lower firmware versions as well, then select the corresponding firmware version in the drop-down menu.

Please note that you will only be able to do that here in the Create user function block dialog box. Once you close the dialog box, you will not be able to change the firmware version to a lower version anymore.

If the currently selected easyE4 device has a lower firmware version, the drop-down menu will only contain the firmware versions that are supported.

## 6. Function blocks

### 6.3 UF - User function block

#### 6.3.3 Configuring a user function bl

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. If you want the changes to be saved beyond runtime, the user function block will need to be saved with the *Program/Save user function block* menu option or with the *UFxx/Save* context menu option.

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.

## 6. Function blocks

### 6.3 UF - User function block

Menu option Program/ Configure user function block

Configure user function block

Name:

Version:  .

For easyE4 base device:

Programming method:

Interface

Bit inputs:  Bit outputs:

Value inputs:  Value outputs:

☐ Interface locked

Know-how protection

Password:

Repeat password:

☐ Show password

Retention

MB:  -  DB:  -

C:  -  T:  -

Sum retention bytes: 0

Comment:

Fig. 273: Configure user function block

#### Firmware version or higher

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.



If the user function block is already being used in the .e80 project, the oldest firmware version will be shown. It will not be possible to select a new firmware version in this case, since the device using the user function block will not support it.

#### 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 input-/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.

## 6. Function blocks

### 6.3 UF - User function block

#### **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 8 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.

Know-how protection will be activated as soon as the user function block is saved in the Programming view and the project is closed. Otherwise, the system will assume that programming is not done yet and that you want to be able to open and edit various UFs without locking them.

Only available on easySoft Version 8.00 or higher.

Know-how protection will apply during simulations as well.

Know-how protection will be lifted if you unlock the user function block with the corresponding password once in the open project. This is intended to make it possible to monitor values from various user function blocks during simulations in the work pane and in the operand pane without having to unlock them every single time.

#### 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

Fig. 274: Project view, System settings tab with Retention section

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 ...1024
- MW : 1...512
- MD : 1...256

The values from the input field will be automatically converted to MB marker bytes.



Marker ranges up to MB1024 can therefore be defined as retentive, since e.g. MD265 corresponds to a marker byte range of 1021-1024 and the retentive marker areas are only stored in MB.

Only available on easySoft Version 8.00 or higher.

If marker bytes are entered in the input field, these are also converted into the highest possible data type, provided the number of marker bytes allows this. The converted data type is displayed after a new switch to the System Settings tab.

## 6. Function blocks

### 6.3 UF - User function block

Fig. 275: Retention section: Marker bytes 1 - 32 entered and display in marker double words after another change to the System settings tab

#### Retention bytes

The entire retentive marker range for an easyE4 must not exceed a specific number of bytes. Depending on the firmware installed on the basic device, the following number of available bytes applies here:

- Firmware  $\geq 2.30$ : 1024 Bytes
- Firmware  $\geq 2.00$ : 512 Bytes
- Firmware  $< 2.00$ : 400 Bytes

The sum of the retentive markers of the main program and the retentive markers of all user block instances (UF) is displayed in the Project view under the System Settings tab. If the retentive marker range exceeds the number of available 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 in the Communication view with the *Communication view/Program/Configuration/Delete device* button
- By any change of the retentive range in the Project view with *Project view/System Settings tab/Retention*.
- By any changes to the parameters for the remote markers of a visualization device.
- When deleting the device from the work pane in 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. Function blocks

### 6.3 UF - User function block

## 6. Function blocks

### 6.3 UF - 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.

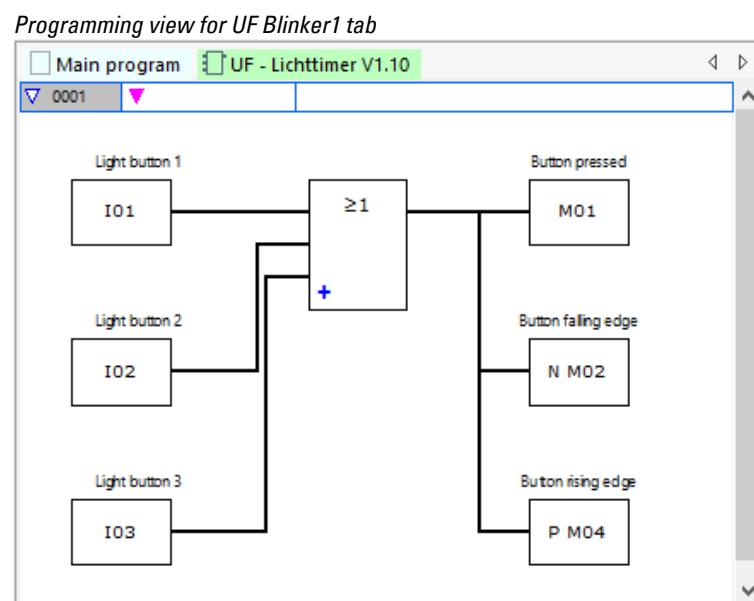


Fig. 276: Programming view for UF Blinker1 user function block

- ▶ 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.



## 6. Function blocks

### 6.3 UF - User function block

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	Tabs
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.

## 6. Function blocks

### 6.3 UF - User function block

#### 6.3.5 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.

You can enter your comment in: → "Configuring a user function bl", page 602.

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.

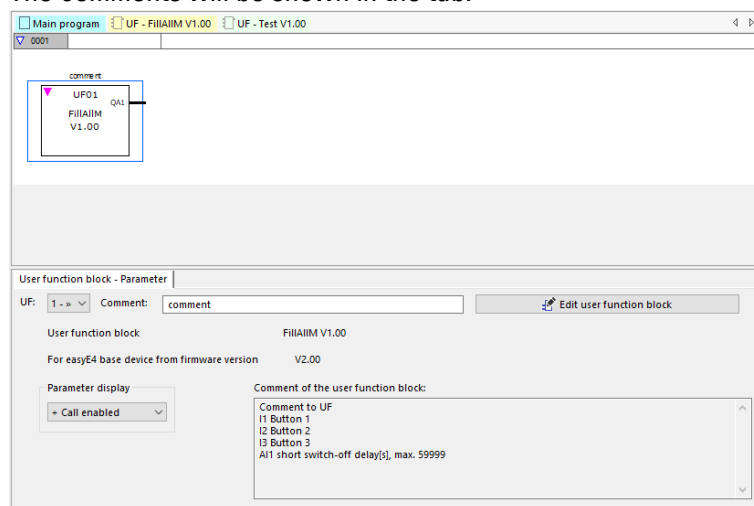


Fig. 277: User function block comments being shown in the tab

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 "Button 1" in the user function block can have a comment different from I1 "POWER ON" in the main program.

### 6.3.6 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.

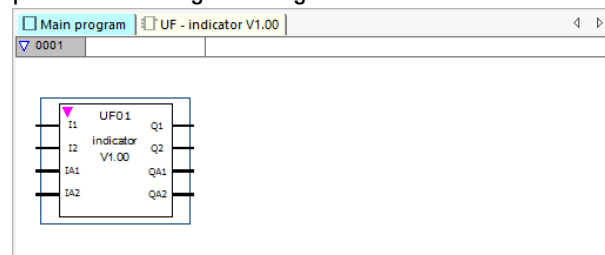


Fig. 278: UF Blinker1 user function block used in the main program

The function block will be shown with its name, version, and configured input-/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.

6. Function blocks

6.3 UF - User function block

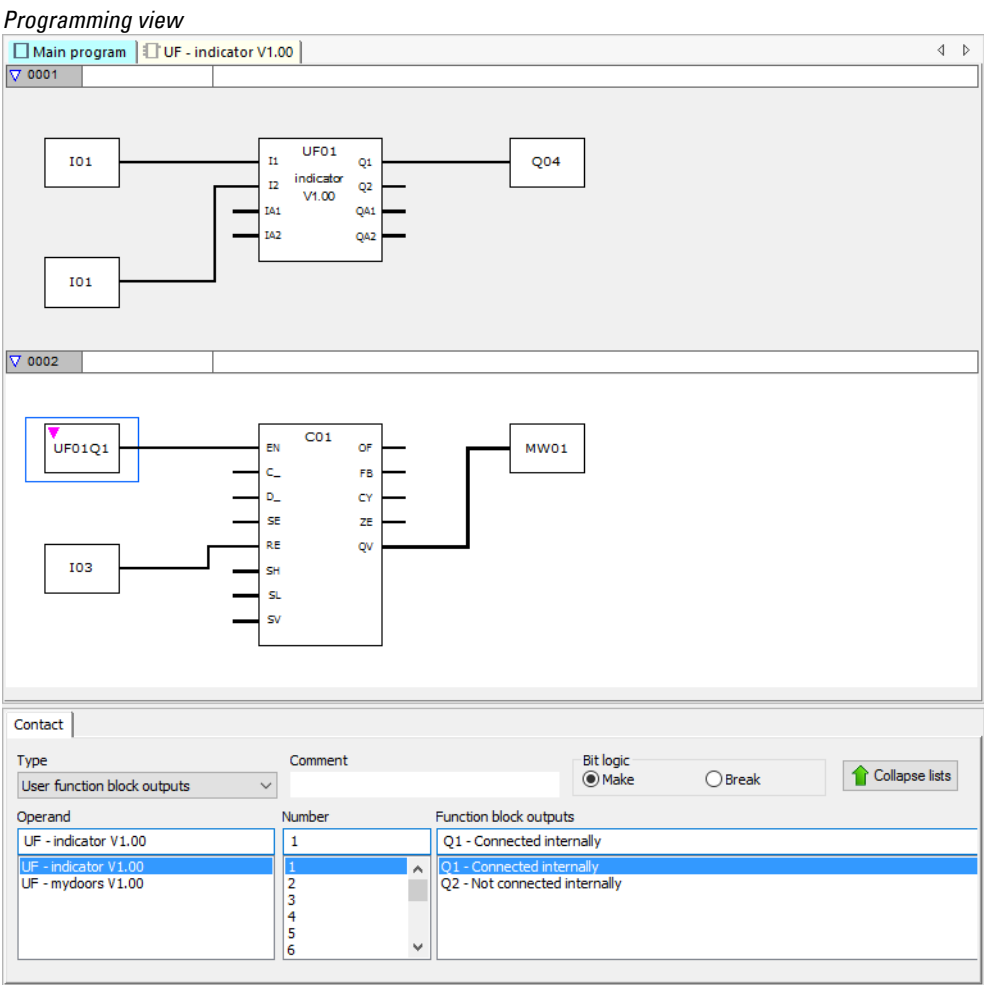


Fig. 279: Inputs/outputs wiring

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:

## 6. Function blocks

### 6.3 UF - User function block

#### Programming view

Fig. 280: Contact tab

#### Programming view

Fig. 281: 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

Fig. 282: Coil tab

6. Function blocks

6.3 UF - User function block

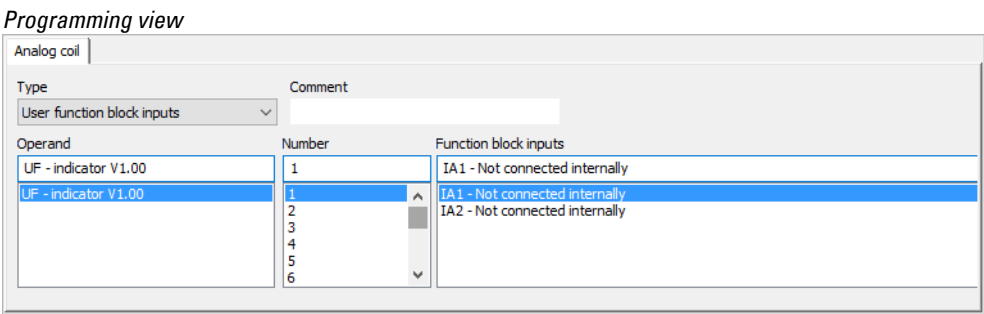


Fig. 283: Analog coil tab

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.

6.3.6.1 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.

#### UF example in ST main program

```
;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. Function blocks

### 6.3 UF - User function block

#### 6.3.7 Opening a project with an existing user function block

The following applies to easySoft 7:

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 8 and then open the project.

easySoft 8:

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 List of operands and function blocks/User function blocks/Project in easySoft 8.

When a project is opened, the user function blocks in it will not be automatically added to User function block/list of operands and function blocks/Archive, meaning that they will not be automatically available for other projects.

If you want them to be available for other projects, you will need to transfer them from the Project folder to the Archive folder. This makes it easy to avoid potential conflicts that happened with easySoft 7.



### 6.3.8 Saving a user function block

All user function blocks saved at the file level will have the same uf7 file extension regardless of the easySoft version that was used to create them.

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.

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.

The following applies to easySoft 7:

User function blocks are already stored in *Programming view/list of operands and function blocks/User function blocks/*when created.

All the user function blocks in this directory are saved as a separate uf7 file in the `\ProgramData\Eaton\easySoft 7\UserFBs` directory.

Only available on easySoft Version 8.00 or higher.

The User function blocks directory contains the Project and Archive subdirectories.

#### **Project**

User function blocks created with the *Program/Create user function block...* menu option will automatically be found in the Project directory after being created.

All the user function blocks in this directory are stored together with the project and not as a separate uf7 file at the file level.

#### **Archive**

User function blocks created once and user function blocks transferred from older versions when installing easySoft 8 will be automatically stored in the *User function block/Archive* directory.

All the user function blocks in this directory are saved as a separate uf7 file in the `\ProgramData\Eaton\easySoft 8\UserFBs` directory.

As soon as a user function block from the archive is used in the main program, it will be automatically copied to the Project directory. If you edit the user function block after that, there will be a discrepancy between the contents of the user function block from the project and the block from the archive.

6. Function blocks

6.3 UF - User function block

This discrepancy will be indicated in red. In other words, the user function block in the archive will be shown in red in the list of operands and function blocks, as will the tab for the user function block in the work pane.

Please note that function blocks shown in red in the Archive directory cannot be used in the main program!

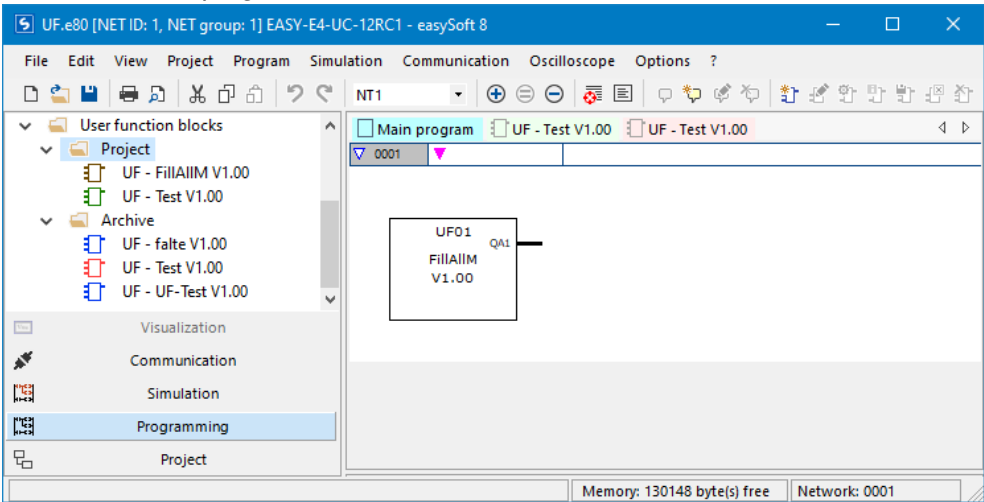


Fig. 284: easySoft 8 with function blocks in list of operands and function blocks, User function block-/Project directory and User function block-/Archive directory with UF-BETest V1.00 versions with different contents

How user function blocks with the same name but different contents are handled

Following is a simple table that shows the directory that a user function block will be added to with a specific command sequence:

Command sequence	Project	Archive
Programming view/Program/ Create user function block... menu option	✓	
Programming view/Program/ Import user function block... menu option	✓	
Programming view/list of operands and function blocks/User function blocks/Project/ Create user function block... context menu option	✓	
Programming view/list of operands and function blocks/User function blocks/Project/ Import user function block... context menu option	✓	
Programming view/list of operands and function blocks/User function blocks/Archive/ Create user function block... context menu option		✓
Programming view/list of operands and function blocks/User function blocks/Archive/ Import user function block... context menu option		✓
Communication view/Con- nection/Online/Program/Configuration/ Device => PC	✓	
easySoft 8 installation/ <input checked="" type="checkbox"/> Transfer user function blocks from easySoft 7		✓

## 6. Function blocks

### 6.3 UF - User function block

To solve a discrepancy between user function blocks with the same name in the Project and Archive directories, use one of the following methods:

1. Rename the user function block either in the Archive directory with the *Programming view/list of operands and function blocks/User function blocks/Archive/Configure...* context menu option or in the Project directory with the *Programming view/list of operands and function blocks/User function blocks/Project/Configure...* context menu option
2. Delete one of the two user function blocks. Then copy the remaining user function block  
to the Project directory with the *Programming view/list of operands and function blocks/User function blocks/Archive/Transfer to Project folder context menu option*  
or to the Archive directory with the *Programming view/list of operands and function blocks/User function blocks/Project/Transfer to Archive folder context menu option*

## 6. Function blocks

### 6.3 UF - User function block

#### 6.3.8.1 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
IA1	8
Q	12
QA	8
M	512
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.

## 6. Function blocks

### 6.3 UF - User function block

- 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. Function blocks

### 6.3 UF - User function block

#### 6.3.9 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 need 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 file 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.

##### 6.3.9.1 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.

## 6. Function blocks

### 6.3 UF - User function block

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 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 exceeds the maximum number of 12 inputs in total (bit/analog).
- FB input exceeds the maximum number of 12 outputs in total (bit/analog).
- Operand is not supported in user function blocks!
- The number of the operand falls outside of the permissible value range of UF user function blocks.

## 6. Function blocks

### 6.3 UF - User function block

#### 6.3.10 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 easySoft 8 is already found in easySoft.  
You do not need to import it. Do you want to select a different file?
- The user function block with different content is already found in easySoft 8.  
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?
- The user function block with different content is already found in easySoft 8. 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

- The user function block with different content is already found in easySoft 8. 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 either to the Project folder or Archive folder in easySoft 8.

#### Transferring user function blocks from easySoft 7 to easySoft 8

Only available on easySoft Version 8.00 or higher.

You have the option of transferring the user function blocks from easySoft 7 when installing easySoft 8. To do this, you will need to enable the

☒ Transfer user function blocks from easySoft 7 option.

At the end of the installation process, the \*.uf7 files in the



"C:\ProgramData\Eaton\easySoft 7\UserFBs"

folder will be copied to the "C:\ProgramData\Eaton\easySoft 8\UserFBs" folder.

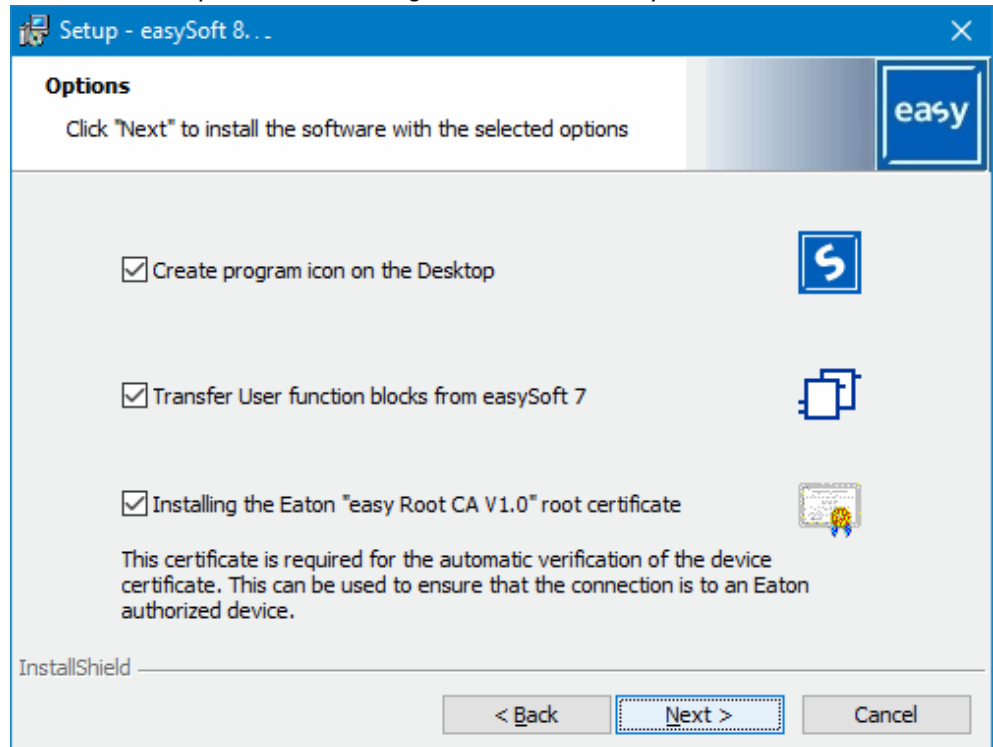


Fig. 285: easySoft 8 installation wizard

If there is a file with the same name ("**ABC.uf7**", for example) in the target directory already, the file will not be overwritten. This can occur if you have installed easySoft 8 at a previous point in time and transferred the user function blocks back then.

A message will appear showing the number of copied user function blocks, as well as the number of any user function blocks that were not copied (if any).

If you want to transfer easySoft 7 user function blocks manually to easySoft 8, follow the steps below:

- ▶ Use Windows Explorer to open the C:\ProgramData\Eaton\easySoft 7\UserFBs folder.
- ▶ Copy the user function block with the name.
- ▶ Switch to the C:\ProgramData\Eaton\easySoft 8\UserFBs folder.
- ▶ Paste the copied \*.uf7 file.
- ▶ Close and reopen easySoft 8.

The user function blocks will be shown in easySoft 8, *Programming view/user function blocks/Archive*.

### 6.3.11 Replacing a user function block

## 6. Function blocks

### 6.3 UF - 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."

### 6.3.12 Deleting a user function block

The function can be used to remove user function blocks from easySoft 8. 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 *Menu bar/Delete user function blocks* will not be available.

The following options are available for deleting a user function block:

The following applies to easySoft 7:

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. *User function blocks folder in list of operands and function blocks <name>/Delete user function block... context menu option*

The following applies to easySoft 8:

1. *Program/Delete user function blocks... menu option*
2. *User function blocks folder in list of operands and function blocks/Project/Delete user function blocks... context menu option*
3. *User function blocks folder in list of operands and function blocks/Project/<name>/Delete user function block... context menu option*
4. *User function blocks folder in list of operands and function blocks/Archive/Delete user function block... context menu option*
5. *User function blocks folder in list of operands and function blocks/Archive/<name>/Delete user function block... context menu option*

The following dialog box will appear if you use one of the first two options:

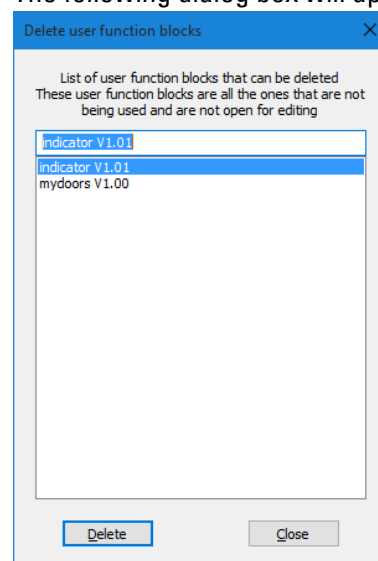



Fig. 286: Delete user function blocks dialog box

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

## 6. Function blocks

### 6.3 UF - User function block

on the  button, the function block will be deleted. After this, the user function block will no longer be part of easySoft 8 and will no longer be available in the *list of operands* 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* directly.

### 6.3.13 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 8 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:

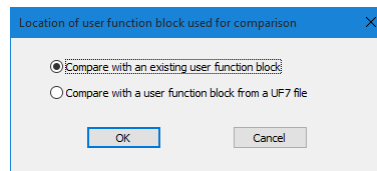


Fig. 287: Location of user function block user for comparison dialog box

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.

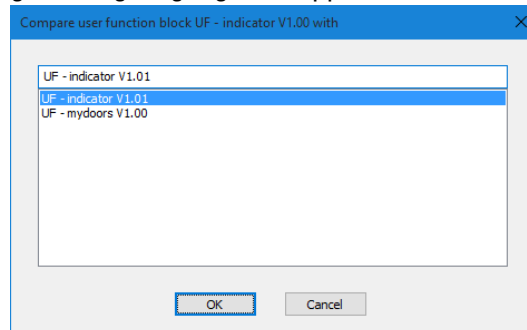


Fig. 288: UF user function block

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.

## 6. Function blocks

### 6.3 UF - User function block

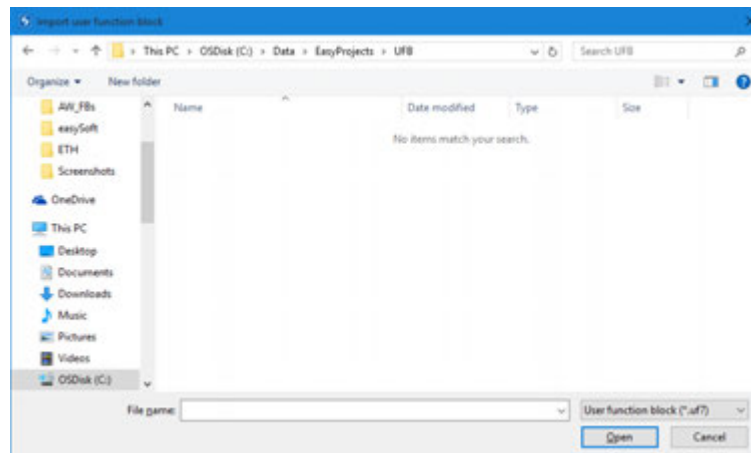


Fig. 289: Import user function block



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 in 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.

#### 6.3.14 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.

## 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.

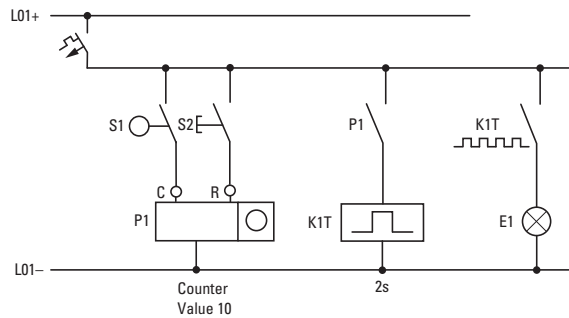


Fig. 290: Hardwiring with relays

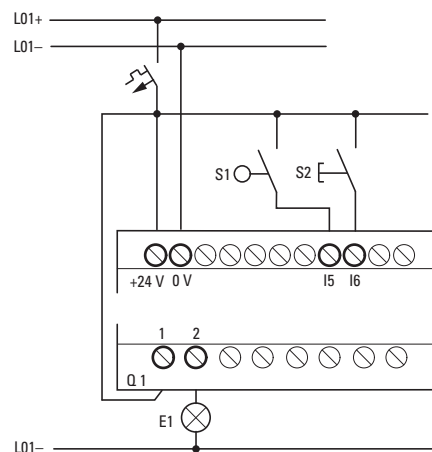


Fig. 291: Wiring with EASY-E4-UC-..., for example

### Enter circuit diagram

► Enter the following circuit diagram while using the EDP programming language.

```
I 05-----Ä C 01C
I 06-----Ä C 01RE
C 01OF-----Ä T 01EN
T 01Q1-----Ä Q 01
```

Fig. 292: 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.

## 6. Function blocks

### 6.4 Timing and counter relay example

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
```

Fig. 293: 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>
```

Fig. 294: 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.



## 6. Function blocks

### 6.4 Timing and counter relay example

- ▶ 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
```

Fig. 295: 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
```

Fig. 296: 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
```

Fig. 297: 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 "C - Counter Relay", page 305
- Section "CF - Frequency counter", page 311
- Section "CH - High-speed counter", page 317
- Section "CI - Incremental Counter", page 323

## 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 8 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 8 as of this writing:

### Connection to other devices

Setting up a NET group	→ page 721
Modbus TCP	→ page 830
Setting up a Web Server	→ page 728
Web Client	→ page 736
Setting up the e-mail function	→ page 758
Define program name	→ page 650
Retention function	→ page 651
Configuring the microSD card and device ID Configuring the card and device ID	→ page 658
Connecting to the AWS Cloud	→ page 792

7. System settings

7.1 System options - Base device with display and buttons

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:

Tab. 88: *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 → Section "Security – password protection", page 654
System	Accessing the system settings: Debounce, → Section "Debounce", page 648 P Buttons, → Section "P buttons", page 649 RUN Start, Card Start, → Section "Setting the startup behavior", page 645 Load Card, → Section "Configuring the microSD card and device ID", page 658 Display, Display settings, → Section "Display", page 636 Device ID, Device IDs, → Section "Device ID", page 636 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 637
Menu language	Used to set the device menu language, → Section "Switch languages", page 644
DELETE PROGR.	Deletes the program on the easyE4 from the device memory
NET	Used to configure a <b>NET GROUP</b> as a group of multiple devices, → Section "Setting up a NET group", page 721 The submenu is only provided in English
ETHERNET	Used to configure the Ethernet settings on the device, → Section "Ethernet", page 640 The submenu is only provided in English
UPDATE	Firmware Update for easyE4 expansion devices and communication modules. → Section "Update", page 642

Tab. 89: *System option-*

sSystem
DEBOUNCE
P BUTTONS ✓
RUN MODE
CARD MODE
LOAD CARD
DISPLAY
DEVICE-ID
BOOT LOGO

7.2 Display

This menu can be used to configure the settings for the display.

Tab. 90: *System Option-  
slSystem\Display*

BRIGHTNESS1	100
BRIGHTNESS2	50
TIMEOUT:	10m
COLOR:	0

BRIGHTNESS1	Display brightness when the device is being operated Default value: 100, can be changed in increments of 10
BRIGHTNESS2	Brightness for sleep mode Default value: 50, can be changed in increments of 10 Value: 0 will cause the display to be switched off in sleep mode
TIMEOUT	Used to set the time in minutes or seconds after which the display will go to sleep if the easyE4 is not being actively operated
PAINT	Relevant to easyE4 remote operation Color value of 0 – 15, This setting will affect the way the device is displayed, e.g., in easySoft 8 or on the web server

7.3 Device ID

Used to set / enter the individual device IDs for transferring programs.

Tab. 91: *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 8 regardless of whether an ID has been set in the program itself.



## 7. System settings

### 7.4 Splash screen

### 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.

Tab. 92: *System Options\Splash Screen*

DISPL.
3 s

#### See also

→ Section "Setting a splash screen for the EASY-E4-...-12...C1(P) display", page 147

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.

Tab. 93: Net configuration on device

Tab. 94: *Main menu*

STOP ✓ RUN
PARAMETERS
SET CLOCK
CARD
INFORMATION
SYSTEM OPTIONS
PROGRAM

Tab. 95: *System options*

SECURITY
SYSTEM
MENU LANGUAGE
DELETE PROGR.
NET
ETHERNET
UPDATE

Tab. 96: *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), 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

## 7. System settings

### 7.5 NET

#### **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 that 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 group", page 721

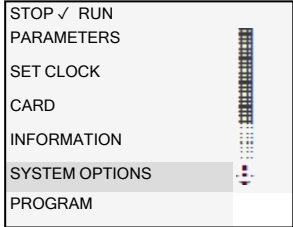
7.6 Ethernet

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*

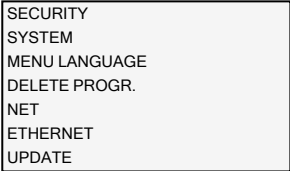
The submenu is only provided in English

Tab. 97: Ethernet configuration on device

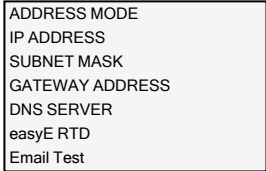
Tab. 98: *Main menu*



Tab. 99: *System options*



Tab. 100: *System option-s\Ethernet*



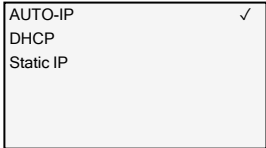
- Use the cursor buttons to enter the device's IP address.

Tab. 101: *System Option-s\Ethernet\IP Address*



- Select the network setting you want.

Tab. 102: *System Option-s\Ethernet\Address mode*



Used to manage remote control permissions for controlling the easyE4 with the EASY-RTD-....

Tab. 103: *System option-s\Ethernet\easyE RTD*





7. System settings

7.6 Ethernet

Tab. 104:

- Set up access permissions for each EASY-RTD-... user group.

Tab. 105: System option-  
s\Ethernet\easyE RTD\access  
inhibit

NO ACCESS	✓
WATCHING	
OPERATION	
ADMINISTER	

See also

→ Section "Establishing an Ethernet connection and transferring a program or visualization project", page 117

## 7.7 Update

This submenu can be used to load new firmware onto easyE4 base devices and easy communication modules.



The easyE4 base devices can only be updated directly using the microSD memory card (no special menu).

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 as .zip files via its Download Center - Software page (under Firmware Updates).



## Download Center - Software

[Eaton.com/software/FirmwareUpdates/easy](http://Eaton.com/software/FirmwareUpdates/easy)

[Eaton.com/software/OS Updates/easy](http://Eaton.com/software/OS%20Updates/easy)

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



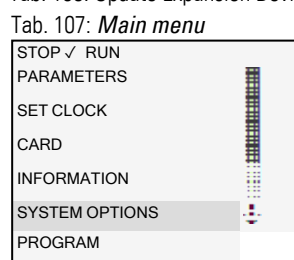
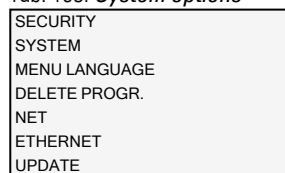
Unzip the required firmware file matching the easyE4 expansion device "\*.FW" on the microSD memory card.

The easyE4 expansion device must be connected to the base device with the plug connector.

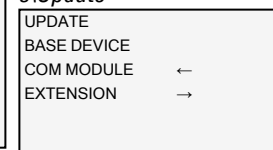
The number of the easyE4 expansion is determined based on the position after the base devices, starting with 1 from the left. The maximum number 11 can be assigned to an expansion in the assembly block.

An update must be carried out separately for each expansion device.

Tab. 106: Update Expansion Devices

Tab. 108: *System options*

Tab. 109: *System option-  
s\Update*

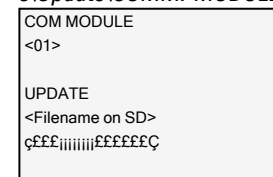


Skip the number <01>.



Select the corresponding firmware file.  
"eComSWD\_B0028.fw", for example

Tab. 110: *System options\Update\COMM.-MODULE*



## 7.7 Update

- Tab. 111: *System option-  
s\Update\Expansion*

## See also

643

## 7.8 Switch languages

The menus are available in several languages.

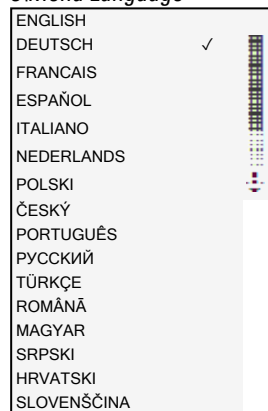


Languages can be set on base devices without a display with easySoft 8.

### 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.

Tab. 112: *System Option-  
s\Menu Language*



The language will change when you exit the menu.

## 7. System settings

### 7.9 Setting the startup behavior

#### 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 \*.e80 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 8 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 115

##### **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 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 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 what this option does, please refer to → "Functions of the microSD memory card", page 146.

Default setting CARD START inactive

## 7. System settings

### 7.9 Setting the startup behavior

#### 7.9.2.1 Configuration on a base device with a display

In order to be able to configure this option, 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 Configuration in easySoft 8

You can enable and disable this option in easySoft 8.

- ▶ 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 658


## **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  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 8**

You can enable and disable the input delay in easySoft 8.

- ▶ 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. System settings

### 7.11 P buttons

#### 7.11 P buttons

"P buttons" are the eight 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.

##### 7.11.1 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.

##### 7.11.2 Configuring the P buttons in easySoft 8

You can enable and disable the P buttons in easySoft 8.

- ▶ 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.

## **7.12 Define program name**

Only possible with easySoft 8.

You can name your program in easySoft 8.

- ▶ 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. System settings

### 7.13 Retention function

### 7.13 Retention function

Only possible with easySoft 8.

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*

Fig. 298: Project view, System settings tab with Retention section

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 ...1024
- MW : 1...512
- MD : 1...256

The values from the input field will be automatically converted to MB marker bytes.



Marker ranges up to MB1024 can therefore be defined as retentive, since e.g. MD265 corresponds to a marker byte range of 1021-1024 and the retentive marker areas are only stored in MB.

Only available on easySoft Version 8.00 or higher.

If marker bytes are entered in the input field, these are also converted into the highest possible data type, provided the number of marker bytes allows this. The converted data type is displayed after a new switch to the System Settings tab.

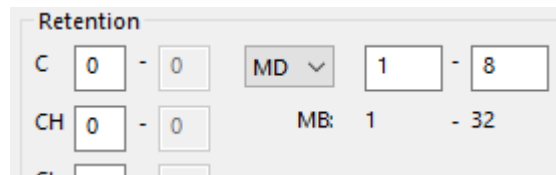


Fig. 299: Retention section: Marker bytes 1 - 32 entered and display in marker double words after another change to the System settings tab

#### Retention bytes

The entire retentive marker range for an easyE4 must not exceed a specific number of bytes. Depending on the firmware installed on the basic device, the following number of available bytes applies here:

- Firmware  $\geq$  2.30: 1024 Bytes
- Firmware  $\geq$  2.00: 512 Bytes
- Firmware  $<$  2.00: 400 Bytes

The sum of the retentive markers of the main program and the retentive markers of all user block instances (UF) is displayed in the Project view under the System Settings tab. If the retentive marker range exceeds the number of available 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 in the Communication view with the *Communication view/Program/Configuration/Delete device* button
- By any change of the retentive range in the Project view with *Project view/System Settings tab/Retention*.
- By any changes to the parameters for the remote markers of a visualization device.
- When deleting the device from the work pane in 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. System settings

### 7.13 Retention function

#### 7.13.1 Retention in easySoft 8

You can configure the retention function both for markers and for function block contents in easySoft 8.

- ▶ 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.

## 7.14 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 8.

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.

### 7.14.1 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.

Tab. 113:

System options\Security\Area

PROGRAM	✓
PARAMETERS	
CLOCK	
OPERAT.	
MEMORY CARD	
INTERFACE	
DELETE FUNCT.	

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.
PARAMETERS	The PARAMETERS menu is protected.
CLOCK	Date and time are protected with the password.
OPERAT.	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.

7. System settings

7.14 Security – password protection

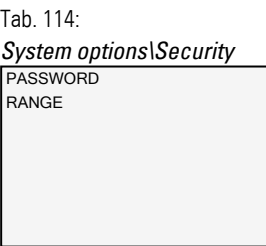
INTERFACE	Protects against access to the Ethernet interface of this device. Data exchange via the net is not affected.
	➔ Take into account the restricting effect of a protected interface if you have to reset the easyE4 device.
DELETE FUNCT.	When this function is not activated, the question "DELETE PROG?" 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.



You can use any numbers or letters for the six-character password. Special characters and umlauts are not permitted.



The first password character will flash.

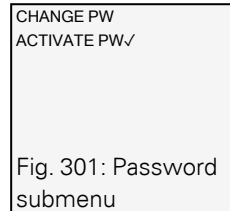
- Select the first letter or number for the password.
- Confirm the entry with the **OK** pushbutton.
- Repeat these steps for the remaining characters in the password.

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.



- ▶ 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 654.

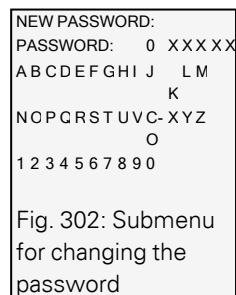
**Changing a password**

- ▶ 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.



To assign a new password, follow the same steps outlined in → Section "Assigning a password", page 655

**Removing password protection**

To disable password protection, assign a password of <000000> .



## 7. System settings

### 7.14 Security – password protection

#### 7.14.1.1 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 Configuring the microSD card and device ID

Only possible with easySoft 8.

- ▶ 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 transferred 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.



This helps prevent the person configuring the project from transferring an .e80 project that is not suitable for the specific application to the easyE4 by accident. In other words, the system would detect this mistake based on the non-matching ID.

### See also

- Section "Transferring programs from and to a microSD memory card", page 213
  - Chapter "7 Functions of the microSD memory card", page 146
  - Section "Device ID", page 636
- easySoft V8 Help, Communication view

## 7. System settings

### 7.16 Time and Date setting

#### 7.16 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.



Setting the date and time on the base device without a display is only possible with easySoft 8

##### Setting 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.

Tab. 115: *Set Clock-*

*lDate&Time*

DD-MM-YYYY
FR 2018/08/13
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.

Tab. 116: *Set clock\Daylight saving*

NONE	✓
MESZ	
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.

Tab. 117: *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.

7. System settings

7.16 Time and Date setting

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 over-written 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.

Tab. 118: *Set Clock\Radio*

<i>Clock</i>	
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 296 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.

Tab. 119: *Set clock\astron.*

*clock*

ASTRO		CLOCK
LAT	N089.	9990000
LON	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 8 for the program. To do this, you can transfer the modified program to easySoft 8 and save it there if you want to apply this information to the location data in the project.



## 7. System settings

### 7.16 Time and Date setting

#### Example

Settings for time zone in Bonn (UTC+1 hour) in decimal degrees

Tab. 120: *Set clocklastron.*

*clock*

ASTRO		CLOCK
LAT	N050.	734012
LON	E007.	082808
OFFSET		: +060'

and DMS

Tab. 121: *Set clocklastron.*

*clock*

ASTRO		CLOCK
LAT	N050°	44'02"
LON	E007°	04'58"
OFFSET		: +060'

#### See also

Timer modules

- "HW - Weekly timer (Hour Week)", page 244
- "HY - Year time switch (Hora Year)", page 254
- "WT - Weekly timer (WeekTable)", page 292
- "YT - Year time switch (Year Table)", page 284
- "AC - Astronomic clock ", page 296

## **8. How easyE4 works internally**

### **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 used 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.

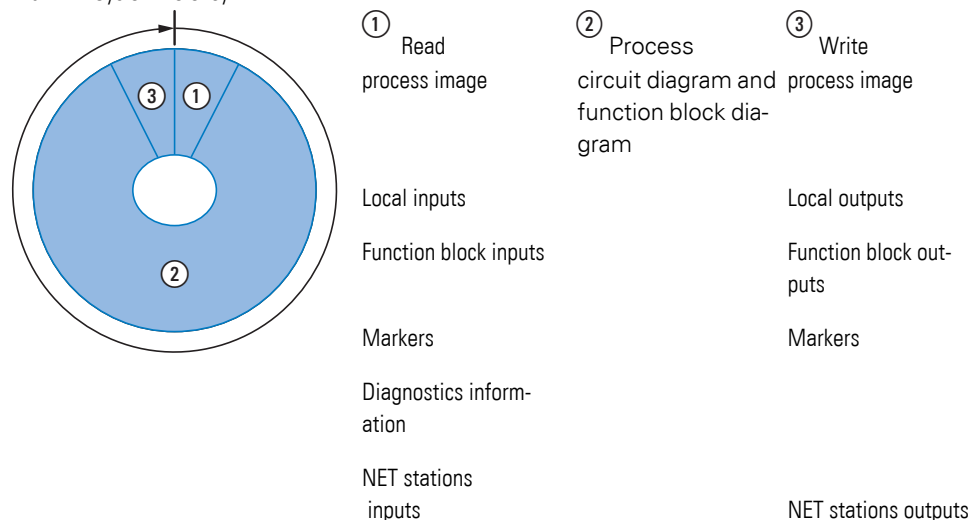
In conventional control systems, a relay or contactor control processes all the rungs in parallel. The speed with which a contactor switches is thus dependent on the components used, and ranges from 15 to 40 ms for relay pick-up and drop-out.



## 8. How easyE4 works internally

### 8.1 Program execution

Tab. 122: Cycle time easyE4



If the operands of the inputs and outputs are addressed in the easyE4 program, 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 ① and the process image for the outputs ③.

During this time, the easyE4 device passes through six segments in succession.

#### 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.

## 8. How easyE4 works internally

### 8.1 Program execution

#### 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.

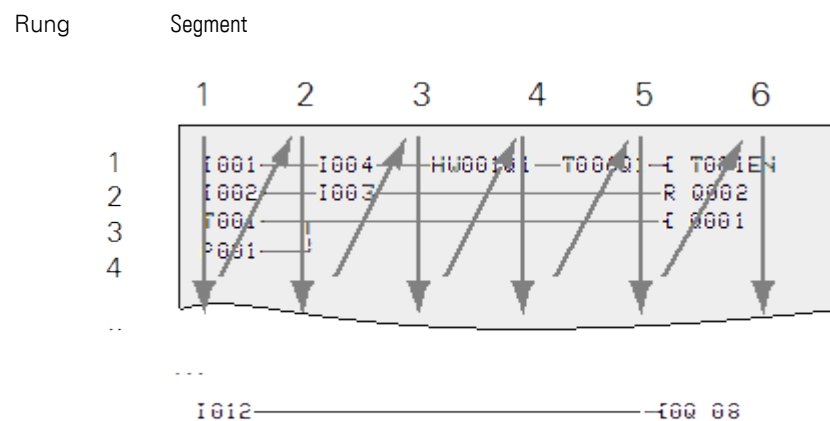


Fig. 303: How the EDP evaluates circuit diagrams and function blocks

## 8. How easyE4 works internally

### 8.2 Transferring an existing circuit diagram

#### 8.2 Transferring an existing circuit diagram

Existing easy.e60/e70 programs can be imported with easySoft 8.

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 8 will output a conversion protocol 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 8 will use the previous devices and the operands used as a basis in order to optimize the easyE4 hardware and the new <xyz>.e80 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 634
- 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 firmware version

- E4- : Part number
- OS : 1.30(Version)
- B : 510(build version)
- CRC : 60268(checksum)

8. How easyE4 works internally

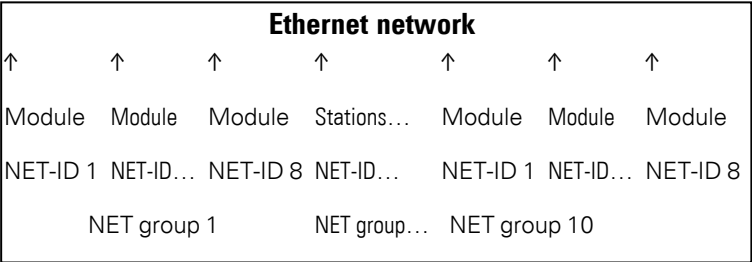
8.4 NET network

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).



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

## 8. How easyE4 works internally

### 8.4 NET network

NET-ID1    NET-ID 2

PT16    →    GT 01  
                  Parameter  
                  NET-ID 1  
                  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).

8. How easyE4 works internally

8.4 NET network

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	Delay without put/get
	ms	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 8, 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 8 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 8 via Ethernet.
- The base device can be switched to the RUN operating mode with the menu or through easySoft 8. 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. How easyE4 works internally

### 8.6 Controlling the backlight with operands

## 8.6 Controlling the backlight with operands

### 8.6.1 Backlight intensity

Only for easyE4 base devices with a display.

easyE4 features three LE operands, LE1 through LE3. These operands are programmable outputs that can be used to control the device display's backlight and visually signal states on the display.

You can set two brightnesses within a value range of 0 to 100% on the easyE4 base device: backlight intensity 1 and backlight intensity 2. The default settings are as follows: backlight intensity 1=100%, backlight intensity 2 = 50%. For more information on how to configure this in the device menu, please refer to → "Display", page 636

You can use the LE1 output operand to set the backlight intensity to brightness 1 for the device display in RUN mode. This means, for example, that you can set and clear the LE1 and LE3 operands cyclically to produce a flashing effect.

You can use the LE2 output operand to set the backlight intensity to brightness 2 for the device display in RUN mode.

LE3 turns the backlight off.

If the easyE4 base device switches to STOP mode, backlight intensity 1 will be reactivated as per the device menu setting.

If more than one operand out of LE1 through LE3 is set, the operand with the highest priority will be used for the backlight setting.

Backlight intensity on device display	LE01	LE02	LE03
Backlight intensity 1	1	0	0
Backlight intensity 2	0	1	0
Off	0	0	1

### 8.6.2 Background color

Available on easySoft V7.30 and higher with FW 1.20 and higher

Visualization devices feature additional outputs for controlling the device display's color. These outputs are switched in the circuit diagram with output operands LE04 to LE06.

Background color on the device display	LE04	LE05	LE06
red	1	0	0
green	0	1	0
white	0	0	1

#### Example: Backlight flashing

## 8. How easyE4 works internally

### 8.6 Controlling the backlight with operands

Say you want the device display to flash white at an interval of two seconds. You also want flashing in red or green to be possible if selected.

The following program must be downloaded onto the device.

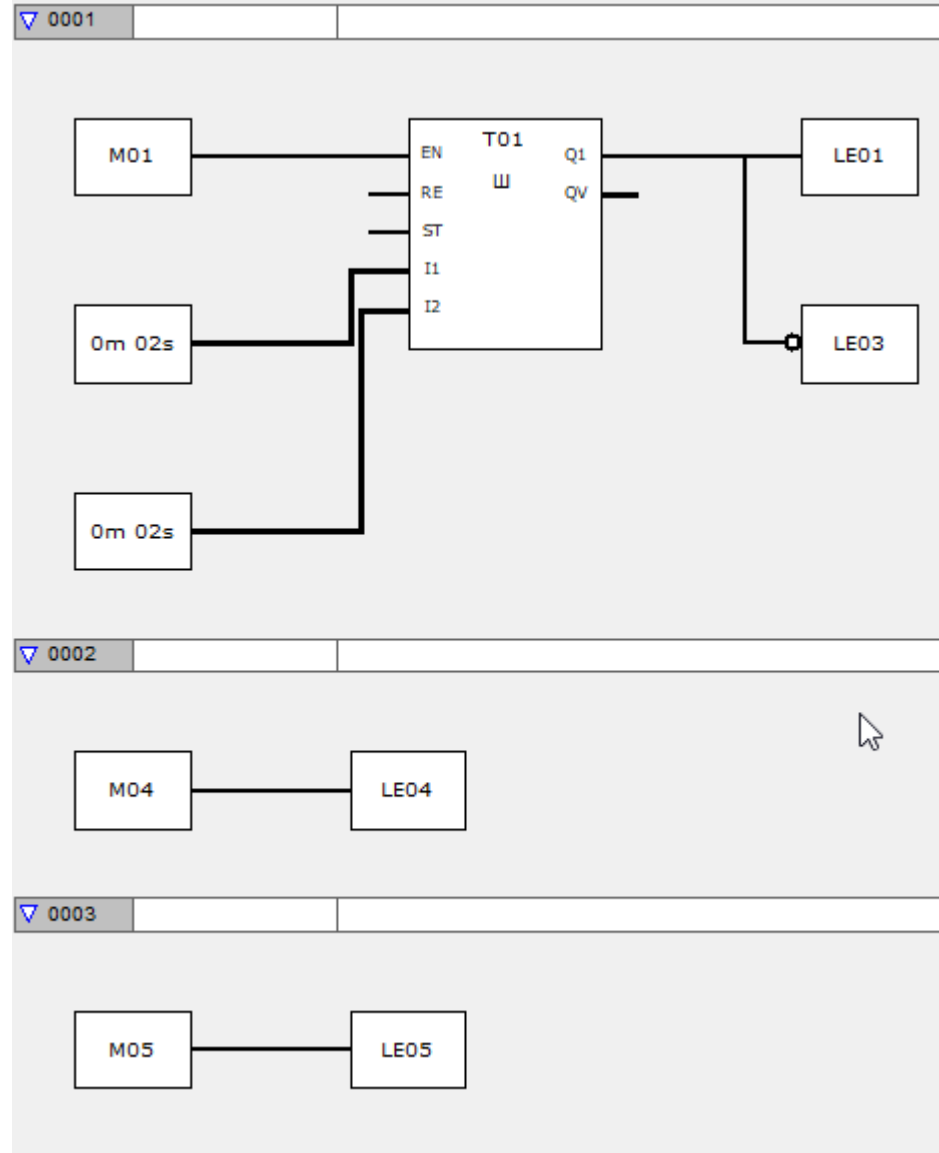


Fig. 304: Programming view/Sample program in FBD

To test it, you will first need to establish online communications with the device.

Set marker M01 to cause the device display to flash.

Then set marker M04 as well so that the device display will flash red. Clear M04.

Then set marker M05 in addition to M01 so that the device display will flash green.



## 8.7 Device easyE4 time responses

### 8.7.1 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 648

This function is useful, for example, in order to ensure a clean switching signal despite contact bounce.

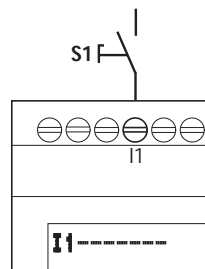


Fig. 306: easyE4 input assigned a switch

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. How easyE4 works internally

### 8.7 Device easyE4 time responses

#### 8.7.2 Base device timing

##### 8.7.2.1 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.

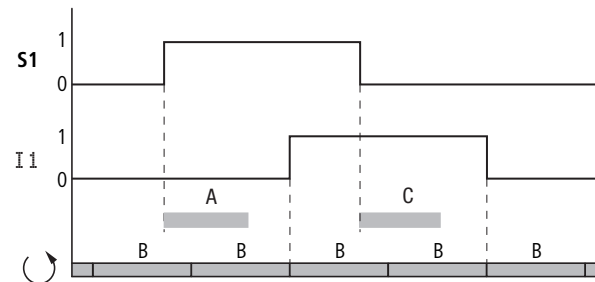


Fig. 307: Delay times for evaluating an DC input signal and an activated debounce

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 883

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.

## 8. How easyE4 works internally

### 8.7 Device easyE4 time responses

#### 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.

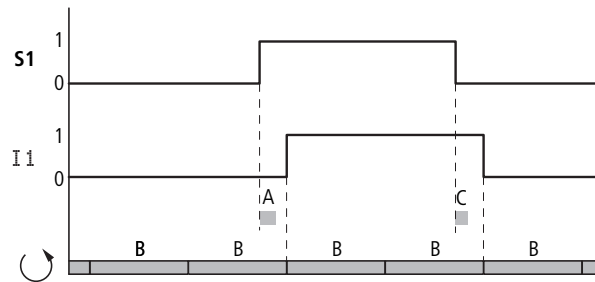


Fig. 308: Switching behavior with debounce deactivated

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 883



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. How easyE4 works internally

### 8.7 Device easyE4 time responses

#### 8.7.2.2 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.

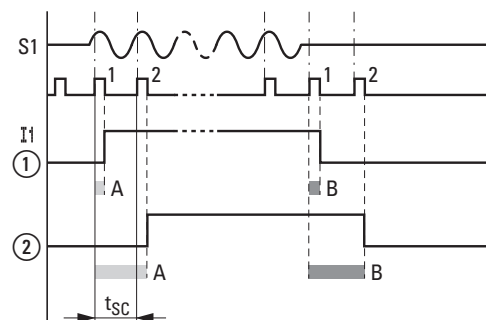


Fig. 309: Delay times for evaluating an AC input signal

- 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  $t_{SC}$  (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.

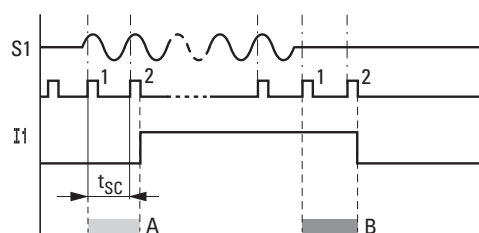


Fig. 310: Switching behavior of AC input signal with DEBOUNCE activated

Accordingly, the typical input delay caused by the debounce option is at least 40 ms (50 Hz). If applicable, this time must also include the cycle time since a easyE4 device does 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.

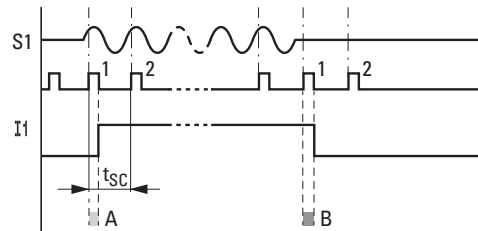


Fig. 311: Switching behavior of AC input signal with DEBOUNCE deactivated

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 out-  
puts", page 676



## 8. How easyE4 works internally

### 8.7 Device easyE4 time responses

#### 8.7.3 Timing characteristics of expansion devices

You can use the 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, i.e., easyConnect (easyConnect is the local bus system to the expansion devices).

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.

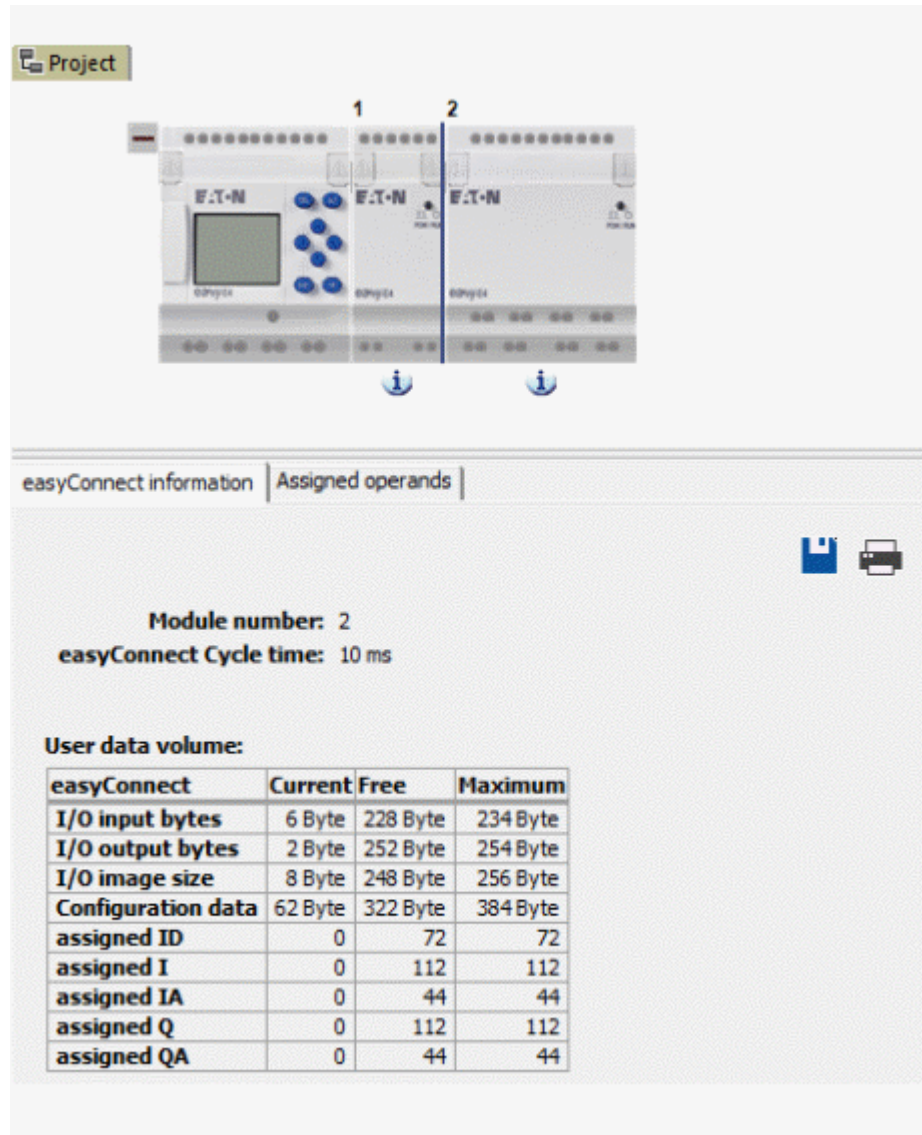
If, on the other hand, the easyConnect cycle is slower than half the program cycle, the inputs and outputs may be refreshed after two program cycles instead.

The easyConnect cycle time will be between 10 ms and 15 ms depending on the specific configuration.

To see the easyConnect cycle time in the Project view, click between the easyE4 base device and the expansion device or between two expansion devices.

## 8. How easyE4 works internally

### 8.7 Device easyE4 time responses



#### 8.7.3.1 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 883

**8. How easyE4 works internally**  
**8.7 Device easyE4 time responses**

### 9. Operating system diagnostic messages

The easyE4 can log operating states with its data logging feature and run a simple analysis of events. In addition, it makes diagnostics easier with status information for all communication nodes and expansion modules.

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. The NET station 1 is not present:
ID02	There are more than two devices in this NET group and the NET is active. The NET station 2 is not present:
ID03	There are more than two devices in this NET group and the NET is active. The NET station 3 is not present:
ID04	There are more than two devices in this NET group and the NET is active. The NET station 4 is not present:
ID05	There are more than two devices in this NET group and the NET is active. The NET station 5 is not present:
ID06	There are more than two devices in this NET group and the NET is active. The NET station 6 is not present:
ID07	There are more than two devices in this NET group and the NET is active. The NET station 7 is not present:
ID08	There are more than two devices in this NET group and the NET is active. 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: <ul style="list-style-type: none"> <li>• "Synchronize clock via NET"</li> <li>• "SNTP synchronization"</li> <li>• Date and time</li> <li>• DCF77 radio clock</li> </ul> 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.

## 9. Operating system diagnostic messages

Operand	Event
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	Configuration not OK
ID16	ComBUS group error
ID17	ComBUS interval too long
ID18	SD card present (firmware version 1.40 and higher).
ID19	There is an interrupt overload. One or more interrupt function blocks are being used, and the interrupt function block sequence is overloading the easy4's controller. Not all interrupt function blocks can be run correctly.
ID20	There is no connection to the cloud.
ID21	Wrong device time

Additional diagnostic messages for the expansion devices can be assigned to diagnostic operands ID25 through ID96 based on the relevant device characteristics.

### Example EASY-E4-DC-6AE1(P)

diagnostic alarm	Meaning
DIAG	General diagnostic indicating that a diagnostic event is present
DIAG 1	Current input overload Current input overloaded (current greater than 23 mA), excessive voltage
DIAG 2	Wire breakage ( $I < 4\text{mA}$ ) Analog output overloaded, excessive current, load too small Open wire at at least one current input ( $I < 4\text{ mA}$ )
DIAG 3	Output overload The measuring range has been physically exceeded at an input
DIAG 4	Value range exceeded at output The measuring range has been physically fallen below at an input, e.g., the current is $< 4\text{ mA}$ for a measuring range of 4–20 mA.
DIAG 5	Value range fallen below at output The measuring range is physically exceeded at an input, e.g., the current is $> 4\text{ mA}$ for a measuring range of 4–20 mA.
PRSNT	Expansion present (firmware version 2.00 and higher)

### Example EASY-E4-DC-4PE1(P)

diagnostic alarm	Meaning
DIAG	General diagnostic indicating that a diagnostic event is present
DIAG 1	Measurement range underrun Configured measuring range fallen below at at least one temperature input, or a short-circuit has occurred
DIAG 2	Measurement range overrun Configured measuring range exceeded at at least one temperature input, or connection cable discontinuity

## 9. Operating system diagnostic messages

### 9.1 Diagnostic messages easy communication module

#### 9.1 Diagnostic messages easy communication module

The following diagnostic alarms can be assigned to base device operands automatically or manually in *Project view/Assigned operands*.

Diagnostic alarms	Description	EASY-COM-SWD-C1	EASY-COM-RTU-M1
PRSNT	Expansion present	✓	✓
RUN	Cyclical data is active	✓	✓
STOP	No cyclical data! (failsafe)	✓	✓
RegMissing	Required module missing (EASY-COM-SWD-C1 only)	✓	–
CfgError	SWD configuration fault	✓	✓
OptMissing	Optional module missing (EASY-COM-SWD-C1 only)	✓	–
ReplByNOP	Module replaced with NOP module (EASY-COM-SWD-C1 only)	✓	–
ReplByComp	Module replaced with compatible module	✓	–
ERROR	Fault Status	–	✓

## 9. Operating system diagnostic messages

### 9.2 Transistor outputs (overload / short-circuit)

#### 9.2 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.

##### **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**.

#### 9.3 Diagnostics and diagnostic buffer

Only possible with easySoft 8.

Helpful information regarding diagnostics and the diagnostic buffer will be shown in the Communication view in online mode in easySoft 8.

## 9.4 LED status messages on the device

For diagnostic purposes, base devices without a display feature two LEDs, while expansion devices feature and easy communication modules 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, Flashing, 1 Hz	Supply voltage OK, STOP mode
Green, Flashing, 4 Hz	Fault at one of the expansions, between the easyE4 device and the connector

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

Off	No Ethernet cable connected; supply voltage off 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, 2 flashes, pause, etc.	NET data flow working; one or more NET stations missing
Green, flashing, 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, Flashing, 1 Hz	Supply voltage OK, no data exchange with base device
Green, Flashing, 3 Hz	Supply voltage OK, no data exchange with base device, diagnostic bit will be set, device not working
Green, Flashing, 10 Hz	Device waiting for firmware update
Green, Flashing, 0.5 Hz	Firmware update in progress



## 9. Operating system diagnostic messages

### 9.4 LED status messages on the device

#### LED POW/RUN easy communication moduleEASY-COM-SWD-C1

Off	Malfunction or no supply voltage
Green, continuous light	Supply voltage OK, RUN mode
Green, Flashing, 1 Hz	Supply voltage OK, STOP mode
Green, Flashing, 3 Hz	Supply voltage OK, STOP mode EASY-COM-SWD-... and easyE4 are unable to exchange data e.g. bus interface connector not plugged in, faulty or easyE4 switched off
Green, Flashing, 10 Hz	Device waiting for firmware update
Green, Flashing, 0.5 Hz	Firmware update in progress

#### LED POW/RUN easy communication moduleEASY-COM-RTU-M1

Off	Malfunction or no supply voltage
Red, flashing, 5 Hz	Serious error; the UART interface between EASY-COM-RTU-... and the easyE4 base device cannot be initialized, i.e., no data transfers between EASY-COM-RTU-... and easyE4
Green, continuous light	Operating mode RUN, normal operating mode: <ul style="list-style-type: none"> <li>No communication errors with the ComBUS</li> <li>No missing slaves on the Modbus (in master mode)</li> </ul>
Green, Flashing, 1 Hz	STOP mode <ul style="list-style-type: none"> <li>The easyE4 base device is in the STOP state.</li> <li>In master mode: One of the slave devices is not present / is not reporting in</li> </ul>
Green, Flashing, 3 Hz	Error in Modbus RTU communications: ComBUS error <ol style="list-style-type: none"> <li>CRC fault</li> <li>Timeout error</li> </ol>
Green, Flashing, 10 Hz	Device waiting for firmware update
Green, Flashing, 0.5 Hz	Firmware update in progress

## 10. Communication Connection to other devices easyE4

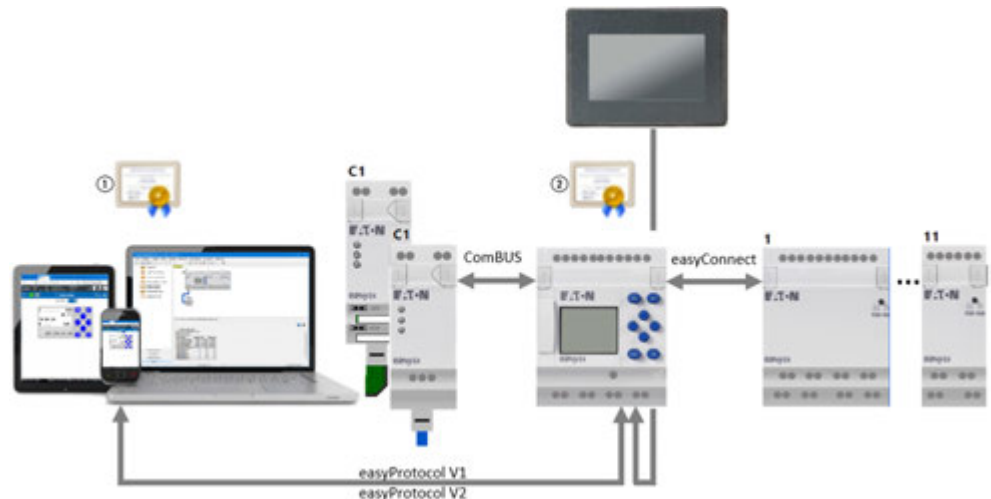


Fig. 312: communication diagram easyE4

- ① Eaton easyE4 root certificate
- ② easyE4 device certificate

The easyE4 base device has various interfaces for communication.

- easyConnect is the interface to the digital and analog expansions.
- ComBUS is the interface to the communication modules, such as EASY-COM-SWD-C1, EASY-COM-RTU-M1.
- The Ethernet interface is used for communication with the easyE RTD visualization devices.

The protocols for these interfaces are proprietary.

If you want to establish a secure connection to the easyE4 base device through easySoft 8, a web browser, or JSON:API but are only getting the option of using non-secure connections, make sure that the device time on the easyE4 base device is up to date. If the device time is not up to date, this can result in problems with the certificate check when attempting to establish a connection.

The Ethernet interface on the easyE4 base device can be used for various purposes. The following scenarios are contemplated:

Purpose of communications	Ethernet interface with the following higher protocols	Certificate request
easyE4 Ethernet interface	easyProtocol V1	—
	easyProtocol V2 SSL/TLS	√
	easyProtocol V2 (not encrypted)	—
easyE4 as Webserver	http	—
	https	√
JSON:API	http	—
	https	√

## 10. Communication Connection to other devices easyE4

### 10.1 Secure communications with easyProtocol V2

#### 10.1 Secure communications with easyProtocol V2



To use secure communications with easyProtocol V2, you will need Windows 8 or higher as an operating system.

Only available on firmware version 2.00 or higher.

Generation 06 easyE4 base devices can be configured and programmed with easyProtocol V2 through connections considered trusted and secure. In addition to this, easyProtocol V2 is not just a secure method of communications, but also one with better performance than easyProtocol V1.



easyProtocol V2, SSL/TLS comes enabled by default on easyE4 base devices with firmware version  $\geq 2.00$ . This ensures that communications can only be established in an encrypted manner with easyProtocol V2 with new devices, which additionally requires the Eaton easyE4 root certificate. If this certificate has been installed on the same PC on which easySoft 8 is running, secure communications will be established through the Ethernet port.

If an easyE4 root certificate has not been installed, a message will appear asking the user whether they want to trust the easyE4's device certificate. If the user agrees, a connection will be established.

There are two basic easyProtocol versions:

- easyProtocol V1, unencrypted  
This version does not require the Eaton easyE4 root certificate. easyE4 base devices with firmware version  $< 2.00$  use this protocol when communicating.
- easyProtocol V2, unencrypted or encrypted  
The encrypted easyProtocol V2 SSL/TLS version requires an Eaton easyE4 root certificate. easyE4 base devices with firmware version  $\geq 2.00$  use this protocol when communicating.

## **10. Communication Connection to other devices easyE4**

### **10.1 Secure communications with easyProtocol V2**

If you want to establish a secure connection to the easyE4 base device through easySoft 8 but are only getting the option of using non-secure connections, make sure that the device time on the easyE4 base device is up to date. If the device time is not up to date, this can result in problems with the certificate check when attempting to establish a connection.

## 10. Communication Connection to other devices easyE4

### 10.2 Secure communications via HTTPS (encrypted)

#### 10.2 Secure communications via HTTPS (encrypted)

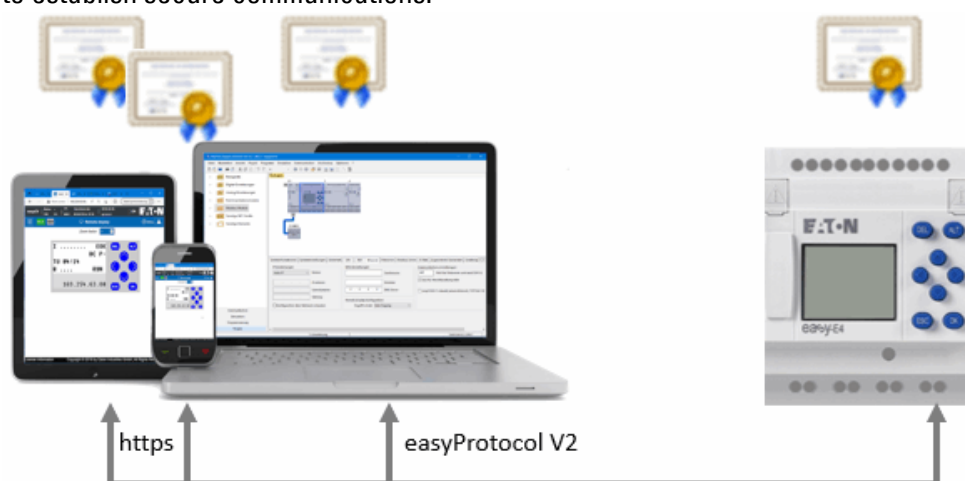
Only available on firmware version 2.00 or higher.

The web server on easyE4 base devices can use the HTTPS protocol to send data through connections that are considered trusted and secure.

The easyE4 web server will ask for the Eaton easyE4 root certificate. If this certificate has been installed on the PC/tablet/cell phone, the web browser will establish the connection and mark it as secure.

If there is no certificate on the Web Client, the next steps will depend on how the web browser has been configured.

If the browser is unable to find an easyE4 easyE4 root certificate, it will ask you whether you want to trust the easyE4's device certificate. If you confirm that you do, you will be able to establish a connection. Installing the easyE4 root certificate makes it possible to avoid this recurring confirmation prompt while still making sure to establish secure communications.



### 10.3 Windows 7 operating systems and easyProtocol V1

Secure communications with easyProtocol V2 are not available with Windows 7. In this case, communications will use easyProtocol V1 exclusively.

You can continue using existing projects. If you transfer an existing project to an easyE4 base device with firmware version 2.00 and start the project, the easyE4 base device will exclusively use easyProtocol V1 when communicating.

You can upgrade existing projects to firmware version 2.00 with easySoft 8 or create a new easySoft 8 project. However, you will need to make sure to configure the following project settings as shown before loading the project onto an easyE4 base device with firmware version 2.00:

1. The following applies to new projects with easySoft 8: The following option must be enabled in the *Project view/Ethernet tab*:  
☒ easyProtocol V1 allowed (not encrypted, TCP port 10001)
2. The following protocol version must be selected in *Communication view/Connection/IP profiles/Edit.../Communication settings*: "easyProtocol V1"



If you load a project onto the easyE4 base device without these settings, it will no longer be possible to communicate with the device with this operating system.

The reason for this is that the easyE4 base device would be expecting the configured easyProtocol V2 communications, which are not supported by the Windows 7 operating system.

This issue can be fixed in one of the following two ways:

1. Delete the project on the device; enable the ☒ easyProtocol V1 allowed (not encrypted, TCP port 10001) option in the *Project view/Ethernet tab* in the project and then transfer the project to the device.
2. Modify the project and transfer it with a microSD memory card.

## **10. Communication Connection to other devices easyE4**

### **10.4 Windows 7 operating systems and easySoft 8 – pay attention to the project size**

#### **10.4 Windows 7 operating systems and easySoft 8 – pay attention to the project size**

If you upgrade an existing project to firmware version 2.00, a larger program memory will be supported.

However, programs that are larger than 16 kB and accordingly need a larger memory than the previous one cannot be transferred using easyProtocol V1. This means that Windows 7 users will need to transfer the program with a microSD memory card in this case.

Make sure to pay attention to the project size when modifying or expanding a project!

Additional data loaded onto the device together with the project (operand comments, mapping lists, etc.) will make the project significantly larger. (Reference to: Checkbox for whether comments should be loaded together with the project). As soon as the project is larger than 16 kB, you will no longer be able to connect to the device. One possible solution is to not store the comments and notes on the device. There is an option for this –.

## 10.5 easyProtocol V1

You can use the easyProtocol V1 protocol for communications with any easyE4 base device. In other words, the use of easyProtocol V1 ensures backward compatibility.

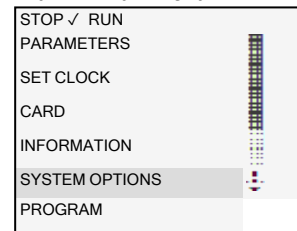
To select easyProtocol V1, you will need to configure the following settings as shown:

1. *Project/Ethernet tab* – you can pre-select options here.
2. *Communication view/Connection/IP profiles/Edit/Edit IP profiles/Communication settings/Protocol version* – you can select the protocol for establishing a connection before going ONLINE with the device here.

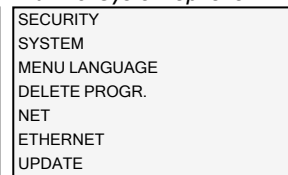
New easyE4 base devices will come with the AUTO IP setting configured by default. In order to configure the settings differently on the EASY-E4-...-12...C1(P), use the menu structure and go to *System Options\Ethernet*

Tab. 123: Ethernet addresses on the device

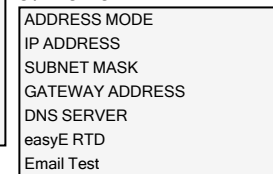
Tab. 124: Main menu



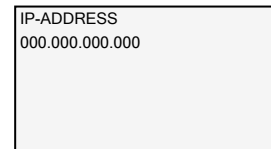
Tab. 125: System options



Tab. 126: System option-s\Ethernet

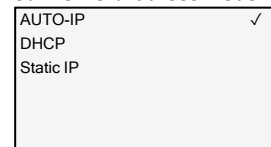


Tab. 127: System Option-s\Ethernet\IP Address



► Use the cursor buttons to enter the device's IP address.

Tab. 128: System Option-s\Ethernet\Address mode



► Select the network setting you want.

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

- 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 relay must be connected to the PC with a standard Ethernet cable featuring an RJ45 connector.



## 10. Communication Connection to other devices easyE4

### 10.5 easyProtocol V1



#### **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 8.

## 10. Communication Connection to other devices easyE4

### 10.6 Compatibility rules for going online

#### 10.6 Compatibility rules for going online

As soon as a connection is established between easySoft 8 and a device (i.e., as soon as you are ONLINE), easySoft 8 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.

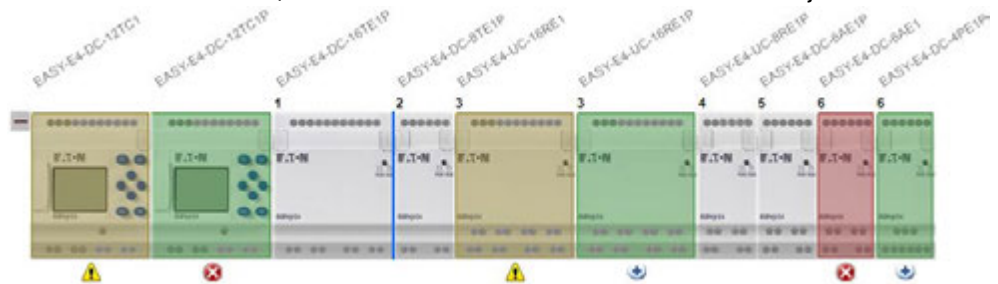


Fig. 313: ONLINE Project view with devices colored differently based on their compatibility

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.

Or, for 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 view, but an EASY-E4-DC-12TC1P is physically present

Red

Expansions / devices that are missing online and that are only found in the configuration

## 10. Communication Connection to other devices easyE4

### 10.6 Compatibility rules for going online

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-4PE1P is physically present

#### Violet

Expansions / devices that are missing online and that are only found in the configuration as optional expansions.

Only available on firmware version 2.00 or higher.

Only available on easySoft 8 or higher.

If a device is not recognized in the Communication view, this indicates that an older version of easySoft 8 is being used and that the physical device is not found in the list of devices. 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.

#### 10.7 Establish a connection to the device

Only possible with easySoft 8.

In easySoft 8, 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 relay:

- 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 91

- ▶ Open easySoft 8 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.

## 10. Communication Connection to other devices easyE4

### 10.7 Establish a connection to the device

MAC	Device type	Grp.	ID	IP address	Device name	State	Required NET stations
00:80:99:09:99:67	EASY-E4-UC-12RC1	0	0	169.254.153.103		STOP	

Selected item

MAC address: 00:80:99:09:99:67 Device type: EASY-E4-UC-12RC1

IP settings: Auto-IP Mode: IP address: Subnet mask: Gateway: ☒ Enable configuration via network

DNS settings: Device name: Domain: DNS server:

NET settings: NET group: 0 NET-ID: 0 Bus delay: Remote RUN: ☐

Required NET stations: ☐ NT1 ☐ NT2 ☐ NT3 ☐ NT4 ☐ NT5 ☐ NT6 ☐ NT7 ☐ NT8

Buttons: Save as IP profile, <= Project, => Device, Close

- ▶ 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."

## 10. Communication Connection to other devices easyE4

### 10.7 Establish a connection to the device

#### Details regarding 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 (static)	Device type (static)
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)
Device name	<p>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.</p> <p>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 currently selected data record, but these changes have not yet been transferred back to the device, the attempt to generate a new profile will be aborted with the following message:</p> <p>"Please first transfer the modified configuration to the device, as obsolete parameters will otherwise be saved in the new IP profile."</p>
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

## 10. Communication Connection to other devices easyE4

### 10.7 Establish a connection to the device

#### 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!	<p>Only relevant if you want to make a change in the "Selected item" section:</p> <ul style="list-style-type: none"><li>▶ Use the device menu to bring the device to the STOP operating status.</li></ul>
Please transfer the modified configuration to the device first. Otherwise, obsolete parameters will be saved in the new IP profile.	<p>If you have made a change in the <b>Selected item</b> 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 8 and the project on the device.</p> <ul style="list-style-type: none"><li>▶ Click on the =&gt;Device button in the Search for devices dialog box.</li><li>▶ Then click on the Save as IP profile button.</li></ul>
The device configuration is locked and you cannot make changes to it!	<p>The device configuration in the <b>Selected item</b> section cannot be modified.</p> <ul style="list-style-type: none"><li>▶ Go to the <i>Project view/Ethernet tab</i> and enable the Enable configuration via network option.</li><li>▶ Go to the <i>Communication view/Connection section</i> and click on the Online button.</li><li>▶ Click on the PC =&gt; Device button in the Program / Configuration section.</li></ul> <p>The project will be transferred to the device.</p> <ul style="list-style-type: none"><li>▶ Click on Offline.</li><li>▶ Click on Search under IP devices.</li></ul> <p>You will now be able to make changes in the <b>Selected item</b> section in the Search for devices dialog box.</p>
No devices matching the selected search filter setting were found.	<ul style="list-style-type: none"><li>▶ Check that the module rack</li><li>▶ Check to make sure that the correct PC interface, NET group, and NET ID are selected.</li><li>▶ 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 117.</li></ul>

## **10.8 Terminating the connection to the device**

The connection to the device will be terminated. The status line will show "OFFLINE."

- In order to terminate the online connection, click on the Offline button in the **Con-**  
**nection** section.



## 10. Communication Connection to other devices easyE4

### 10.9 Setting up a connection to multiple devices on the NET

## 10.9 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:

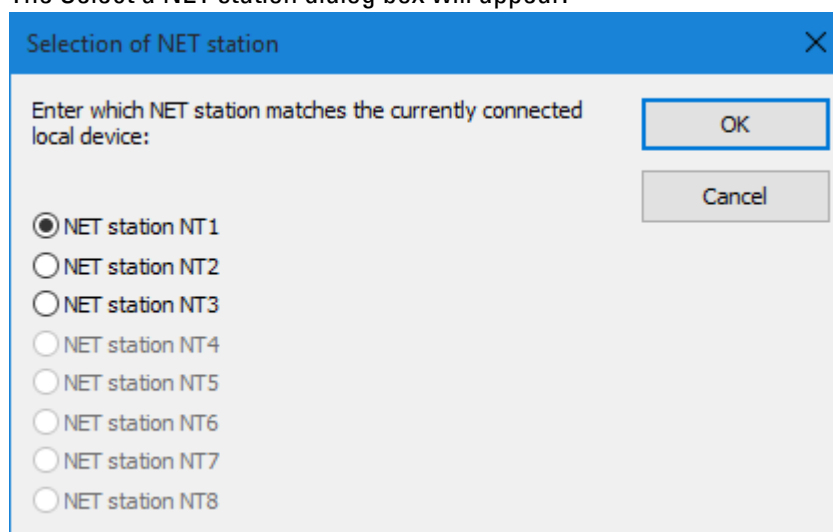
1. → "Connection parameters and program on the device", page 705: Loading the program, with the NET ID and Ethernet settings, onto each device.
2. → "Connection parameters on device", page 706: Using the Search for devices dialog box to load the NET ID and Ethernet settings onto each device.
3. Device 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 a connection to the device", page 700
- ▶ In the Connection section, click on the PC => Device button. The Select a NET station dialog box will appear.



## 10. Communication Connection to other devices easyE4

### 10.9 Setting up a connection to multiple devices on the NET

Fig. 314: 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 a connection to the device", page 700.
- ▶ 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

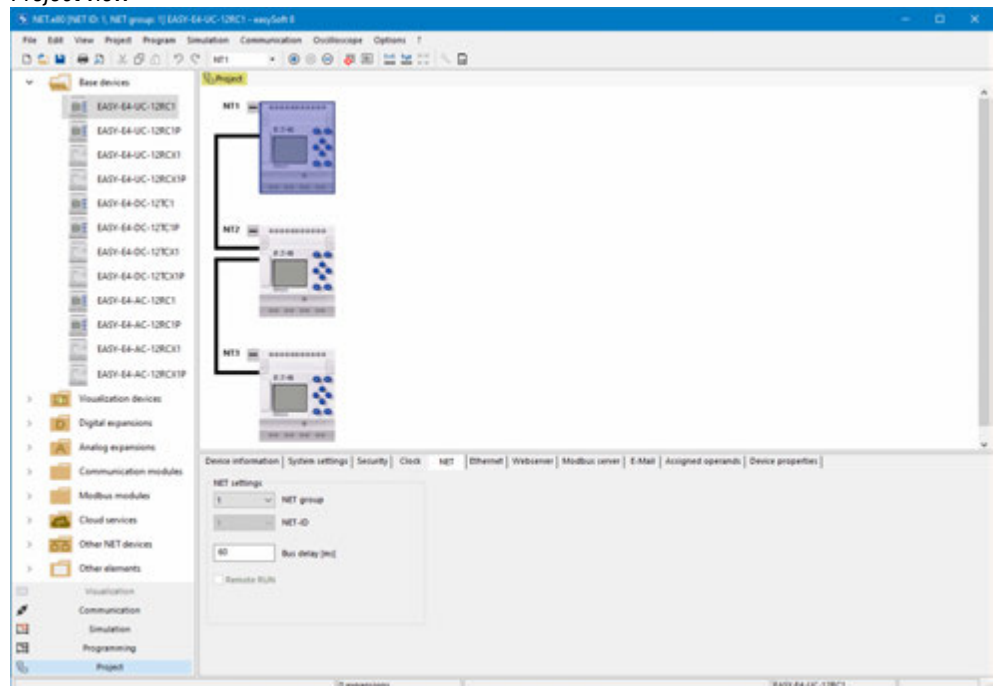


Fig. 315: NET configuration with project and program

#### Connection parameters on device

The computer can use easySoft 8 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..

## 10. Communication Connection to other devices easyE4

### 10.9 Setting up a connection to multiple devices on the NET

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 a connection to the device", page 700
- ▶ 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 634).
- ▶ 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 a connection to the device", page 700
- ▶ 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 634).
- ▶ Click on the => Device button.

Repeat these steps for every device that you want to configure in the project.

## 10. Communication Connection to other devices easyE4

### 10.10 Taking the Ethernet and NET configuration from the device

#### 10.10 Taking the Ethernet and NET configuration from the device

- ▶ Search for device → Section "Establish a connection to the device", page 700
- ▶ 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 8 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. Communication Connection to other devices easyE4

### 10.11 Secure communication with certificates

#### 10.11 Secure communication with certificates

Only available on firmware version 2.00 or higher.

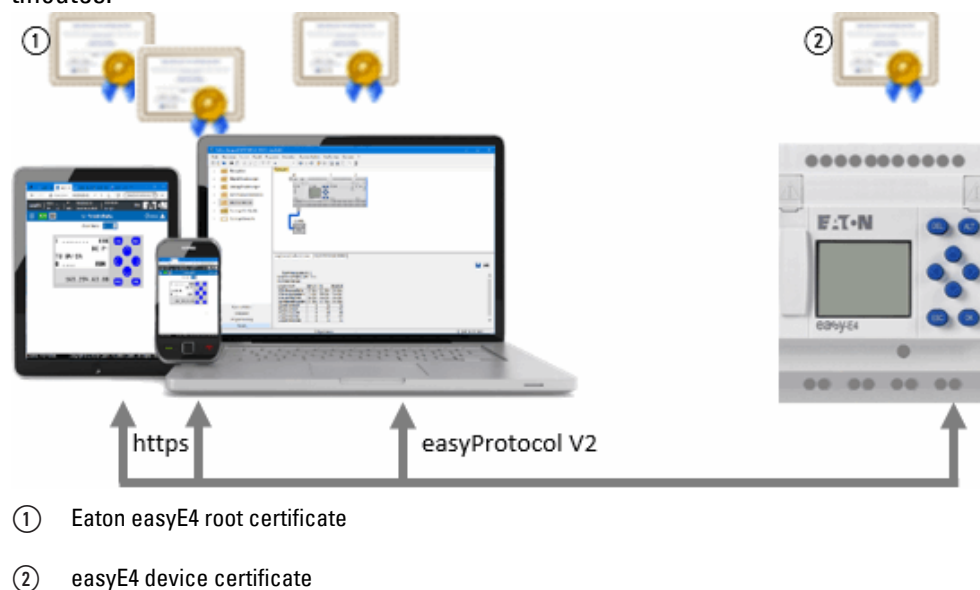
Only available on easySoft 8 or higher.

In order to ensure secure communications between the easyE4 base device and other devices such as PCs, tablets, and cell phones, you can automatically install the Eaton easyE4 root certificate on your device when installing easySoft 8 or higher. The Eaton easyE4 root certificate is provided in a reliable manner. It can be downloaded at any time from the Software Download Center and can be installed retrospectively. The easyE4 root certificate is installed once only on the PC/Tablet/Mobile.

and has a validity period of 50 years.

easyE4 base devices with firmware version 2.00 or higher are delivered with a device certificate that comes installed on the device already. easyE4 base devices will renew this device certificate automatically after a year.

As a general rule, secure communications cannot be established without valid certificates.



##### 10.11.1 What is the Eaton easyE4 root certificate for?

The Eaton easyE4 root certificate is requested as soon as the easy E4 base device's Ethernet interface is accessed externally.

If the browser is unable to find an easyE4 easyE4 root certificate, it will ask you whether you want to trust the easyE4's device certificate. If you confirm that you do, you will be able to establish a connection. Installing the easyE4 root certificate

## 10. Communication Connection to other devices easyE4

### 10.11 Secure communication with certificates

makes it possible to avoid this recurring confirmation prompt while still making sure to establish secure communications.

#### 10.11.2 When is the Eaton easyE4 root certificate needed?

One example is when easySoft 8 wants to establish communications with the easyE4's Ethernet port.

The Eaton easyE4 root certificate is also requested the moment a browser wants to access the easyE4 base device's web server.

The same applies when establishing a connection for the JSON API. If the certificate is not valid, a connection will not be established.

Purpose of communications	Ethernet interface with the following higher protocols	Certificate request
easyE4 Ethernet interface	easyProtocol V1	—
	easyProtocol V2 SSL/TLS	√
	easyProtocol V2 (not encrypted)	—
easyE4 as Webserver	http	—
	https	√
JSON:API	http	—
	https	√

The Eaton easyE4 root certificate is not requested for the following types of communication:

- Modbus TCP connection
- NET
- easyProtocol V1
- easyProtocol V2 without TLS (not encrypted)

If the easyE4 base device easyProtocol is in its state of delivery, easyProtocol V1 can be used to communicate through port 10001.



Make sure, especially when running easySoft 8 on Windows 7, that the ☒ easyProtocol V1 allowed (not encrypted, TCP port 10001) option under the *Project Settings/"Ethernet"* tab is enabled before downloading the first project. Please note that the aforementioned state of delivery will cease to exist the moment the first project is loaded onto the easyE4 base device.

#### 10.11.3 What should I do if I am unable to establish a connection due to a certificate error?

There are several potential sources of error that can result in a secure connection not being established even if the Eaton easyE4 root certificate appears to have been installed correctly.

## 10. Communication Connection to other devices easyE4

### 10.11 Secure communication with certificates

The following messages can appear:

- Unable to establish an encrypted connection.  
Make sure that easySoft 8 is attempting to connect to the right easyE4 base device; check the device's IP address and domain (if any).
- Unable to check the server's domain name or IP address with the server certificate.  
Make sure that easySoft 8 is attempting to connect to the right easyE4 base device;  
check the device's IP address and domain (if any).
- Certificate error: The certificate cannot be used for communications.  
The easyE4 device certificate or the Eaton easyE4 root certificate may be locked or may not be released for communications.
- Certificate error: The issuer certificate is not valid and/or is unknown!,  
The Eaton easyE4 Root certificate is probably not installed successfully,  
see → "How can I check to make sure that the Eaton easyE4 root certificate has been successfully installed on my PC/tablet/cell phone?", page 717
- The certificate has expired!  
Check the device time of the easyE4. The device time may be incorrect, causing a TLS certificate (level 4) that has already elapsed or that is not yet valid to be generated.

#### 10.11.4 How does a certificate request work?

Every time a connection is established between your PC/tablet/cell phone and the easyE4 device, the device certificate is checked in order to make sure that the PC/tablet/cell phone is actually connecting to the easyE4 and not to an unauthorized device. The easyE4 certificate chain consists of a total of four certificates.

When the connection is being established, the authenticity of the requested easyE4 is verified with the certificate chain.

## 10. Communication Connection to other devices easyE4

### 10.11 Secure communication with certificates



Fig. 316: easyE4 certificate chain

#### 10.11.5 Installing the Eaton easyE4 root certificate automatically at the same time as easySoft 8

To do this, make sure that the following option is enabled when installing easySoft 8:

- ☒ Installation of the Eaton certificate "easy Root CA V1.0"



## 10. Communication Connection to other devices easyE4

### 10.11 Secure communication with certificates

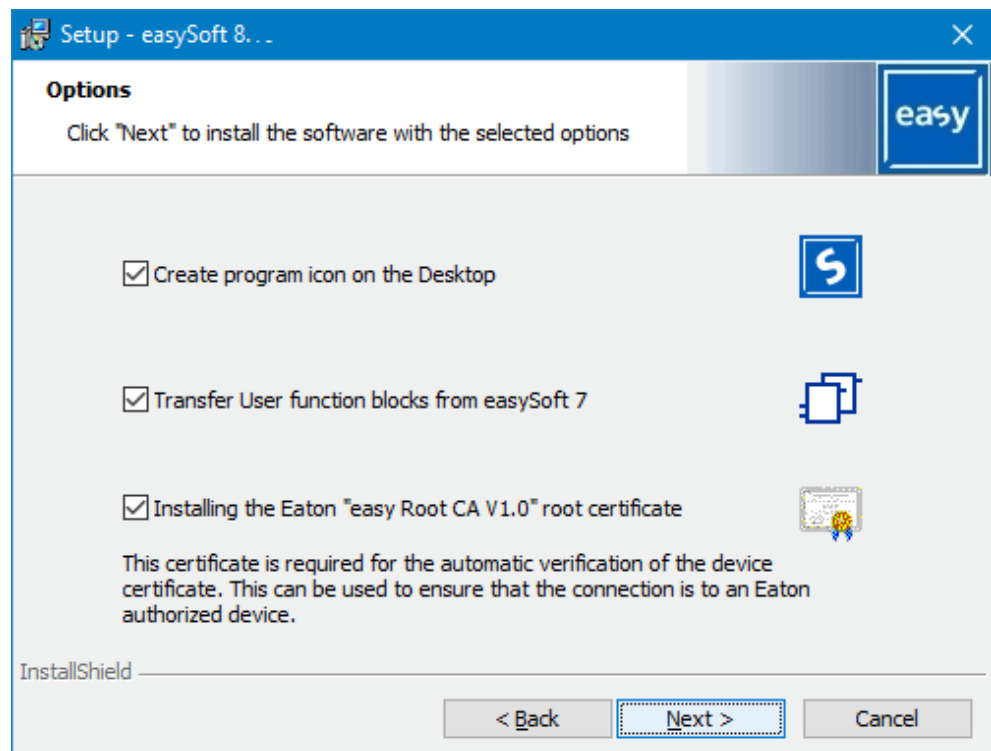


Fig. 317: Installing easySoft 8 with the Eaton easyE4 root certificate installation option enabled

The file `easyRootCertV1.crt` is stored in directory `C:\Program Files (x86)\Common Files\Eaton\easyRootCA` and the certificate `easy Root CA V1.0` is installed.

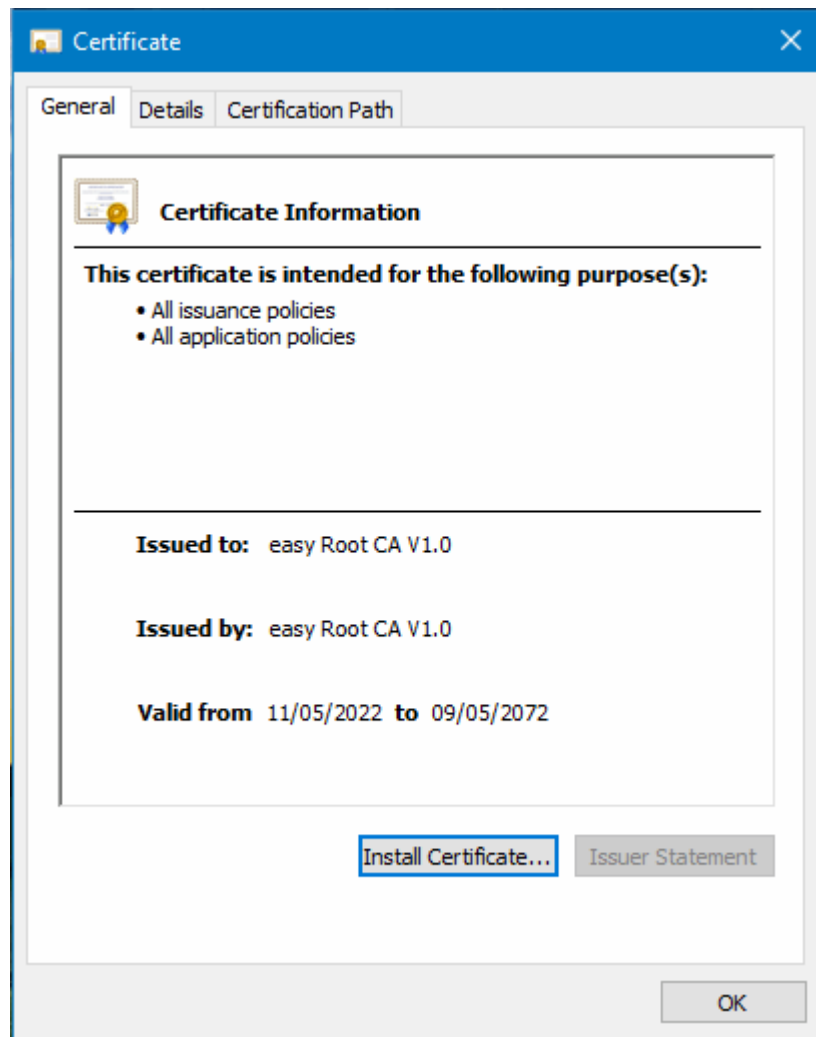
#### 10.11.6 Installing the Eaton easyE4 root certificate separately

You can install the Eaton easyE4 root certificate independently from or after installing easySoft 8 (regardless, you can establish a connection that does not require a certificate check at any time). To install the Eaton easyE4 root certificate, follow the steps below:

- ▶ Download the installation package for the Eaton easyE4 root certificate. To do so, click on [Eaton.com/easyE4RootZertifikat](https://www.eaton.com/easyE4RootZertifikat). The installation package is a ZIP file that contains both files, the "easyRootCertVxx.crt" and an installation manual in pdf format.
- ▶ Double-click the "easyRootCertVxx.crt" file to run it. The Eaton easyE4 root certificate is offered to you for installation.

## 10. Communication Connection to other devices easyE4

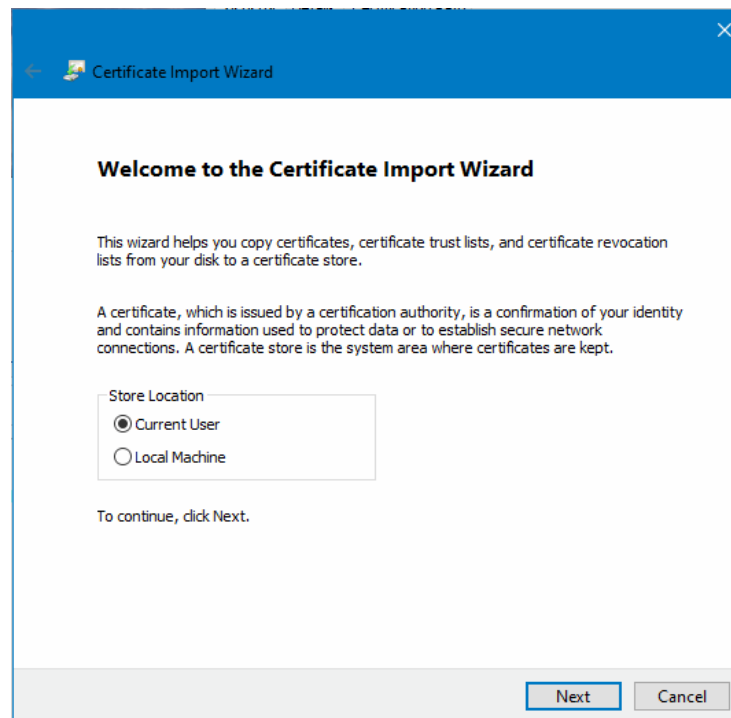
### 10.11 Secure communication with certificates



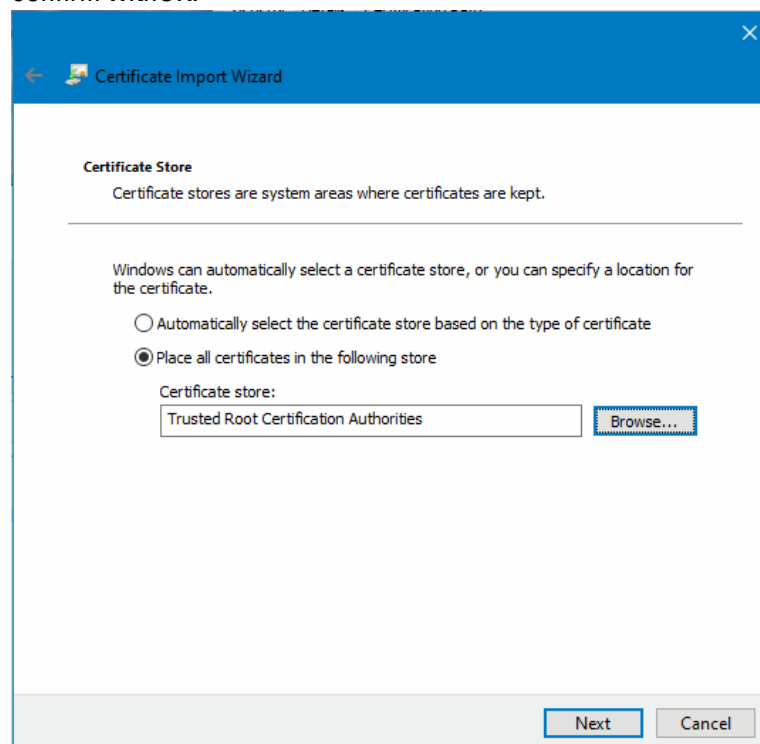
► Click on the Install Certificate.... button

## 10. Communication Connection to other devices easyE4

### 10.11 Secure communication with certificates



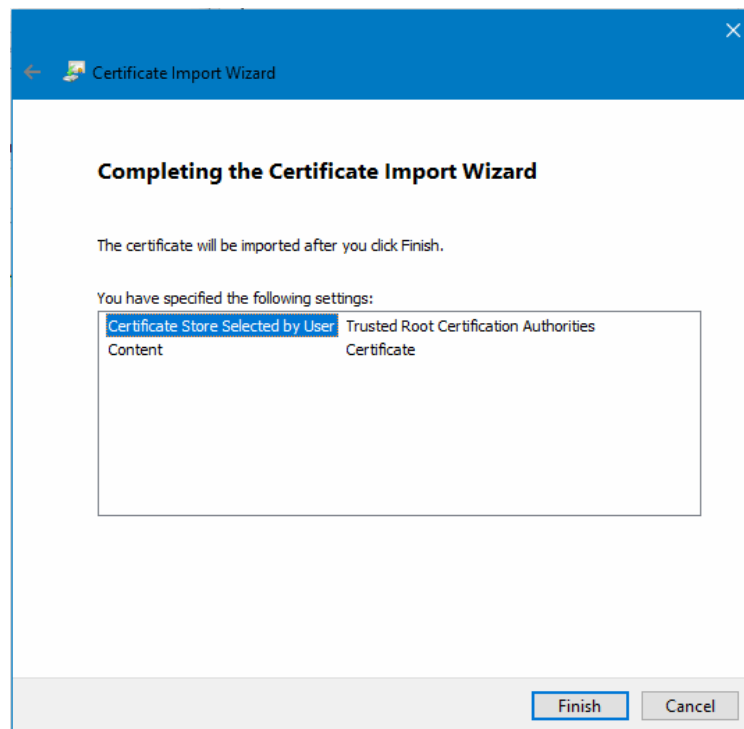
- ▶ Leave "Current User" selected and click on Next.
- ▶ Select the "Place all certificates in the following store" option
- ▶ Click on Browse....
- ▶ Select "Trusted Root Certification Authorities" in the dialog box that appears and confirm with OK.



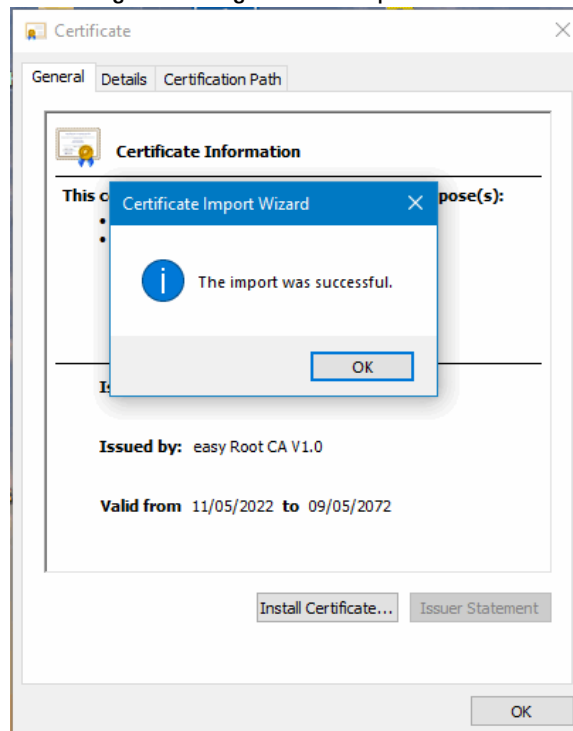
## 10. Communication Connection to other devices easyE4

### 10.11 Secure communication with certificates

- Click Continue.



- Click on Finish.  
A message showing that the import was successful will appear.



- Confirm by clicking on OK
- Click on OK on the "Certificate" dialog box.

## 10. Communication Connection to other devices easyE4

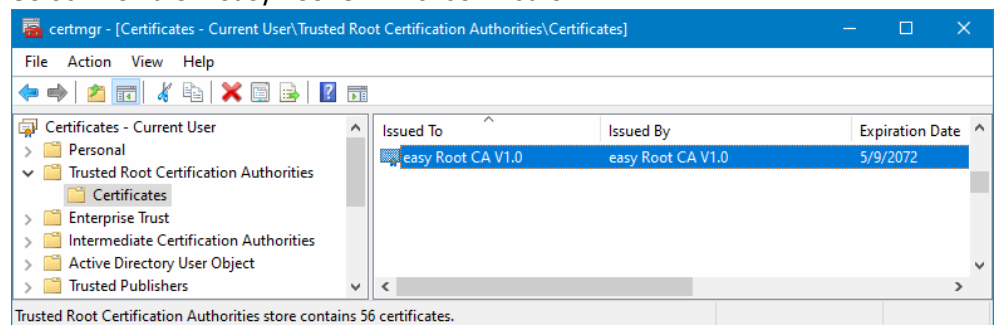
### 10.11 Secure communication with certificates

- ▶ Restart the application or Web browser.

The Eaton easyE4 root certificate has now been successfully installed.

#### 10.11.7 How can I check to make sure that the Eaton easyE4 root certificate has been successfully installed on my PC/tablet/cell phone?

- ▶ Open the command prompt by entering the `cmd` command into the Windows search.
- ▶ Use the `certmgr.msc` command to access the certificates for your device.
- ▶ Open the *Certificates – Current User / Trusted Root Certification Authorities / Certificates* folder
- ▶ Select the Eaton "easy Root CA V1.0" certificate.

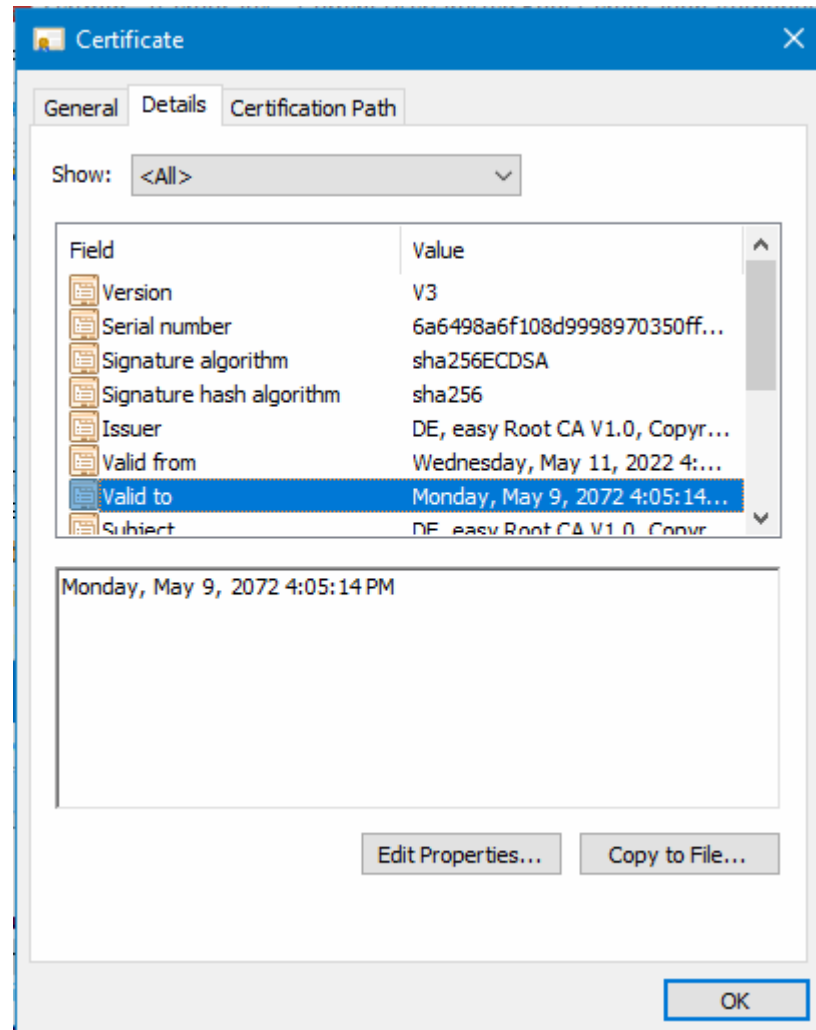


- ▶ Double-click on the <easy Root CA V1.0> certificate and switch to the Details tab.

## 10. Communication Connection to other devices easyE4

### 10.11 Secure communication with certificates


You can check whether the certificate is valid there.



You can also use your browser to check whether the certificate was installed successfully.

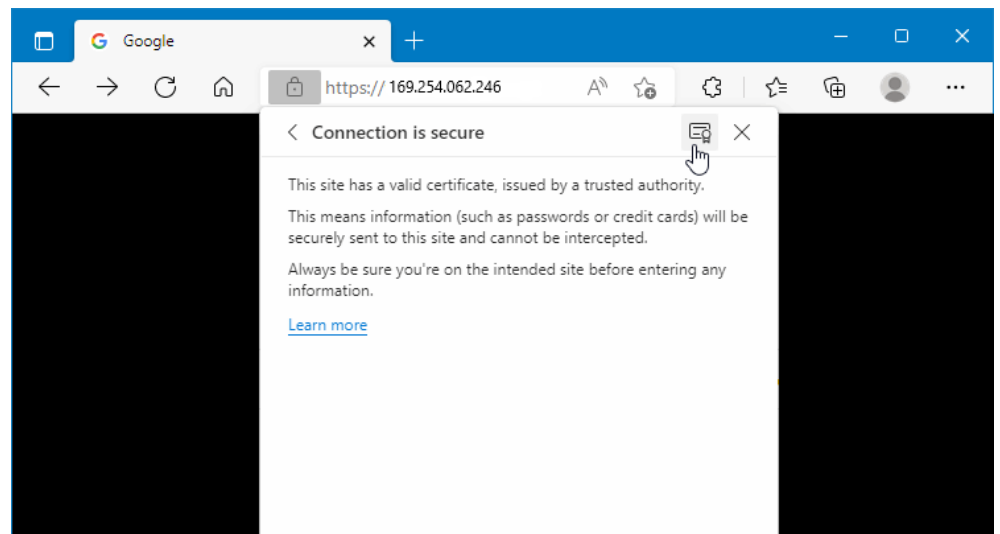
- Open the browser.
- Establish a secure connection to the easyE4 base device by entering HTTPS and the device's IP address, e.g., <https://169.254.63.80>.

If you are using Edge as a Web browser:

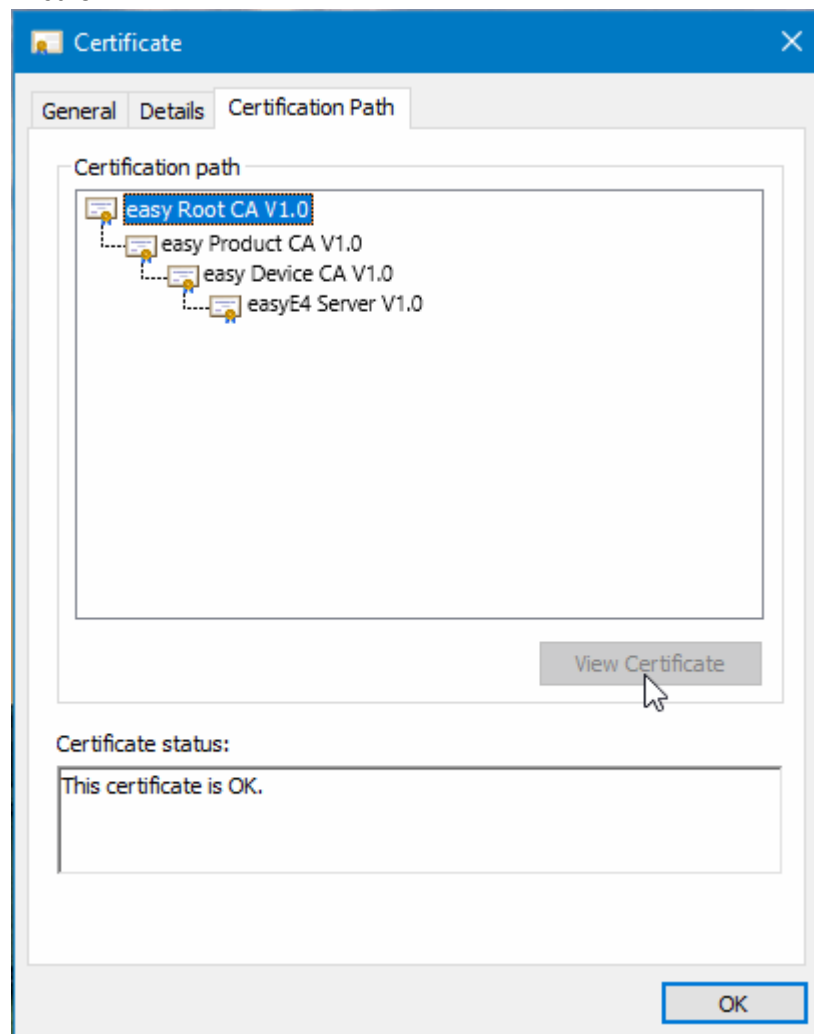
- Click on the lock in the browser's address bar > Connection is secure and then on .

## 10. Communication Connection to other devices easyE4

### 10.11 Secure communication with certificates



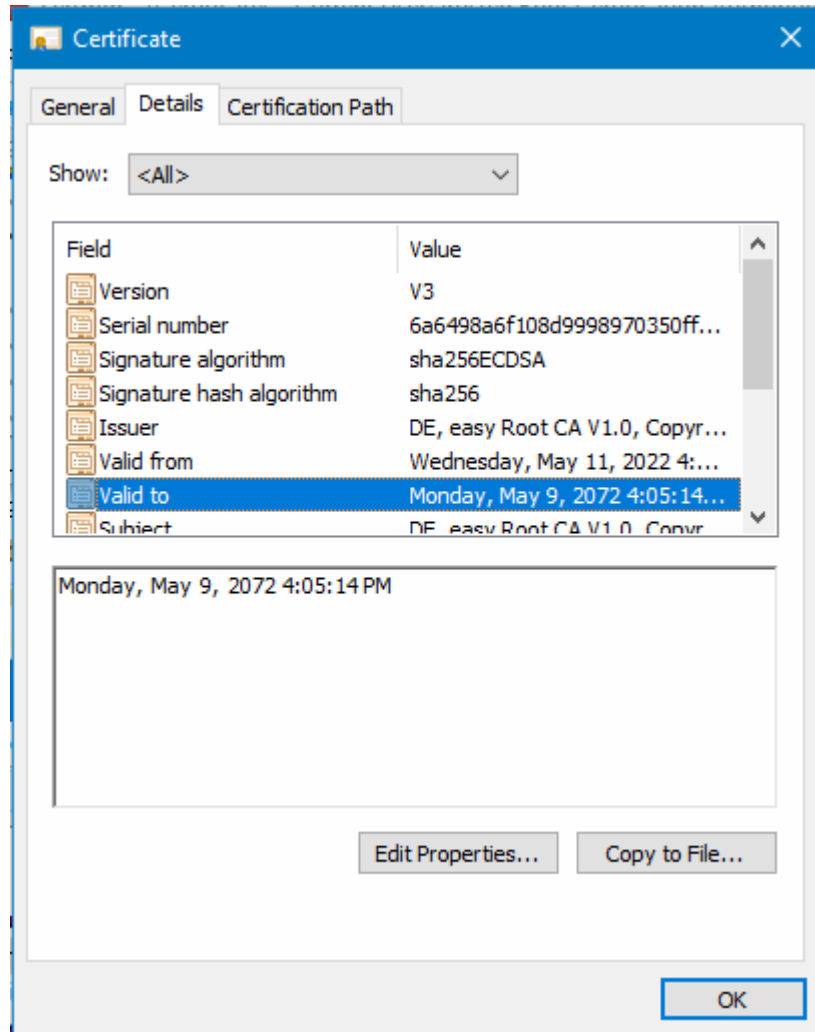
- Select one of the device certificates or remain on the "easy Root CA V1.0" certificate



- Switch to the Details tab.

## 10. Communication Connection to other devices easyE4

### 10.11 Secure communication with certificates





10. Communication Connection to other devices easyE4

10.12 Setting up a NET group

10.12 Setting up a NET group

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.

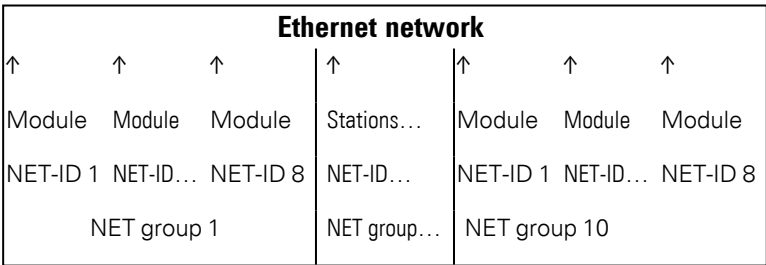
The name "NET" refers to Ethernet/UDP-based communications between the easyE4 devices. It was designed specifically with the needs of straightforward transfers between easyE4 devices in mind. Within a NET, every device has read access to the NET operands of any other device in its group. In addition 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.



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.

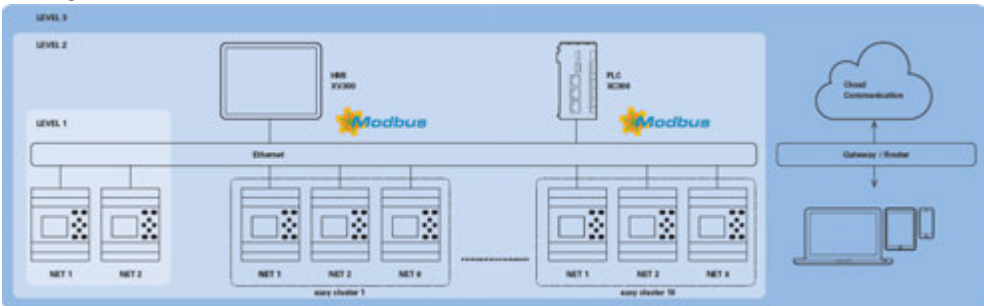


Fig. 318: NET diagram

## 10. Communication Connection to other devices easyE4

### 10.12 Setting up a NET group

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.

In order to be able to run a NET group, there must be an Ethernet connection between the devices or device and PC.

#### 10.12.1 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 "Time and Date setting", page 659

#### 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 .e80 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. Communication Connection to other devices easyE4

### 10.12 Setting up a NET group

#### 10.12.2 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  
                 Parameter  
                 NET-ID 1  
                 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

## 10. Communication Connection to other devices easyE4

### 10.12 Setting up a NET group

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. Communication Connection to other devices easyE4

### 10.12 Setting up a NET group

#### 10.12.3 NET settings

##### Prerequisites

The Ethernet settings must have already been configured.

In offline mode, setting up the configuration in easySoft 8 under the Ethernet tab is enough, → Section "Establishing an Ethernet connection and transferring a program or visualization project", page 117

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*

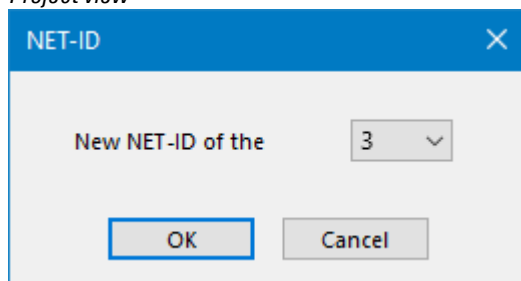


Fig. 319: NET-ID dialog box used to assign a NET ID when adding a new base device



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> option 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.

## 10. Communication Connection to other devices easyE4

### 10.12 Setting up a NET group

#### Project view

Fig. 320: 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), 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.

## 10. Communication Connection to other devices easyE4

### 10.12 Setting up a 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 that 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 8, 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 8 to change the bus delay point by point.

→ Section "GT - Get values from NET", page 460

→ Section "PT - Put values to NET", page 464

→ Section "SC - Synchronizing clock via NET", page 468

→ Section "Establishing an Ethernet connection and transferring a program or visualization project", page 117

## 10.13 Setting up a Web Server

Only possible with easySoft 8.

The easyE4's encrypted Web Server makes it possible to quickly and easily access data and parameters on any mobile device, including smartphones and tablets. This integrated Web Server is used for visualization purposes and can deliver automatic notifications and be used for control purposes. The corresponding connection can be encrypted with an SSL/TLS certificate.

The Web Server is a subcomponent of the easyE4 operating system that is responsible for the Web-Visu functionality.

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, the Web Client has been developed based on responsive design principles.

The device status can be ready directly on the display on EASY-E4-...-12...C1(P) devices – please refer to → Section "Status display on control relay easyE4 with display and keypad", page 111.

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 8 by opening the Project view and then going to the Webserver tab.

### 10.13.1 Webserver tab

For easySoft 8 and higher, the settings for web server communications are configured under the Ethernet tab.

The screenshot displays the 'Webserver' tab within the 'Project view' of easySoft 8. The interface is divided into several sections:

- Web server configuration:**
  - ☒ Web server enabled
  - ☒ Always enabled
  - ☐ Activation by program
  - Port and SSL/TLS settings can be changed on property tab "Ethernet".
  - ☐ CORS active
  - ☒ Parameter list active
  - ☒ Web-Visu active
  - ☐ Load Web-Visu from card
- Enable marker (write):**
  - From: [dropdown] To: [dropdown]
- Enable NET marker (write):**
  - From: [dropdown] To: [dropdown]
- ☐ Enable Read I/Os

On the right side, there is an **Access control** section:

- ☐ Anonymous read access allowed
- Define passwords and user names:**
  - User name: [text input]
  - Rights: [dropdown menu]
- Mode:** [checkbox]
- Clock:** [checkbox]
- Parameters:** [checkbox]
- E-mail:** [checkbox]
- Remote display keys:** [checkbox]

Fig. 321: Project view, Webserver tab



## 10. Communication Connection to other devices easyE4

### 10.13 Setting up a Web Server

#### 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 732

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

All of the program's AL alarm function blocks will be read before the web server is started. At least one alarm function block must start the web server – otherwise, it will remain deactivated.

The → " Web server startup behavior", page 733 table describes the various possibilities regarding the web server's startup behavior.

☐ CORS active

Only available on firmware version 2.00 or higher.

Only available on easySoft 8 or higher.

Enabling this option will make it possible to access the easyE4 base device's data from other websites.


One possible use case is using the JSON:API to access the easyE4 base device's data and then publish it on a selected website.

☐ Parameter list enabled

If this option is enabled, the **Parameter list** menu option will be shown in the Web Client's left pane. In this case, a custom **parameter list** with operands can be put together in the Web Client. This will make monitoring and controlling the relevant operands significantly easier.

☐ Web-Visu enabled

If this option is enabled,

it will be possible to start the Web Editor with the **Visualization** view or with the toolbar  in the **Simulation** view or **Communication** view.

☐ Load Web-Visu from card

If this option is enabled, Web-Visu will not be stored on the easyE4 base device and must instead be manually stored on the microSD memory card in the easyE4 base device.



This option needs to be enabled if the Web-Visu's size is greater than 2 MB, as the device's memory capacity is limited to sizes smaller than this. Please note that the microSD memory card needs to remain inside the easyE4 base device at all times during ongoing operation.

Enable marker (write)

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.

Enable NET marker (write)

This is where the NET marker range for access through the Web Client is enabled. The enabled range will apply to the administrator and to all defined users equally.

copy from  up to 

☐ Enable read I/O

Enabling this option will make it possible to access the data for the easyE4 base device's input and output signals from other websites. One possible use case is using the JSON:API to access the easyE4 base device's data and then publish it on a selected website.

## 10. Communication Connection to other devices easyE4

### 10.13 Setting up a Web Server

#### 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 → "Web server passwords and user names dialog box", page 733

User Name:

If additional users have been set up in addition to the administrator, they will be shown here.

Rights:

Shows the  or  permissions for the user.

The following options are equivalent to the settings in *Project view/Security tab/Password input section*.

☒ Mode

If this option is enabled, the relevant user will be able to change the RUN/STOP operating mode for the easyE4 base device with the Web Client's menu bar (the administrator always has the required permissions for this).

☒ Clock

If this option is enabled, the device time of the device clock can be modified with the Web Client. This function may be helpful during commissioning.

However, if the

**Synchronize clock via radio (DCF77)** option is enabled in *Project view/Clock*, the device will get its device time as a client from an SNTP server or from a radio-controlled clock (DCF77).

As a result, the time modified with the Web Client will be overwritten.

☒ 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 on 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*.

☒ E-mail

If this option is enabled, the relevant user will be able to use the *Catalog settings/Email* menu option in the Web Client in order to access the EMAIL menu on the remote display and edit the **e-mail recipient groups** there. In order for this to work, the project on the device must already contain an **e-mail recipient group**.

In addition, the user can change the **mail server settings**, such as the **IP address** and **DNS name**. The corresponding changes will be written to the project on the device.

Please note that this option will always be available to admins even if not explicitly enabled.



If this option is not being displayed, check to make sure that firmware version 1.30 or higher is selected in the *Project view/System settings tab*.

## 10. Communication Connection to other devices easyE4

### 10.13 Setting up a Web Server

☒ Remote display keys

If this option is enabled, the user will be able to use the enabled P buttons for a D function block in the Web Client if the text display has been configured and other functions in the program use it for control purposes. A configured D function block will always be visible on the display if the program switches to STOP mode.  
Please note that this option will always be available to admins even if not explicitly enabled.



If this option is not being displayed, check to make sure that firmware version 1.40 or higher is selected in the *Project view/System settings tab*.

#### Administrator permissions include:

- The administrator can operate the remote display even if the ☒ Remote display keys option is disabled.
- Changing the STOP/RUN operating mode
- Writing markers, provided that they have been enabled in the Web server configuration section.
- Reading diagnostics

## 10. Communication Connection to other devices easyE4

### 10.13 Setting up a Web Server

#### 10.13.2 Configuring the web server function in easySoft 8

You can configure the web server functions you want for each device in the project in easySoft 8. 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.

##### 10.13.2.1 Setting up users

Under the tab, there is the Webserver configuration section, which can be used to enable and configure the web server functionality, as well as an Access control section that can be used to set up the access permissions for the various users.

- ▶ 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.

10. Communication Connection to other devices easyE4

10.13 Setting up a Web Server

Fig. 322: 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.

10.13.2.2 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.

10.13.2.3 Configuring the web server's startup behavior

This section explains the conditions under which the web server will be started. The corresponding options can be configured under the *Project view/Webserver tab* and in the settings for the alarm function block parameters *Programming view/Alarm function block parameters*.

Tab. 129: Web server startup behavior options

Web server startup behavior	Webserver tab	Alarm function block parameters
Will never start	<input type="checkbox"/> Web server enabled	—
Will start depending on additional options	<input checked="" type="checkbox"/> Web server enabled	—
Will start as soon as the easyE4 base device is switched on; the device's operating state is irrelevant; the program must be found on the device	<input checked="" type="radio"/> Always enabled	—

## 10. Communication Connection to other devices easyE4

### 10.13 Setting up a Web Server

Web server startup behavior	Webserver tab	Alarm function block parameters
Will never start	<input checked="" type="radio"/> Activation by program	<input type="checkbox"/> Function block release by EN is necessary <input type="checkbox"/> Web server active as long as there is a state of 1 at input EN
Will start as soon as the program starts		<input type="checkbox"/> Function block release by EN is necessary <input checked="" type="checkbox"/> Web server active as long as there is a state of 1 at input EN
Will never start		<input checked="" type="checkbox"/> Function block release by EN is necessary <input type="checkbox"/> Web server active as long as there is a state of 1 at input EN
Will start as soon as the program starts and function block input AL_EN=1		<input checked="" type="checkbox"/> Function block release by EN is necessary <input checked="" type="checkbox"/> Web server active as long as there is a state of 1 at input EN

#### 10.13.2.4 Configuring the settings in the Webserver tab

##### Web server configuration

- Please refer to the → "Web server startup behavior", page 733 table for the web server startup behavior options.
- Now select whether you want the web server to be ☒ Always enabled or
- if ☒ Activation by program should instead be required.  
Accordingly, all AL alarm function blocks in the program will be read before the web server starts.  
At least one alarm function block must start the web server – otherwise, it will remain deactivated.
- Set the HTTP port you want.  
80 will be set as the default "HTTP port". If SSL/TLS encryption is enabled, it will be set to 443 instead by default.

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

## 10. Communication Connection to other devices easyE4

### 10.13 Setting up a Web Server

- ▶ Enable this option if you want anonymous read access to be 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.
- ▶ 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 736
- Section "AL - Alarm function block", page 472

## 10.14 Web Client

The Web Client can only be started if the web server function has already been configured under the *Webserver tab* and the password for the administrator or for a different user is known. The following Web browser are supported:

- Internet Explorer 11 or higher, Chrome, Safari, MS Edge, Firefox.
- Chrome,
- Safari,
- MS Edge,
- Opera,
- Brave,
- Firefox.

We recommend using Chrome, since the Web Client has been optimized for it.

The Web Client has been developed based on responsive design principles and is optimized for display on all devices, including monitors, laptops, tablets, and even smartphones.



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



## 10. Communication Connection to other devices easyE4

### 10.14 Web Client

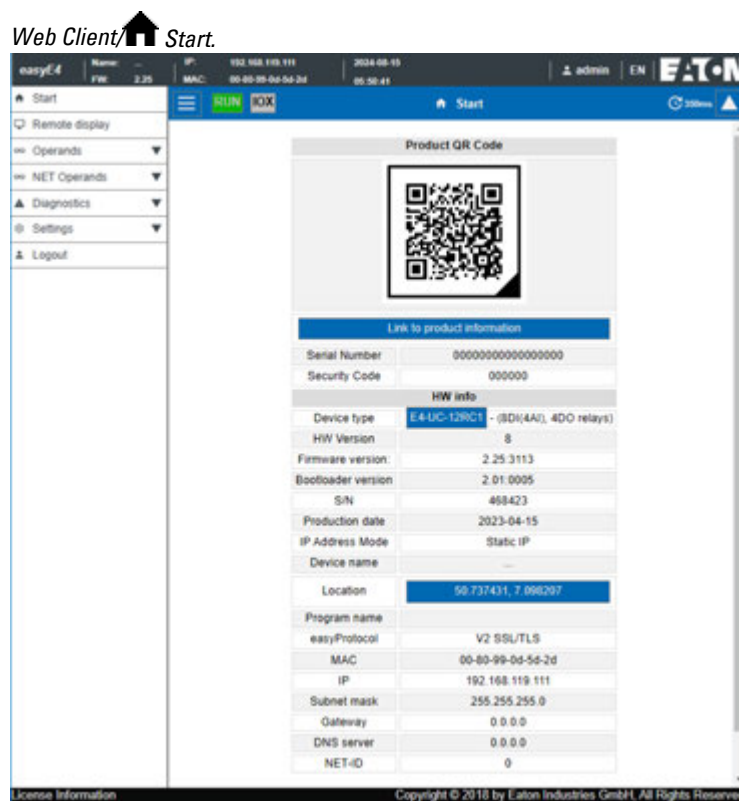


Fig. 323: Web Client, home page

Depending on the protocol being used, we recommend limiting the number of client programs accessing an easyE4 base device simultaneously:

- https: 2 Client programs
- http:  $\leq 4$  Client programs

"Client programs" here refers to both Web Client and the JSON:API. If a larger number of client programs accesses the base device simultaneously, this may result in longer wait times for the display to be updated in the Web Client.

#### 10.14.1 Web Client start

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://192.119.153>.  
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:

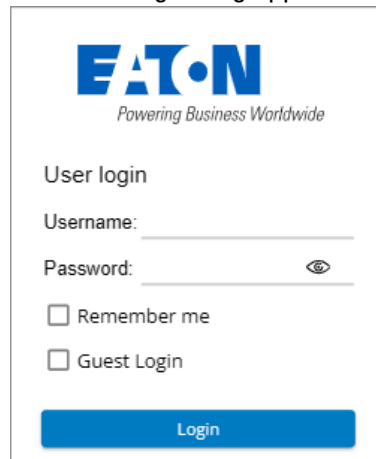
The image shows a web client login dialog box. At the top is the Eaton logo with the tagline "Powering Business Worldwide". Below the logo, the text "User login" is displayed. There are two input fields: "Username:" and "Password:". The password field has an eye icon to its right. Below the input fields are two checkboxes: "Remember me" and "Guest Login". At the bottom is a blue button labeled "Login".

Fig. 324: Web Client login dialog box



The Web Server muss im Programm auf der easyE4 aktiv sein, sonst erscheint die Seite mit den Lizenzen.

- ▶ 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 configuring the web server function into the dialog box.
- ▶ Confirm by clicking on the Login button.
- ▶ If you want to log in as a guest, enable the Guest login option.  
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.

## 10. Communication Connection to other devices easyE4

### 10.14 Web Client

#### Stay logged in

If you enable this option, the user will remain logged in until the corresponding browser data is cleared and the Web Client login dialog box will not appear again until then.

#### Log on as guest

This requires that the option Anonymous read access permitted is enabled in the \*.e80 project 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 will start and you will only have read access to the easyE4 base device.



Alternatively, the Web Client can also be accessed through Web-Visu.

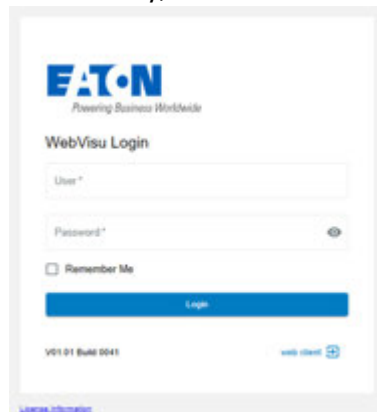


Fig. 325: Web Client login dialog box

### 10.14.2 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 clientWeb 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 have been set up in easySoft 8 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 → "Deactivate automatic scrolling to input fields", page 756 as well.

#### 10.14.2.1 Menu bar Menu bar

The menu bar contains non-editable and editable information. Editable information can be edited in easySoft 8 and, depending on the access permissions assigned in easySoft 8, in the Web Client and on the device as well. In the following section, the contents of the menu bar are explained and their editing options are indicated:







Menu bar 1	Meaning	easySoft 8	Web Client	Device
easyE4 (NT1)	Device (station)	x	—	—
Name: <aws-tecdoc>	Device name	x	x	—
IP: 192.168.119.153	The device's IP address	x	x	x
2024-05-16	Current device date	x	x	x
FW: 2.25	Firmware version of the device	—	—	—
MAC: 00-80-99-0d-bf-1f	The device's MAC address	—	—	—
15:45:09	Current device time	x	x	x
 admin	Display of the logged on user	—	x	—
DE	Used to select the language for the Web Client (EN, for example). There are thirteen languages	—	x	—

## 10. Communication Connection to other devices easyE4

### 10.14 Web Client

Menu bar 1	Meaning	easySoft 8	Web Client	Device
	available, including DE, EN, IT, ES, PL, and FR.			
– Information that cannot be edited				









The language selection for the Web Client may be different from the language selection in the device. Since the language selection is stored exclusively in the browser, each Web Client can show the device contents in a different language.

Menu bar 2	Meaning	easySoft 8	Web Client	easyE4
	Show/hide catalog	–	x	–
	Button used to select the easyE4 operating state : green RUN, red STOP	x	x	x
	Displays the status of the easyConnect bus (IO eXtension) IOX - highlighted in gray: No expansion devices are connected or fault at the easyConnect bus. Possible causes: <ul style="list-style-type: none"> <li>• Configuration not OK</li> <li>• Expansion device defective</li> <li>• Expansion device has no supply voltage</li> <li>• Communication error to an expansion device</li> </ul> IOX - highlighted in green: easyConnect bus in operation	–	–	–
 Start	Show selection in the catalog	–	x	–
	Cycle timeWeb Client	–	x	–
	Show/hide menu bar	–	x	–

## 10. Communication Connection to other devices easyE4

### 10.14 Web Client

#### 10.14.2.2 List

Menu bar 2	Meaning
 Start	Web Client start menu with the most important information regarding the connected device – please refer to <a href="#">EPAS-Code</a> .
 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 8 and, if necessary, read permissions for I/O must be assigned as well – please refer to → "Enable marker (write)", page 729 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 8 – please refer to → "Enable marker (write)", page 729 as well. 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 730
 Parameter list	Option: The *.e80 program on the easyE4 must allow access in order to display this menu item. To allow access, enable the "Parameter list enabled" option in the <i>Project view/Web-server tab</i> or in the Web Client in <i>Settings/Web Client/Separate operands</i> . The user can put together a list of operands that they would like to monitor and/or edit.
 Diagnostics	Shows the diagnostic messages that are currently present – please refer to → "Operating system diagnostic messages", page 684 as well.
 settings	Can be used to configure the <b>General settings</b> for the device, <b>Network settings</b> , <b>Email settings</b> and Settings for the Web Client.
 logout	Logs off a logged on user.

## 10. Communication Connection to other devices easyE4

### 10.14 Web Client

#### 10.14.3 Updating operands

The Web Client scans all the 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 memory area. The operands shown in the Web Client are no older than one second.



After the files have been displayed for a certain period of time, the loading circuit appears.

Depending on the protocol being used, we recommend limiting the number of client programs accessing an easyE4 base device simultaneously:

- https: 2 Client programs
- http: ≤ 4 Client programs

"Client programs" here refers to both Web Client and the JSON:API. If a larger number of client programs accesses the base device simultaneously, this may result in longer wait times for the display to be updated in the Web Client.

##### 10.14.3.1 Updating the 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 will be started as a Web Client.

#### 10.14.4 Display

The keypad can be operated on the Web Client display exactly in the same way as on the device itself. It is advisable to switch to the special menu using key combination **Alt+Shift** instead of the usual operator action on the device with the **Alt** key. Alternatively, you can control the keypad with mouse clicks.

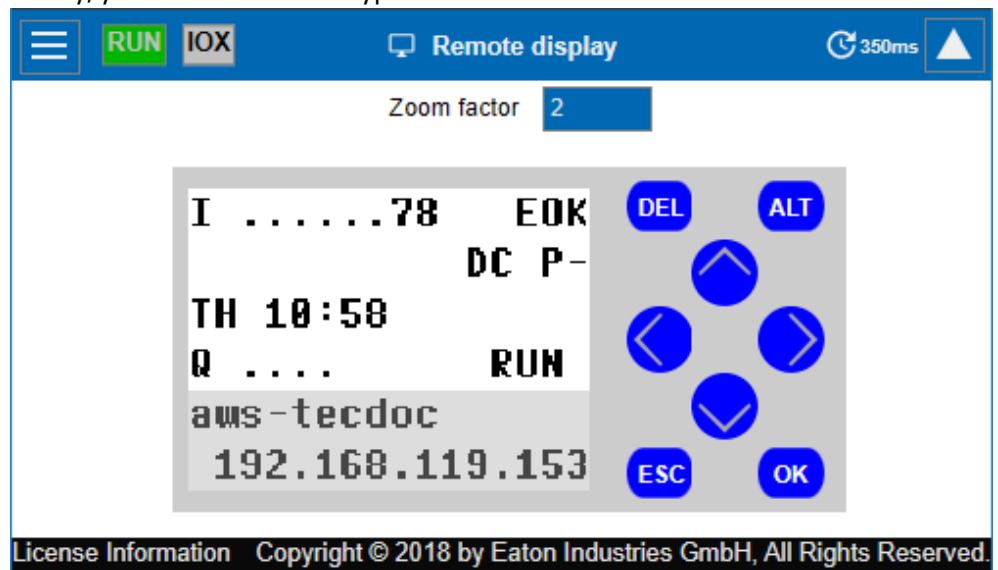


Fig. 326: Web Client, Display

##### 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.



10. Communication Connection to other devices easyE4

10.14 Web Client

10.14.5 Operands

In the operating range, operands show the states of the local bit and value operands of the device..

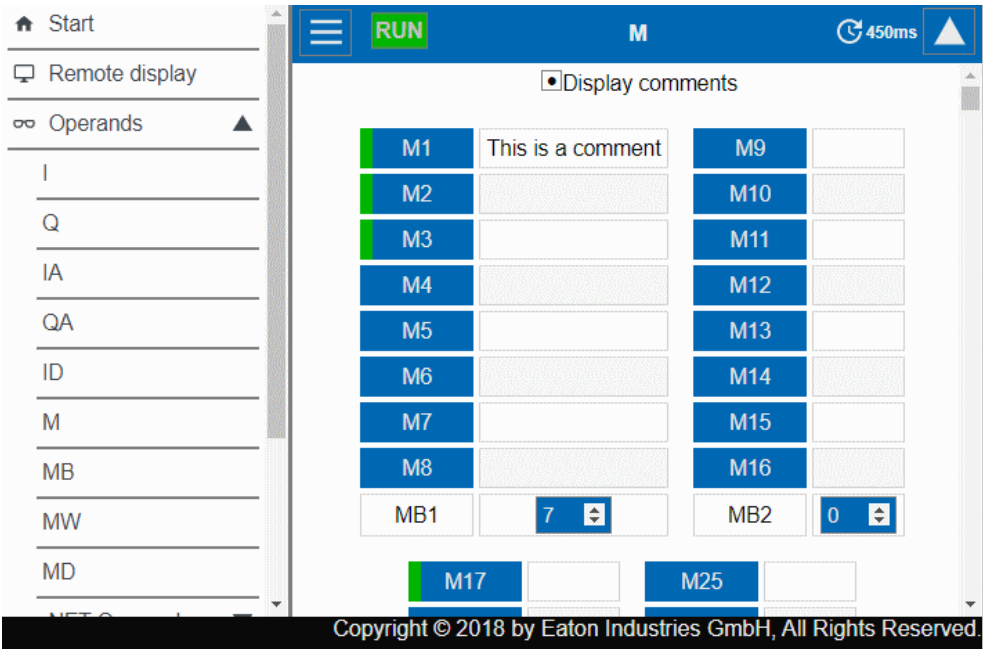



Fig. 327: Web Client, Operands

#### 10.14.6 NET Operands

 In the operating range, NET operands display the states of the local NET bit and value operands of the device or the NET bit and value operands of the other NET stations..

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 to the NET operands of the local device. The NET operands of other NET subscribers are read-only.

You can click on the NETWeb Client 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".

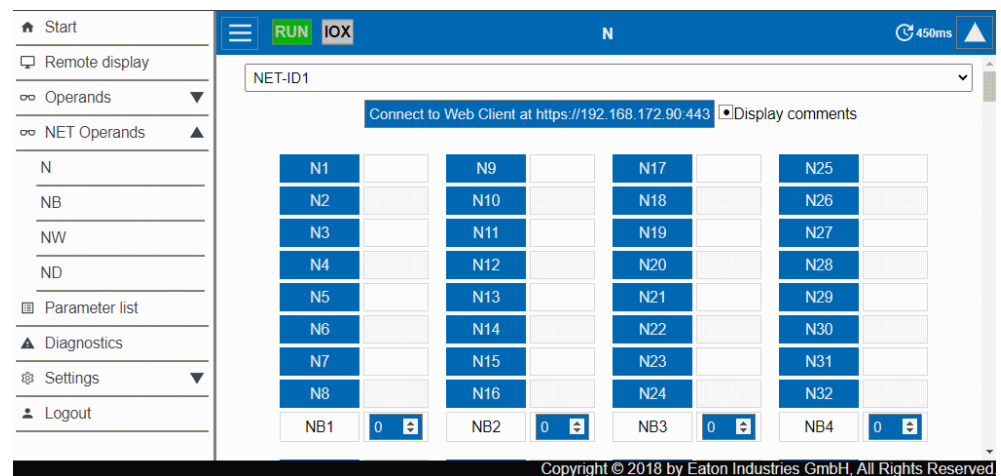


Fig. 328: Web Client, NET operands

## 10. Communication Connection to other devices easyE4

### 10.14 Web Client

#### 10.14.7 Parameter list

The \*.e80 program on the easyE4 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 729 as well) or enable it in *Catalog settings/Web client/Separate operands* in the Web Client (please refer to → "Parameter List", page 756 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, easyE4 control relay I/O expansion operands, NET operands, and function block operands. UF user function blocks are excluded. The parameter list is stored in the browser's local memory and not on the EASY-E4-.... The parameter list will be retained when opening 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.

Red borders around function block inputs or outputs:

0

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".

## 10. Communication Connection to other devices easyE4

### 10.14 Web Client

The screenshot shows the Web Client interface for the Parameter list. The sidebar on the left contains navigation options: Start, Remote display, Operands, NET Operands, Parameter list (selected), Diagnostics, and Settings. The main area displays a table titled 'Save FB inputs' with the following columns: Name, Value, Comment, Note, and Write Access. The table lists various function blocks (FB) and their current values. A red box highlights the value '0' for the parameter FB.A01.F1.

Name	Value	Comment	Note	Write Access
FB.C01.QV	0		output counter relay	<input type="checkbox"/>
FB.C01.CY	C01CY		output carry counter relay	<input type="checkbox"/>
FB.C01.ZE	C01ZE		output zero counter relay	<input type="checkbox"/>
FB.T01.I1	00:00:00.005		timing relay 1, flashing on	<input type="checkbox"/>
FB.T01.I2	00:00:00.005		timing relay 1, flashing off	<input type="checkbox"/>
FB.T01.EN	T01EN		enable timing relay 1	<input type="checkbox"/>
2N3	N3		device2:N3	<input type="checkbox"/>
I1	I1		input ventil V211	<input type="checkbox"/>
FB.A01.F1	0		not used in program - analo	<input type="checkbox"/>

https://169.254.93.45/int/index.html# Copyright © 2018 by Eaton Industries GmbH, All Rights Reserved.

Fig. 329: Web Client, Separate operands

Column	Meaning
Name	<p>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 V8 that contain the entered text anywhere in the operand or in the comment.</p> <p>The proposed text can be assumed by performing the following steps:</p> <ul style="list-style-type: none"> <li>Use the arrow keys ↑ and ↓ to navigate through the proposed entries</li> <li>Make a selection with a click of the mouse and hit <b>Enter</b>.</li> </ul>
Value	<p>Depending on the device's operating status, the states of the selected operands are displayed in the workspace.</p> <p>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., <b>T01EN</b>. For a status of 0, no bar will be shown.</p> <p>In the case of analog operands, the operand's current value will be shown. 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., <b>FB.A01.F1</b></p> <p><b>0</b></p>
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	<p>This option is available exclusively to the administrator.</p> <p>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.</p>

10. Communication Connection to other devices easyE4

10.14 Web Client

Manage list

Pick File

None selected

Export list

Manage list ▲

Column	Meaning
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 the changes for all web services at function block inputs since the last time the easyE4 base device was started being retentively copied to the device.

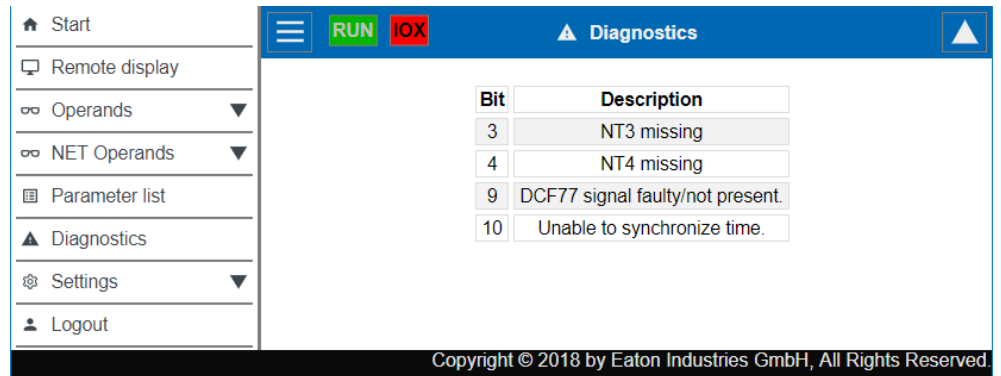
Only analog constant and time constant values will be copied. Within this context, "changes for all web services" means changes carried out with Web Clients and JSON:API.

➔ Changes that come from other Web Clients and that have been made with JSON:API will also be copied even if made across several sessions.

This means that the modified constants will be immediately available on the device and will be retained the next time the device is started.

#### 10.14.8 Diagnostics

Diagnostics displays which diagnostic operands are set and their meaning. In the Web Client, the values listed in the Bit column correspond to the diagnostic operands. For more information on diagnostic options, see also → "Operating system diagnostic messages", page 684.



Bit	Description
3	NT3 missing
4	NT4 missing
9	DCF77 signal faulty/not present.
10	Unable to synchronize time.

Copyright © 2018 by Eaton Industries GmbH, All Rights Reserved.

Fig. 330: Web Client, Diagnostics

## 10. Communication Connection to other devices easyE4

### 10.14 Web Client

#### 10.14.9 Settings

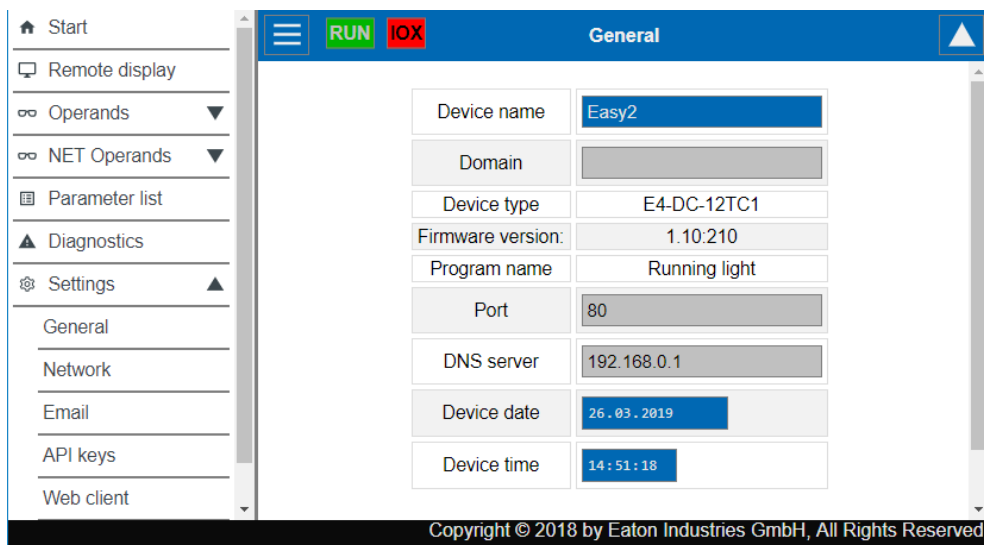
The fields highlighted in blue can be edited: Only the admin can modify the settings. The following settings can be displayed:

- General settings
- Network settings
- E-mail settings
- API key (is only displayed for the admin)
- Web client (is only displayed for the admin)

##### 10.14.9.1 General settings

The admin can modify the device name, device date and device time. Any changes made in the Web Client must be confirmed in the prompt that appears. The modified data will not be transferred to the device until then. The standard user has read-only access to the General settings.

*Web Client/Settings/General*



General	
Device name	Easy2
Domain	
Device type	E4-DC-12TC1
Firmware version:	1.10.210
Program name	Running light
Port	80
DNS server	192.168.0.1
Device date	26.03.2019
Device time	14:51:18

Copyright © 2018 by Eaton Industries GmbH, All Rights Reserved.

Fig. 331: Web Client, General settings

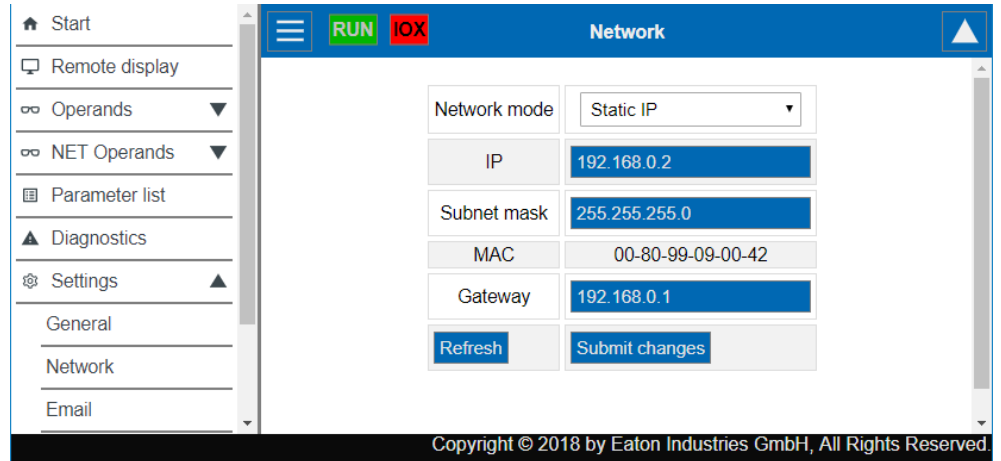
##### 10.14.9.2 Network settings

The admin can modify the network settings and make changes to the IP address, sub-net mask, and the gateway's IP address. Any changes made in the Web Client must be confirmed in the prompt that appears. The modified data will not be transferred to the device until then. The standard user has read-only access to the Network settings.

## 10. Communication Connection to other devices easyE4

### 10.14 Web Client

Web Client/Settings/Network



Network mode	Static IP
IP	192.168.0.2
Subnet mask	255.255.255.0
MAC	00-80-99-09-00-42
Gateway	192.168.0.1
Refresh	Submit changes

Copyright © 2018 by Eaton Industries GmbH, All Rights Reserved.

Fig. 332: Web Client, Network settings

#### 10.14.9.3 E-mail settings

The admin can modify the mail server's email settings. These are the same parameters that are configured in *easySoft 8 Project view/Email tab/Mail server settings area*.. They include the IP address or DNS name of the mail server, mail server domains, the mail server connection encryption, the login name or user and login password of the mail server user, and the mail server port. Any changes made in the Web Client must be confirmed in the prompt that appears. The modified data will then be transferred to the device. The standard user has read-only access to the email settings.



10. Communication Connection to other devices easyE4

10.14 Web Client

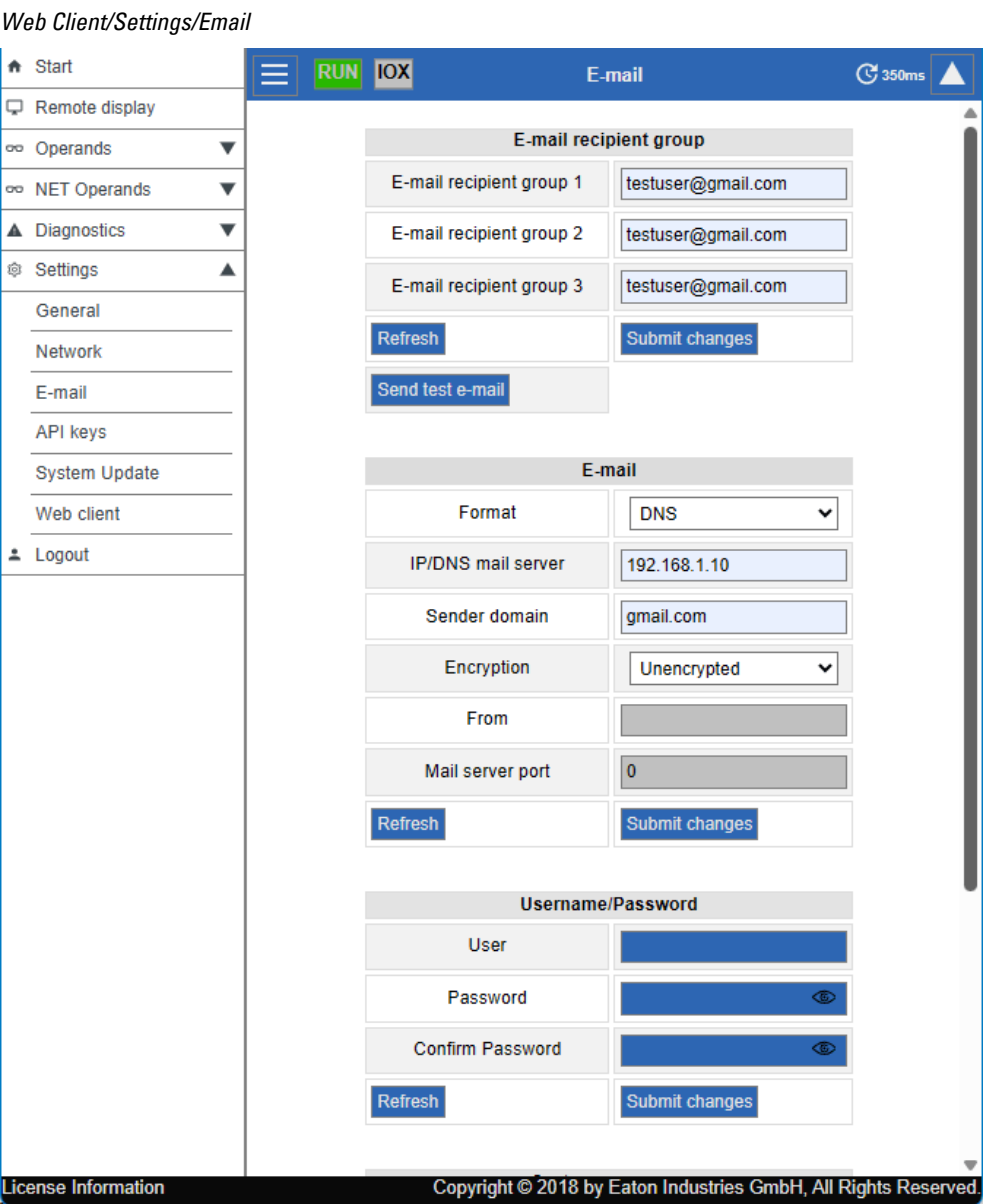


Fig. 333: Web Client, E-mail settings

10.14.9.4 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 of using a JSON:API application programming interface. Any program can access and edit the easyE4 data from this interface, for example, a program from a piece of enterprise software. easySoft 8 is not required. The API can be used in all high-level languages that provide a library for HTTP GetRequests, such as JavaScript, Python, VBA, C++.

## 10. Communication Connection to other devices easyE4

### 10.14 Web Client

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  
<Web Client username>:<Web Client username password>@<device IP address>.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 application programming interface is described in a separate document – please visit [Eaton.com/easy-jsonapi](https://www.eaton.com/easy-jsonapi).

#### Web Client/Settings/API key

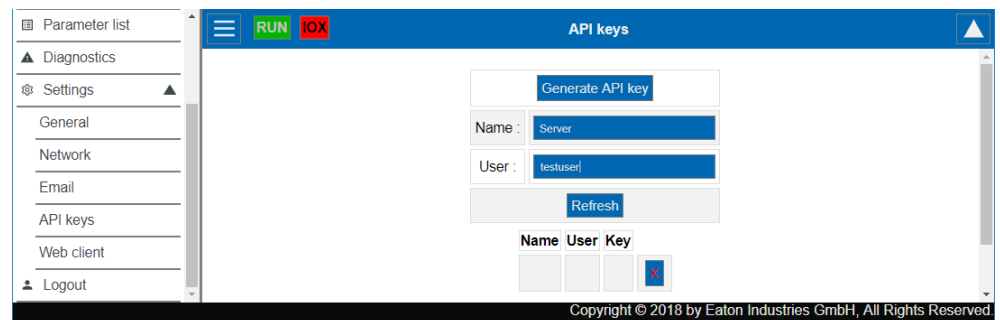


Fig. 334: Web Client, API key

#### 10.14.9.5 System update

easyE4 base devices belonging to generation 09 with firmware version 2.25 or higher can be updated through the Web Client or through the AWS Cloud (in addition to updates made with microSD memory cards).

- Download the easyE4 firmware you want from the Eaton Download Center to your PC.  
Category: Firmware Updates; Product group: easy / RTD; Product: easyE4 base devices
- Follow the on screen instructions.

10. Communication Connection to other devices easyE4

10.14 Web Client

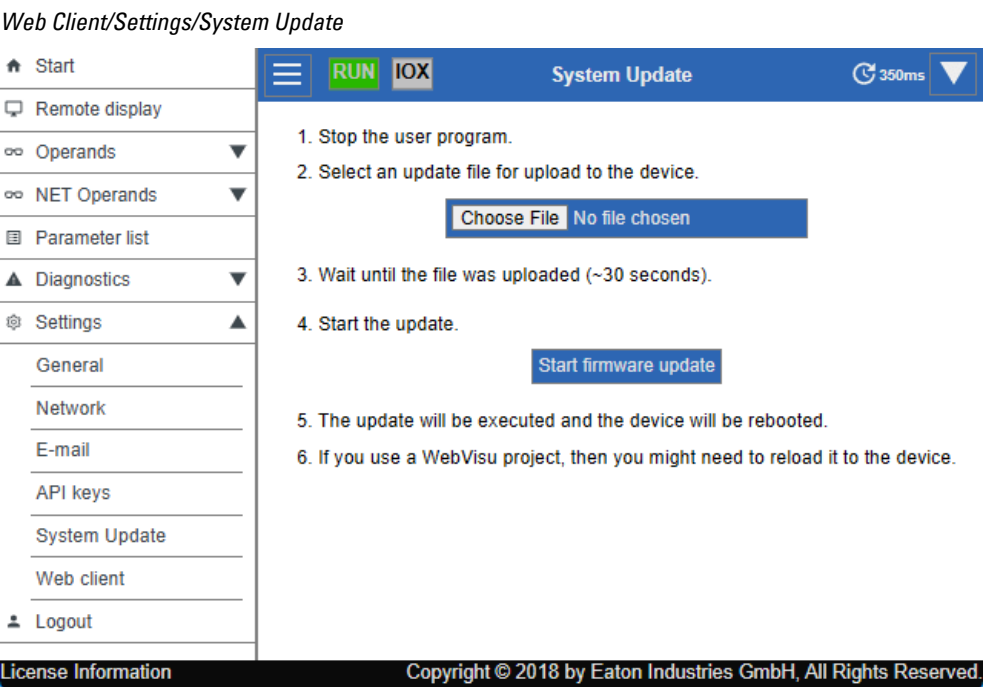


Fig. 335: Web Client, System update

➔ The easyE4 device must be stopped in order for the update to be carried out.

The web client will show messages indicating whether the upload and/or update has been successful.

10.14.9.6 Web Client

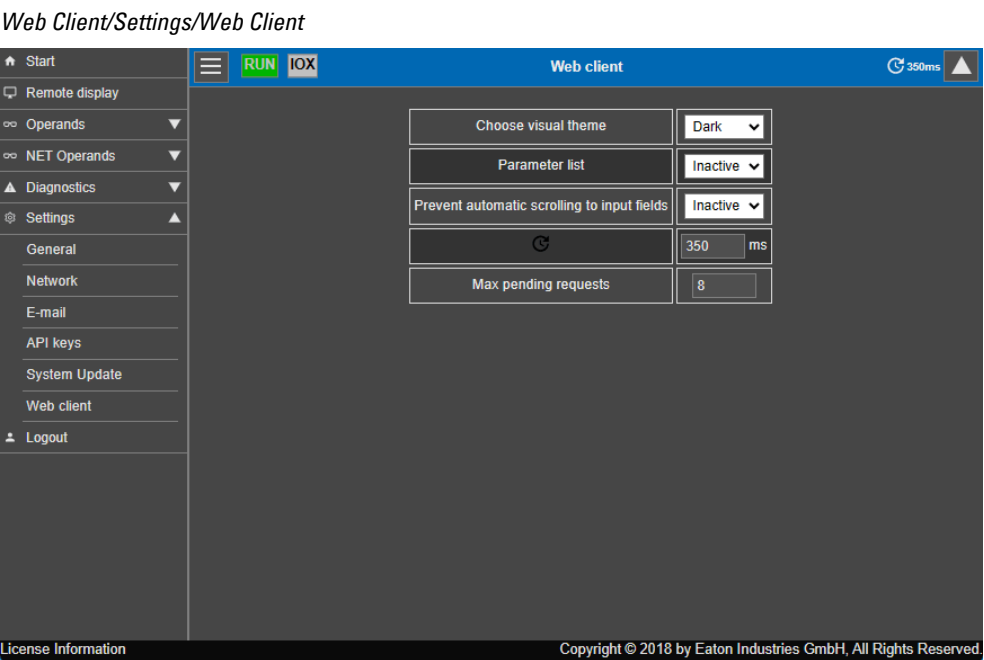


Fig. 336: Web Client, Web Client

### **Select display topic**

- White – The Web Client's user interface will be shown with a bright white background (light mode).
- Dark – The Web Client's user interface will be shown with a dark gray background (dark mode).

### **Parameter List**

- Active  
If this option is set to Enabled, creating a parameter list will be allowed. The Separate operands menu option can be accessed in the web client Web Client's left pane. This option is equivalent to the Parameter list active option in *Project view/Web server tab*; see also: → "Parameter list enabled", page 729.
- Inactive  
If this option is set to Disabled, creating a parameter list will not be allowed. The Separate operands menu option will not be shown in the Web Client's left pane. This option is equivalent to the Parameter list active option in *Project view/Web server tab*; see also: → "Parameter list enabled", page 729.

### **Deactivate automatic scrolling to input fields**

- 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.

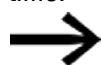
## 10. Communication Connection to other devices easyE4

### 10.14 Web Client

#### **Web Client cycle time**

The Web Client's cycle time is the time between two requests for the device that are 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 are 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.

#### **Maximum number of unanswered enquiries**

permitted input limits: 0-99

Number of enquiries per second is limited here before the connection is closed in order to prevent an endless loop.

Changes in the Web Client will not affect the settings in the \*e80 program. 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 728
- Section "AL - Alarm function block", page 472

## **10.15 Setting up the e-mail function**

Only possible with easySoft 8.

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, the easyE4 control relay must be able to establish an Ethernet connection to a public or private mail server.

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

## 10. Communication Connection to other devices easyE4

### 10.15 Setting up the e-mail function

#### 10.15.1 E-mail tab

All settings required to send an email are made in *Project view/Email tab*. The e-mail time stamp will incorporate the time zone configured for the device location.

*Project view/e-mail tab*

The screenshot shows the 'Project view/e-mail tab' configuration window. It has a tabbed interface with the following tabs: Device information, System settings, Security, Clock, NET, Ethernet, Webserver, Modbus server, E-Mail, Assigned operands, and Device properties. The 'E-Mail' tab is active. The window is divided into several sections: 'E-mail recipient' with three text input fields for 'Group 1:', 'Group 2:', and 'Group 3:'. 'Mail server settings' with radio buttons for 'IP address:' and 'DNS name:', and text input fields for 'Sender (From):', 'Sender domain:', 'E-mail service port:', 'Login name:', and 'Login password:'. A 'Connection security:' dropdown menu is set to 'STARTTLS'. 'System messages' with checkboxes for 'NET error occurred', 'Operating state changed', 'Program deleted', and 'Configuration error occurred'. A 'Send to recipient group:' dropdown menu is set to '1'.

Fig. 337: E-mail tab

#### E-mail recipient

You can enter up to three recipient groups. Put together, the three groups must not exceed a maximum of 254 bytes.

A recipient group can contain one recipient or several recipients separated with a semicolon.

The definition for each recipient group can have a maximum length of 254 bytes. E-mails will only be sent to recipient groups that contain recipients (e.g., when the e-mails are triggered by the alarm function block).



Please keep in mind that using characters that do not conform to ASCII code will take more than one byte per character.



Only when using FW version 1.x

In an e-mail recipient group used for system notifications, only the first ten entries will be processed.

Each of the e-mail addresses in this group must not exceed 60 ASCII characters.

Please note that an error message will not be output if one of these conditions is not met. Instead, the e-mail will simply not be sent.

#### 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 the mail server's IP address  
Enter the mail server's full name, e.g., "smtp.gmail.com"  
Use numbers and letters without special characters or umlauts.

## 10. Communication Connection to other devices easyE4

### 10.15 Setting up the e-mail function

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 (From)

The sender entered into this field will be used as the sender address in the e-mail. A maximum of 64 ASCII characters can be entered with 64 bytes.



Please keep in mind that using characters that do not conform to ASCII code will take more than one byte per character.

- 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, Gmail, 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.

☐ NET error occurred



## 10. Communication Connection to other devices easyE4

### 10.15 Setting up the e-mail function

- ☐ Operating state changed
- ☐ Program deleted
- ☐ Configuration error occurred

possible causes can be that one or more SWD participants are missing, the connection between easyE4 base device and easy communication module is interrupted because, for example, the plug connector is missing or the easy communication module is deenergized.

**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.

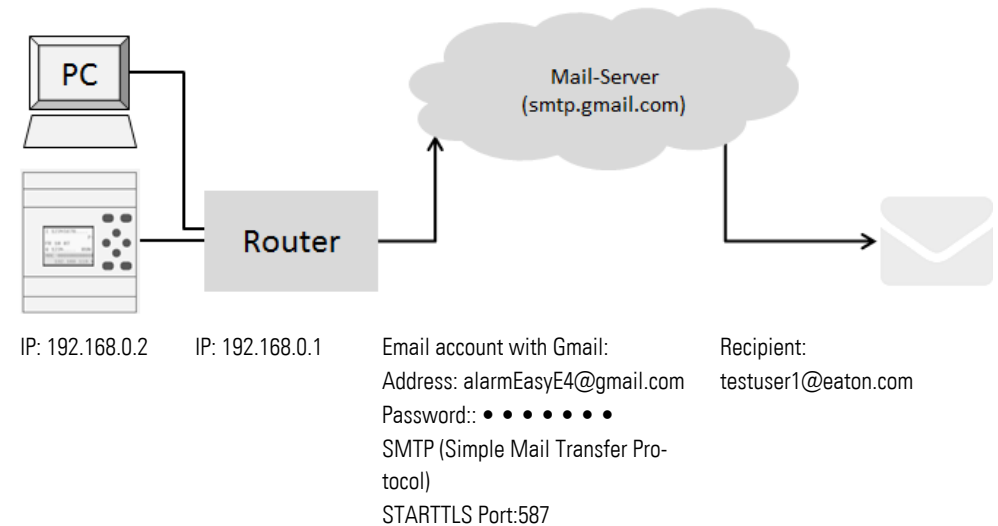
If the recipient group is empty and does not contain any recipients, the plausibility check will report an error.

## 10. Communication Connection to other devices easyE4

### 10.15 Setting up the e-mail function

#### 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.



#### 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 8.

- ▶ 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.

- ▶ Enter the recipient's e-mail address (e.g., <testuser1@eaton.com>) into one of the recipient groups (e.g., <Group 1>).

In the System messages section, select the events that should trigger an e-mail to the recipient group.

- ▶ Enable the Operating state changed option.
- ▶ In the Send to recipient group drop-down menu, select the 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.

## 10. Communication Connection to other devices easyE4

### 10.15 Setting up the e-mail function

- ▶ 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.

The screenshot shows the 'E-Mail' configuration tab within a software interface. The top navigation bar includes tabs for Device information, System settings, Security, Clock, NET, Ethernet, Webserver, Modbus server, E-Mail (selected), Assigned operands, and Device properties. The 'E-mail recipient' section has three input fields for Group 1, Group 2, and Group 3, with Group 1 containing 'testuser1@eaton.com'. The 'Mail server settings' section has radio buttons for 'IP address' and 'DNS name' (selected), with the 'DNS name' field containing 'smtp.gmail.com'. Other fields include 'Sender domain' (easyE4), 'E-mail service port' (587), 'Connection security' (STARTTLS), 'Login name' (alarmeasy@gmail.com), and 'Login password' (masked with dots and a checkmark). A 'System messages' section on the right has checkboxes for 'NET error occurred', 'Operating state changed' (checked), 'Program deleted', and 'Configuration error occurred', along with a 'Send to recipient group' dropdown set to 1.

Fig. 338: 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 Static 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.
- ▶ Enable the checkmark to activate the Enable configuration via network option. 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.

## 10. Communication Connection to other devices easyE4

### 10.15 Setting up the e-mail function

- ▶ 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

The screenshot shows the 'Ethernet' tab in the 'Project view'. It contains two main sections: 'IP settings' and 'DNS settings'.  
In 'IP settings', there is a dropdown for 'Static IP address' and a 'Mode' dropdown. Below these are three input fields: 'IP address' (192 . 168 . 0 . 2), 'Subnet mask' (255 . 255 . 255 . 0), and 'Gateway' (192 . 168 . 0 . 1). There is also a checkbox for 'Enable configuration via network' which is checked.  
In 'DNS settings', there is a 'Device name' field (myEasyE4), a 'Domain' field, and a 'DNS server' field (192 . 168 . 0 . 1). Below these is a 'Remote display configuration' section with an 'Access control' dropdown set to 'No access'.

Fig. 339: 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 LD or FBD.
- ▶ Drag a N/O to the work pane, such as I01.
- ▶ Drag a contactor to the work pane, such as Q01, in such a way that the coil will connect to the contact.

### 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.

## 10. Communication Connection to other devices easyE4

### 10.15 Setting up the e-mail function

#### 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

---

Fig. 340: Email example when the operating mode changes

## 10. Communication Connection to other devices easyE4

### 10.15 Setting up the e-mail function

#### Example: Sending an email with alarm function block AL

You now expand the previous example → Chapter "10 Setting up the e-mail function", page 758 by adding an alarm function block AL.

If the P button **P1** is pressed on the easyE4 base device, easyE4 should still be able to send an e-mail;

Only available on firmware version 2.00 or higher.

The value of marker word MW12 is also sent at this time.

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 Programming view.
- ▶ 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.
- ▶ Insert a free text in the message text field and insert the placeholder for the operand value with \$MW12\$ comprising a maximum of 160 bytes.
  - ➔ Please keep in mind that using characters that do not conform to ASCII code will take more than one byte per character.
- ▶ 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*.

## 10. Communication Connection to other devices easyE4

### 10.15 Setting up the e-mail function

#### Programming view/AL01

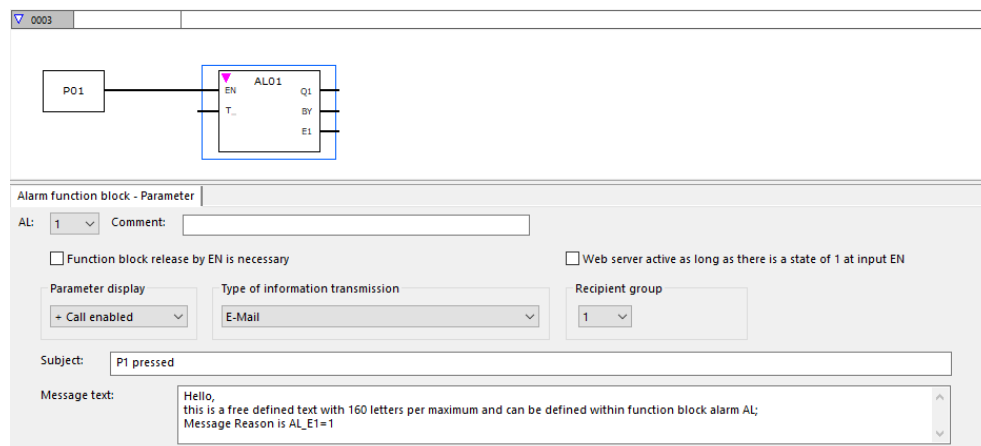


Fig. 341: Alarm function block tab with parameters from the example and FBD program with alarm function block and P button P01


#### 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.
- ▶ In the program, set the value of marker word MW12 to 255 with the help of a constant.

#### 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.

## 10. Communication Connection to other devices easyE4

### 10.15 Setting up the e-mail function

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  
MW12:255

---

Fig. 342: Example email when triggered by alarm function block AL01

#### See also

→ Section "AL - Alarm function block", page 472



## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

#### 10.16 easy communication modules

easy communication modules make it possible for the easyE4 base device to communicate with other devices, including devices from other manufacturers. This communication can be through a standard bus system such as Modbus RTU, as well as through SmartWire-DT. The device has its own firmware and no configurations are stored on it even though these configurations can be read with the easy communication module. Instead, configurations are forwarded to the easyE4 base device and stored there.

EASY-COM-... easy communication modules can be used with an easyE4 base device starting with generation 05.

(ID on the nameplate, → page 36)



It may be necessary to update the firmware on the easyE4 base device before use.



Only one of the easy communication modules is supported per easyE4 base device.

The easy communication modules need to be connected to the left side of the easyE4, while an I/O expansion for easyE4 control relays is connected to the right side.

The easy communication modules for easyE4 control relays are configured in easySoft 8. You can find the device catalog in the Communication modules folder.

For assignment purposes, easy communication modules are numbered, with the corresponding ID starting with the letter "C". If modules are added later on to a communication module, they will be numbered accordingly (e.g., C1.1, C1.2, C1.3).

The following easy communication modules are available:

- EASY-COM-SWD-C1 as a SmartWire-DT coordinator  
Only available on firmware version 1.30 or higher.  
By using an EASY-COM-SWD-... communication module, you can have the easyE4 function as a SmartWire-DT coordinator that coordinates the SmartWire-DT line with all the modules on the line and manages the data transmission process. This communication module is hereafter referred to as SmartWire-DT coordinator.
- EASY-COM-RTU-M1 to Modbus RTU communications  
Only available on firmware version 1.40 or higher.  
The communication module can be configured as a Modbus RTU master or as a Modbus RTU slave.

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

#### 10.16.1 easyE4 as a SmartWire-DT coordinator

easyE4 control relays that feature an EASY-COM-SWD-C1 module can be used as SmartWire-DT coordinators for lean automation applications.



easyE4 supports the Eaton Lean Automation concept that provides several significant benefits within the scope of the Lean Solution strategy: With SWD, the I/O level is integrated directly into the switchgear. This means that

easyE4 can obtain direct access to the digital and analog data from command devices up to power switches via SWD.

This dispenses with the gateway and the I/O level. This enables users to create flexible automation solutions with fewer components and less engineering overhead.

Eaton calls this concept Lean Automation, and uses it to provide users in the machine building and plant engineering industries with unparalleled freedom so that they can design creative and profitable solutions.

##### 10.16.1.1 SmartWire-DT - The System

The SmartWire-DT (SWD) communication system is an intelligent bus system and makes possible the reliable and easy connection of switching devices, pilot devices and I/O components with overriding bus systems.

An easy communication module EASY-COM-SWD-C1 is used to connect the SmartWire-DT components directly to the easyE4.

Up to 99 SmartWire-DT modules with a total of up to 224 digital inputs/outputs and/or up to 88 analog inputs/outputs can be connected on a SmartWire-DT line.

These SWD modules can be SmartWire-DT modules used to connect DIL contactors, PKE motor-protective circuit-breakers and motor starters, DS7 soft starters, field bus modules, and NZM circuit-breakers, as well as SmartWire-DT I/O modules, SmartWire-DT RMQ modules, and base modules for signal towers.

The electrical connection is effected via a special 8-pole connecting lead and the relevant plugs.

easySoft 8 is extremely useful when it comes to configuring the hardware and software for a SmartWire-DT line. As soon as an EASY-COM-SWD-C1 module is added to the project, the SWD tab will appear on the catalog. This SWD tab will help you select and configure the SmartWire-DT modules on the SmartWire-DT line.

The SWD tab has the current consumption specifications for all SmartWire-DT modules. During planning, it will automatically calculate and display the corresponding system's current consumption.



The inputs/outputs on a SmartWire-DT line are available in addition to the inputs/outputs of the I/O expansion for easyE4 control relays; the limit is the number of operands used in the .e80 project.

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

For up-to-date information on the SmartWire-DT communication system, please visit [Eaton.com/SWD](http://Eaton.com/SWD).

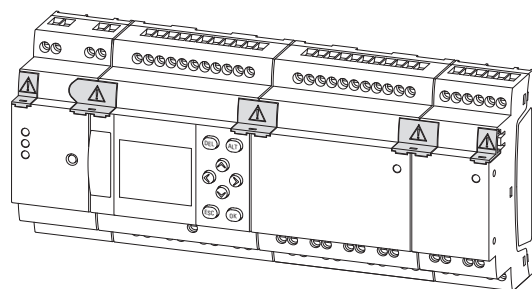


Fig. 343: Example showing an easyE4 control relay with I/O expansions and an easy communication module EASY-COM-SWD-...

To set up a SmartWire-DT line and install and run easyE4 as a SmartWire-DT coordinator, you will need to be familiar with the basic contents in the SmartWire-DT documents.

System description, engineering, installation, commissioning, and diagnostics for a SWD line



SmartWire-DT The System Manual

MN05006002Z

Setup, engineering, installation, etc. for the individual SWD modules



Manual for SmartWire-DT IP20 modules

MN05006001Z



Manual for SmartWire-DT IP6x modules

MN120006



Manual EMS2... Electronic motor starter with SWD

MN120008



PowerXL™ DX-NET-SWD manual

MN04012009Z



Installation instructions SWD4-...

IL04716001Z

For more information on how to set up, connect, and wire a SmartWire-DT line, please consult the Eaton Download Center - Documentation and the Eaton Online Catalog. Enter "SWD" or "SWD4" for SmartWire-DT accessories into the search box and the catalog will take you directly to the corresponding product group in the Automation, Control and visualization section.



[Eaton.com/documentation](http://Eaton.com/documentation)



[Eaton.com/easy](http://Eaton.com/easy)

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

#### 10.16.1.2 EASY-COM-SWD-... easy communication module

EASY-COM-SWD-..., with its connection to the easyE4 base device, acts as the SmartWire-DT coordinator on the SmartWire-DT line.

EASY-COM-SWD-... combines the functionality of an easyE4 with direct connection to SmartWire-DT communication system.

The easy communication module is where the SmartWire-DT line starts and has a connection to the 8-pin SmartWire-DT ribbon cable used inside the control panel in order to connect the SmartWire-DT modules. In addition to communication and control wires, this SmartWire-DT ribbon cable also carries the supply voltages for connected SmartWire-DT modules (15 V<sub>DC</sub>) and optionally used switchgear (24 V<sub>DC</sub>).

These two supply voltages are supplied through the EASY-COM-SWD-... module via the POW and AUX connection terminals.

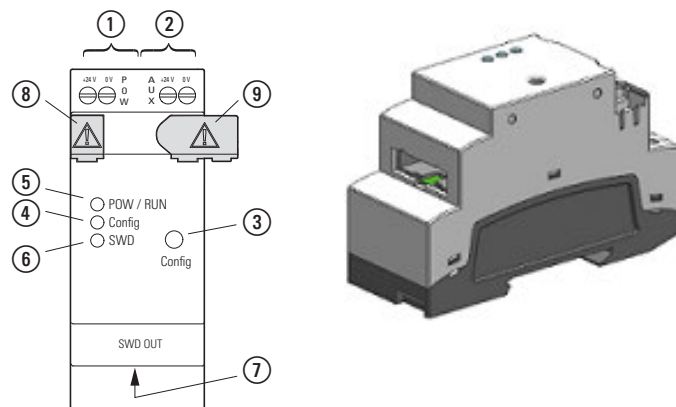


Fig. 344: Device model with 2SU

- |                    |               |                             |
|--------------------|---------------|-----------------------------|
| ① POW power supply | ④ LED Config  | ⑦ SWD OUT connection socket |
| ② AUX power supply | ⑤ LED POW/RUN | ⑧ Covering cap              |
| ③ Button Config    | ⑥ LED SWD     | ⑨ Bus connector plug        |

Installing a SmartWire-DT line involves the following steps:

1. Physically setting up the SmartWire-DT line
  - a. Installation in control panels
  - b. Installation in the periphery
  - c. Connecting external pilot devices
2. Commissioning the SmartWire-DT line
  - a. Configuring the SWD line
  - b. Testing the connected SWD modules
  - c. Connection to an easyE4 control relay

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

For basic information on how to install the EASY-COM-SWD-..., please refer to the following subsections in the Installation section:

- "Installation position", page 54
- "Mounting", page 58
- and
- "Connection terminals", page 66

#### Connecting the power supply through POW/AUX

The following supply voltages are required for a SmartWire-DT line:

- POW supply voltage:

The 24 V<sub>DC</sub> POW voltage input will first power the EASY-COM-SWD-... module itself.

The EASY-COM-SWD-... additionally contains a power supply that delivers 15 V<sub>DC</sub> voltage needed to power the SmartWire-DT modules in a control panel.

The maximum current load is 0.7 A. The voltage is not galvanically isolated from POW.

The device supply voltage for the electronics in the SmartWire-DT modules downstream (15 V<sub>DC</sub>) is generated using the 24 V<sub>DC</sub> supply voltage applied at the POW terminal.



If the current required by the connected SmartWire-DT modules exceeds the maximum capacity of 0.7 A, an EU5C-SWD-PF2-1 power feeder module needs to be included in the SmartWire-DT line configuration.

The power feeder module contains a power supply that can be used to feed a new 15 V<sub>DC</sub> voltage to power the SmartWire-DT modules inside a control panel.

The generated 15 V<sub>DC</sub> voltage will be galvanically isolated from the power feeder module's 24 V<sub>DC</sub> supply voltage.

- AUX supply voltage:

The AUX 24 V<sub>DC</sub> voltage input is used exclusively to power the 24 V<sub>DC</sub> contactors. The maximum current carrying capacity is 3 A (CE/IEC/EN) or 2 A (UL/CSA).

If there are any contactors or motor starters in the SmartWire-DT topology, a 24 V<sub>DC</sub> AUX voltage must additionally be supplied as a control voltage for the contactor coils.



If the current required by the connected switchgear exceeds the maximum capacity of 3 A or 2 A, an EU5C-SWD-PF1-1 or EU5C-SWD-PF2-1 power feeder module needs to be included in the SmartWire-DT line configuration.

For EASY-COM-SWD-..., use cable protection (F1) with a capacity of at least 3 A (slow).

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

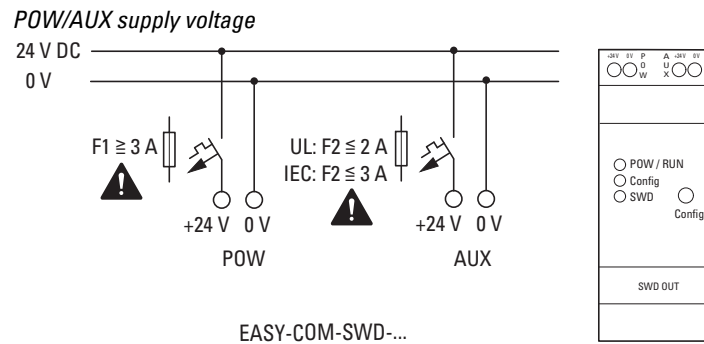
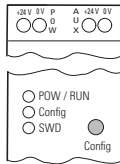


Fig. 345: Connecting the EASY-COM-SWD-... power supply

#### Terminal layout

Tab. 130:

signal	Description
+24 V <sub>DC</sub> POW	Supply voltage $U_{POW}$ +24 V DC
0 V POW	Supply voltage $U_{POW}$ 0 V
+24 V <sub>DC</sub> AUX	Supply voltage $U_{AUX}$ +24 V DC
0 V AUX	Supply voltage $U_{AUX}$ +0 V




10. Communication Connection to other devices easyE4

10.16 easy communication modules

Connecting the SmartWire-DT line to the SWD OUT female connector

The EASY-COM-SWD-C1 module features an SWD OUT connector.

This SWD OUT connection is not galvanically isolated from the POW supply voltage.



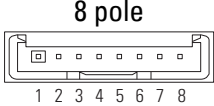
**CAUTION**

**RISK OF DAMAGE TO DEVICES**

▶ Do not connect the SmartWire-DT line to the easy communication module or disconnect it without first de-energizing the system.

SmartWire-DT uses an 8-pin ribbon cable in control panels. In addition to communication wires, this ribbon cable carries the power supply for the SWD modules, the switchgear, as well as control wires for assigning addresses.


Tab. 131: Pin assignment for SmartWire-DT ribbon cable interface (pin header, 8-pin)

Plug connector SWD4-8MF2	PIN	signal	Configuration
	1	+24 V <sub>DC</sub>	Contactor control voltage
	2	Chassis ground	Contactor control voltage
	3	GND	for device supply voltage and data cable
	4	Data B	Data cable B
	5	Data A	Data cable A
	6	GND	for device supply voltage and data (Data A, Data B)
	7	SEL	Select cable for automatic addressing of the SWD slaves
	8	+15 V <sub>DC</sub>	Device supply voltage

8-pin SWD4-8MF2 blade terminals need to be connected at the beginning and end of the SWD ribbon cable. Look for the directional arrow on the ribbon cable in order to identify where the SmartWire-DT ribbon cable starts and connect the corresponding connector to the EASY-COM-SWD-... module's SWD OUT connector.

Only use the following ribbon cables to connect the SmartWire-DT line to the SWD OUT connector:

- SWD4-100LF8-24 with the relevant blade terminals SWD4-8MF2 or
- SWD4-(3/5/10) F8-24-2S (pre-terminated cable).



Do not connect the SmartWire-DT line to the EASY-COM-SWD-... or disconnect it without first de-energizing the system.

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

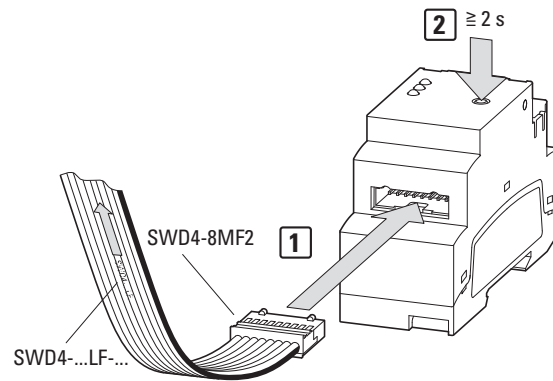


Fig. 346: Connecting EASY-COM-SWD-...

- ▶ 1. Connect the SmartWire-DT ribbon cable to the SWD-OUT connector.
- ▶ 2. Switch on the supply voltage.
- ▶ 3. Configure the SmartWire-DT line



## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

#### Commissioning the SmartWire-DT line

Prerequisite for commissioning the SmartWire-DT line

The following requirements must be met before switching on the line during initial commissioning, as well as after replacement, or modifying the SmartWire-DT configuration:

- All SmartWire-DT modules must be connected to each other with SmartWire-DT cables.
- The SmartWire-DT line must be connected to the SWD OUT connector.
- The power supply for the easyE4 device and for EASY-COM-SWD-... must be on and connected.
- The POW LED on EASY-COM-SWD-... must be on.
- The status LEDs of the connected SWD modules must be flashing or showing a solid light.
- There must be an existing \*.e80 easySoft 8 project in which the base device is configured with the EASY-COM-SWD-... (project configuration).

#### Configuring the SWD line



Required every time that a new SmartWire-DT module is added or removed,  
regardless of the \*.e80 easySoft 8 project being used.

Follow the steps below:

- Press and hold down the Config button for at least 2 seconds.

The SWD LED on the EASY-COM-SWD-... will start to flash yellow.

The status LEDs on the connected SmartWire-DT modules will flash.

The SWD LED on the EASY-COM-SWD-... will start to flash green.

Addresses will be assigned to all SmartWire-DT modules.

The SmartWire-DT line's physical configuration will be stored in the easyE4 retentive memory as a target configuration.

The SWD LED on the EASY-COM-SWD-... will light up green.

- Load the easySoft 8 project.

#### SmartWire-DT configuration tests

The SmartWire-DT module configurations are compared every time the power supply is switched on.

- The modules that are actually found on the SmartWire-DT line will be compared with the target configuration stored on the easyE4 base device:  
If the SmartWire-DT line's physical configuration matches the target

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

configuration, the SmartWire-DT line will be ready to start transferring data.

- The target configuration stored in the easyE4 base device will be compared with the project configuration defined in easySoft 8:

If the target configuration matches the project configuration, the Config LED will light up with a solid green light.

#### 10.16.1.3 LED status messages on the EASY-COM-SWD-... communication module

##### LED POW/RUN EASY-COM-SWD-...

Shows the status of the POW supply voltage, as well as the RUN and STOP modes.

Off	Malfunction or no supply voltage
Green, continuous light	Supply voltage OK, RUN mode
Green, Flashing, 1 Hz	Supply voltage OK, STOP mode
Green, Flashing, 3 Hz	Supply voltage OK, STOP mode EASY-COM-SWD-... and easyE4 are unable to exchange data e.g. bus interface connector not plugged in, faulty or easyE4 switched off
Green, Flashing, 10 Hz	Device waiting for firmware update
Green, Flashing, 0.5 Hz	Firmware update in progress

##### Config LED on EASY-COM-SWD-...

Shows whether the SmartWire-DT coordinator project configuration defined in easySoft 8 matches the target configuration for the SmartWire-DT line stored on the easyE4 base device.

Off	<ul style="list-style-type: none"> <li>• No project configuration present.</li> <li>• Incorrect target configuration (see LED SWD).</li> </ul>
Red, continuous light	The project configuration and the stored target configuration are not compatible with each other.
Green, Flashing, 2.5 Hz	The project configuration is compatible with the stored target configuration.
Green, continuous light	The project configuration matches the stored target configuration

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

#### SWD LED on EASY-COM-SWD-...

Indicates whether the physical configuration of the SWD line matches the target configuration stored in the easyE4.

Off	No target configuration present
Red, continuous light	<ul style="list-style-type: none"><li>• Short-circuit on the 15 V<sub>DC</sub> power supply.</li><li>• No SmartWire-DT modules found.</li></ul>
Red, flashing, 2.5 Hz	<ul style="list-style-type: none"><li>• The modules found on the SmartWire-DT line do not match the target configuration.</li><li>• A SmartWire-DT module configured as necessary is missing.</li></ul>
Yellow, flashing, 2.5 Hz	The SmartWire-DT line's physical configuration is being imported and stored as a new target configuration on the device.
Green, Flashing, 2.5 Hz	<ul style="list-style-type: none"><li>• The physical configuration of the SWD-T line is being compared with the target configuration.</li><li>• Addresses are being assigned to the SmartWire-DT modules.</li></ul>
Green, continuous light	<ul style="list-style-type: none"><li>• The modules found on the SmartWire-DT line match the target configuration.</li><li>• The SmartWire-DT line is ready for data transfers.</li></ul>

As soon as all the LEDs on the EASY-COM-SWD-... turn on with a solid green light, the easy communication module can be configured accordingly in easySoft 8 and the easyE4 control relay can be used as a SmartWire-DT coordinator in a user program.

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

Programming with easySoft V8 describes how to create an \*.e80 project with an easy communication module

#### Project creation with easySoft 8

Only available on firmware version 1.30 or higher.

By using an EASY-COM-SWD-... communication module, you can have the easyE4 function as a SmartWire-DT coordinator that coordinates the SmartWire-DT line with all the modules on the line and manages the data transmission process. This communication module is hereafter referred to as "SmartWire-DT coordinator."

As soon as a SmartWire-DT coordinator is dragged to the left side of the easyE4 base device in the Project view, the "SWD" tab will appear above the catalog. This tab shows a list of devices from which you can drag all the SmartWire-DT modules you need onto the work pane one after the other to create a project with a SmartWire-DT line.

Please note than only one SmartWire-DT coordinator is allowed per base device!

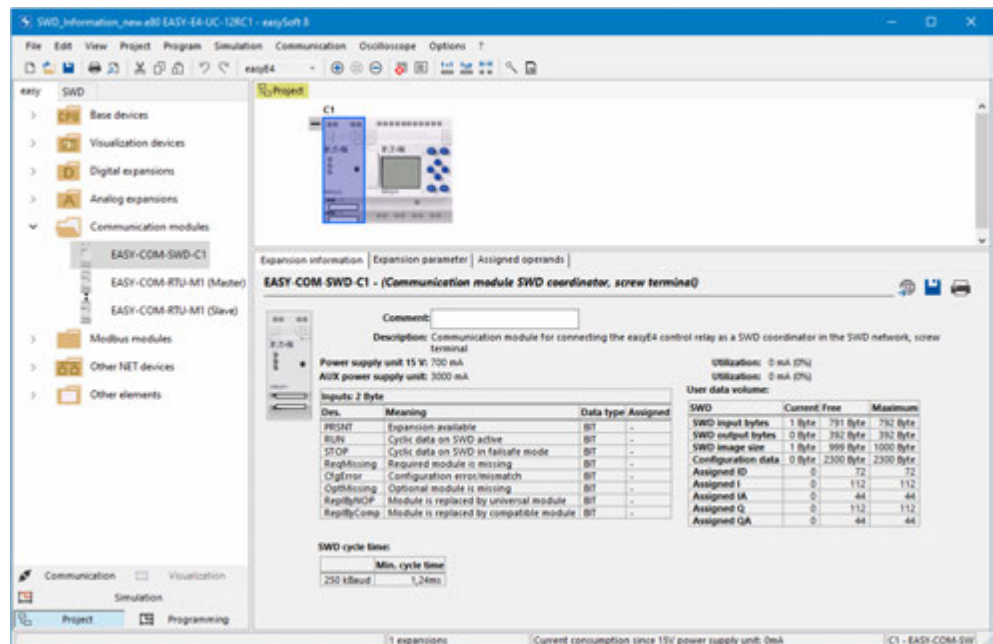


Fig. 347: Work pane with base device and communication module; catalog expanded with "SWD" tab

## 10. Communication Connection to other devices easyE4

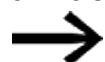
### 10.16 easy communication modules

#### Problems on the SmartWire-DT line

If an error occurs on the SmartWire-DT line,

- The SWD LED on the EASY-COM-SWD-... is flashing red or showing a solid red light,
- And the Stop on SWD error option is enabled in the .e80 project,

the easyE4 base device will immediately be switched to STOP mode and the outputs on the SmartWire-DT modules experiencing the problem will be switched off.



If the Stop on SWD error option is not enabled in the .e80 project, the easyE4 base device will remain in RUN mode. The output on the SmartWire-DT module experiencing the problem will be switched off.

A SmartWire-DT module that is experiencing an error can be identified with easySoft 8 in the program.

- ▶ To analyze the error, connect a computer with easySoft 8 to the control relay.
- ▶ Use easySoft 8 to check the SmartWire-DT modules.

After replacing the faulty SmartWire-DT → page 777. Once you are done with this, the line will be ready for use immediately.



There is the option of mapping diagnostic alarms to the corresponding operands (e.g., the PSNT bit) for each SmartWire-DT module in the .e80 project. Please refer to → page 686

For easyE4 base devices with a display, an indication of whether the SmartWire-DT module is being detected can be output for assistance.

Tab. 132: *Example*

M22 present: <input checked="" type="checkbox"/>
on I17: <input checked="" type="checkbox"/>
off I18: <input type="checkbox"/>

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

#### 10.16.2 easyE4 Communication via Modbus RTU

The EASY-COM-RTU-... communication module allows the control relays in the easyE series to establish a communication connection via Modbus RTU. This not only makes it possible to use Modbus-RTU-capable Eaton devices, but also Modbus-RTU-capable devices from third parties.

The Modbus RTU communication module can be configured as a master or as a slave.

If the EASY-COM-RTU-... is used as a master, the easyE4 base device will control all data traffic on the bus by sending requests to the relevant slaves in the Modbus RTU communication system.

If the EASY-COM-RTU-... is used as a slave, the easyE4 base device will respond to requests from the Modbus RTU master.

Among other things, this means that Modbus RTU communications are also possible between multiple easyE4 base devices.

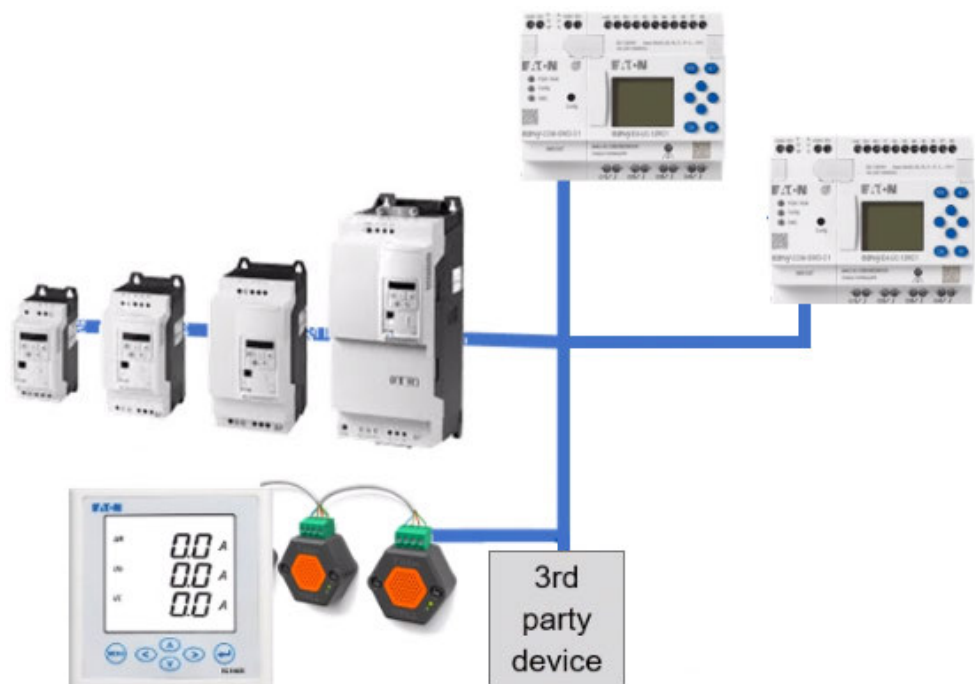


Fig. 348: General diagram: The easyE4, set up as a Modbus RTU master, communicates with the devices set up as Modbus RTU slaves (DE1, DC1, DG1, DA1, easyE4, and others)

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

The EASY-COM-RTU-... communication module only supports half-duplex communication mode.

Two types of dialog are possible between master and slave:

- The master sends a request to a slave and waits for a response.
- The master sends a request to all slaves and does not wait for a response (broadcast).



For more information on Modbus communications, please visit [modbus.org](http://modbus.org) and read the following documents:

- MODBUS over serial line specification and implementation guide
- MODBUS application protocol specification

The EASY-COM-RTU-... easy communication module supports Modbus communications with up to 32 slaves.

The length of the bus should not exceed 600 m. Stub lines are not recommended.

Combining an easyE4 control relay and an EASY-COM-RTU-... communication module results in up to 224 digital operands (112 inputs, 112 outputs) and up to 88 analog operands (44 inputs, 44 outputs) that can be assigned.

The EASY-COM-RTU-... communication module is supported by generation 05 and higher easyE4 base devices when used with firmware 1.40 or higher.

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

#### 10.16.2.1 EASY-COM-RTU-... easy communication module

The external power supply (24 VDC) is connected to one of the two POW terminals on the EASY-COM-RTU-... and is protected against reverse polarity.

The Modbus RTU network is connected to the COM, B+, A- RS-485 terminals on the EASY-COM-RTU-... module.

The module features integrated bus polarization (bias) and bus termination and can be activated separately with easySoft 8.

The following can be configured:

- The baud rate – 2400, 4800, 9600, 19200, 38400, 57600, and 115200
- The stop bits – 1 or 2  
and
- The parity bit – none, even, or odd

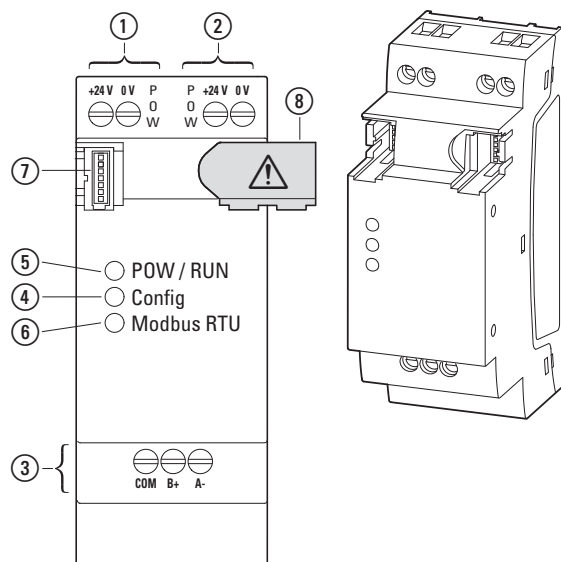


Fig. 349: Versions

- |                                       |                  |                                    |
|---------------------------------------|------------------|------------------------------------|
| ① POW power supply left               | ④ LED Config     | ⑦ Cover<br>(for easyE4 connection) |
| ② POW power supply right              | ⑤ LED POW/RUN    | ⑧ Bus connector plug               |
| ③ Modbus RTU terminals<br>COM, B+, A- | ⑥ LED Modbus RTU |                                    |



10. Communication Connection to other devices easyE4

10.16 easy communication modules

To install a Modbus RTU communication system, follow the steps below in the specified order:

- 1. Install the module mechanically and connect it to the easyE4 base device
- 2. Connect the Modbus RTU signal cables to the connection terminals on the EASY-COM-RTU-... module
- 3. Connection of power supply
- 4. Configure the EASY-COM-RTU-... module in easySoft 8

Please note that the EASY-COM-RTU-... connection can only be configured with easySoft 8.

See also

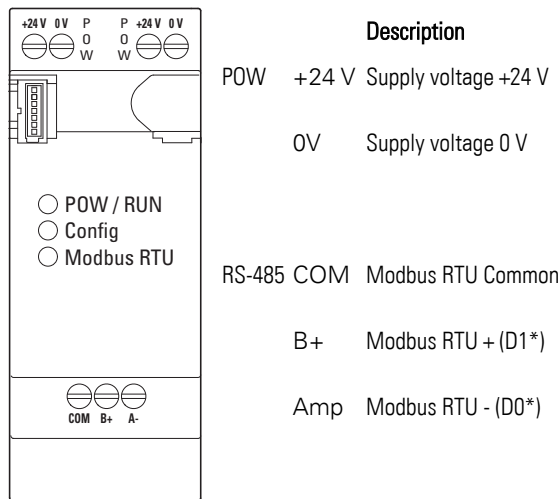
- "Installation position", page 54
- "Mounting", page 58
- and
- "Connection terminals", page 66

Connecting the Modbus RTU signal cables

Terminal layout

The EASY-COM-RTU-... module features an RS-485 interface that is galvanically isolated from the power supply (POW).

Tab. 133: Terminal assignment EASY-COM-RTU-...



\* D1 and D0 are the designators in conformity with [modbus.org](https://modbus.org) as defined in the following documents:

- MODBUS over serial line specification and implementation guide
- MODBUS application protocol specification

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

#### Wiring topic

► Use shielded twisted pair cables.



Signals B+(D1) and A-(D0) must be connected to twisted pair cables.  
The shielding must be connected to the protective earth.

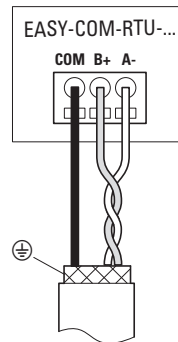


Fig. 350: Connecting EASY-COM-RTU-... outputs



## **10. Communication Connection to other devices easyE4**

### **10.16 easy communication modules**

Commissioning requires easySoft programming software.

The configuration test is run every time the power is turned on, as well as every time after a project is transferred to the easyE4 control relay.

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

#### 10.16.2.2 LED status messages on EASY-COM-RTU-... communication module

##### LED POW/RUN EASY-COM-RTU-...

Shows the status of the POW supply voltage, as well as the RUN and STOP modes.

Off	Malfunction or no supply voltage
Red, flashing, 5 Hz	Serious error; the UART interface between EASY-COM-RTU-... and the easyE4 base device cannot be initialized, i.e., no data transfers between EASY-COM-RTU-... and easyE4
Green, continuous light	Operating mode RUN, normal operating mode: <ul style="list-style-type: none"><li>• No communication errors with the ComBUS</li><li>• No missing slaves on the Modbus (in master mode)</li></ul>
Green, Flashing, 1 Hz	STOP mode <ul style="list-style-type: none"><li>• The easyE4 base device is in the STOP state.</li><li>• In master mode: One of the slave devices is not present / is not reporting in</li></ul>
Green, Flashing, 3 Hz	Error in Modbus RTU communications: ComBUS error <ol style="list-style-type: none"><li>1. CRC fault</li><li>2. Timeout error</li></ol>
Green, Flashing, 10 Hz	Device waiting for firmware update
Green, Flashing, 0.5 Hz	Firmware update in progress

##### Config LED on EASY-COM-RTU-...

Indicates whether the defined project configuration has been transferred

Off	There is no project configuration on the EASY-COM-RTU-..., i.e., a project was not received from the easyE4 base device when turning on or the old project configuration was deleted as a result of a user command.
Red, continuous light	Invalid project configuration received from easyE4 base device
Green, continuous light	There is a valid project configuration (master and slave modes), i.e., all project settings are valid and accepted.

##### LED Modbus RTU on EASY-COM-RTU-...

Indicates whether the physical configuration of the Modbus RTU communication system is working.

Yellow	Turns on for 50 ms when a new message is received or sent via Modbus.
--------	---

As soon as the POW/RUN LED and the Config LED light up green, the EASY-COM-RTU-... module is ready for communications via Modbus RTU.

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

easySoft 8 describes how to create an \*.e80 project with an easy communication module



Continuing as described below is only possible with easySoft 8.

#### Project creation with easySoft 8

Only available on firmware version 1.40 or higher.

The EASY-COM-RTU-... communication module makes it possible for easyE4 to establish Modbus RTU communications with other devices.

If you drag a Modbus RTU master onto the left side of the easyE4 base device in the Project view, easyE4 will be able to communicate with up to 32 Modbus RTU slaves. Meanwhile, if you drag a Modbus RTU slave onto the left side of the easyE4 base device in the Project view, easyE4 will be able to communicate with a Modbus RTU master.

Only one easy communication module is allowed per base device!

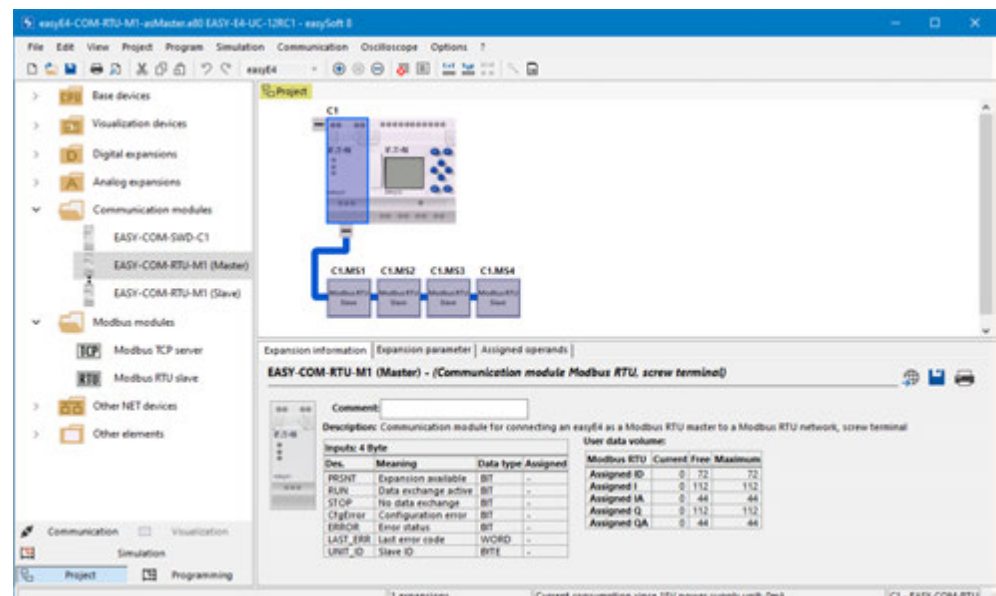


Fig. 352: Work pane with base device and EASY-COM-RTU-M1 master communication module

## 10. Communication Connection to other devices easyE4

### 10.16 easy communication modules

#### Modbus RTU communication errors

If an error occurs, this will be indicated on the easy communication module:

- The Config LED will light up red if an invalid project configuration is detected
- The Modbus RTU LED will not light up yellow

#### See also

→ "LED status messages on EASY-COM-RTU-... communication module", page 789

#### Modbus RTU communication errors

Problem	Explanation	Remedy
POW/RUN LED flashes red at a frequency of 5 Hz	Connection between easyE4 base device and EASY-COM-RTU-... lost	Check the connector contact
Modbus RTU LED no longer flashing yellow	Modbus RTU packages are not being received/sent	
In master mode, the POW/RUN LED flashes green at a frequency of 1 Hz	A slave device is not reporting in	

## 10.17 Connecting to the AWS Cloud

Only available on easySoft 8.25 or higher.

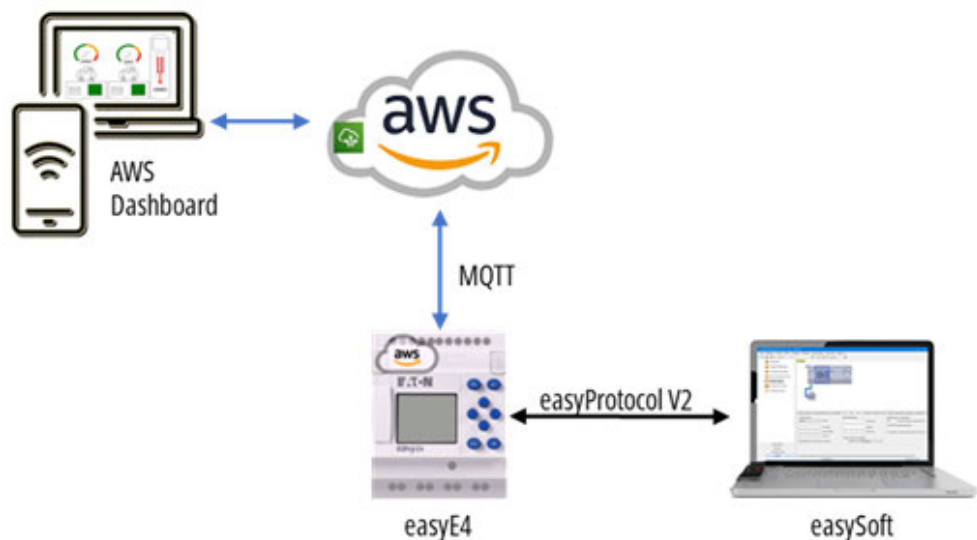
Generation 08 easyE4 base devices with firmware version 2.20 or higher can connect directly to cloud service provider Amazon Web Services (AWS) and exchange data with it.

AWS is a cloud computing platform that offers comprehensive features with more than 200 services. The easyE4 has the option of communicating with this platform so that individual operands can be made available in the cloud and written to the easyE4 from there, for example.

Communication takes place via the AWS IoT Core service, which connects IoT devices to other devices and to AWS Cloud services. The files and data required for this purpose are managed with the Amazon S3 service.

This makes the following possible:

- Publishing configured easyE4 operands
- Remotely modifying easyE4 operands configured for this purpose in the project
- Publishing device parameters such as: firmware version, serial number, and program name
- Updating the user program on easyE4 devices
- And
- Updating the firmware on easyE4 devices belonging to generation 09 or higher



The cloud connection is disabled by default.



## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

#### Function principle

The easyE4 will use the MQTT protocol to send its data to an object (thing) in the AWS IoT Core. Once there, a device shadow will be used.

An easyE4 device configured with the AWS Cloud service will connect to the AWS IoT Core service when turned on. In RUN mode, this device will send its data. In STOP mode, the transfer of data from the easyE4 to AWS will be paused (but the connection to the cloud will remain in place).

The fact that data is securely transmitted to the cloud means that users can also use other AWS services, including data storage, data analyses, and data visualizations. This in turn makes it possible to implement IoT (Internet of Things) solutions ranging from remote monitoring to machine learning. In addition, Eaton is an AWS partner with device registration for the easyE4 product line, which makes the setup and configuration processes much easier.

With a direct connection to AWS, device data will not only be in the cloud, but can also be quickly and easily accessed even on mobile devices.

For more information on AWS, please visit the AWS homepage and the links found there, including for:

- Cloud computing with AWS  
[https://aws.amazon.com/what-is-aws/?nc1=f\\_cc](https://aws.amazon.com/what-is-aws/?nc1=f_cc)
- AWS Documentation  
<https://docs.aws.amazon.com/>
- AWS Pricing Calculator  
<https://aws.amazon.com/iot-core/pricing/?nc=sn&loc=4>
- AWS Free Tier  
<https://aws.amazon.com/free>
- Device Shadow service  
<https://docs.aws.amazon.com/iot/latest/developerguide/iot-device-shadows.html>
- AWS IoT  
<https://aws.amazon.com/iot/>
- AWS IoT rules  
<https://docs.aws.amazon.com/iot/latest/developerguide/iot-rules-tutorial.html>
- Getting started with AWS IoT Core  
<https://docs.aws.amazon.com/iot/latest/developerguide/iot-gs.html>
- What is AWS IoT?  
<https://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html>
- MQTT  
<https://docs.aws.amazon.com/iot/latest/developerguide/mqtt.html>
- STS documentation for AWS  
<https://docs.aws.amazon.com/STS/latest/APIReference/welcome.html>
- AWS Identity and Access Management (IAM)  
<https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html>

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

- AWS IoT Jobs

<https://docs.aws.amazon.com/iot/latest/developerguide/iot-jobs.html>

The way in which the easyE4 maps data is based on the device shadow service principle. A device shadow will reflect the latest version of easyE4 data that was accepted in AWS.

With AWS IoT Core, this device shadow will be available on the corresponding object (thing). Meanwhile, the classic shadow document is a JSON document that is used to retrieve the latest status information for a device.

Data exchanges between AWS and the easyE4 must take place using the device shadow for the easyE4. This can be edited:

Manually in AWS, e.g., with the editing window,  
with the update topic in the MQTT test client, or  
with programming using the AWS APIs.

Program and firmware updates for easyE4 devices are carried out with AWS IoT Jobs.

AWS Identity and Access Management is extremely secure. Identification requires not only a username and password, but also a certificate.

As soon as you have created your own AWS account, you will be able to log in to AWS and register your easyE4 devices. When registering, a certificate signing request (CSR) will be sent to AWS by the easyE4. After this, AWS will return a certificate that will enable the easyE4 to access the AWS Cloud even without a username and password.

You will need the following credentials (provided by AWS) in order to register an easyE4 in the AWS Cloud:

- AWS access key ID
- AWS secret access key
- AWS session token

```
SET AWS_ACCESS_KEY_ID=ASIA4JOASV5LPQOVFFOP
SET AWS_SECRET_ACCESS_KEY=bK4QJHSpqzp0sbqEAVa2nyVKkqR9o2S+c/H2EOdn
SET AWS_SESSION_TOKEN=IQoJb3JpZ2luX2VjEBMaC...(892 characters)
Example: AWS Cloud security
```

AWS credentials are valid only temporarily. Nevertheless, a registered easyE4 device will be able to continue communicating with AWS without issue and it will not be necessary to enter any credentials again, since the signed certificate will be stored permanently on the easyE4 base device.

#### Steps required in order to enable remote access to an easyE4 through the cloud:

1. Make sure that the easyE4 has a constant Ethernet connection to the Internet. For static IP addresses and Auto-IP mode, you will need to specify the corresponding gateway and DNS server address.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

2. Register as a customer with the cloud service provider  
(→ Section "AWS - access", page 796)
3. Create an \*.e80 program with an AWS configuration and transfer it to the easyE4.

If you want to use the option of registering through the web server, you will need to enable the web server as well

4. Register the easyE4 with the cloud service  
(→ Section "AWS IoT Core – registering an easyE4", page 808)

After this, the easyE4 device will be able to communicate with the AWS Cloud and exchange data accordingly.

### **10.17.1 AWS - access**

To use the Amazon Web Services (AWS) cloud platform, you will need your own AWS account. Visit AWS, register, and create an account, or have your company do so.

To do this, go to the AWS homepage (<https://aws.amazon.com/>) or directly to the AWS Identity and Access Management (IAM) page (<https://aws.amazon.com/iam/>...)

- [https://signin.aws.amazon.com/signup?request\\_type=register](https://signin.aws.amazon.com/signup?request_type=register)
- <https://pages.awscloud.com/IAM-communication-preferences.html>

The cloud service provider's homepage is available in several languages and provides all the information you will need, including pricing information.

- <https://aws.amazon.com/free>

Follow the steps for creating an AWS account – please refer to → Section "Creating accounts for Amazon Web Services (AWS) ", page 802.

Once you are done, you will get a 12-digit account number and will be able to log in with your credentials.

Make a note of the region where you are logging in.

You will receive the following for access to the individual services:

- AWS access key ID
- AWS secret access key
- AWS session token

These AWS security credentials will be required when registering your easyE4 devices in the AWS Cloud.



Depending on the AWS account and type of access, you may first have to generate an AWS session token with AWS temporary IAM security credentials.

For more information on exchanging data, please refer to the AWS STS documentation.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

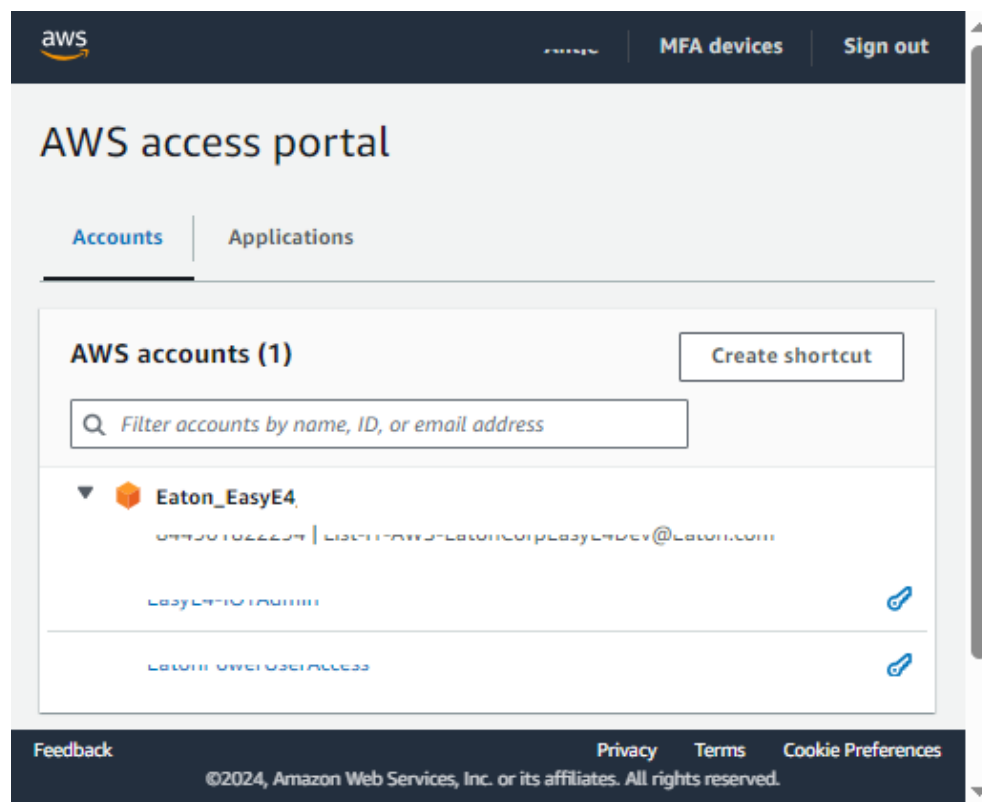


Fig. 353: Example: Access with a user's own AWS account



For an update via AWS IoT, you will need an IAM role.

#### **IAM role**


An IAM role is an IAM identity that you can create in your AWS account and that will have specific permissions.

This IAM role must have permissions for downloading and/or uploading files from/to the S3 bucket .

Meanwhile, an Amazon Resource Name (ARN) is a unique identifier that is used to identify an IAM role within the AWS system.

You may have to create a new IAM role in your AWS account or contact your AWS admin.

#### **10.17.1.1 Enabling data transfers**

To establish a connection between the devices and the cloud, use the  AWS IoT Core service.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

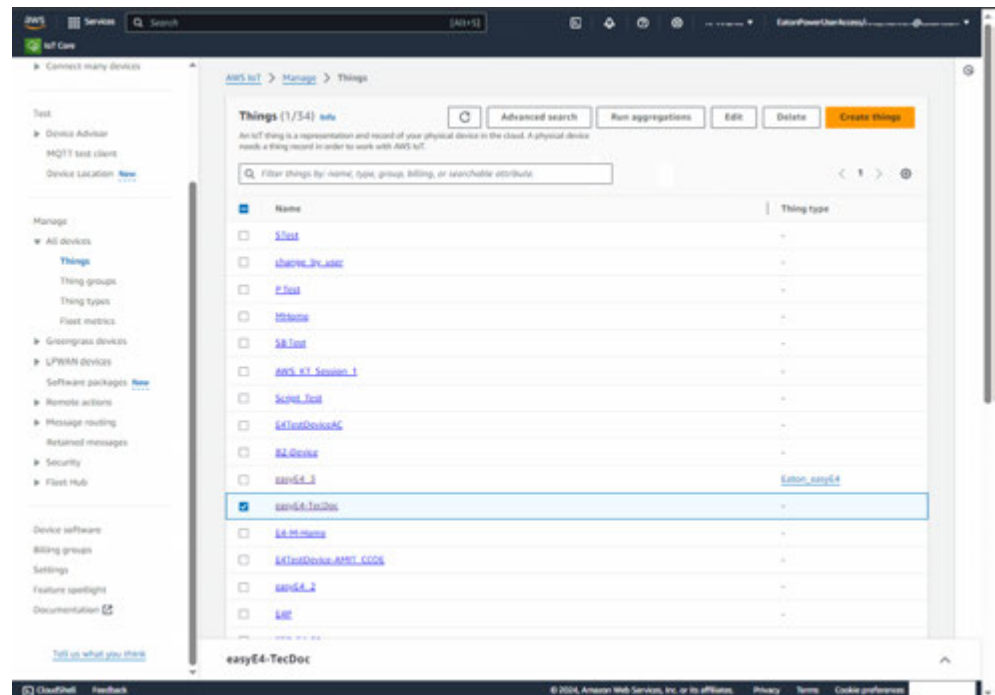


Fig. 354: Example with easyE4 in AWS; objects - things - available for selection

As soon as the easyE4 device has been created as an object (thing) in the cloud, there will be various tabs and other AWS functions available for use.

#### Certificates tab

The login certificate issued will be stored here.

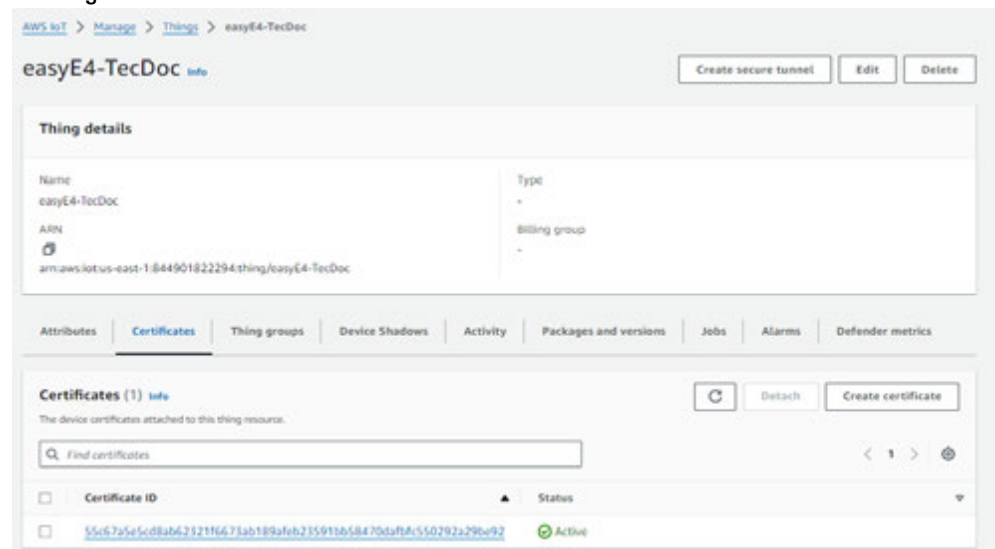


Fig. 355: Example: AWS Certificates tab

#### Device Shadows tab / Activity tab

The data exchanged between the easyE4 device and the cloud can be edited in these tabs.

10. Communication Connection to other devices easyE4

10.17 Connecting to the AWS Cloud

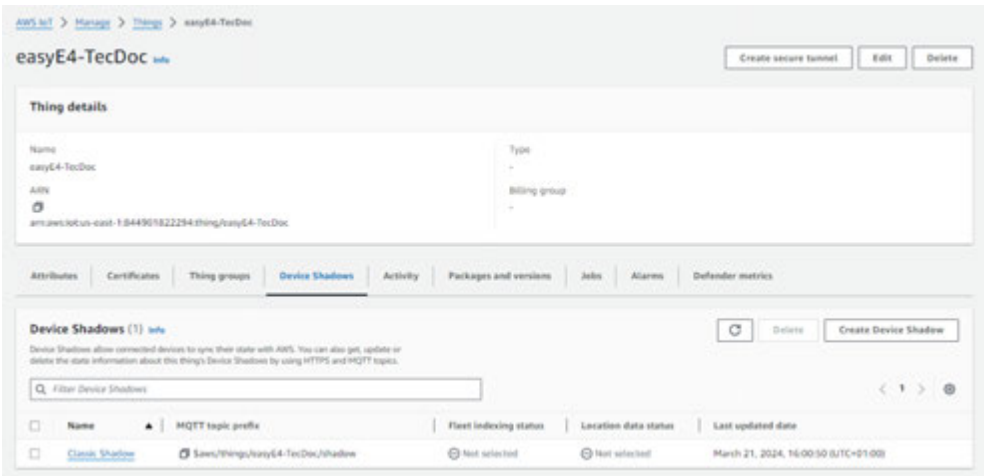


Fig. 356: Example: AWS Device Shadows tab

The Classic Shadow will show the most recent version of the easyE4 data.

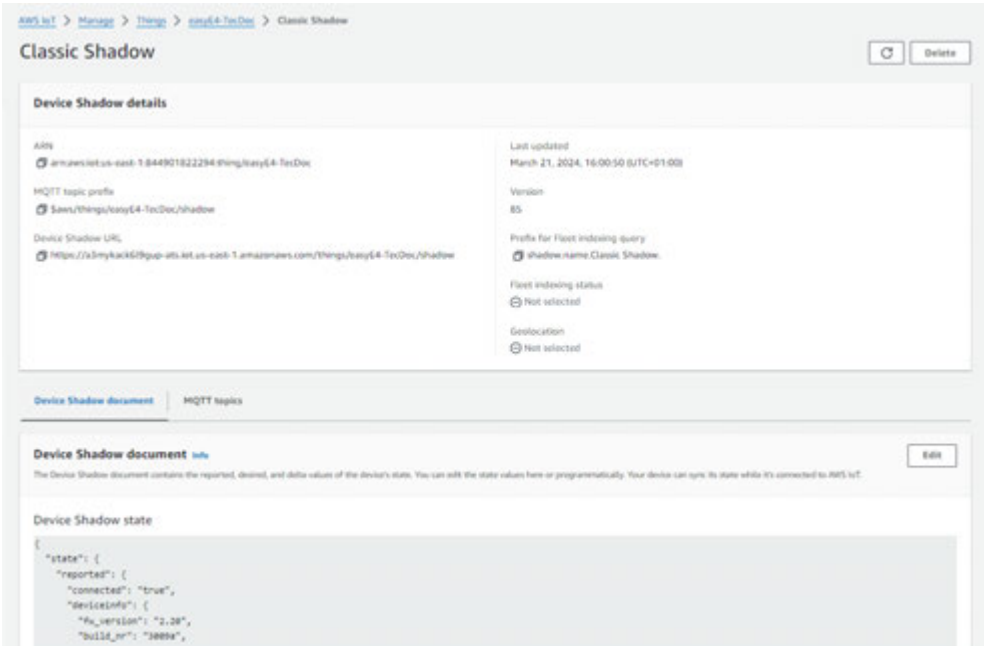


Fig. 357: Example: AWS Device Classic Shadow tab

You can open the MQTT test client directly for editing with the Activity tab.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

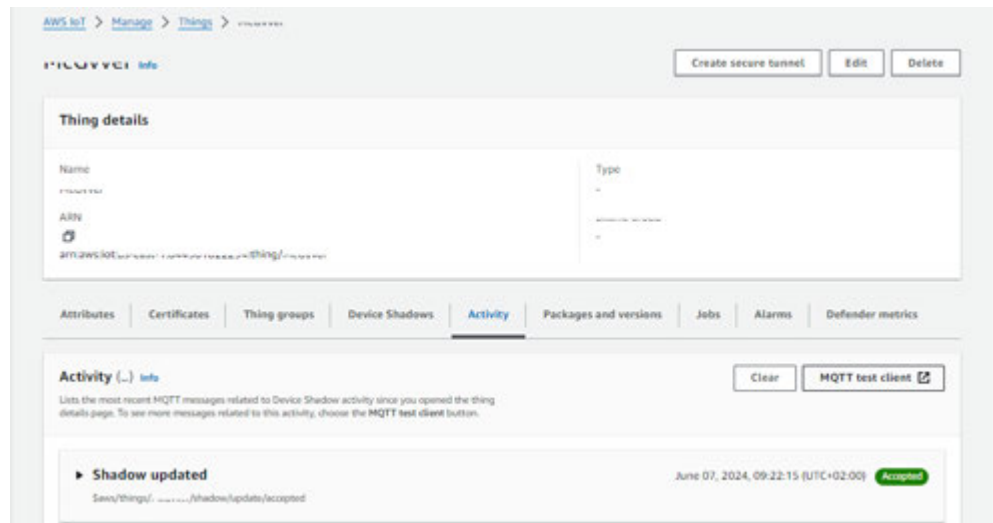


Fig. 358: Example: AWS Activity tab

#### Leading Questions:

The editor window can be used to send data to the easyE4 device.

A new object in JSON format with the "desired" identifier must be created for this purpose.

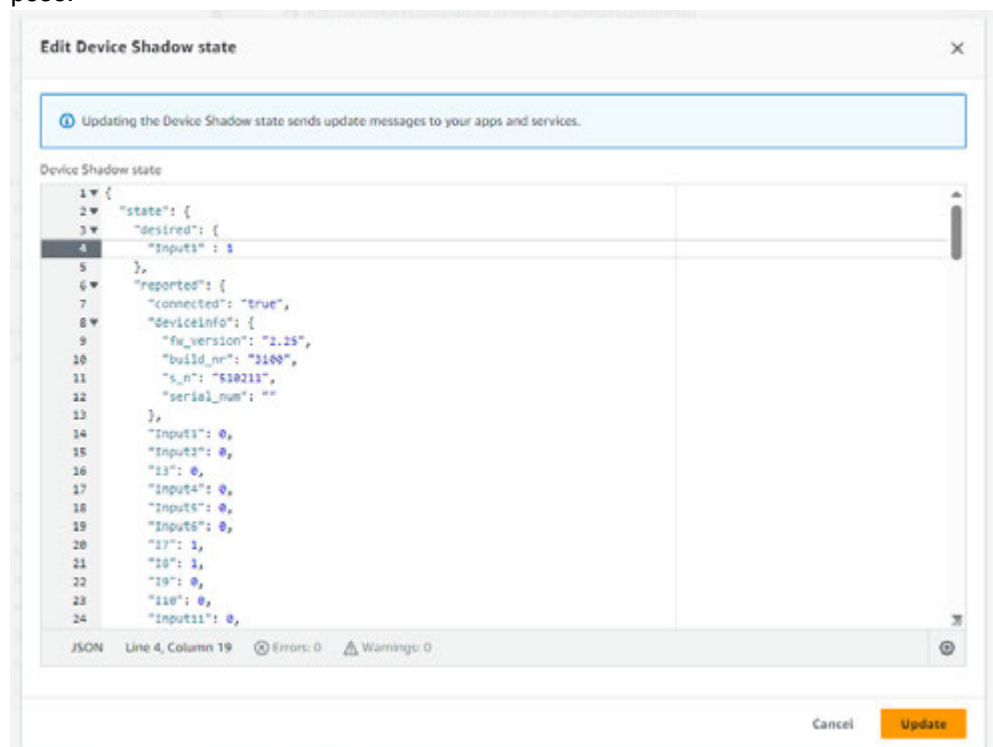


Fig. 359: Example: AWS Device Classic Shadow, Editor

Additional keys (operand name or alias names) and values (data values) can be entered.



## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

If one of the operands is not writable, data will still be sent to the easyE4 device, but the device will not implement this data and the "desired" identifier will be retained in the device shadow.

The value in "desired" will remain as is until it is deleted manually (this always applies, even in cases in which the operand can be written to).

To remove the "desired" identifier from the device shadow, you will need to set the value for the corresponding key to "null".

For more information on exchanging data, please refer to the AWS MQTT documentation and Eaton AP050027EN.

#### ***ATTENTION***

Make sure that the certificate information and AWS parameters are correct before using AWS to exchange data!

### 10.17.2 Creating accounts for Amazon Web Services (AWS)

Make sure to learn more about the cloud service provider and AWS Identity and Access Management (IAM) first so that you will be able to choose the right option for you when signing up.

AWS Identity and Access Management (IAM) is a web service that you can use to securely control access to AWS resources. With IAM, you can manage permissions centrally to define who is authenticated (signed in) and who is authorized (has permissions) to use AWS as a cloud service.

If you create an AWS account, you will start with a sign-in identity that will provide full access to all AWS service resources in the account.

This identity will be known as the AWS account root user. To access this identity, sign in with the e-mail address and password that you chose when creating the account.



AWS explicitly discourages the use of the root user for every-day tasks. Instead, you should safeguard your root user credentials and use them exclusively for root user tasks.

For a full list of the tasks that will require you to sign in as the root user, please refer to [Tasks that require root user credentials](https://docs.aws.amazon.com/IAM/latest/UserGuide/root-user-tasks.html) in the AWS documentation (<https://docs.aws.amazon.com/IAM/latest/UserGuide/root-user-tasks.html>).

#### Creating a root user

- Create an AWS account.

Fig. 360: <https://pages.awscloud.com/IAM-communication-preferences.html>

- Set up the root user.

Follow the instructions for creating and activating a new AWS account at: <https://repost.aws/knowledge-center/create-and-activate-aws-account>

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

#### Creating and activating a new AWS account

If you are using AWS for the first time, you will need to start by creating and activating an AWS account.

##### 1. Sign up with your e-mail address

- ▶ Go to the [Amazon Web Services \(AWS\) homepage](https://aws.amazon.com/resources/create-account/).( <https://aws.amazon.com/resources/create-account/>)

- ▶ Click on "Create an AWS Account."



If you have signed in to AWS recently, click on "Sign in to the Console." If the "Create a new AWS account" option is not visible, click on "Sign in to a different account" first and then on "Create a new AWS account."

- ▶ Enter a corporate e-mail distribution list or mailbox as an e-mail address in "Root user email address."



This e-mail address can be used to reset account credentials if necessary. Accordingly, make absolutely sure to safeguard access to these distribution lists. Moreover, do not use the AWS account root user login for other tasks. One tried-and-true approach is to enable multi-factor authentication (MFA) for the root account in order to better secure your AWS resources.

- ▶ Enter the AWS account name you want.



Make sure to use a standard account name for the AWS account name so that you will be able to recognize the name in your invoice and the Billing and Cost Management console. In other words, the name must make it possible to determine the entity to which it belongs.

- ▶ Click on "Verify email address."

An AWS confirmation e-mail with a confirmation code will be sent to the address you entered.

Please note that you can change the account name later on in the account settings after signing up.

##### 2. Confirm your e-mail address

- ▶ Enter the code you received.
- ▶ Then click on "Verify."

It may take a few minutes before the code arrives. Make sure to check your e-mail and your spam folder for the message with the confirmation code.

### **3. Create a password**

- ▶ Enter the root user password you want and then enter it again into the password confirmation field.
- ▶ Then click on "Continue."

### **4. Add your contact information**

- ▶ Select "Personal" or "Business"

(both accounts have the same features and functions).

- ▶ Enter your personal or business information.

For business AWS accounts, it is recommended to enter the company's phone number instead of a personal cell phone number (configuring a root account with an individual e-mail address or a personal phone number can put your account at risk by making it insecure).

- ▶ Read and accept the AWS Customer Agreement.
- ▶ Then click on "Continue."

You will receive an e-mail confirming that your account has been created. You can then sign in to your new account with the e-mail address and password you used when signing up.

However, please note that you will not be able to use the actual AWS services until after you have activated your account.

### **5. Add a payment method**

- ▶ On the "Billing information" page, enter the information for your payment method.
- ▶ Then click on Verify and Add.

If you are signing up for an Amazon Web Services India Private Limited (AWS India) account, you will need to provide your CVV for the verification process. You may also have to enter a one-time password depending on your bank. Please note that AWS India will charge two Indian Rupees (INR) to your payment method as part of the verification process (AWS India will refund the two INR after the verification is complete).

If you want to use a different billing address for your AWS billing information, click on "Use a new address." Then click on "Verify and Continue."



You will not be able to continue with the signup process until you have added a valid payment method.

### **6. Confirm your phone number**

- ▶ On the "Confirm your identity" page, select a contact method where you can receive a confirmation code.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

- ▶ Select the country or region code for your phone number from the list.
- ▶ Enter a cell phone number where you can be reached in the next few minutes.
- ▶ If a CAPTCHA appears, enter the code shown and submit it.
- ▶ An automated system will contact you in a little bit.
- ▶ Enter the PIN you receive and then click on "Continue."

#### 7. Customer verification

If you are signing up with a billing or contact address located in India, you will need to follow the steps below:

- ▶ Select the "Primary purpose of account registration" for which you are creating an account. If your account is tied to a business, select the option that applies most closely to your business.
- ▶ Select the "Ownership type" that best represents the account owner. If you select a company, organization, or partnership as the ownership type, enter the name of a key manager. This key manager can be a director, head of operations, or a person in charge of operations at your business.
- ▶ Click on "Continue."

#### 8. Select an AWS Support plan

- ▶ On the "Select a Support plan" page, choose one of the available support plans.

You can find a description of the available support plans and their benefits on the "Compare AWS Support plans" page.

- ▶ Click on "Complete signup."

#### Wait for account activation

Once you have selected a support plan, a confirmation page will appear to let you know that your account is being activated. Accounts are normally activated within a matter of minutes, but in some cases may take up to 24 hours to be activated.

You will still be able to sign in to your AWS account during this time (please note that the AWS homepage may still show the "Complete Sign Up" button during this time even if you have already completed all the signup steps).

Once your account has been fully activated, you will receive a confirmation e-mail (make sure to check your mailbox and spam folder for this e-mail). Once you receive this e-mail, you will have full access to all AWS services.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

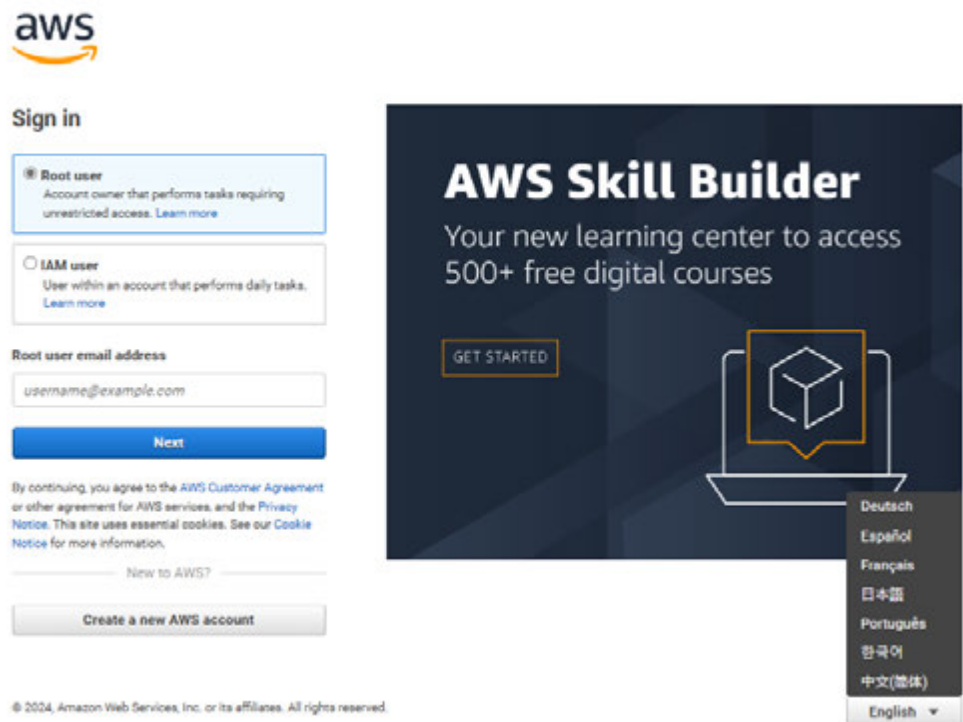


Fig. 361: <https://signin.aws.amazon.com>

### Setting up AWS IoT Core as a work environment

AWS IoT Core is a managed cloud service that makes it possible for connected devices to easily and securely interact with cloud applications and other devices. AWS IoT Core manages the devices and supports notifications to AWS endpoints and other devices.

For more information, please visit the AWS IoT Core website.

After you sign in, the homepage for your AWS console will open.

- Select "IoT Core" from the available AWS services.

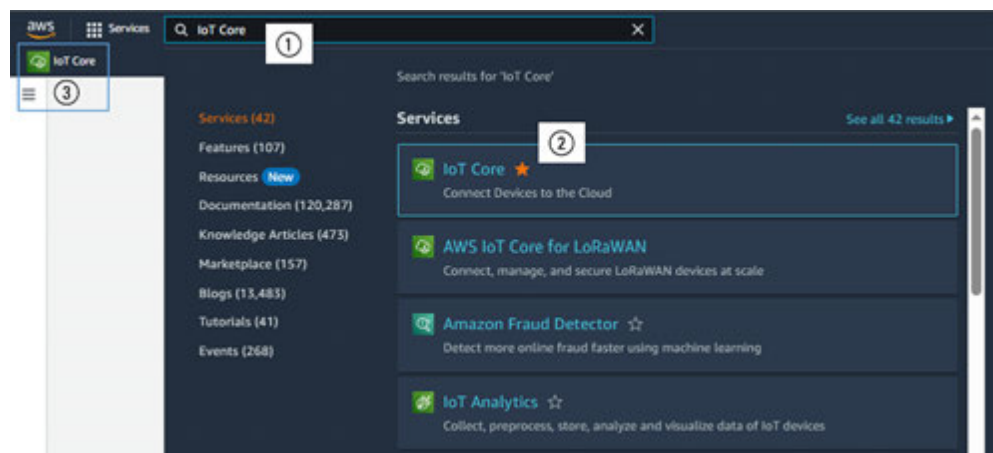


Fig. 362: Example: AWS console; selecting the IoT Core service

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

As soon as you have set the "IoT Core" AWS service as a favorite (2), it will appear at the top of the console.

This will make it faster to access it next time you open the console.

#### Setting up the Amazon S3 AWS service as a work environment

The Amazon S3 AWS service is an object storage service used to store and retrieve any amount of data from any storage location.

For more information, please visit the AWS Amazon S3 website.

After you sign in, the homepage for your AWS console will open.

► Select "S3" from the available AWS services.

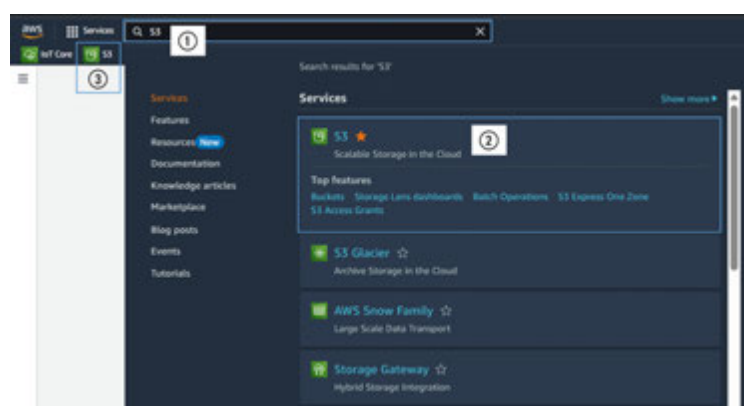


Fig. 363: Example: AWS console; selecting the IoT Core service

As soon as you have set the "S3" AWS service as a favorite (2), it will appear at the top of the console.

This will make it faster to access it next time you open the console.

### **10.17.3 AWS IoT Core – registering an easyE4**

There are several ways to register easyE4 devices in the AWS cloud.



The AWS IoT Core platform's option for connecting a device cannot be used for easyE4 devices.

Instead, the easyE4 registration must be started from the easyE4 page.

Registering and deregistering easyE4 devices on AWS requires an active Internet connection.

#### **Basic prerequisite for AWS IoT Core registration for an easyE4 device:**

- Known AWS security credentials
- The easyE4 device must already have a loaded \*.prg program with AWS set up
- The connection to the easyE4 must have already been established with the Communication view

#### **10.17.3.1 Method 1: using the easyE4 Wizard**

You can use the easyE4 Wizard to register the easyE4 device in the AWS cloud from easySoft 8.

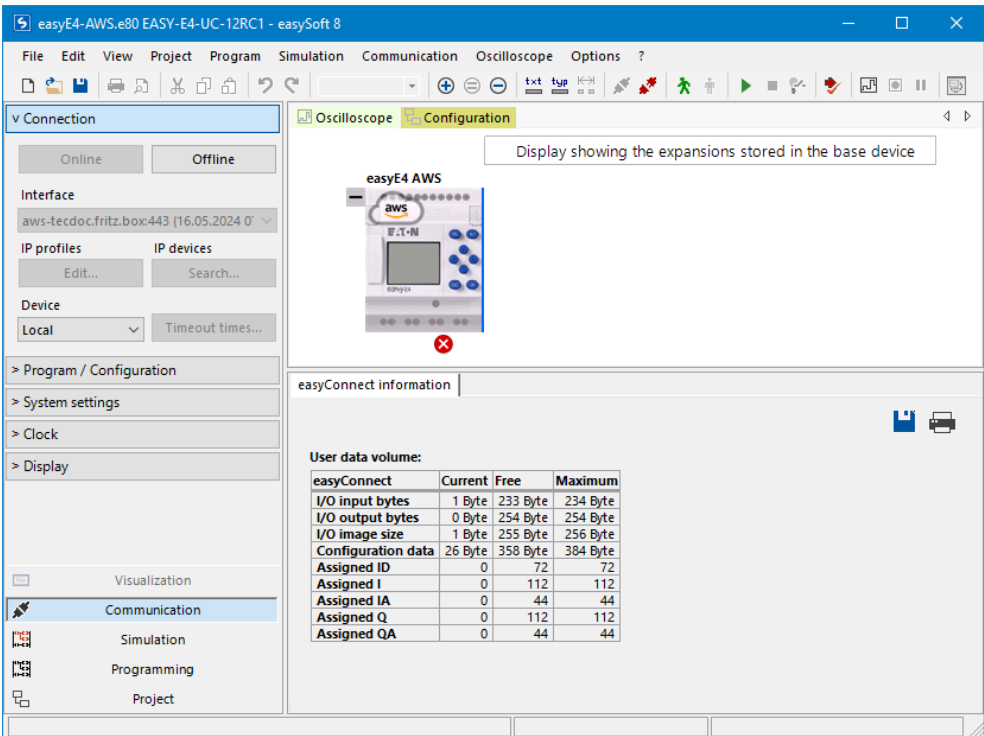
##### **Requirement:**


- Known AWS security credentials
- The easyE4 device must already have a loaded \*e80-program with AWS set up
- The connection to the easyE4 must have already been established with the Communication view



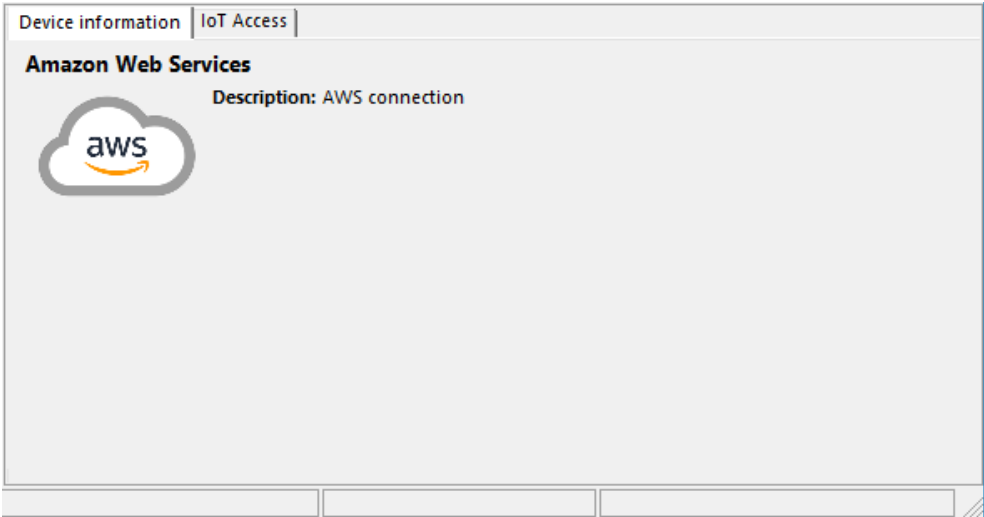
10. Communication Connection to other devices easyE4

10.17 Connecting to the AWS Cloud



► Click on the  AWS symbol on the base device.

The "Device information" tab will open.

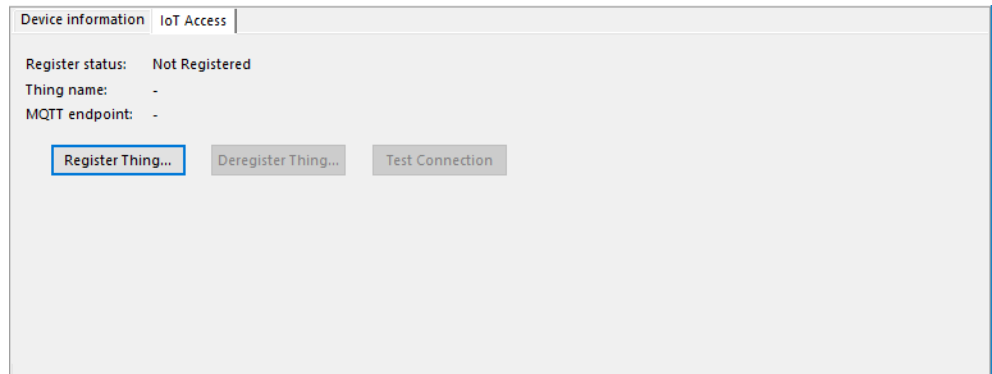


► Switch to the IoT Access tab.

Once the IoT Access tab is open, you can use it to register or deregister the easyE4 as an object (thing) on the AWS account, as well as to test the connection.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud



#### Register object...

The easyE4 must be in STOP mode when being registered.

The wizard will take you through the registration process in three steps.

#### Step 1: Credentials

► Have your AWS security credentials ready.

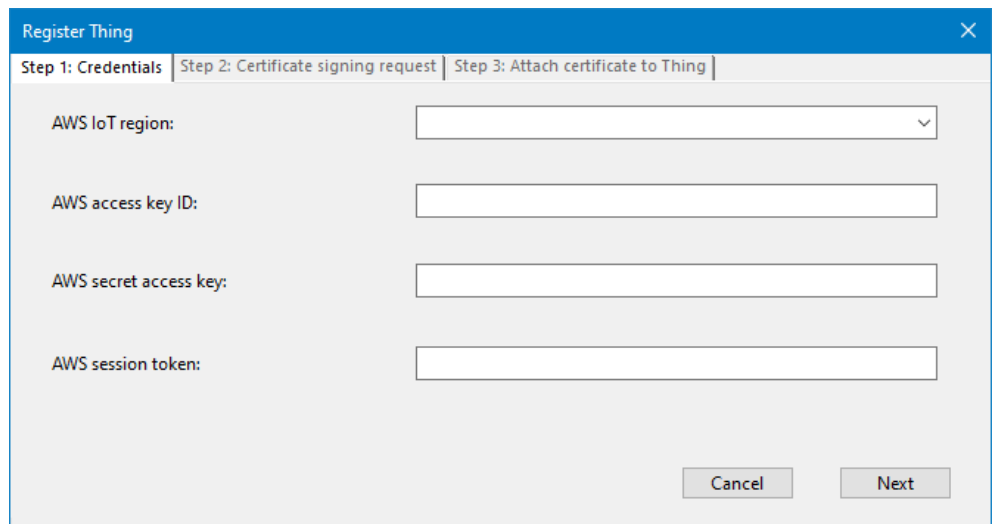


Fig. 364: easyE4 Wizard, Credentials



Depending on the AWS account and type of access, you may first have to generate an AWS session token with AWS temporary IAM security credentials.

For more information on exchanging data, please refer to the AWS STS documentation.

As soon as you are done entering the required information, click on **Next**. The system will check the information in the easyE4 before proceeding to the next step.

If any of the entered information is invalid, an error message will appear.

#### Step 2: Certificate signing request

Certificate signing will be requested in the second step.

10. Communication Connection to other devices easyE4

10.17 Connecting to the AWS Cloud

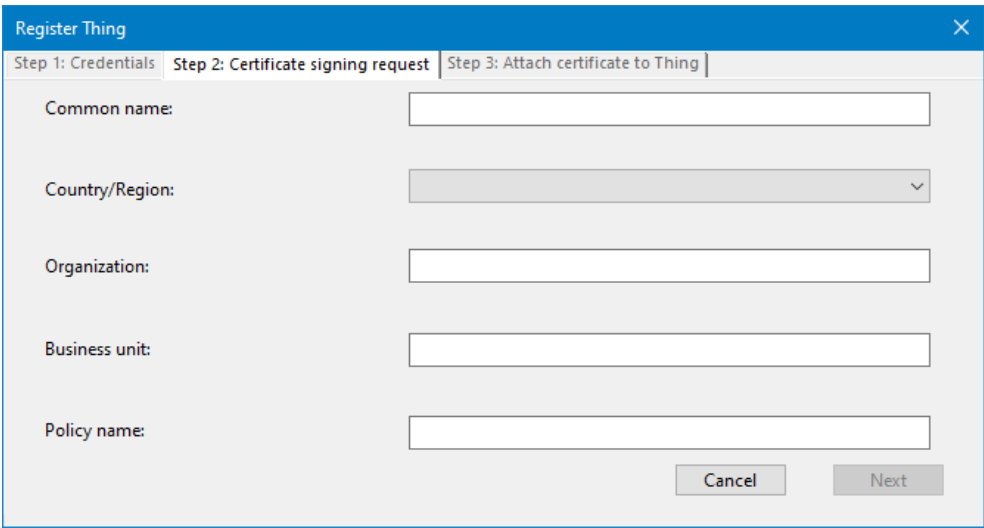


Fig. 365: easyE4 Wizard, certificate

The common name is the name of the certificate signing requester (Subject in AWS).  
The policy name must match the policy created in AWS without fail.

The policy name determines the permissions for the easyE4 in the cloud.  
This policy name can contain letters, numbers, and special characters \_ - , . @ = and must not exceed 128 characters total.

For more information on registering, please refer to the documentation for AWS IoT Core policies(<https://docs.aws.amazon.com/iot/latest/developerguide/iot-policies.html>).

**Step 3: Attach certificate to object**

In the final step, the generated certificate will be stored on the easyE4 device.

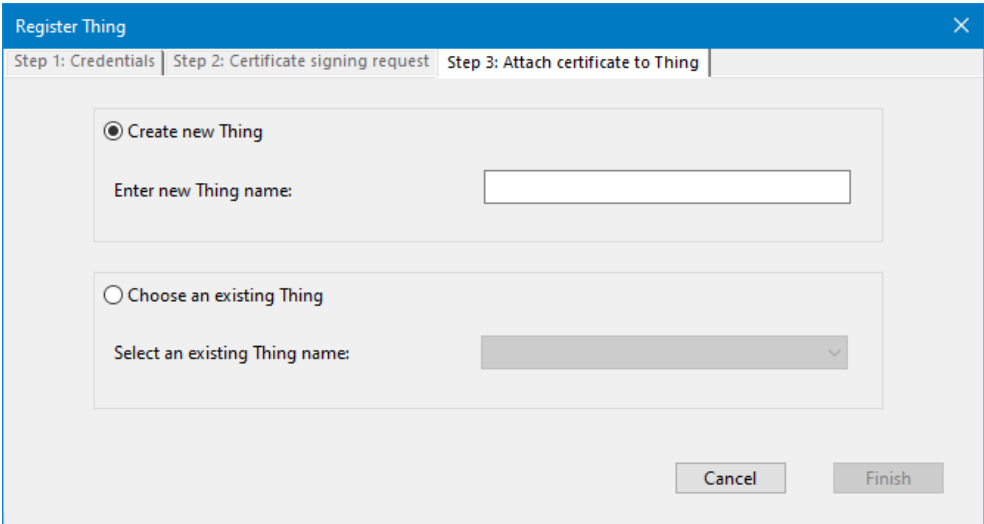


Fig. 366: easyE4 Wizard, certificate

Enter an object name (thing) for the easyE4 device in the cloud if the registration is new or select an existing object name from the list.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

When creating a new object, enter the object name in order to generate the object (thing) and link it to the certificate. This object name can contain letters, numbers, and special characters \_ - : and must not exceed 128 characters total.

#### Deregister object...

After a confirmation prompt, the easyE4 will be deregistered from AWS.

The certificate data will be deleted in the easyE4. However, the certificate data will be retained in AWS.

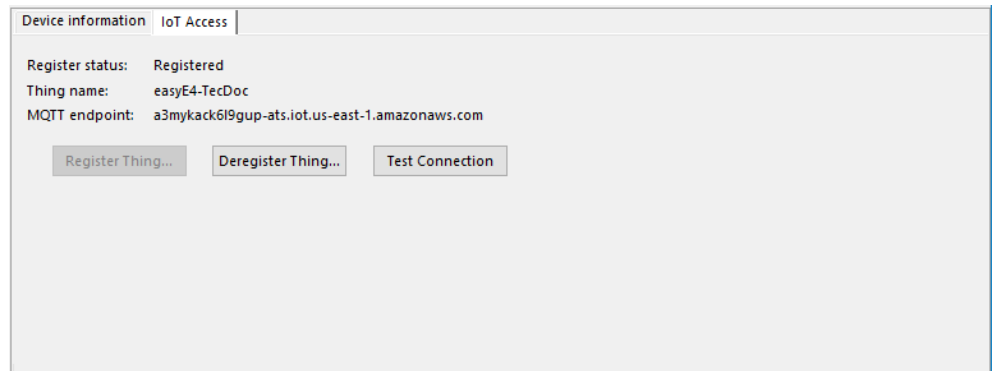


Fig. 367: Deregistering an easyE4

#### Test connection

Clicking on this button will make the system check whether the last transmission to the cloud was successful.

A message with the corresponding result will appear.

#### 10.17.3.2 Method 2: Using a Python script

This method is recommended when connecting a large number of easyE4 base devices in production runs.

If necessary, contact Eaton Support so that support staff can provide you with the script.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

```
aws_register_device_with_WebAPI.py
1  #!/usr/bin/env python3
2  """ aws_register_device_with_WebAPI.py
3
4  This script will register a easyE4 device at AWS via the devices web API.
5
6  Therefore, it is necessary to provide the connection details to the device:
7  - The ip address or the FQDN needs to be inserted in the variable "ip_addr"
8  - The username for the web API of the device needs to be inserted in the variable
9    "user_name"
10 - The password for the web API of the device needs to be inserted in the variable
11   "password"
12
13 To establish encrypted communication with the device, the root certificate must be
14 provided.
15 The root certificate must have the name easy_Root_V1.0.pem and must be stored in the
16 same directory as the script.
17 The device time needs to be set to a proper date and time, otherwise the communication
18 can't be established.
19
20 It is necessary to provide the credentials to connect to AWS.
21 The credentials can be provided as environment variables or as a file.
22 The file with the credentials must have the name credential without a file extension and
23 must be located in the same
24 directory as the script.
25
26 Please make sure that certificate parameters and AWS parameters are correct before
27 running this script!!!
28
```

Fig. 368: Excerpt from Python script

This Python script will register an easyE4 base device with AWS using the device web API.

In order to use this option, you will need to provide the connection details for the device:

- The IP address or fully qualified domain name (FQDN) needs to be assigned to the "ip\_addr" variable.
- The username for the device's web API needs to be assigned to the "user\_name" variable.
- The password for the device's web API needs to be assigned to the "password" variable.

In order to establish encrypted communications with the easyE4 base device, the root certificate needs to be specified.

This root certificate must be named easy\_Root\_V1.0.pem and be located in the same directory as the script.



The date and time on the easyE4 base devices must be configured correctly in order for communications to be successfully established.

You will need to provide the security credentials for connecting to AWS.

These security credentials can be provided as environment variables or as a file.

The file with the security credentials must be named credential without a file extension and be located in the same

directory as the script.

#### **ATTENTION**

Make sure that the certificate parameters and AWS parameters are correct before running the Python script!

## **10. Communication Connection to other devices easyE4**

### **10.17 Connecting to the AWS Cloud**

For more information on the Python script, please refer to Eaton AP050027EN.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

#### 10.17.4 MQTT test client

Using the AWS MQTT test client is a good way to monitor the MQTT messages passed in the AWS account.

easyE4 supports the following protocol for communications with a cloud server:

- MQTT over Transport Layer Security (TLS) in conformity with OASIS Standard Version 3.1/3.1.1

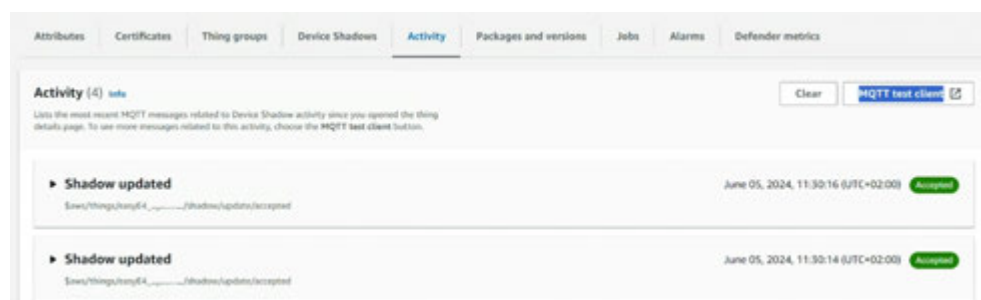


Fig. 369: Example: Access to the MQTT test client in the event of data transfer activities

For more information on exchanging data, please refer to the AWS MQTT documentation and Eaton AP050027EN.

#### 10.17.5 Update via AWS IoT Jobs

You can use AWS IoT Jobs to deploy updates to a large number of easyE4 devices securely and efficiently via remote access, which can make managing and maintaining your IoT infrastructure significantly easier.

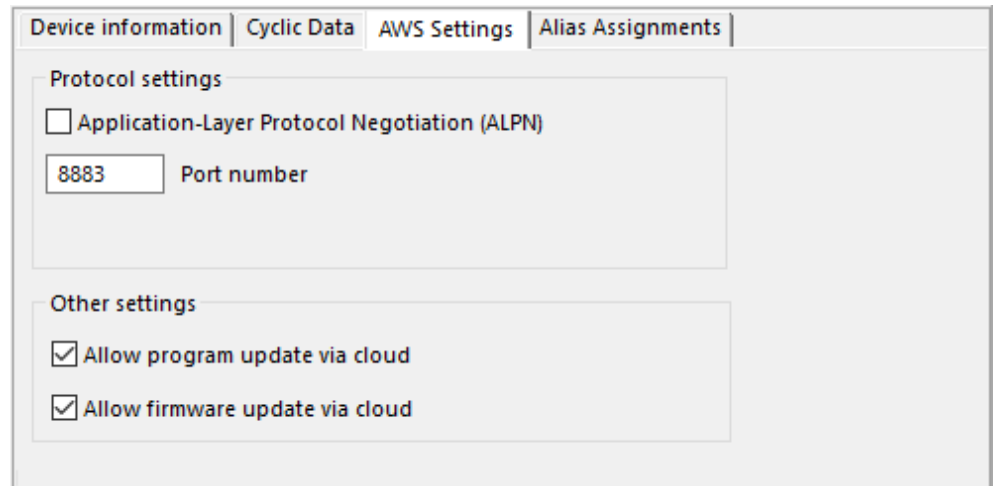
Within this context, Amazon Simple Storage Service (Amazon S3) is used to update the easyE4 base devices. More specifically, the device update files stored in a bucket there are transferred to the corresponding easyE4 base devices via AWS IoT Jobs.

##### Basic prerequisites for updates through the cloud

- The easyE4 must be connected to AWS.
- The S3 service must be available – please refer to → Section "Setting up the Amazon S3 AWS service as a work environment", page 807
- An S3 bucket must have been created, → Section "S3 bucket", page 824
- An IAM role with permissions for uploading files must be available, → Section "IAM role", page 797
- The .e80 project on the easyE4 must have the **Allow program update via cloud** option enabled under the AWS Settings tab.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud



The screenshot shows the 'AWS Settings' tab in a software interface. It contains two main sections: 'Protocol settings' and 'Other settings'. In 'Protocol settings', the checkbox for 'Application-Layer Protocol Negotiation (ALPN)' is unchecked, and the 'Port number' is set to 8883. In 'Other settings', both checkboxes for 'Allow program update via cloud' and 'Allow firmware update via cloud' are checked.

Fig. 370: Project view, AWS, AWS settings tab



If none of the options are enabled, access through the cloud will be disabled.

#### Steps for updates through the cloud

- Creating a job file and storing it in the S3 bucket
- Creating a job and running it in AWS IoT

#### 10.17.5.1 Updating firmware through the cloud

easyE4 base devices belonging to generation 09 with firmware version 2.30 or higher can be updated through the Web Client or through the AWS Cloud (in addition to updates made with microSD memory cards).

To identify the generation to which your easyE4 device belongs, please refer to the nameplate.

For this see also:

- Section "Updating firmware", page 136
- "easyE4 compatibility overview", page 881

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.

#### Additional prerequisite

In addition to the basic prerequisites for updates through the cloud, the following applies to firmware updates:


- The date and time set on the easyE4 base device must be later than the date of the desired firmware update.  
If this is not the case, the update will not be carried out. If the device has firmware version 2.30 or higher, a corresponding diagnostic ID will be output.



## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

#### Steps for updating the firmware on easyE4 base devices

- ▶ 1. Download the easyE4 firmware you want from the Eaton Download Center to your PC.  
Category: Firmware Updates; Product group: easy / RTD; Product: easyE4 base devices  
[Eaton Download Center](#)
- ▶ 2. Upload the **E4\_B\*.fw** firmware update file to your Amazon S3 bucket, → Section "Files in the S3 bucket", page 824.
- ▶ 3. Use a text editor to create a job file in JSON format.  
Enter the instruction details in this document.
  - "operation"  
This property is used to specify the job operation.  
For an easyE4 firmware update, enter the following: UpdateFirmware
  - "url"  
This property is used to specify the firmware update file's name and storage location in the S3 bucket.  
 The firmware update file in the S3 bucket can also contain a placeholder that will be replaced with the pre-signed URL if the easyE4 requests the job file.

#### Notation for job file for firmware update

```
{  
  "operation": "UpdateFirmware",  
  "url": "${aws:iot:s3-presigned-url:https://s3. XXX.amazonaws.com/XXX/E4_BXXXX.fw}"  
}
```

#### Sample job file for firmware update

The following example shows the content of a JSON job file used to update an easyE4 base device through AWS.

Please note that the url must specify the correct file storage location

```
{  
  "operation": "UpdateFirmware",  
  "url": "${aws:iot:s3-presigned-url:https://myawsbucket-tecdoc.s3.us-east-1.amazonaws.com/E4_B3204.fw}"  
}
```

Text highlighted in color: Can also be replaced with a placeholder

- ▶ 4. Upload this job file in JSON format to your Amazon S3 bucket.

You are now done with the required preparation steps.

- ▶ 5. Create and run the AWS IoT job  
→ Section "Creating a job", page 820
  - Create a job that will deploy the job file to your easyE4 base device.
  - Specify the easyE4 target device or a group of easyE4 devices.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

After the job is triggered, the easyE4 target device should receive the instructions from the job file and start downloading the firmware file via AWS. The easyE4 base device will then check the firmware and load the update.

- ▶ 6. Make sure to track the update's progress by checking the job run status in AWS.



Monitor the AWS IoT job's status in AWS IoT / Manage / Remote actions / Jobs and the relevant devices to make sure that the update is completed successfully.



Monitor the status of the devices to make sure that the update is completed successfully.



If an error occurs, you can view detailed logs and status reports in the AWS IoT console in order to diagnose and fix the corresponding issues.

#### 10.17.5.2 Updating easyE4 programs through the cloud

User programs on easyE4 base devices belonging to generation 08 or higher with firmware version 2.25 or higher can be updated through the AWS cloud (in addition to updates made with an online connection to easySoft 8 and updates made with microSD memory cards).

##### Additional prerequisite

In addition to the basic prerequisites for updates through the cloud, the following applies to program updates:

- Create the program file you want with easySoft 8 and save the .prg file on your PC (you can protect the .prg program with an update ID if you want).
- A Web-Visu and a program update through AWS cannot be enabled simultaneously within an easyE4 program.

The cloud connection of the easyE4 itself can be combined with Web-Visu.

If there is a NET project, you can configure a different setting for each NET station.

##### Steps for updating the program

- ▶ 1. Upload the .prg program file to your Amazon S3 bucket, → Section "Files in the S3 bucket", page 824.
- ▶ 2. Use a text editor to create a job file in JSON format.  
Enter the instruction details in this document.
  - "operation"  
This property is used to specify the job operation.  
For an easyE4 program update, enter the following: LoadProject
  - "url"  
This property is used to specify the program file's name and storage location in

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

the S3 bucket.

Notation for job file for program update

```
{
  "operation": "LoadProject",
  "url": "${aws:iot:s3-presigned-url:https://s3. XXX.amazon-
aws.com/XXX/XXXXXXX.prg}"
}
```

#### Sample job file for program update

The following example shows the content of a JSON job file used to update a program through AWS. Please note that the url must specify the correct file storage location

```
{
  "operation": "LoadProject",
  "url": "${aws:iot:s3-presigned-url:https://myawsbucket-tecdoc.s3.us-east-1.amazonaws.com/update_via_AWS.prg}"
}
```

Text highlighted in color: The job file's url



Copy the object URL from the uploaded program in the S3 bucket.

- ▶ 3. Upload this job file in JSON format to your Amazon S3 bucket.

You are now done with the required preparation steps.

- ▶ 4. Create and run the AWS IoT job
  - Section "Creating a job", page 820
- Create a job that will deploy the job file to your easyE4 base device.
- Specify the easyE4 easyE4 target device or a group of easyE4 devices.

After the job is triggered, the easyE4 target device should receive the instructions from the job file and start downloading the program file via AWS. The easyE4 base device will then check the program and load the update.

- ▶ 6. Make sure to track the update's progress by checking the job run status in AWS.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

#### 10.17.5.3 Creating a job

You can use the AWS IoT service to create a job by going to Manage / Remote actions / Jobs.

Within this context, a custom job is a remote operation that is sent to one or more devices that are connected to AWS IoT and run on it.

#### **Job = job target + job file**

The job target is the devices on which you want to run the job.

Meanwhile, the job file contains the commands that should be run on these job target devices.

#### Prerequisites

- The job target must already be available as an object (thing) in AWS IoT.
- The job document must be stored in the bucket.

#### Custom job properties

- ▶ In AWS, go to IoT Core / Manage / Remote actions / Jobs
- ▶ Select Create custom job

*AWS IoT/Manage/Remote actions/Jobs*

AWS IoT > Manage > Remote actions > Jobs > Create job

### Create job [Info](#)

Jobs define remote operations to send to and run on devices that are connected to AWS IoT. Create a custom job or a FreeRTOS over-the-air (OTA) update job.

**Job type**

- ☒ **Create custom job**  
Create a job to send an executable job file to one or more devices connected to AWS IoT.
- ☐ **Create FreeRTOS OTA update job**  
Send a request to acquire an executable job file from one of your S3 buckets to one or more devices connected to AWS IoT.
- ☐ **Deploy a single Software Package version**  
You will be redirected to SPC to start the deployment flow

Cancel Next

Fig. 371: Create Order

- ▶ Enter a unique name for the job.

You can use alphanumeric characters, hyphens, and underscores. Spaces are not allowed.

- ▶ Enter a description for the job so that it can be referenced later.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

The job will be managed in AWS IoT / Manage / Remote actions / Jobs with this name.

Meanwhile, the description will be available in the corresponding job details.

You can optionally assign tags.

*AWS IoT/Manage/Remote actions/Jobs*

The screenshot shows the 'Custom job properties' step in the AWS IoT console. The breadcrumb trail is 'AWS IoT > Manage > Remote actions > Jobs > Create job > Create custom job'. On the left, a sidebar shows 'Step 1 Custom job properties', 'Step 2 File configuration', and 'Step 3 Job configuration'. The main area is titled 'Custom job properties' with a 'info' icon. It contains a 'Job properties' section with a 'Name' field (placeholder 'job\_name', note: 'Enter a unique name that contains only alphanumeric characters, hyphens, or underscores. Job names can't contain any spaces.'), a 'Description - optional' text area (placeholder 'Description', note: 'The description can have up to 2,028 characters.'), and a 'Tags - optional' section. At the bottom right are 'Cancel' and 'Next' buttons.

Fig. 372: Creating a job, step 1

#### File configuration

The next step consists of selecting the job target and job document.

The job target is the devices on which you want to run the job.

Meanwhile, the job file contains the commands that should be run on these job target devices.

- ▶ Select the object (thing) you want as the job target.  
You can select more than one object.  
The object (thing) is the easyE4 that is registered in AWS IoT.

*AWS IoT/Manage/Remote actions/Jobs*

The screenshot shows the 'File configuration' step in the AWS IoT console. The breadcrumb trail is 'AWS IoT > Manage > Remote actions > Jobs > Create job > Create custom job'. On the left, a sidebar shows 'Step 1 Custom job properties', 'Step 2 File configuration', and 'Step 3 Job configuration'. The main area is titled 'File configuration' with a 'info' icon. It contains a 'Job targets' section with a note: 'A custom job is a remote operation that is sent to and runs on one or more devices connected to AWS IoT. Job targets are the things and thing groups that represent the devices that should run this job.' Below this are two sections: 'Things to run this job' with a 'Choose existing things' dropdown and a selected item 'easyE4-5ecDoc' with a close button, and 'Thing groups to run this job' with a 'Choose existing thing groups' dropdown. At the bottom right are 'Cancel' and 'Next' buttons.

Fig. 373: Creating a job, step 2, job target

- ▶ Select From file

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

#### AWS IoT/Manage/Remote actions/Jobs

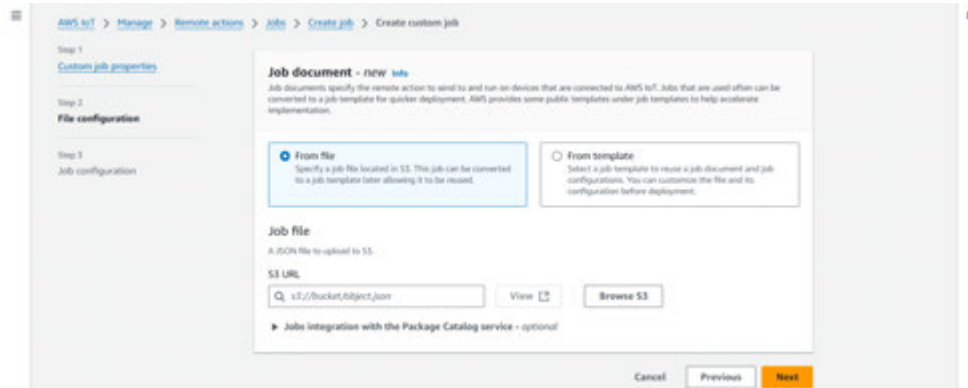


Fig. 374: Creating a job, step 2, job file

Now you will need the URL for the JSON job file. This job file must be stored in the S3 bucket, from where the URL can then be copied.

#### Amazon S3/Buckets/...

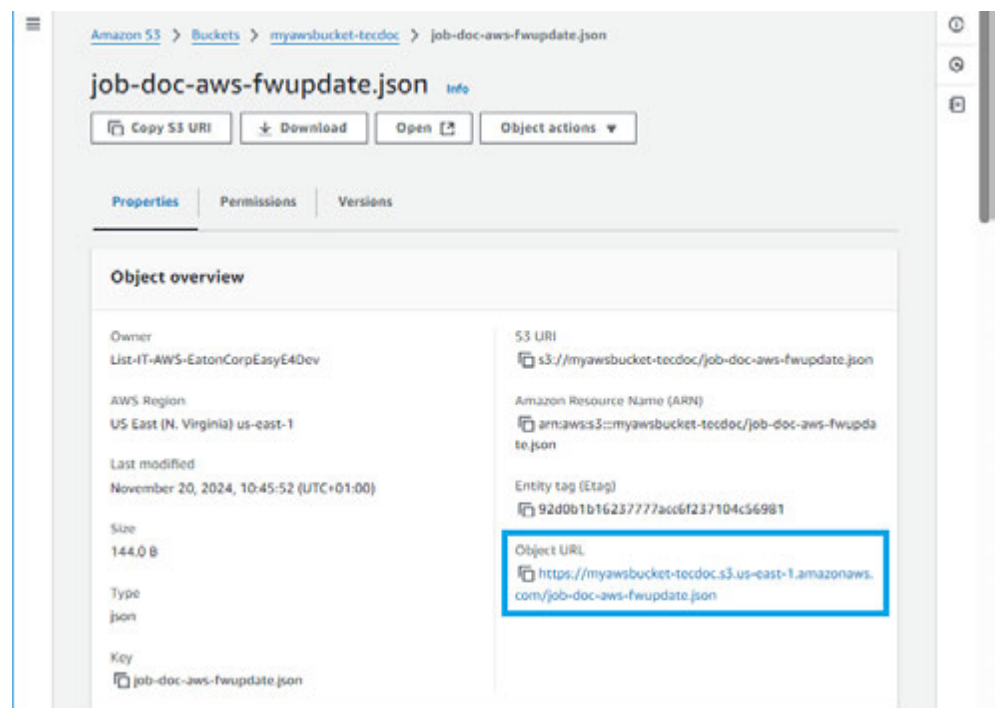


Fig. 375: Example of an object (thing) URL for the job file in the bucket

► Enter the object URL for the JSON job file.

If the URL is valid, the pre-signed URL will be entered and the Pre-sign resource URL configuration option will become available

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

#### AWS IoT/Manage/Remote actions/Jobs

The screenshot shows the 'Pre-sign resource URL configuration' step in the AWS IoT console. At the top, there's an 'S3 URL' field with a placeholder 's3://mybucket-test/doc/job-doc-aw' and buttons for 'View' and 'Browse S3'. Below this is a section for 'Jobs integration with the Package Catalog service - optional', with a checkbox for 'Pre-sign resource URL configuration' that is checked. A blue information box states: 'No code signing URL found. To use code signing, a code signing placeholder snippet is required in the job document.' Under 'Detected Pre-sign URLs', there's a table with columns 'Pre-sign URL type' and 'URL'. The table contains one row: 'Download' with the URL 'https://mybucket-test.s3.us-east-1.amazonaws.com/E4\_85204.fw'. The 'Pre-sign resource URLs' section explains that pre-signing URLs are for extra security and provides a 'Pre-signing role' dropdown menu (set to 'Choose an IAM role') and a 'Create role' button. At the bottom, there's a 'Timeout' section with input fields for 'Minutes' (set to 60) and 'Seconds' (set to 0), with a range of 1-60 minutes and 0-59 seconds. Navigation buttons 'Cancel', 'Previous', and 'Next' are at the bottom right.

Fig. 376: Creating a job, step 2, job file

► Enter your IAM role.

➔ This IAM role must have the permissions needed to download and/or upload files from/to the S3 bucket.

You can also configure additional optional settings.

#### Job configuration

The next step consists of configuring the job run type.

► Select Snapshot.

#### AWS IoT/Manage/Remote actions/Jobs

The screenshot shows the 'Job configuration' step in the AWS IoT console. On the left, a sidebar shows the progress: 'Step 1: Custom job properties', 'Step 2: File configuration', and 'Step 3: Job configuration' (which is highlighted). The main area is titled 'Job configuration' and contains a 'Job run type' section with two options: 'Snapshot' (selected with a radio button) and 'Continuous'. Below this is an 'Additional configurations - optional' section with expandable options: 'Rollout configuration', 'Scheduling configuration', 'Job executions timeout configuration', 'Job executions retry configuration', and 'Abort configuration'. At the bottom right are 'Cancel', 'Previous', and 'Submit' buttons.

Fig. 377: Creating a job, step 3, job configuration

Clicking on **Submit** will create the job and trigger it at the same time.  
The job will then be listed in AWS IoT / Manage / Remote actions / Jobs.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

The "status" will indicate the corresponding processing status.

#### 10.17.5.4 S3 bucket

A bucket is a container for any number of files that are stored in Amazon S3 for your AWS account.

To upload updates, you will first need to create a bucket in your AWS region. The required files for an update (job file and update file) need to be stored in this S3 bucket before they can be used with an easyE4.

Please note that the required S3 bucket should only be created once.



To download and/or upload files from/to the S3 bucket, you will need an IAM role with the appropriate permissions.

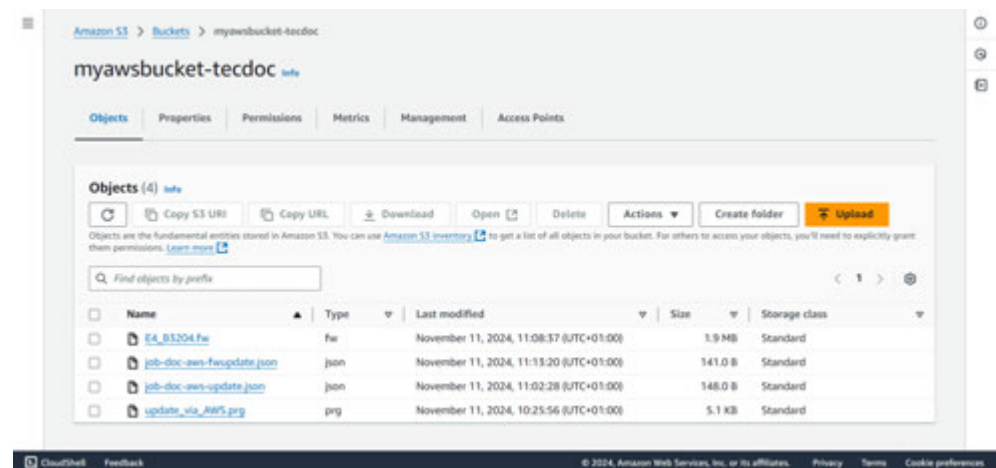


Fig. 378: Example of a bucket with two job documents and the corresponding files.

#### Files in the S3 bucket

As soon as you have created a bucket, you can store files in it.



The files provided by Eaton for download are signed.

- ▶ 1. Open your S3 bucket.
- ▶ 2. Click on **Upload**.

The Upload page will appear.



## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

Amazon S3/Buckets/**myawsbucket**/Upload

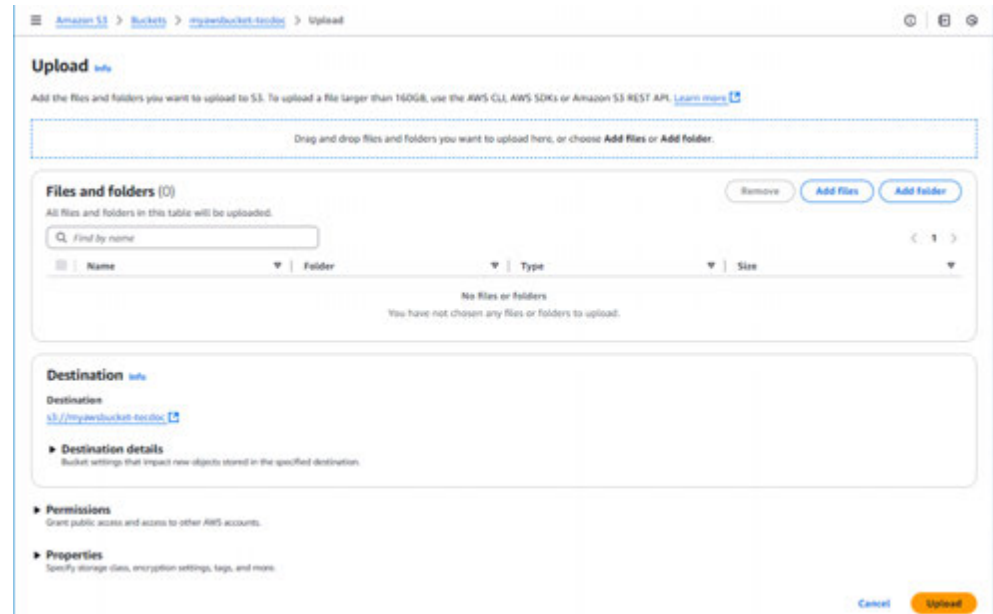


Fig. 379: Bucket example: Uploading files

- Drag and drop the files you want

or

- Click on Add file or Add folder and use your computer's file manager to select the files you want

Once you have stored all the files you need, you will need to upload them to the S3 bucket.

- Upload the files to the S3 bucket by clicking on **Upload**.

If you want to change the content of individual files in the S3 bucket, you will need to use the download function to store those files on your PC. Then, after editing the files on the PC, you will have to use the upload function to update them in the S3 bucket.

#### Creating an S3 bucket

- 1. Open the S3 service in AWS  
→ Section "Setting up the Amazon S3 AWS service as a work environment", page 807
  - 2. Click on the **Create bucket** button under Amazon S3 / Buckets
- ➔ Use the default settings.

#### General configuration

- Check whether the bucket is being created in your AWS region.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

- ▶ Select the General purpose bucket type.
- ▶ Enter the name you want for the bucket

The bucket name must be unique and conform to the rules for naming buckets. ([AWS - General purpose bucket naming rules](#))

#### Object ownership

- ▶ Select **ACLs disabled (recommended)**

You can optionally configure additional settings. However, these settings are not required for this use case.

- ▶ 3. Click on the **Create bucket** button to finish and create the bucket.

The created bucket will be shown under Amazon S3/ Buckets.

#### 10.17.5.5 Setting an update ID for cloud connections

You can use an update ID (also referred to as a "salt") to prevent unauthorized parties from uploading a program to a base device.

The update ID stored on the base device and the update ID stored in the program file must match!

When a program is uploaded, the update ID on the easyE4 base device and the update ID in the program will be compared with each other, and the system will prevent the program from being overwritten if the IDs do not match.

For easyE4 programs, the update ID is set when saving the program on a card / USB drive in the Project view. .

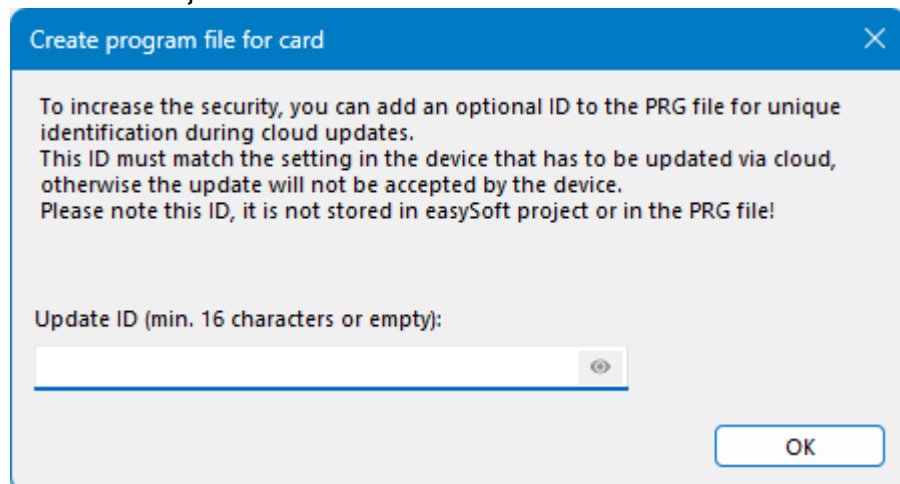


Fig. 380: Update-ID

Only the admin created in the .e80 project for the base device (in the "Webserver passwords and user names" dialog box accessed under the Webserver tab) will be able to set the update ID on the easyE4 device.

The program's update ID will still need to be set on the easyE4 device.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

The following URL will write the update ID to the device when using an encrypted web server:

```
"https://192.168.119.151/api/set/adm?op=salt&v1=codierte  
Update-ID"
```

**codierte Update-ID:** The update ID for the user program, encoded with URL encoding

To send the HTTP request, you will need a suitable command-line utility or script.

Example with update ID: **Ab1!cdefghijklm?**

```
"https://192.168.119.151/api/set/adm?op=salt&v1=Ab1%21cdefghijklm%3F"
```

### **10.17.6 AWS diagnostics**

There are two types of errors that can occur (on the side of the easyE4 and on the side of the AWS server) while the easyE4 device registration process is in progress.

#### **easyE4 errors**

If the following errors occur with the easyE4 FW during registration, check the corresponding potential reasons:

1. **Missing or invalid information**

If the AWS login information is incorrect, it will not be possible to carry out the next registration step.

If there is no active \*.e80 project with a valid AWS configuration on the easyE4 device, it will not be possible to establish a connection.

2. **Missing Internet access during registration**

If the connection to the Internet or AWS server is disrupted, or if a connection is not possible due to settings and the network environment, it is possible for requests to not be replied to in a timely manner.

This will result in wait times for the user before a request can be started again.

#### **AWS server errors**

These are the errors received from the AWS server for the various steps during registration.

These errors will primarily be generic HTTP error codes with additional error description texts.

For more information on these AWS server errors, please refer to the AWS documentation.

If you need additional support with registering a device, please refer to Eaton AP050027EN or contact Eaton Support.

#### **10.17.6.1 Messages for cloud computing diagnostics via easyE4**

In easySoft 8, messages regarding the cloud connection will be shown when in online mode.

## 10. Communication Connection to other devices easyE4

### 10.17 Connecting to the AWS Cloud

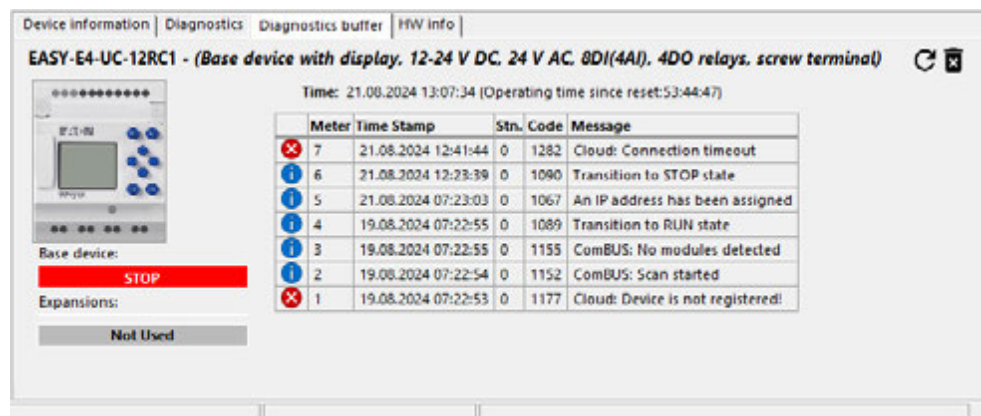


Fig. 381: In **Online** mode: Communication view, Diagnostics buffer tab

code	Message
1173 "Cloud: Dial-up started"	Cloud: Attempting to connect
1174 "Cloud: Connection established"	Cloud: Connection established
1175 "Cloud: Connection closed"	Cloud: Connection terminated
1176 "Cloud: Connection problems!"	Cloud: Connection problems
1177 "Cloud: Device not registered!"	Cloud: Device not registered in the cloud
1178 "MQTT: Buffer too small!"	MQTT: MQTT packet discarded. Insufficient network memory
1179 "Cloud: Configuration is not supported!"	Cloud: Configuration is not supported!
1181 "Cloud: Update too big!"	Cloud: Update too big!
1182 "Cloud: JSON string invalid!"	Cloud: Invalid JSON string!
1183 "Cloud: A Cloud job has been triggered."	Cloud: A Cloud job has been triggered
1184 "Cloud: Job is not supported."	Cloud: Job is not supported
1185 "Cloud: Project update via Cloud has been started."	Cloud: Project update through the cloud has been started
1186 "Cloud: Project update via Cloud has been finished successfully."	Cloud: Project update through the cloud completed successfully
1187 "Cloud: Project update via Cloud has failed."	Cloud: Project update through the cloud failed

## **10.18 Modbus TCP**

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.

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 and lower-level control systems
- Platform-independent communication
- Communication with devices that are not part of the easyE4 series
- Setting the device clock at runtime  
with firmware version 1.21 or higher: → page 855;  
This option can be disabled on easySoft 8 on Version 7.30 and higher

easyE4 can be configured both as a Modbus TCP client and as a Modbus TCP server simultaneously in the same project.

Each Modbus TCP client and Modbus TCP server knows the Modbus TCP map in order to be able to exchange data for communication. easyE4 exchanges the data by using function codes to assign all the values for the request(s) to easyE4 base device operands.

### **Modbus TCP map**

Information about the Modbus TCP Map can be found in easySoft 8 Help.

### **easyE4 as Modbus TCP Server**

easyE4 can be configured as a Modbus TCP server under *Project view/Modbus Server tab*.

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.

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

#### easyE4 as Modbus TCP Client

Only available on firmware version 1.30 or higher.

If a Modbus TCP server module is connected to an easyE4 base device on the Project view work pane by dragging and dropping it there, the easyE4 base device will automatically become a Modbus TCP client.

Up to 16 Modbus TCP server modules can be configured for an easyE4 base device.



Firmware version 2.25 and lower supports a maximum of four Modbus TCP server modules.

#### easyE4 response times

When working with applications where timing is critical, it is important to consider the response times for Modbus TCP communication.

The following performance conditions apply to firmware version 2.30 and higher:

1. A maximum of 200 Modbus TCP client requests can be sent per second.
2. The processing time for an individual request can be configured with a minimum possible response time of 30 ms.
3. The computation time for a single application program cycle will limit the Modbus cycle accordingly.
4. The response time of the external server (slave) will limit the Modbus cycle for a connection accordingly.

When using fast external servers (slaves), an application cycle time that is much shorter than 5 ms, and multiple jobs configured in parallel with their own configured connection each, a throughput of up to 200 requests per second can be achieved depending on the specific circumstances.

#### Parallel processing example

If an easyE4 functioning as a Modbus TCP client controls a four Modbus TCP servers then the easyE4 will be able to transmit its requests simultaneously and directly process the responses that arrive at the same time.

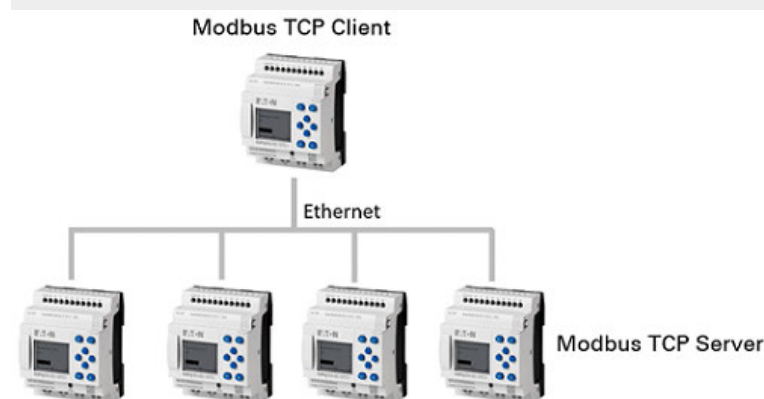


Fig. 382: Example: easyE4 as Modbus TCP Client

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

#### Use case with a single easyE4 functioning as a Modbus TCP server handling two Modbus TCP clients.

Use case with a single easyE4 functioning as a Modbus TCP server handling two Modbus TCP clients.

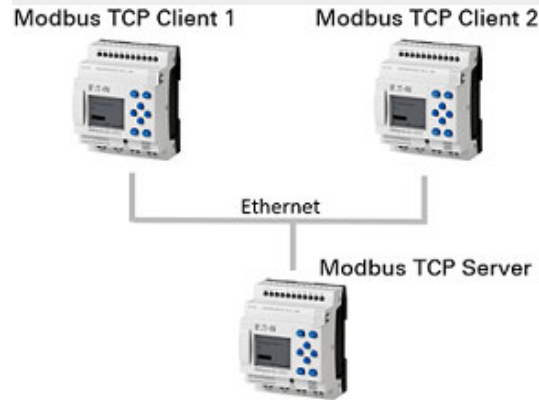


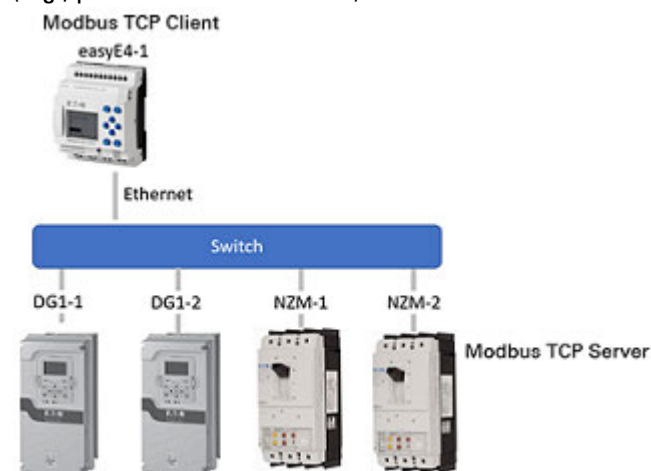
Fig. 383: Example: easyE4 as Modbus TCP Server

#### 10.18.1 easyE4 as a Modbus TCP client

Only available on easySoft Version 7.30 or higher.

Only available on firmware version 1.30 or higher.

By using the Modbus TCP server module, easyE4 can be used with the functionality of a higher-level Modbus TCP client. The Modbus TCP server module is a placeholder for hardware that can be addressed by easyE4 via a separate communication channel. Automation components that feature the corresponding communication capabilities can be connected as a Modbus TCP server to easyE4. easyE4 can perform control activities and evaluate and display diagnostic data and other process data (e.g., position data for a drive).



A maximum of 16 Modbus TCP server modules are allowed per base device when using firmware version 2.30 or higher.



## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP



A maximum of four Modbus TCP server modules are allowed when using firmware version 2.25 or lower.

As soon as a Modbus module is dragged from the catalog and dropped on the work pane next to the bottom of an easyE4 base device, the base module's Modbus TCP client functionality will be activated.

This means that easyE4 will act as a Modbus TCP client and the Modbus module will represent the Modbus TCP server as a "virtual" module. Modbus TCP servers can be automation components that mostly perform control activities or work by themselves and occasionally send status data for display or statistics to the Modbus TCP client. Examples include variable frequency drives (e.g., DG1, PowerXL, 9000X), circuit-breakers (e.g., NZM), and other easyE4 base devices.

The Project view can be used to configure frames that will be transmitted cyclically within fixed time intervals. Function codes need to be defined for this purpose under the "Cyclical data" tab in the Project view.

For acyclical (i.e., frames that are only triggered once), use the MC function block (Acyclical Modbus client request).

Modbus modules are identified with "MSn", e.g., MS1.

The configuration will be saved with the .e80 file.

When you select the Modbus module on the work pane, the tabs that can be used to configure the parameters for communicating with the Modbus TCP server will be shown.

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

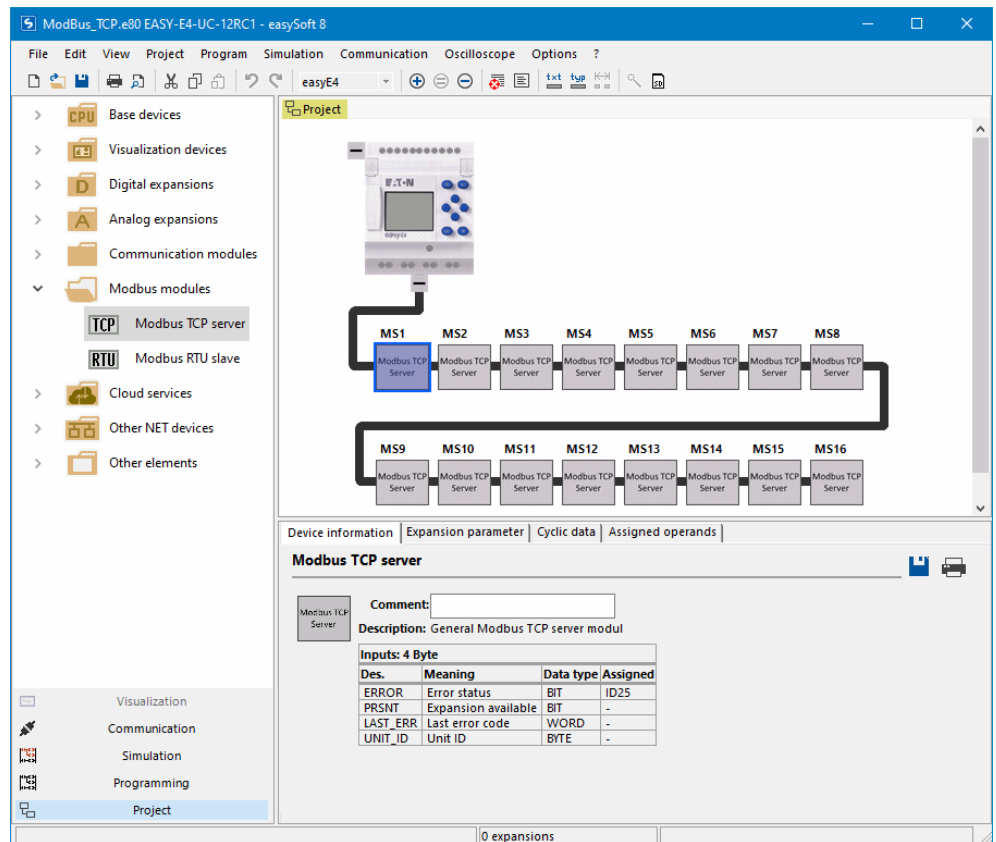


Fig. 384: Work pane with base device and Modbus TCP server module

#### Device information tab

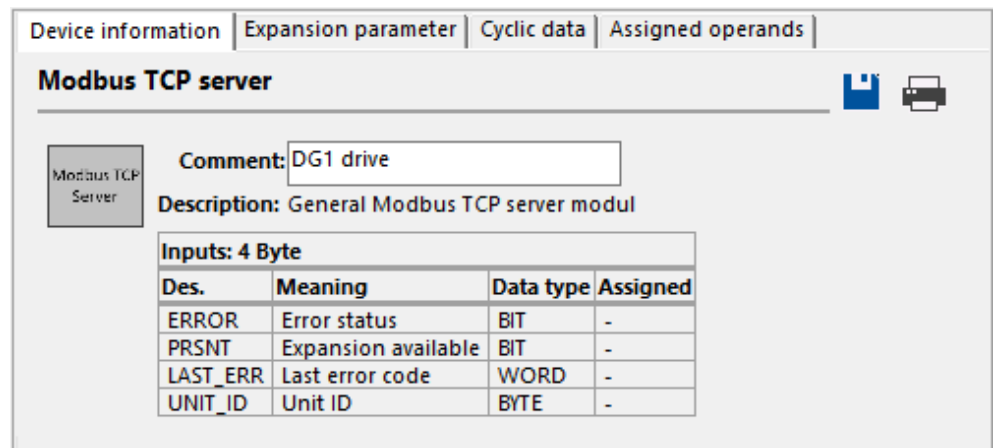


Fig. 385: Device information tab

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

#### Expansion parameter tab

The Expansion parameter tab can be used to configure the relevant Modbus TCP parameters

for the Modbus module, i.e., for the Modbus TCP server.

Modbus communication occurs either via a fixed IP address or through the DNS name in the domain name system (DNS).

The screenshot shows the 'Expansion parameter' tab for Modbus TCP servers. It is divided into two main sections: 'IP settings' and 'Modbus settings'. In the 'IP settings' section, there are two radio buttons: 'IP address' (which is selected) and 'DNS name'. The 'IP address' field shows '0 . 0 . 0 . 0'. In the 'Modbus settings' section, there is a 'Byte order' section with three radio buttons: 'Big-endian' (selected), 'Little-endian', and 'Twisted'. To the right of this, there are two input fields: 'Modbus service port' with the value '502' and 'Timeout for server response [ms]' with the value '3000'. At the bottom of the 'Modbus settings' section, there is an unchecked checkbox labeled 'Auto decrement on all addresses'.

Fig. 386: Expansion parameter tab for Modbus TCP servers Expansion parameter tab for Modbus TCP servers Expansion parameter tab for Modbus TCP servers Expansion parameter tab for Modbus TCP servers

#### IP address

This is where the IP address of the Modbus TCP server is set. The default setting is: 0.0.0.0.

The IP address must have the same network as the Modbus TCP client (the easyE4 base device). Please refer to → "Basic information on assigning IP addresses", page 117 as well.

#### DNS name

If this option is enabled, easyE4, as a Modbus TCP client, will use a DNS name to communicate with the Modbus TCP server.

The DNS server will resolve the DNS name and replace it with the current IP address.

The field will be blank by default.

Naming convention for DNS name:

If you are using ASCII characters, the name must not exceed 63 characters. If you use characters outside the ASCII range, you can use fewer than 63 characters as needed, as all characters are converted to Punycode internally.

Special characters : / ? # [ ] @ ! \$ & ' ( ) \* + , ; = are not allowed. ASCII characters that are not printed out, such as spaces, newlines, and tabs, are not allowed either.

### **Byte order**

The byte order setting defines how the values from the Modbus communication will be interpreted. Modbus big-endian (Motorola format) will be used normally. If the Modbus client or master sends the data in Intel format, you will need to switch the setting to little-endian. The Twisted checkbox can be additionally enabled to get Big-EndianTwisted or Little-EndianTwisted for the data interpretation..

☒ Big-Endian (default)

☐ Little-Endian

☐ Twisted

### **Modbus service port**

The value range is 1 to 65535. The default port address should be 502.

### **Server response timeout [ms]**

This parameter is used with cyclical data transfers to specify how long the system should wait for a response from the Modbus server or slave. The value range is 1000 to 10000 ms. The default value is 3000 ms. The time can be set in steps of 10 ms. If the set time is exceeded, easyE4 will assume that communications have been disrupted.

If the ☐ Clear register on timeout option is not enabled under the Cyclical data tab, the last value that was transmitted by the server or slave will be retained.

If the option is enabled, the easyE4 will reset the operand back to its initial state of "0".

For cyclical data transfers, the minimum update rate can be set for each function code in the Update Rate column under the Cyclical data tab.

☐ **Auto decrement on all addresses**

Only available on firmware version 1.40 or higher.

This option is disabled by default.

The value range is 1 to 65535. The default is ID 1.

As specified in the Modbus specification, the start address of a data packet minus 1 (address offset) will be transferred.

Older devices still use this address range and will interpret the transmitted address with an offset of +1.

On newer devices, such as easyE4 devices, addresses start from start address 0 instead.

If you want to configure Modbus communications with a Modbus server/slave for the easyE4 base device and the corresponding addresses start from start address 0, then the Auto-decrement on all addresses option must be disabled. The Modbus client's/master's address will be sent without any additional conversion steps and the address will be the exact same on the Modbus server/slave.

10. Communication Connection to other devices easyE4

10.18 Modbus TCP

If you want to configure Modbus communications with a Modbus server/slave for the easyE4 base device and the corresponding addresses start from start address 1, then you will need to enable the Auto-decrement on all addresses option for the Modbus server/slave. To ensure that addresses are used correctly, an offset bit of 1 will be subtracted from all the addresses for the Modbus client/master.

If the Auto-decrement on all addresses option were disabled, the easyE4 would, for example, send address 1 so that address 2 on the Modbus map would be selected for the Modbus server/slave if the latter were a device for which addresses start with start address 1.

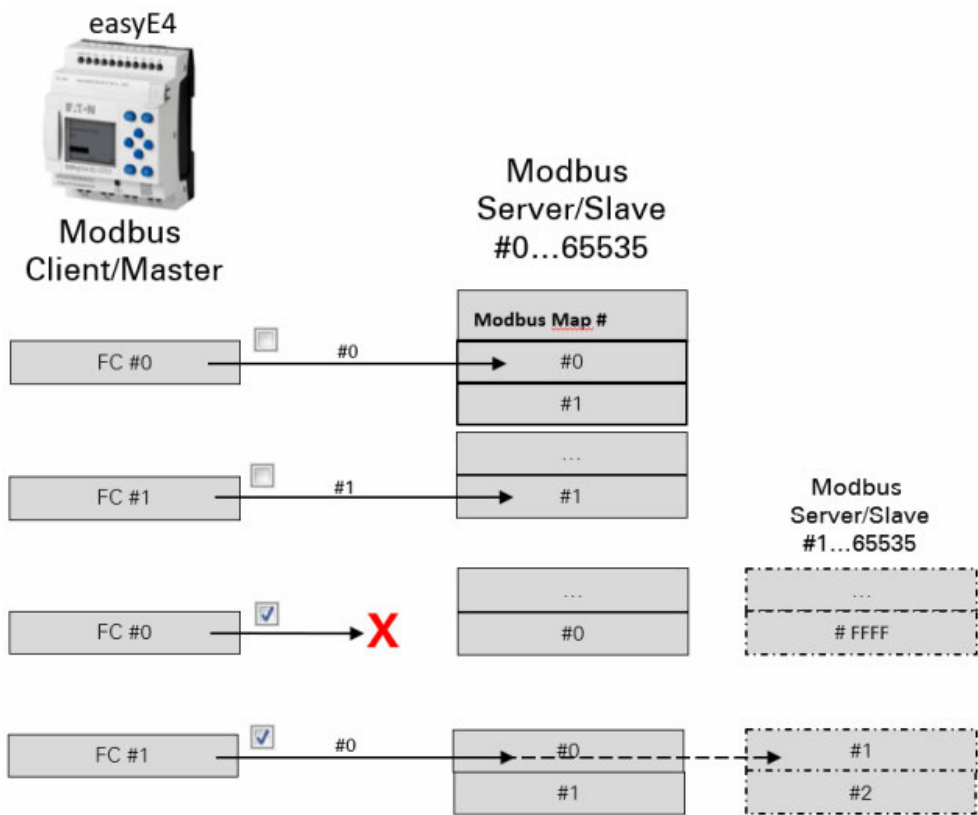


Fig. 387: Address range addressing:

- ☐ without the auto-decrement option for all addresses
- ☒ with the auto-decrement option for all addresses
- X** plausibility check reports an error.

Cyclical data tab

You can use the settings under the Cyclical data tab to define the type of access to the Modbus TCP map that the selected MS... Modbus TCP server module should have.

Within this context, you can define which function code will be used to read and/or write

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

which Modbus TCP server module I/O points. The I/O points will then be found under the Assigned operands tab, where they can be linked to base device operands.

The first five columns describe the Modbus TCP server exclusively and are used to put together the frame – please refer to → "Additional information for use", page 893

By default, the last Modbus TCP server response to the request will be assigned to the easyE4 operands and kept until the next request.

As a Modbus TCP client, easyE4 will send the request to the selected Modbus TCP server module. The selected function code will determine whether easyE4 reads or writes, whether it will do so for one or more elements, and whether the elements are of data type BIT or WORD. The elements in the server's Modbus TCP map will be read starting with the start address in the Modbus TCP server module's I/O points. Meanwhile, the Modbus TCP server module I/O points will be written to the server's Modbus TCP map starting with the start address.

The I/O points in the Modbus TCP server module will be automatically created with the function code definitions.

After the function codes are defined, they can be found in the Assigned operands tab.

#### Modbus module Project view/Cyclical data tab

Device information | Expansion parameter | Cyclic data | Assigned operands

☐ Skip all requests

☐ Reset registers on timeout

General					1. request		2. request (FC23: write)			
Unit ID	Update rate	Function code	Start addr.	No. of items	Op. class	Start addr.	No. of items	Op. class		
1	255	100	FC1 - Read Coils	2	I				-	
2	255	100	FC2 - Read Discrete Inputs	20	I				-	
3	255	100	FC3 - Read Multiple Holding Registers	222	I	1A16			-	
4	255	100	FC4 - Read Input Registers	40	I				-	
5	255	100	FC5 - Write Single Coil	666	Q				-	
6	255	100	FC6 - Write Single Holding Register	65535	Q	QA16			-	
7	255	100	FC15 - Write Multiple Coils	10	Q				-	
8	255	100	FC16 - Write Multiple Holding Registers	15	Q	QA16			-	
9	255	100	FC23 - Read and write Multiple Registers	25	I	1A16	0	1 QA16	-	
10										

Fig. 388: Cyclical data tab with sample function codes that have been configured and added range boxes

- ① Modbus TCP server tab
- ② easyE4 base device operands

#### ☐ Skip all requests

If you enable this option, the base device will ignore the following function codes in the table and not send them. This option can be helpful when configuring a project or at the beginning of tests if you already know that a Modbus TCP server will not be reachable because it has not yet been installed even though it is already configured.

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

#### ☐ Clear register on timeout

If you enable this option, the operand will be switched to the initial state of "0" when the Modbus TCP server response to a read or write request takes longer than a defined time.

This time is defined in the Expansion parameter tab for the selected MS... Modbus TCP server module – please refer to→ "Server response timeout [ms]", page 836 as well.

#### Unit ID

Some Modbus TCP servers evaluate the unit ID in order to forward to submodules (e.g., Modbus RTU modules). One example of this is when multiple Modbus TCP servers are working through a Modbus TCP bridge with the same IP address.

In this case, Modbus TCP server read and write access is only possible with the unit ID. For Modbus TCP, the default setting is 255.

Accordingly, please check whether the Modbus TCP server being used is evaluating the unit ID and, if necessary, set the required unit ID.

#### Update Rate

The update rate defines the time intervals at which requests will be sent to the Modbus TCP server. The value range is 30 to 10,000 [ms], with the default value being 100 [ms]. Moreover, the value can be changed in increments of 10 [ms]. The update rate should not be too low so as to ensure that the Modbus TCP server's communication load will not become excessive. Please note that the actual real-life time intervals may deviate from the setting depending on the easyE4 base device's load.

#### Function Code

easyE4 supports the following function codes as a Modbus TCP client:

FC <sub>dec</sub>	Function description	Function code <sub>hex</sub>
FC1	Read Coils	0x01
FC2	Read Discrete Inputs	0x02
FC3	Read Multiple Holding Registers	0x03
FC4	Read Input Registers	0x04
FC5	Write Single Coil	0x05
FC6	Write Single Holding Register	0x06
FC15	Write Multiple Coils	0x15
FC16	Write Multiple Holding Registers	0x10
FC23	Read and Write Multiple Holding Registers	0x17

#### Start addr.

The address of the first Modbus TCP server element that should be written to or read. The value range is 0 to 65535.



Keep the 0-based address system in mind.  
If the address range does not match the Modbus server's address range because the former starts from 0 and the latter from 1, you will need to use an offset.  
In this case, you will need to set the start address to the original value minus 1.  
Alternatively, the option ☒ Auto-decrement on all addresses can be activated with a checkmark.

### Amount Items

You can use the Amount Items field to define a contiguous range in order to accelerate communication, as only a single frame request will be needed for a large number of elements.

Depending on the specific function code, "element" may refer to various data formats. The element will be of data type BIT in the case of the following function codes: FC1, FC2, FC5, FC15. Meanwhile, it will be of data type WORD in the case of the following function codes: FC3, FC4, FC6, FC16, FC23.

### Operand class

The operand class limits the assignment of Modbus TCP server data to easyE4 base device operands.

The Modbus TCP server's registers are automatically assigned to Modbus TCP server module I/O points. Based on the selected operand class, they will be available in the subregisters under the Assigned operands tab: bit inputs, bit outputs, analog inputs, analog outputs, or diagnostic alarms. You can then assign them to easyE4 base device operands in the Assigned operands register.

FC <sub>dec</sub>	Available operand classes
FC1	I, ID
FC2	I, ID
FC3	IA16, IA32
FC4	I, ID, IA16, IA32
FC5	Q
FC6	QA16, QA32
FC15	Q
FC16	QA16, QA32
FC23 read	IA16, IA32
FC23 write	QA16, QA32

### Example: Function code FC4

Function code FC4, Read Input Registers, is available for a value assignment.

I can be selected as the assigned operand class. After this, the register with start address 40 will be automatically assigned to a Modbus TCP server module input register. The corresponding bits will then be available in a bitwise manner. In an additional step, you can go to the Assigned operands tab and assign the 16 bits in the



10. Communication Connection to other devices easyE4

10.18 Modbus TCP

Modbus TCP server module input register to the easyE4 base device's input operands. This means, for example, that you can select the 1st bit and the 5th bits and the 15th bit and assign them to input operands I11, I12, and I13.

If you select ID as the operand class, the register with start address 40 will be automatically assigned to a Modbus TCP server module input register as well. The corresponding bits will then be available in a bitwise manner too. In this case, however, you will only be able to assign them to the easyE4 base device diagnostic alarms in a bitwise manner under the Assigned operands tab.

If you select IA16 as the operand class, the register with start address 40 will be automatically assigned to a Modbus TCP server module input register as well. However, it will not be available in a bitwise manner. You will, however, be able to assign it to the operand for an easyE4 base device analog input under the Assigned operands tab.

If you select IA32 as the operand class, two consecutive registers starting from start address 40 will be merged into a 32-bit process value. You will then be able to assign this value to the operand for an easyE4 base device analog input under the Assigned operands tab.

Modbus TCP Client

Expansion parameter tab  
for Modbus TCP servers  
Expansion parameter tab  
for Modbus TCP servers

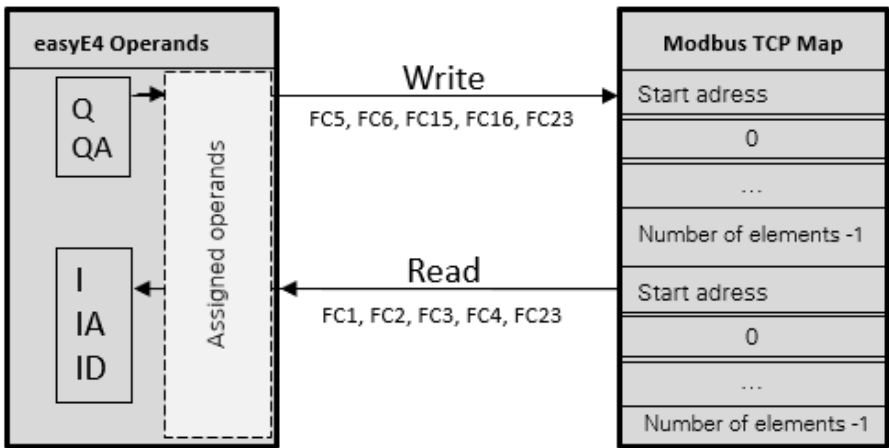


Fig. 389: Overview of cyclical data function codes

Assigned operands tab

Modbus TCP server module I/O points that you want to use in the program need to be assigned to easyE4 base device operands. easyE4 will organize all Modbus TCP communication data in words. A comparison with the easyE4 base device operands will not be carried out until the operands are assigned, in which case the corresponding type will be converted if necessary.

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

On the left side, this tab shows the Modbus TCP server module's I/O points. Before, this, however, it is necessary for requests to have been defined under the Cyclical data tab. For function codes with a read request, the I/O points will be identified with Rxx. For function codes with a write request, the I/O points will be identified with Wxx.

The right side will show the easyE4 base device's operands. In order to be able to use the I/O points in the program, the Modbus TCP server module's I/O points must be assigned to easyE4 base device operands. You can carry out this assignment in easySoft 8.

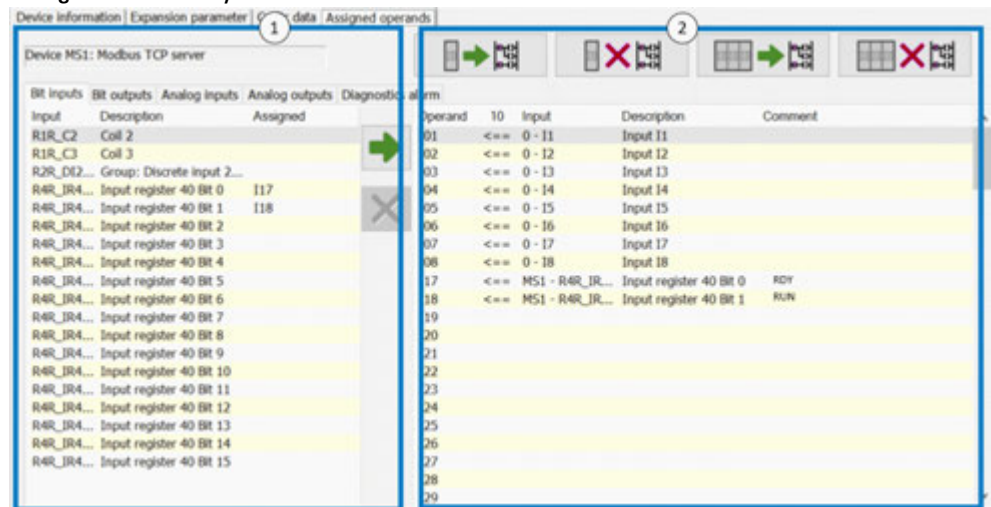


Fig. 390: Assigned operands tab after defining FC1, FC2, and FC4; bit inputs R4R\_IR40x0 and R4R\_IR40x1 have already been assigned to base device operands I17 and I18.

- ① Modbus TCP server I/O points
- ② easyE4 base device operands

### Grouping

If 50 or more I/O points are generated by a function code, a group entry will be generated on the left side of the table. Double-clicking on this group entry (e.g., on R2R\_DI20-DI69) will open a dialog box where you can then double-click on an entry (e.g., on R2R\_DI20) to assign it to the previously selected base device operand (e.g., I19).

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

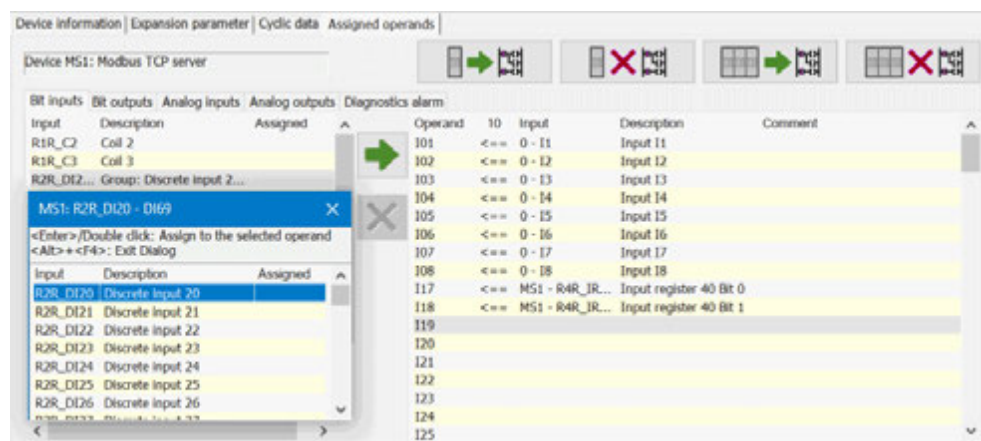


Fig. 391: Assigned operands tab; bit input R2R\_DI20 has already been assigned to base device operand I19.

#### Other

##### Using Modbus TCP server modules on the work pane

- If a Modbus communication module is deleted or cut from the work pane, all the Modbus registers in the Assigned operands tab will be cleared.
- If a Modbus communication module is copied and pasted, all the parameters from the original's Expansion parameter tab will be copied and pasted as well. The original's assigned operands will not be copied, however.
- If an easyE4 base device with a Modbus communication module is copied and pasted, the entire Modbus TCP configuration will be copied and pasted as well, including the corresponding expansion parameters and assigned operands.
- Modbus communication modules will not appear in the order list.
- Modbus communication modules will appear in the cross-reference list. Clicking within the cross-reference list will take you to the corresponding Modbus TCP server module.
- The Modbus TCP information tab will appear only if you click on the blue bar in between.

##### Modbus TCP information tab

To show the Modbus TCP information tab, click between the base device and the communication module.

Shows the number of Modbus TCP server modules and the payload size in bytes. Also shows the total of all assigned operands to Modbus TCP servers, diagnostic messages included.

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

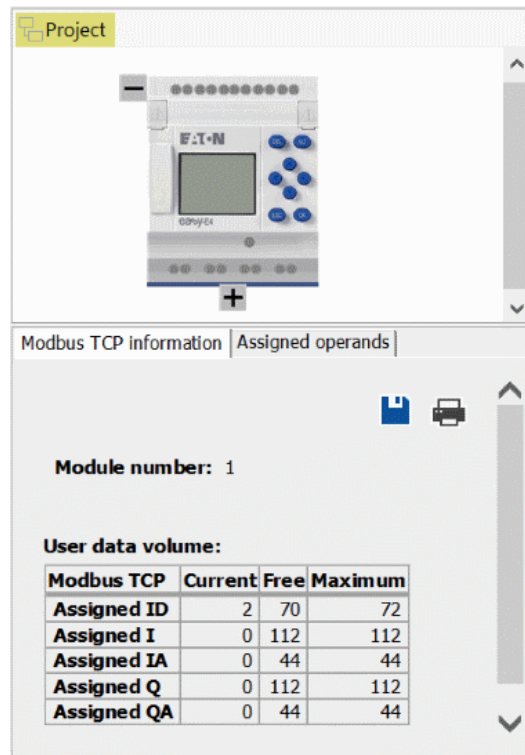


Fig. 392: Modbus TCP information tab

#### 10.18.2 easyE4 as a Modbus TCP server

Only possible with easySoft 8.

The easyE4 base device will be activated as a Modbus TCP server if you enable the Modbus TCP server option under the *Project view/Modbus Server tab* and the data for Modbus TCP communication is enabled..

##### 10.18.2.1 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 control relay easyE4 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:

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

- 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)

easyE4 supports the following function codes as a Modbus TCP server:

FC <sub>dec</sub>	Function description		Function Code <sub>hex</sub>
FC1	Read Coils	Used to read outputs	0x01
FC2	Read Discrete Inputs	Used to read inputs	0x02
FC3	Read Multiple Holding Registers	Used to read multiple input registers	0x03
FC4	Read Input Registers	Used to read input registers	0x04
FC5 <sup>1)</sup>	Write Single Coil	Used to write to exactly one output	0x05
FC6	Write Single Holding Register	Used to write to a single output register	0x06
FC15 <sup>1)</sup>	Write Multiple Coils	Used to write to multiple outputs	0x15
FC16	Write Multiple Holding Registers	Used to write to multiple output registers	0x10
FC23 <sup>1)</sup>	Read and Write Multiple Holding Registers	Used to read or write from/to multiple word output registers	0x17

1) Only available for Modbus TCP clients or Modbus RTU master in easyE4

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:

#### Read Coils 0x01:

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

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)

Tab. 135: 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)

Tab. 136: Response-PDU

Function Code	1 byte	0x01 ;Read Coils
Byte 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 bit 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.

#### 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)

Tab. 137: 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)

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

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)

Tab. 138: Response-PDU

Function Code	1 byte	0x02 ;Read Discrete Inputs
Byte Count	1 byte	N
Output values	n* 1 byte	Value

$n = \text{Number of inputs read} / 8$

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.

### Read Holding Registers 0x03:

Function 0x03 reads internal registers (e.g., marker words in the easyE4) word by word.

Tab. 139: 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

Tab. 140: Response-PDU

Function Code	1 byte	0x03 ;Read Holding Registers
Byte Count	1 byte	A value of = 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. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

#### 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.

Tab. 141: 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)

Tab. 142: Response-PDU

Function Code	1 byte	0x04 ;Read Input Registers
Byte Count	1 byte	A value of = 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.

### Write Single Registers 0x06:

This function writes 16 bites to a register ((NET) marker word in the easy)

Tab. 143: 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 850 Request PDU)

In other words, the response is identical to the corresponding request and is used for confirmation purposes only.

### Write Multiple Registers 0x10:

This function writes  $n * 16$  bits to N registers ((NET) marker words in the easyE4)

Tab. 144: 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

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

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 850 Request PDU)

Tab. 145: 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)

#### 10.18.2.2 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 TCP map" table) or
- Un-enabled\*

Exception Code 03 = The number of outputs is not  $\geq 0x0001$  and  $\leq 0x07D0$

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 TCP map" table) or
- Un-enabled\*

Exception Code 03 = The number of inputs is not  $\geq 0x0001$  and  $\leq 0x07D0$

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

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 TCP map" table) or
- Un-enabled\*

Exception Code 03 = The number of inputs is not  $\geq 0x0001$  and  $\leq 0x07D0$

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)
- Not defined\* or
- Un-enabled\*

Exception Code 03 = The number of inputs is not  $\geq 0x0001$  and  $\leq 0x07D0$

Exception Code 04 = (Error in server) n.a.\*\*

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

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)
- Not defined\* 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)
- Not defined\* or
- Un-enabled\*

Exception Code 03 = The number of registers is not  $\geq 0x0001$  and  $\leq 0x007B$

OR

Number of bytes  $\neq$  Number of registers  $\times 2$

Exception Code 04 = Error when attempting to write to the registers\*\*

## 10. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

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. Communication Connection to other devices easyE4

### 10.18 Modbus TCP

#### Setting the device clock at runtime

Starting with firmware version 1.21.

When configured as a Modbus TCP server, easyE4 will provide device clock information with the date and time function codes via Modbus TCP data communication. easyE4 will automatically fill this register with the current data from the device clock. A Modbus TCP client can read the data from the Modbus TCP map and write to it as well.

If a Modbus TCP client writes a date, the device clock will set itself to the date and time and then switch back to the mode for filling the registers with data from the device clock that is now set.

There are two options for the writing operation:

1. RTC format in Modbus TCP map register 5000 – 5005
2. GALILEO format Modbus TCP map register 5006 – 5009

The following function codes can be used to write to registers 5000 – 5009:

FC6	Write Single Holding Register
FC16	Write Multiple Holding Registers

#### Note for GALILEO users



In order to prevent the easyE4 device time from being written cyclically, do not permanently set control bit 11 in the Control system tag's 1st data word in GALILEO.

There is the option of preventing writing by enabling the Clock setting locked option..

#### 10.19 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.

To make visualization of projects much more convenient, Remote Touch Displays or HMIs are available as an operating panel.

These displays provide an inexpensive solution to remote visualization.

##### 10.19.1 easyE Remote Touch Display

You have extended visualization solutions for the easyE4 control relay available in the form of the Standard and Advanced variants of the easyE Remote Touch Display easyE4 (RTD).

The display and operating elements of the easyE4 base device are displayed in color on the full-color display of the RTD. With more than 65,000 possible colors, texts, values, parameters or even graphics are displayed. This enables the status of the device to be established quickly.

The RTD can be installed in a control panel door or directly on systems, among other locations. In addition, RTDs require little space and are intended for front flush mounting, meaning that they are inserted into the corresponding enclosure from the front.

The touch display is connected to the base device of the easyE4 by means of a Plug&Play solution with a standard RJ45 Ethernet cable. Setting up of the RTD is assisted by an Assistant (Setup Wizard). The Remote Touch Display provides menu navigation in various languages.

Access permissions can be set up for specific user groups – Watchers, Operators, and Administrators. In addition, password-protected access for all three user groups prevents third-party access.

##### easyE RTD Standard - EASY-RTD-DC-43-03B1-00

The display and operating elements of the easyE4 base device are mirrored on the Standard Remote Touch Display. Programming of the easyE RTD Standard is not required. The status of the easyE4 connected to the RTD is displayed and parameters can be adapted directly using the operating elements mirrored in the RTD.



## 10. Communication Connection to other devices easyE4

### 10.19 Convenient visualization for easyE4



Fig. 393: Mirroring of the easyE4 display on the easyE RTD Standard

For more information, please consult the "easy Remote Touchdisplay" manual, MN048027.

## 10. Communication Connection to other devices easyE4

### 10.19 Convenient visualization for easyE4

#### easyE RTD Advanced - EASY-RTD-DC-43-03B2-00

The Remote Touch Display Advanced can be used to get a custom visualization with easySoft 8, making it possible to use user-defined texts, images, and controls. In addition, data from multiple easyE4 devices can be visualized at the same time.

The visualization can be created with easySoft Editor in easySoft 8, and the \*.rtd visualization project file can then be transferred via Ethernet/easySoft or with a USB drive. easySoft 8 supports the use of images and other simple visualization elements, as well as remote access to the device menu of the connected easyE4. It is possible to edit timer function blocks using the easyE RTD Advanced.

In addition to a visualization project file on the easyE RTD Advanced, this visualization requires an easyE4 control relay Generation 08 or higher with firmware  $\geq$  V2.10 that supports the visualization.



Fig. 394: Application example for an easyE RTD Advanced



Only easyE4 control relays from generation 08 with firmware version  $\geq$  2.10 and easySoft version 8.10 support both easy Remote Touch Displays. Up until generation 07, only the easyE RTD Standard was supported.

For more information, please consult the "easy Remote Touchdisplay" manual, MN048027.

For information on how the easyE RTD Advanced handles, generates, and transfers \*.rtd files, please refer to easySoft 8.

## 10. Communication Connection to other devices easyE4

### 10.19 Convenient visualization for easyE4

#### 10.19.2 HMI Touchdisplays

The HMIs and the GALILEO visualization software can be used to show contents from connected easyE4 control relays on the color display panels and operate them remotely and externally.



Fig. 395: Visualization on HMI operating terminal

Data is exchanged between the devices using GALILEO's internal tag import format (\*.itf). easySoft 8 supports this export format for Modbus TCP..

Communication between EASY-E4-... and HMI touch displays happens via Modbus TCP.



For GALILEO users

It is advisable not to configure the system time cyclically.

Accordingly, Eaton recommends not setting bit 11.1 in the 1st word of the Control system tag permanently.

An overview of the available displays is listed under Accessories.

→ Section "Accessory devices", page 884

For more information on how to connect the controller, please refer to the tutorials and the various documents → Section "Additional information for use", page 893.

For additional product information, as well as access to the software demo version, please visit the product page.



[Eaton.com/easy](https://Eaton.com/easy)



[Eaton.com/galileo](https://Eaton.com/galileo)

## 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. 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 8.

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 8, the power flow display in the EASY-E4-...-12...C1(P) device enables you to test the logic operations of the circuit diagram.



The → Section "Problems on the SmartWire-DT line", page 781 section describes problems related to the SmartWire-DT line.

Only qualified persons should test electrical voltages while the easyE4 device is in operation.

## 11. Faults

### 11.1 Messages from the operating system

#### 11.1 Messages from the operating system

Messages on the LCD display	Explanation	Remedy
No display	Power supply interrupted	Restore power
	LCD is faulty	Replace easyE4
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.	Replace easyE4
ERROR: CLOCK	Clock error	Replace easyE4
microSD card access		
Waiting	The LCD screen cannot be used for a short bit. Potential causes include very high system loads and faulty hardware, e.g., the microSD slot.	If the problem continues, remove the slot permanently if it is not needed or contact your local Eaton Support contact.
Error (red background color)	The LCD screen cannot be used for more than one minute. Potential causes include very high system loads and faulty hardware, e.g., the microSD slot.	

## 11.2 Possible situations when creating programs


## 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 Circuit diagram on the memory card uses contacts/ relays that the easyE4 device does not recognize	Change easyE4 device type or the circuit diagram on the memory card
Power flow display does not show changes to the rungs	easyE4 device is in STOP mode	Select RUN mode
	Association/ connection not fulfilled	Check the circuit diagram and parameter sets and modify as required
	Relay does not activate coil	
	Incorrect parameter values/time	
	Analog value comparison is incorrect Time value of timing relay is incorrect 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	Replace easyE4 device
	easyE4 device relay is faulty	

## 11. Faults

### 11.3 Event

### 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.	The 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 RUN START is deactivated at easyE4.	Load, input circuit diagram 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  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: <ul style="list-style-type: none"><li>• Net is initialized and at least one station has not been detected. Check the plug-in connections.</li><li>• You have modified the NET ID or baud rate for at least one station after the configuration has been completed. Change the configuration.</li><li>• You have deleted the program on a NET station and thus also its NET configuration. Reconfigure the NET via station 1.</li><li>• You have expanded an existing NET station and replaced it with a new device which cannot be assigned parameters.</li></ul>
Flashing	NET operating fault-free



## 11. Faults

### 11.5 Issues related to the microSD memory card

#### 11.5 Issues related to the microSD memory card

If the microSD memory card cannot be accessed, a code will be shown on the easyE4 display.

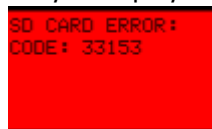


Fig. 396: Example of code display on display

##### Code microSD memory card

code	Message	Notes
33028	Invalid / wrong program length	
33032	Bad / wrong checksum	
33088	microSD not formatted or write error	
33152	Internal program and card program do not match	Depends on the easySoft 8 setting
33153	Program header general error	
33154	Program memory general error or wrong microSD format	
33155	Card is not (or no longer) present	Depends on the easySoft 8 setting
33156	Program IDs do not match	Depends on the easySoft 8 setting

The most likely cause of problems that do not depend on custom easySoft 8 settings:

- Problem with the contact with the microSD card holder  
=> insert the microSD card holder correctly
- Corrupted filesystem on the microSD card  
=> Reformat the microSD card
- Faulty microSD card  
=> Replace the microSD card



##### **CAUTION DATA LOSS**

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

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

Do not write to microSD cards constantly:

- microSD 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 microSD card only when the easyE4 is de-energized.

### 11.5 Issues related to the microSD memory card

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

Additional possible causes for codes 33028, 33032, 33153, and 33154:

- The project file on the microSD card has been changed manually outside of easySoft 8 (e.g., with a text editor).
- The microSD card was removed from the device while there was an ongoing write operation writing to the project file.

## **11. Faults**

### **11.5 Issues related to the microSD memory card**

## 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. Maintenance
12.3 Storage, transport and disposal

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. Do not expose the device to direct thermal radiation from heating appliances.

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:

Table with 2 columns: Ambient climatic conditions, and values. Rows include Air pressure (in operation), Temperature Operation, Storage / Transport, Humidity, and Condensation.

➔ 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 series 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. To learn more, please visit:



[Eaton.com/recycling](https://Eaton.com/recycling)

**Materials used in the packaging**

Packaging	Material
Outer packaging	Cardboard
Inner packaging	Cardboard Plastic bag: polyethylene (PE)

## **12. Maintenance**

### **12.3 Storage, transport and disposal**

## Appendix

---

<a href="#"><u>A.1 Dimensions</u></a>	<b>873</b>
<a href="#"><u>A.2 Approvals and declarations</u></a>	<b>878</b>
<a href="#"><u>A.3 easyE4 compatibility overview</u></a>	<b>881</b>
<a href="#"><u>A.4 Parts of an easyE4 project file (*.e80)</u></a>	<b>882</b>
<a href="#"><u>A.5 Technical data</u></a>	<b>883</b>
A.5.1 Data sheets	883
A.5.2 Overview of select characteristics	885
<a href="#"><u>A.6 Required memory for function blocks</u></a>	<b>889</b>
<a href="#"><u>A.7 Additional information for use</u></a>	<b>893</b>
A.7.1 Documents	893
A.7.2 Download Center, Eaton Online Catalog	894
A.7.3 Product information	894
A.7.4 Product training	894
A.7.5 Community	894
A.7.6 Cybersecurity	894
A.7.7 Internet links	895
<a href="#"><u>A.8 Sample Projects</u></a>	<b>896</b>



A.1 Dimensions

Base devices with 4 space unit front dimension

EASY-E4-UC-12RC1(P), EASY-E4-DC-12TC1(P), EASY-E4-AC-12RC1(P)

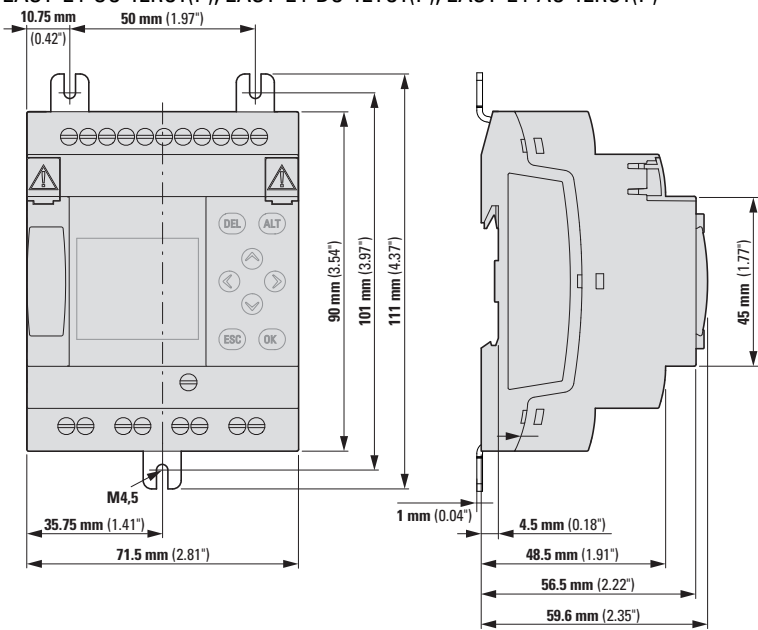


Fig. 397: Dimensions in mm (Inch) Base devices EASY-E4-...-12...C1(P)

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	Please refer to the device data sheet varies between 139 g and 230 g depending on the specific model

Base devices with 4 space unit front dimension

EASY-E4-UC-12RCX1(P), EASY-E4-DC-12TCX1(P), EASY-E4-DC-12TCX1(P)

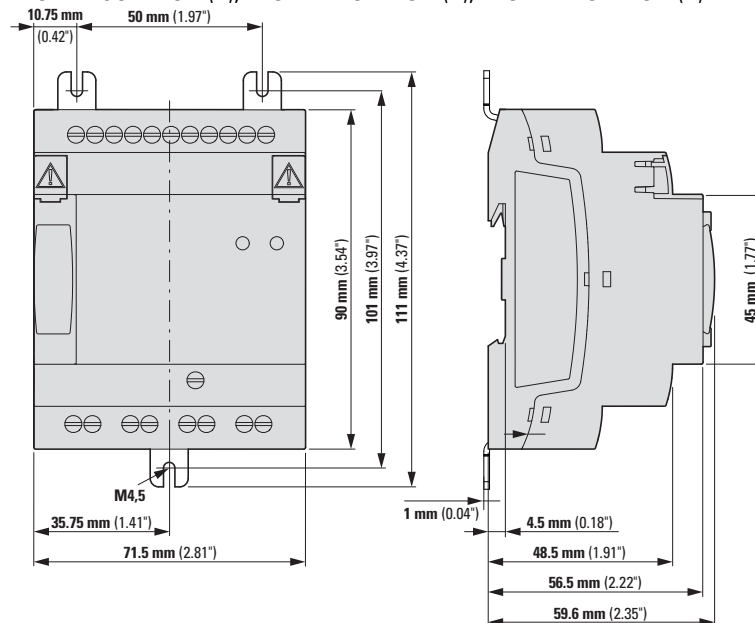


Fig. 398: Dimensions in mm (Inch) Base devices EASY-E4-...-12...CX1(P)

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	Please refer to the device data sheet varies between 139 g and 230 g depending on the specific model

Appendix  
A.1 Dimensions

Expansion devices with 4 space unit front dimension  
EASY-E4-UC-16RE1(P), EASY-E4-DC-16TE1(P), EASY-E4-AC-16RE1(P),

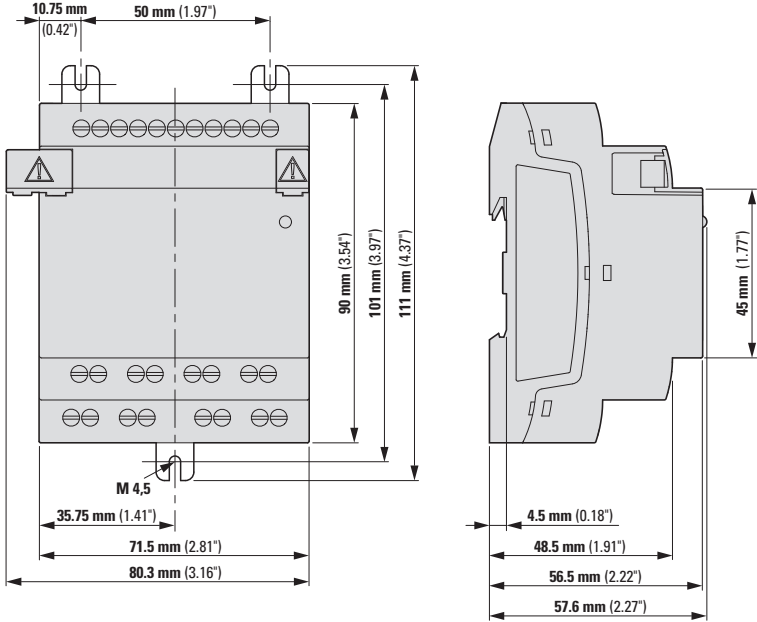


Fig. 399: Dimensions in mm (Inch) extensions 4SU

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	Please refer to the device data sheet 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)

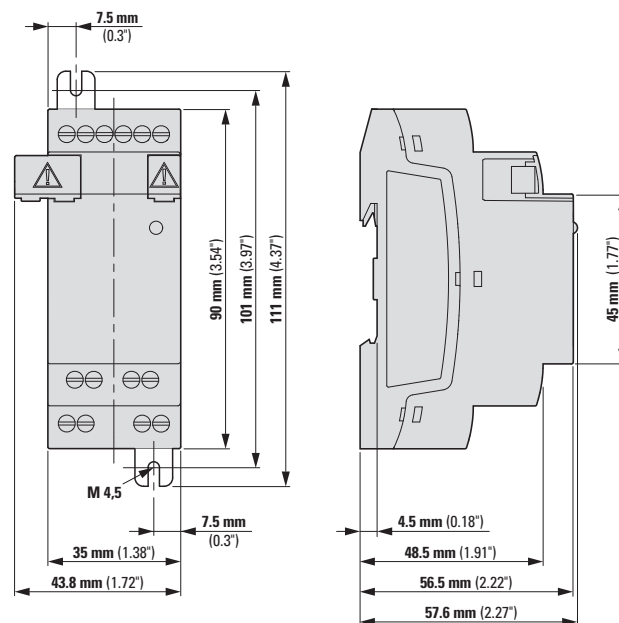


Fig. 400: Dimensions in mm (Inch) extensions 2SU

EASY-E4-DC-4PE1(P)

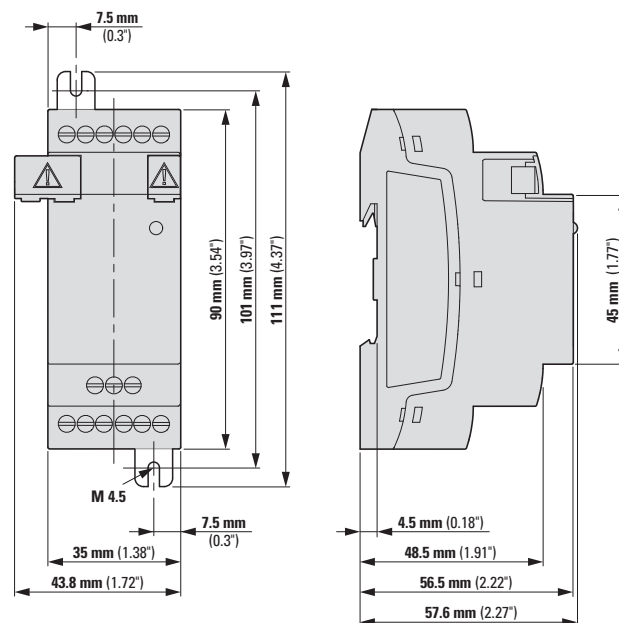


Fig. 401: Dimensions in mm (Inch)

Width x Height x Depth (without plug)	35 mm x 90 mm x 58 mm (1.38" x 3.54" x 2.28")
Weight	Please refer to the device data sheet varies between 79 g and 232 g depending on the specific model

Appendix  
A.1 Dimensions

Communication modules with front dimension of 2 space units  
EASY-COM-SWD-C1

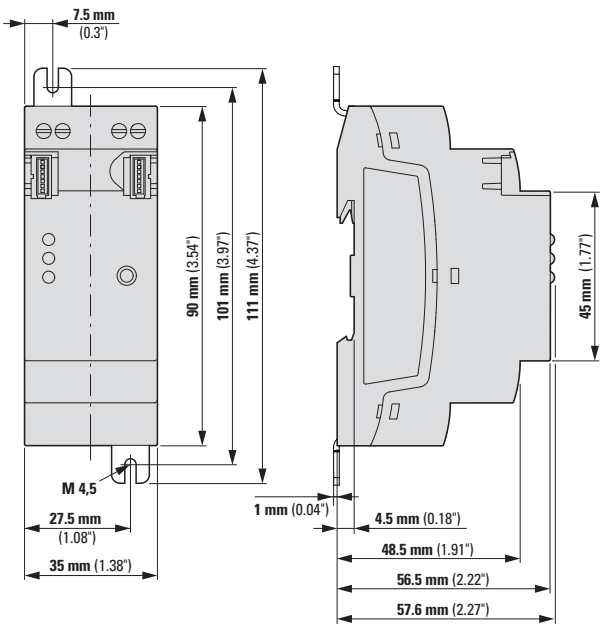


Fig. 402: Dimensions in mm (Inch) extensions 2SU

EASY-COM-RTU-M1

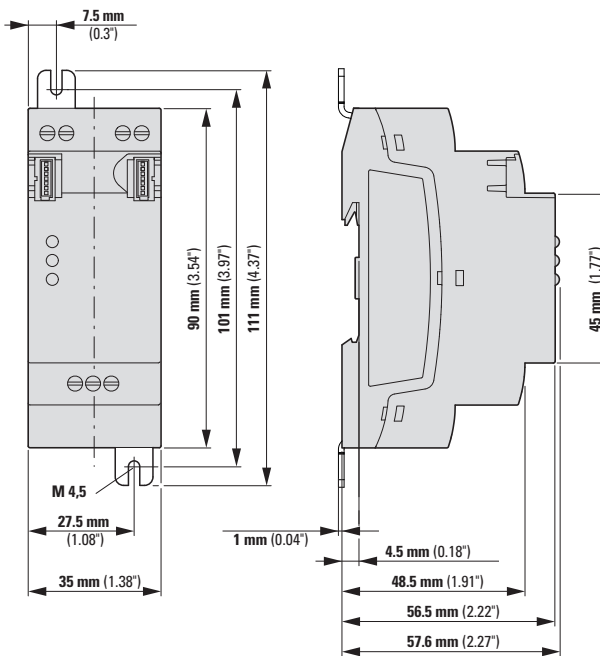


Fig. 403: Dimensions in mm (Inch) extensions 2SU

Width x Height x Depth (without plug)	35 mm x 90 mm x 58 mm (1.38" x 3.54" x 2.28")
Weight	87 gr EASY-COM-SWD-C1 82 gr EASY-COM-RTU-M1

## A.2 Approvals and declarations

The following specifications apply to all easyE4 devices.

Approvals and declarations		
cUL	UL File No. E205091, Volume 4 Application for easyE4 approval submitted	
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)	DNV-GL Certificate No. TAA00002HT Application for easyE4 approval submitted	
Applied standards and directives		
EMC (relevant for CE)		2014/30/EU
	IEC/EN 61000-6-2	Interference immunity for industrial environments
	IEC/EN 61000-6-3	Interference immunity for residential areas, commercial and light industrial areas as well as small businesses
from 07/2024	EN 61131-2	Programmable controllers - Part 2: Equipment requirements and tests
	IEC/EN 61000-6-3	Interference immunity for residential areas, commercial and light industrial areas as well as small businesses
Security		
	IEC/EN 61010-2-201	Safety requirements for electrical equipment for measurement, control and laboratory use - Part 2-201: Particular requirements for control equipment
Product standards		
	IEC/EN 61131-2	Programmable controllers: Equipment requirements and tests

## Appendix

### A.2 Approvals and declarations

#### Marine approvals:

Base devices	Starting with HW revision
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	Starting with HW revision
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

Communication modules	Starting with HW revision
EASY-COM-SWD-C1	01
EASY-COM-RTU-M1	01



Base and expansion devices as well as communication modules 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.

### UL-Approval

Notice of Authorization (NoA) for the  
easyE4: UL File No. E205091, Volume 4.

<b>Base devices</b>	<b>Starting with HW revision</b>
EASY-E4-UC-12RC1	02
EASY-E4-UC-12RC1P	03
EASY-E4-UC-12RCX1	02
EASY-E4-UC-12RCX1P	03
EASY-E4-DC-12TC1	02
EASY-E4-DC-12TC1P	03
EASY-E4-DC-12TCX1	02
EASY-E4-DC-12TCX1P	03
EASY-E4-AC-12RC1	03
EASY-E4-AC-12RC1P	03
EASY-E4-AC-12RCX1	03
EASY-E4-AC-12RCX1P	03

<b>I/O expansions</b>	<b>Starting with HW revision</b>
EASY-E4-UC-16RE1	03
EASY-E4-UC-16RE1P	03
EASY-E4-UC-8RE1	03
EASY-E4-UC-8RE1P	03
EASY-E4-DC-16TE1	03
EASY-E4-DC-16TE1P	03
EASY-E4-DC-8TE1	03
EASY-E4-DC-8TE1P	03
EASY-E4-AC-8RE1	02
EASY-E4-AC-8RE1P	02
EASY-E4-AC-16RE1	02
EASY-E4-AC-16RE1P	02
EASY-E4-DC-6AE1	03
EASY-E4-DC-6AE1P	03
EASY-E4-DC-4PE1	01
EASY-E4-DC-4PE1P	01

<b>Communication modules</b>	<b>Starting with HW revision</b>
EASY-COM-SWD-C1	01
EASY-COM-RTU-M1	01



## Appendix

### A.3 easyE4 compatibility overview

### A.3 easyE4 compatibility overview

The following table shows which generation of easyE4 base devices can be run with which firmware.

Base devices																					
	Main function extension	easyE4 base devices	FW	Functional Extensions	Cloud Service, Web Editor	RTD Advanced			CPU Update			Modbus-RTU, RTD Standard		SWID card							
			HW	V2.30	V2.25	V2.10	V2.02	V2.01	V2.00	V1.42	V1.41	V1.40	V1.31	V1.30	V1.23	V1.22	V1.21	V1.20	V1.12	V1.10	V1.01
12 2018	First Release OR-Code (HW, serial number, FW during production)		01						✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
05 2019	EMV 6 KV (UC, DC variants, AC from 01 6 KV)		02						✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
09 2019	UL approval (for all devices)		03						✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
02 2021	Update Display PCBA		04						✓	✓	✓	✓	✓	✓	✓	✓	✓				
04 2021	easyCOM Interface for communication modules		05						✓	✓	✓	✓	✓								
06 2021	New keypad		06						✓	✓	✓	✓	✓								
01 2023	New hardware		08	✓	✓	✓	✓	✓	✓												
04 2024	EPAS instead of QR code		08	✓	✓	✓	✓	✓	✓												
07 2024	Update PCBA		09	✓	✓	✓	✓	✓	✓												
Legend:																					
✓ updateable																					
HW Hardware Revision/Generation																					
FW Firmware Version																					

Fig. 404: Hardware (HW) / firmware (FW) compatibility

Programs created for an easyE4 device are forward compatible, i.e., \*.e80 projects can be run by easyE4 base devices with higher HW revisions.

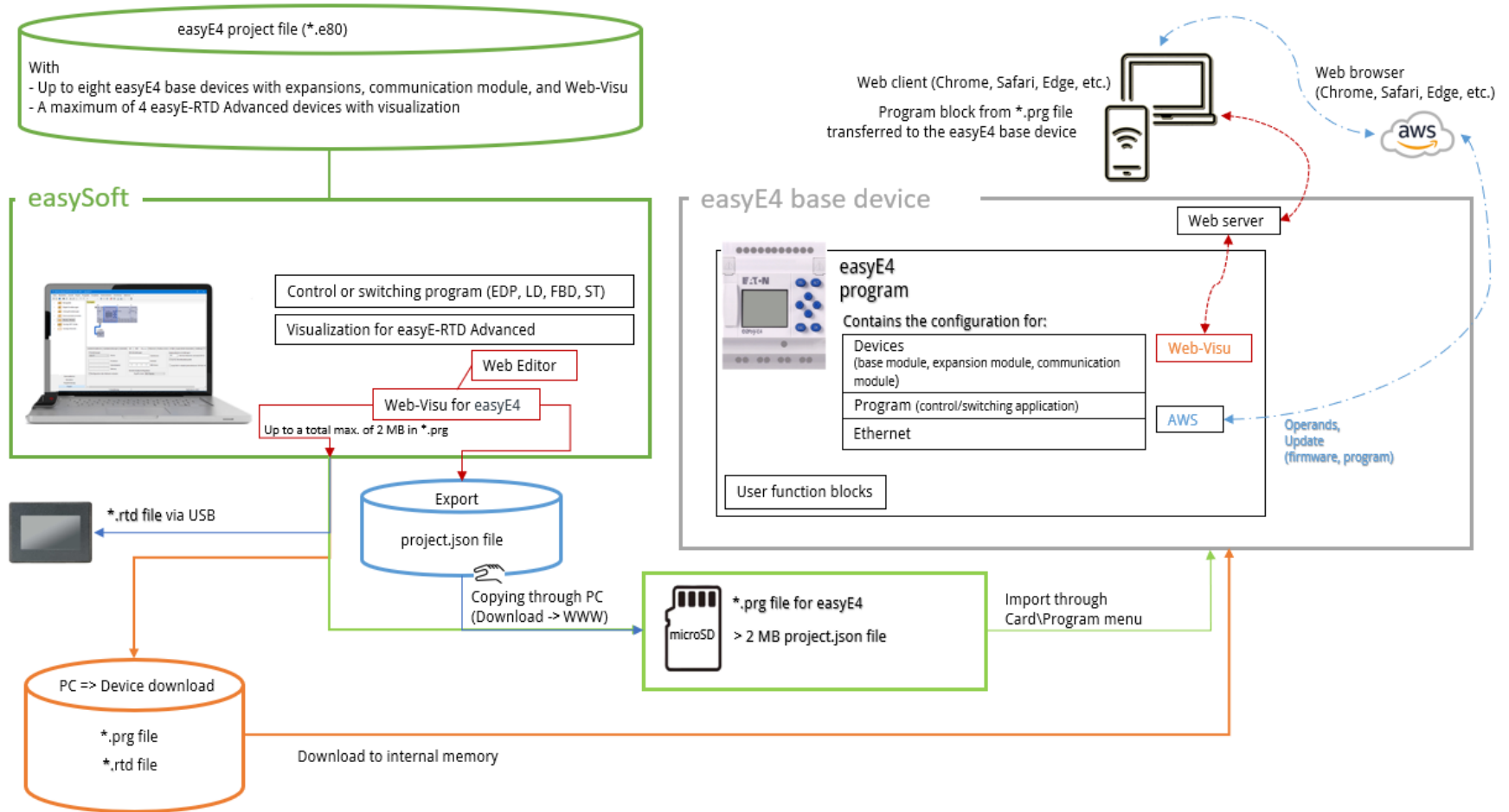
EASY-COM-... easy communication modules can be used with an easyE4 base device starting with generation 05.

For example:

- Communication module EASY-COM-RTU-M1 requires an easyE4 base device with FW V1.40 or higher
- Communication module EASY-COM-SWD-C1 requires an easyE4 base device with FW V1.30 or higher

The ACTUAL state of the easyE4 device will be shown in the Information\System device menu → Section "MenuInformation", page 165.

## A.4 Parts of an easyE4 project file (\*.e80)



\*.prg - program-relevant parts (circuit diagram, Ethernet configuration, device configuration, etc.)

## A.5 Technical data

### A.5.1 Data sheets

The current specifications for the device can be found in the data sheet for the device in the Eaton online catalog.

#### Base devices

with terminal type screw terminals

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 terminals

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	197509 EASY-E4-AC-12RCX1P

#### Expansions

with terminal type screw terminals

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 terminals

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

#### easy communication modules for easyE4 control relays

with terminal type screw terminals

SmartWire-DT	Modbus-RTU
199452 EASY-COM-SWD-C1	199453 EASY-COM-RTU-M1

**Accessory devices**

<b>Cat No. and type</b>	<b>Description</b>
<a href="#">198513 XV-102-A0-35TQRB-1E4</a>	Touch display for easyE4, 3.5 inches, 24 V <sub>DC</sub> , TFTcolor, QVGA 320 x 240 pixels, Ethernet
<a href="#">199734 XV-102-A3-57TVRB-1E4</a>	Touch display for easyE4, 5.7 inches, 24 V <sub>DC</sub> , TFTcolor, VGA 640 x 480 pixels, Ethernet
<a href="#">199740 EASY-RTD-DC-43-03B1-00</a>	easy Remote Touch Display, 4.3 inches, easyE RTD Standard 24 V <sub>DC</sub> , TFTcolor, 480x272 px, Res., Ethernet, RS485
<a href="#">EP-401057 EASY-RTD-DC-43-03B2-00</a>	easyE Remote Touch Display, easyE RTD Advanced 4.3 inches 24 V <sub>DC</sub> , FTcolor, 480x272 px, Res., Ethernet, RS485
<a href="#">191087 MEMORY-SUD-A1</a>	microSD 2 GB memory card with adapter, I Grade, without an operating system
<a href="#">197226 EASYSOFT-SWLIC</a>	Programming software license easySoft 8
<a href="#">061360 ZB4-101-GF1</a>	Device foot for screw mounting
<a href="#">197225 EASY-E4-CONNECT1</a>	Spare parts package for expansion modules, consisting of three (3) connectors and three (3) end covers
<a href="#">199513 EASY-E4-CONNECT-COM1</a>	Spare parts package for communication modules, consisting of three (3) connectors and three (3) end covers
<a href="#">229424 EASY200-POW</a>	Switched-mode power supply unit, 100-240 V <sub>AC</sub> / 24 V <sub>DC</sub> / 12 V <sub>DC</sub> , 0.35 A / 0.02 A, single-phase, controlled
<a href="#">212319 EASY400-POW</a>	Switched-mode power supply unit, 100-240 V <sub>AC</sub> / 24 V <sub>DC</sub> , 1.25 A, single-phase, controlled
<a href="#">272484 TR-G2/24</a>	Transformer, 230 V, 12/24 V, 2/1 A
<a href="#">199711 XN-332-5ETH-UMS</a>	Industrial standalone switch as slice module, 5 ports, 100 Mbit/s
<a href="#">EP-401058 EASY-E4-BOX-SKF-4TE</a>	Hinged inspection window for 4RU
<a href="#">EP-401059 EASY-E4-BOX-SKF-6TE</a>	Hinged inspection window for 6RU

## Appendix

### A.5 Technical data

#### A.5.2 Overview of select characteristics

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.

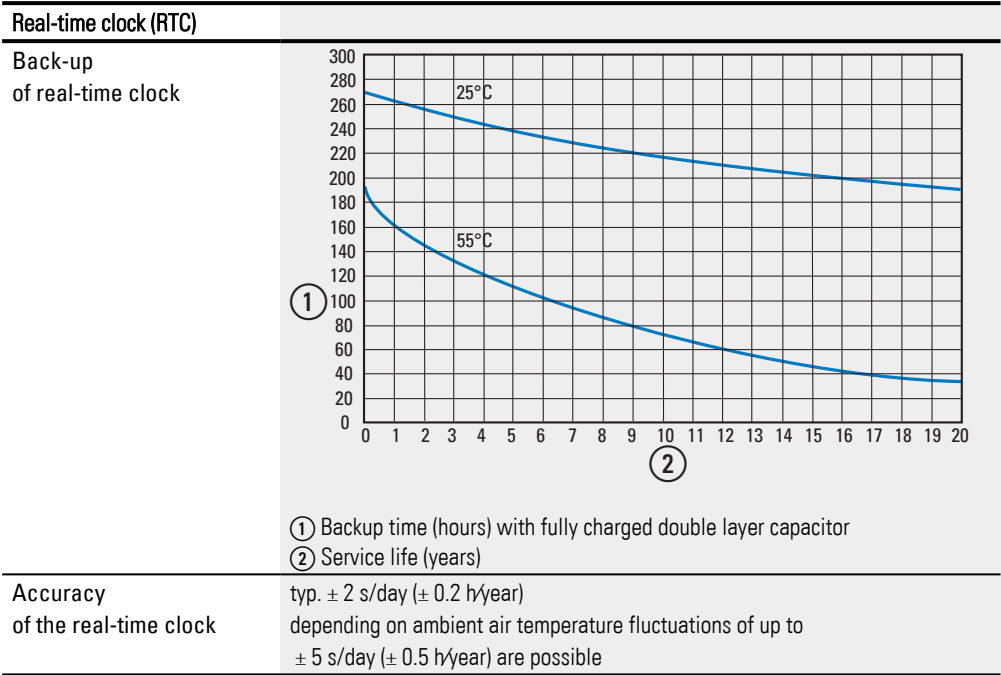
EASY-E4-	UC-12RC1(P)	UC-12RCX1(P)	DC-12TC1(P)	DC-12TCX1(P)	AC-12RC1(P)	AC-12RCX1(P)
Basic function	Control relay, expandable with I/O expansions from the easyE4 series; Ethernet port for connecting to a network Real-time clock					
Display with keypad	Monochrome 6 x 16 lines	-	Monochrome 6 x 16 lines	-	Monochrome 6 x 16 lines	-
Supply voltage	12/24 V <sub>DC</sub> or 24 V <sub>AC</sub>		24 V <sub>DC</sub>		100 - 240 V <sub>AC</sub> or 100 - 240 V <sub>DC</sub> (cULus 100 - 110 V <sub>DC</sub> )	
Input points	Digital: 8, 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) The display is readable between $\vartheta$ -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



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/pollution degree		Overvoltage category/pollution degree III/2
Electrostatic discharge (ESD)		according to IEC EN 61000-4-2
Air discharge		8 kV
Contact discharge	Generation	
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
	from 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 1.4 - 2 GHz: 3 V/m 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 Signal cables: 2 kV

## Appendix

### A.5 Technical data

power pulses (Surge)	according to IEC/EN 61000-4-5	1 kV (supply cables, symmetrical) 2 kV (supply cables, asymmetrical)
Radiated RFI	according to IEC/EN 61000-4-6	10 V



## Appendix

### A.6 Required memory for function blocks

## A.6 Required memory for function blocks

The memory required for unconnected function blocks is the same for all programming languages.

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. Some modules require additional system parameters that are created once when the 1st instance is used.

Tab. 146: Memory Required FB in Bytes

Function blocks	Instance 1	Instance 2	Note
A	68	68	
AC	68	68	
AL	540	38	+1 per character in subject and message
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	36	
DB	36	36	
DC	120	120	
DL	92	–	
FT	56	56	
GT	28	28	
HW	68	68	+4 per channel
HY	68	68	+4 per channel
IC	56 <sup>1)</sup>	56 <sup>1)</sup>	+12 min. per interrupt program
IE	52 <sup>1)</sup>	52 <sup>1)</sup>	+12 min. per interrupt program
IT	52 <sup>1)</sup>	52 <sup>1)</sup>	+12 min. per interrupt program
JC	20	20	
LB	16	16	
LS	64	64	
MC	84	84	
MM	48	48	
MR	20	20	
MU	64	64	
MX	96	96	
NC	32	32	

## Appendix

### A.6 Required memory for function blocks

Function blocks	Instance 1	Instance 2	Note
OT	64	64	
PM	72	56	+8 per support point
PO	96	96	
PW	48	48	
PT	40	40	
RC	76	–	
RE	128	112	+32 per record as soon as a flag is used in the recipe, this applies to every constant used in that recipe: +4 per constant;
SC	20	–	
SR(BIT)	96	96	
SR(DWORD)	96	96	
ST	24	–	
T	52	52	
TB	112	112	
TC	76	76	
VC	48	48	
WT	84	84	+4 per channel
YT	96	96	+4 per channel

1) As soon as an interrupt module is used, +12 bytes of memory are required once

#### 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 in LD/FBD, 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 bytes or MD marker double words.

Depending on the complexity of the pre-wiring, the actual consumption can also be higher. Each numerical constant used requires an additional 4 bytes in all programming methods.

In EDP, each rung occupies 20 bytes, regardless of its content, while an input / output circuit in the block diagram does not require any additional memory.

The following information was determined using the LD/FBD programming language.

Tab. 147: Memory Required FB CP

CP - Comparator	Connected to	Memory Required
Function block inputs/outputs	Operand	bytes
CP (not connected)		35
EN	I1	7
I1	IA1	7
I2	IA2	7
LT	Q1	7
EQ	Q2	7

## Appendix

### A.6 Required memory for function blocks

<b>CP - Comparator</b>	<b>Connected to</b>	<b>Memory Required</b>
<b>Function block inputs/outputs</b>	<b>Operand</b>	<b>bytes</b>
GT	Q3	7
TOTAL		77

Tab. 148: Memory Required FB T

<b>T – Timing relay</b>	<b>Connected to</b>	<b>Memory Required</b>
<b>Function block inputs/outputs</b>	<b>Operand</b>	<b>bytes</b>
T (not connected)		55
EN	I1	7
RE	I2	7
ST	I3	7
I1	5 ms	11
I2	–	0
Q1	Q1	7
QV	QA1	7
TOTAL		101

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 several display or input elements hardly require any additional memory due to compression technology.

## Appendix

### A.6 Required memory for function blocks

Tab. 149: Memory required by D text display function block in bytes

<b>D - Text display</b>	<b>Memory Required</b>
<b>Function block inputs/outputs</b>	<b>bytes</b>
<b>Display elements</b>	
Value display, without scaling	12
Value display, with scaling	32
Bargraph	24
Static text (without text)	12 + 2 per character <sup>1)</sup>
Running text	12 + 2 per character <sup>1)</sup>
Rolling text without association	16 + 2 per character <sup>1)</sup>
Rolling text with association	28 + 4 per value + 2 per character <sup>1)</sup>
Message text, Bit link	16 + 2 per character <sup>1)</sup>
Message text, Value link	28 + 4 per value + 2 per character <sup>1)</sup>
Date and time display	12
DZ Weekday	8
Timing relay value display	12
<b>Entry elements</b>	
Value entry	12
Latching pushbutton	12
Message text selection (without text) + per text with 16 characters	28 40 <sup>1)</sup>
Date and time entry	8
Timing relay value entry	8
<sup>1)</sup> Potentially less memory required if optimization is possible	

## Appendix







### A.7 Additional information for use

## A.7 Additional information for use

### A.7.1 Documents

For more information on additional devices and modules, please refer to the following documentation:

#### A.7.1.1 Instruction leaflets

	Installation instructions Base devices	IL050020ZU
	Installation leaflet for I/O expansions	IL050021ZU
	Installation instructions Fixing bracket	IL05009005Z
	Installation instructions EASY-E4-SIM	IL050022ZU
	Installation instructions EASY-COM-SWD...	IL050024ZU
	Installation instructions EASY-COM-RTU...	IL050035ZU

#### A.7.1.2 Manuals






	easy Remote Touch display manual easyE RTD	MN048027EN
---	---	------------

#### A.7.1.3 Documents for SmartWire-DT communication system

System description, engineering, installation, commissioning, and diagnostics for a SWD line

	SmartWire-DT The System Manual	MN05006002Z
---	--------------------------------	-------------

Setup, engineering, installation, etc. for the individual SWD modules

	Manual for SmartWire-DT IP20 modules	MN05006001Z
	Manual for SmartWire-DT IP6x modules	MN120006
	Manual EMS2... Electronic motor starter with SWD	MN120008
	PowerXL™ DX-NET-SWD manual	MN04012009Z
	Installation instructions SWD4-...	IL04716001Z

### **A.7.2 Download Center, Eaton Online Catalog**

Enter "easy" or "SWD" into the search box on the Eaton website and the catalog will take you directly to the corresponding product group in the Automation, Control and visualization section.

There are various publications available for download under Documentation.

 [Eaton.com/documentation](https://eaton.com/documentation)

### **A.7.3 Product information**

For up-to-date information, please consult the product page on the Internet.

 [Eaton.com/easy](https://eaton.com/easy)

#### **Tutorials**

For helpful videos that explain how to use specific functions, please visit the product page at [Eaton.com/easy-tutorial](https://eaton.com/easy-tutorial).

### **A.7.4 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:

 [Eaton.com/training](https://eaton.com/training)

### **A.7.5 Community**

easyForum is an additional source of help that can be found on the Internet at:

 [Easy-forum.net](https://easy-forum.net)

### **A.7.6 Cybersecurity**

Eaton recommends implementing measures for protecting against cyberattacks.

 Eaton cyber security

 [Eaton.com/cybersecurity](https://eaton.com/cybersecurity)



Product Cybersecurity, Secure Hardening Guideline

MZ049001EN

## Appendix

### A.7 Additional information for use

#### A.7.7 Internet links



[anybus.com/.../industrial-ethernet/modbus-tcp](http://anybus.com/.../industrial-ethernet/modbus-tcp)

## A.8 Sample Projects

To obtain a quick impression of what is possible with the easyE4 series of devices, visit the Product page on the Internet. This provides examples of applications as well as 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

[Eaton.com/software/Anwendungsbeispiele/easy/Deutsch](http://Eaton.com/software/Anwendungsbeispiele/easy/Deutsch)

[Eaton.com/software/Application Samples/easy/English](http://Eaton.com/software/Application Samples/easy/English)

These examples come with a task description, the circuit diagram, and the easySoft 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 [Eaton.com/easy-tutorial](http://Eaton.com/easy-tutorial).

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 8:



The application examples created by Eaton can only be transferred to the easyE4 device if a license has been added for easySoft 8.

### easyE4\_Lauflicht\_EDP.e80 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 and 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 Light 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
- H1 Chaser light on times

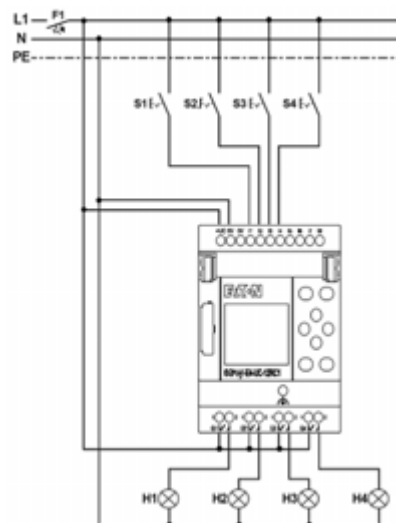


Fig. 405: Circuit diagram for easyE4 chaser light

## Alphabetical index

### 2

24 Volt impulses ..... 387

### A

A - Analog value comparison ..... 330

A - Analog value comparison visualization  
device ..... 331

AC - Astronomic clock ..... 296

AC device notes ..... 48

Acceleration sequence

PO - pulse output ..... 389

Access control ..... 730

Accessory devices ..... 34

Activation P buttons ..... 485

Actuators ..... 368

Acyclical Modbus client request ..... 531

Acyclical Modbus RTU request ..... 547

### ADD

AR - Arithmetic ..... 338

Adding ..... 338

Rung ..... 202

Aeration and de-aeration ..... 55

After Sales Service ..... 2

AL - Alarm function block ..... 472

Alarm ..... 484

Alarm function block ..... 472

Alarm function blocks ..... 734

Ambient climatic conditions ..... 55, 869, 885

Analog Signals ..... 51

Analog value comparison ..... 330

### AND

BV - Boolean operation ..... 478

API key ..... 753

Approvals ..... 878

AR - Arithmetic ..... 336

Arithmetic ..... 336

Assigning IP addresses ..... 117

Assigning variables, function block input ..... 221

Associated variable ..... 495

Astronomic clock ..... 296

Auxiliary relay ..... 234

AV - Average ..... 342

Available operands

User function blocks ..... 620

Average ..... 342

AWS - access ..... 796

AWS Account ..... 802

AWS Cloud ..... 792

AWS diagnostic buffer ..... 828

AWS diagnostics ..... 828

AWS IoT Core ..... 808

### B

Background ..... 673

Backlight ..... 228, 673

Backward jumps ..... 208

Bargraph ..... 496

BC- Block comparison ..... 411

### BCD

Example ..... 567

NC - Numerical converter ..... 564

BCD-coded decimal value ..... 563

### BIN

Example ..... 566

NC - Numerical converter ..... 564

BIN binary-coded numbers ..... 563

BIP		Elements .....	189
Operating mode .....	377	Saving .....	203
Block comparison .....	411	Circuit diagram creation, Troubleshooting .....	862
Block transfer .....	419	Cleaning .....	868
Blue .....	740	Clock .....	254
BOOL definition .....	226	Weekly timer .....	244
Boolean operation .....	477	Code Memory card .....	865
BOOT.TXT .....	128, 132	Coil	
Brake sequence .....	391	Connect .....	201
Brand names		Definition .....	191
Product names .....	2	Deleting .....	200
BT - Block transfer .....	419	Entering, modifying .....	199
Bus delay .....	639, 727	Field .....	187
BV - Boolean operation .....	477	Function, overview .....	191
<b>C</b>		Negation .....	194
C - Counter Relay .....	305	Searching .....	204
Cable protection .....	68	Colors in Communication view .....	698
Canceling, circuit diagram input .....	204	ComBUS .....	690
Card manager .....	519, 521	Comment	
CARD START .....	646	User function block .....	606
Carry .....	307, 319, 326	Commissioning .....	105
Certificate .....	709	SmartWire-DT .....	777
Installation file .....	713	Communication	
Name .....	713	View .....	690
Certificate errors .....	710	Community .....	894
CF - Frequency counter .....	311	Company information .....	2
CH - High-speed counter .....	317	Comparison of analog and setpoint values .....	330
Changing input values, on function blocks .....	223	Comparison of variables and constants .....	350
Channel number		Compatibility .....	881
MX - data multiplexer .....	437	Compatibility rules .....	698
Characteristic curve .....	362	Conditional jump .....	524
Characteristic map .....	362	Config LED .....	778, 789
CI - Incremental counters .....	323	Connection .....	66
Circuit diagram .....	187	Deleting .....	202
Checking .....	206	Representation in circuit diagram display ...	188
		To the device .....	700

Connection rules for operands .....	228	CP - Comparator .....	350
Connections		CP - Comparators for visualization devices .....	351
External .....	87	Create log files .....	521
Constants		Creating a job .....	820
Assigning, function block input .....	221	Creating an S3 bucket .....	824-825
Contact		Creating, editing an operand list .....	756
Changing, N/O – N/C .....	198	Cybersecurity .....	894
Connect .....	201	Cycle pulse	
Cursor buttons .....	205	Falling edge .....	195
Definition .....	190	Rising edge .....	194
Deleting .....	200	Cycle time .....	368, 383, 569, 649
Entering, modifying .....	197	<b>D</b>	
Fields .....	187	D - Text display	
Name .....	197	Bar graph .....	496
Number .....	197	Date and time display .....	503
Searching .....	204	Date and time entry .....	508
Contact function .....	192	Display and input elements .....	494
Continuous mode		Latching button .....	507
AV - Average .....	342	Message text .....	500
Continuous operation .....	347	Message text selection .....	508
Copy		Rolling text .....	498
Marker contents .....	421	Running text .....	497
Copy-protected .....	2	Static text .....	496
Copyright .....	2	Timing relay value display .....	504
CORS .....	729	Timing relay value entry .....	508
Count direction .....	306, 318	Value display .....	494
Counter .....	317	Value entry .....	505
C - Counter Relay .....	305	D - Text display (Display) .....	481
CF - Frequency counter .....	311	D - Text display editor .....	491
CI - incremental counter .....	305	Damage .....	57
CI - Incremental counters .....	323	Data block .....	411, 419
OT - Operating hours counter .....	264	Data block comparator .....	411
Counter input .....	317, 323	Data block transfer .....	419
C - Counter relay .....	305	Data function block .....	425
CF - Frequency counter .....	311	Data logger .....	510
Counter relay .....	305		

Data multiplexer .....	437	Device	
Data types .....	226	Changing language .....	108
Date and time display .....	503	Device certificate .....	709
Date and time entry .....	508	Device display .....	673
Date setting .....	659	Device family .....	136
DB - Data function block .....	425	Device ID .....	636
DC - PID controller .....	375	Device reset .....	135
Deactivate automatic scrolling to input fields ...	756	Device versions .....	27, 29
Debounce .....	648	Devices shown in color .....	698
Debounce activated .....	677	Diagnostic messages .....	686
Debounce deactivated .....	678	Dimensions .....	873
Deceleration ramp .....	390	Directives .....	878
Decimal numbers .....	563	Display	
Declarations .....	878	Default colors .....	485
Default colors .....	485	Elements .....	481
Default colors display .....	485	Display and input elements .....	494
Default IP address .....	700	Display elements .....	481
Default NET ID .....	700	Display messages .....	861
Defining password-protected areas .....	654	Display priority .....	484
Delay time .....	677	Disposal	
Delay time AC voltage .....	679	Recycling .....	870
Delay value .....	382	DIV	
DELETE		AR - Arithmetic .....	338
Function blocks .....	224	Divide .....	338
Deleting		DL - Data logger .....	510
Operands at function block inputs/outputs ...	222	DNS name .....	759
Rung .....	202	DNS name (Modbus TCP communication) .....	835
Demo version .....	96	Download .....	487
Derivative term .....	376	easyE4 .....	122
Description .....	23	Visualization devices .....	122
Destination address .....	419	Download Center - Documentation .....	894
Destination range .....	412, 420	DST .....	660
Detecting or changing bit patterns .....	477	DWORD definition .....	226
Determining counter frequency		<b>E</b>	
CF - Frequency counter .....	311	E-mail .....	730, 758

E1 control output .....	370	Ethernet network .....	113
e4settings.ini .....	148	Evaluating a falling edge .....	195
EASY-COM-RTU-... commissioning .....	788	Evaluating a rising edge .....	194
easy certificate .....	709	Example DL as ring buffer .....	517
easy Root CA .....	712-713	Execution time for an interrupt .....	572, 583, 589
easy root certificate .....	97	Exp firmware update .....	141, 143
easyConnect .....	681, 690, 741		
easyE4		<b>F</b>	
Download .....	122	Factory communication parameters .....	700
easyE4 certificate .....	709	Factory IP address setting .....	700
easyE4 device certificate .....	709	Factory NET ID setting .....	700
easyNET - NET - Compatibility .....	722	Faults .....	860
easyProtocol .....	691	Faults on the SWD line .....	791
easyProtocol on devices in their state of deliv- ery .....	710	Features .....	24
easyProtocol V2 .....	136	FF - FlipFlop .....	434
easyRootCert .....	713	Files in the S3 bucket .....	824
easySoft		Firmware version .....	138, 600, 603
Installing multiple versions .....	37	Firmware version 1.12 .....	830
Eaton easyE4 root certificate .....	709	Fixed IP address .....	117
ecat .....	894	FlipFlop .....	434
ED - EdgeDetector .....	430	Frequency change	
EdgeDetector .....	430	PO - Pulse output .....	390
Ejecting the memory card .....	146	FT - PT1-Signal smoothing filter .....	382
Electrical isolation .....	63	Full version .....	96
Electromagnetic compatibility .....	887	Function .....	24
EQ .....	332, 351, 411	Function block editor .....	220
Equal .....	330, 351	Function block list .....	219
Equipment supplied .....	56	Function block outputs .....	538, 554
Error		Function block value ranges .....	241
Rectification, on event .....	863	Function blocks	
Establishing an Ethernet connection .....	117	Adding to a circuit diagram for the first time .....	217
Ethernet .....	90, 640	Assign operands, output .....	222
Configuration .....	705	Assigning operands, input .....	221
Physical connection .....	117	Check .....	224
Ethernet interface .....	690	Configuration editor .....	220
		Definition .....	189

Delete .....	224
List .....	219
Function code .....	534, 551

## G

Gain factor .....	331
General .....	317, 323
Generation .....	136, 887
GET .....	460
Getting a license key .....	94
GO TO other rungs .....	203
Gray .....	740
Greater than .....	330, 336, 351
Green user function block .....	608
GT .....	331, 351
GT - "GET" Network .....	460
GT - Get values from NET .....	460
Guest logon .....	739

## H

Hardware counter .....	317
Hardware inputs .....	317
Hardware output .....	368
Hazards	
Device-specific .....	42
High-speed counter .....	317
High-speed counter functions .....	311
High-Speed Counters .....	323
HW - Weekly timer .....	244
HY - Year time switch .....	254
Hysteresis .....	330

## I

I/O point .....	842
IAM role .....	797
IC - Counter-controlled interrupt .....	572

IE - Edge-controlled interrupt .....	583
Impulse relay .....	192
Incremental Encoder .....	323
Information for use .....	893
Initial commissioning .....	105
Initialization mode .....	422
Input cables	
Length .....	46
Input delay .....	648
Installation .....	53
Installation position	
SD card .....	54
Selection .....	54
Installing multiple easySoft versions .....	37
Integer value .....	563
Integral term .....	376
Intended use .....	23
Interface	
User function block .....	603
Interfaces	
Ethernet .....	90
Interrupt function block .....	583
Counter-controlled .....	572
Time-controlled .....	589
Interrupt load .....	577, 594
Inversion	
Contactor function .....	194
Inverting	
Contact .....	198
IOX .....	741
IP address, fixed .....	117
IP addresses .....	117
IT - Interrupt function block .....	589

## J

JC - Conditional jump .....	524
-----------------------------	-----

Jog mode .....	389	LED ETHERNET .....	107, 688
JSON		LED Modbus RTU .....	789
API .....	737, 743	LED POW/RUN .....	107, 688-689, 778, 789
Jump		Less than .....	330, 351
JC - Conditional jump .....	524	LI - lengthy calculations .....	378
Jump label .....	529	Lifespan	
Jump label .....	529	Backlight .....	106
Jumps .....	207	LIFO .....	455
<b>K</b>		Lighting .....	673
K		Limits of PW pulse width modulation function block .....	370
MX - data multiplexer .....	437	List	
Key to part numbers .....	887	Web client .....	742
Know-how protection		Load torque	
User function block .....	604	PO - Pulse output .....	389
KP .....	382	Loading circuit .....	743
<b>L</b>		Loading programs onto multiple NET stations ...	725
Label .....	36	Location of use .....	54
Language .....	487	Login guest .....	739
Language changeover .....	486	Lower and upper limits .....	407
Languages .....	486	LS - Value scaling .....	354
Changing on the device .....	108	LT .....	351
Latching button .....	507	Analog value comparator .....	330
Latching relay .....	193	<b>M</b>	
LB - Jump label .....	529	Maintenance .....	868
LE .....	228, 673	Manipulated variable .....	376
LE operand .....	673	Manipulated variable SV .....	368-369
LE01 .....	673	Mark-to-space ratio .....	317, 324
LE02 .....	673	Marker .....	434, 543
LE03 .....	673	Definition .....	234
LE04 .....	673	Initialize MB, MW + MD .....	422
LE05 .....	673	Marker range accessible with offset .....	419
LE06 .....	673	Retention .....	239
LED		Value range .....	229
Checking of the NET .....	864	Marker range .....	236, 573, 584, 590



Marker range assignment .....	234	MU - Acyclical Modbus RTU request .....	547
Marker range mapping .....	235	MUL	
Marker table .....	236	AR - Arithmetic .....	338
Markers		Multiply .....	338
Assigning, function block input .....	221	MX - Data multiplexer .....	437
Copy MB, MW + MD .....	421		
Master Reset .....	543	<b>N</b>	
MC - Acyclical Modbus TCP request .....	531	N/C contact .....	190
Memory card .....	146	Inverting .....	198
Memory display, circuit diagram .....	188	N/O contact .....	190
Message		Inverting .....	198
PROG INVALID .....	862	Name	
Message text .....	500	User function block .....	600
Message text selection .....	508	Nameplate .....	36
microSD .....	146	Naming convention for DNS name .....	835
Min-/Max function .....	359	NC - Numerical converter .....	563
Minimum on duration .....	368	Negation, coil .....	194
Minimum on duration = Minimum off duration ..	370	NET .....	638, 721
Minimum period duration .....	370	-ID .....	210
Missing parts .....	57	Configuration .....	705
MM- Min-/Max function .....	359	Operands .....	209
Modbus RTU .....	558, 782	NET-GROUP .....	638, 726
Broadcast .....	552	NET-ID .....	638, 726
Modbus RTU broadcast .....	552	NET - Definition .....	721
Modbus RTU map .....	558	NET settings .....	725
Modbus RTU slave .....	558	NET station heartbeat .....	724
Modbus TCP .....	830, 844	Network function blocks .....	464
Modbus TCP client .....	832	Network Function Blocks .....	460
Modbus TCP server .....	844	Network operation .....	113
Modifying		Network portion .....	117
Connections .....	201	NO .....	411, 419
Contacts and coils .....	196	Normal mode .....	395
Motion control .....	387	Normalized variables of the PID controller .....	375
Mounting .....	58	NOT	
MQTT test client .....	815	BV - Boolean operation .....	478
MR - Master reset .....	543		

Number of pulses		OR	
PO - Pulse output .....	389	BV - Boolean operation .....	478
Numeric formats .....	229	Organizing marker ranges .....	234
Numerical converter .....	563	Original Operating Instructions .....	2
Example using EDP .....	568	OT - Operating hours counter .....	264
Numerical converter mode .....	563	Output line	
<b>0</b>		Length .....	47
Offset .....	330, 419	Overview of operands .....	229
On the CH High-Speed Counter .....	317	<b>P</b>	
On the CI Incremental Encoder .....	323	P buttons .....	205, 481
One-time mode		Package contents .....	56
AV - Average .....	342	Parameter setting .....	484
ONLINE .....	698	Parameters	
Online Catalog .....	894	Enabling/disabling access .....	218
Operand		Part number .....	33
LE .....	673	Password	
Operand connection rules .....	228	Assigning .....	655
Operand overview .....	227	Changing .....	656
Operand table .....	236	Enabling .....	656
Operands .....	226	Forgotten .....	657
Assigning .....	221	Performance map .....	362
Assigning, function block output .....	222	Period duration .....	368
Deleting at function block inputs/outputs ....	222	Period duration to minimum ON duration ratio ..	370
Operating frequency .....	388-389	PID controller .....	375
Operating hours counter .....	264	Operating mode .....	375
OT - Operating hours counter .....	264	PID Controller	
Operating mode .....	157, 377, 382, 569	Scan time .....	375
AV continuous mode .....	345	Plausibility check .....	622
AV one-time mode .....	345	PM - Performance map .....	362
Timing relay .....	275	<b>PO</b>	
Operating sequence		Jog mode .....	397
Pulse output .....	389	Normal mode .....	395
Operating states .....	672	PO - Pulse output .....	387
Operation		Pulse output .....	387
Proper .....	41	Ports .....	87

POW/AUX	
SmartWire-DT power supply .....	773
Power supply	
POW/AUX .....	773
Problems on the SmartWire-DT line .....	781
Process image .....	665
Product information .....	894
Product training .....	894
Program cycle time .....	378
Program name .....	650
Programming method	
User function block .....	600
Proportional actuators .....	368
Proportional Factor .....	382
Proportional gain .....	382
Proportional gain Kp .....	377
Proportional term .....	376
PRSNT .....	685
PT - Put values to NET .....	464
PT1 signal smoothing filter .....	382
Pulse duration .....	368
Pulse output .....	369
PO - Pulse output .....	387
Pulse sequence .....	368
Pulse Shape of Counter Signals .....	323
Pulse width .....	368
Pulse width modulation .....	368
Puny code .....	835
Punycode .....	835
PUT .....	464
PW - Pulse width modulation .....	368

## Q

Q01/Q02 .....	368
Q1 (boolean function block output)	
PT - "PUT" Network .....	464

SC - Synchronizing time via network .....	468
Q1 (Boolean function block output) .....	330, 425, 460
OT - Operating hours counter .....	264
Quickly entering values with the keyboard .....	230

## R

Rate time Tv .....	377
RC - Real-time clock .....	269
RE - Recipe records .....	441
Reading values from the network .....	460
Real-time clock .....	269
Synchronizing via network .....	468
Rebuilding a project .....	123
Recipe .....	441
Recovery time .....	382
Reference data block .....	411
Reference value table .....	362
Relay	
Coil function .....	191
Relays	
Definition .....	189
Remote display keys .....	731
Remote operation .....	114
Remote RUN .....	639, 726
Removing password protection .....	656
Repairs .....	868
Required function block memory .....	889
Required memory for function blocks .....	889
Reset .....	135, 377, 382, 563
VC - value limitation .....	407
Reset time Tn .....	377
Reset, coil function .....	193
Resolution .....	368
Retention .....	425, 605, 651
Retention with Function & Relays .....	323
Retention with function relays & blocks .....	311

Retention with Function Relays & Blocks .....	305	Setting a boot program .....	128, 132
Retentive markers .....	239	Setting a starting program .....	128, 132
Rolling text .....	498	Setting the web server login text .....	733
Rolling time .....	484	Setting up a web server .....	728
Root certificate .....	709	Setting up users .....	732
Root certificates .....	709	Settings web client .....	751
RTU .....	782	SH .....	407
RUN .....	157	Shift register .....	447
RUN START .....	646	Shifting bits forward/backward .....	447
Rung		Shifting double words forward/backward .....	447
Adding/Deleting .....	202	Showing easyE4 operands in the web client ....	747
Change .....	203	Signal smoothing .....	382
Deleting .....	202	SL .....	407
Rungs .....	188	SmartWire-DT .....	770
Running text .....	497	SmartWire-DT coordinator .....	780
<b>S</b>		Source address .....	419
Safety .....	39	Source range .....	413, 420
SaveAllFBChanges .....	749	Splash screen .....	147
Saving, Circuit diagram .....	203	Splash screen display duration .....	637
SC - Synchronizing clock via NET .....	468	SR - Shift register .....	447
Scaling		ST - Set cycle time .....	569
Value .....	354	Standards .....	878
Scan time .....	378	Start frequency .....	389
SD card .....	88	Start ramp .....	390
Search for devices .....	119	Starting program .....	146
Searching, contacts and coils .....	204	Starting the web client .....	736
Secure communication with certificates .....	709	Startup mode .....	645
Sender .....	760	State of delivery .....	710
Sending e-mail .....	472	Static text .....	496
Separate web client operands .....	747	Status display .....	111
Series production .....	137	Step response .....	382
Service .....	36	Stepper motor .....	387
Set cycle time .....	569	STOP .....	157
Set, coil function .....	193	Storage .....	869
Setpoint .....	375	Storage location	
		UF .....	617

User function block .....	617	Temperature measuring .....	80
SUB		Terminal capacity .....	66
AR - Arithmetic .....	338	Terminal layout .....	774, 785
Subtracting .....	338	Testing, circuits via the P buttons .....	205
Support .....	36	Text display .....	481, 484
SWD .....	770	Text display editor .....	491
SWD LED .....	779	Static text .....	496
Switching contact → please refer to Contact ...	190	TG .....	382
Switching duration		Three step controller .....	402
T - Timing relay logic relay .....	272	Threshold value switch .....	330
Synchronize date via NET .....	468	Time .....	254, 468
Synchronizing NET stations .....	468	SC - Synchronizing time via network .....	468
Synchronizing the device clock at runtime .....	855	Time reference value	
Synchronizing the Modbus TCP device clock ...	855	T - Time relay/logic relay .....	274
System parameters .....	148	Time response .....	676
System requirements .....	38	Time response;Base devices .....	677
System settings .....	728	Time setting .....	659
System update .....	754	Time switch .....	244, 284, 292
T		Time value .....	275
T-Timing relay .....	272	T-Timing relay logic relay .....	275
Flashing .....	272	Timer .....	244
On-delayed .....	272	HY - Year time switch .....	254
Retention .....	282	Timer constant .....	230
Single pulse switching .....	275	Timing characteristics;Expansion devices .....	681
Stop .....	272	Timing relay .....	272
Switch off time .....	272	Operating mode .....	275
Timing and counter relay example .....	631	Timing relay value display .....	504
Trigger input .....	272	Timing relay value entry .....	508
T – Timing relay logic relay .....	275	TN	
Tab		Control system .....	375
Webserver .....	728	Transit damage .....	56
Table function .....	455	Transport .....	869
TB - Table function .....	455	Trigger input (trigger coil)	
TC - Three step controller .....	402	"PT - PUT" Network .....	464
Technical data .....	883	Troubleshooting .....	860
		During circuit diagram creation .....	862

Two's complement .....	477	User function block yellow .....	611
<b>U</b>		User function blocks	
UF		Comparing .....	629
Archive .....	617	Using the web client .....	740
UF - User function block .....	597	<b>V</b>	
UNP .....	377	Value display .....	494
Up/down counters .....	305	Value entry .....	505
Update .....	412, 642	Value limitation .....	407
Update data .....	743	Value range .....	241, 338
Update operating system V1.00 .....	138	Value range, marker .....	229
Update via AWS IoT Jobs .....	815	Value scaling .....	354
Update web client		VC - Value limitation .....	407
Update data .....	743	Version	
Updating firmware .....	136	User function block .....	600
Updating the firmware .....	138	Versions .....	27, 33
Updating the web client .....	743	View	
Upper limit .....	354	Communication .....	690
Upper threshold value .....	264	Visualization .....	856
User function block .....	597	Visualization devices	
Archive .....	617	Download .....	122
Calling in the main program .....	611	<b>W</b>	
Configuring .....	602	Web client	
Creating .....	598	Login guest .....	739
Exporting .....	622	Operand list .....	756
Importing .....	624	Separate operands .....	747
In an ST main program .....	614	Settings .....	751
Printing .....	630	Updating operands .....	743
Programming .....	608	Web client cycle time .....	757
Replacing .....	625	Web operand list .....	756
Same name but different content .....	618	Web server configuration .....	729
Saving .....	617	Web server startup behavior .....	733
Transferring from easySoft 7 to easySoft 8 ..	624	Weekly timer .....	244
User function block archive .....	617	What is transferred during the download .....	122
User function block green .....	611	Window discriminator .....	407
User function block storage location .....	600		

Wiring	
Grid .....	187
Wiring arrow .....	201
WORD definition .....	226
WT - Weekly timer .....	292

## X

XOR	
BV - Boolean operation .....	478

## Y

Year time switch .....	254
Yellow user function block .....	608
YT - Year time switch .....	284

## Z

Zoom function .....	744
Zoom increment .....	744

## List of Figures

---

Fig. 1: Device model with EASY-E4-...-12...C1(P) display and button controls or with EASY-E4-...-12...CX1(P) LED display for diagnostics .....	27
Fig. 2: Device models in 4SU .....	29
Fig. 3: Device models in 2SU .....	29
Fig. 4: AC input with suppression diode easyE4 AC .....	48
Fig. 5: AC input with ballast M22-XLED-T .....	49
Fig. 6: Increased input current with X2 safety capacitor .....	49
Fig. 7: Limitation of the input current through resistors .....	50
Fig. 8: An increase in input current with M22-XLED230-T .....	50
Fig. 9: Min. clearance of 3 cm .....	59
Fig. 10: Assembling a base device with expansions .....	60
Fig. 11: Assembling the base device with an easy communication module as example EASY-COM-SWD-C1 .....	61
Fig. 12: Installation on IEC/EN 60715 mounting rail .....	62
Fig. 13: Inserting a fastening bracket. ....	64
Fig. 14: Example: Screw mounting for a 4U device .....	64
Fig. 15: Remove adjacent connectors .....	65
Fig. 16: Dismantling .....	65
Fig. 17: Connecting the power supply for base devices .....	68
Fig. 18: Connecting the power supply for expansions .....	69
Fig. 19: Connecting the digital inputs on base devices .....	71
Fig. 20: Connecting the digital inputs on expansions .....	71
Fig. 21: Connect digital counter inputs .....	73
Fig. 22: Connecting the analog inputs on base devices .....	74
Fig. 23: Connecting relay outputs .....	75
Fig. 24: Connecting base device transistor outputs .....	76
Fig. 25: Connecting expansion device transistor outputs .....	76
Fig. 26: Inductive load with suppressor circuit .....	77
Fig. 27: Device parameters tab, using the EASY-E4-DC-6AE1 as an example ..	78
Fig. 28: Connecting analog inputs EASY-E4-DC-6AE1(P) .....	79
Fig. 29: Connecting analog outputs EASY-E4-DC-6AE1(P) .....	79



## List of Figures

---

Fig. 30: Connecting analog inputs EASY-E4-DC-4PE1(P)	80
Fig. 31: Expansion parameter tab, using the EASY-E4-DC-4PE1 as an example	81
Fig. 32: Slot for microSD	87
Fig. 33: Ethernet port on base device	87
Fig. 34: Inserting a memory card	88
Fig. 35: Removing the memory card	89
Fig. 36: RJ-45 socket, 8-pole	90
Fig. 37: Connecting the Ethernet cable	91
Fig. 38: Removing the Ethernet cable	92
Fig. 39: Removing the Ethernet cable	92
Fig. 40: license product certificate	94
Fig. 41: Input screen for the license product certificate No.	94
Fig. 42: License dialog box	96
Fig. 43: Options in ? menu	97
Fig. 44: InstallShield Wizard	98
Fig. 45: Step 1	99
Fig. 46: Step 2 License agreement	99
Fig. 47: Step 3 License key	99
Fig. 48: Step 4 Destination folder	100
Fig. 49: Step 4.1 Changing the destination folder	100
Fig. 50: Step 4.2 Creating your own destination folder	101
Fig. 51: Step 5 Selecting options	101
Fig. 52: Step 6 Starting the installation	101
Fig. 53: Step 7 Confirmation prompt	102
Fig. 54: Step 7 Progress display	102
Fig. 55: Step 7.1 Messages	102
Fig. 56: Step 8 Finishing	103
Fig. 57: easySoft 8 icon depending on the screen resolution or position	103
Fig. 58: LED status indication	106
Fig. 59: Example of status display on display	108
Fig. 60: Main menu in English	109
Fig. 61: Menu path in English	109
Fig. 62: Start displays for easyE4 base device in English	111
Fig. 63: Example of status display on display	112

---

Fig. 64: Startup procedure with device initialization .....	116
Fig. 65: Establishing an Ethernet connection .....	119
Fig. 66: Search for devices with an IP address .....	120
Fig. 67: Saving the found device's IP profile .....	120
Fig. 68: Selecting the easyE4 device's IP address .....	121
Fig. 69: Connection to the easyE4 device established and program transferred .....	122
Fig. 70: Offline dialog box for memory card .....	126
Fig. 71: microSD memory card drive with PROGRAM folder contains BOOT.TXT and compiled test.prg program .....	128
Fig. 72: Offline dialog box for memory card .....	130
Fig. 73: microSD memory card drive with PROGRAM folder contains BOOT.TXT and compiled test.prg program .....	132
Fig. 74: microSD memory card content when using bootloader version 1.01 ..	138
Fig. 75: microSD memory card content when using bootloader version 2.00 ..	140
Fig. 76: boot.bmp .....	147
Fig. 77: Storing the boot.bmp file .....	147
Fig. 78: Color scheme from the index during remote operation of the easyE4	151
Fig. 79: Display and keypad .....	153
Fig. 80: Example of status display on display .....	153
Fig. 81: Empty circuit diagram .....	171
Fig. 82: Fields in the circuit diagram .....	172
Fig. 83: Lighting control circuit .....	173
Fig. 84: Circuit diagram with inputs I01, I02 and output Q1 .....	173
Fig. 85: Completed circuit diagram .....	175
Fig. 86: SAVE menu option in the status line .....	175
Fig. 87: Power flow display 1 .....	177
Fig. 88: Power flow display 2 .....	177
Fig. 89: Display with zoom, power flow .....	178
Fig. 90: Display with zoom, power flow interrupted .....	178
Fig. 91: Sample program open .....	181
Fig. 92: Card setup dialog box .....	182
Fig. 93: File selection dialog box .....	183
Fig. 94: The program was transferred to the memory card. ....	184

## List of Figures

Fig. 95: Ethernet connection on PC .....	186
Fig. 96: Circuit diagram display .....	187
Fig. 97: Contactor function signal diagram .....	192
Fig. 98: Impulse relay signal diagram .....	192
Fig. 99: Set and Reset signal diagram .....	193
Fig. 100: Simultaneous triggering of Q 01 .....	193
Fig. 101: Inverse contactor function signal diagram .....	194
Fig. 102: Signal diagram of cycle pulse with rising edge .....	194
Fig. 103: Signal diagram of cycle pulse with negative edge .....	195
Fig. 104: Circuit diagram with inputs .....	196
Fig. 105: Contact legend .....	197
Fig. 106: Change contact I 03 from N/O to N/C .....	198
Fig. 107: Relay coil "Output Q" .....	199
Fig. 108: Relay coil for timing relay function block with control coil .....	199
Fig. 109: Relay coil of a NET station .....	199
Fig. 110: Circuit diagram with five contacts, invalid .....	201
Fig. 111: Circuit diagram with M marker relay .....	201
Fig. 112: Inserting a new rung .....	202
Fig. 113: The cursor buttons are wired in the circuit diagram as contacts P 01 to P 04. ....	205
Fig. 114: Switch Q1 via I1, I2, Í, or Ú .....	205
Fig. 115: I5 switches to cursor buttons. ....	205
Fig. 116: Paralleling link .....	206
Fig. 117: Power flow display .....	206
Fig. 118: 1 slave .....	212
Fig. 119: 2 slave .....	212
Fig. 120: Explanation of the function block list .....	220
Fig. 121: Manufacturer function block display in the function block editor ....	220
Fig. 122: Programming view: Selected time constant on function block input I1 and unconfirmed value of <9> entered with the keyboard .....	231
Fig. 123: Programming view: Selected timer constant on function block input I1 and unconfirmed value of <t#5m10s> entered with the keyboard .....	231
Fig. 124: Programming view: Selected timer constant on function block input I1 and unconfirmed value of <t#3h25m> entered with the keyboard .....	232

Fig. 125: Marker range map with write conflict for MW1 .....	235
Fig. 126: Signal diagram .....	248
Fig. 127: Programming view Weekly timer parameters tab .....	248
Fig. 128: Signal diagram .....	249
Fig. 129: Programming view Weekly timer parameters tab .....	249
Fig. 130: Signal diagram .....	250
Fig. 131: Programming view Weekly timer parameters tab .....	250
Fig. 132: Signal diagram .....	251
Fig. 133: Programming view Weekly timer parameters tab Settings Time overlap .....	251
Fig. 134: Signal diagram .....	252
Fig. 135: Programming view Weekly timer parameters tab - 24 hours setting .....	252
Fig. 136: Programming view Weekly timer parameters tab .....	253
Fig. 137: Year time switch parameters tab with and example in which a year range is being selected .....	258
Fig. 138: Entry screen in the programming software .....	260
Fig. 139: Entry screen in the programming software .....	260
Fig. 140: Entry screen in the programming software .....	261
Fig. 141: Entry screen in the programming software .....	261
Fig. 142: Entry screen in the programming software .....	262
Fig. 143: Entry screen in the programming software .....	262
Fig. 144: Entry screen in the programming software .....	263
Fig. 145: Signal diagram of timing relay, on-delayed (with and without random switching) .....	277
Fig. 146: Signal diagram of timing relay, on-delayed (with and without random switching) .....	278
Fig. 147: Signal diagram of timing relay, off-delayed (with/without random switching, with/without retriggering) .....	279
Fig. 148: Signal diagram of timing relay, off-delayed (with/without random switching, with/without retriggering) .....	279
Fig. 149: Operational diagrams timing relay, on and off delayed .....	280
Fig. 150: Signal diagram timing relay, single pulse 1 .....	281
Fig. 151: Signal diagram timing relay, single pulse 2 .....	281
Fig. 152: Signal diagram timing relay, single pulse .....	282
Fig. 153: Wiring the function block coils .....	283

## List of Figures

Fig. 154: Wiring of the function block contact .....	283
Fig. 155: Year time switch (new) parameters tab for YT function block with example showing all four modes .....	287
Fig. 156: Entry screen in the programming software .....	289
Fig. 157: Entry screen in the programming software .....	289
Fig. 158: Entry screen in the programming software .....	290
Fig. 159: Entry screen in the programming software .....	290
Fig. 160: Entry screen in the programming software .....	291
Fig. 161: Entry screen in the programming software .....	291
Fig. 162: Weekly timer (new) parameters tab with an example .....	294
Fig. 163: Sunrise and sunset in Bonn .....	300
Fig. 164: Sunrise and sunset in Drevja .....	300
Fig. 165: Offset; O1=-2; O2=2; Q1=1 will switch on 2 hours before sunrise and off 2 hours after sunset .....	301
Fig. 166: No offset; O1=0; O2=0; Q1=1 between sunrise and sunset .....	302
Fig. 167: Offset; O1=1; O2= -1; Q1 will switch on 1 hour after sunrise and off 1 hour before sunset .....	302
Fig. 168: Offset; O1=-2; O2=2; Q1=1 will switch on 2 hours before sunrise and off 2 hours after sunset .....	302
Fig. 169: Offset; O1=-2; O2=-2; Q1=1 will switch on 2 hours before sunrise and off 2 hours before sunset .....	303
Fig. 170: Q1 does not switch off during the summer months .....	303
Fig. 171: Q1 does not switch on during the winter months .....	304
Fig. 172: Signal diagram of counter relay .....	309
Fig. 173: Signal diagram of frequency counter .....	315
Fig. 174: Signal diagram High-speed counter .....	321
Fig. 175: CI function block counting up; QV=QV+4 .....	324
Fig. 176: CI function block counting down; QV=QV-4 .....	324
Fig. 177: Signal diagram High-speed incremental value counter .....	328
Fig. 178: Analog comparator signal diagram .....	334
Fig. 179: Parameters on the display .....	335
Fig. 180: Wiring the contacts .....	340
Fig. 181: Parameters on the device display .....	340
Fig. 182: Sample curve for hourly temperature measurement, over 7 days ....	348
Fig. 183: Wiring the contacts .....	352

Fig. 184: Parameters on the display .....	353
Fig. 185: Figure: Scaling the input values - reducing .....	354
Fig. 186: Scaling the input values - increasing .....	354
Fig. 187: Mathematical interrelationship .....	355
Fig. 188: Example of a characteristic curve for the PM function block .....	366
Fig. 189: PW pulse at the function block output when SV =1400, ME = 93 ms, PD=1000 ms .....	374
Fig. 190: PW pulse at the function block output when SV = 3218, ME = 93 ms, PD=1000 ms .....	374
Fig. 191: A constant signal is signaled at the function block output when SV = 3768, ME = 93 ms, PD=1000 ms; E1 = 1 .....	374
Fig. 192: Wiring the function block coils .....	380
Fig. 193: Wiring of the function block contact .....	380
Fig. 194: Parameters on the device display .....	381
Fig. 195: Response of the FT function block .....	382
Fig. 196: Wiring the function block coils .....	386
Fig. 197: Parameters shown on display .....	386
Fig. 198: Typical pulse profile for a stepper motor in normal mode .....	389
Fig. 199: Signal diagram for PO pulse output with specified number of pulses for I1 - possible normal mode phases .....	396
Fig. 200: Signal diagram for jog mode with specified number of steps P1 .....	398
Fig. 201: Signal diagram for jog mode with specified jog frequency, P1 after deceleration phase reached .....	399
Fig. 202: Signal diagram for jog mode with specified jog frequency, P1 not reached after deceleration phase .....	400
Fig. 203: Three-step controller schematic diagram .....	402
Fig. 204: Timing diagram for three-step controller .....	402
Fig. 205: Signal diagram for three-step controller .....	405
Fig. 206: Figure: Restriction of the input values to the specified limits. ....	407
Fig. 207: *.e80 project with circuit diagram BC in FBD .....	417
Fig. 208: Wiring the enable coil .....	418
Fig. 209: Wiring the contacts .....	418
Fig. 210: Parameters on the display .....	418
Fig. 211: Parameters on the display .....	423
Fig. 212: Wiring the trigger coil .....	424

## List of Figures

Fig. 213: Wiring the contacts .....	424
Fig. 214: Signal diagram of data function block .....	428
Fig. 215: Wiring the trigger coil .....	428
Fig. 216: Wiring of the function block contact .....	428
Fig. 217: Parameters on the display .....	428
Fig. 218: Recipe with five records; record 5 contains a mixture of values, marker bytes, marker words, and marker double words .....	442
Fig. 219: Shift register SR.: Forwards operation in BIT operating mode .....	448
Fig. 220: Shift register SR.: Backwards operation in DW operating mode .....	449
Fig. 221: Circuit diagram with EDP programming language for user example 2 .....	453
Fig. 222: Parameters on the device display .....	453
Fig. 223: Text display parameters tab for text display function block in the Programming view .....	484
Fig. 224: Default colors tab for text display .....	486
Fig. 225: Text display function block, language tab .....	486
Fig. 226: Signal diagram Text display .....	488
Fig. 227: Signal diagram for text display with text function blocks with an identical priority of 3 .....	488
Fig. 228: Text display editor with static text in the first line .....	492
Fig. 229: Character table Special characters .....	493
Fig. 230: Value display with original and double character sizes .....	494
Fig. 231: Two value displays with two characters overlapping .....	495
Fig. 232: Example showing an exact value message text .....	500
Fig. 233: Value range message text example .....	502
Fig. 234: Example of first data logger instance as a ring buffer .....	520
Fig. 235: Workspace with function block and device button .....	522
Fig. 236: Data logger tab with set parameters for the programming view .....	522
Fig. 237: Activated function block in the function block status display .....	525
Fig. 238: Acyclical Modbus TCP request - Parameters tab .....	534
Fig. 239: Overview of how function codes are used .....	535
Fig. 240: Acyclical Modbus TCP request - 2nd write request tab .....	537
Fig. 241: Function block outputs tab .....	539
Fig. 242: Signal diagram of frequency counter .....	540

---

Fig. 243: Acyclical Modbus TCP request tab .....	541
Fig. 244: Acyclical Modbus TCP request tab .....	542
Fig. 245: Wiring the function block coils .....	545
Fig. 246: Wiring of the function block contact .....	545
Fig. 247: Parameters on the display .....	546
Fig. 248: Acyclical Modbus RTU request - Parameters tab .....	550
Fig. 249: Overview of how function codes are used .....	551
Fig. 250: Acyclical Modbus master request - 2nd write request tab .....	553
Fig. 251: Function block outputs tab .....	554
Fig. 252: Signal diagram of frequency counter .....	555
Fig. 253: Acyclical Modbus RTU request tab .....	556
Fig. 254: Acyclical Modbus client request tab .....	557
Fig. 255: Wiring the function block coils .....	568
Fig. 256: Setting of the parameters .....	568
Fig. 257: Input and output states being passed between the main program and interrupt program .....	573
Fig. 258: easySoft 8 Main program Pulse counter with external direction .....	579
Fig. 259: easySoft 8 Interrupt program Pulse counter with external direction .....	579
Fig. 260: easySoft 8 Main program, two counter inputs .....	580
Fig. 261: easySoft 8 Interrupt program, two counter inputs .....	580
Fig. 262: easySoft 8 Main program Incremental counter .....	581
Fig. 263: easySoft 8 Interrupt program Incremental counter .....	581
Fig. 264: easySoft 8 Main program Frequency measurement .....	582
Fig. 265: easySoft 8 Interrupt program Frequency measurement .....	582
Fig. 266: Input and output states being passed between the main program and interrupt program .....	584
Fig. 267: easySoft 8 Main program Slope .....	588
Fig. 268: easySoft 8 Interrupt program Slope .....	588
Fig. 269: Input and output states being passed between the main program and interrupt program .....	590
Fig. 270: easySoft 8 Main program Time-controlled .....	596
Fig. 271: easySoft 8 Interrupt program Time-controlled .....	596
Fig. 272: Create user function block .....	599
Fig. 273: Configure user function block .....	603



## List of Figures

---

Fig. 274: Project view, System settings tab with Retention section .....	605
Fig. 275: Retention section: Marker bytes 1 - 32 entered and display in marker double words after another change to the System settings tab .....	606
Fig. 276: Programming view for UF Blinker1 user function block .....	608
Fig. 277: User function block comments being shown in the tab .....	610
Fig. 278: UF Blinker1 user function block used in the main program .....	611
Fig. 279: Inputs/outputs wiring .....	612
Fig. 280: Contact tab .....	613
Fig. 281: Analog contact tab .....	613
Fig. 282: Coil tab .....	613
Fig. 283: Analog coil tab .....	614
Fig. 284: easySoft 8 with function blocks in list of operands and function blocks, User function block/Project directory and User function block- /Archive directory with UF-BETest V1.00 versions with different contents .....	618
Fig. 285: easySoft 8 installation wizard .....	625
Fig. 286: Delete user function blocks dialog box .....	627
Fig. 287: Location of user function block user for comparison dialog box .....	629
Fig. 288: UF user function block .....	629
Fig. 289: Import user function block .....	630
Fig. 290: Hardwiring with relays .....	631
Fig. 291: Wiring with EASY-E4-UC-..., for example .....	631
Fig. 292: Wiring of counter and timing relay .....	631
Fig. 293: Enter parameter C01 .....	632
Fig. 294: Enter ParameterT01 .....	632
Fig. 295: Testing the circuit diagram .....	633
Fig. 296: Testing the circuit diagram +10 .....	633
Fig. 297: Doubling the flashing frequency .....	633
Fig. 298: Project view, System settings tab with Retention section .....	651
Fig. 299: Retention section: Marker bytes 1 - 32 entered and display in marker double words after another change to the System settings tab .....	652
Fig. 300: Password assignment .....	655
Fig. 301: Password submenu .....	656
Fig. 302: Submenu for changing the password .....	656
Fig. 303: How the EDP evaluates circuit diagrams and function blocks .....	666

---

Fig. 304: Programming view/Sample program in FBD .....	674
Fig. 305: Online Communication view with display marker, causing the device display to flash green .....	675
Fig. 306: easyE4 input assigned a switch .....	676
Fig. 307: Delay times for evaluating an DC input signal and an activated debounce .....	677
Fig. 308: Switching behavior with debounce deactivated .....	678
Fig. 309: Delay times for evaluating an AC input signal without I-DEBOUNCE and for an activated I-DEBOUNCE .....	679
Fig. 310: Switching behavior of AC input signal with DEBOUNCE activated ..	679
Fig. 311: Switching behavior of AC input signal with DEBOUNCE deactivated	680
Fig. 312: communication diagram easyE4 .....	690
Fig. 313: ONLINE Project view with devices colored differently based on their compatibility .....	698
Fig. 314: Selection of NET station .....	706
Fig. 315: NET configuration with project and program .....	706
Fig. 316: easyE4 certificate chain .....	712
Fig. 317: Installing easySoft 8 with the Eaton easyE4 root certificate installation option enabled .....	713
Fig. 318: NET diagram .....	721
Fig. 319: NET-ID dialog box used to assign a NET ID when adding a new base device .....	725
Fig. 320: NET tab for the selected base device in the NET group .....	726
Fig. 321: Project view, Webserver tab .....	728
Fig. 322: Web server passwords and user names dialog box .....	733
Fig. 323: Web Client, home page .....	737
Fig. 324: Web Client login dialog box .....	738
Fig. 325: Web Client login dialog box .....	739
Fig. 326: Web Client, Display .....	744
Fig. 327: Web Client, Operands .....	745
Fig. 328: Web Client, NET operands .....	746
Fig. 329: Web Client, Separate operands .....	748
Fig. 330: Web Client, Diagnostics .....	750
Fig. 331: Web Client, General settings .....	751
Fig. 332: Web Client, Network settings .....	752

## List of Figures

Fig. 333: Web Client, E-mail settings .....	753
Fig. 334: Web Client, API key .....	754
Fig. 335: Web Client, System update .....	755
Fig. 336: Web Client, Web Client .....	755
Fig. 337: E-mail tab .....	759
Fig. 338: Email tab with the settings from the example .....	763
Fig. 339: Ethernet tab with settings from the example .....	764
Fig. 340: Email example when the operating mode changes .....	765
Fig. 341: Alarm function block tab with parameters from the example and FBD program with alarm function block and P button P01 .....	767
Fig. 342: Example email when triggered by alarm function block AL01 .....	768
Fig. 343: Example showing an easyE4 control relay with I/O expansions and an easy communication module EASY-COM-SWD-... .....	771
Fig. 344: Device model with 2SU .....	772
Fig. 345: Connecting the EASY-COM-SWD-... power supply .....	774
Fig. 346: Connecting EASY-COM-SWD-... .....	776
Fig. 347: Work pane with base device and communication module; catalog expanded with "SWD" tab .....	780
Fig. 348: General diagram: The easyE4, set up as a Modbus RTU master, communicates with the devices set up as Modbus RTU slaves (DE1, DC1, DG1, DA1, easyE4, and others) .....	782
Fig. 349: Versions .....	784
Fig. 350: Connecting EASY-COM-RTU-... outputs .....	786
Fig. 351: Connecting the EASY-COM-RTU-... power supply .....	787
Fig. 352: Work pane with base device and EASY-COM-RTU-M1 master com- munication module .....	790
Fig. 353: Example: Access with a user's own AWS account .....	797
Fig. 354: Example with easyE4 in AWS; objects - things - available for selec- tion .....	798
Fig. 355: Example: AWS Certificates tab .....	798
Fig. 356: Example: AWS Device Shadows tab .....	799
Fig. 357: Example: AWS Device Classic Shadow tab .....	799
Fig. 358: Example: AWS Activity tab .....	800
Fig. 359: Example: AWS Device Classic Shadow, Editor .....	800
Fig. 360: <a href="https://pages.awscloud.com/IAM-communication-pref-">https://pages.awscloud.com/IAM-communication-pref-</a> .....	802

---

erences.html .....	
Fig. 361: <a href="https://signin.aws.amazon.com">https://signin.aws.amazon.com</a> .....	806
Fig. 362: Example: AWS console; selecting the IoT Core service .....	806
Fig. 363: Example: AWS console; selecting the IoT Core service .....	807
Fig. 364: easyE4 Wizard, Credentials .....	810
Fig. 365: easyE4 Wizard, certificate .....	811
Fig. 366: easyE4 Wizard, certificate .....	811
Fig. 367: Deregistering an easyE4 .....	812
Fig. 368: Excerpt from Python script .....	813
Fig. 369: Example: Access to the MQTT test client in the event of data transfer activities .....	815
Fig. 370: Project view, AWS, AWS settings tab .....	816
Fig. 371: Create Order .....	820
Fig. 372: Creating a job, step 1 .....	821
Fig. 373: Creating a job, step 2, job target .....	821
Fig. 374: Creating a job, step 2, job file .....	822
Fig. 375: Example of an object (thing) URL for the job file in the bucket .....	822
Fig. 376: Creating a job, step 2, job file .....	823
Fig. 377: Creating a job, step 3, job configuration .....	823
Fig. 378: Example of a bucket with two job documents and the corresponding files. ....	824
Fig. 379: Bucket example: Uploading files .....	825
Fig. 380: Update-ID .....	826
Fig. 381: In Online mode: Communication view, Diagnostics buffer tab .....	829
Fig. 382: Example: easyE4 as Modbus TCP Client .....	831
Fig. 383: Example: easyE4 as Modbus TCP Server .....	832
Fig. 384: Work pane with base device and Modbus TCP server module .....	834
Fig. 385: Device information tab .....	834
Fig. 386: Expansion parameter tab for Modbus TCP servers Expansion parameter tab for Modbus TCP servers Expansion parameter tab for Modbus TCP servers .....	835
Fig. 387: Address range addressing: .....	837
Fig. 388: Cyclical data tab with sample function codes that have been configured and added range boxes .....	838
Fig. 389: Overview of cyclical data function codes .....	841

## List of Figures

---

Fig. 390: Assigned operands tab after defining FC1, FC2, and FC4; bit inputs R4R_IR40x0 and R4R_IR40x1 have already been assigned to base device operands I17 and I18. ....	842
Fig. 391: Assigned operands tab; bit input R2R_DI20 has already been assigned to base device operand I19. ....	843
Fig. 392: Modbus TCP information tab .....	844
Fig. 393: Mirroring of the easyE4 display on the easyE RTD Standard .....	857
Fig. 394: Application example for an easyE RTD Advanced .....	858
Fig. 395: Visualization on HMI operating terminal .....	859
Fig. 396: Example of code display on display .....	865
Fig. 397: Dimensions in mm (Inch) Base devices EASY-E4-...-12...C1(P) .....	873
Fig. 398: Dimensions in mm (Inch) Base devices EASY-E4-...-12...CX1(P) .....	874
Fig. 399: Dimensions in mm (Inch) extensions 4SU .....	875
Fig. 400: Dimensions in mm (Inch) extensions 2SU .....	876
Fig. 401: Dimensions in mm (Inch) .....	876
Fig. 402: Dimensions in mm (Inch) extensions 2SU .....	877
Fig. 403: Dimensions in mm (Inch) extensions 2SU .....	877
Fig. 404: Hardware (HW) / firmware (FW)compatibility .....	881
Fig. 405: Circuit diagram for easyE4 chaser light .....	897

## Glossary

---

### Client

The term "client" refers to an application that requests specific services from a server.

---

### URL

Uniform Resource Locator

---

\*

### \*.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

### \*.prg

The program created with easySoft is compiled together with the project information and stored on the microSD card as a PRG file.

### \*.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

### API

Application Programming Interface

### Application

Short for "application software," a computer program that performs a function useful to the user.

### AWS

Amazon Web Services, a Cloud platform

---

## 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.

**CORS**

Cross-Origin Resource Sharing

**CRC**

Cyclic Redundancy Check, CRC

**CSR**

Certificate signing request

---

**D****DCF77**

German long-wave time signal, Frankfurt frequency 77 kHz

**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.

**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. DNS records are the IP addresses for these servers.

**DST**

Daylight Saving Time

---

**E****easyConnect**

Data connection for easyE4 with each other via bus connector plug

**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.

### FQDN

Fully-Qualified Domain Name

### FTP

File Transfer Protocol

---

## G

### 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

### IoT

Internet of Things

### IP Address

IP addresses are 32 bits (4 bytes) long and are used to uniquely identify networks, sub-networks, 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).

---

## L

### LAN

Local Area Network

### LD

Ladder diagram - programming method

### Lean Automation

Eaton uses this concept "" to provide users in the machine building and plant engineering industries with unparalleled freedom so that they can design creative and profitable solutions.



**Lean Solution**

Lean automation strategy in which the I/O level is integrated directly into switchgear.

**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"

**MQTT**

Message Queuing Telemetry Transport (M2M)

---

**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.

**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

---

## **R**

### **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

### **SmartWire-DT**

Eaton communication system

### **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.

## **SWD**

Abbreviation of SmartWire-DT

## **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**

### **Tabs**

Subpages in a dialog box or object

### **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**

### **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.

## **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 is an intelligent power management company dedicated to improving the quality of life and protecting the environment for people everywhere. We are guided by our commitment to do business right, to operate sustainably and to help our customers manage power – today and well into the future. By capitalizing on the global growth trends of electrification and digitalization, we're accelerating the planet's transition to renewable energy, helping to solve the world's most urgent power management challenges, and doing what's best for our stakeholders and all of society.

For more information, please visit [Eaton.com](https://www.eaton.com).



*Powering Business Worldwide*

**Eaton Industries GmbH**  
Hein-Moeller-Str. 7–11  
D-53115 Bonn

© 2018 Eaton  
All rights reserved.  
04/25 MN050009EN