BECKHOFF New Automation Technology

Documentation | EN

EL407x

EtherCAT Terminals, Analog Output, multi-function, ±10 V, ±20 mA, 16 bit, 2 ksps



2026-02-27 | Version: 1.1.0

Table of contents

1 Foreword 5
1.1 Product overview of analog multi-functional terminals 5
1.2 Notes on the documentation 6
1.3 Guide through documentation 7
1.4 Safety instructions 8
1.5 Documentation issue status 9
1.6 Version identification of EtherCAT devices 10
1.6.1 General notes on marking 10
1.6.2 Version identification of EL terminals 11
1.6.3 Beckhoff Identification Code (BIC) 12
1.6.4 Electronic access to the BIC (eBIC) 14
2 Product description 16
2.1 EL407x 16
2.1.1 Introduction 16
2.1.2 Technical data EL407x 17
2.1.3 Connection, display and diagnostics 18
2.2 Further documentation for I/O components with analog in and outputs 27
2.3 Start 28
3 Basics communication 29
3.1 EtherCAT basics 29
3.2 EtherCAT cabling – wire-bound 29
3.3 General notes for setting the watchdog 31
3.4 EtherCAT State Machine 33
3.5 CoE Interface 35
4 Mounting and wiring 41
4.1 Instructions for ESD protection 41
4.2 Installation on mounting rails 42
4.3 Positioning of passive Terminals 45
4.4 Note - power supply 46
4.5 Connection 47
4.5.1 Connection system 47
4.5.2 Wiring 49
4.5.3 Shielding 50
4.6 Installation positions 51
4.7 Disposal 54
5 Commissioning 55
5.1 TwinCAT/EtherCAT slave 55
5.1.1 TwinCAT Quick Start 55
5.1.2 TwinCAT Development Environment 82
5.1.3 General Commissioning Instructions for an EtherCAT Slave 125
5.2 Basic Function Principles 133
5.2.1 Commissioning EL407x 133
5.2.2 Commissioning analog output 139

5.3 Overview of parameter objects (CoE) ... 168
5.3.1 Restore objects ... 168
5.3.2 Configuration data ... 169
5.3.3 Input data ... 172
5.3.4 Output data ... 173
5.3.5 Information and diagnostic data ... 174
5.3.6 Standard objects ... 176
6 Appendix ... 190
6.1 EtherCAT AL Status Codes ... 190
6.2 Firmware compatibility ... 191
6.3 Firmware Update ... 192
6.3.1 Device description ESI file/XML ... 193
6.3.2 Firmware explanation ... 196
6.3.3 Updating controller firmware *.efw ... 197
6.3.4 FPGA firmware *.rbf ... 199
6.3.5 Simultaneous updating of several EtherCAT devices ... 203
6.4 Restoring the delivery state ... 204
6.5 Support and Service ... 206

1 Foreword

1.1 Product overview of analog multi-functional terminals

EL4072 [▶ 16]

EtherCAT Terminal, 2-channel analog output, multi-function, ±10 V, ±20 mA, 16 bit, 2 ksps

EL4074 [▶ 16]

EtherCAT Terminal, 4-channel analog output, multi-function, ±10 V, ±20 mA, 16 bit, 2 ksps

EL4078 [▶ 16]

EtherCAT Terminal, 8-channel analog output, multi-function, ±10 V, ±20 mA, 16 bit, 2 ksps

1.2 Notes on the documentation

Intended audience

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with the applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning these components.
The qualified personnel is obliged to always use the currently valid documentation.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.

We reserve the right to revise and change the documentation at any time and without prior announcement.

No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar® and XTS® are registered trademarks of and licensed by Beckhoff Automation GmbH.

Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.



EtherCAT® is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.
The reproduction, distribution and utilization of this document as well as the communication of its contents to others without express authorization are prohibited.
Offenders will be held liable for the payment of damages. All rights reserved in the event of the grant of a patent, utility model or design.

Third-party brands

Trademarks of third parties may be used in this documentation. You can find the trademark notices here:
https://www.beckhoff.com/trademarks

1.3 Guide through documentation

NOTICE



Further components of documentation

This documentation describes device-specific content. It is part of the modular documentation concept for Beckhoff I/O components. For the use and safe operation of the device / devices described in this documentation, additional cross-product descriptions are required, which can be found in the following table.

Title	Description
EtherCAT System Documentation (PDF)	• System overview • EtherCAT basics • Cable redundancy • Hot Connect • EtherCAT devices configuration
I/O Analog Manual (PDF)	Notes on I/O components with analog in and outputs
Infrastructure for EtherCAT/Ethernet (PDF)	Technical recommendations and notes for design, implementation and testing
Software Declarations I/O (PDF)	Open source software declarations for Beckhoff I/O components

The documentations can be viewed at and downloaded from the Beckhoff website (www.beckhoff.com) via:

- the “Documentation and Download” area of the respective product page,
- the Download finder,
- the Beckhoff Information System.

If you have any suggestions or proposals for our documentation, please send us an e-mail stating the documentation title and version number to: documentation@beckhoff.com

1.4 Safety instructions

Safety regulations

Please note the following safety instructions and explanations!
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

Personal injury warnings

⚠ DANGER
Hazard with high risk of death or serious injury.

⚠ WARNING
Hazard with medium risk of death or serious injury.

⚠ CAUTION
There is a low-risk hazard that could result in medium or minor injury.

Warning of damage to property or environment

NOTICE
The environment, equipment, or data may be damaged.

Information on handling the product



This information includes, for example:
recommendations for action, assistance or further information on the product.

1.5 Documentation issue status

Version	Comment
1.1.0	• Update Technical data • Update structure
1.0.0	• 1st public issue
0.1.0 – 0.4.0	• provisional documentations for EL407x

1.6 Version identification of EtherCAT devices

1.6.1 General notes on marking

Designation

A Beckhoff EtherCAT device has a 14-digit designation, made up of

- family key
- type
- version
- revision

Example	Family	Type	Version	Revision
EL3314-0000-0016	EL terminal 12 mm, non-pluggable connection level	3314 4-channel thermocouple terminal	0000 basic type	0016
ES3602-0010-0017	ES terminal 12 mm, pluggable connection level	3602 2-channel voltage measurement	0010 high-precision version	0017
CU2008-0000-0000	CU device	2008 8-port fast ethernet switch	0000 basic type	0000

Notes

- The elements mentioned above result in the **technical designation**. EL3314-0000-0016 is used in the example below.
- EL3314-0000 is the order identifier, in the case of “-0000” usually abbreviated to EL3314. “-0016” is the EtherCAT revision.
- The **order identifier** is made up of
  - family key (EL, EP, CU, ES, KL, CX, etc.)
  - type (3314)
  - version (-0000)
- The **revision** -0016 shows the technical progress, such as the extension of features with regard to the EtherCAT communication, and is managed by Beckhoff.
  In principle, a device with a higher revision can replace a device with a lower revision, unless specified otherwise, e.g. in the documentation.
  Associated and synonymous with each revision there is usually a description (ESI, EtherCAT Slave Information) in the form of an XML file, which is available for download from the Beckhoff web site.
  From 2014/01 the revision is shown on the outside of the IP20 terminals, see Fig. *“EL2872 with revision 0022 and serial number 01200815”*.
- The type, version and revision are read as decimal numbers, even if they are technically saved in hexadecimal.

1.6.2 Version identification of EL terminals

The serial number/ data code for Beckhoff IO devices is usually the 8-digit number printed on the device or on a sticker. The serial number indicates the configuration in delivery state and therefore refers to a whole production batch, without distinguishing the individual modules of a batch.

Structure of the serial number: **KK YY FF HH**	Example with serial number 12 06 3A 02:
KK - week of production (CW, calendar week)	12 - production week 12
YY - year of production	06 - production year 2006
FF - firmware version	3A - firmware version 3A
HH - hardware version	02 - hardware version 02



Fig. 1: EL2872 with revision 0022 and serial number 01200815

1.6.3 Beckhoff Identification Code (BIC)

The Beckhoff Identification Code (BIC) is increasingly being applied to Beckhoff products to uniquely identify the product. The BIC is represented as a Data Matrix Code (DMC, code scheme ECC200), the content is based on the ANSI standard MH10.8.2-2016.

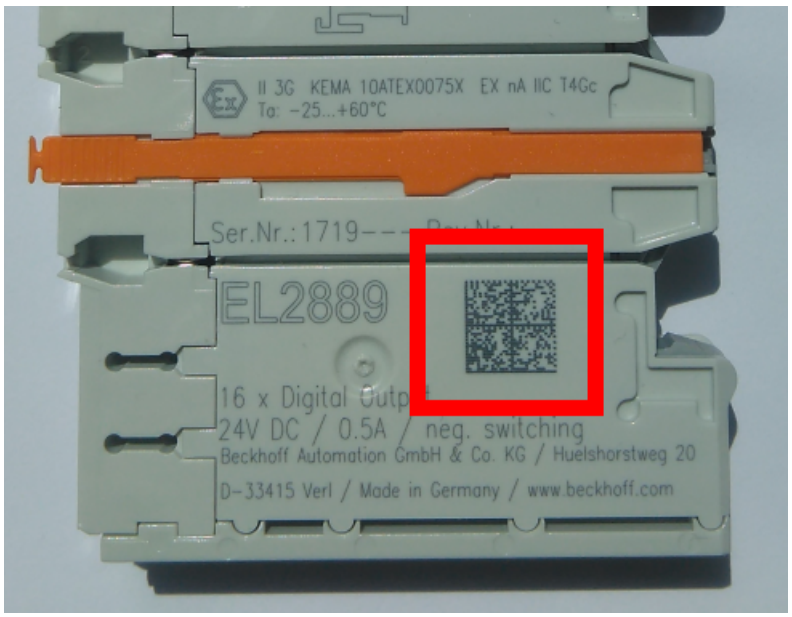


Fig. 2: BIC as data matrix code (DMC, code scheme ECC200)

The BIC will be introduced step by step across all product groups.

Depending on the product, it can be found in the following places:

- on the packaging unit
- directly on the product (if space suffices)
- on the packaging unit and the product

The BIC is machine-readable and contains information that can also be used by the customer for handling and product management.

Each piece of information can be uniquely identified using the so-called data identifier (ANSI MH10.8.2-2016). The data identifier is followed by a character string. Both together have a maximum length according to the table below. If the information is shorter, spaces are added to it.

Following information is possible, positions 1 to 4 are always present, the other according to need of production:

Position	Type of information	Explanation	Data identifier	Number of digits incl. data identifier	Example
1	Beckhoff order number	**Beckhoff order number**	1P	8	**1P**072222
2	Beckhoff Traceability Number (BTN)	**Unique serial number, see note below**	SBTN	12	**SBTN**k4p562d7
3	Article description	**Beckhoff article description, e.g. EL1008**	1K	32	**1K**EL1809
4	Quantity	**Quantity in packaging unit, e.g. 1, 10, etc.**	Q	6	**Q**1
5	Batch number	Optional: Year and week of production	2P	14	**2P**401503180016
6	ID/serial number	Optional: Present-day serial number system, e.g. with safety products	51S	12	**51S**678294
7	Variant number	Optional: Product variant number on the basis of standard products	30P	12	**30P**F971, 2*K183
...					

Further types of information and data identifiers are used by Beckhoff and serve internal processes.

Structure of the BIC

Example of composite information from positions 1 to 4 and with the above given example value on position 6. The data identifiers are highlighted in bold font:

1P072222**SBTN**k4p562d7**1K**EL1809 **Q**1 **51S**678294

Accordingly as DMC:



Fig. 3: Example DMC **1P**072222**SBTN**k4p562d7**1K**EL1809 **Q**1 **51S**678294

BTN

An important component of the BIC is the Beckhoff Traceability Number (BTN, position 2). The BTN is a unique serial number consisting of eight characters that will replace all other serial number systems at Beckhoff in the long term (e.g. batch designations on IO components, previous serial number range for safety products, etc.). The BTN will also be introduced step by step, so it may happen that the BTN is not yet coded in the BIC.

NOTICE

This information has been carefully prepared. However, the procedure described is constantly being further developed. We reserve the right to revise and change procedures and documentation at any time and without prior notice. No claims for changes can be made from the information, illustrations and descriptions in this documentation.

1.6.4 Electronic access to the BIC (eBIC)

Electronic BIC (eBIC)

The Beckhoff Identification Code (BIC) is applied to the outside of Beckhoff products in a visible place. If possible, it should also be electronically readable.

The interface that the product can be electronically addressed by is crucial for the electronic readout.

K-bus devices (IP20, IP67)

Currently, no electronic storage or readout is planned for these devices.

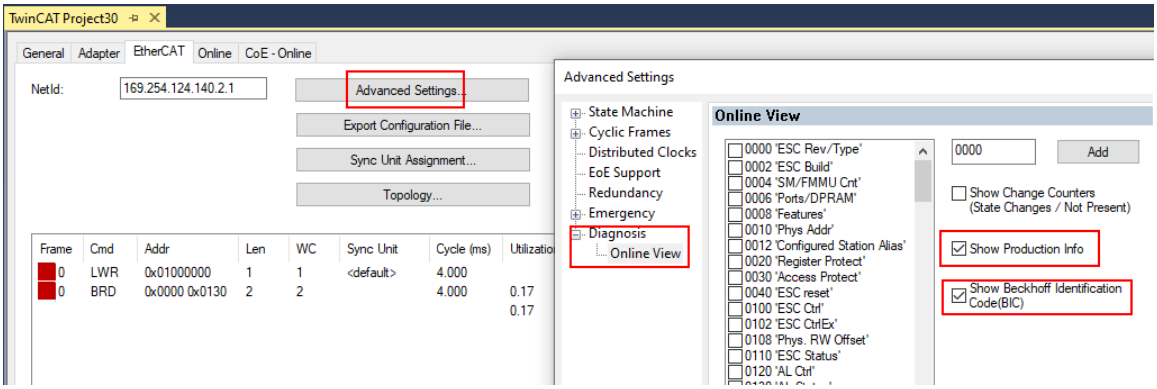
EtherCAT devices (IP20, IP67)

All Beckhoff EtherCAT devices have an ESI-EEPROM which contains the EtherCAT identity with the revision number. The EtherCAT slave information, also colloquially known as the ESI/XML configuration file for the EtherCAT master, is stored in it. See the corresponding chapter in the EtherCAT system manual (Link) for the relationships.

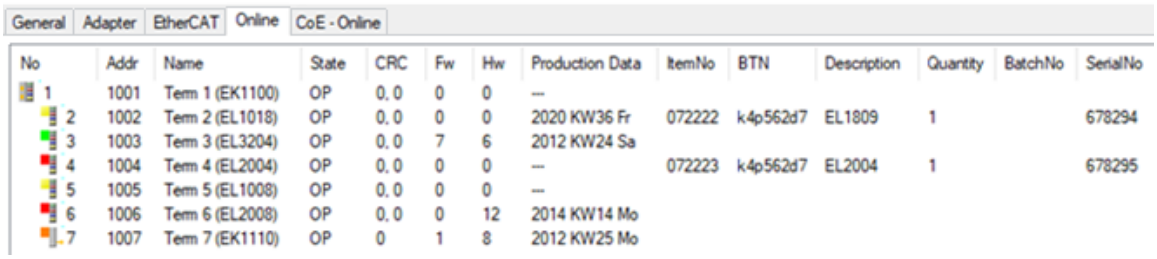
Beckhoff also stores the eBIC in the ESI-EEPROM. The eBIC was introduced into Beckhoff IO production (terminals, box modules) in 2020; as of 2023, implementation is largely complete.

The user can electronically access the eBIC (if present) as follows:

- With all EtherCAT devices, the EtherCAT master (TwinCAT) can read the eBIC from the ESI-EEPROM
  - From TwinCAT 3.1 build 4024.11, the eBIC can be displayed in the online view.
  - To do this, check the "Show Beckhoff Identification Code (BIC)" checkbox under EtherCAT → Advanced Settings → Diagnostics:

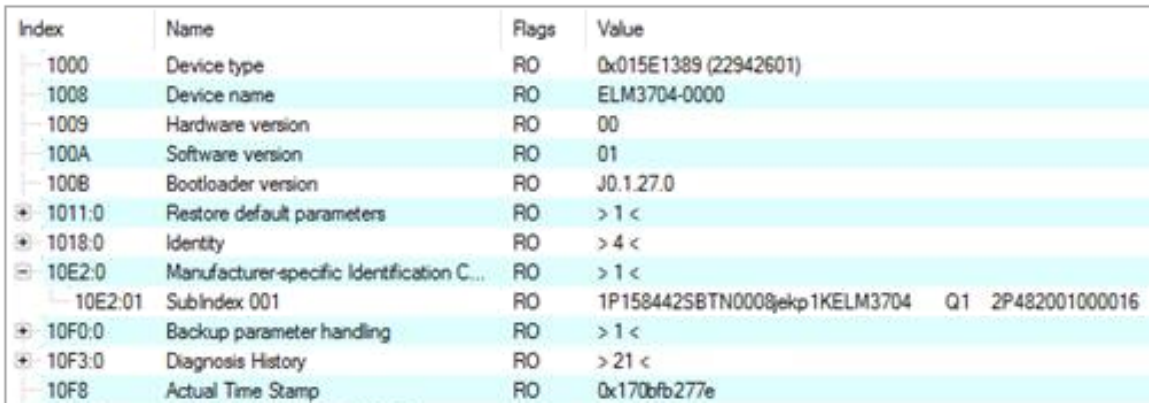


  - The BTN and its contents are then displayed:



  - Note: As shown in the figure, the production data HW version, FW version, and production date, which have been programmed since 2012, can also be displayed with "Show production info".
  - Access from the PLC: From TwinCAT 3.1. build 4024.24, the functions *FB_EcReadBIC* and *FB_EcReadBTN* for reading into the PLC are available in the Tc2_EtherCAT library from v3.3.19.0.
- EtherCAT devices with a CoE directory may also have the object 0x10E2:01 to display their own eBIC, which can also be easily accessed by the PLC:

- The device must be in PREOP/SAFEOP/OP for access:



Index	Name	Flags	Value
1000	Device type	RO	0x015E1389 (22942601)
1008	Device name	RO	ELM3704-0000
1009	Hardware version	RO	00
100A	Software version	RO	01
100B	Bootloader version	RO	J0.1.27.0
1011:0	Restore default parameters	RO	> 1 <
1018:0	Identity	RO	> 4 <
10E2:0	Manufacturer-specific Identification C...	RO	> 1 <
10E2:01	SubIndex 001	RO	1P158442SBTN0008jekp1KELM3704 Q1 2P482001000016
10F0:0	Backup parameter handling	RO	> 1 <
10F3:0	Diagnosis History	RO	> 21 <
10F8	Actual Time Stamp	RO	0x170bfb277e

  - The object 0x10E2 will be preferentially introduced into stock products in the course of necessary firmware revision.
  - From TwinCAT 3.1. build 4024.24, the functions *FB_EcCoEReadBIC* and *FB_EcCoEReadBTN* for reading into the PLC are available in the Tc2_EtherCAT library from v3.3.19.0
- The following auxiliary functions are available for processing the BIC/BTN data in the PLC in *Tc2_Utilities* as of TwinCAT 3.1 build 4024.24
  - F_SplitBIC: The function splits the Beckhoff Identification Code (BIC) sBICValue into its components using known identifiers and returns the recognized substrings in the ST_SplittedBIC structure as a return value
  - BIC_TO_BTN: The function extracts the BTN from the BIC and returns it as a return value
- Note: If there is further electronic processing, the BTN is to be handled as a string(8); the identifier "SBTN" is not part of the BTN.
- Technical background
  The new BIC information is written as an additional category in the ESI-EEPROM during device production. The structure of the ESI content is largely dictated by the ETG specifications, therefore the additional vendor-specific content is stored using a category in accordance with the ETG.2010. ID 03 tells all EtherCAT masters that they may not overwrite these data in the event of an update or restore the data after an ESI update.
  The structure follows the content of the BIC, see here. The EEPROM therefore requires approx. 50..200 bytes of memory.
- Special cases
  - If multiple hierarchically arranged ESCs are installed in a device, only the top-level ESC carries the eBIC information.
  - If multiple non-hierarchically arranged ESCs are installed in a device, all ESCs carry the eBIC information.
  - If the device consists of several sub-devices which each have their own identity, but only the top-level device is accessible via EtherCAT, the eBIC of the top-level device is located in the CoE object directory 0x10E2:01 and the eBICs of the sub-devices follow in 0x10E2:nn.

PROFIBUS, PROFINET, and DeviceNet® devices

Currently, no electronic storage or readout is planned for these devices.

2 Product description

2.1 EL407x

2.1.1 Introduction

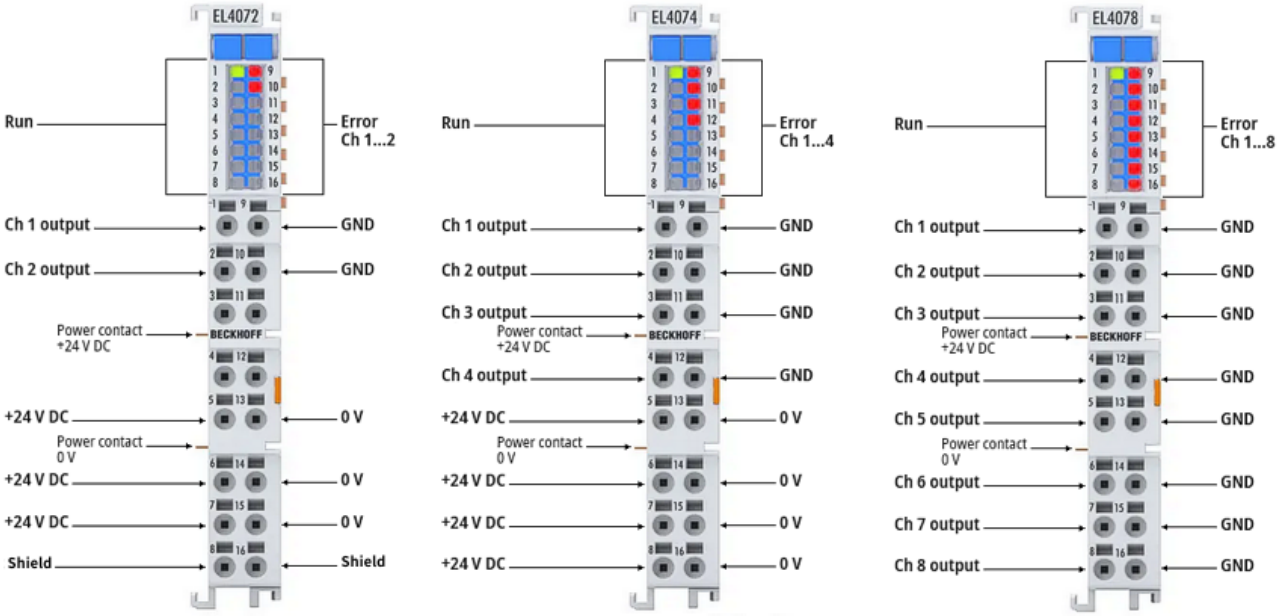


Fig. 4: EL407x

EtherCAT Terminals, analog output, multi-function, ±10 V, ±20 mA, 16 bit, 2 ksps

The analog output terminals EL4072, EL4074 and EL4078 output 10 V/20 mA standard signals with a resolution of 16 bit and up to 2000 values per second. Each channel can be parameterized individually via the controller/TwinCAT. All standard signals in the range of ±10 V and ±20 mA can be output per channel, such as 0...10 V and 4…20 mA. Each channel reports a short circuit in the voltage output or a wire break in the current output. The current outputs can also drive high load impedances. The outputs are single-ended, which means that each channel connection point is already connected to +24 V or GND in the terminal and does not need to be wired separately. The EL4072 and EL4074 have 24 V/0 V contacts for the supply of the actuators. With a technical output range of ±107% of the nominal range, the channels can even transport unconventional setpoints, such as for error reporting purposes.

The terminals mentioned are largely identical in function except for the number of channels and are therefore referred to as EL407x in the following.

Quick links

- EtherCAT basics
- Basic Function Principles [▶ 133]
- Connection, display and diagnostics [▶ 18]
- Commissioning [▶ 55]
- CoE object description and parameterization [▶ 168]

2.1.2 Technical data EL407x

Technical data	EL4072	EL4074	EL4078
Number of outputs	2	4	8
Wiring	2-wire		
Ground reference	single-ended		
Max. sampling rate	min. 500 µs (max. 2 ksps) per channel		
Sampling type	Multiplex, approx. 25 µs delay between channels n/n+1		
Load	Voltage: > 2 kΩ (short-circuit proof) Current: < 750 Ω	Voltage: > 2 kΩ (short-circuit proof) Current: < 500 Ω	Voltage: > 2 kΩ (short-circuit proof) Current: < 300 Ω
Output range, nominal	-10 V / 0 V…+10 V -20 mA / 0 mA / +4 mA…+20 mA Observe note [1]		
Output end value (AEW, FSV)	10 V / 20 mA		
Output range, technical	-10.73 V…+10.73 V -21.47 mA…+21.47 mA (AEW_{techn}) Observe note [1]		
Resolution	16 bit (including sign)		
Output error / uncertainty	< ±0.07 $\%_{AEW}$ typ. (voltage and current, at 25 °C ±5 °C, otherwise < ±0.1 $\%_{AEW}$, with averaging) [2]		
Temperature coefficient	Tk_{Gain}: 10 ppm/K typ. [2]		
Noise	Voltage: ±0.005 $\%_{AEW}$ typ. (in the range 0..5 kHz) [2] Current: ±0.017$\%_{AEW}$ typ. (in the range 0..5 kHz, at 200 Ω) [2]		
Channel crosstalk	Voltage: < -70 dB typ. [2] Current: < -60 dB typ. [2]		
Load type	Ohmic		
Electrical isolation channel/ channel	no		

[1] EL4078: if more than 4 channels are configured for current output, the remaining voltage channels can only be controlled up to max. ±5 V for thermal reasons.

[2] Preliminary information

Technical data	General		
Power supply	via power contacts, 24 V DC (-15 % / +20 %)		
Distributed clocks	–		
Electrical isolation channel/ SGND	functional separation 707 V DC (type test)		
Electrical isolation channel/E-bus	functional separation 707 V DC (type test)		
Current consumption of power contacts	60 mA typ. (required for operation of the outputs, specification at full load, additional current requirement at +24 V/0 V connected loads)		
Current consumption via E-bus	100 mA typ.	125 mA typ.	180 mA typ.
Min. EtherCAT cycle time	50 µ		
Special features	107 % measuring range/output range, output diagnostics		
Weight	approx. 55 g		
Operating/storage temperature	-25 °C…+60 °C / -40 °C…+85 °C		-25 °C…+55 °C / -40 °C…+85 °C
Relative humidity	95%, no condensation		
EMC immunity/emission	conforms to EN 61000-6-2/EN 61000-6-4		
Vibration/shock resistance	conforms to EN 60068-2-6/EN 60068-2-27		
Protection rating/installation position	IP20/any		
Approvals/markings*)	CE, UKCA, EAC		

*) Real applicable approvals/markings see type plate on the side (product marking).

2.1.3 Connection, display and diagnostics

2.1.3.1 EL4072

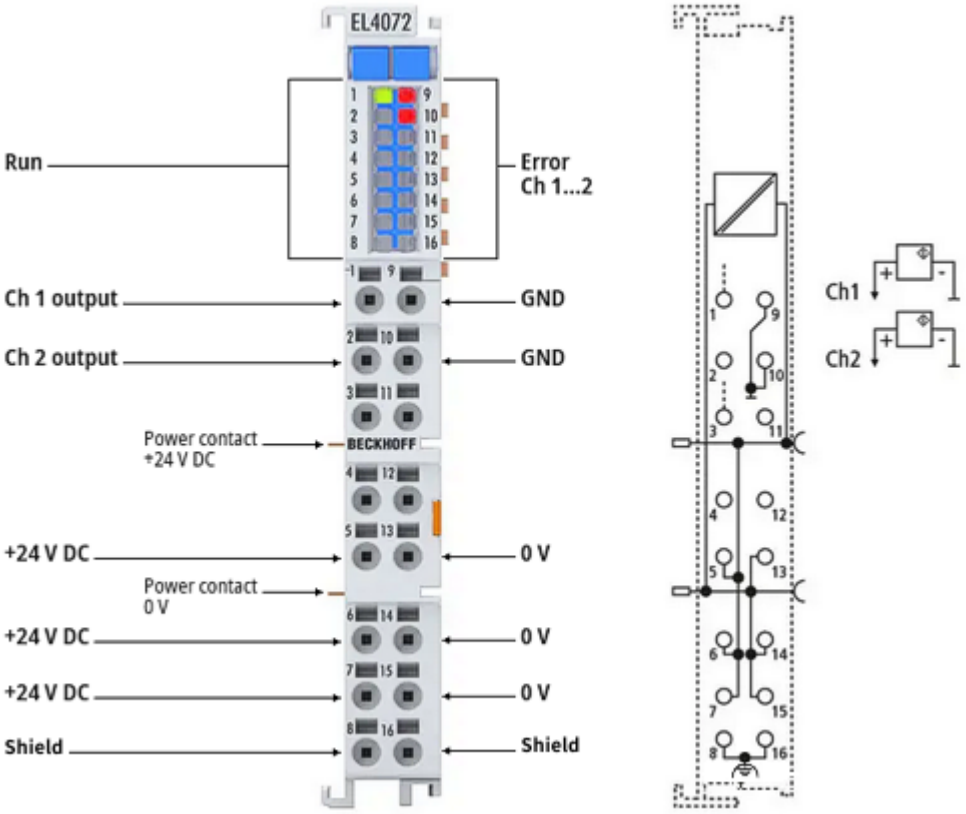


Fig. 5: EL4072 LED and connection

LED	Color	Meaning	
RUN	green	These LEDs indicate the terminal's operating state:	
		off	State of the EtherCAT State Machine [▶ 108]: **INIT** = initialization of the terminal
		flashing	State of the EtherCAT State Machine: **PREOP** = function for mailbox communication and different default settings set
		Single flash	State of the EtherCAT State Machine: **SAFEOP** = verification of the Sync Manager [▶ 108] channels and the distributed clocks. Outputs remain in safe state
		on	State of the EtherCAT State Machine: **OP** = normal operating state; mailbox and process data communication is possible
		flickering	State of the EtherCAT State Machine: **BOOTSTRAP** = function for Firmware updates [▶ 192] of the terminal
ERROR Ch1, 2	red	- Analog output overload: wire break at I output, short circuit at U output - Setpoint exceeds Limiter: below low limit /above high limit - Internal error DAC	

NOTICE

Cable lengths > 30 m

For longer cable lengths > 30 m, suitable overvoltage protection must be provided (e.g. EL9540-0010) if corresponding interference could affect the signal cable.

EL4072 connection

Terminal point	No.	Description	Internally connected with connection	Max. current carrying capacity *)
Ch1 output	1	Voltage/current output, channel 1	-	overload protected
Ch2 output	2	Voltage/current output, channel 2	-	Overload protected
n.c.	3	-	-	-
n.c.	4	-	-	-
+24 V DC	5	+24 V	6; 7, +24 V power contact	1 A
+24 V DC	6	+24 V	5, 7, +24 V power contact	1 A
+24 V DC	7	+24 V	5, 6, +24 V power contact	1 A
Shield	8	Shield connection	16	1 A ***)
GND	9	Analog ground (reference potential for Ch1...Ch2)	10	100 mA **)
GND	10	Analog ground (reference potential for Ch1...Ch2)	9	100 mA **)
n.c.	11	-	-	-
n.c.	12	-	-	-
0 V	13	0 V	14, 15, 0 V power contact	1 A
0 V	14	0 V	13, 15, 0 V power contact	1 A
0 V	15	0 V	13, 14, 0 V power contact	1 A
Shield	16	Shield connection	8	1 A ***)

*) Constant current; short-term higher currents are to be avoided and can cause thermal overload (damage)

**) The "GND" potential is internally coupled to the "0V" potential, but is subject to a reduced current carrying capacity

***) The connection point connects directly to the DIN rail contact.
Shield connections should generally be de-energized (DC)! See analog technical information in the Beckhoff analog manual [▶ 7].
If possible, an external shield connection, e.g. via Beckhoff ZB8511 or similar, is preferable.

Voltage output 0...10 V / -10...+10 V

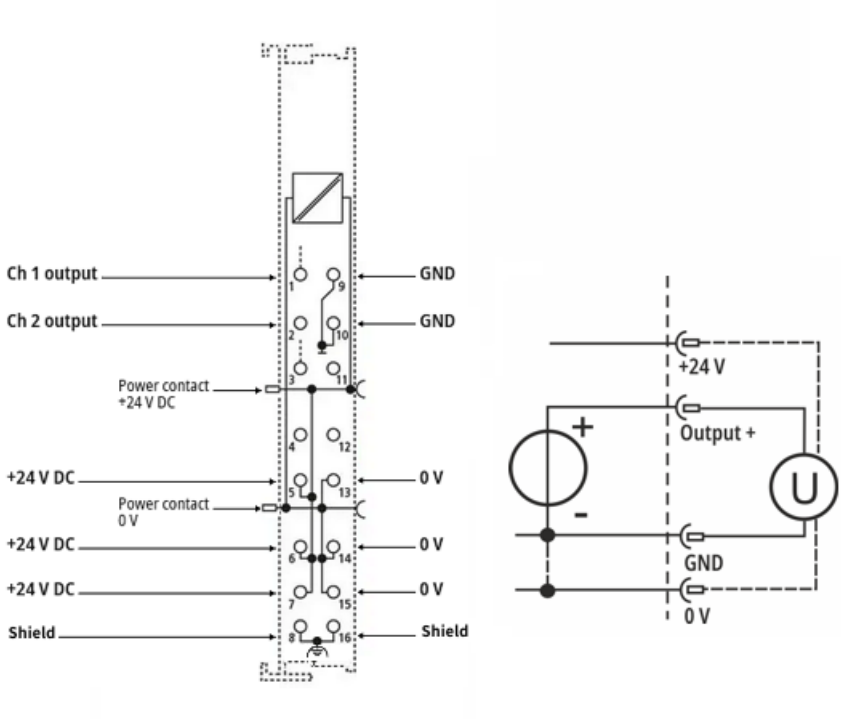


Fig. 6: Connection example EL4072, voltage output, 2-/4-wire

- 2-wire: 2-pin voltage output; 4-wire: with additional 24 V power supply to the actuator

Current output -20 / 0 / +4…+20 mA

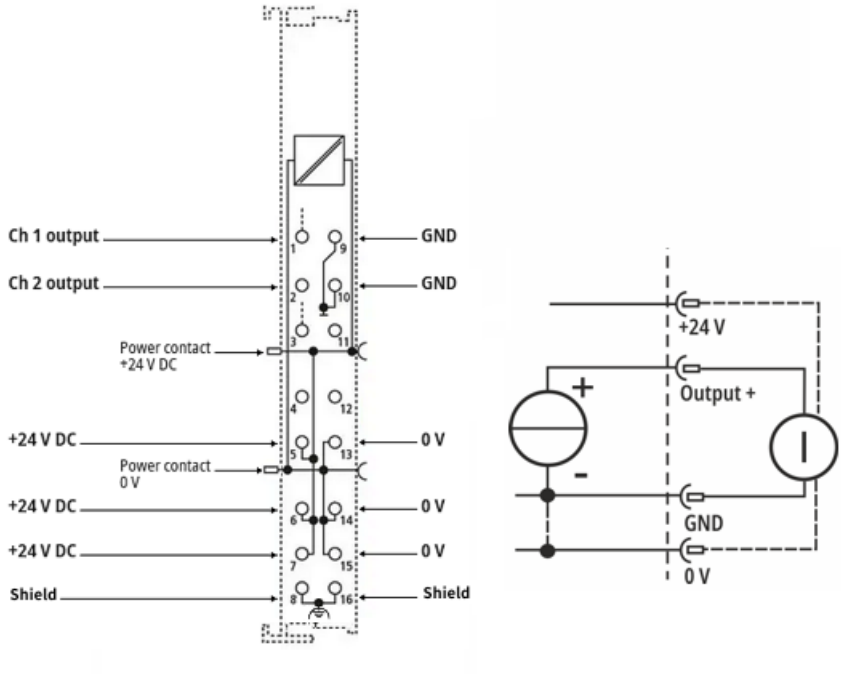


Fig. 7: Connection example EL4072, current output, 2-/4-wire

- 2-wire: 2-pin current output; 4-wire: with additional 24 V power supply to the actuator

2.1.3.2 EL4074

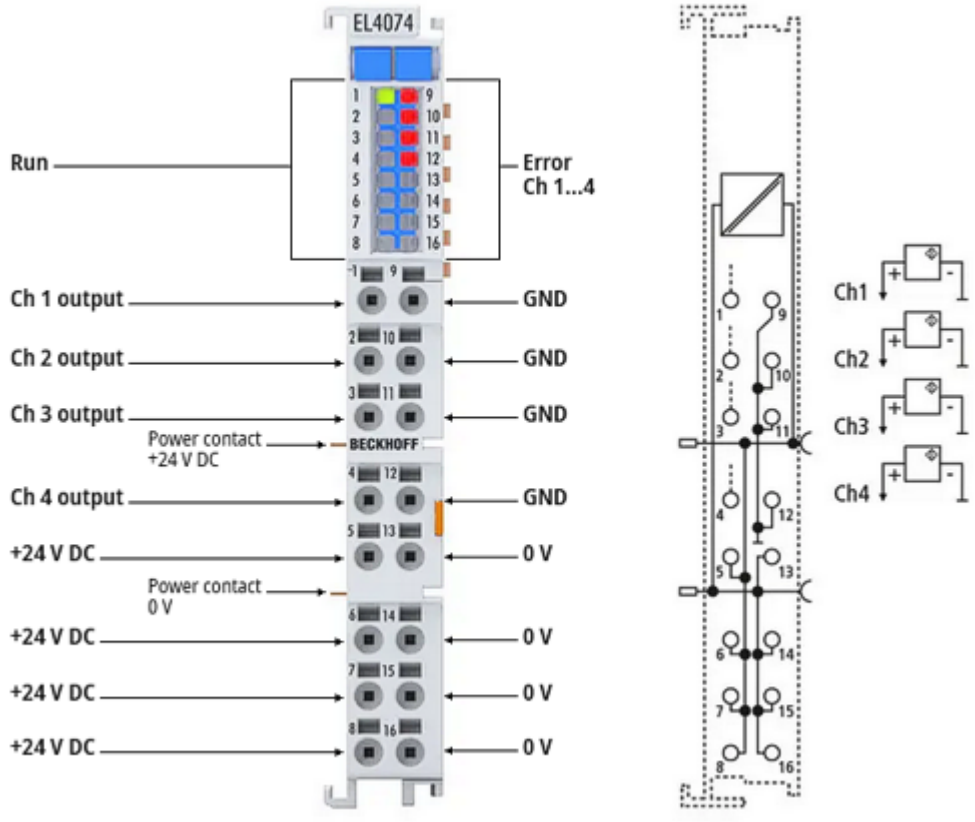


Fig. 8: EL4074 LED and connection

LED	Color	Meaning	
RUN	green	These LEDs indicate the terminal's operating state:	
		off	State of the EtherCAT State Machine [▶ 108]: **INIT** = initialization of the terminal
		flashing	State of the EtherCAT State Machine: **PREOP** = function for mailbox communication and different default settings set
		Single flash	State of the EtherCAT State Machine: **SAFEOP** = verification of the Sync Manager [▶ 108] channels and the distributed clocks. Outputs remain in safe state
		on	State of the EtherCAT State Machine: **OP** = normal operating state; mailbox and process data communication is possible
		flickering	State of the EtherCAT State Machine: **BOOTSTRAP** = function for Firmware updates [▶ 192] of the terminal
ERROR Ch1…4	red	- Analog output overload: wire break at I output, short circuit at U output - Setpoint exceeds Limiter: below low limit /above high limit - Internal error DAC	

NOTICE

Cable lengths > 30 m

For longer cable lengths > 30 m, suitable overvoltage protection must be provided (e.g. EL9540-0010) if corresponding interference could affect the signal cable.

EL4074 connection

Terminal point	No.	Description	Internally connected with connection	Max. current carrying capacity *)
Ch1 output	1	Voltage/current output, channel 1	-	overload protected
Ch2 output	2	Voltage/current output, channel 2	-	overload protected
Ch3 output	2	Voltage/current output, channel 3	-	overload protected
Ch4 output	2	Voltage/current output, channel 4	-	overload protected
+24 V DC	5	+24 V	6; 7, 8 +24 V power contact	1 A
+24 V DC	6	+24 V	5, 7, 8 +24 V power contact	1 A
+24 V DC	7	+24 V	5, 6, 8 +24 V power contact	1 A
+24 V DC	8	+24 V	5, 6, 7, +24 V power contact	1 A
GND	9	Analog ground (reference potential for Ch1...Ch4)	10, 11, 12	100 mA **)
GND	10	Analog ground (reference potential for Ch1...Ch4)	9, 11, 12	100 mA **)
GND	11	Analog ground (reference potential for Ch1...Ch4)	9, 10, 12	100 mA **)
GND	12	Analog ground (reference potential for Ch1...Ch4)	9, 10, 11	100 mA **)
0 V	13	0 V	14, 15, 16, 0 V power contact	1 A
0 V	14	0 V	13, 15, 16, 0 V power contact	1 A
0 V	15	0 V	13, 14, 16, 0 V power contact	1 A
0 V	16	0 V	13, 14, 15, 0 V power contact	1 A

*) Constant current; short-term higher currents are to be avoided and can cause thermal overload (damage)

**) The "GND" potential is internally coupled to the "0V" potential, but is subject to a reduced current carrying capacity

Voltage output 0...10 V / -10...+10 V

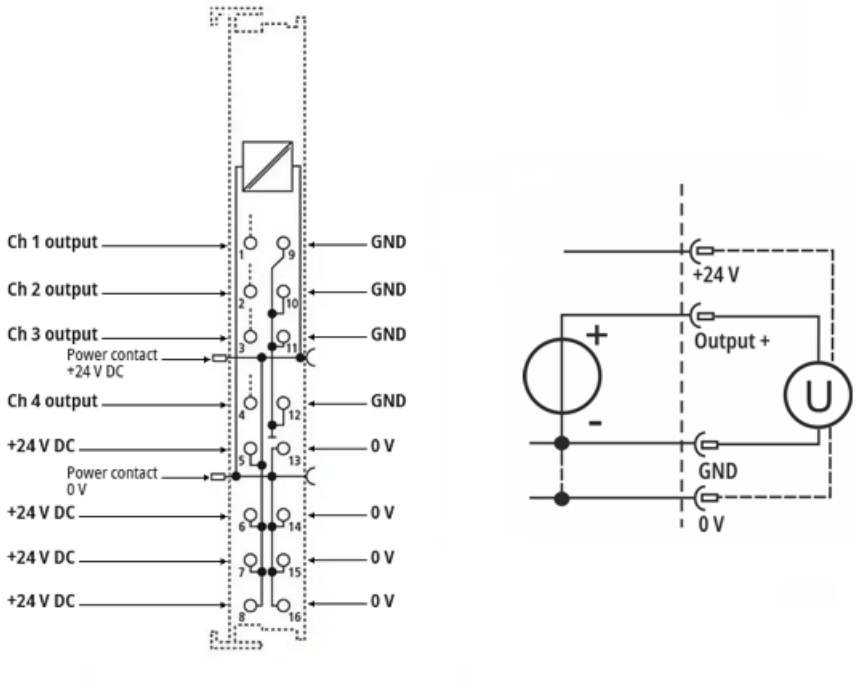


Fig. 9: Connection example EL4074, voltage output, 2-/4-wire

- 2-wire: 2-pin voltage output; 4-wire: with additional 24 V power supply to the actuator

Current output -20 / 0 / +4…+20 mA

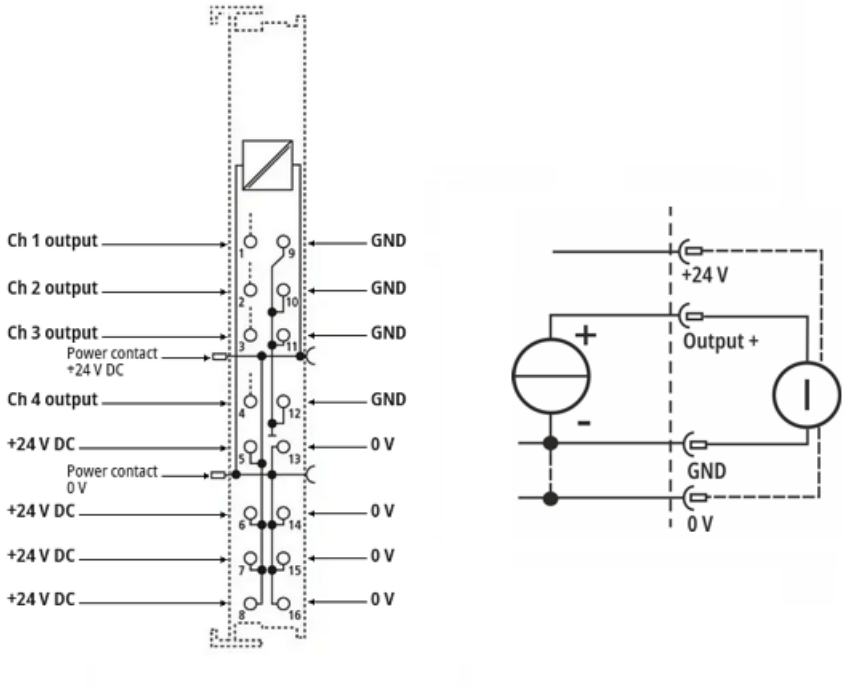


Fig. 10: Connection example EL4074, current output, 2-/4-wire

- 2-wire: 2-pin current output; 4-wire: with additional 24 V power supply to the actuator

2.1.3.3 EL4078

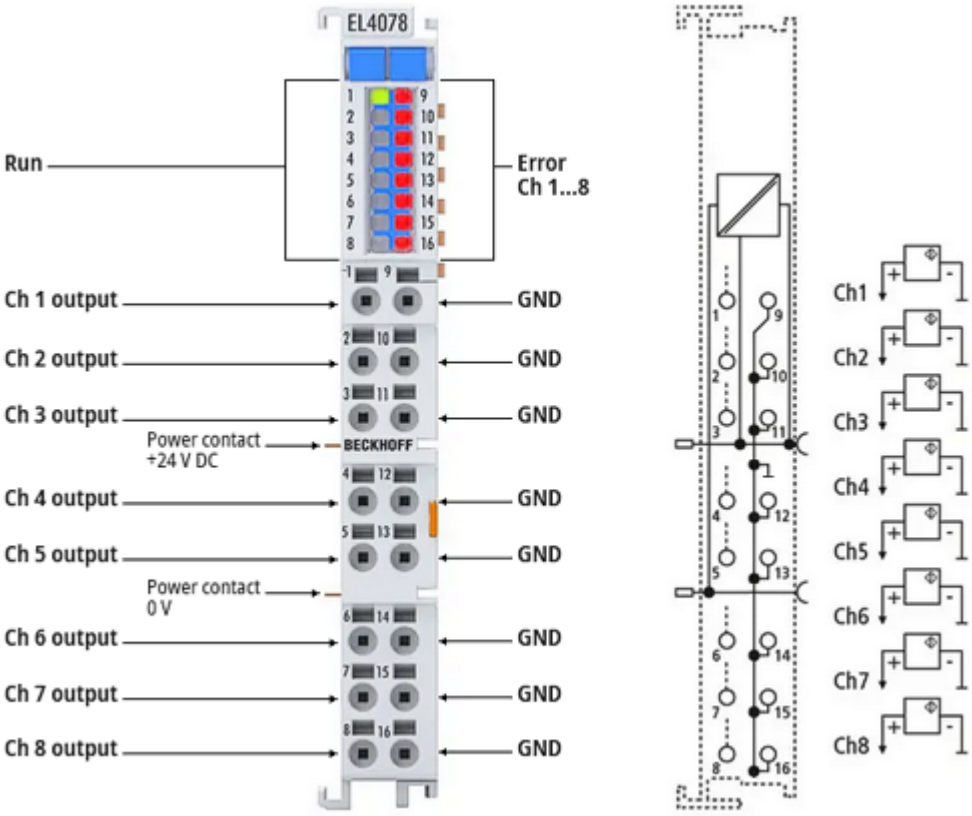


Fig. 11: EL4078 LED and connection

LED	Color	Meaning	
RUN	green	These LEDs indicate the terminal's operating state:	
		off	State of the EtherCAT State Machine [▶ 108]: **INIT** = initialization of the terminal
		flashing	State of the EtherCAT State Machine: **PREOP** = function for mailbox communication and different default settings set
		Single flash	State of the EtherCAT State Machine: **SAFEOP** = verification of the Sync Manager [▶ 108] channels and the distributed clocks. Outputs remain in safe state
		on	State of the EtherCAT State Machine: **OP** = normal operating state; mailbox and process data communication is possible
		flickering	State of the EtherCAT State Machine: **BOOTSTRAP** = function for Firmware updates [▶ 192] of the terminal
ERROR Ch1…8	red	- Analog output overload: wire break at I output, short circuit at U output - Setpoint exceeds Limiter: below low limit /above high limit - Internal error DAC	

NOTICE

Cable lengths > 30 m

For longer cable lengths > 30 m, suitable overvoltage protection must be provided (e.g. EL9540-0010) if corresponding interference could affect the signal cable.

EL4078 connection

Terminal point	No.	Description	Internally connected with connection	Max. current carrying capacity *)
Ch1 output	1	Voltage/current output, channel 1	-	overload protected
Ch2 output	2	Voltage/current output, channel 2	-	overload protected
Ch3 output	2	Voltage/current output, channel 3	-	overload protected
Ch4 output	2	Voltage/current output, channel 4	-	overload protected
Ch5 output	5	Voltage/current output, channel 5	-	overload protected
Ch6 output	6	Voltage/current output, channel 6	-	overload protected
Ch7 output	7	Voltage/current output, channel 7	-	overload protected
Ch8 output	8	Voltage/current output, channel 8	-	overload protected
GND	9	Analog ground (reference potential for Ch1...Ch8)	10, 11, 12, 13, 14, 15, 16	100 mA **)
GND	10	Analog ground (reference potential for Ch1...Ch8)	9, 11, 12, 13, 14, 15, 16	100 mA **)
GND	11	Analog ground (reference potential for Ch1...Ch8)	9, 10, 12, 13, 14, 15, 16	100 mA **)
GND	12	Analog ground (reference potential for Ch1...Ch8)	9, 10, 11, 13, 14, 15, 16	100 mA **)
GND	13	Analog ground (reference potential for Ch1...Ch8)	9, 10, 11, 12, 14, 15, 16	100 mA **)
GND	14	Analog ground (reference potential for Ch1...Ch8)	9, 10, 11, 12, 13, 15, 16	100 mA **)
GND	15	Analog ground (reference potential for Ch1...Ch8)	9, 10, 11, 12, 13, 14, 16	100 mA **)
GND	16	Analog ground (reference potential for Ch1...Ch8)	9, 10, 11, 12, 13, 14, 15	100 mA **)

*) Constant current; short-term higher currents are to be avoided and can cause thermal overload (damage)

**) The "GND" potential is internally coupled to the "0V" potential, but is subject to a reduced current carrying capacity

Voltage output 0...10 V / -10...+10 V

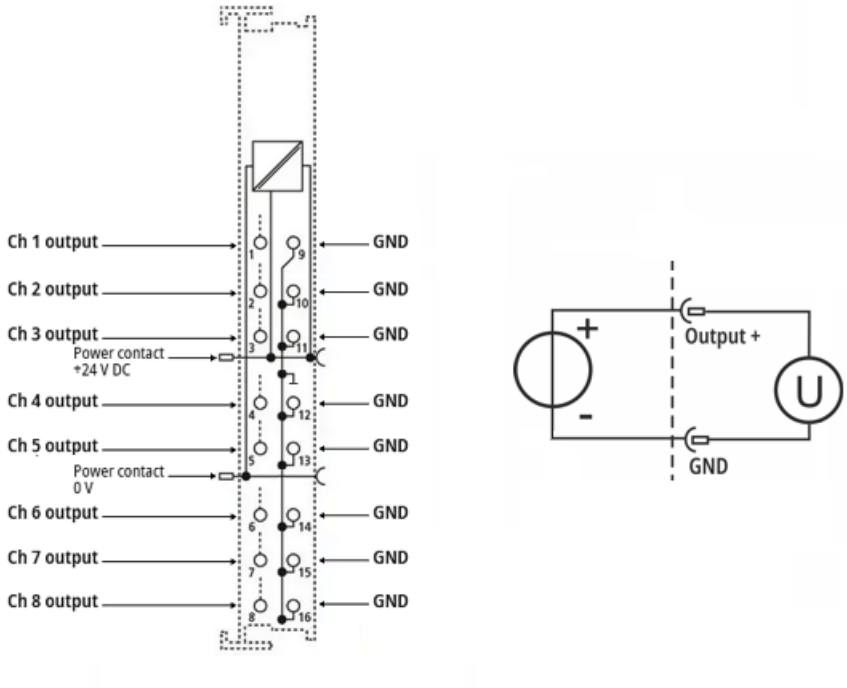


Fig. 12: Connection example EL4078, voltage output, 2-wire

- 2-wire: 2-pin voltage output

Current output -20 / 0 / +4…+20 mA

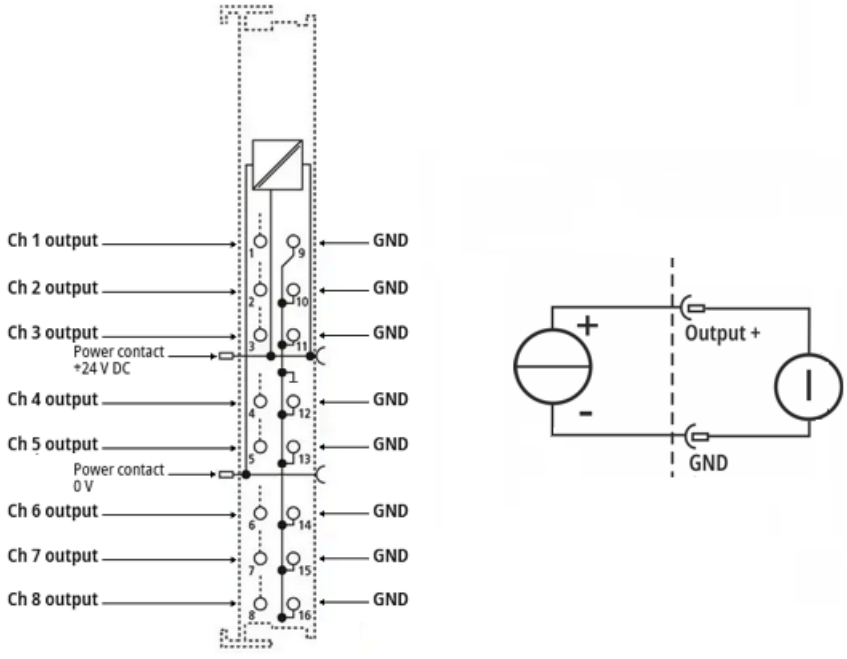


Fig. 13: Connection example EL4078, current output, 2-wire

- 2-wire: 2-pin current output

2.2 Further documentation for I/O components with analog in and outputs

NOTICE	
	**Further documentation for I/O components with analog in and outputs** Also pay attention to the further documentation: **I/O Analog Manual** Notes on I/O components with analog inputs and outputs, which is available in the Beckhoff Information-System and for download on the Beckhoff website www.beckhoff.com on the respective product pages! The content includes the basics of sensor technology and information on analog measured values.

2.3 Start

For commissioning

- mount the EL407x as described in the chapter Mounting and wiring [▶ 41].
- configure the EL407x in TwinCAT as described in the chapter Commissioning [▶ 55].

3 Basics communication

3.1 EtherCAT basics

Please refer to the EtherCAT System Documentation for the EtherCAT fieldbus basics.

3.2 EtherCAT cabling – wire-bound

The cable length between two EtherCAT devices must not exceed 100 m. This results from the FastEthernet technology, which, above all for reasons of signal attenuation over the length of the cable, allows a maximum link length of 5 + 90 + 5 m if cables with appropriate properties are used. See also the Design recommendations for the infrastructure for EtherCAT/Ethernet.

Cables and connectors

For connecting EtherCAT devices only Ethernet connections (cables + plugs) that meet the requirements of at least category 5 (CAt5) according to EN 50173 or ISO/IEC 11801 should be used. EtherCAT uses 4 wires for signal transfer.

EtherCAT uses RJ45 plug connectors, for example. The pin assignment is compatible with the Ethernet standard (ISO/IEC 8802-3).

Pin	Color of conductor	Signal	Description
1	yellow	TD +	Transmission Data +
2	orange	TD -	Transmission Data -
3	white	RD +	Receiver Data +
6	blue	RD -	Receiver Data -

Due to automatic cable detection (auto-crossing) symmetric (1:1) or cross-over cables can be used between EtherCAT devices from Beckhoff.

Recommended cables

It is recommended to use the appropriate Beckhoff components e.g.
- cable sets ZK1090-9191-xxxx respectively
- RJ45 connector, field assembly ZS1090-0005
- EtherCAT cable, field assembly ZB9010, ZB9020

Suitable cables for the connection of EtherCAT devices can be found on the Beckhoff website!

E-Bus supply

A bus coupler can supply the EL terminals added to it with the E-bus system voltage of 5 V; a coupler is thereby loadable up to 2 A as a rule (see details in respective device documentation).
Information on how much current each EL terminal requires from the E-bus supply is available online and in the catalogue. If the added terminals require more current than the coupler can supply, then power feed terminals (e.g. EL9410) must be inserted at appropriate places in the terminal strand.

The pre-calculated theoretical maximum E-Bus current is displayed in the TwinCAT System Manager. A shortfall is marked by a negative total amount and an exclamation mark; a power feed terminal is to be placed before such a position.

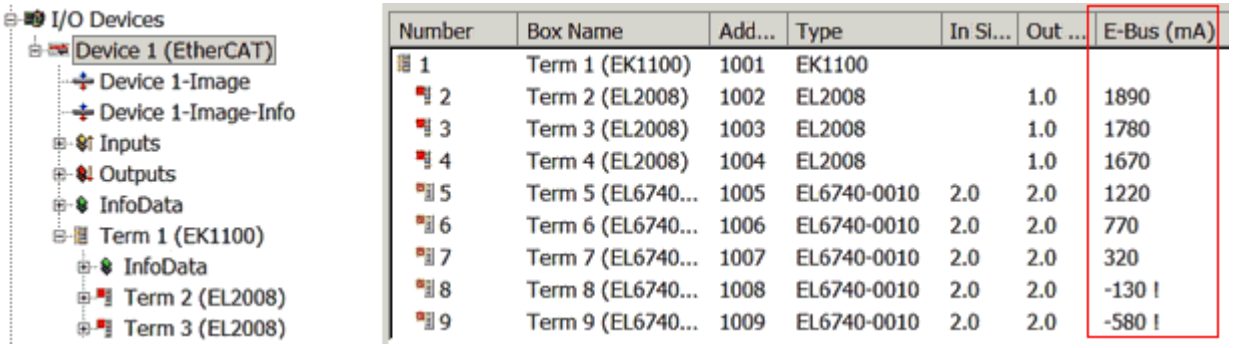


Fig. 14: System manager current calculation

NOTICE

Malfunction possible!

The same ground potential must be used for the E-Bus supply of all EtherCAT terminals in a terminal block!

3.3 General notes for setting the watchdog

The EtherCAT terminals are equipped with a safety device (watchdog) which, e. g. in the event of interrupted process data traffic, switches the outputs (if present) to a presettable state after a presettable time, depending on the device and setting, e. g. to FALSE (off) or an output value.

The EtherCAT slave controller features two watchdogs:

- Sync Manager (SM) watchdog (default: 100 ms)
- Process Data (PDI) watchdog (default: 100 ms)

Their times are individually parameterized in TwinCAT as follows:

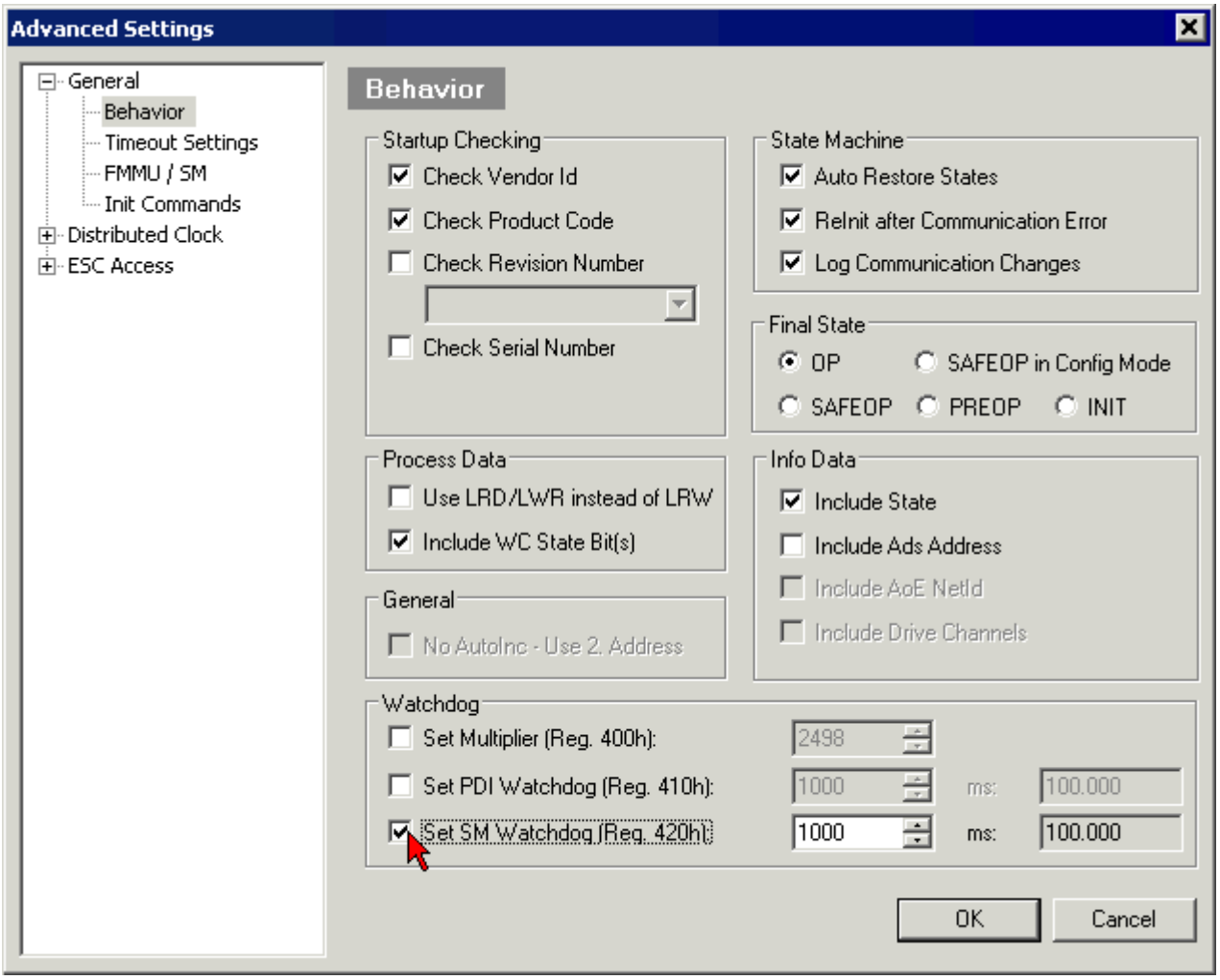


Fig. 15: eEtherCAT tab -> Advanced Settings -> Behavior -> Watchdog

Notes:

- the Multiplier Register 400h (hexadecimal, i. e. 0x0400) is valid for both watchdogs.
- each watchdog has its own timer setting 410h or 420h, which together with the Multiplier results in a resulting time.
- important: the Multiplier/Timer setting is only loaded into the slave at EtherCAT startup if the checkbox in front of it is activated.
- if it is not checked, nothing is downloaded and the setting located in the ESC remains unchanged.
- the downloaded values can be seen in the ESC registers 400h, 410h and 420h: ESC Access -> Memory

SM watchdog (SyncManager Watchdog)

The SyncManager watchdog is reset with each successful EtherCAT process data communication with the terminal. If, for example, no EtherCAT process data communication with the terminal takes place for longer than the set and activated SM watchdog time due to a line interruption, the watchdog is triggered. The status of the terminal (usually OP) remains unaffected. The watchdog is only reset again by a successful EtherCAT process data access.

The SyncManager watchdog is therefore a monitoring for correct and timely process data communication with the ESC from the EtherCAT side.

The maximum possible watchdog time depends on the device. For example, for "simple" EtherCAT slaves (without firmware) with watchdog execution in the ESC it is usually up to 170 seconds. For complex EtherCAT slaves (with firmware) the SM watchdog function is usually parameterized via register 400h/420h but executed by the microcontroller (µC) and can be significantly lower. In addition, the execution may then be subject to a certain time uncertainty. Since the TwinCAT dialog may allow inputs up to 65535, a test of the desired watchdog time is recommended.

PDI watchdog (Process Data Watchdog)

If there is no PDI communication with the ESC for longer than the set and activated Process Data Interface (PDI) watchdog time, this watchdog is triggered.

The PDI is the internal interface of the ESC, e.g. to local processors in the EtherCAT slave. With the PDI watchdog this communication can be monitored for failure.

The PDI watchdog is therefore a monitoring for correct and timely process data communication with the ESC, but viewed from the application side.

Calculation

Watchdog time = [1/25 MHz * (Watchdog multiplier + 2)] * SM/PDI watchdog

Example: default setting Multiplier = 2498, SM watchdog = 1000 => 100 ms

The value in “Watchdog multiplier + 2” in the formula above corresponds to the number of 40ns base ticks representing one watchdog tick.

⚠ CAUTION

Undefined state possible!

The function for switching off the SM watchdog via SM watchdog = 0 is only implemented in terminals from revision -0016. In previous versions this operating mode should not be used.

⚠ CAUTION

Damage of devices and undefined state possible!

If the SM watchdog is activated and a value of 0 is entered the watchdog switches off completely. This is the deactivation of the watchdog! Set outputs are NOT set in a safe state if the communication is interrupted.

3.4 EtherCAT State Machine

The state of the EtherCAT slave is controlled via the EtherCAT State Machine (ESM). Depending upon the state, different functions are accessible or executable in the EtherCAT slave. Specific commands must be sent by the EtherCAT master to the device in each state, particularly during the bootup of the slave.

A distinction is made between the following states:

- Init
- Pre-Operational
- Safe-Operational
- Operational
- Bootstrap

The regular state of each EtherCAT slave after bootup is the OP state.

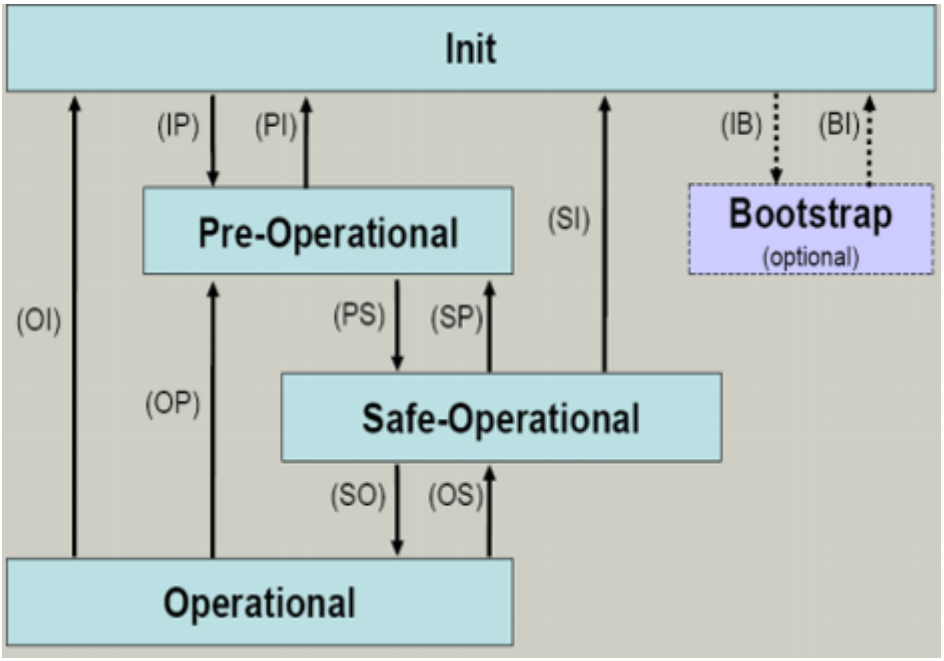


Fig. 16: States of the EtherCAT State Machine

Init

After switch-on the EtherCAT slave in the *Init* state. No mailbox or process data communication is possible. The EtherCAT master initializes sync manager channels 0 and 1 for mailbox communication.

Pre-Operational (Pre-Op)

During the transition between *Init* and *Pre-Op* the EtherCAT slave checks whether the mailbox was initialized correctly.

In *Pre-Op* state mailbox communication is possible, but not process data communication. The EtherCAT master initializes the sync manager channels for process data (from sync manager channel 2), the Fieldbus Memory Management Unit (FMMU) channels and, if the slave supports configurable mapping, PDO mapping or the sync manager PDO assignment. In this state the settings for the process data transfer and perhaps terminal-specific parameters that may differ from the default settings are also transferred.

Safe-Operational (Safe-Op)

During transition between *Pre-Op* and *Safe-Op* the EtherCAT slave checks whether the sync manager channels for process data communication and, if required, the Distributed Clocks settings are correct. Before it acknowledges the change of state, the EtherCAT slave copies current input data into the associated Dual Port (DP)-RAM areas of the ESC.

In *Safe-Op* state mailbox and process data communication is possible, although the slave keeps its outputs in a safe state, while the input data are updated cyclically.

Outputs in SAFEOP state

The default set watchdog monitoring sets the outputs of the ESC module in a safe state - depending on the settings in SAFEOP and OP - e.g. in OFF state. If this is prevented by deactivation of the monitoring in the module, the outputs can be switched or set also in the SAFEOP state.

Operational (Op)

Before the EtherCAT master switches the EtherCAT slave from *Safe-Op* to *Op* it must transfer valid output data.

In the *Op* state the slave copies the output data of the masters to its outputs. Process data and mailbox communication is possible.

Boot

In the *Boot* state the slave firmware can be updated. The *Boot* state can only be reached via the *Init* state.

In the *Boot* state mailbox communication via the file access over EtherCAT (FoE) protocol is possible, but no other mailbox communication and no process data communication.

3.5 CoE Interface

General description

The CoE interface (CAN application protocol over EtherCAT interface) is used for parameter management of EtherCAT devices. EtherCAT slaves or the EtherCAT master manage fixed (read only) or variable parameters which they require for operation, diagnostics or commissioning.

CoE parameters are arranged in a table hierarchy. In principle, the user has access via the fieldbus. The EtherCAT master (TwinCAT System Manager) can access the local CoE lists of the slaves via EtherCAT in read or write mode, depending on the attributes.

Different CoE data types are possible, including string (text), integer numbers, Boolean values or larger byte fields. They can be used to describe a wide range of features. Examples of such parameters include manufacturer ID, serial number, process data settings, device name, calibration values for analog measurement or passwords.

The order is specified in two levels via hexadecimal numbering: (main)index, followed by subindex.

The value ranges are

- Index: 0x0000 …0xFFFF ($0...65535_{dec}$)
- Subindex: 0x00…0xFF ($0...255_{dec}$)

A parameter localized in this way is normally written as 0x8010:07, with preceding “0x” to identify the hexadecimal numerical range and a colon between index and subindex.

The relevant ranges for EtherCAT fieldbus users are:

- 0x1000: This is where fixed identity information for the device is stored, including name, manufacturer, serial number etc., plus information about the current and available process data configurations.
- 0x8000: This is where the operational and functional parameters for all channels are stored, such as filter settings or output frequency.

Other important ranges are:

- 0x4000: here are the channel parameters for some EtherCAT devices. Historically, this was the first parameter area before the 0x8000 area was introduced. EtherCAT devices that were previously equipped with parameters in 0x4000 and changed to 0x8000 support both ranges for compatibility reasons and mirror internally.
- 0x6000: Input PDOs (“inputs” from the perspective of the EtherCAT master)
- 0x7000: Output PDOs (“outputs” from the perspective of the EtherCAT master)

Availability

Not every EtherCAT device must have a CoE list. Simple I/O modules without dedicated processor usually have no variable parameters and therefore no CoE list.

If a device has a CoE list, it is shown in the TwinCAT System Manager as a separate tab with a listing of the elements:

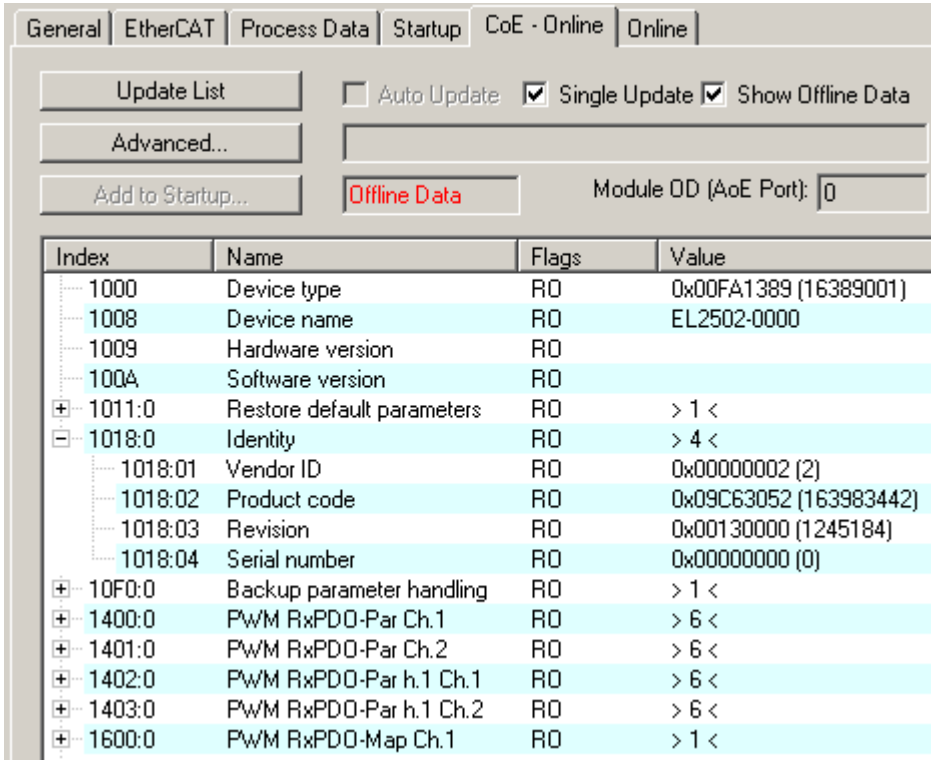


Fig. 17: “CoE Online” tab

The figure “’CoE Online’ tab” shows the CoE objects available in device “EL2502”, ranging from 0x1000 to 0x1600. The subindices for 0x1018 are expanded.

NOTICE

Changes in the CoE directory (CAN over EtherCAT directory), program access

When using/manipulating the CoE parameters observe the general CoE notes in chapter "CoE interface" of the EtherCAT system documentation:

- Keep a startup list if components have to be replaced,
- Distinction between online/offline dictionary,
- Existence of current XML description (download from the Beckhoff website),
- "CoE-Reload" for resetting the changes
- Program access during operation via PLC (see TwinCAT 3 | PLC Library: "Tc2_EtherCAT" and Example program R/W CoE)

Data management and function “NoCoeStorage”

Some parameters, particularly the setting parameters of the slave, are configurable and writeable,

- via the System Manager (Fig. “CoE Online” tab) by clicking.
  This is useful for commissioning of the system or slaves. Click on the row of the index to be parameterized and enter a value in the “SetValue” dialog.
- from the control system or PLC via ADS, e.g. through blocks from the TcEtherCAT.lib library.
  This is recommended for modifications while the system is running or if no System Manager or operating staff are available.

Data management

If slave CoE parameters are modified online, Beckhoff devices store any changes in a fail-safe manner in the EEPROM, i.e. the modified CoE parameters are still available after a restart.
The situation may be different with other manufacturers.

An EEPROM is subject to a limited lifetime with respect to write operations. From typically 100,000 write operations onwards it can no longer be guaranteed that new (changed) data are reliably saved or are still readable. This is irrelevant for normal commissioning. However, if CoE parameters are continuously changed via ADS at machine runtime, it is quite possible for the lifetime limit to be reached. Support for the NoCoeStorage function, which suppresses the saving of changed CoE values, depends on the firmware version.
Please refer to the technical data in this documentation as to whether this applies to the respective device.

- If the function is supported: the function is activated by entering the code word 0x12345678 once in CoE index 0xF008 and remains active as long as the code word is not changed. After switching the device on it is then inactive. Changed CoE values are not saved in the EEPROM and can thus be changed any number of times.
- If the function is not supported: continuous changing of CoE values is not permissible in view of the lifetime limit.

Startup list

Changes in the local CoE list of the terminal are lost if the terminal is replaced. If a terminal is replaced with a new Beckhoff terminal, it will have the default settings. It is therefore advisable to link all changes in the CoE list of an EtherCAT slave with the Startup list of the slave, which is processed whenever the EtherCAT fieldbus is started. In this way a replacement EtherCAT slave can automatically be parameterized with the specifications of the user.

If EtherCAT slaves are used which are unable to store local CoE values permanently, the Startup list must be used.

Recommended approach for manual modification of CoE parameters

- Make the required change in the System Manager
  (the values are stored locally in the EtherCAT slave).
- If the value is to be stored permanently, enter it in the Startup list.
  The order of the Startup entries is usually irrelevant.

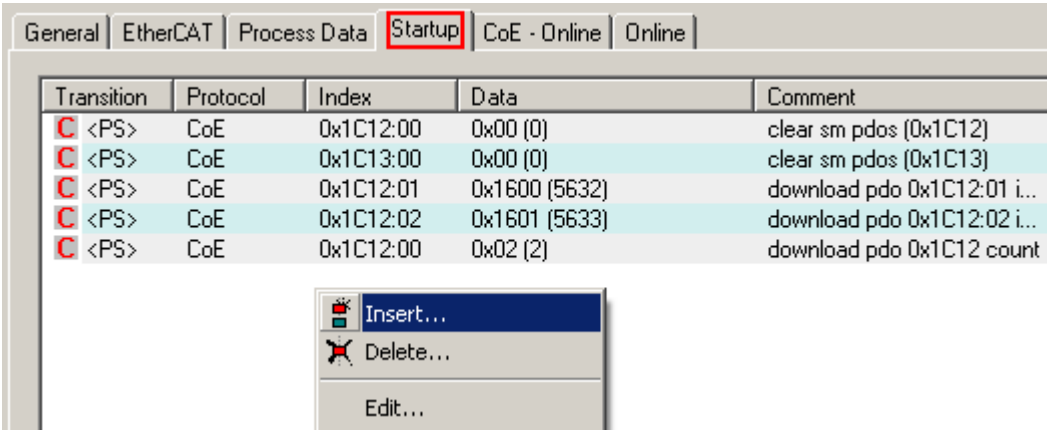


Fig. 18: Startup list in the TwinCAT System Manager

The Startup list may already contain values that were configured by the System Manager based on the ESI specifications. Additional application-specific entries can also be created.

Online / offline list

When working with the TwinCAT System Manager, a distinction must be made as to whether the EtherCAT device is currently "available", i.e. switched on and connected via EtherCAT - i.e. **online** - or whether a configuration is created **offline** without slaves being connected.

In both cases a CoE list as shown in Fig. “CoE online tab” is displayed. The connectivity is shown as offline/ online.

- If the slave is offline:
  - The offline list from the ESI file is displayed. In this case modifications are not meaningful or possible.
  - The configured status is shown under Identity.
  - No firmware or hardware version is displayed since these are features of the physical device.
  - **Offline Data** is shown in red.

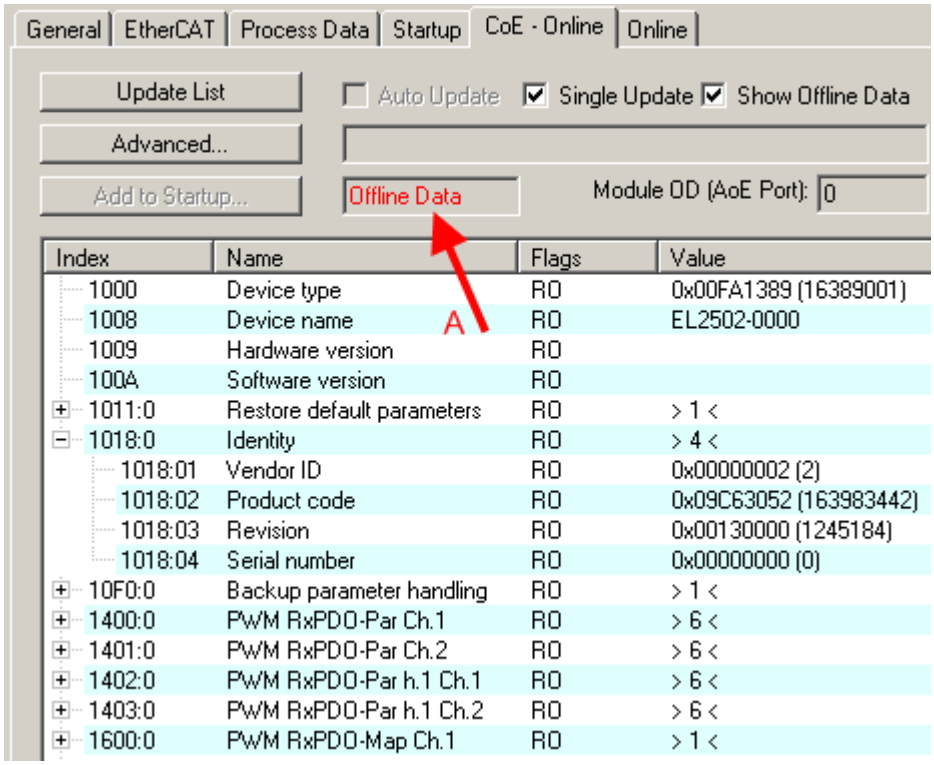


Fig. 19: Offline list

- If the slave is online:
  - The actual current slave list is read. This may take several seconds, depending on the size and cycle time.
  - The actual identity is displayed.
  - The firmware and hardware status of the device is displayed in the CoE.
  - **Online Data** is shown in green.

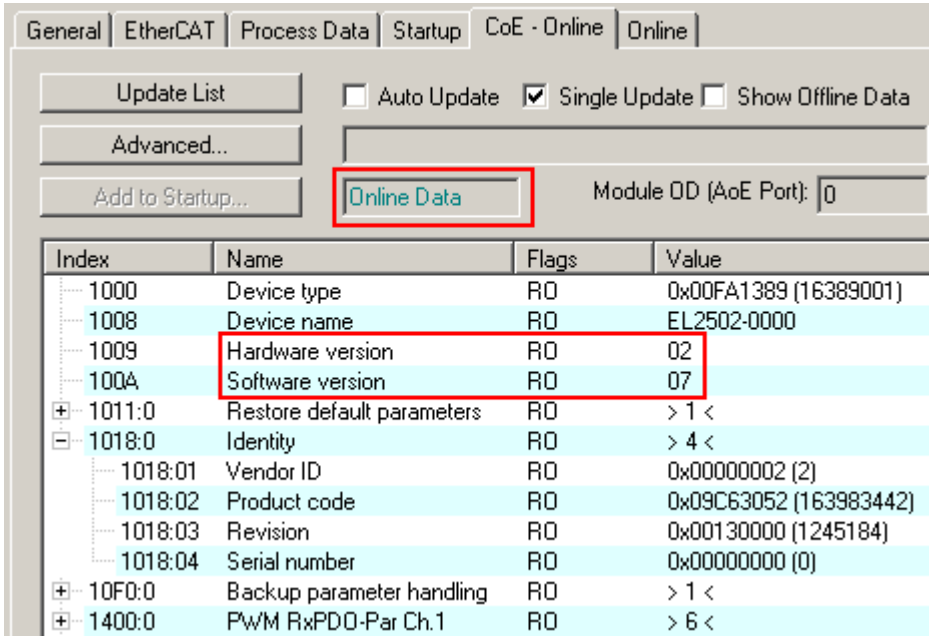


Fig. 20: Online list

Channel-based order

The CoE list is available in EtherCAT devices that usually feature several functionally equivalent channels, for example, a 4-channel analog input terminal also has four logical channels and therefore four identical sets of parameter data for the channels. In order to avoid having to list each channel in the documentation, the placeholder “n” tends to be used for the individual channel numbers.

In the CoE system 16 indices, each with 255 subindices, are generally sufficient for representing all channel parameters. The channel-based order is therefore arranged in 16_{dec} or 10_{hex} steps. The parameter range 0x8000 exemplifies this:

- Channel 0: parameter range 0x8000:00 ... 0x800F:255
- Channel 1: parameter range 0x8010:00 ... 0x801F:255
- Channel 2: parameter range 0x8020:00 ... 0x802F:255
- ...

This is generally written as 0x80n0.

Detailed information on the CoE interface can be found in the EtherCAT system documentation on the Beckhoff website.

BackUp objects and checksum 0x10F0:01

The following object types, among others, are defined for the CoE parameter/object directory with regard to information retention, but they do not all have to appear in every device at the same time:

- Vendor objects
  - Are stored persistently (fail-safe) in the device.
  - Technically with the property ReadWrite (RW).
  - Can only be modified/deleted if the corresponding vendor password is known.
  - Used for vendor-specific adjustment or identity data.
- BackUp objects
  - These are objects that are stored persistently in the device, even after changes.
  - Technically with the property ReadWrite (RW).
  - They can be modified/deleted at any time via CoE access from the EtherCAT master

- In particular, they are reset to their initial state, which is stored permanently in the firmware, by selecting "Restore Default Parameters" (see chapter "Restoring the delivery state"). Since this reset to a previous value resembles the restoration of a backup, these are referred to as “BackUp objects”.
- They are used for regular function parameters of the device that determine the behavior.

- BackUp objects with write protection option.
  - The same applies as for the BackUp objects.
  - In addition, the user can activate write protection for these objects using a code word in xF009, thereby preventing accidental changes. Details can be found in the device documentation of the devices that contain these objects.
- Volatile objects
  - These are objects that are not stored persistently in the device.
  - They are used to display internal information (process data, states, temperatures ...) and are available as ReadOnly (RO) or as function parameters (ReadWrite). However, the latter must be written by the EtherCAT master for each PowerOn if they are to have a value other than the default value.

The device indicates a 16-bit CRC in the 32-bit object 0x10F0:01 CheckSum, subindex 01 of Backup Parameter Handling by means of the current state of the so-called BackUp objects:

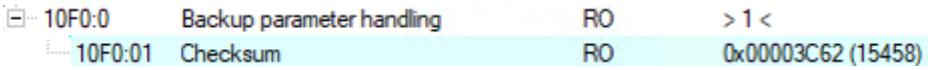


Fig. 21: CoE index 10F0

If a BackUp object is changed, the firmware calculates a new checksum accordingly. This can be used to detect changes to the BackUp objects.

Note: The initial value of the checksum may vary depending on the firmware version, as function extensions can add additional objects to the CRC’s detection range.

4 Mounting and wiring

4.1 Instructions for ESD protection

NOTICE

Destruction of the devices by electrostatic discharge possible!

The devices contain components at risk from electrostatic discharge caused by improper handling.

- When handling the components, ensure that there is no electrostatic discharge; also avoid touching the spring contacts directly (see illustration).
- Contact with highly insulating materials (synthetic fibers, plastic films, etc.) should be avoided when handling components at the same time.
- When handling the components, ensure that the environment (workplace, packaging and persons) is properly earthed.
- Each bus station must be terminated on the right-hand side with the EL9011 or EL9012 end cap to ensure the degree of protection and ESD protection.

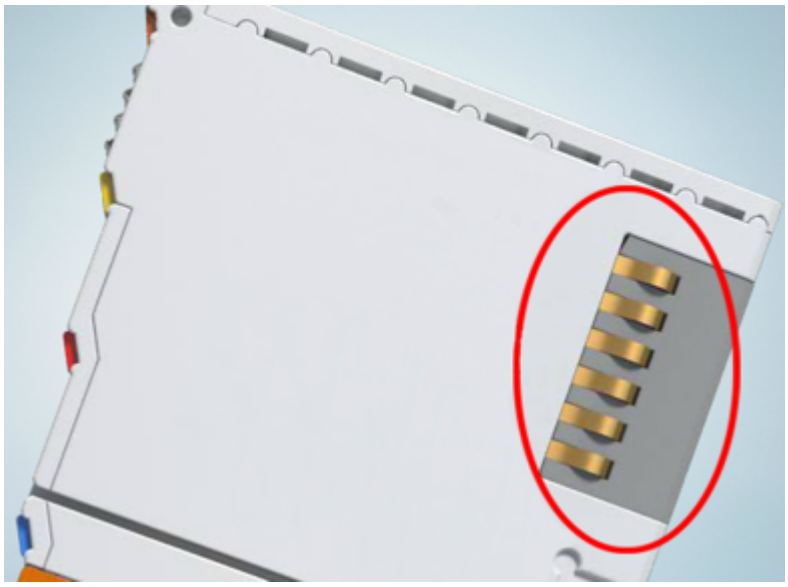


Fig. 22: Spring contacts of the Beckhoff I/O components

4.2 Installation on mounting rails

⚠ WARNING

Risk of electric shock and damage of device!

Bring the bus terminal system into a safe, powered down state before starting installation, disassembly or wiring of the bus terminals!

The Bus Terminal system and is designed for mounting in a control cabinet or terminal box.

Assembly

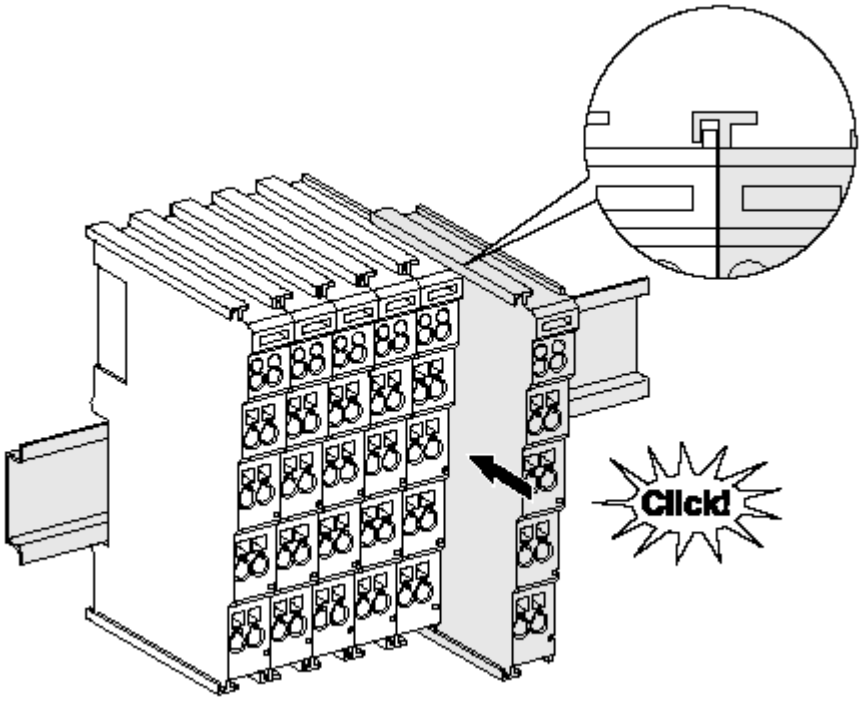


Fig. 23: Attaching on mounting rail

The bus coupler and bus terminals are attached to commercially available 35 mm mounting rails (DIN rails according to EN 60715) by applying slight pressure:

1. First attach the fieldbus coupler to the mounting rail.
2. The bus terminals are now attached on the right-hand side of the fieldbus coupler. Join the components with tongue and groove and push the terminals against the mounting rail, until the lock clicks onto the mounting rail.
   If the terminals are clipped onto the mounting rail first and then pushed together without tongue and groove, the connection will not be operational! When correctly assembled, no significant gap should be visible between the housings.

Fixing of mounting rails

The locking mechanism of the terminals and couplers extends to the profile of the mounting rail. At the installation, the locking mechanism of the components must not come into conflict with the fixing bolts of the mounting rail. To mount the mounting rails with a height of 7.5 mm under the terminals and couplers, you should use flat mounting connections (e.g. countersunk screws or blind rivets).

NOTICE

Ground the mounting rail!

Ensure that the mounting rail is sufficiently grounded.

Connections within a bus terminal block

The electric connections between the Bus Coupler and the Bus Terminals are automatically realized by joining the components:

- The six spring contacts of the E-Bus/K-Bus deal with the transfer of the data and the supply of the Bus Terminal electronics.
- The power contacts deal with the supply for the field electronics and thus represent a supply rail within the bus terminal block. The power contacts are supplied via terminals points on the Bus Coupler (up to 24 V) or for higher voltages via power feed terminals.

Power Contacts

During the design of a bus terminal block, the pin assignment of the individual Bus Terminals must be taken account of, since some types (e.g. analog Bus Terminals or digital 4-channel Bus Terminals) do not or not fully loop through the power contacts. Power Feed Terminals (EL91xx, EL92xx or KL91xx, KL92xx) interrupt the power contacts and thus represent the start of a new supply rail.

Power contact ⏚

The power contact labeled ⏚ (earthing connection according to IEC 60417-5017, British English: earth, American English: ground) can be used as grounding. For safety reasons this contact mates first when plugging together, and can ground short-circuit currents of up to 125 A.

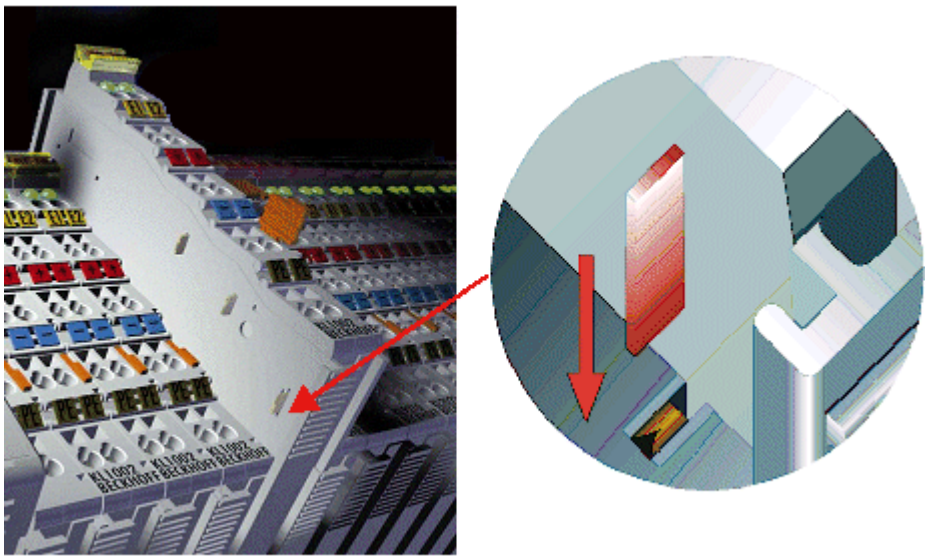


Fig. 24: Power contact on left side

⚠ WARNING

Risk of electric shock!

The power contact labeled ⏚ must not be used for other potentials!

NOTICE

Possible damage of the device

Note that, for reasons of electromagnetic compatibility, the earthing contacts are capacitatively coupled to the mounting rail. This may lead to incorrect results during insulation testing or to damage on the terminal (e.g. disruptive discharge to the earthing line during insulation testing of a consumer with a nominal voltage of 230 V). For insulation testing, disconnect the earthing supply line at the Bus Coupler or the Power Feed Terminal! In order to decouple further feed points for testing, these Power Feed Terminals can be released and pulled at least 10 mm from the group of terminals.

Disassembly

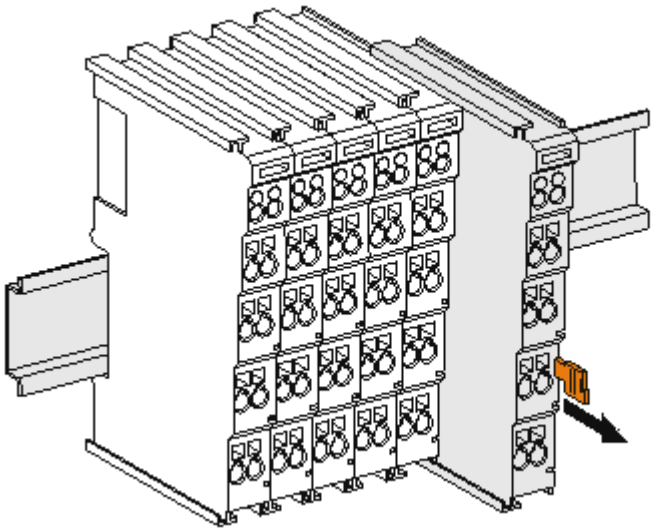


Fig. 25: Disassembling of terminal

Each terminal is secured by a lock on the mounting rail, which must be released for disassembly:

1. Pull the terminal by its orange-colored lugs approximately 1 cm away from the mounting rail. In doing so for this terminal the mounting rail lock is released automatically and you can pull the terminal out of the bus terminal block easily without excessive force.
2. Grasp the released terminal with thumb and index finger simultaneous at the upper and lower grooved housing surfaces and pull the terminal out of the bus terminal block.

4.3 Positioning of passive Terminals

Hint for positioning of passive terminals in the bus terminal block

EtherCAT Terminals, which do not take an active part in data transfer within the bus terminal block are so called passive terminals. The passive terminals have no current consumption out of the E-Bus.
To ensure an optimal data transfer, you must not directly string together more than two passive terminals!

Examples for positioning of passive terminals (highlighted)

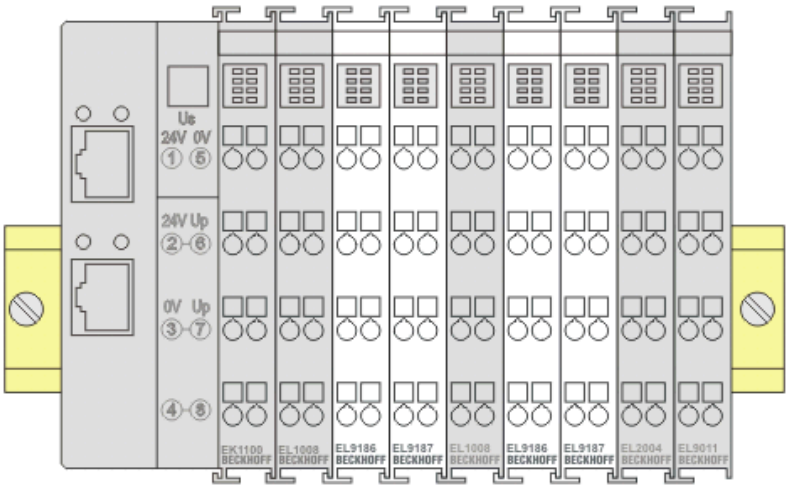


Fig. 26: Correct positioning

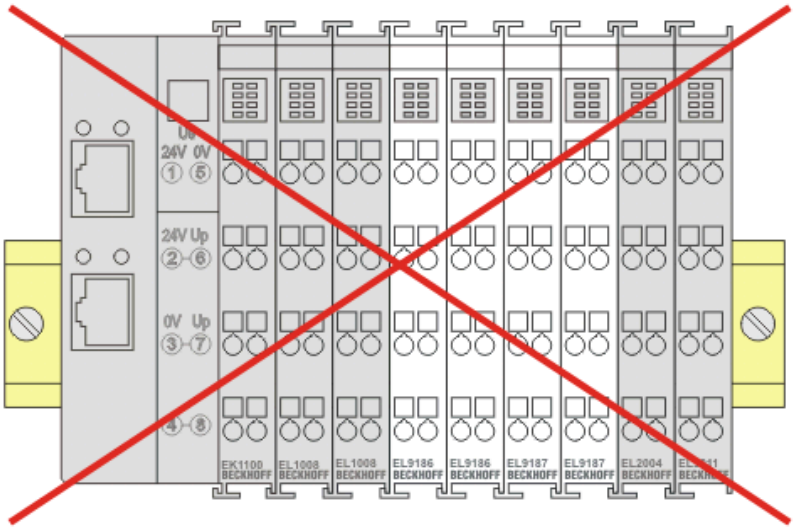


Fig. 27: Incorrect positioning

4.4 Note - power supply

⚠ WARNING

Power supply from SELV / PELV power supply unit!

SELV / PELV circuits (safety extra-low voltage / protective extra-low voltage) according to IEC 61010-2-201 must be used to supply this device.

Notes:

- SELV / PELV circuits may give rise to further requirements from standards such as IEC 60204-1 et al, for example with regard to cable spacing and insulation.
- A SELV supply provides safe electrical isolation and limitation of the voltage without a connection to the protective conductor, a PELV supply also requires a safe connection to the protective conductor.

4.5 Connection

4.5.1 Connection system

⚠ WARNING

Risk of electric shock and damage of device!

Bring the bus terminal system into a safe, powered down state before starting installation, disassembly or wiring of the bus terminals!

Overview

The bus terminal system offers different connection options for optimum adaptation to the respective application:

- The terminals of ELxxxx and KLxxxx series with standard wiring include electronics and connection level in a single enclosure.
- The terminals of ESxxxx and KSxxxx series feature a pluggable connection level and enable steady wiring while replacing.
- The High Density Terminals (HD Terminals) include electronics and connection level in a single enclosure and have advanced packaging density.

Standard wiring (ELxxxx / KLxxxx)

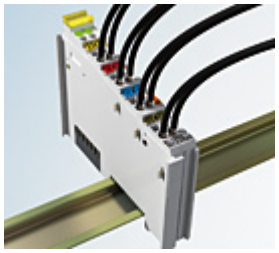


Fig. 28: Standard wiring

The terminals of the ELxxxx and KLxxxx series integrate screwless spring-cage technology for quick and easy wiring.

Pluggable wiring (ESxxxx / KSxxxx)

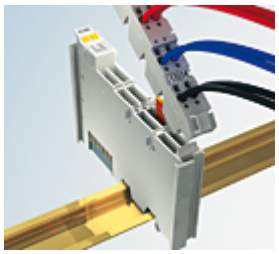


Fig. 29: Pluggable wiring

The terminals of ESxxxx and KSxxxx series feature a pluggable connection level.
The assembly and wiring procedure is the same as for the ELxxxx and KLxxxx series.
The pluggable connection level enables the complete wiring to be removed as a plug connector from the top of the housing for servicing.
The lower section can be removed from the terminal block by pulling the unlocking tab.
Insert the new component and plug in the connector with the wiring. This reduces the installation time and eliminates the risk of wires being mixed up.

The familiar dimensions of the terminal only had to be changed slightly. The new connector adds about 3 mm. The maximum height of the terminal remains unchanged.

A tab for strain relief of the cable simplifies assembly in many applications and prevents tangling of individual connection wires when the connector is removed.

Conductor cross sections between 0.08 mm^2 and 2.5 mm^2 can continue to be used with the proven spring force technology.

The overview and nomenclature of the product names for ESxxxx and KSxxxx series has been retained as known from ELxxxx and KLxxxx series.

High Density Terminals (HD Terminals)



Fig. 30: High Density Terminals

The terminals from these series with 16/32 terminal points are distinguished by a particularly compact design, as the packaging density is twice as large as that of the standard 12 mm bus terminals. Massive conductors and conductors with a wire end sleeve can be inserted directly into the spring loaded terminal point without tools.

Wiring HD Terminals

The High Density Terminals of the ELx8xx and KLx8xx series doesn't support pluggable wiring.

Ultrasonically compacted (ultrasonically welded) strands

Ultrasonically compacted (ultrasonically welded) strands

Ultrasonically compacted (ultrasonically welded) strands can also be connected to the standard and high-density terminals. In this case, please note the tables concerning the wire-size width [▶ 50]!

4.5.2 Wiring

⚠ WARNING

Risk of electric shock and damage of device!

Bring the bus terminal system into a safe, powered down state before starting installation, disassembly or wiring of the bus terminals!

Terminals for standard wiring ELxxxx/KLxxxx and for pluggable wiring ESxxxx/KSxxxx

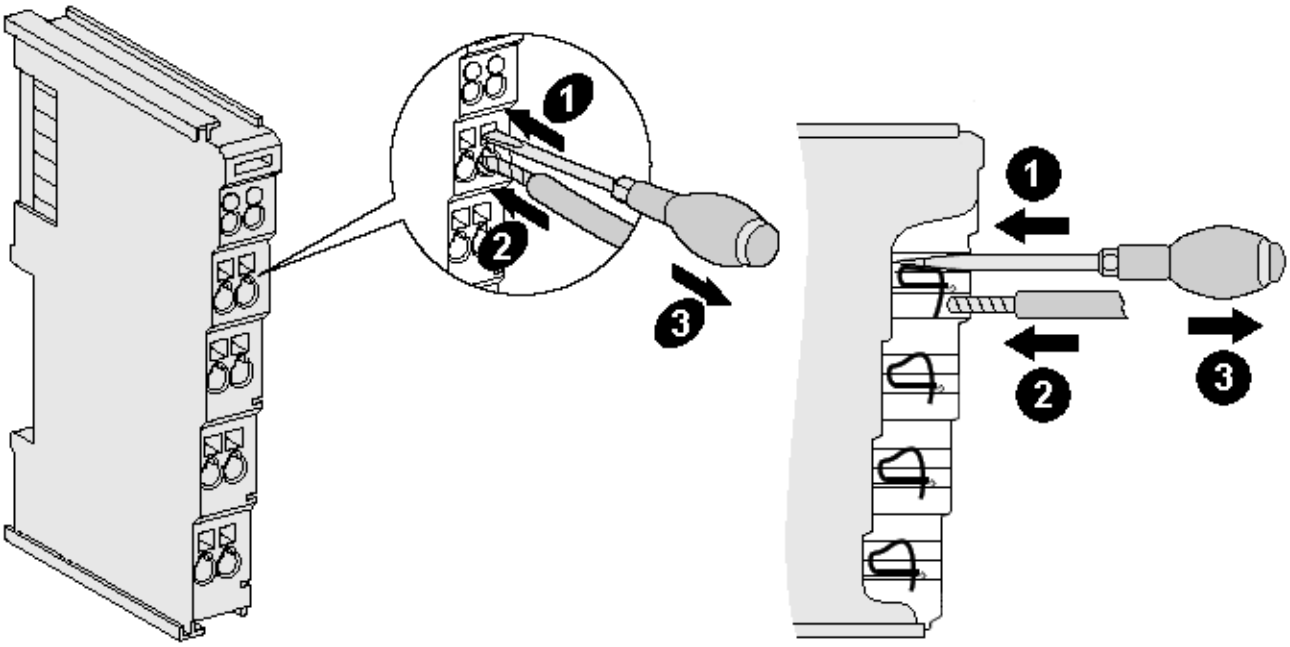


Fig. 31: Connecting a cable on a terminal point

Up to eight terminal points enable the connection of solid or finely stranded cables to the bus terminal. The terminal points are implemented in spring force technology. Connect the cables as follows (see fig. “Connecting a cable on a terminal point”:

1. Open a terminal point by pushing a screwdriver straight against the stop into the square opening above the terminal point. Do not turn the screwdriver or move it alternately (don't toggle).
2. The wire can now be inserted into the round terminal opening without any force.
3. When the screwdriver is removed, the terminal point closes automatically and holds the wire securely and permanently in place

See the following table for the suitable wire size width:

Terminal housing	ELxxxx, KLxxxx	ESxxxx, KSxxxx
Wire size width (single core wires)	0.08 ... 2.5 mm²	0.08 ... 2.5 mm²
Wire size width (fine-wire conductors)	0.08 ... 2.5 mm²	0.08 ... 2.5 mm²
Wire size width (conductors with a wire end sleeve)	0.14 ... 1.5 mm²	0.14 ... 1.5 mm²
Wire stripping length	8 ... 9 mm	9 ... 10 mm

High Density Terminals (HD Terminals [▶ 48]) with 16/32 terminal points

The conductors of the HD Terminals are connected without tools for single-wire conductors using the direct plug-in technique, i.e. after stripping the wire is simply plugged into the terminal point. The cables are released, as usual, using the contact release with the aid of a screwdriver. See the following table for the suitable wire size width.

Terminal housing	High Density Housing
Wire size width (single core wires)	0.08 ... 1.5 mm²
Wire size width (fine-wire conductors)	0.25 ... 1.5 mm²
Wire size width (conductors with a wire end sleeve)	0.14 ... 0.75 mm²
Wire size width (ultrasonically compacted [ultrasonically welded] strands)	only 1.5 mm² (see notice [▶ 48])
Wire stripping length	8 ... 9 mm

4.5.3 Shielding

Shielding

Encoder, analog sensors and actuators should always be connected with shielded, twisted paired wires.

4.6 Installation positions

NOTICE

Constraints regarding installation position and operating temperature range

- Please refer to the technical data of the device to ascertain whether any restrictions regarding the installation position and/or the operating temperature range have been specified.
- When installing devices with increased heat dissipation, ensure that there is sufficient space above and below the devices during operation to guarantee adequate ventilation!

The installation positions and their names for mounting devices on mounting rails are specified below. The illustration of the devices in the following figures is an example.
The following applies to all installation positions: The reference direction "down" (see arrow) is the acceleration of gravity.

Horizontal installation (standard installation)

The mounting rail is mounted horizontally on a vertical mounting plate.
The connection level of the devices points to the front.

The devices are ventilated from below, which enables optimum cooling of the electronics through convection. This is therefore also the recommended installation position.

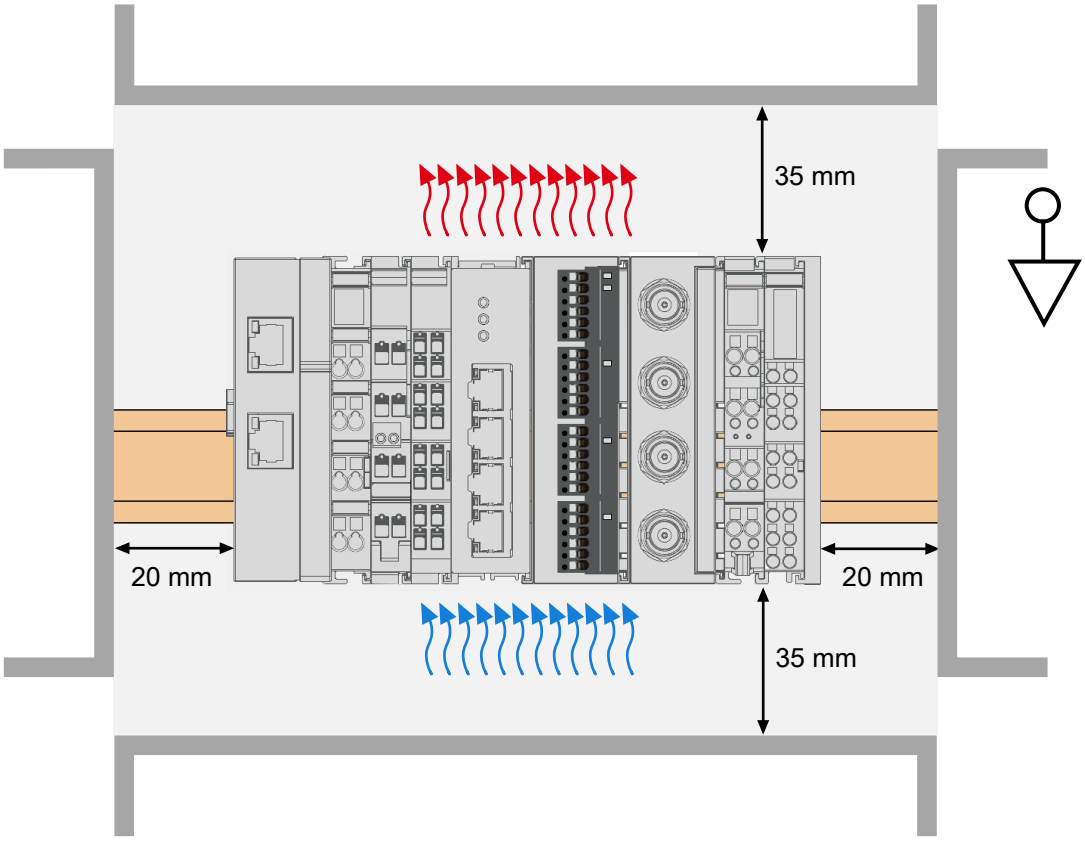


Fig. 32: Recommended minimum distances for standard installation position

NOTICE

Compliance with the minimum distances

Compliance with the minimum distances shown in the figure "Recommended minimum distances for standard installation position" is strongly recommended in all installation positions.

Vertical installation

The mounting rail is mounted vertically on a vertical mounting plate.
The connection level of the devices points to the front.
The devices can be arranged as follows:

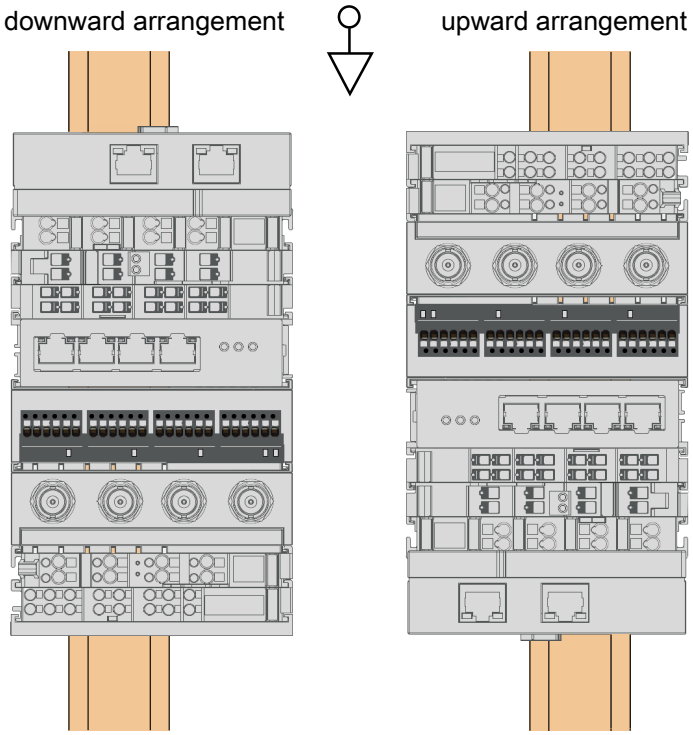


Fig. 33: Installation position Vertical, downward arrangement (left) / upward arrangement (right)

Flat installation

In the flat installation position, the mounting rail is laid on a horizontal mounting plate.
The connection level of the devices points upwards.

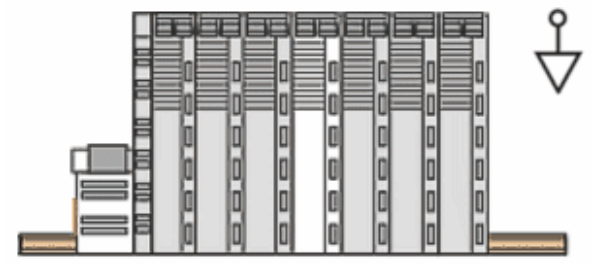


Fig. 34: Flat installation position

NOTICE

Danger of sliding off the mounting rail

Particularly in the "vertical" installation position, but also in other installation positions under corresponding mechanical load, the terminal segment may move on the mounting rail. These can lead to undesirable malfunctions.

- If this risk exists, secure the terminal segment with appropriate locking devices, e.g. by clamping it to the mounting rail.

NOTICE

Compliance with the minimum distances

Compliance with the minimum distances shown in the figure "Recommended minimum distances for standard installation position" is strongly recommended in all installation positions.

Installation positions with ZB8610 fan cartridge

If the cooling should or must be increased for the intended application, the ZB8610 fan cartridge can be mounted on the underside of the device. In the horizontal installation position, the devices are ventilated from bottom to top by the fan cartridge. The optimum cooling is further enhanced by convection ventilation (see following figure).

The fan cartridge can be used in any installation position.

Further information on operation with and without a fan can be found in the technical data for the device (e.g. derating, information on installation positions, etc.).

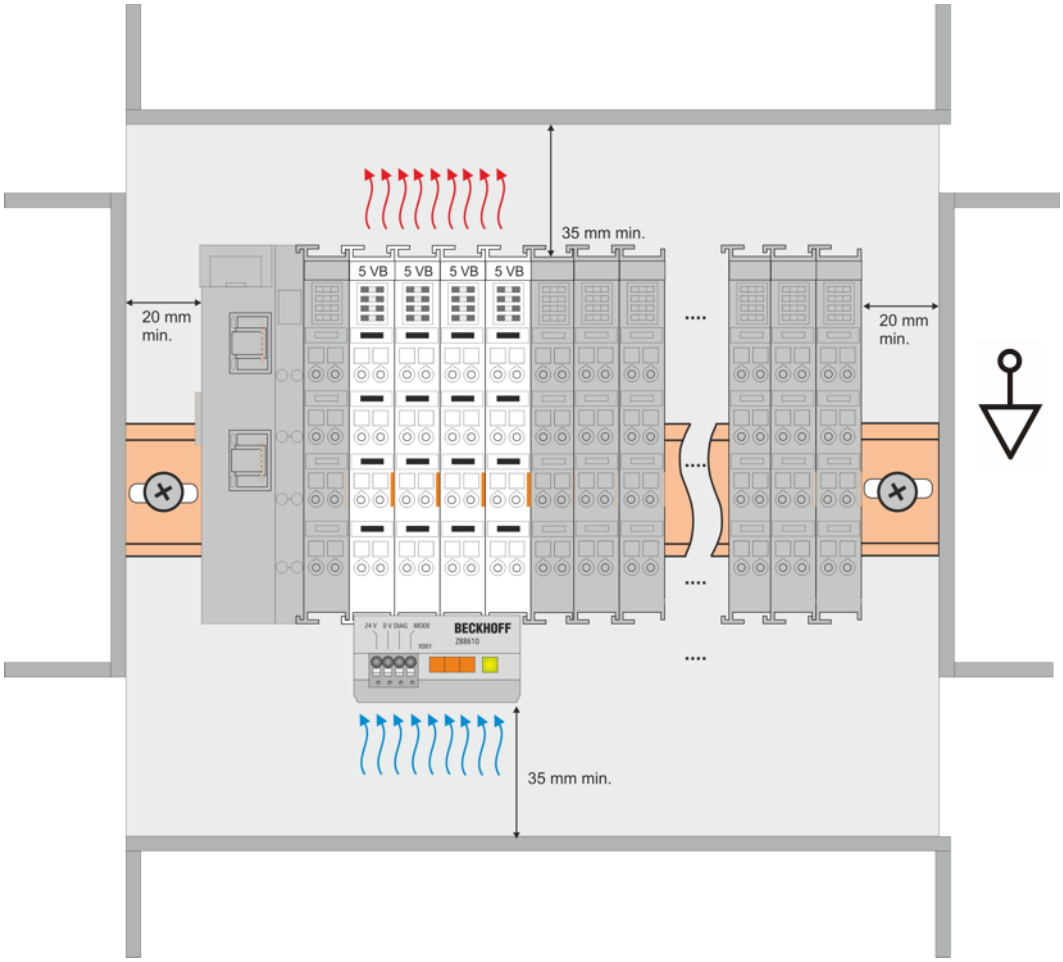


Fig. 35: Recommended minimum distances for operation with fan, using horizontal installation position as an example

NOTICE

Compliance with the minimum distances

Compliance with the minimum distances as shown in the figure "Recommended minimum distances for operation with fan" is strongly recommended.

4.7 Disposal



Products marked with a crossed-out wheeled bin shall not be discarded with the normal waste stream. The device is considered as waste electrical and electronic equipment. The national regulations for the disposal of waste electrical and electronic equipment must be observed.

5 Commissioning

5.1 TwinCAT/EtherCAT slave

5.1.1 TwinCAT Quick Start

TwinCAT is a development environment for real-time control including a multi PLC system, NC axis control, programming and operation. The whole system is mapped through this environment and enables access to a programming environment (including compilation) for the controller. Individual digital or analog inputs or outputs can also be read or written directly, in order to verify their functionality, for example.

For further information, please refer to http://infosys.beckhoff.com:

- **EtherCAT System Manual:**
  Fieldbus Components → EtherCAT Terminals → EtherCAT System Documentation → Setup in the TwinCAT System Manager
- **TwinCAT 2** → TwinCAT System Manager → I/O Configuration
- In particular, for TwinCAT – driver installation:
  Fieldbus components → Fieldbus Cards and Switches → FC900x – PCI Cards for Ethernet → Installation

Devices contain the relevant terminals for the actual configuration. All configuration data can be entered directly via editor functions (offline) or via the `scan function (online):

- **“offline”**: The configuration can be customized by adding and positioning individual components. These can be selected from a directory and configured.
  - The procedure for the offline mode can be found under http://infosys.beckhoff.com:
    TwinCAT 2 → TwinCAT System Manager → IO Configuration → Add an I/O device
- **“online”**: The existing hardware configuration is read
  - See also http://infosys.beckhoff.com:
    Fieldbus components → Fieldbus Cards and Switches → FC900x – PCI Cards for Ethernet → Installation → Searching for devices

The following relationship is envisaged between the user PC and individual control elements:

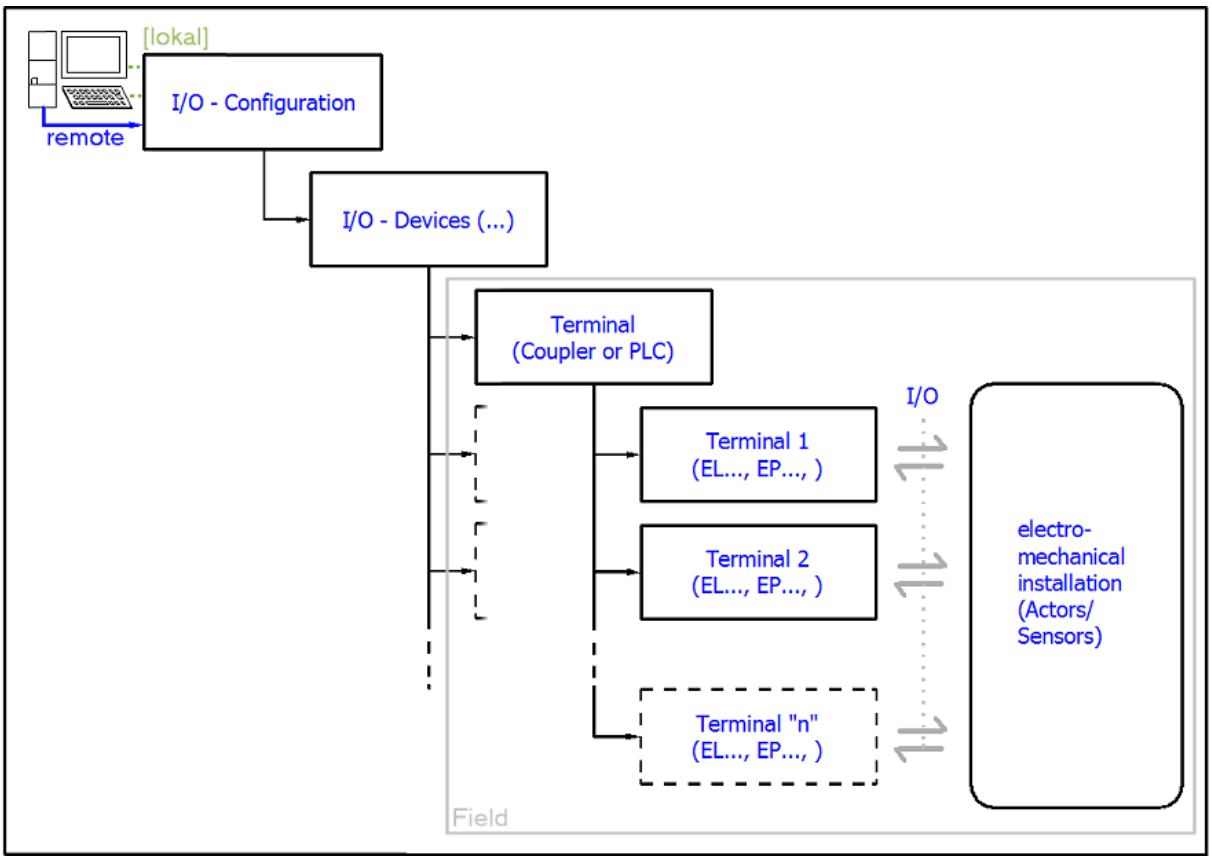


Fig. 36: Relationship between user side (commissioning) and installation

Insertion of certain components (I/O device, terminal, box...) by users functions the same way as in TwinCAT 2 and TwinCAT 3. The descriptions below relate solely to the online procedure.

Example configuration (actual configuration)

Based on the following example configuration, the subsequent subsections describe the procedure for TwinCAT 2 and TwinCAT 3:

- **CX2040** control system (PLC) including **CX2100-0004** power supply unit
- Connected to CX2040 on the right (E-bus):
  EL1004 (4-channel digital input terminal 24 V_{DC})
- Linked via the X001 port (RJ-45): **EK1100** EtherCAT Coupler
- Connected to the EK1100 EtherCAT Coupler on the right (E-bus):
  EL2008 (8-channel digital output terminal 24 V_{DC}; 0.5 A)
- (Optional via X000: a link to an external PC for the user interface)

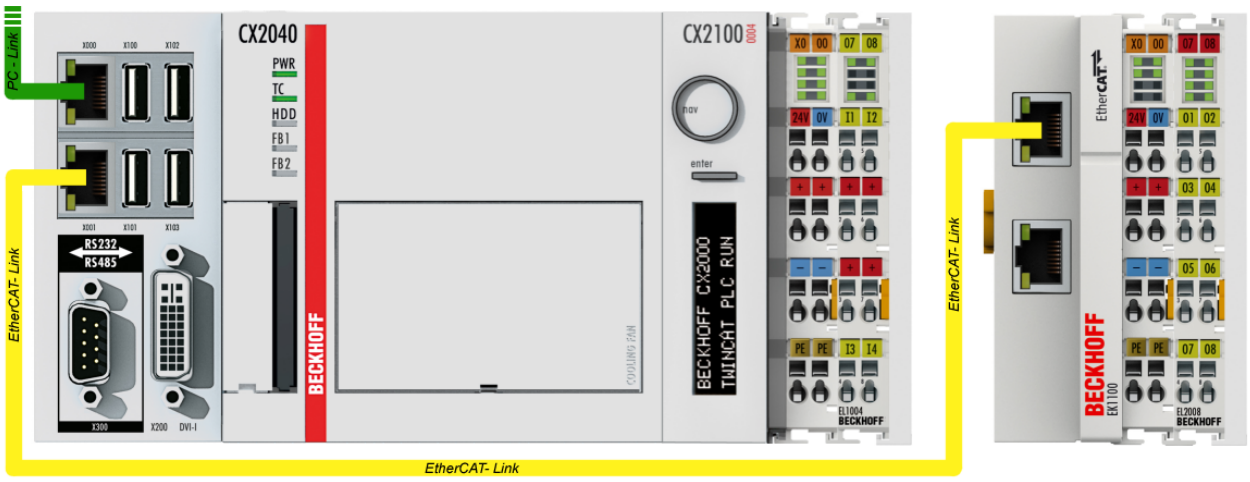


Fig. 37: Control configuration with Embedded PC, input (EL1004) and output (EL2008)

Note that all combinations of a configuration are possible; for example, the EL1004 terminal could also be connected after the coupler, or the EL2008 terminal could additionally be connected to the CX2040 on the right, in which case the EK1100 coupler wouldn't be necessary.

5.1.1.1 TwinCAT 2

Startup

TwinCAT 2 basically uses two user interfaces: the TwinCAT System Manager for communication with the electromechanical components and TwinCAT PLC Control for the development and compilation of a controller. The starting point is the TwinCAT System Manager.

After successful installation of the TwinCAT system on the PC to be used for development, the TwinCAT 2 System Manager displays the following user interface after startup:

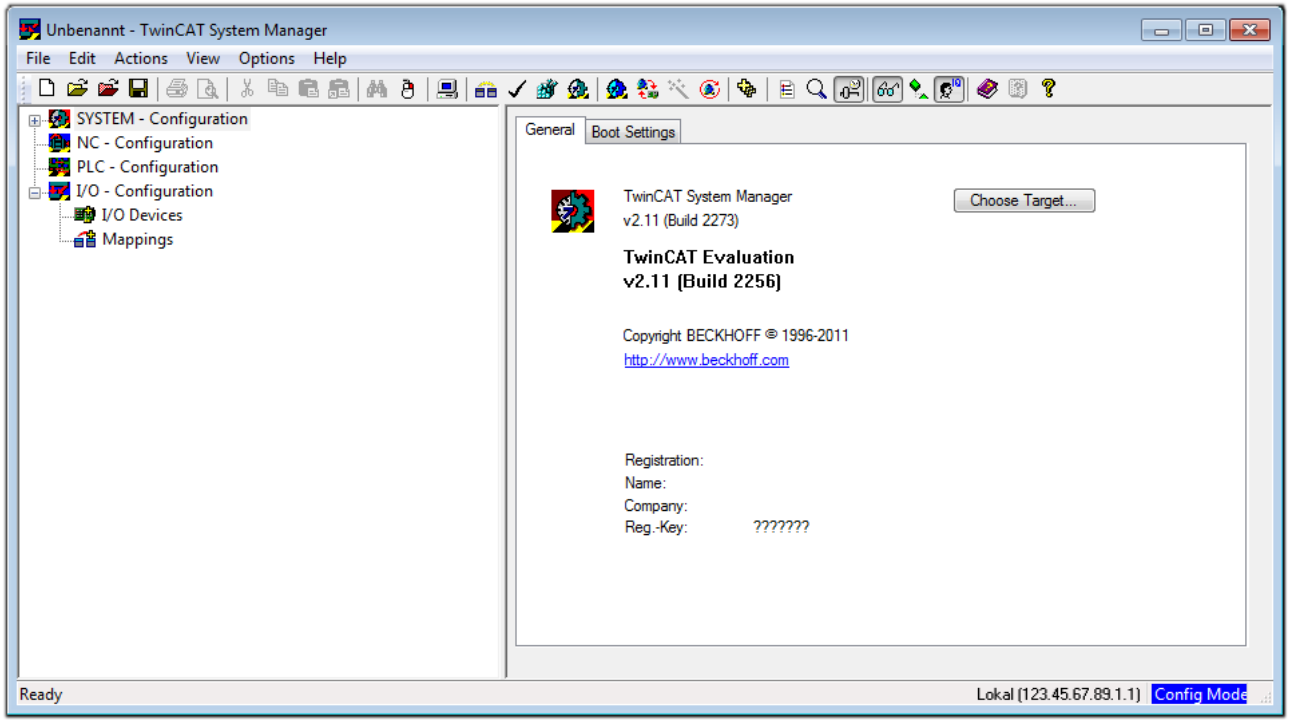


Fig. 38: Initial TwinCAT 2 user interface

Generally, TwinCAT can be used in local or remote mode. Once the TwinCAT system, including the user interface (standard) is installed on the respective PLC, TwinCAT can be used in local mode and thus the next step is “Insert Device [▶ 60]”.

If the intention is to address the TwinCAT runtime environment installed on a PLC remotely from another system used as a development environment, the target system must be made known first. In the menu under “Actions” → “Choose Target System...”, the following window is opened for this via the symbol “  ” or the “F8” key:

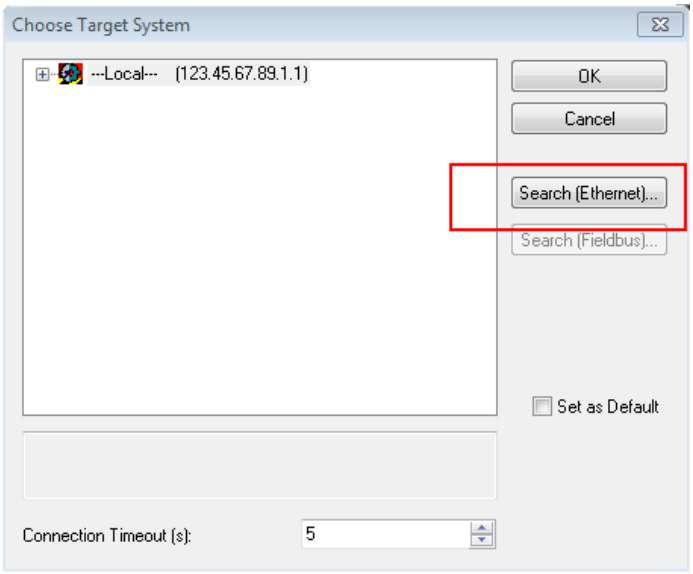


Fig. 39: Selection of the target system

Use “Search (Ethernet)...” to enter the target system. Thus another dialog opens to either:

- enter the known computer name after “Enter Host Name / IP:” (as shown in red)
- perform a “Broadcast Search” (if the exact computer name is not known)
- enter the known computer – IP or AmsNetID

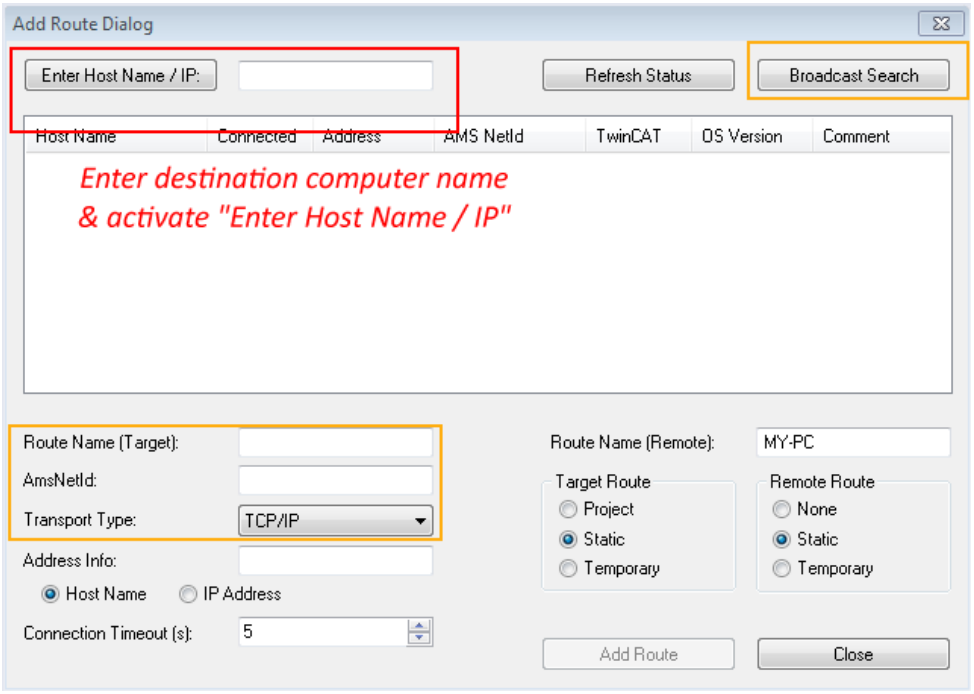
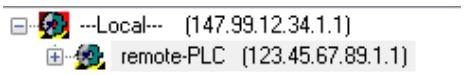


Fig. 40: specify the PLC for access by the TwinCAT System Manager: selection of the target system

Once the target system has been entered, it is available for selection as follows (a correct password may have to be entered before this):



After confirmation with “OK”, the target system can be accessed via the System Manager.

Adding devices

In the configuration tree of the TwinCAT 2 System Manager user interface on the left, select “I/O Devices” and then right-click to open a context menu and select “Scan Devices…”, or start the action in the menu bar via  . The TwinCAT System Manager may first have to be set to “Config Mode” via  or via the menu
“Actions” → “Set/Reset TwinCAT to Config Mode…” (Shift + F4).

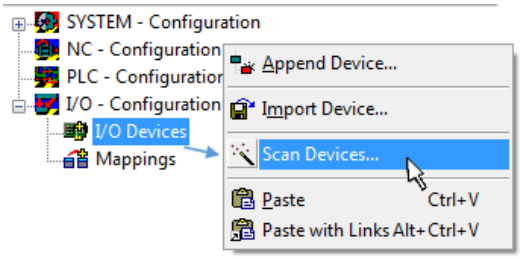


Fig. 41: Select “Scan Devices...”

Confirm the warning message, which follows, and select the “EtherCAT” devices in the dialog:

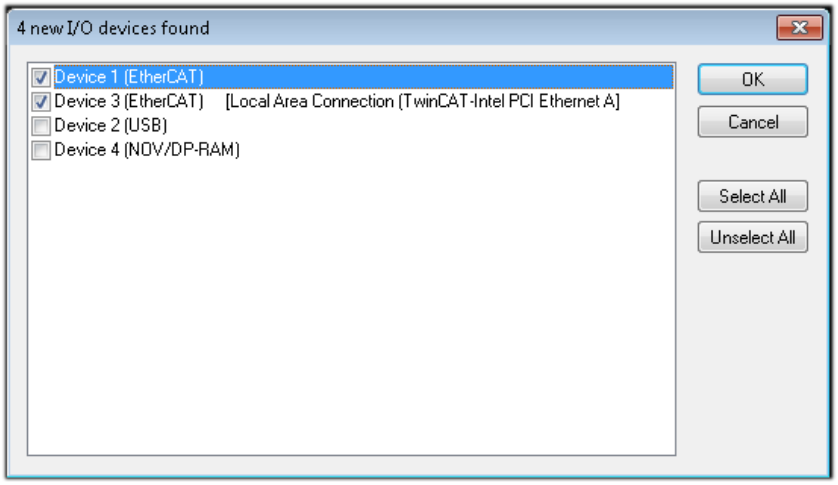


Fig. 42: Automatic detection of I/O devices: selection of the devices to be integrated

Confirm the message “Find new boxes”, in order to determine the terminals connected to the devices. “Free Run” enables manipulation of input and output values in “Config Mode” and should also be acknowledged.

Based on the example configuration [▶ 56] described at the beginning of this section, the result is as follows:

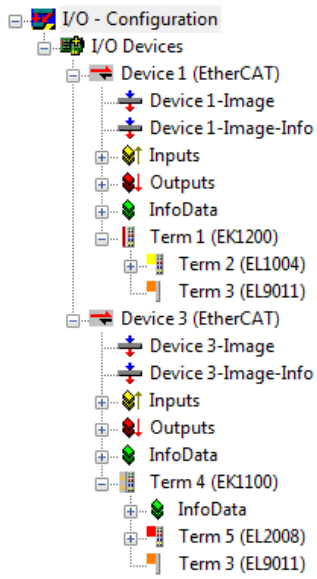


Fig. 43: Mapping of the configuration in the TwinCAT 2 System Manager

The whole process consists of two stages, which can also be performed separately (first determine the devices, then determine the connected elements such as boxes, terminals, etc.). A scan (search function) can also be initiated by selecting “Device ...” from the context menu, which then only reads the elements below which are present in the configuration:

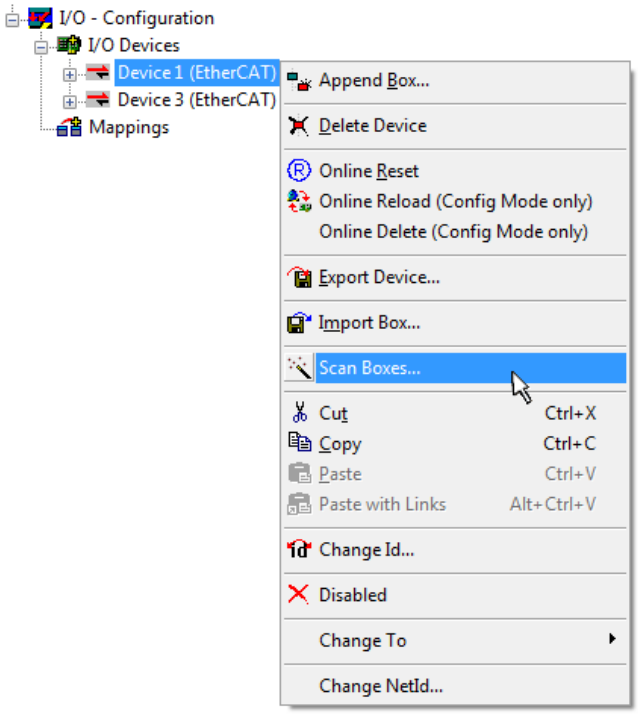


Fig. 44: Reading of individual terminals connected to a device

This functionality is useful if the actual configuration is modified at short notice.

Programming and integrating the PLC

TwinCAT PLC Control is the development environment for generating the controller in different program environments: TwinCAT PLC Control supports all languages described in IEC 61131-3. There are two text-based languages and three graphical languages.

- **Text-based languages**
  - Instruction List (IL)
  - Structured Text (ST)

- **Graphical languages**
  - Function Block Diagram (FBD)
  - Ladder Diagram (LD)
  - The Continuous Function Chart Editor (CFC)
  - Sequential Function Chart (SFC)

The following section refers solely to Structured Text (ST).

After starting TwinCAT PLC Control, the following user interface is shown for an initial project:

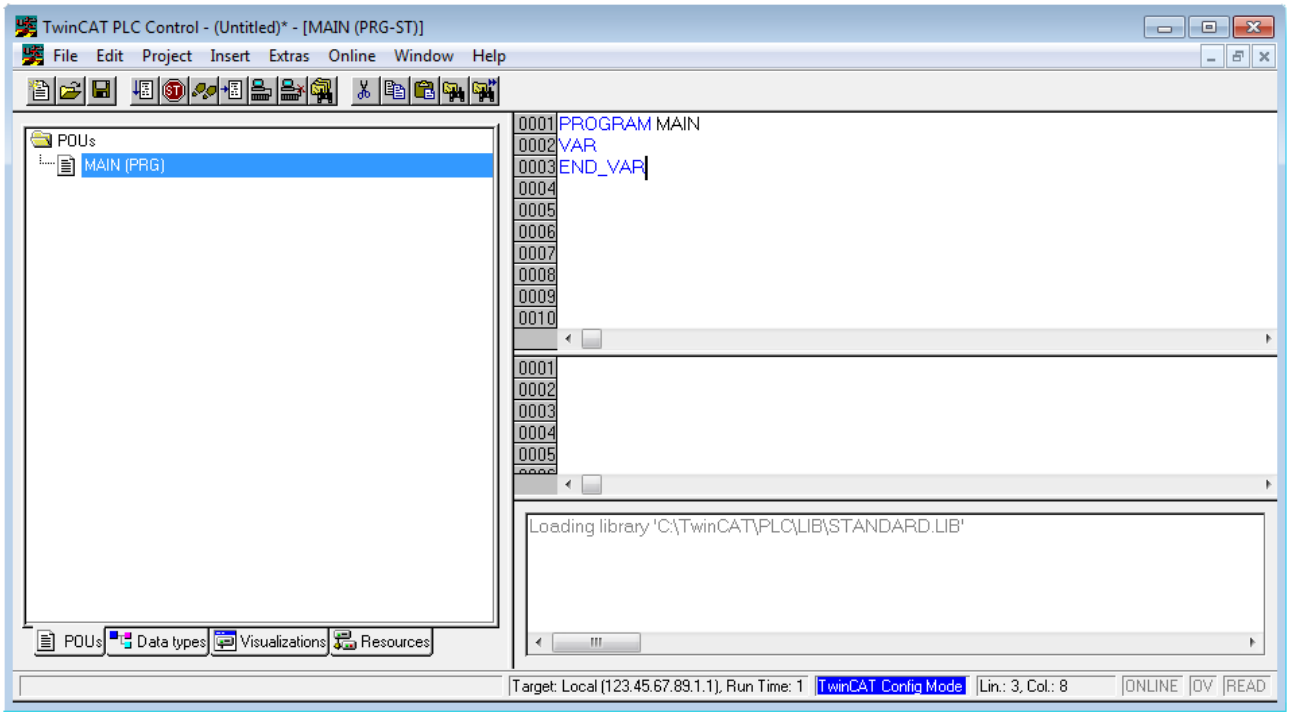


Fig. 45: TwinCAT PLC Control after startup

Example variables and an example program have been created and stored under the name “PLC_example.pro”:

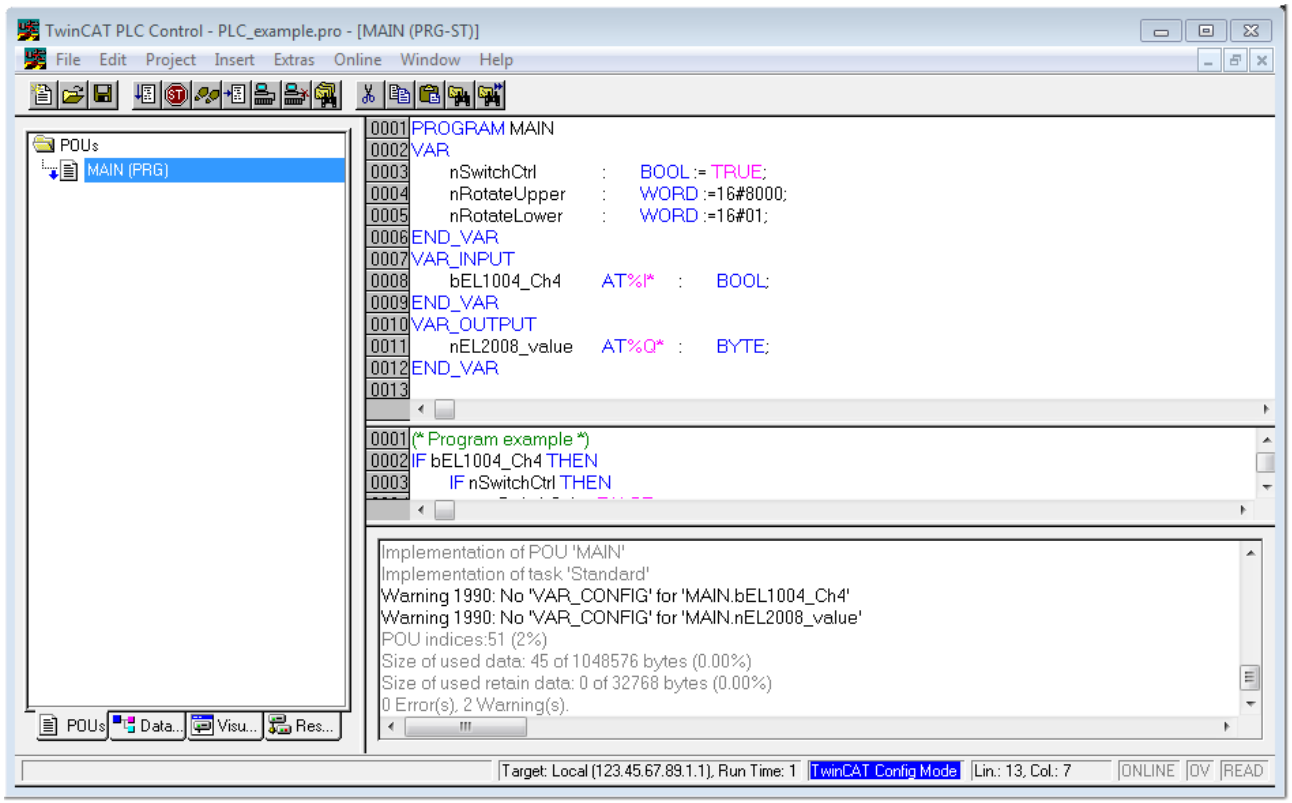


Fig. 46: Example program with variables after a compile process (without variable integration)

Warning 1990 (missing “VAR_CONFIG”) after a compile process indicates that the variables defined as external (with the ID “AT%I*” or “AT%Q*”) have not been assigned. After successful compilation, TwinCAT PLC Control creates a “*.tpy” file in the directory in which the project was stored. This file (“*.tpy”) contains variable assignments and is not known to the System Manager, hence the warning. Once the System Manager has been notified, the warning no longer appears.

First, integrate the TwinCAT PLC Control project in the **System Manager**. This is performed via the context menu of the PLC configuration (right-click) and selecting “Append PLC Project…”:

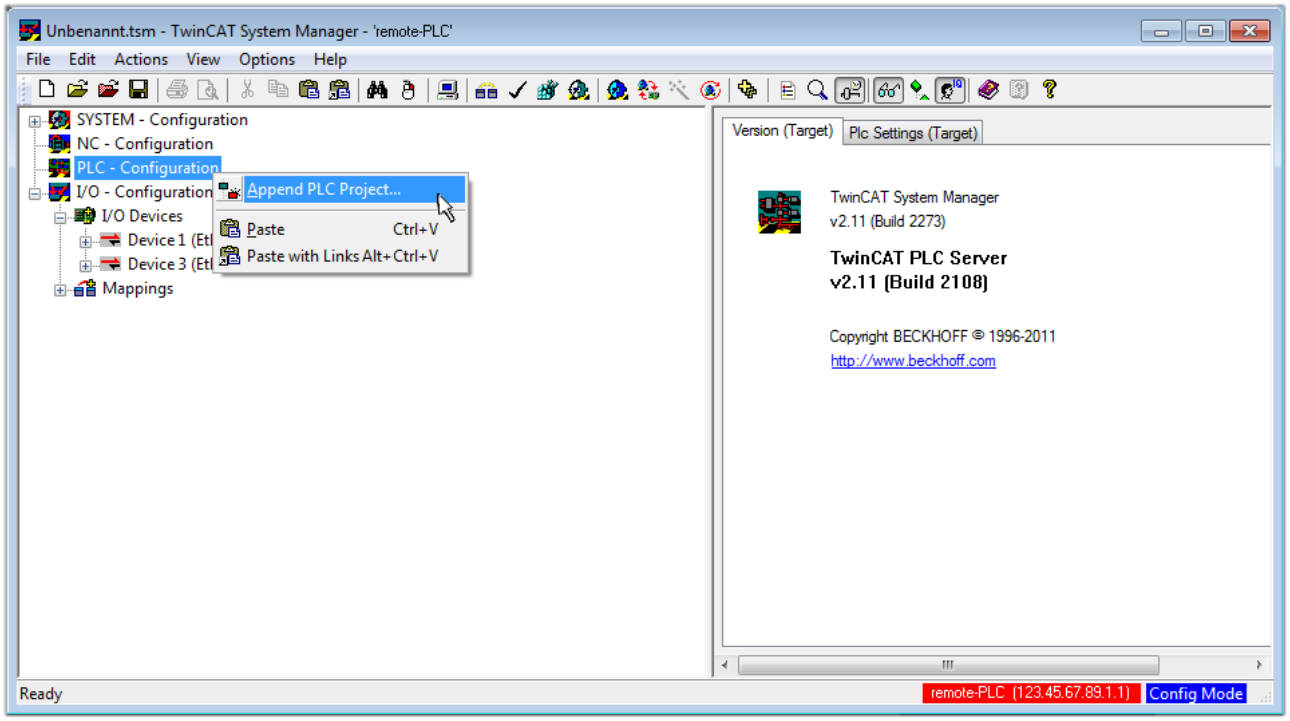


Fig. 47: Appending the TwinCAT PLC Control project

Select the PLC configuration “PLC_example.tpy” in the browser window that opens. The project including the two variables identified with “AT” are then integrated in the configuration tree of the System Manager:

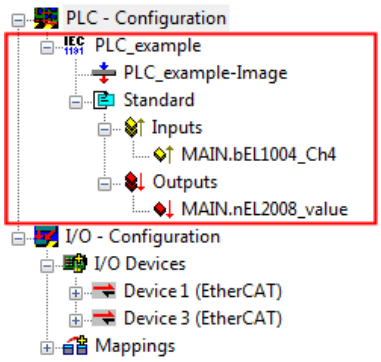


Fig. 48: PLC project integrated in the PLC configuration of the System Manager

The two variables “bEL1004_Ch4” and “nEL2008_value” can now be assigned to certain process objects of the I/O configuration.

Assigning variables

Open a window for selecting a suitable process object (PDO) via the context menu of a variable of the integrated project “PLC_example” and via “Modify Link...” “Standard”:

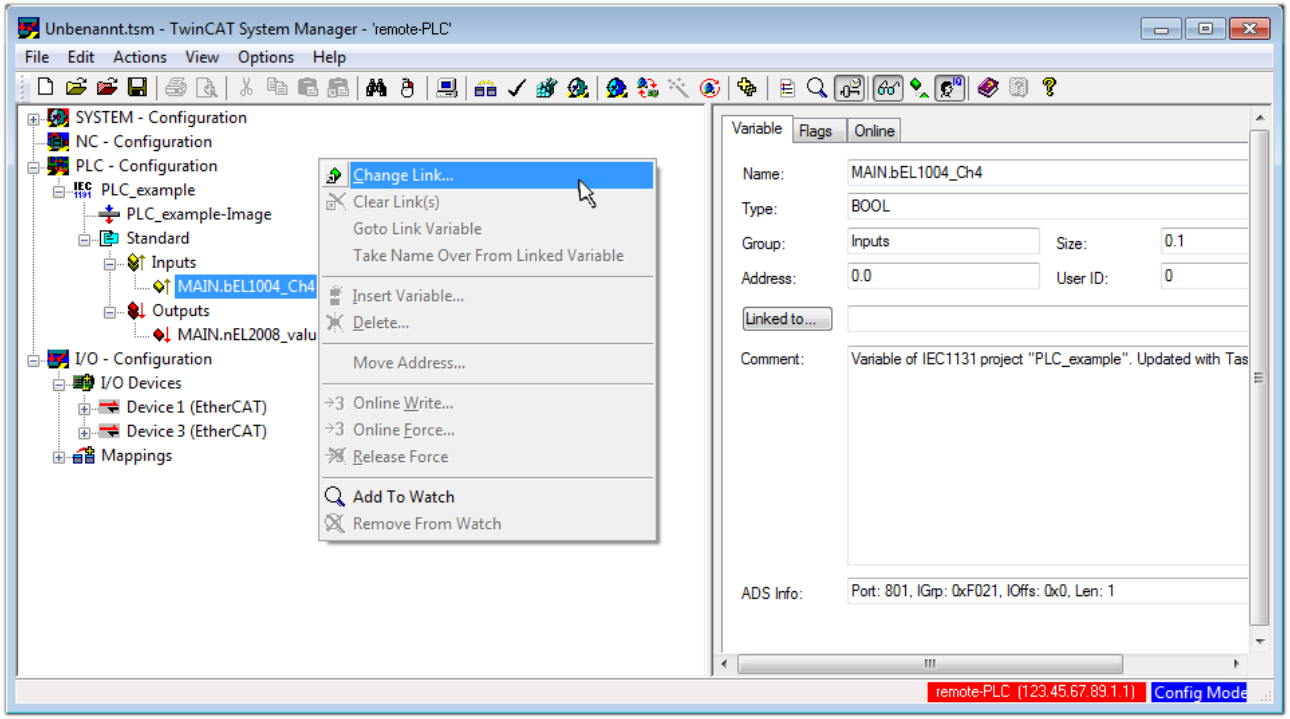


Fig. 49: Creating the links between PLC variables and process objects

In the window that opens, the process object for the “bEL1004_Ch4” BOOL-type variable can be selected from the PLC configuration tree:

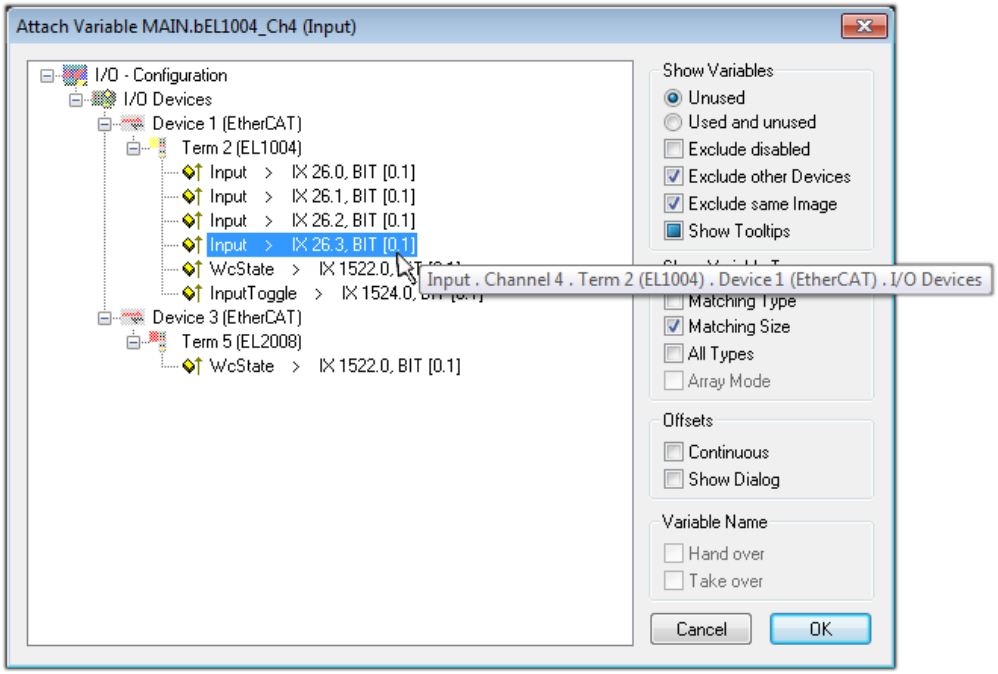


Fig. 50: Selecting BOOL-type PDO

According to the default setting, only certain PDO objects are now available for selection. In this example, the input of channel 4 of the EL1004 terminal is selected for linking. In contrast, the checkbox “All types” must be ticked to create the link for the output variables, in order to allocate a set of eight separate output bits to a byte variable in this case. The following diagram shows the whole process:

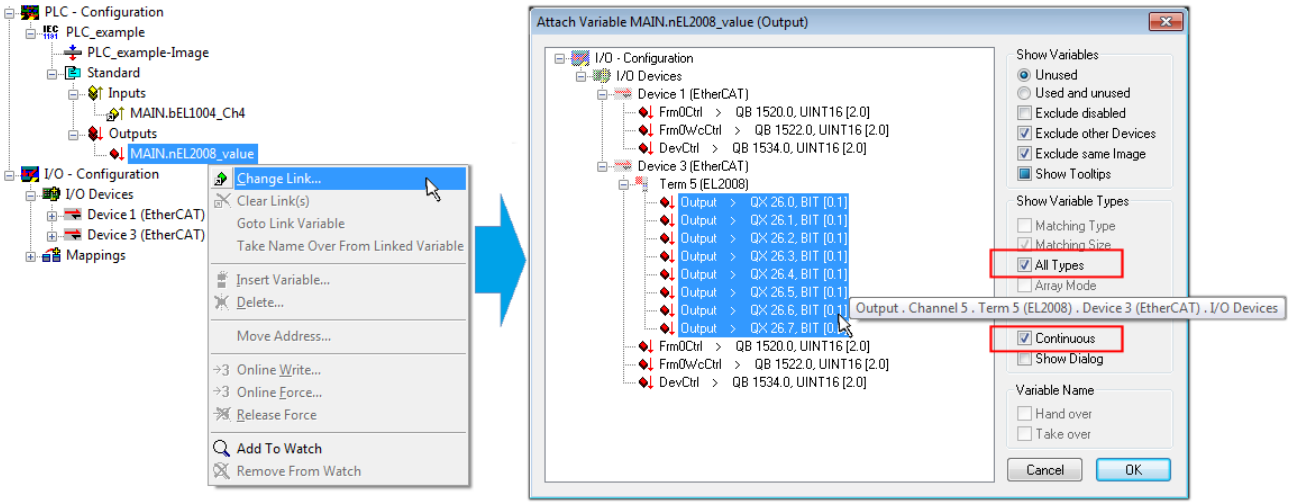


Fig. 51: Selecting several PDOs simultaneously: activate “Continuous” and “All types”

Note that the “Continuous” checkbox was also activated. This is designed to allocate the bits contained in the byte of the “nEL2008_value” variable sequentially to all eight selected output bits of the EL2008 Terminal. It is thus possible to subsequently address all eight outputs of the terminal in the program with a byte corresponding to bit 0 for channel 1 to bit 7 for channel 8 of the PLC. A special symbol ( ) on the yellow or red object of the variable indicates that a link exists. The links can also be checked by selecting “Goto Link Variable” from the context menu of a variable. The opposite linked object, in this case the PDO, is automatically selected:

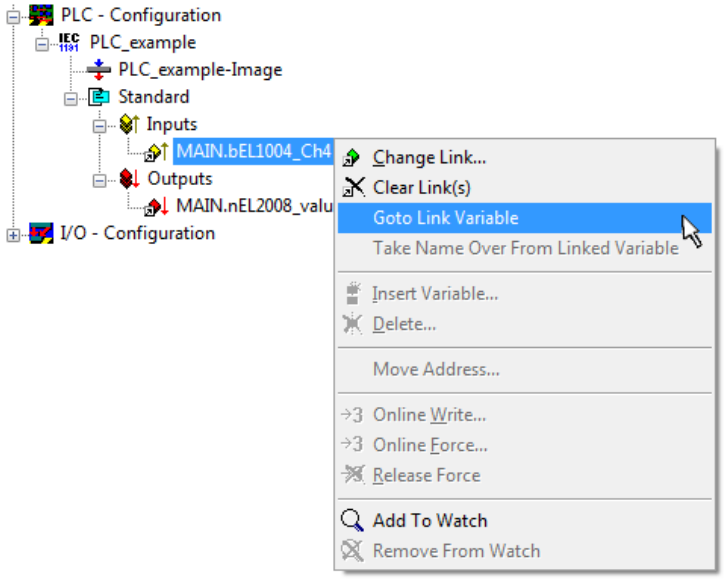
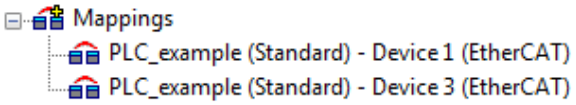


Fig. 52: Application of a “Goto Link Variable”, using “MAIN.bEL1004_Ch4” as an example

The process of assigning variables to the PDO is completed via the menu option “Actions” → “Create assignment”, or via  .

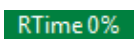
This can be visualized in the configuration:



The process of creating links can also be performed in the opposite direction, i.e. starting with individual PDOs to a variable. However, in this example, it would not be possible to select all output bits for the EL2008, since the terminal only makes individual digital outputs available. If a terminal has a byte, word, integer or similar PDO, it is also possible to allocate this to a set of bit-standardized variables. Here, too, a “Goto Link Variable” can be executed in the other direction, so that the respective PLC instance can then be selected.

Activation of the configuration

The allocation of PDO to PLC variables has now established the connection from the controller to the inputs and outputs of the terminals. The configuration can now be activated. First, the configuration can be verified via  (or via “Actions” → “Check Configuration”). If no error is present, the configuration can be activated via  (or via “Actions” → “Activate Configuration…”) to transfer the System Manager settings to the runtime system. Confirm the messages “Old configurations will be overwritten!” and “Restart TwinCAT system in Run mode” with “OK”.

A few seconds later, the real-time status 

is displayed at the bottom right in the System Manager. The PLC system can then be started as described below.

Starting the controller

Starting from a remote system, the PLC control has to be linked with the embedded PC over the Ethernet via “Online” → “Choose Runtime System…”:

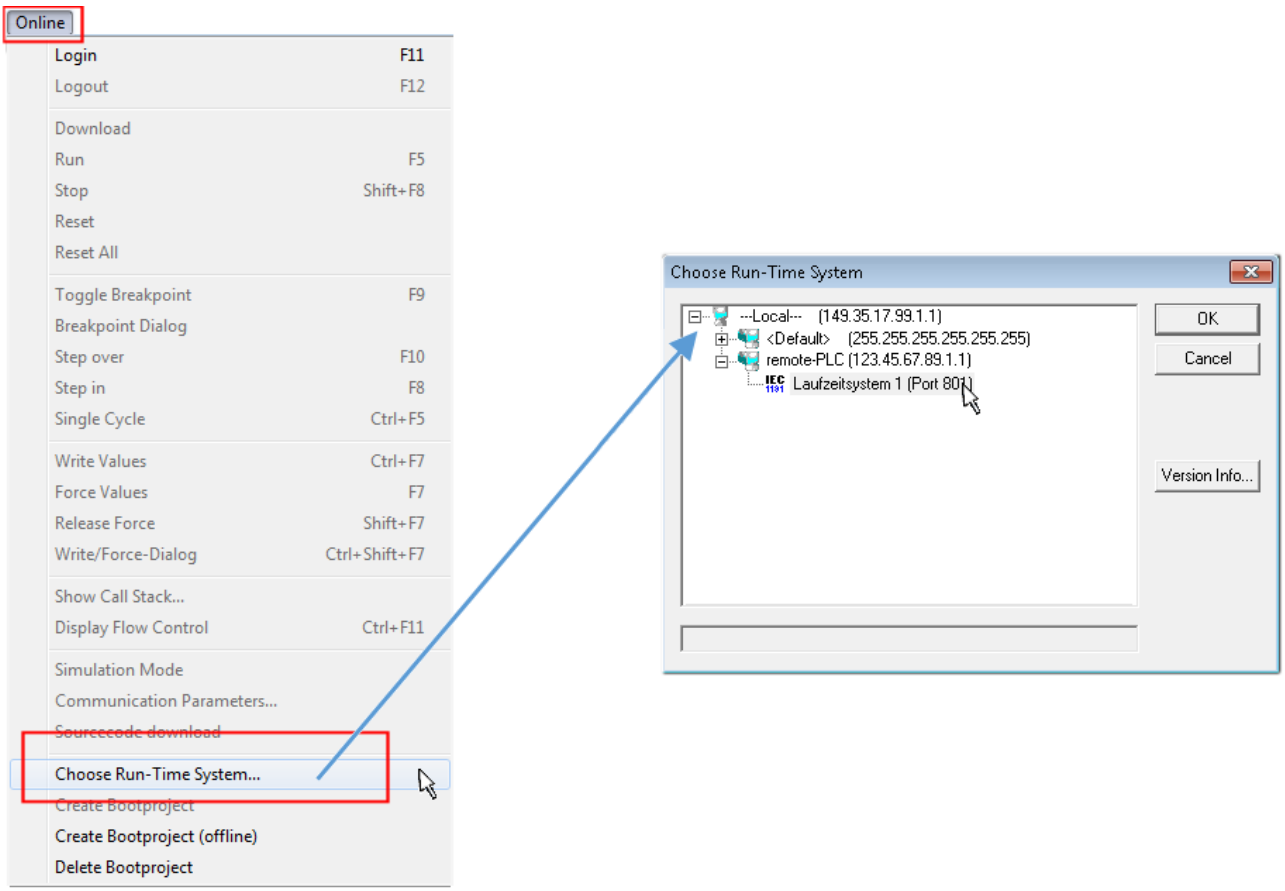


Fig. 53: Choose target system (remote)

In this example, “Runtime system 1 (port 801)” is selected and confirmed. Link the PLC with the real-time system via the menu option “Online” → “Login”, the F11 key or by clicking on the symbol . The control program can then be loaded for execution. This results in the message “No program on the controller! Should the new program be loaded?”, which should be confirmed with “Yes”. The runtime environment is ready for the program start:

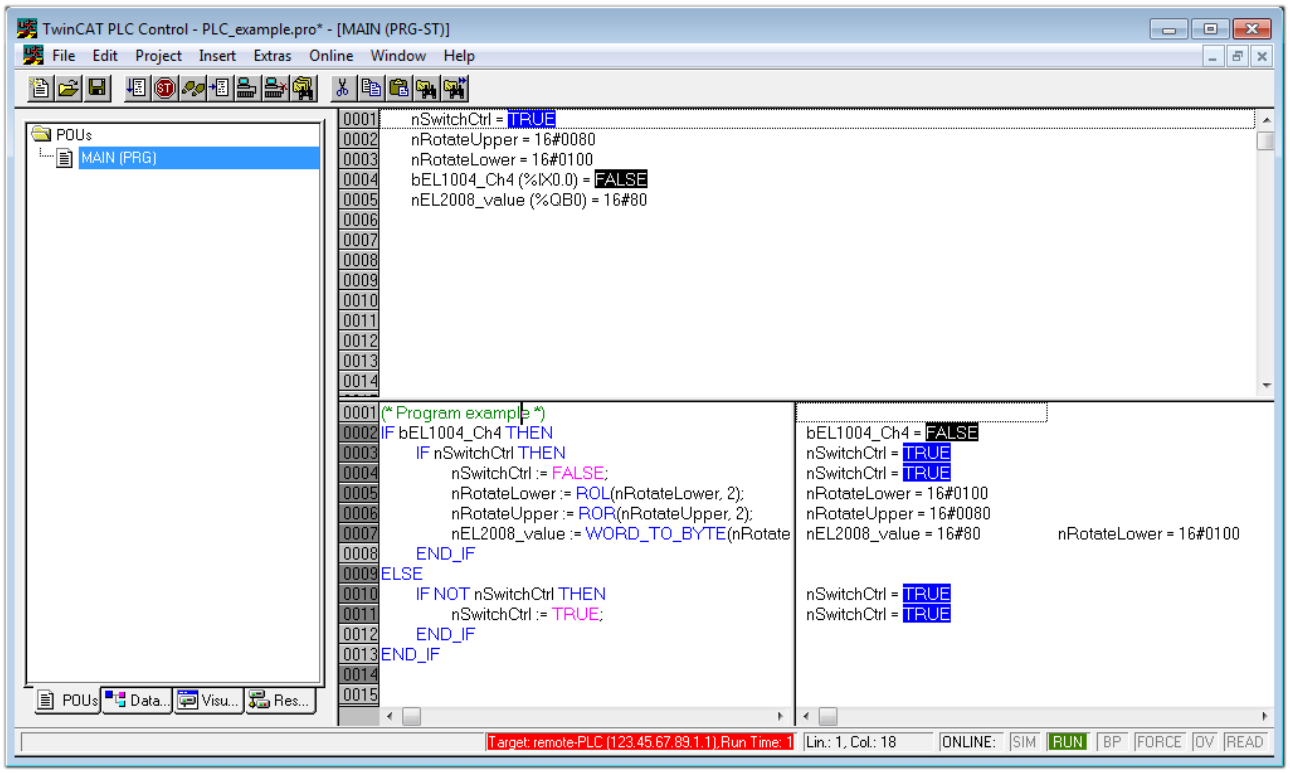


Fig. 54: PLC Control logged in, ready for program startup

The PLC can now be started via “Online” → “Run”, F5 key or .

5.1.1.2 TwinCAT 3

Startup

TwinCAT 3 makes the development environment areas available all together, with Microsoft Visual Studio: after startup, the project folder explorer appears on the left in the general window area (see “TwinCAT System Manager” of TwinCAT 2) for communication with the electromechanical components.

After successful installation of the TwinCAT system on the PC to be used for development, TwinCAT 3 (shell) displays the following user interface after startup:

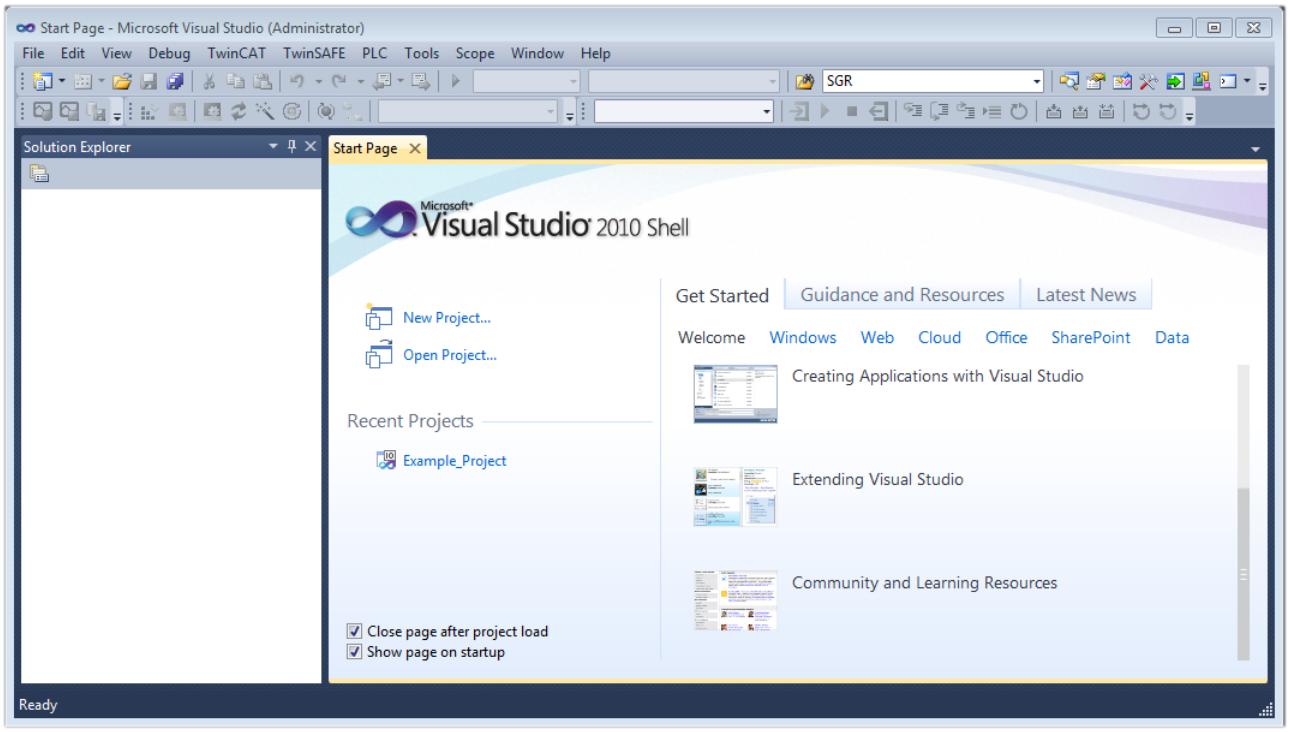


Fig. 55: Initial TwinCAT 3 user interface

First create a new project via  New TwinCAT Project... (or under “File”→“New”→ “Project…”). In the following dialog, make the corresponding entries as required (as shown in the diagram):

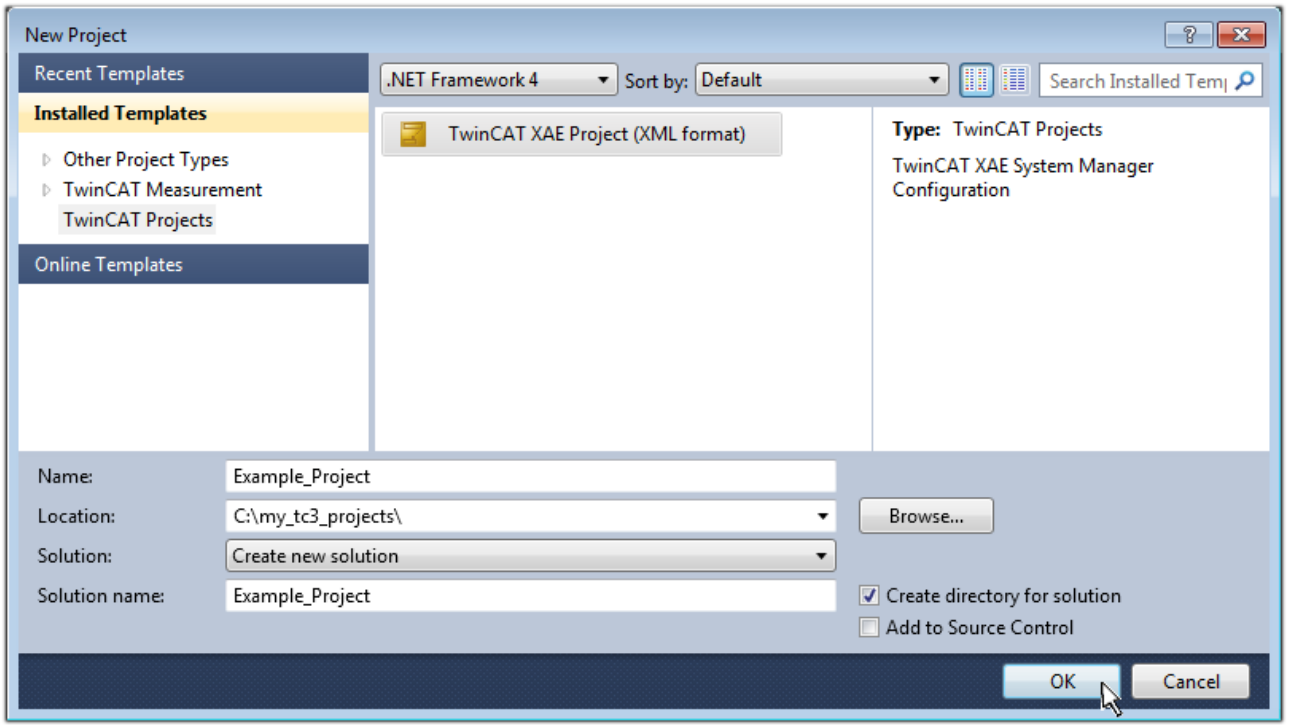


Fig. 56: Create new TwinCAT 3 project

The new project is then available in the project folder explorer:

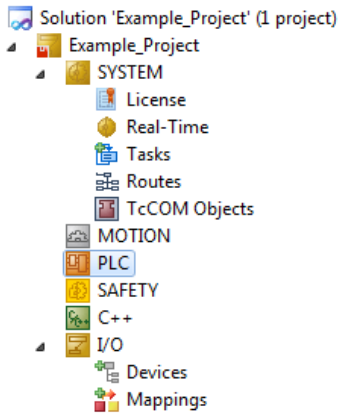
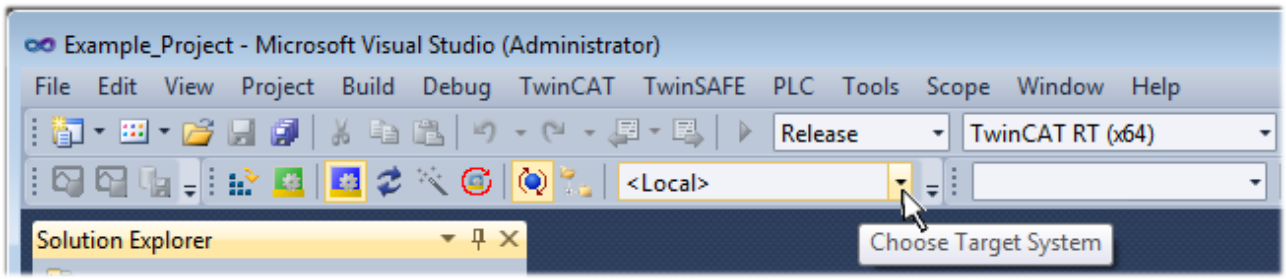


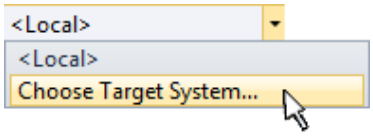
Fig. 57: New TwinCAT 3 project in the project folder explorer

Generally, TwinCAT can be used in local or remote mode. Once the TwinCAT system including the user interface (standard) is installed on the respective PLC (locally), TwinCAT can be used in local mode and the process can be continued with the next step, "Insert Device [▶ 71]".

If the intention is to address the TwinCAT runtime environment installed on a PLC remotely from another system used as a development environment, the target system must be made known first. Via the symbol in the menu bar:



expand the pull-down menu:



and open the following window:

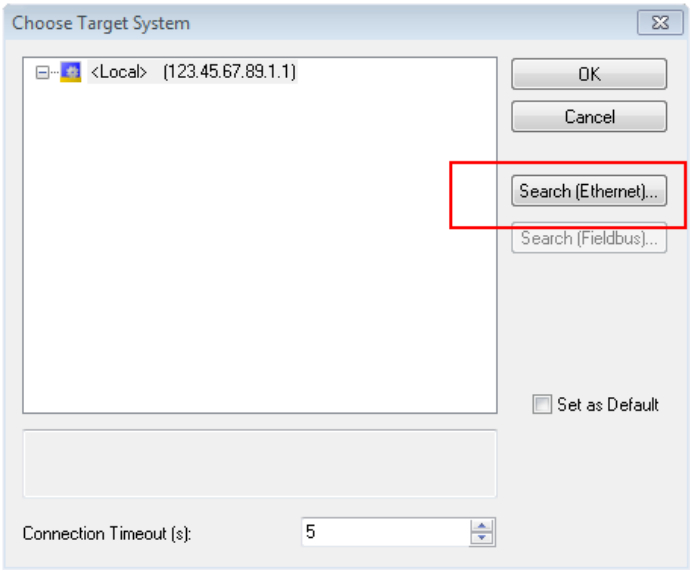


Fig. 58: Selection dialog: Choose the target system

Use “Search (Ethernet)...” to enter the target system. Thus another dialog opens to either:

- enter the known computer name after “Enter Host Name / IP:” (as shown in red)
- perform a “Broadcast Search” (if the exact computer name is not known)
- enter the known computer – IP or AmsNetID

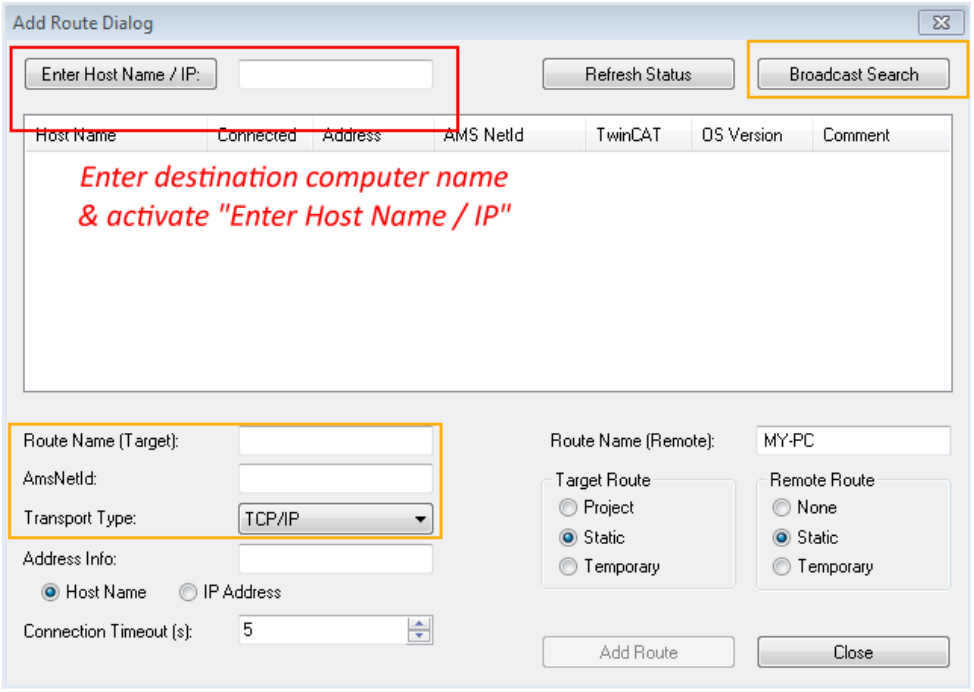
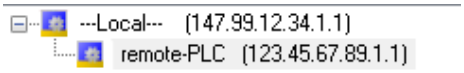


Fig. 59: specify the PLC for access by the TwinCAT System Manager: selection of the target system

Once the target system has been entered, it is available for selection as follows (the correct password may have to be entered beforehand):



After confirmation with “OK” the target system can be accessed via the Visual Studio shell.

Adding devices

In the project folder explorer on the left of the Visual Studio shell user interface, select “Devices” within the element “I/O”, then right-click to open a context menu and select “Scan” or start the action via  in the menu bar. The TwinCAT System Manager may first have to be set to “Config mode” via  or via the menu “TwinCAT” → “Restart TwinCAT (Config Mode)”.

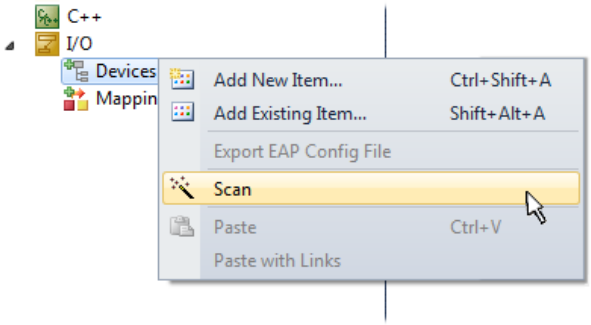


Fig. 60: Select “Scan”

Confirm the warning message, which follows, and select the “EtherCAT” devices in the dialog:

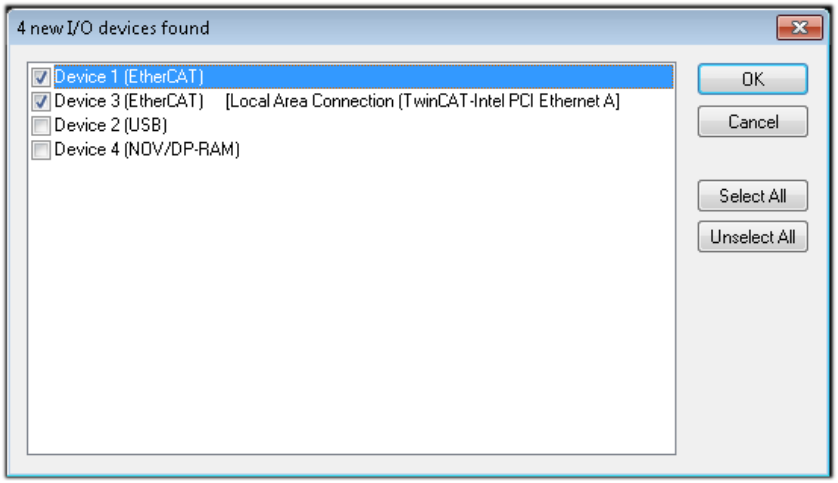


Fig. 61: Automatic detection of I/O devices: selection of the devices to be integrated

Confirm the message “Find new boxes”, in order to determine the terminals connected to the devices. “Free Run” enables manipulation of input and output values in “Config Mode” and should also be acknowledged.

Based on the example configuration [▶ 56] described at the beginning of this section, the result is as follows:

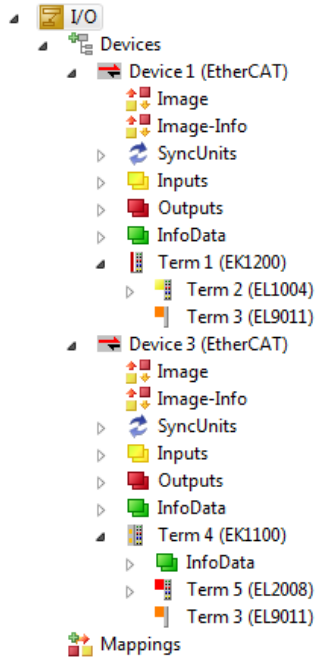


Fig. 62: Mapping of the configuration in VS shell of the TwinCAT 3 environment

The whole process consists of two stages, which can also be performed separately (first determine the devices, then determine the connected elements such as boxes, terminals, etc.). A scan (search function) can also be initiated by selecting “Device ...” from the context menu, which then only reads the elements below which are present in the configuration:

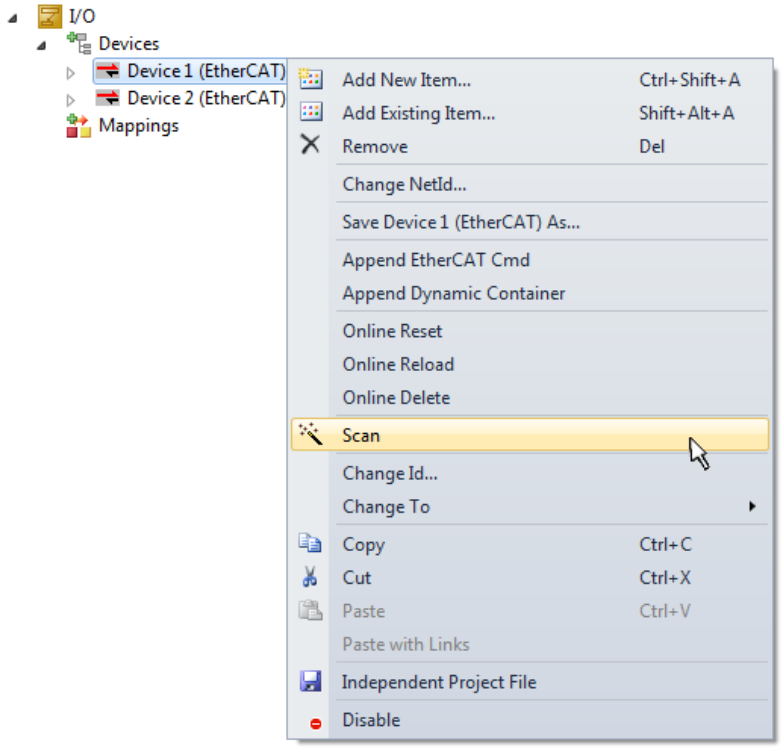


Fig. 63: Reading of individual terminals connected to a device

This functionality is useful if the actual configuration is modified at short notice.

Programming the PLC

TwinCAT PLC Control is the development environment for generating the controller in different program environments: TwinCAT PLC Control supports all languages described in IEC 61131-3. There are two text-based languages and three graphical languages.

- **Text-based languages**
  - Instruction List (IL)
  - Structured Text (ST)
- **Graphical languages**
  - Function Block Diagram (FBD)
  - Ladder Diagram (LD)
  - The Continuous Function Chart Editor (CFC)
  - Sequential Function Chart (SFC)

The following section refers solely to Structured Text (ST).

In order to create a programming environment, a PLC subproject is added to the example project via the context menu of the “PLC” in the project folder explorer by selecting “Add New Item….”:

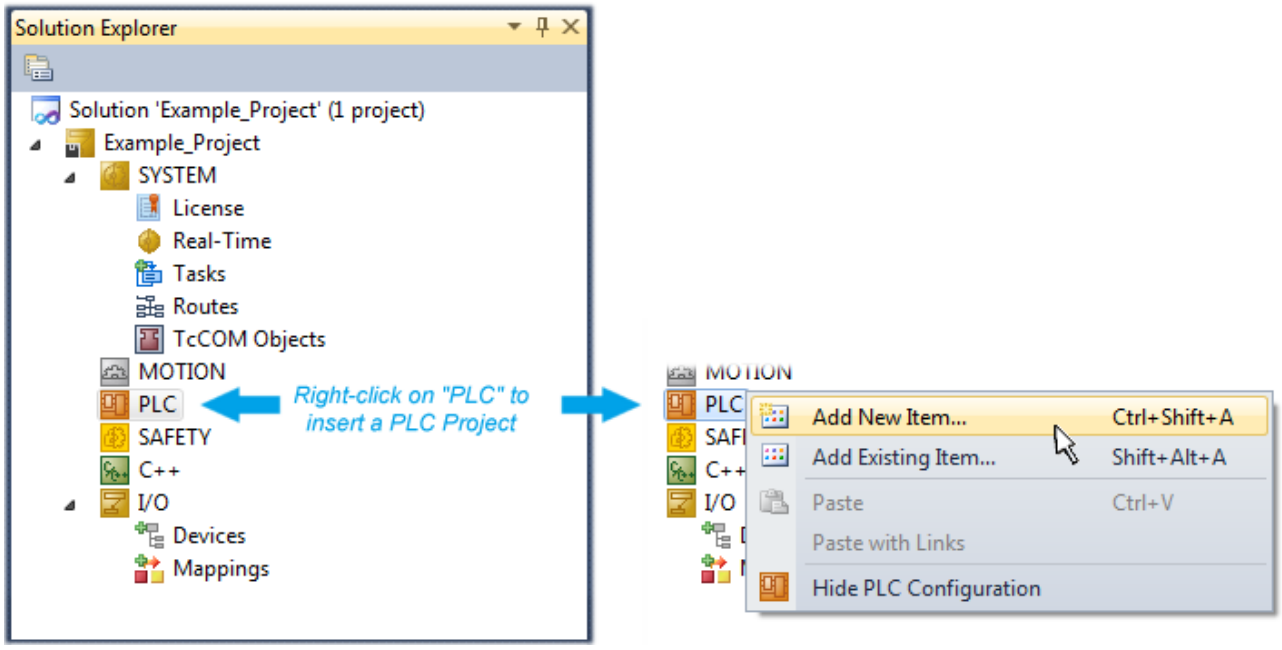


Fig. 64: Adding the programming environment in “PLC”

In the dialog that opens, select “Standard PLC project” and enter “PLC_example” as project name, for example, and select a corresponding directory:

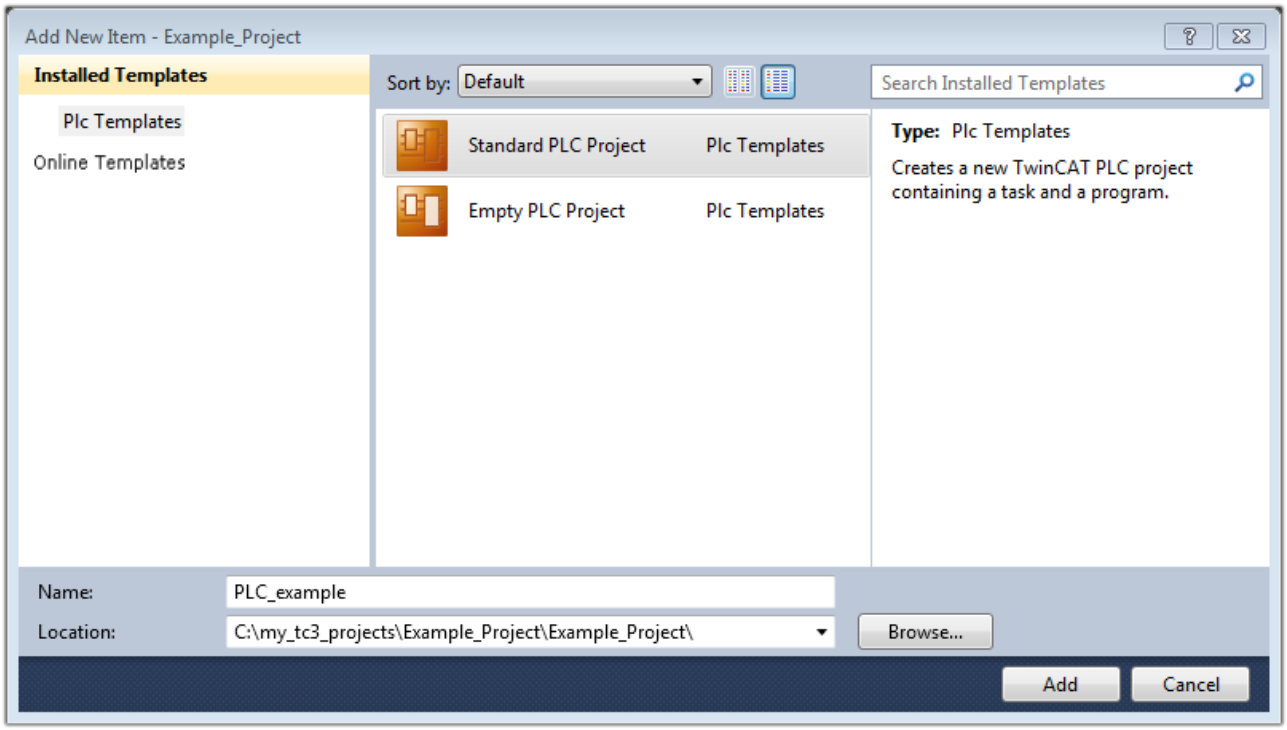


Fig. 65: Specifying the name and directory for the PLC programming environment

The “Main” program, which already exists due to selecting “Standard PLC project”, can be opened by double-clicking on “PLC_example_project” in “POUs”. The following user interface is shown for an initial project:

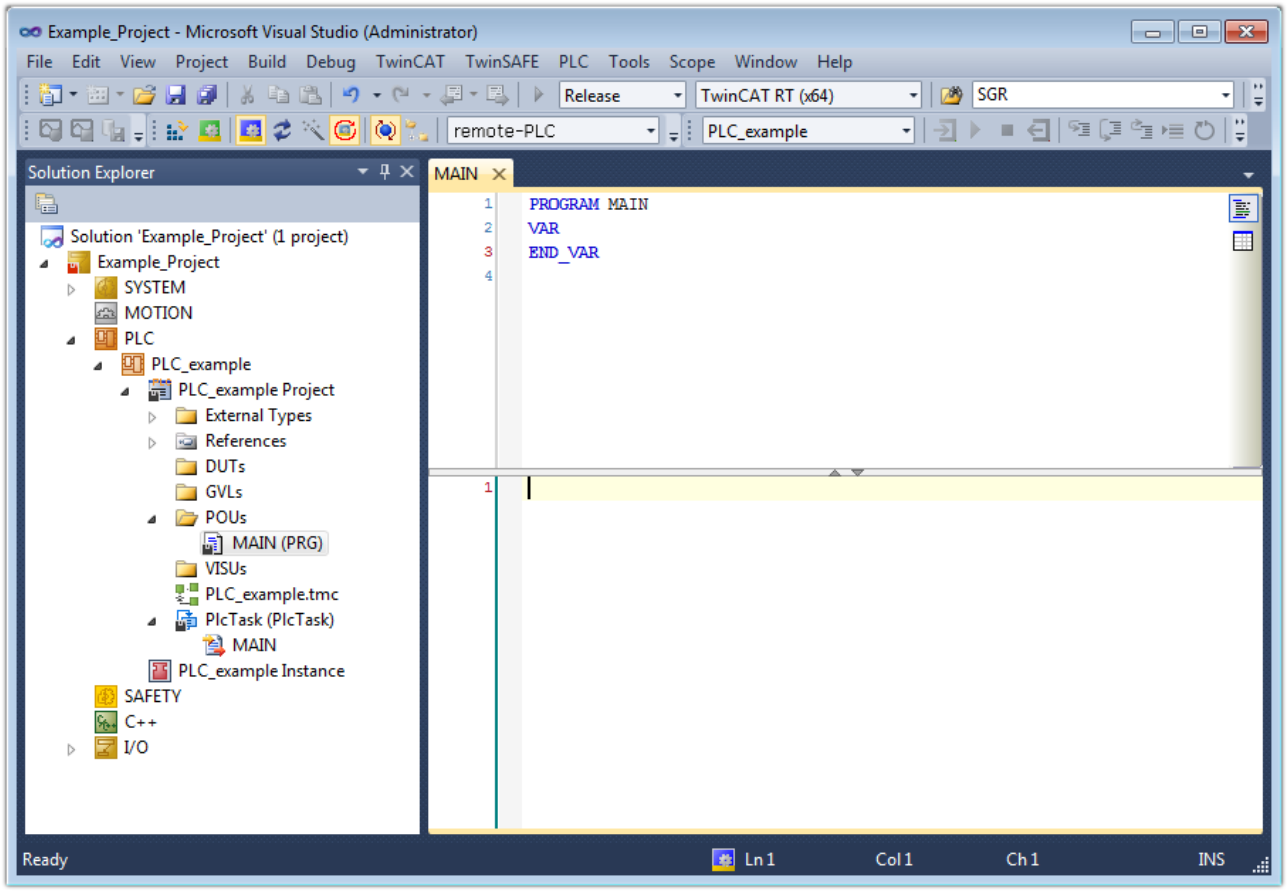


Fig. 66: Initial “Main” program for the standard PLC project

Now example variables and an example program have been created for the next stage of the process:

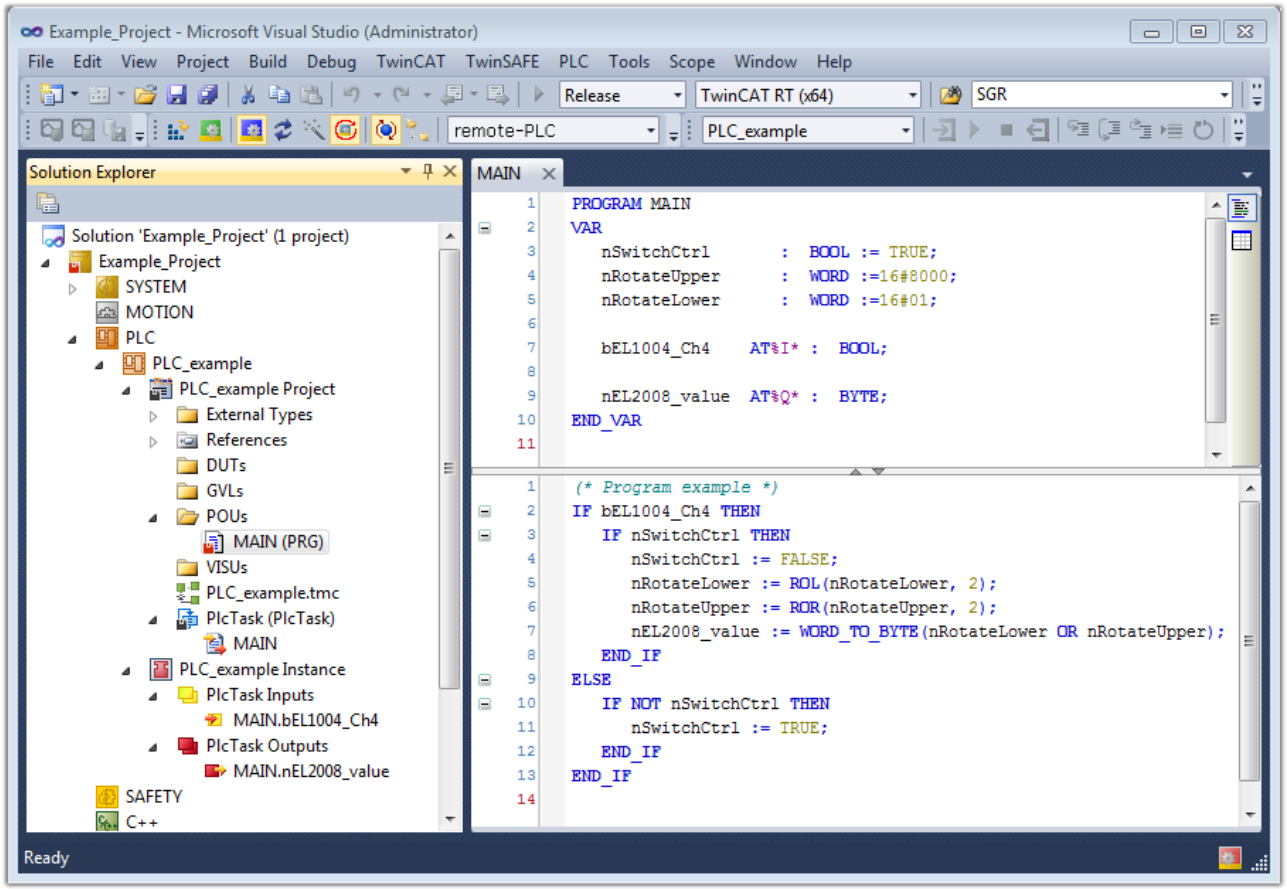


Fig. 67: Example program with variables after a compile process (without variable integration)

The control program is now created as a project folder, followed by the compile process:

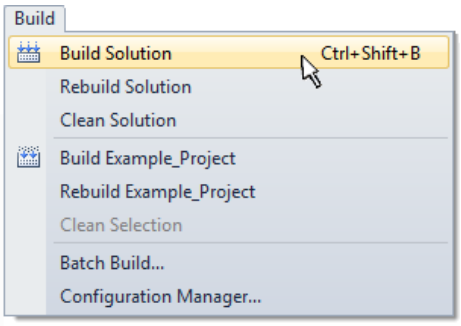
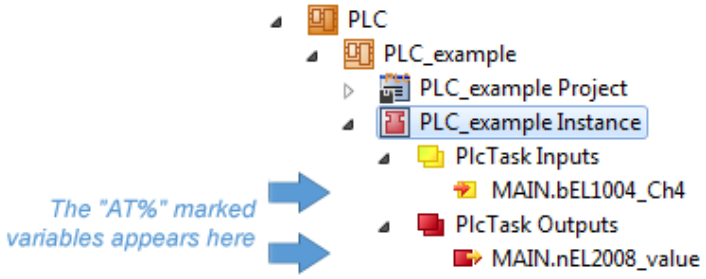


Fig. 68: Start program compilation

The following variables, identified in the ST/PLC program with “AT%”, are then available under “Assignments” in the project folder explorer:



Assigning variables

Via the menu of an instance – variables in the “PLC” context, use the “Modify Link…” option to open a window to select a suitable process object (PDO) for linking:

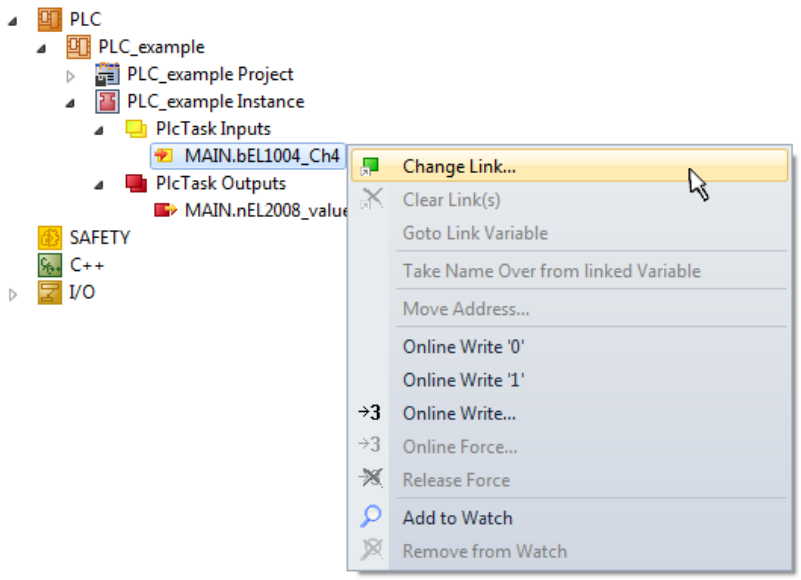


Fig. 69: Creating the links between PLC variables and process objects

In the window that opens, the process object for the “bEL1004_Ch4” BOOL-type variable can be selected from the PLC configuration tree:

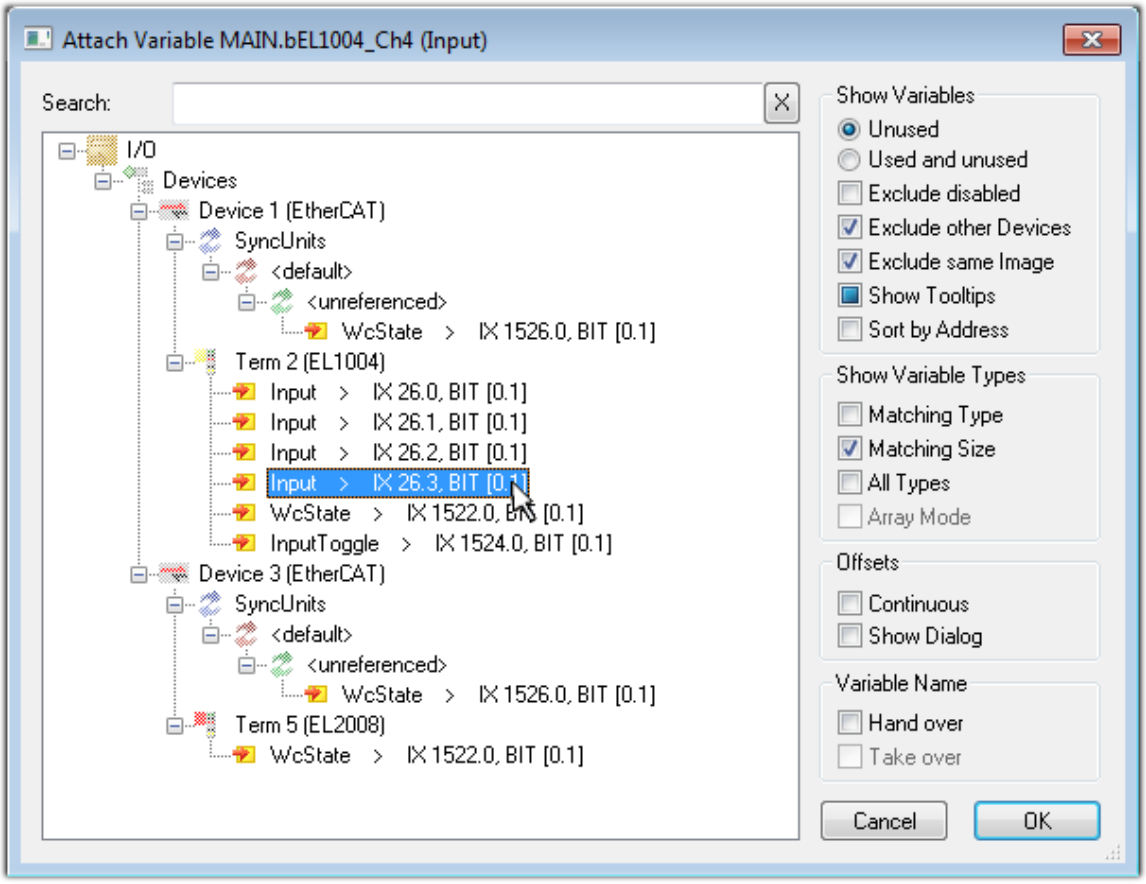


Fig. 70: Selecting BOOL-type PDO

According to the default setting, only certain PDO objects are now available for selection. In this example, the input of channel 4 of the EL1004 terminal is selected for linking. In contrast, the checkbox “All types” must be ticked to create the link for the output variables, in order to allocate a set of eight separate output bits to a byte variable in this case. The following diagram shows the whole process:

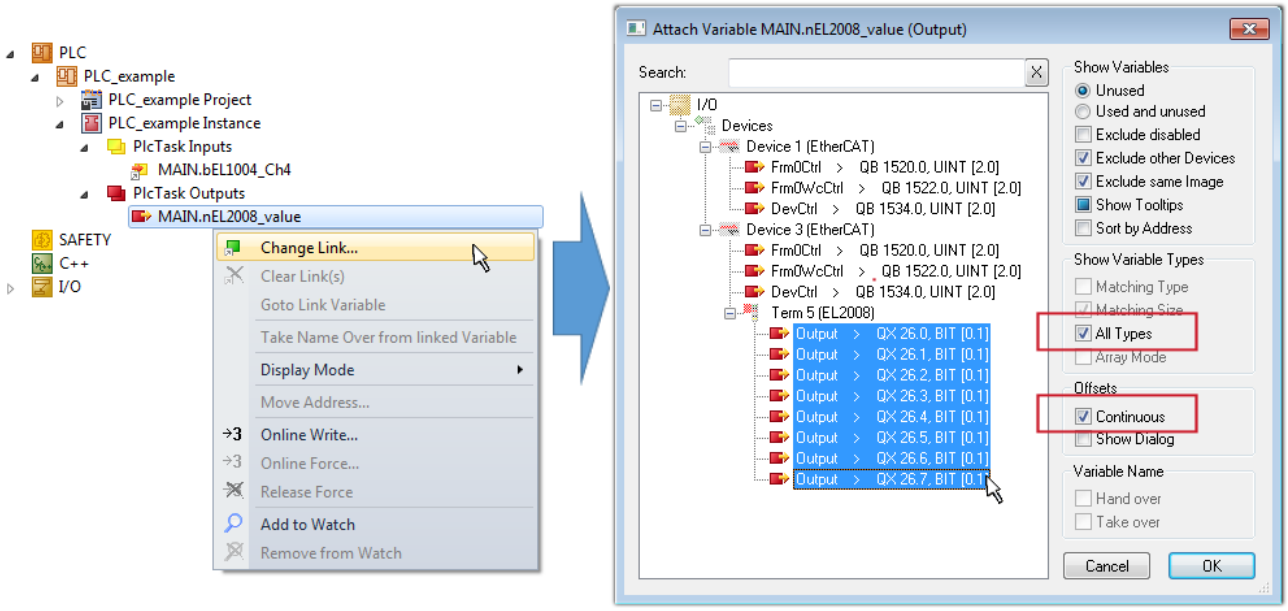


Fig. 71: Selecting several PDOs simultaneously: activate “Continuous” and “All types”

Note that the “Continuous” checkbox was also activated. This is designed to allocate the bits contained in the byte of the “nEL2008_value” variable sequentially to all eight selected output bits of the EL2008 Terminal. It is thus possible to subsequently address all eight outputs of the terminal in the program with a byte corresponding to bit 0 for channel 1 to bit 7 for channel 8 of the PLC. A special symbol ( ) on the yellow or red object of the variable indicates that a link exists. The links can also be checked by selecting “Goto Link Variable” from the context menu of a variable. The opposite linked object, in this case the PDO, is automatically selected:

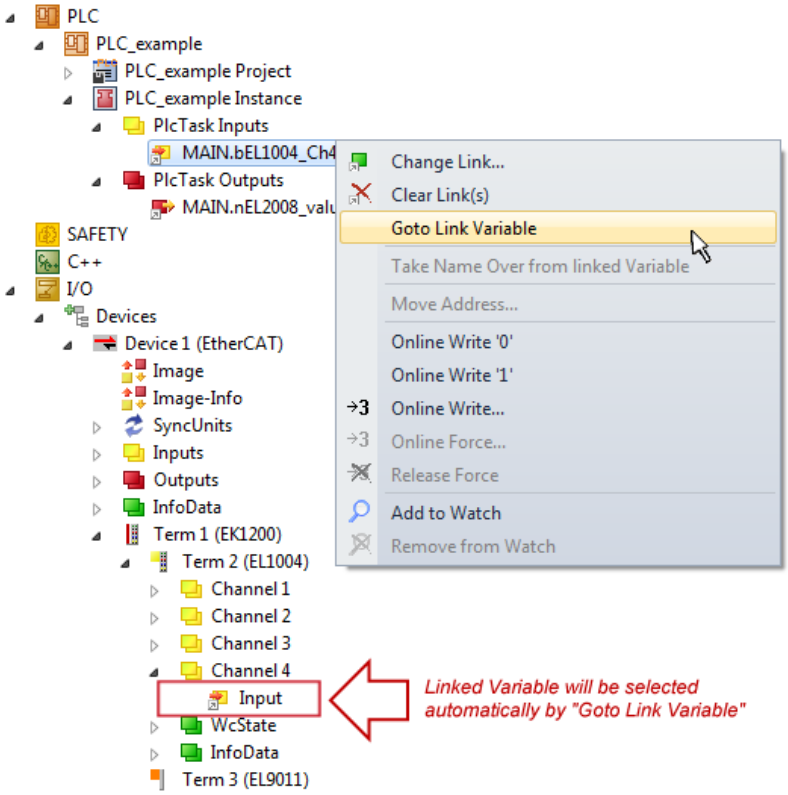


Fig. 72: Application of a “Goto Link Variable”, using “MAIN.bEL1004_Ch4” as an example

The process of creating links can also be performed in the opposite direction, i.e. starting with individual PDOs to a variable. However, in this example, it would not be possible to select all output bits for the EL2008, since the terminal only makes individual digital outputs available. If a terminal has a byte, word,

integer or similar PDO, it is also possible to allocate this to a set of bit-standardized variables. Here, too, a “Goto Link Variable” can be executed in the other direction, so that the respective PLC instance can then be selected.

Note on type of variable assignment

The following type of variable assignment can only be used from TwinCAT version V3.1.4024.4 onwards and is only available for terminals with a microcontroller.

In TwinCAT, a structure can be created from the mapped process data of a terminal. An instance of this structure can then be created in the PLC, so it is possible to access the process data directly from the PLC without having to declare own variables.

The procedure for the EL3001 1-channel analog input terminal -10...+10 V is shown as an example.

1. First, the required process data must be selected in the “Process data” tab in TwinCAT.
2. After that, the PLC data type must be generated in the “PLC” tab via the check box.
3. The data type in the “Data Type” field can then be copied using the “Copy” button.

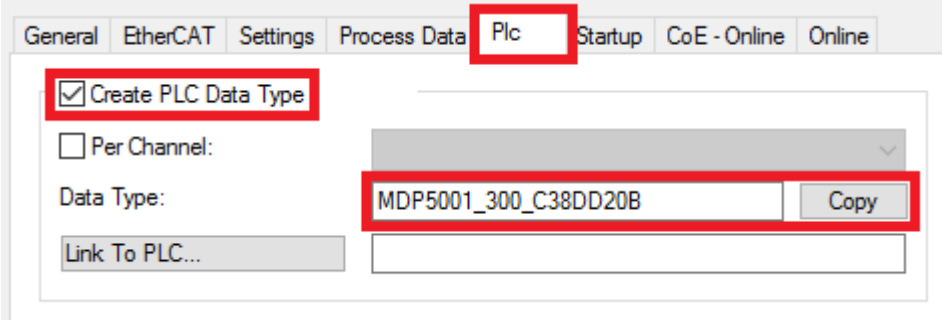


Fig. 73: Creating a PLC data type

4. An instance of the data structure of the copied data type must then be created in the PLC.

```
MAIN
1  PROGRAM MAIN
2  VAR
3      EL3001  : MDP5001_300_C38DD20B;
4  END_VAR
```

Fig. 74: Instance_of_struct

5. Then the project folder must be created. This can be done either via the key combination “CTRL + Shift + B” or via the “Build” tab in TwinCAT.
6. The structure in the “PLC” tab of the terminal must then be linked to the created instance.

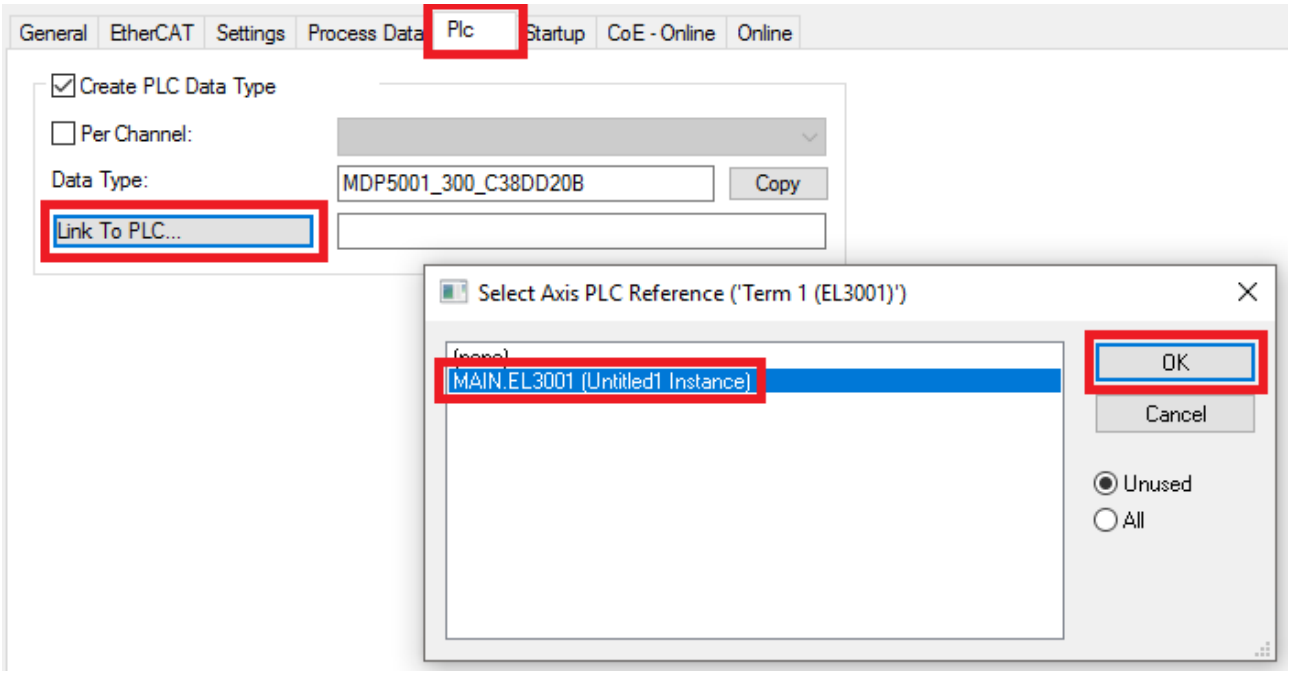


Fig. 75: Linking the structure

7. In the PLC, the process data can then be read or written via the structure in the program code.

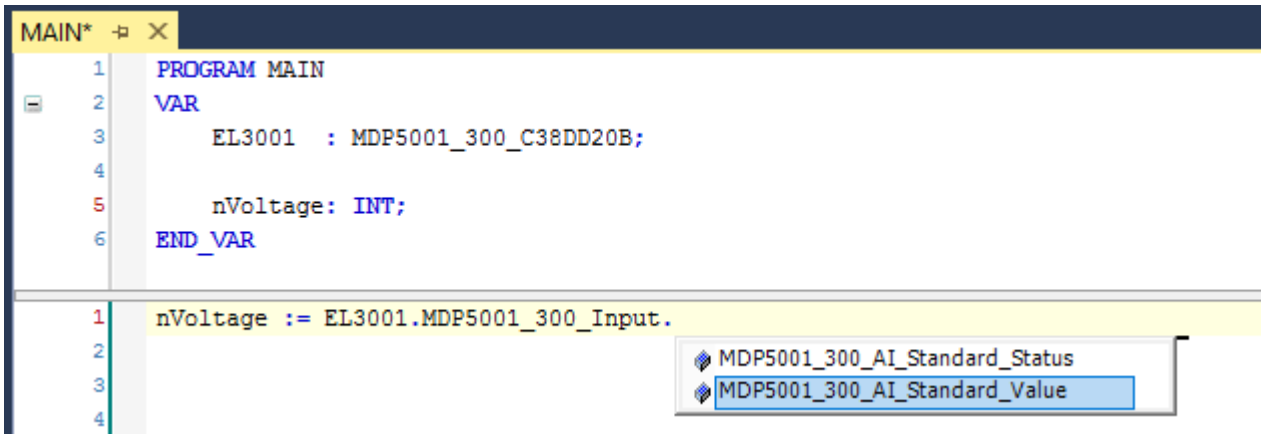
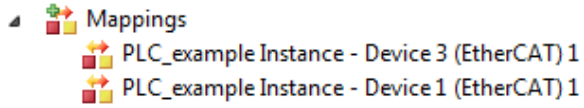


Fig. 76: Reading a variable from the structure of the process data

Activation of the configuration

The allocation of PDO to PLC variables has now established the connection from the controller to the inputs and outputs of the terminals. The configuration can now be activated with  or via the menu under “TwinCAT” in order to transfer the settings of the development environment to the runtime system. Confirm the messages “Old configurations will be overwritten!” and “Restart TwinCAT system in Run mode” with “OK”. The corresponding assignments can be seen in the project folder explorer:



A few seconds later, the corresponding status of the Run mode is displayed in the form of a rotating symbol  at the bottom right of the VS shell development environment. The PLC system can then be started as described below.

Starting the controller

Select the menu option “PLC” → “Login” or click on to link the PLC with the real-time system and load the control program for execution. This results in the message ”*No program on the controller! Should the new program be loaded?*”, which should be acknowledged with “Yes”. The runtime environment is ready for the program to be started by clicking on symbol , the “F5” key or via “PLC” in the menu, by selecting “Start”. The started programming environment shows the runtime values of individual variables:

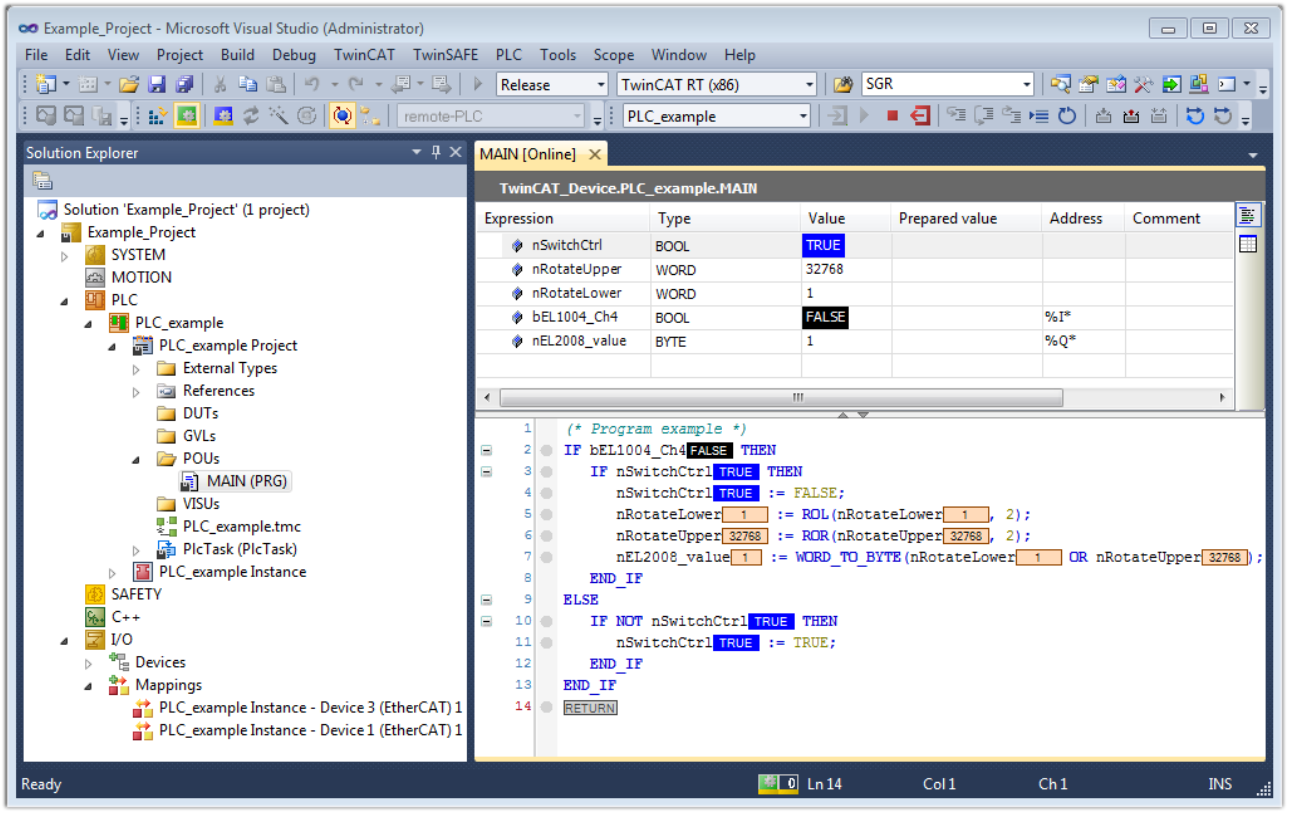


Fig. 77: TwinCAT 3 development environment (VS shell): logged-in, after program startup

The two operator control elements for stopping and logout result in the required action (also, “Shift + F5” can be used for stop, or both actions can be selected via the PLC menu).

5.1.2 TwinCAT Development Environment

The Software for automation TwinCAT (The Windows Control and Automation Technology) will be distinguished into:

- TwinCAT 2: System Manager (Configuration) and PLC Control (Programming)
- TwinCAT 3: Enhancement of TwinCAT 2 (Programming and Configuration takes place via a common Development Environment)

Details:

- **TwinCAT 2:**
  - Connects I/O devices to tasks in a variable-oriented manner
  - Connects tasks to tasks in a variable-oriented manner
  - Supports units at the bit level
  - Supports synchronous or asynchronous relationships
  - Exchange of consistent data areas and process images
  - Datalink on NT - Programs by open Microsoft Standards (OLE, OCX, ActiveX, DCOM+, etc.)
  - Integration of IEC 61131-3-Software-SPS, Software- NC and Software-CNC within Windows NT/ 2000/XP/Vista, Windows 7, NT/XP Embedded, CE
  - Interconnection to all common fieldbusses
  - More...

Additional features:

- **TwinCAT 3** (eXtended Automation)**:**
  - Visual Studio® integration
  - Choice of the programming language
  - Supports object orientated extension of IEC 61131-3
  - Usage of C/C++ as programming language for real time applications
  - Connection to MATLAB®/Simulink®
  - Open interface for expandability
  - Flexible run-time environment
  - Active support of multi-core- and 64 bit operating system
  - Automatic code generation and project creation with the TwinCAT Automation Interface
  - More...

Within the following sections commissioning of the TwinCAT Development Environment on a PC System for the control and also the basically functions of unique control elements will be explained.

Please see further information to TwinCAT 2 and TwinCAT 3 at http://infosys.beckhoff.com.

5.1.2.1 Installation of the TwinCAT real-time driver

In order to assign real-time capability to a standard Ethernet port of an IPC controller, the Beckhoff real-time driver has to be installed on this port under Windows.

This can be done in several ways.

A: Via the TwinCAT Adapter dialog

In the System Manager call up the TwinCAT overview of the local network interfaces via Options → Show Real Time Ethernet Compatible Devices.

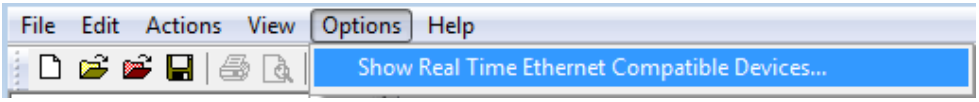


Fig. 78: System Manager “Options” (TwinCAT 2)

This have to be called up by the menu “TwinCAT” within the TwinCAT 3 environment:

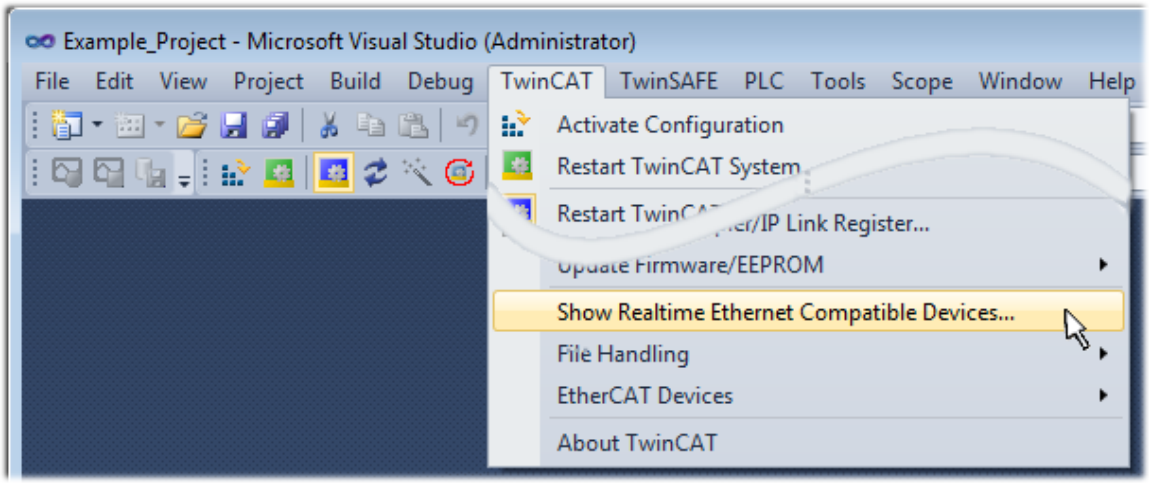


Fig. 79: Call up under VS Shell (TwinCAT 3)

B: Via TcRteInstall.exe in the TwinCAT directory

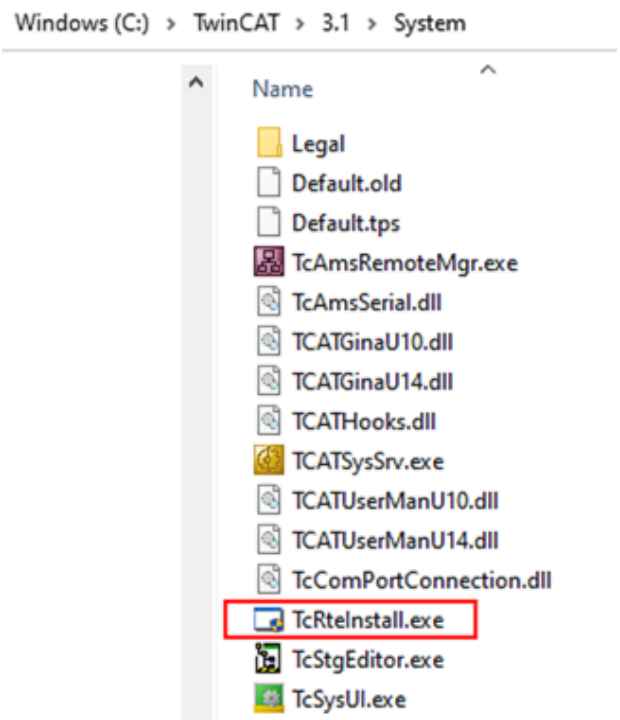


Fig. 80: TcRteInstall in the TwinCAT directory

In both cases, the following dialog appears:

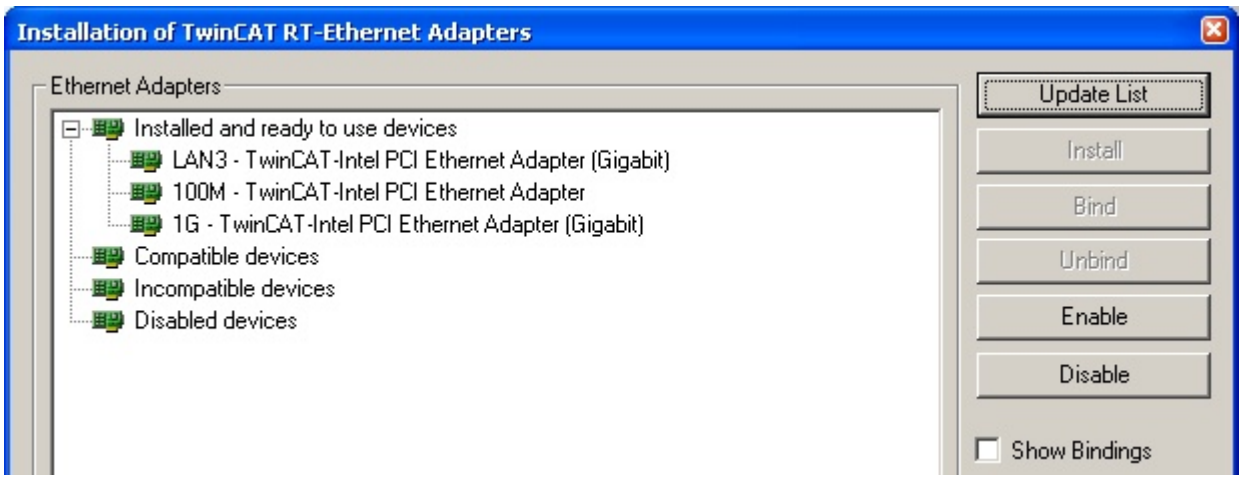


Fig. 81: Overview of network interfaces

Interfaces listed under “Compatible devices” can be assigned a driver via the “Install” button. A driver should only be installed on compatible devices.

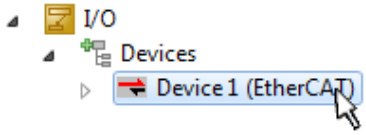
A Windows warning regarding the unsigned driver can be ignored.

Alternatively an EtherCAT-device can be inserted first of all as described in chapter Offline configuration creation, section “Creating the EtherCAT device” [▶ 94] in order to view the compatible ethernet ports via its EtherCAT properties (tab “Adapter”, button “Compatible Devices…”):



Fig. 82: EtherCAT device properties (TwinCAT 2): click on “Compatible Devices…” of tab “Adapter”

TwinCAT 3: the properties of the EtherCAT device can be opened by double click on “Device .. (EtherCAT)” within the Solution Explorer under “I/O”:



After the installation the driver appears activated in the Windows overview for the network interface (Windows Start → System Properties → Network)

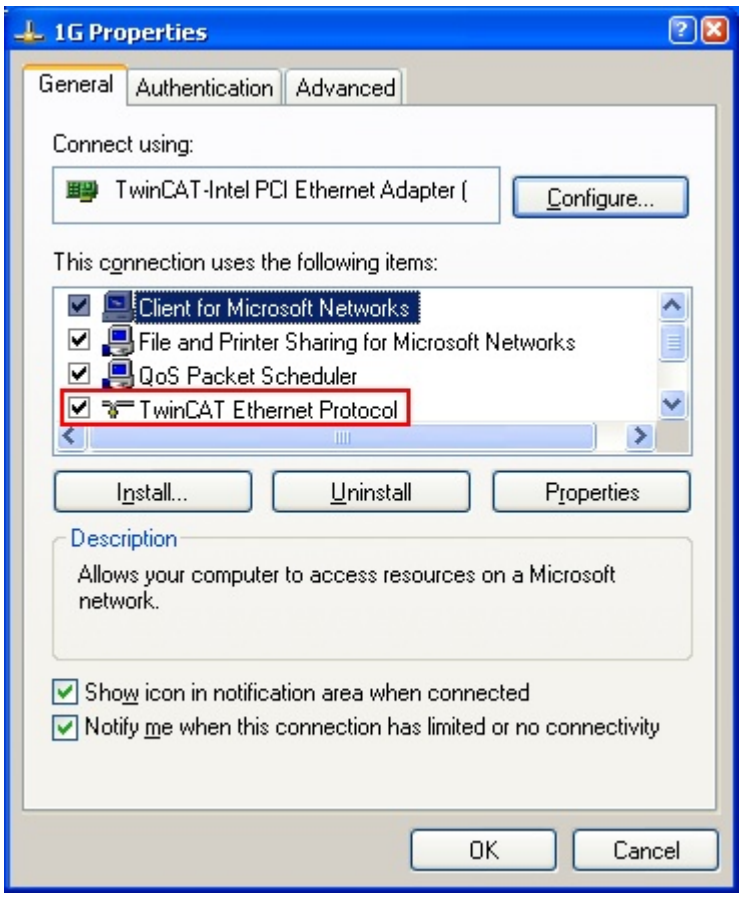


Fig. 83: Windows properties of the network interface

A correct setting of the driver could be:

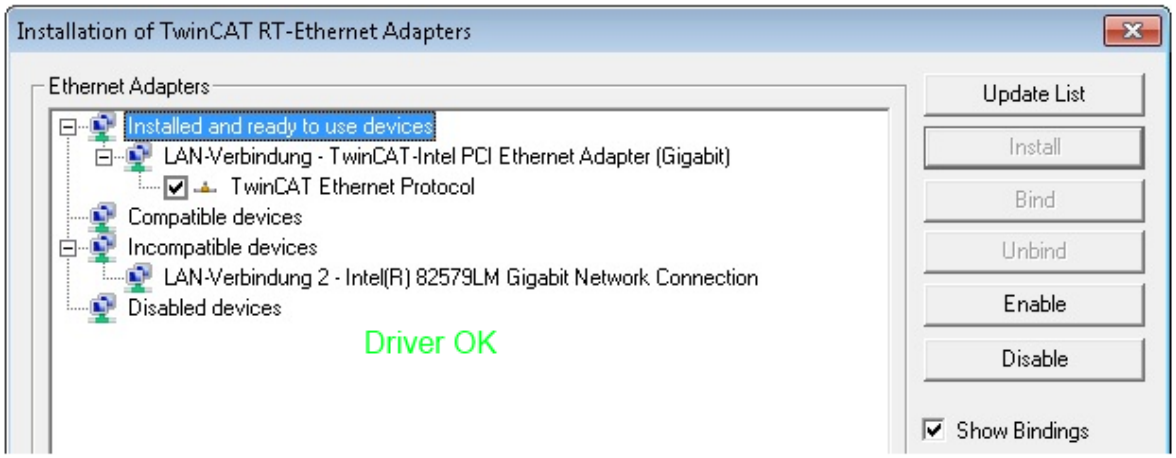
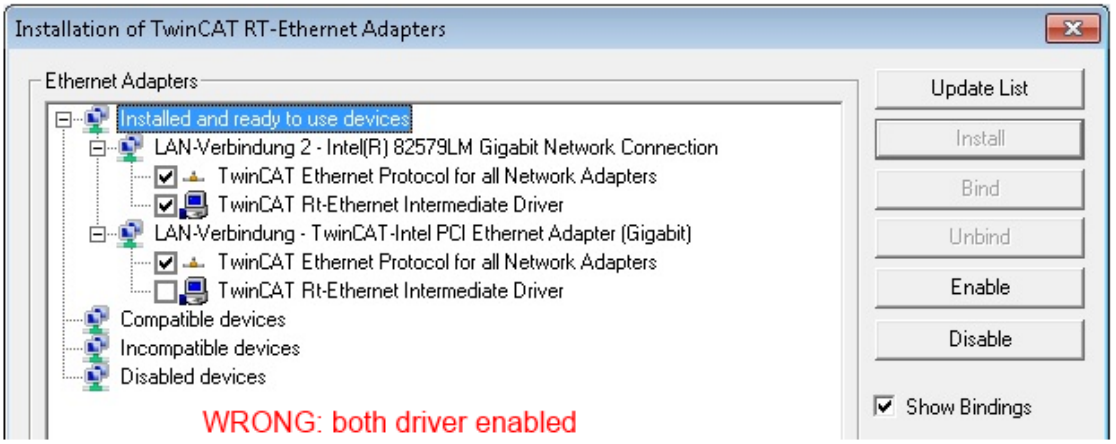
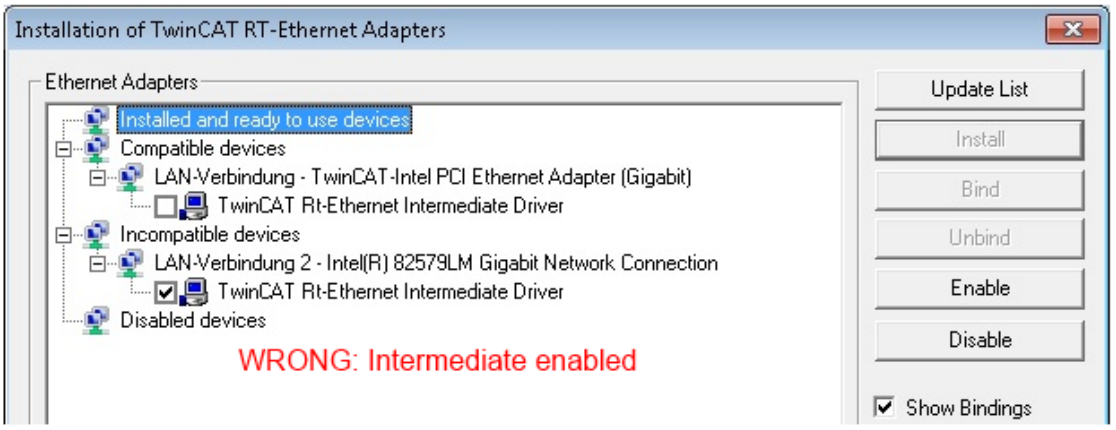
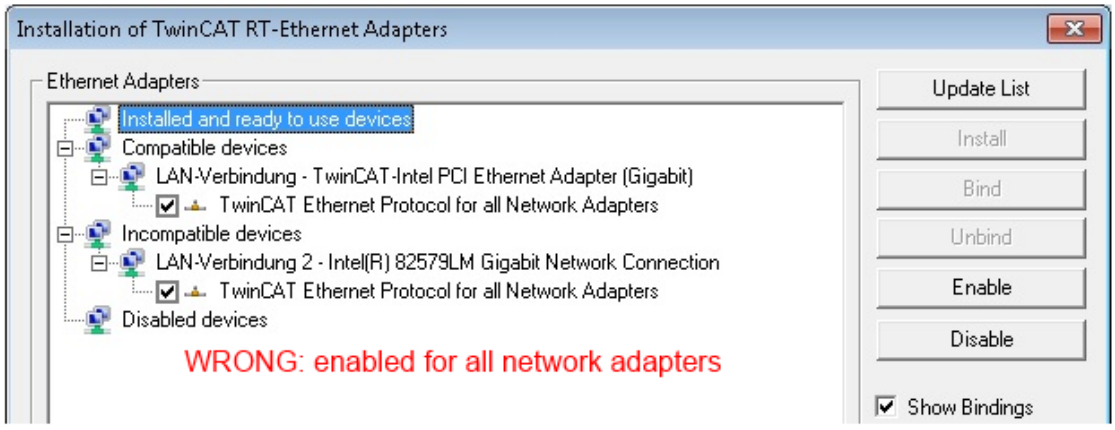


Fig. 84: Exemplary correct driver setting for the Ethernet port

Other possible settings have to be avoided:







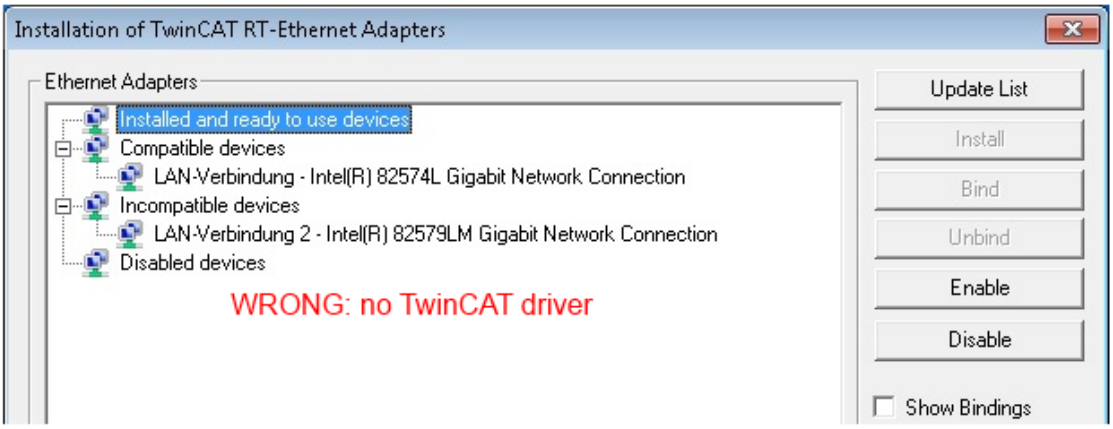


Fig. 85: Incorrect driver settings for the Ethernet port

IP address of the port used

IP address/DHCP

In most cases an Ethernet port that is configured as an EtherCAT device will not transport general IP packets. For this reason and in cases where an EL6601 or similar devices are used it is useful to specify a fixed IP address for this port via the “Internet Protocol TCP/IP” driver setting and to disable DHCP. In this way the delay associated with the DHCP client for the Ethernet port assigning itself a default IP address in the absence of a DHCP server is avoided. A suitable address space is 192.168.x.x, for example.

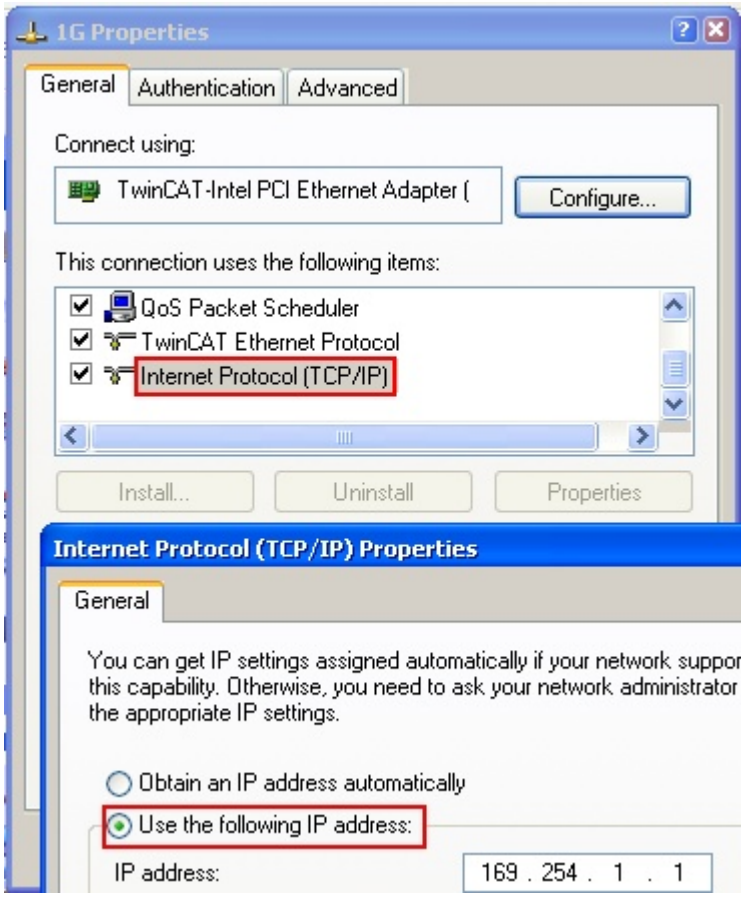


Fig. 86: TCP/IP setting for the Ethernet port

5.1.2.2 Notes regarding ESI device description

Installation of the latest ESI device description

The TwinCAT EtherCAT master/System Manager needs the device description files for the devices to be used in order to generate the configuration in online or offline mode. The device descriptions are contained in the so-called ESI files (EtherCAT Slave Information) in XML format. These files can be requested from the respective manufacturer and are made available for download. An *.xml file may contain several device descriptions.

The ESI files for Beckhoff EtherCAT devices are available on the Beckhoff website.

The ESI files should be stored in the TwinCAT installation directory.

Default settings:

- **TwinCAT 2**: C:\TwinCAT\IO\EtherCAT
- **TwinCAT 3**: C:\TwinCAT\3.1\Config\Io\EtherCAT

The files are read (once) when a new System Manager window is opened, if they have changed since the last time the System Manager window was opened.

A TwinCAT installation includes the set of Beckhoff ESI files that was current at the time when the TwinCAT build was created.

For TwinCAT 2.11/TwinCAT 3 and higher, the ESI directory can be updated from the System Manager, if the programming PC is connected to the Internet; by

- **TwinCAT 2**: Option → “Update EtherCAT Device Descriptions”
- **TwinCAT 3**: TwinCAT → EtherCAT Devices → “Update Device Descriptions (via ETG Website)…”

The TwinCAT ESI Updater [▶ 92] is available for this purpose.

ESI

The *.xml files are associated with *.xsd files, which describe the structure of the ESI XML files. To update the ESI device descriptions, both file types should therefore be updated.

Device differentiation

EtherCAT devices/slaves are distinguished by four properties, which determine the full device identifier. For example, the device identifier EL2521-0025-1018 consists of:

- family key “EL”
- name “2521”
- type “0025”
- and revision “1018”

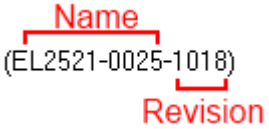


Fig. 87: Identifier structure

The order identifier consisting of name + type (here: EL2521-0025) describes the device function. The revision indicates the technical progress and is managed by Beckhoff. In principle, a device with a higher revision can replace a device with a lower revision, unless specified otherwise, e.g. in the documentation. Each revision has its own ESI description. See further notes [▶ 10].

Online description

If the EtherCAT configuration is created online through scanning of real devices (see section Online setup) and no ESI descriptions are available for a slave (specified by name and revision) that was found, the System Manager asks whether the description stored in the device should be used. In any case, the System Manager needs this information for setting up the cyclic and acyclic communication with the slave correctly.

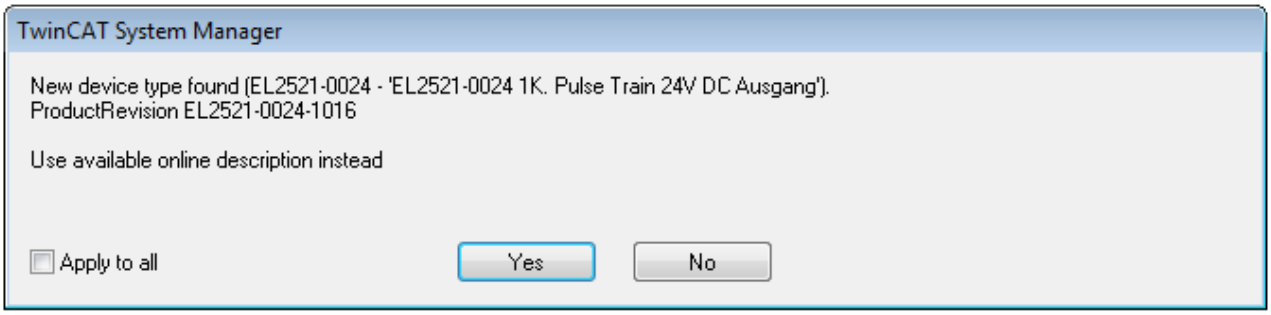


Fig. 88: OnlineDescription information window (TwinCAT 2)

In TwinCAT 3 a similar window appears, which also offers the Web update:

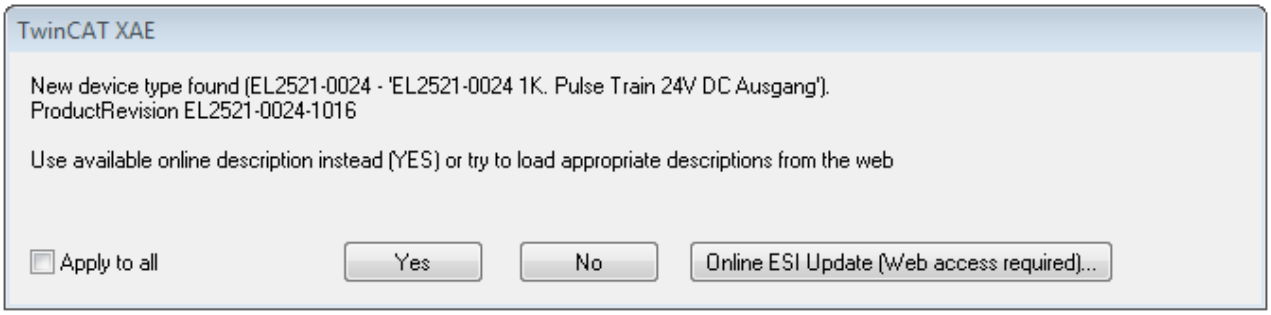


Fig. 89: Information window OnlineDescription (TwinCAT 3)

If possible, the *Yes* is to be rejected and the required ESI is to be requested from the device manufacturer. After installation of the XML/XSD file the configuration process should be repeated.

NOTICE

Changing the "usual" configuration through a scan

✓ If a scan discovers a device that is not yet known to TwinCAT, distinction has to be made between two cases. Taking the example here of the EL2521-0000 in the revision 1019

a) no ESI is present for the EL2521-0000 device at all, either for the revision 1019 or for an older revision. The ESI must then be requested from the manufacturer (in this case Beckhoff).

b) an ESI is present for the EL2521-0000 device, but only in an older revision, e.g. 1018 or 1017. In this case an in-house check should first be performed to determine whether the spare parts stock allows the integration of the increased revision into the configuration at all. A new/higher revision usually also brings along new features. If these are not to be used, work can continue without reservations with the previous revision 1018 in the configuration. This is also stated by the Beckhoff compatibility rule.

Refer in particular to the chapter "General notes on the use of Beckhoff EtherCAT IO components" and for manual configuration to the chapter "Offline configuration creation [▶ 94]".

If the OnlineDescription is used regardless, the System Manager reads a copy of the device description from the EEPROM in the EtherCAT slave. In complex slaves the size of the EEPROM may not be sufficient for the complete ESI, in which case the ESI would be *incomplete* in the configurator. Therefore it's recommended using an offline ESI file with priority in such a case.

The System Manager creates for online recorded device descriptions a new file "OnlineDescription0000...xml" in its ESI directory, which contains all ESI descriptions that were read online.

OnlineDescriptionCache00000002.xml

Fig. 90: File OnlineDescription.xml created by the System Manager

Is a slave desired to be added manually to the configuration at a later stage, online created slaves are indicated by a prepended symbol “>” in the selection list (see Figure *Indication of an online recorded ESI of EL2521 as an example)*.

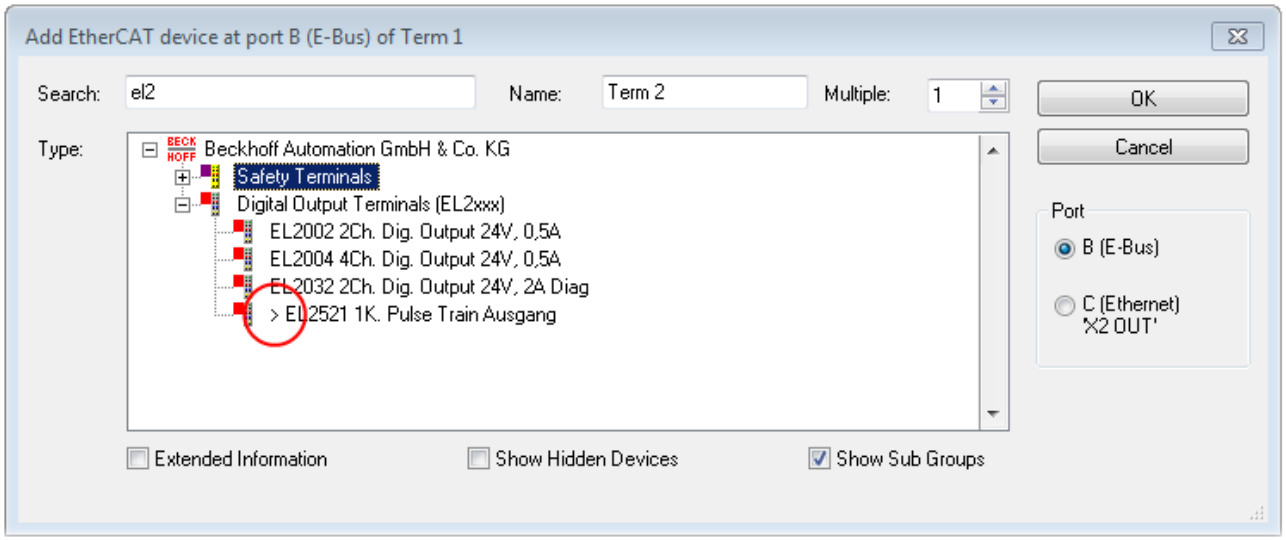


Fig. 91: Indication of an online recorded ESI of EL2521 as an example

If such ESI files are used and the manufacturer's files become available later, the file OnlineDescription.xml should be deleted as follows:

- close all System Manager windows
- restart TwinCAT in Config mode
- delete “OnlineDescription0000...xml”
- restart TwinCAT System Manager

This file should not be visible after this procedure, if necessary press <F5> to update

OnlineDescription for TwinCAT 3.x

In addition to the file described above “OnlineDescription0000...xml”, a so called EtherCAT cache with new discovered devices is created by TwinCAT 3.x, e.g. under Windows 7:

C:\User\[USERNAME]\AppData\Roaming\Beckhoff\TwinCAT3\Components\Base\EtherCATCache.xml

(Please note the language settings of the OS!)
You have to delete this file, too.

Faulty ESI file

If an ESI file is faulty and the System Manager is unable to read it, the System Manager brings up an information window.

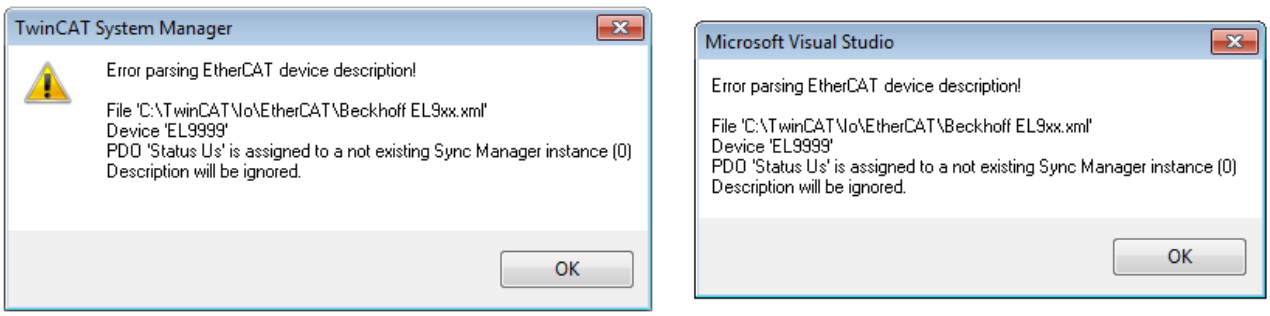


Fig. 92: Information window for faulty ESI file (left: TwinCAT 2; right: TwinCAT 3)

Reasons may include:

- Structure of the *.xml does not correspond to the associated *.xsd file → check your schematics
- Contents cannot be translated into a device description → contact the file manufacturer

5.1.2.3 TwinCAT ESI Updater

The ESI Updater is a convenient option for automatic downloading of ESI data provided by EtherCAT manufacturers via the Internet into the TwinCAT directory (ESI = EtherCAT slave information). TwinCAT accesses the central ESI ULR directory list stored at ETG; the entries can then be viewed in the Updater dialog, although they cannot be changed there.

5.1.2.3.1 TwinCAT 3

Step 1) Updating the ESI data sets

The Updater can be accessed under:
“TwinCAT” → “EtherCAT Devices” → “Update Device Description (via ETG Website)...”

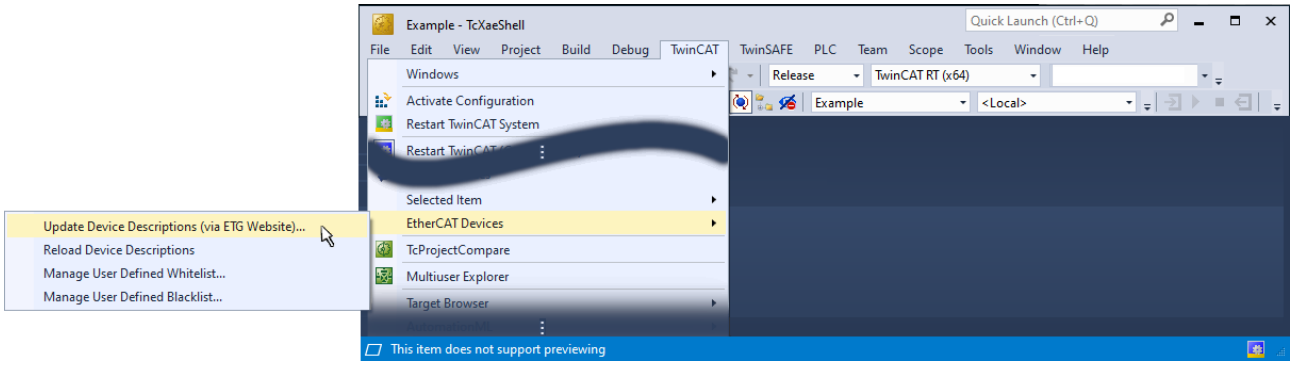
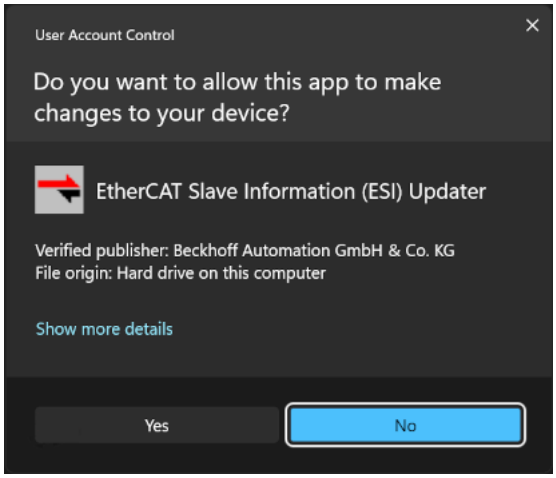


Fig. 93: Calling the ESI Updater (TwinCAT 3)

A prompt for Windows user account control appears after the call, which must be confirmed with [Yes]:



The "Updater" window then appears, showing the configuration data source:

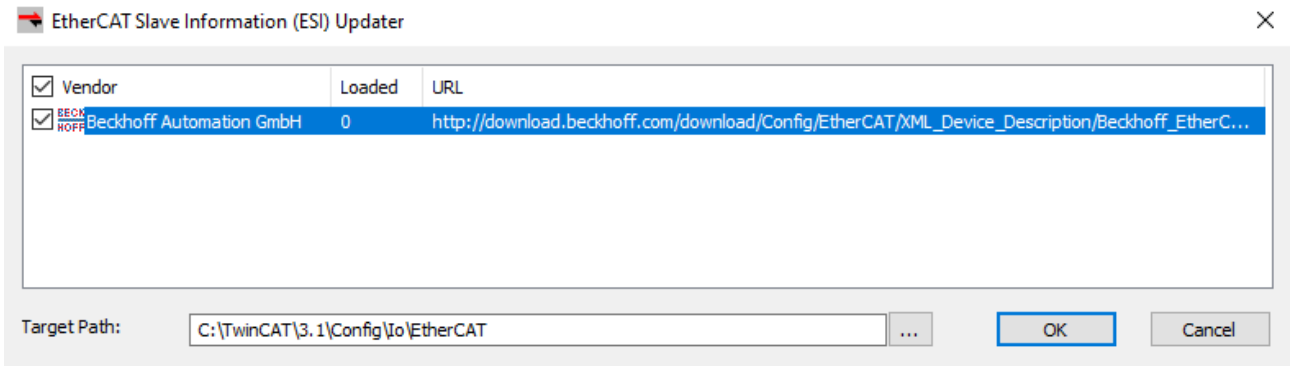
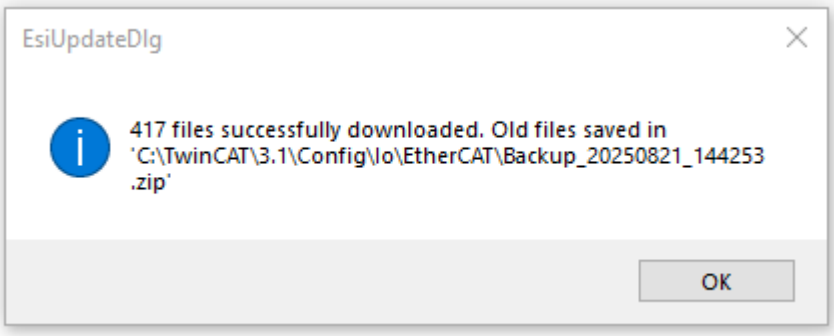


Fig. 94: (ESI) Updater

Select the desired line and click [OK] to start the download or update process. This may take a few minutes.

The completion of the download process is indicated by the message



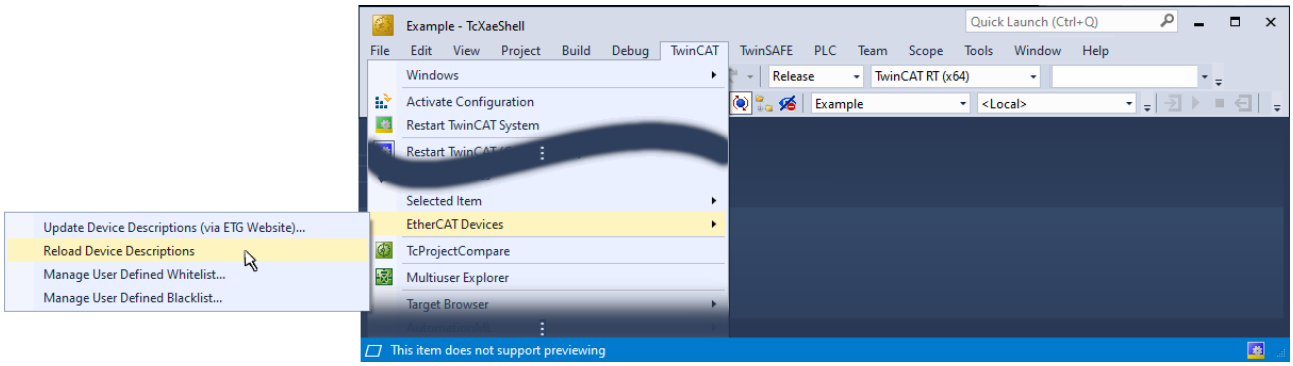
, which shows the number of downloaded files and the storage location or the backup zip folder. Click [OK] to close the window.

Note: Manually added ESI files are not moved to the backup zip folder and remain accessible in TwinCAT. Only the ESI files supplied by Beckhoff are managed automatically.

The (ESI) Updater window that is still open can now be closed with [X] (top right) or [Cancel].

Step 2) Update ESI cache

The ESI files must now be loaded into the application cache in order to be able to use them. This is done either by restarting TwinCAT or by calling up
"TwinCAT" → "EtherCAT Devices" → "Reload Device Descriptions":



The status of the cache rebuild can be seen in the lower status bar in TwinCAT.

5.1.2.3.2 TwinCAT 2

For TwinCAT 2.11 and higher, the System Manager can search for current Beckhoff ESI files automatically, if an online connection is available:

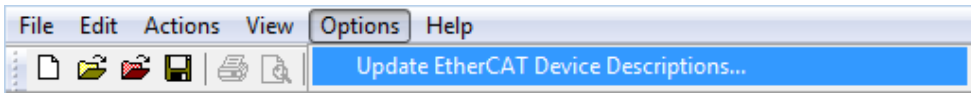


Fig. 95: Using the ESI Updater (>= TwinCAT 2.11)

The call is made via:
“Options” → “Update EtherCAT Device Descriptions”.

5.1.2.4 Distinction between Online and Offline

The distinction between online and offline refers to the presence of the actual I/O environment (drives, terminals, EJ-modules). If the configuration is to be prepared in advance of the system configuration as a programming system, e.g. on a laptop, this is only possible in “Offline configuration” mode. In this case all components have to be entered manually in the configuration, e.g. based on the electrical design.

If the designed control system is already connected to the EtherCAT system and all components are energised and the infrastructure is ready for operation, the TwinCAT configuration can simply be generated through “scanning” from the runtime system. This is referred to as online configuration.

In any case, during each startup the EtherCAT master checks whether the slaves it finds match the configuration. This test can be parameterised in the extended slave settings. Refer to note “Installation of the latest ESI-XML device description” [▶ 88].

For preparation of a configuration:

- the real EtherCAT hardware (devices, couplers, drives) must be present and installed
- the devices/modules must be connected via EtherCAT cables or in the terminal/ module strand in the same way as they are intended to be used later
- the devices/modules be connected to the power supply and ready for communication
- TwinCAT must be in CONFIG mode on the target system.

The online scan process consists of:

- detecting the EtherCAT device [▶ 100] (Ethernet port at the IPC)
- detecting the connected EtherCAT devices [▶ 101]. This step can be carried out independent of the preceding step
- troubleshooting [▶ 104]

The scan with existing configuration [▶ 105] can also be carried out for comparison.

5.1.2.5 OFFLINE configuration creation

Creating the EtherCAT device

Create an EtherCAT device in an empty System Manager window.

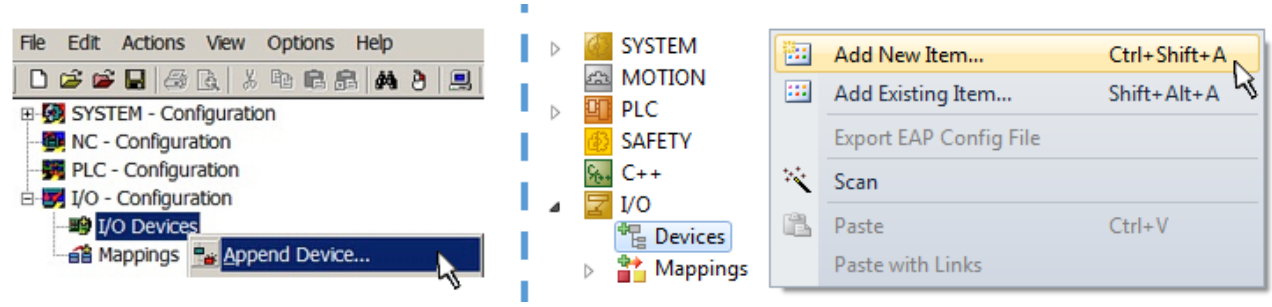


Fig. 96: Append EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

Select type “EtherCAT” for an EtherCAT I/O application with EtherCAT slaves. For the present publisher/subscriber service in combination with an EL6601/EL6614 terminal select “EtherCAT Automation Protocol via EL6601”.

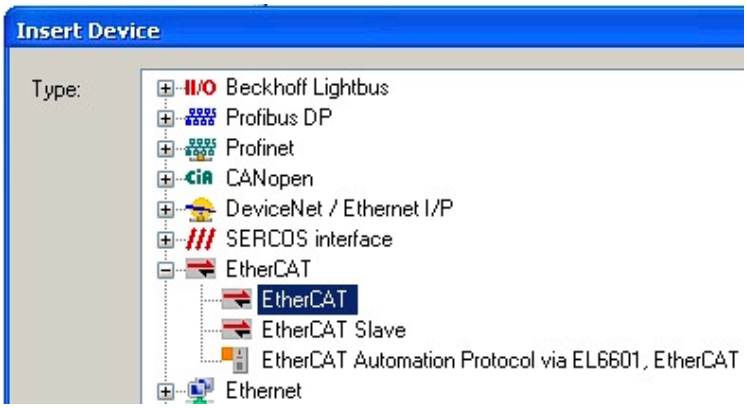


Fig. 97: Selecting the EtherCAT connection (TwinCAT 2.11, TwinCAT 3)

Then assign a real Ethernet port to this virtual device in the runtime system.

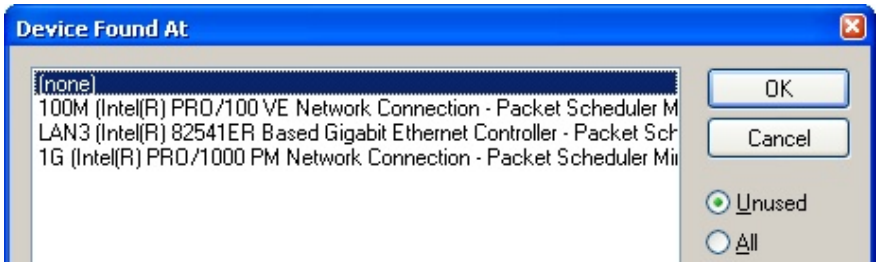


Fig. 98: Selecting the Ethernet port

This query may appear automatically when the EtherCAT device is created, or the assignment can be set/modified later in the properties dialog; see Fig. “EtherCAT device properties (TwinCAT 2)”.

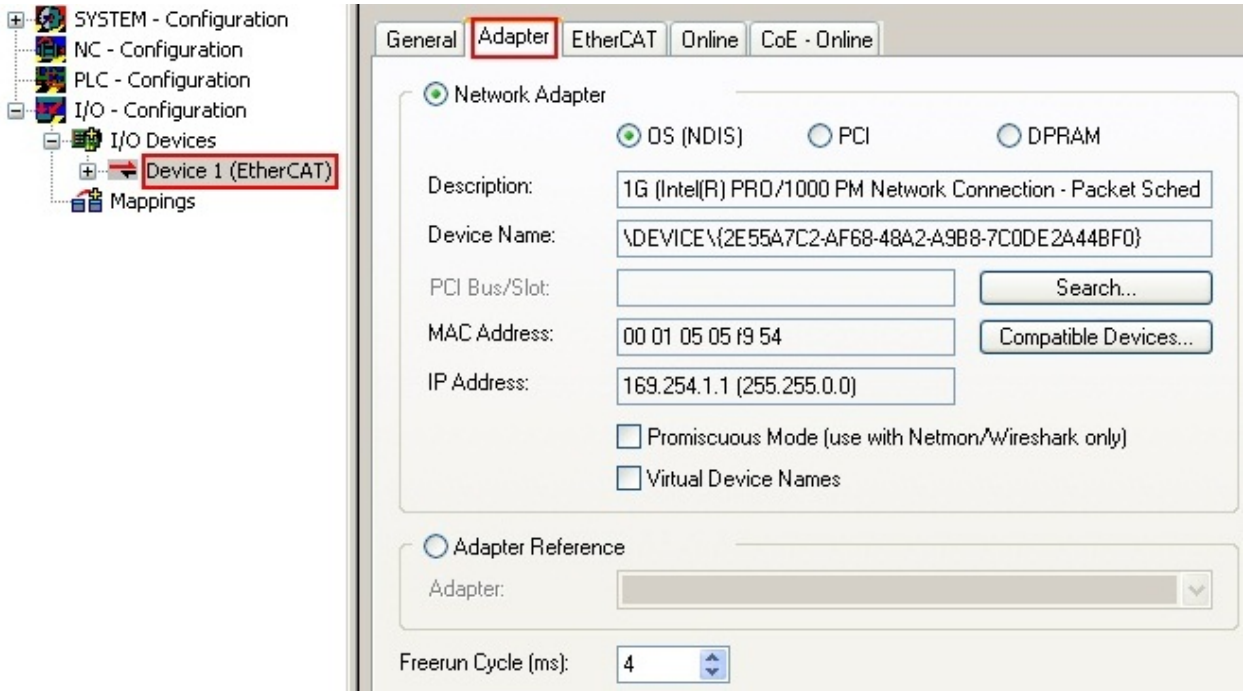
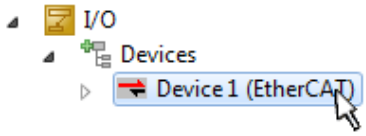


Fig. 99: EtherCAT device properties (TwinCAT 2)

TwinCAT 3: the properties of the EtherCAT device can be opened by double click on “Device .. (EtherCAT)” within the Solution Explorer under “I/O”:



Selecting the Ethernet port

Ethernet ports can only be selected for EtherCAT devices for which the TwinCAT real-time driver is installed. This has to be done separately for each port. Please refer to the respective installation page [▶ 82].

Defining EtherCAT slaves

Further devices can be appended by right-clicking on a device in the configuration tree.

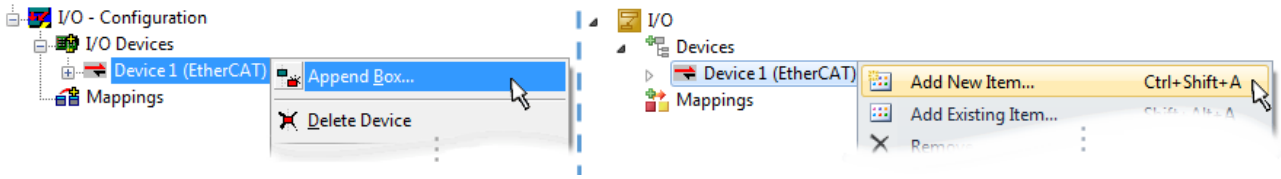


Fig. 100: Appending EtherCAT devices (left: TwinCAT 2; right: TwinCAT 3)

The dialog for selecting a new device opens. Only devices for which ESI files are available are displayed.

Only devices are offered for selection that can be appended to the previously selected device. Therefore, the physical layer available for this port is also displayed (Fig. “Selection dialog for new EtherCAT device”, A). In the case of cable-based Fast-Ethernet physical layer with PHY transfer, then also only cable-based devices are available, as shown in Fig. “Selection dialog for new EtherCAT device”. If the preceding device has several free ports (e.g. EK1122 or EK1100), the required port can be selected on the right-hand side (A).

Overview of physical layer

- “Ethernet”: cable-based 100BASE-TX: couplers, box modules, devices with RJ45/M8/M12 connector
- “E-Bus”: LVDS “terminal bus”, EtherCAT plug-in modules (EJ), EtherCAT terminals (EL/ES), various modular modules

The search field facilitates finding specific devices (from TwinCAT 2.11 or TwinCAT 3).

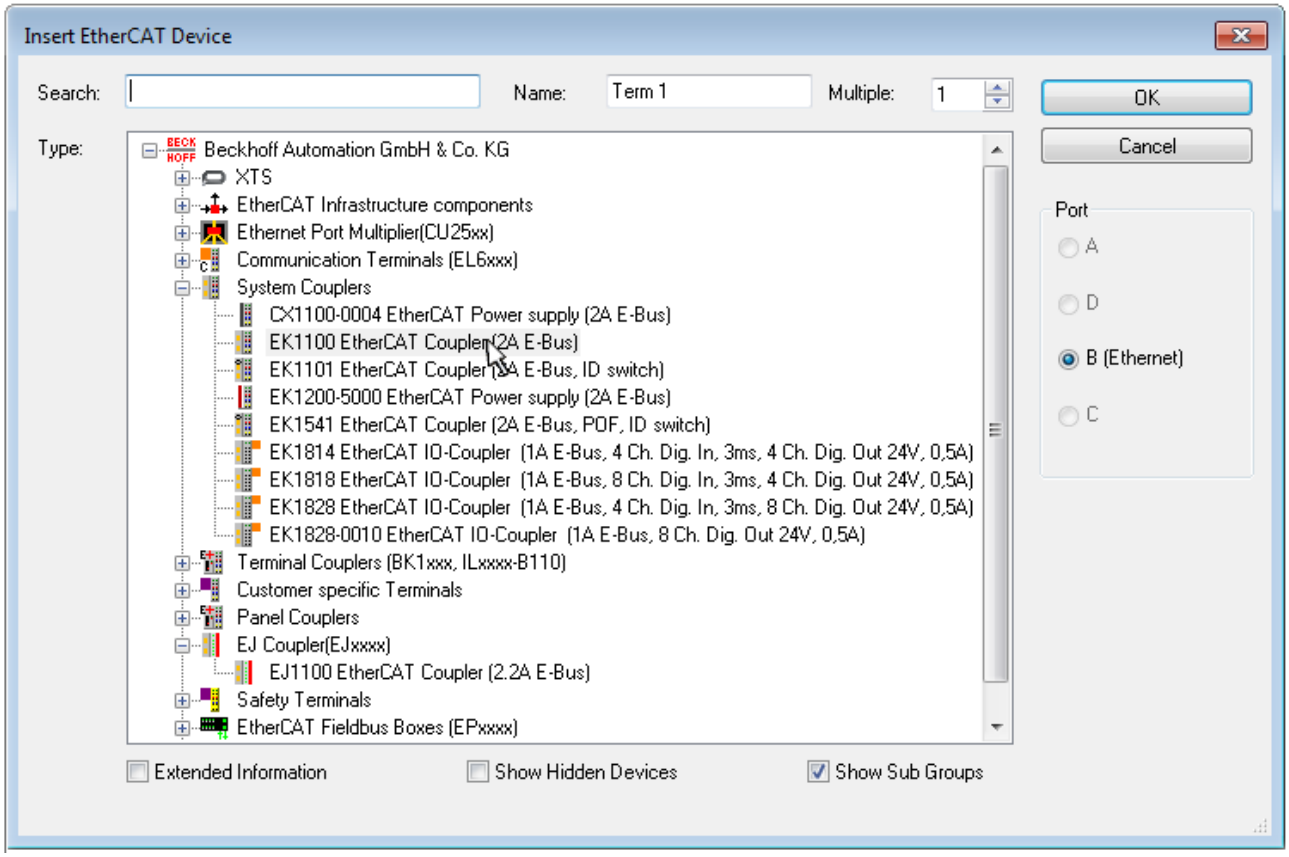


Fig. 101: Selection dialog for new EtherCAT device

By default, only the name/device type is used as selection criterion. For selecting a specific revision of the device, the revision can be displayed as “Extended Information”.

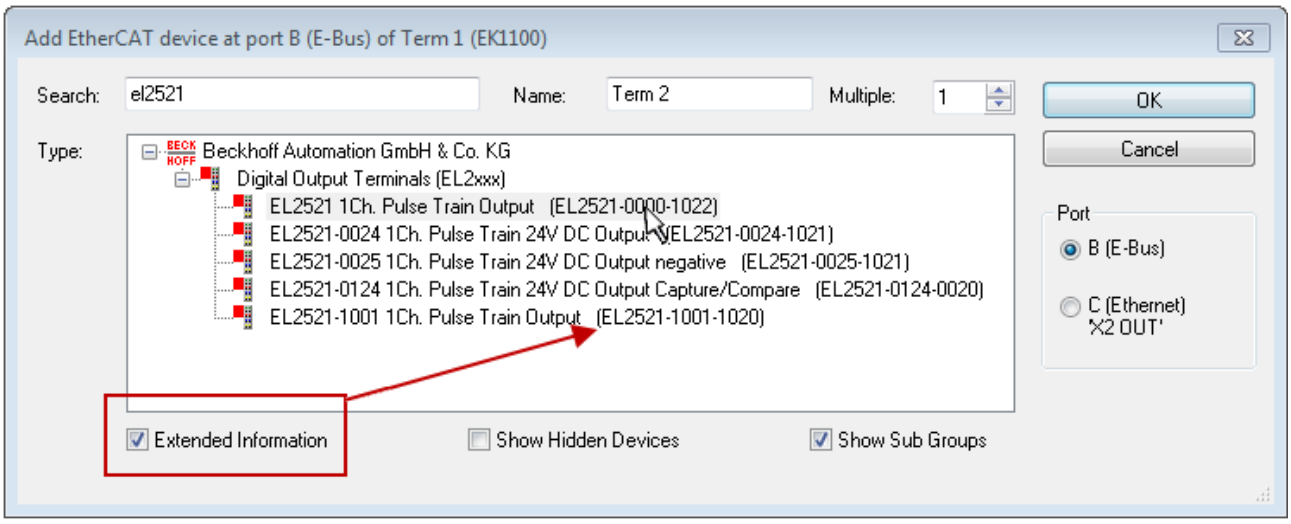


Fig. 102: Display of device revision

In many cases several device revisions were created for historic or functional reasons, e.g. through technological advancement. For simplification purposes (see Fig. “Selection dialog for new EtherCAT device”) only the last (i.e. highest) revision and therefore the latest state of production is displayed in the selection dialog for Beckhoff devices. To show all device revisions available in the system as ESI descriptions tick the “Show Hidden Devices” check box, see Fig. “Display of previous revisions”.

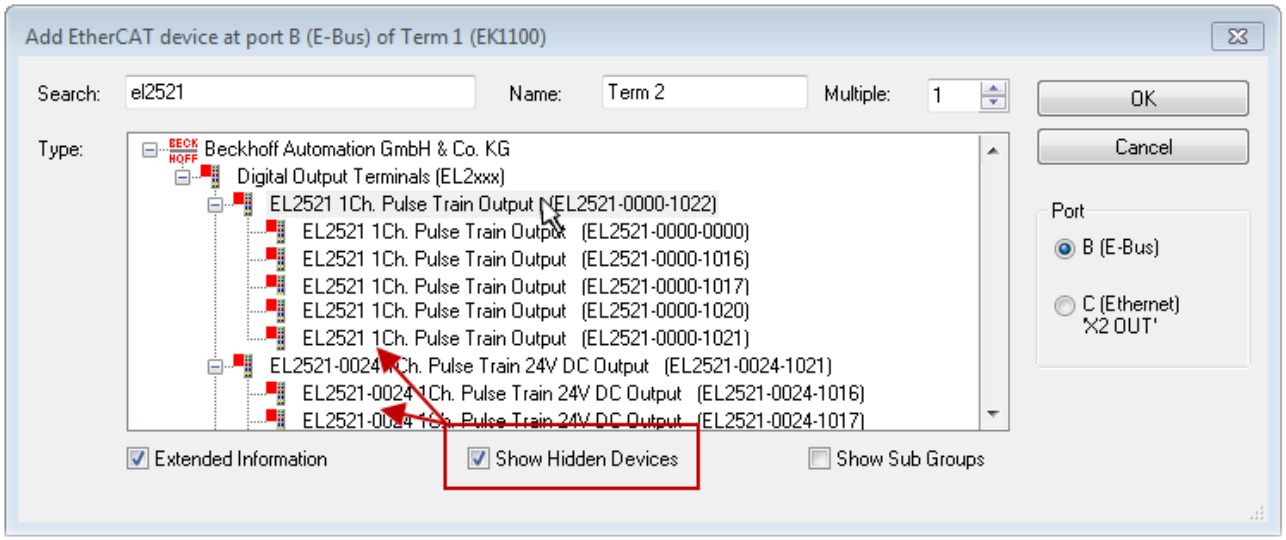


Fig. 103: Display of previous revisions

Device selection based on revision, compatibility

The ESI description also defines the process image, the communication type between master and slave/device and the device functions, if applicable. The physical device (firmware, if available) has to support the communication queries/settings of the master. This is backward compatible, i.e. newer devices (higher revision) should be supported if the EtherCAT master addresses them as an older revision. The following compatibility rule of thumb is to be assumed for Beckhoff EtherCAT Terminals/ Boxes/ EJ-modules:

device revision in the system >= device revision in the configuration

This also enables subsequent replacement of devices without changing the configuration (different specifications are possible for drives).

Example

If an EL2521-0025-**1018** is specified in the configuration, an EL2521-0025-**1018** or higher (-**1019**, -**1020**) can be used in practice.

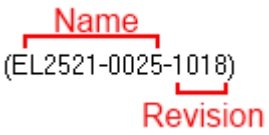


Fig. 104: Name/revision of the terminal

If current ESI descriptions are available in the TwinCAT system, the last revision offered in the selection dialog matches the Beckhoff state of production. It is recommended to use the last device revision when creating a new configuration, if current Beckhoff devices are used in the real application. Older revisions should only be used if older devices from stock are to be used in the application.

In this case the process image of the device is shown in the configuration tree and can be parameterized as follows: linking with the task, CoE/DC settings, plug-in definition, startup settings, ...

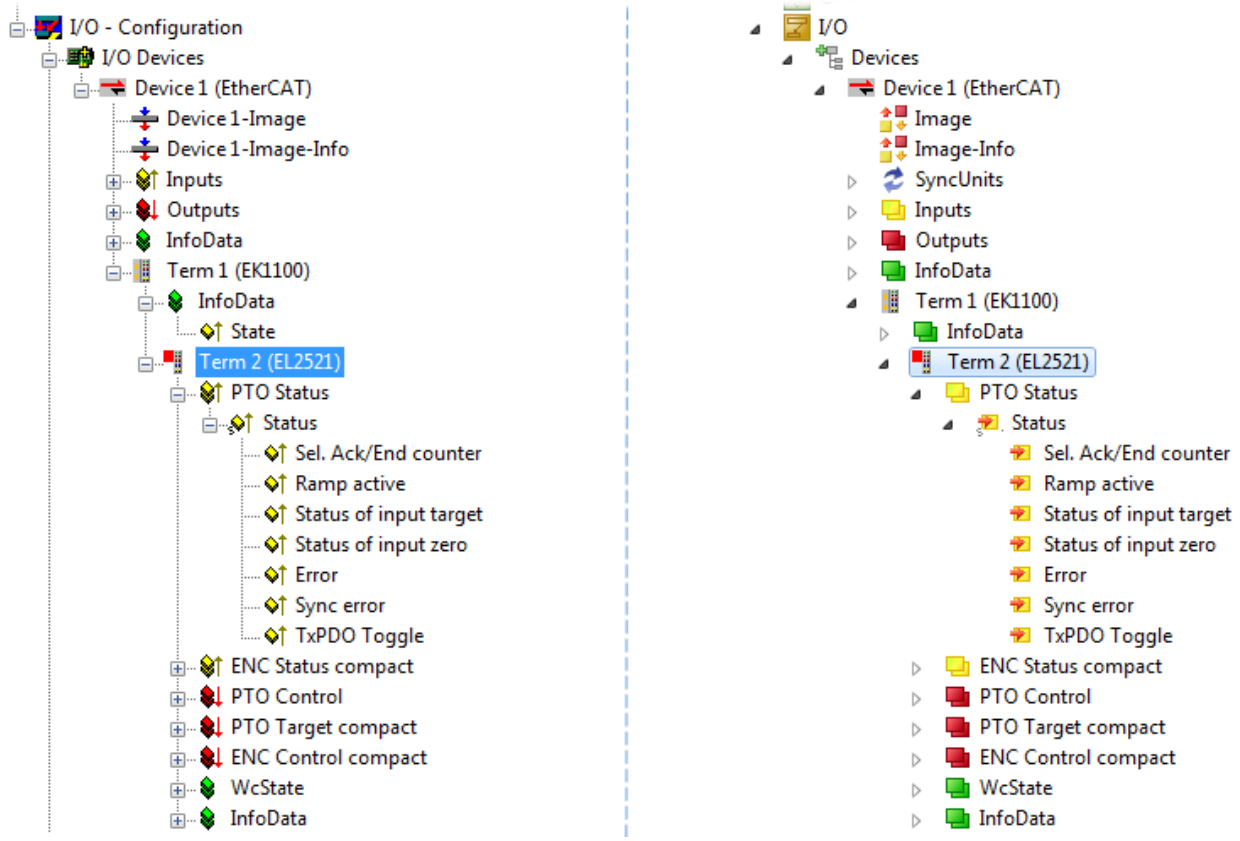


Fig. 105: EtherCAT terminal in the TwinCAT tree (left: TwinCAT 2; right: TwinCAT 3)

5.1.2.6 ONLINE configuration creation

Detecting/scanning of the EtherCAT device

The online device search can be used if the TwinCAT system is in CONFIG mode. This can be indicated by a symbol right below in the information bar:

- on TwinCAT 2 by a blue display “Config Mode” within the System Manager window: Config Mode .
- on TwinCAT 3 within the user interface of the development environment by a symbol .

TwinCAT can be set into this mode:

- TwinCAT 2: by selection of in the Menubar or by “Actions” → “Set/Reset TwinCAT to Config Mode…”
- TwinCAT 3: by selection of in the Menubar or by “TwinCAT” → “Restart TwinCAT (Config Mode)”

Online scanning in Config mode

The online search is not available in RUN mode (production operation). Note the differentiation between TwinCAT programming system and TwinCAT target system.

The TwinCAT 2 icon ( ) or TwinCAT 3 icon ( ) within the Windows-Taskbar always shows the TwinCAT mode of the local IPC. Compared to that, the System Manager window of TwinCAT 2 or the user interface of TwinCAT 3 indicates the state of the target system.

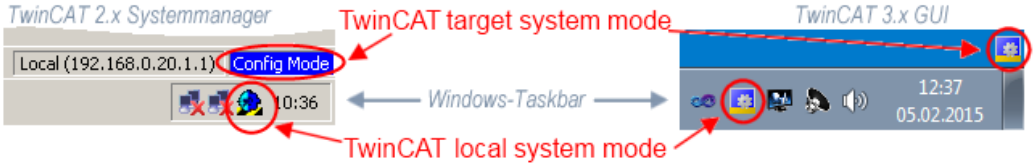


Fig. 106: Differentiation local/target system (left: TwinCAT 2; right: TwinCAT 3)

Right-clicking on “I/O Devices” in the configuration tree opens the search dialog.

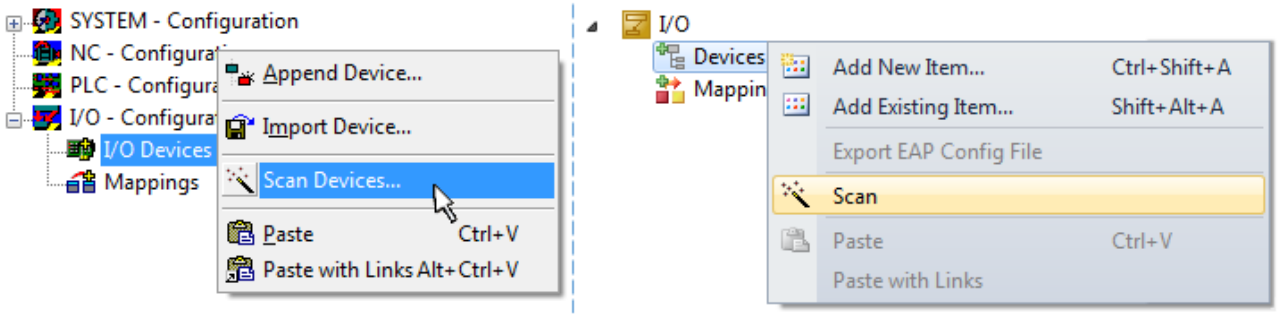


Fig. 107: Scan Devices (left: TwinCAT 2; right: TwinCAT 3)

This scan mode attempts to find not only EtherCAT devices (or Ethernet ports that are usable as such), but also NOVRAM, fieldbus cards, SMB etc. However, not all devices can be found automatically.

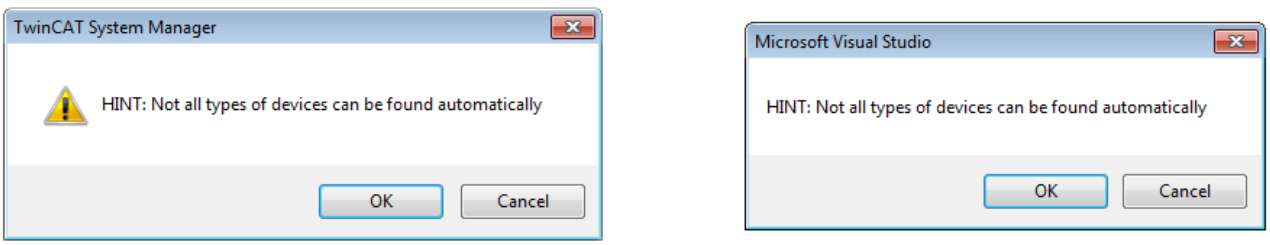


Fig. 108: Note for automatic device scan (left: TwinCAT 2; right: TwinCAT 3)

Ethernet ports with installed TwinCAT real-time driver are shown as “RT Ethernet” devices. An EtherCAT frame is sent to these ports for testing purposes. If the scan agent detects from the response that an EtherCAT slave is connected, the port is immediately shown as an “EtherCAT Device” .

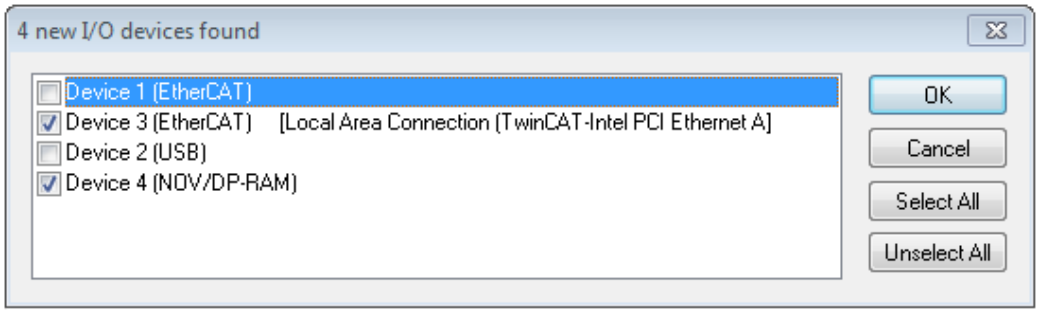


Fig. 109: Detected Ethernet devices

Via respective checkboxes devices can be selected (as illustrated in Fig. “Detected Ethernet devices” e.g. Device 3 and Device 4 were chosen). After confirmation with “OK” a device scan is suggested for all selected devices, see Fig.: “Scan query after automatic creation of an EtherCAT device”.

Selecting the Ethernet port

Ethernet ports can only be selected for EtherCAT devices for which the TwinCAT real-time driver is installed. This has to be done separately for each port. Please refer to the respective installation page [▶ 82].

Detecting/Scanning the EtherCAT devices

Online scan functionality

During a scan the master queries the identity information of the EtherCAT slaves from the slave EEPROM. The name and revision are used for determining the type. The respective devices are located in the stored ESI data and integrated in the configuration tree in the default state defined there.

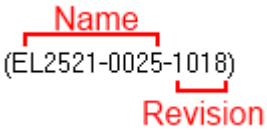


Fig. 110: Example default state

NOTICE

Slave scanning in practice in series machine production

The scanning function should be used with care. It is a practical and fast tool for creating an initial configuration as a basis for commissioning. In series machine production or reproduction of the plant, however, the function should no longer be used for the creation of the configuration, but if necessary for comparison [▶ 105] with the defined initial configuration.Background: since Beckhoff occasionally increases the revision version of the delivered products for product maintenance reasons, a configuration can be created by such a scan which (with an identical machine construction) is identical according to the device list; however, the respective device revision may differ from the initial configuration.

Example:

Company A builds the prototype of a machine B, which is to be produced in series later on. To do this the prototype is built, a scan of the IO devices is performed in TwinCAT and the initial configuration “B.tsm” is created. The EL2521-0025 EtherCAT terminal with the revision 1018 is located somewhere. It is thus built into the TwinCAT configuration in this way:

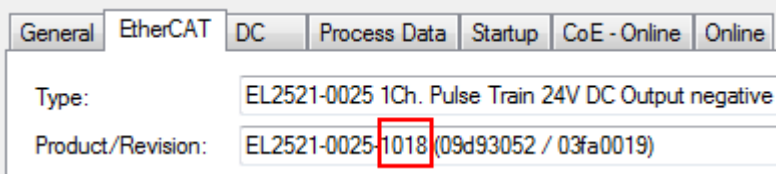


Fig. 111: Installing EthetCAT terminal with revision -1018

Likewise, during the prototype test phase, the functions and properties of this terminal are tested by the programmers/commissioning engineers and used if necessary, i.e. addressed from the PLC “B.pro” or the NC. (the same applies correspondingly to the TwinCAT 3 solution files).

The prototype development is now completed and series production of machine B starts, for which Beckhoff continues to supply the EL2521-0025-0018. If the commissioning engineers of the series machine production department always carry out a scan, a B configuration with the identical contents results again for each machine. Likewise, A might create spare parts stores worldwide for the coming series-produced machines with EL2521-0025-1018 terminals.

After some time Beckhoff extends the EL2521-0025 by a new feature C. Therefore the FW is changed, outwardly recognizable by a higher FW version and **a new revision -1019**. Nevertheless the new device naturally supports functions and interfaces of the predecessor version(s); an adaptation of “B.tsm” or even “B.pro” is therefore unnecessary. The series-produced machines can continue to be built with “B.tsm” and “B.pro”; it makes sense to perform a comparative scan [▶ 105] against the initial configuration “B.tsm” in order to check the built machine.

However, if the series machine production department now doesn’t use “B.tsm”, but instead carries out a scan to create the productive configuration, the revision **-1019** is automatically detected and built into the configuration:

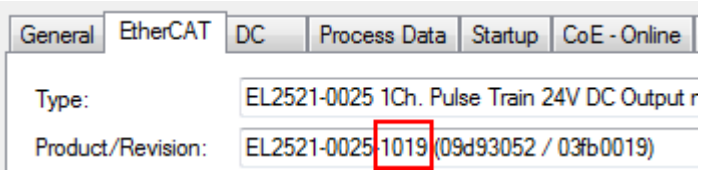


Fig. 112: Detection of EtherCAT terminal with revision -1019

This is usually not noticed by the commissioning engineers. TwinCAT cannot signal anything either, since a new configuration is essentially created. According to the compatibility rule, however, this means that no EL2521-0025-**1018** should be built into this machine as a spare part (even if this nevertheless works in the vast majority of cases).

In addition, it could be the case that, due to the development accompanying production in company A, the new feature C of the EL2521-0025-1019 (for example, an improved analog filter or an additional process data for the diagnosis) is discovered and used without in-house consultation. The previous stock of spare part devices are then no longer to be used for the new configuration “B2.tsm” created in this way. Þ if series machine production is established, the scan should only be performed for informative purposes for comparison with a defined initial configuration. Changes are to be made with care!

If an EtherCAT device was created in the configuration (manually or through a scan), the I/O field can be scanned for devices/slaves.



Fig. 113: Scan query after automatic creation of an EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

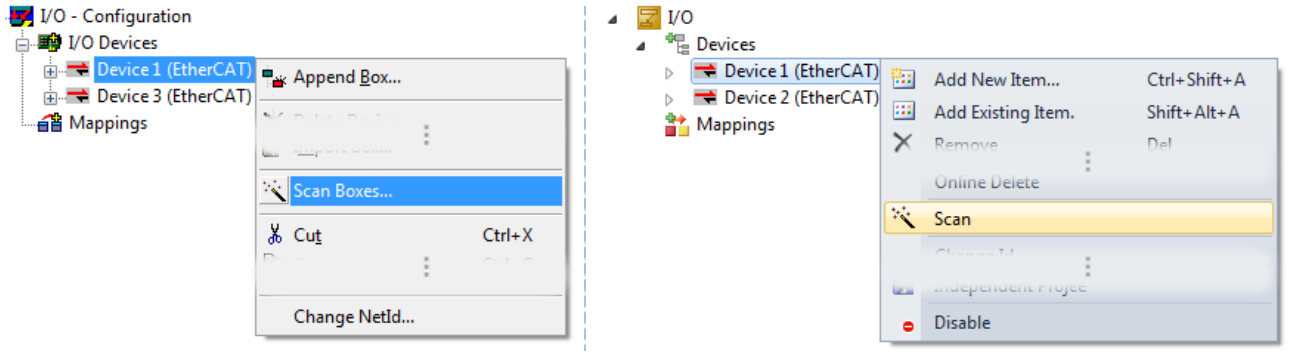


Fig. 114: Manual scanning for devices on a specified EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

In the System Manager (TwinCAT 2) or the User Interface (TwinCAT 3) the scan process can be monitored via the progress bar at the bottom in the status bar.

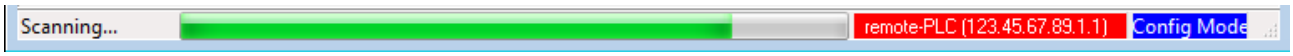


Fig. 115: *Scan progressexemplary by TwinCAT 2*

The configuration is established and can then be switched to online state (OPERATIONAL).



Fig. 116: Config/FreeRun query (left: TwinCAT 2; right: TwinCAT 3)

In Config/FreeRun mode the System Manager display alternates between blue and red, and the EtherCAT device continues to operate with the idling cycle time of 4 ms (default setting), even without active task (NC, PLC).

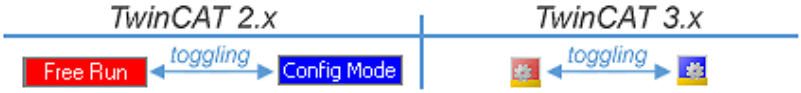


Fig. 117: Displaying of “Free Run” and “Config Mode” toggling right below in the status bar



Fig. 118: TwinCAT can also be switched to this state by using a button (left: TwinCAT 2; right: TwinCAT 3)

The EtherCAT system should then be in a functional cyclic state, as shown in Fig. *Online display example*.

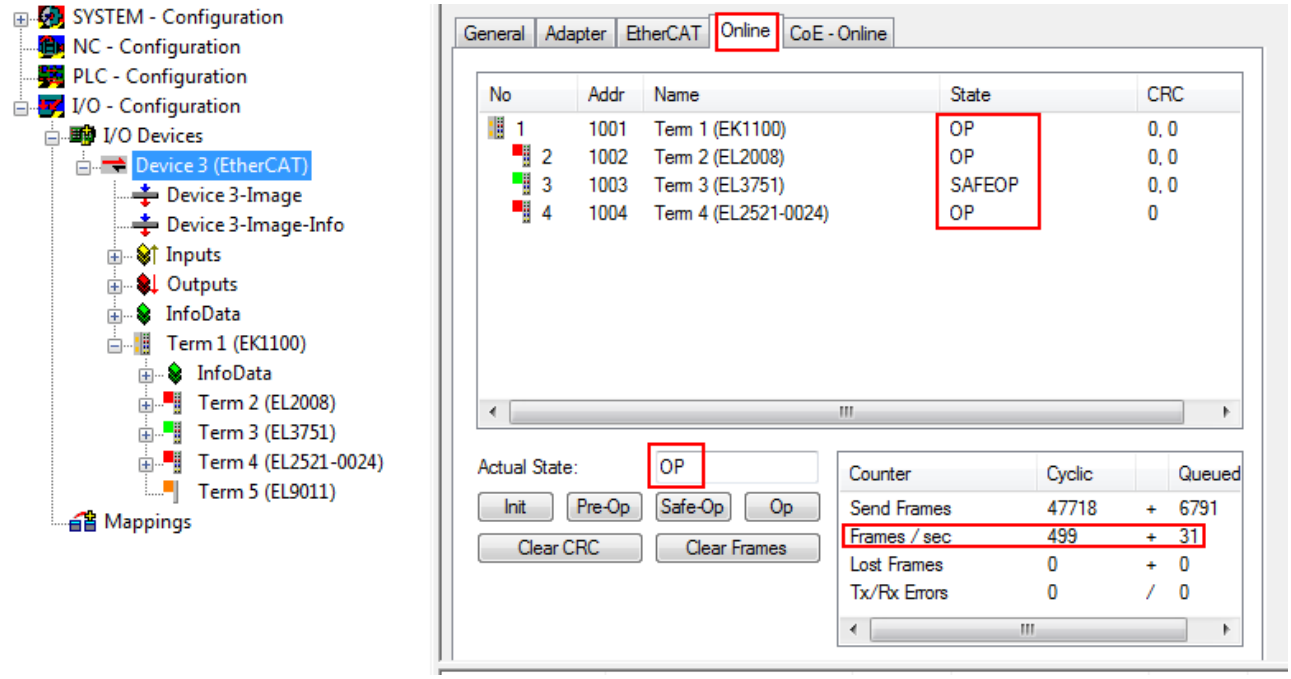


Fig. 119: Online display example

Please note:

- all slaves should be in OP state
- the EtherCAT master should be in “Actual State” OP
- “frames/sec” should match the cycle time taking into account the sent number of frames
- no excessive “LostFrames” or CRC errors should occur

The configuration is now complete. It can be modified as described under manual procedure [▶ 94].

Troubleshooting

Various effects may occur during scanning.

- An **unknown device** is detected, i.e. an EtherCAT slave for which no ESI XML description is available. In this case the System Manager offers to read any ESI that may be stored in the device. This case is described in the chapter “Notes regarding ESI device description”.
- **Device are not detected properly**
  Possible reasons include:
  - faulty data links, resulting in data loss during the scan
  - slave has invalid device description

  The connections and devices should be checked in a targeted manner, e.g. via the emergency scan. Then re-run the scan.

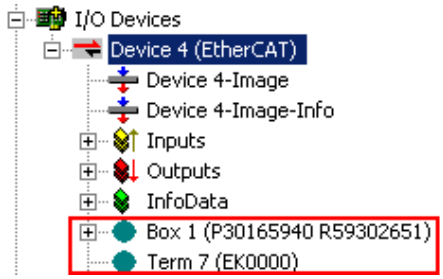


Fig. 120: Faulty identification

In the System Manager such devices may be set up as EK0000 or unknown devices. Operation is not possible or meaningful.

Scan over existing Configuration

NOTICE

Change of the configuration after comparison

With this scan (TwinCAT 2.11 or 3.1) only the device properties vendor (manufacturer), device name and revision are compared at present! A “ChangeTo” or “Copy” should only be carried out with care, taking into consideration the Beckhoff IO compatibility rule (see above). The device configuration is then replaced by the revision found; this can affect the supported process data and functions.

If a scan is initiated for an existing configuration, the actual I/O environment may match the configuration exactly or it may differ. This enables the configuration to be compared.



Fig. 121: Identical configuration (left: TwinCAT 2; right: TwinCAT 3)

If differences are detected, they are shown in the correction dialog, so that the user can modify the configuration as required.

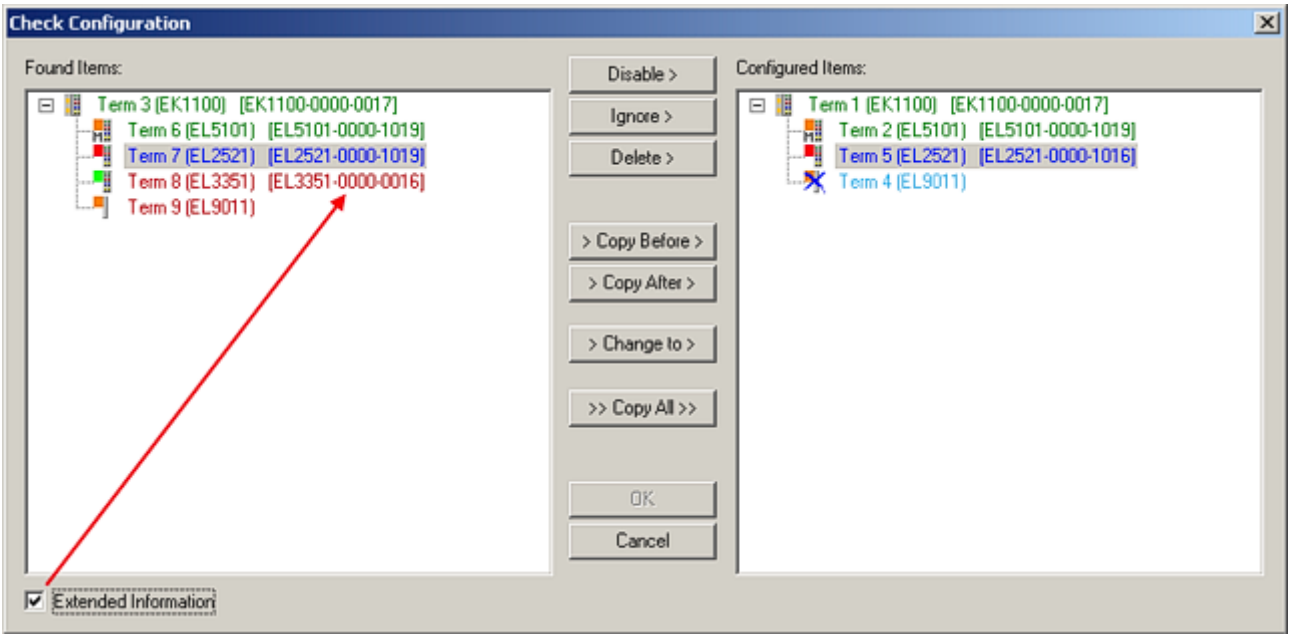


Fig. 122: Correction dialog

It is advisable to tick the “Extended Information” check box to reveal differences in the revision.

Color	Explanation
green	This EtherCAT slave matches the entry on the other side. Both type and revision match.
blue	This EtherCAT slave is present on the other side, but in a different revision. This other revision can have other default values for the process data as well as other/additional functions. If the found revision is higher than the configured revision, the slave may be used provided compatibility issues are taken into account. If the found revision is lower than the configured revision, it is likely that the slave cannot be used. The found device may not support all functions that the master expects based on the higher revision number.
light blue	This EtherCAT slave is ignored (“Ignore” button)
red	• This EtherCAT slave is not present on the other side. • It is present, but in a different revision, which also differs in its properties from the one specified. The compatibility principle then also applies here: if the found revision is higher than the configured revision, use is possible provided compatibility issues are taken into account, since the successor devices should support the functions of the predecessor devices. If the found revision is lower than the configured revision, it is likely that the slave cannot be used. The found device may not support all functions that the master expects based on the higher revision number.

Device selection based on revision, compatibility

The ESI description also defines the process image, the communication type between master and slave/device and the device functions, if applicable. The physical device (firmware, if available) has to support the communication queries/settings of the master. This is backward compatible, i.e. newer devices (higher revision) should be supported if the EtherCAT master addresses them as an older revision. The following compatibility rule of thumb is to be assumed for Beckhoff EtherCAT Terminals/ Boxes/ EJ-modules:

device revision in the system >= device revision in the configuration

This also enables subsequent replacement of devices without changing the configuration (different specifications are possible for drives).

Example

If an EL2521-0025-**1018** is specified in the configuration, an EL2521-0025-**1018** or higher (-**1019**, -**1020**) can be used in practice.

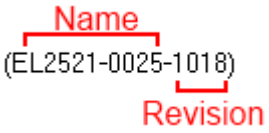


Fig. 123: Name/revision of the terminal

If current ESI descriptions are available in the TwinCAT system, the last revision offered in the selection dialog matches the Beckhoff state of production. It is recommended to use the last device revision when creating a new configuration, if current Beckhoff devices are used in the real application. Older revisions should only be used if older devices from stock are to be used in the application.

In this case the process image of the device is shown in the configuration tree and can be parameterized as follows: linking with the task, CoE/DC settings, plug-in definition, startup settings, ...

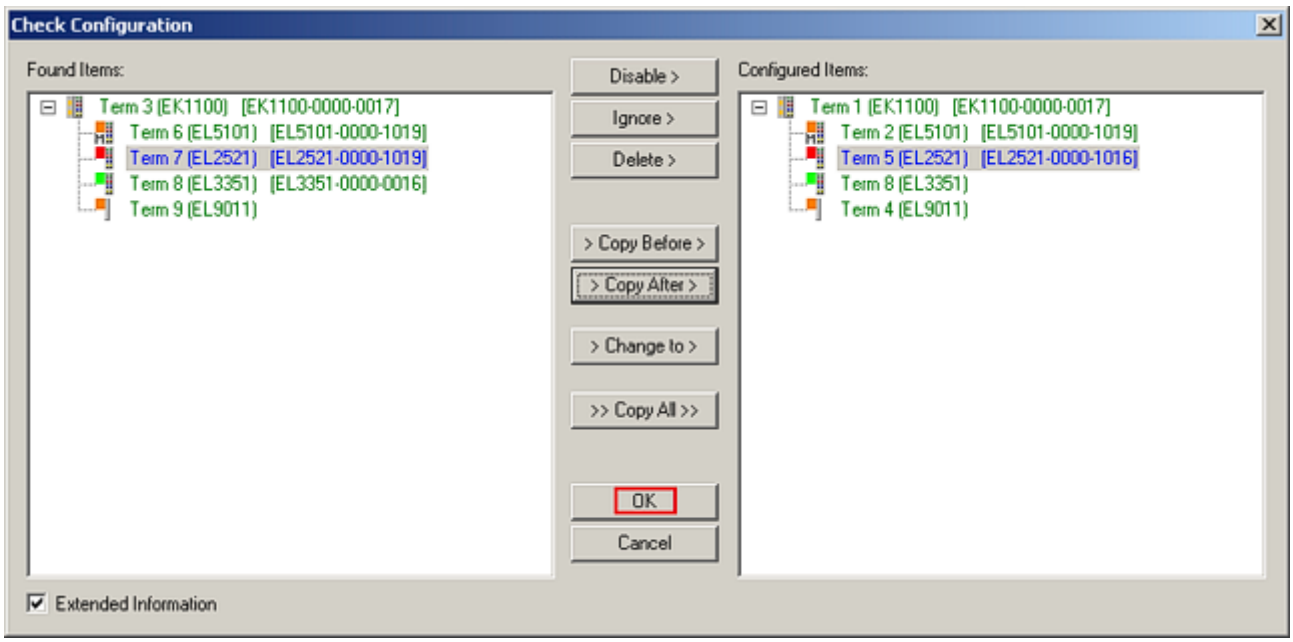


Fig. 124: Correction dialog with modifications

Once all modifications have been saved or accepted, click “OK” to transfer them to the real *.tsm configuration.

Change to Compatible Type

TwinCAT offers a function *Change to Compatible Type…* for the exchange of a device whilst retaining the links in the task.

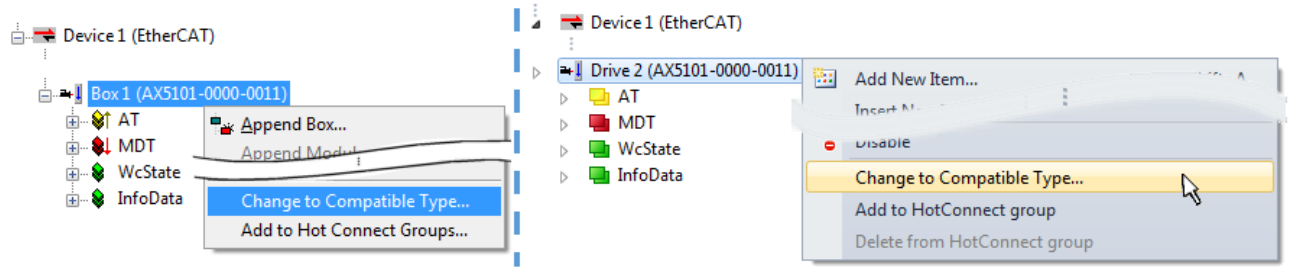


Fig. 125: Dialog “Change to Compatible Type…” (left: TwinCAT 2; right: TwinCAT 3)

The following elements in the ESI of an EtherCAT device are compared by TwinCAT and assumed to be the same in order to decide whether a device is indicated as "compatible":

- Physics (e.g. RJ45, Ebus...)
- FMMU (additional ones are allowed)
- SyncManager (SM, additional ones are allowed)
- EoE (attributes MAC, IP)
- CoE (attributes SdoInfo, PdoAssign, PdoConfig, PdoUpload, CompleteAccess)
- FoE
- PDO (process data: Sequence, SyncUnit SU, SyncManager SM, EntryCount, Ent-ry.Datatype)

This function is preferably to be used on AX5000 devices.

Change to Alternative Type

The TwinCAT System Manager offers a function for the exchange of a device: Change to Alternative Type

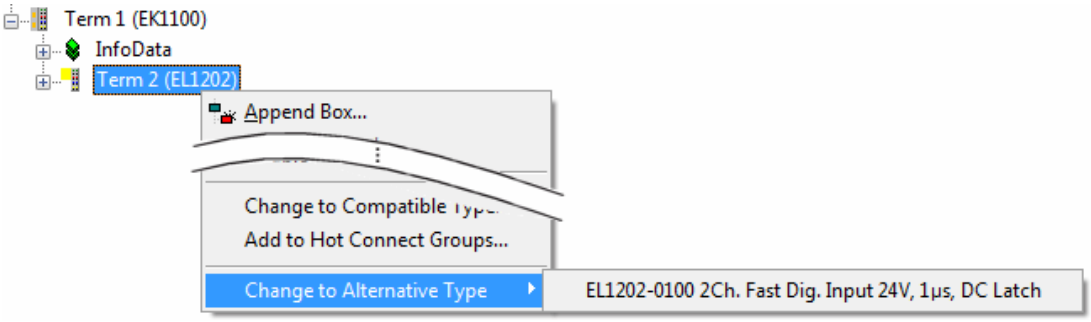


Fig. 126: TwinCAT 2 Dialog Change to Alternative Type

If called, the System Manager searches in the procured device ESI (in this example: EL1202-0000) for details of compatible devices contained there. The configuration is changed and the ESI-EEPROM is overwritten at the same time – therefore this process is possible only in the online state (ConfigMode).

5.1.2.7 EtherCAT subscriber configuration

In the left-hand window of the TwinCAT 2 System Manager or the Solution Explorer of the TwinCAT 3 Development Environment respectively, click on the element of the terminal within the tree you wish to configure (in the example: EL3751 Terminal 3).

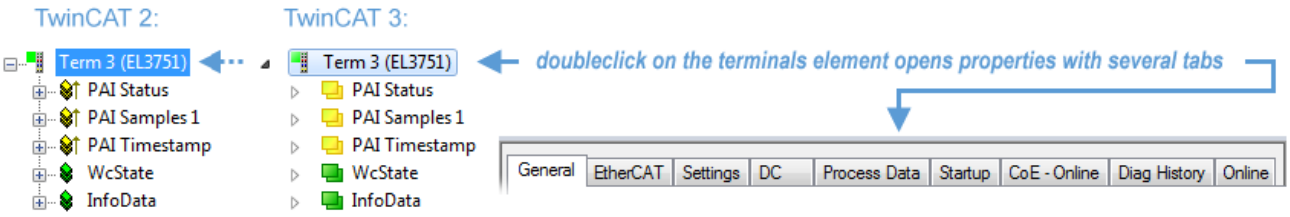


Fig. 127: Branch element as terminal EL3751

In the right-hand window of the TwinCAT System Manager (TwinCAT 2) or the Development Environment (TwinCAT 3), various tabs are now available for configuring the terminal. And yet the dimension of complexity of a subscriber determines which tabs are provided. Thus as illustrated in the example above the terminal EL3751 provides many setup options and also a respective number of tabs are available. On the contrary by the terminal EL1004 for example the tabs “General”, “EtherCAT”, “Process Data” and “Online“ are available only. Several terminals, as for instance the EL6695 provide special functions by a tab with its own terminal name, so “EL6695” in this case. A specific tab “Settings” by terminals with a wide range of setup options will be provided also (e.g. EL3751).

“General” tab

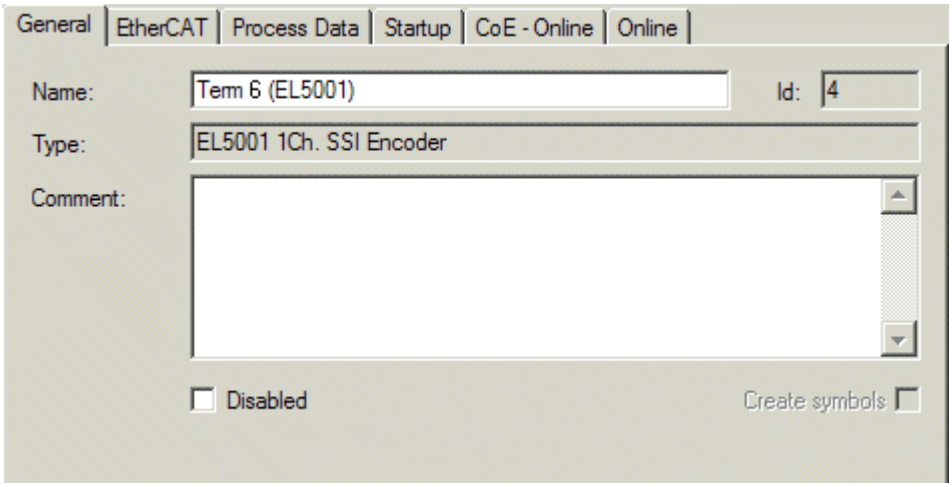


Fig. 128: “General” tab

Name	Name of the EtherCAT device
Id	Number of the EtherCAT device
Type	EtherCAT device type
Comment	Here you can add a comment (e.g. regarding the system).
Disabled	Here you can deactivate the EtherCAT device.
Create symbols	Access to this EtherCAT slave via ADS is only available if this control box is activated.

“EtherCAT” tab

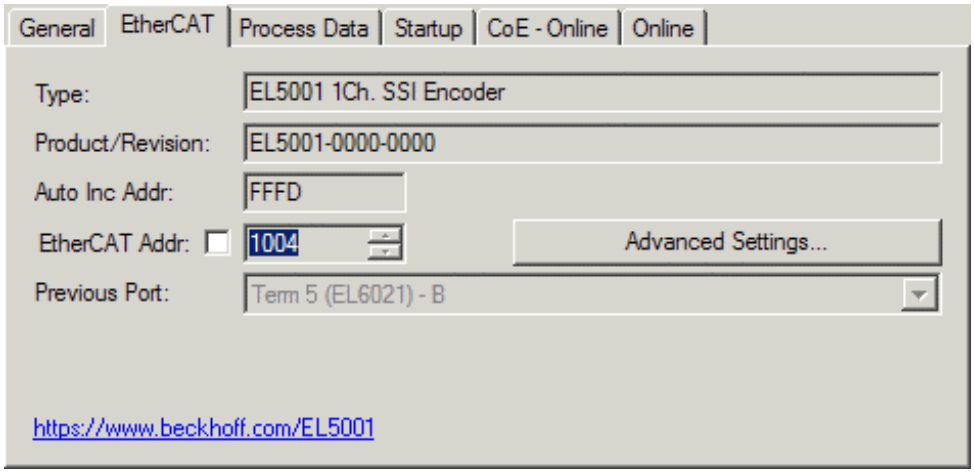


Fig. 129: “EtherCAT” tab

Type	EtherCAT device type
Product/Revision	Product and revision number of the EtherCAT device
Auto Inc Addr.	Auto increment address of the EtherCAT device. The auto increment address can be used for addressing each EtherCAT device in the communication ring through its physical position. Auto increment addressing is used during the start-up phase when the EtherCAT master allocates addresses to the EtherCAT devices. With auto increment addressing the first EtherCAT slave in the ring has the address 0000_{hex}. For each further slave the address is decremented by 1 ($FFFF_{hex}$, $FFFE_{hex}$ etc.).
EtherCAT Addr.	Fixed address of an EtherCAT slave. This address is allocated by the EtherCAT master during the start-up phase. Tick the control box to the left of the input field in order to modify the default value.
Previous Port	Name and port of the EtherCAT device to which this device is connected. If it is possible to connect this device with another one without changing the order of the EtherCAT devices in the communication ring, then this combination field is activated and the EtherCAT device to which this device is to be connected can be selected.
Advanced Settings	This button opens the dialogs for advanced settings.

The link at the bottom of the tab points to the product page for this EtherCAT device on the web.

“Process Data” tab

Indicates the configuration of the process data. The input and output data of the EtherCAT slave are represented as CANopen process data objects (**P**rocess **D**ata **O**bjects, PDOs). The user can select a PDO via PDO assignment and modify the content of the individual PDO via this dialog, if the EtherCAT slave supports this function.

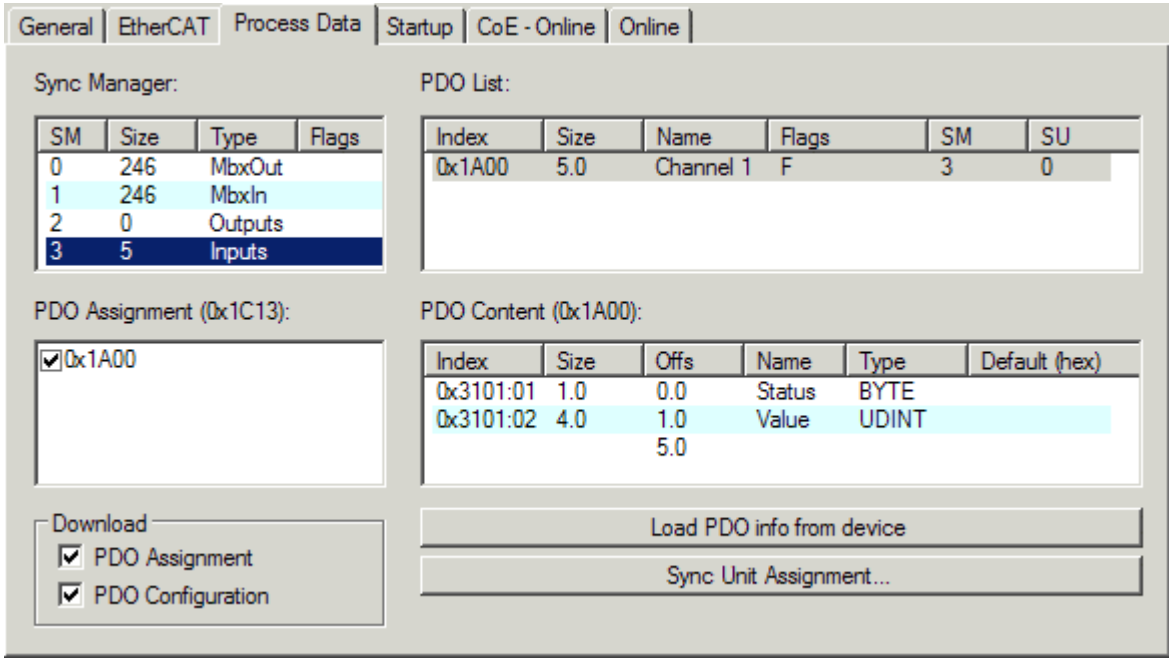


Fig. 130: “Process Data” tab

The process data (PDOs) transferred by an EtherCAT slave during each cycle are user data which the application expects to be updated cyclically or which are sent to the slave. To this end the EtherCAT master (Beckhoff TwinCAT) parameterizes each EtherCAT slave during the start-up phase to define which process data (size in bits/bytes, source location, transmission type) it wants to transfer to or from this slave. Incorrect configuration can prevent successful start-up of the slave.

For Beckhoff EtherCAT EL, ES, EM, EJ and EP slaves the following applies in general:

- The input/output process data supported by the device are defined by the manufacturer in the ESI/XML description. The TwinCAT EtherCAT Master uses the ESI description to configure the slave correctly.
- The process data can be modified in the System Manager. See the device documentation.
  Examples of modifications include: mask out a channel, displaying additional cyclic information, 16-bit display instead of 8-bit data size, etc.
- In so-called “intelligent” EtherCAT devices the process data information is also stored in the CoE directory. Any changes in the CoE directory that lead to different PDO settings prevent successful startup of the slave. It is not advisable to deviate from the designated process data, because the device firmware (if available) is adapted to these PDO combinations.

If the device documentation allows modification of process data, proceed as follows (see Figure *Configuring the process data*).

- A: select the device to configure
- B: in the “Process Data” tab select Input or Output under SyncManager (C)
- D: the PDOs can be selected or deselected
- H: the new process data are visible as linkable variables in the System Manager
  The new process data are active once the configuration has been activated and TwinCAT has been restarted (or the EtherCAT master has been restarted)
- E: if a slave supports this, Input and Output PDO can be modified simultaneously by selecting a so-called PDO record (“predefined PDO settings”).

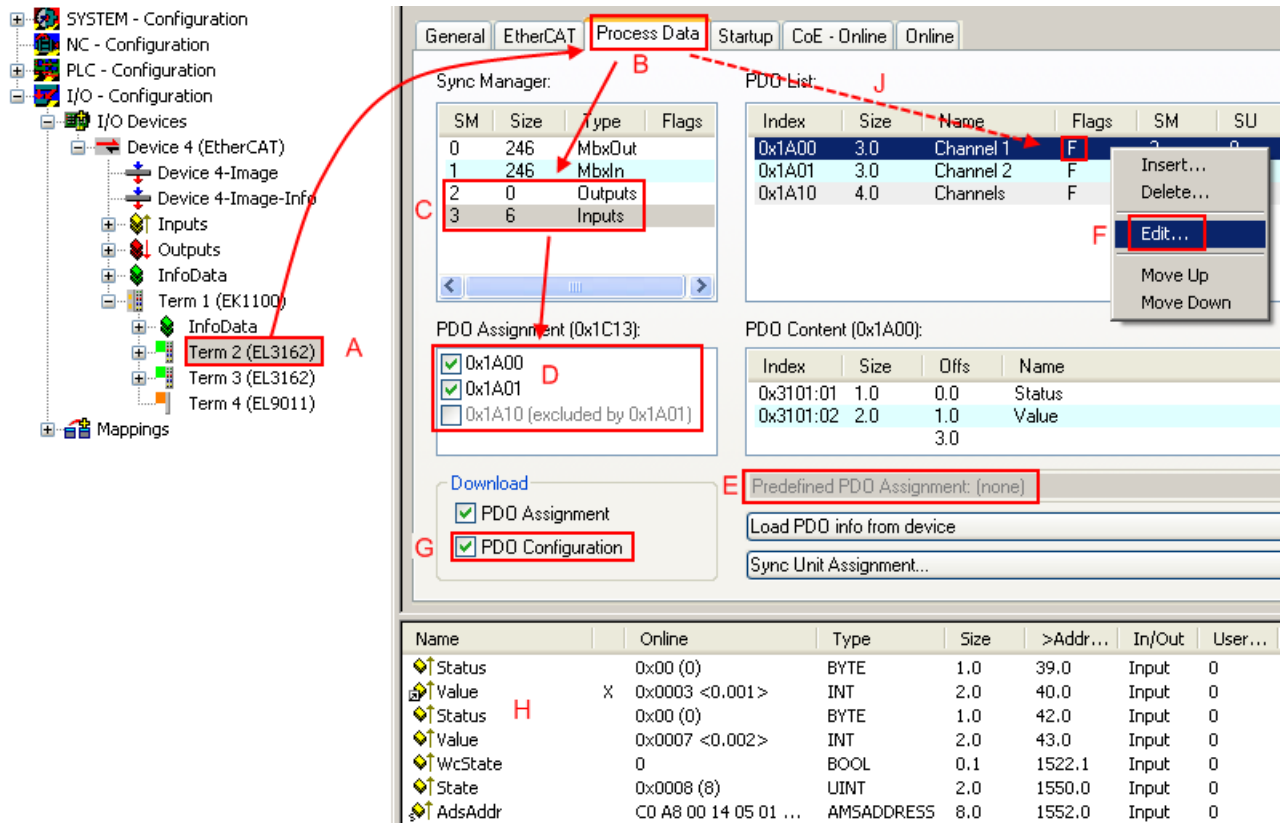


Fig. 131: Configuring the process data

Manual modification of the process data

According to the ESI description, a PDO can be identified as “fixed” with the flag “F” in the PDO overview (Fig. *Configuring the process data*, J). The configuration of such PDOs cannot be changed, even if TwinCAT offers the associated dialog (“Edit”). In particular, CoE content cannot be displayed as cyclic process data. This generally also applies in cases where a device supports download of the PDO configuration, “G”. In case of incorrect configuration the EtherCAT slave usually refuses to start and change to OP state. The System Manager displays an “invalid SM cfg” logger message: This error message (“invalid SM IN cfg” or “invalid SM OUT cfg”) also indicates the reason for the failed start.

A detailed description [▶ 116] can be found at the end of this section.

“Startup” tab

The *Startup* tab is displayed if the EtherCAT slave has a mailbox and supports the *CANopen over EtherCAT* (CoE) or *Servo drive over EtherCAT* protocol. This tab indicates which download requests are sent to the mailbox during startup. It is also possible to add new mailbox requests to the list display. The download requests are sent to the slave in the same order as they are shown in the list.

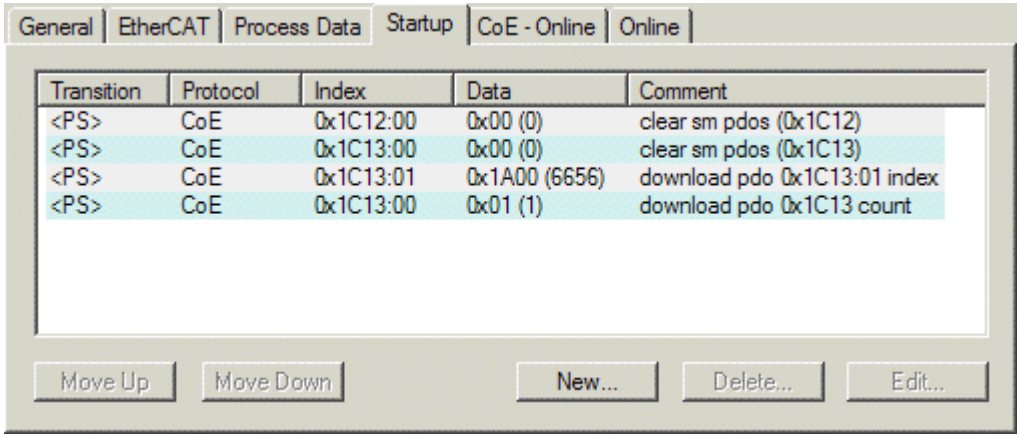


Fig. 132: “Startup” tab

Column	Description
Transition	Transition to which the request is sent. This can either be • the transition from pre-operational to safe-operational (PS), or • the transition from safe-operational to operational (SO). If the transition is enclosed in “<>” (e.g. <PS>), the mailbox request is fixed and cannot be modified or deleted by the user.
Protocol	Type of mailbox protocol
Index	Index of the object
Data	Date on which this object is to be downloaded.
Comment	Description of the request to be sent to the mailbox

Move Up	This button moves the selected request up by one position in the list.
Move Down	This button moves the selected request down by one position in the list.
New	This button adds a new mailbox download request to be sent during startup.
Delete	This button deletes the selected entry.
Edit	This button edits an existing request.

“CoE - Online” tab

The additional *CoE - Online* tab is displayed if the EtherCAT slave supports the *CANopen over EtherCAT* (CoE) protocol. This dialog lists the content of the object list of the slave (SDO upload) and enables the user to modify the content of an object from this list. Details for the objects of the individual EtherCAT devices can be found in the device-specific object descriptions.

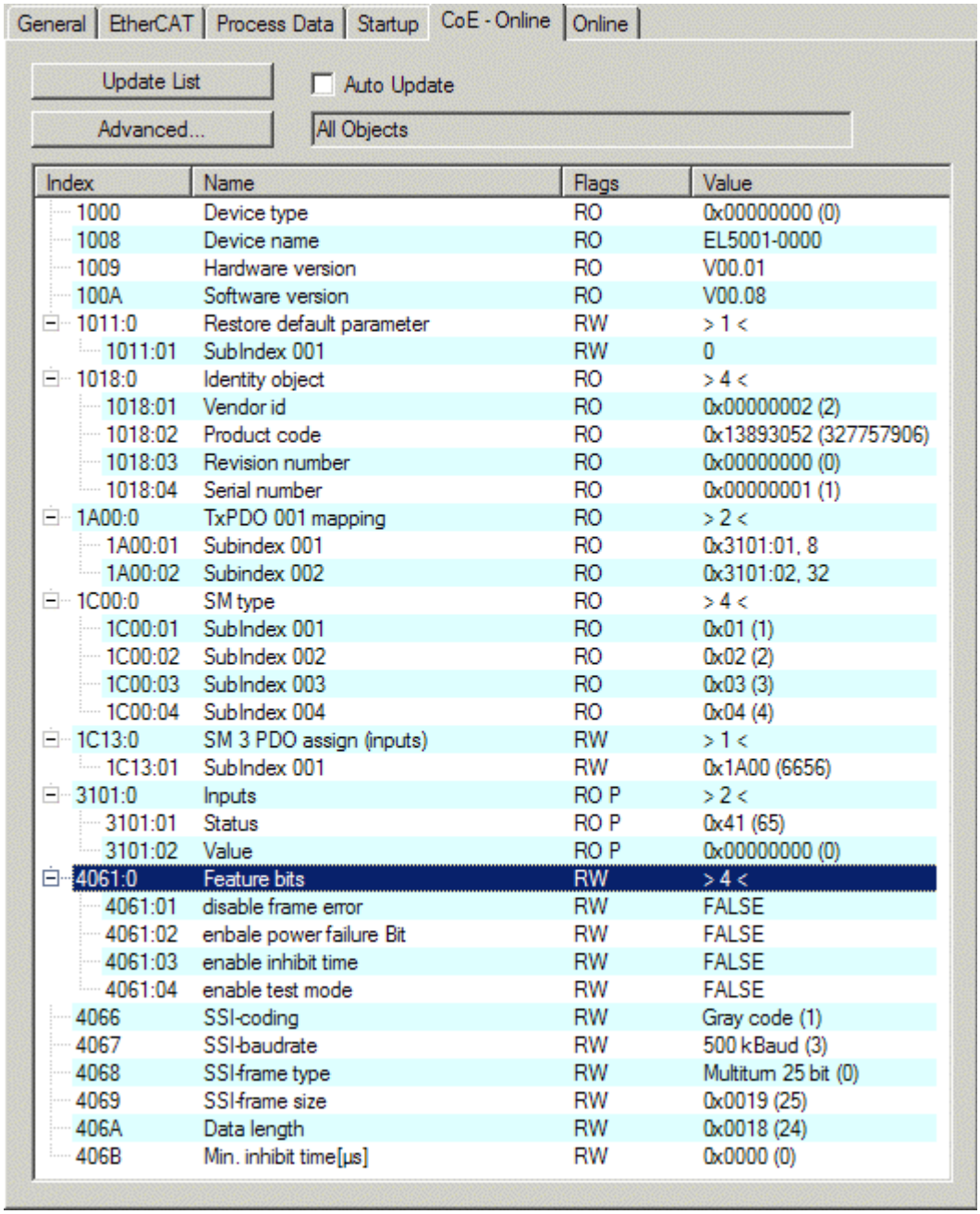


Fig. 133: “CoE - Online” tab

Object list display

Column	Description	
Index	Index and sub-index of the object	
Name	Name of the object	
Flags	RW	The object can be read, and data can be written to the object (read/write)
	RO	The object can be read, but no data can be written to the object (read only)
	P	An additional P identifies the object as a process data object.
Value	Value of the object	

Update List The *Update list* button updates all objects in the displayed list

Auto Update If this check box is selected, the content of the objects is updated automatically.

Advanced The *Advanced* button opens the *Advanced Settings* dialog. Here you can specify which objects are displayed in the list.

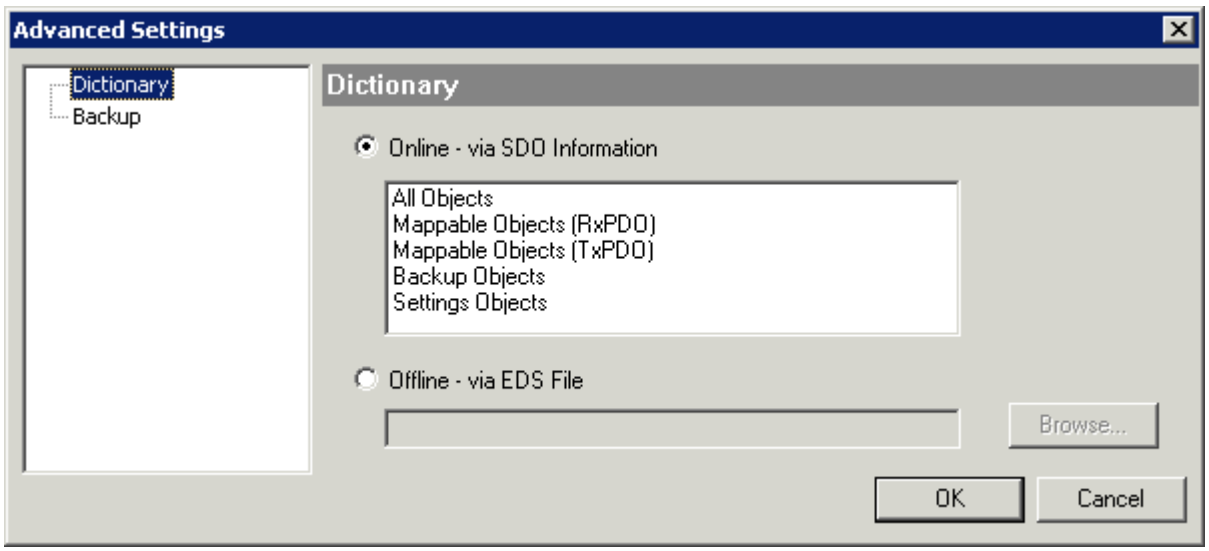


Fig. 134: Dialog “Advanced settings”

Online - via SDO Information	If this option button is selected, the list of the objects included in the object list of the slave is uploaded from the slave via SDO information. The list below can be used to specify which object types are to be uploaded.
Offline - via EDS File	If this option button is selected, the list of the objects included in the object list is read from an EDS file provided by the user.

“Online” tab

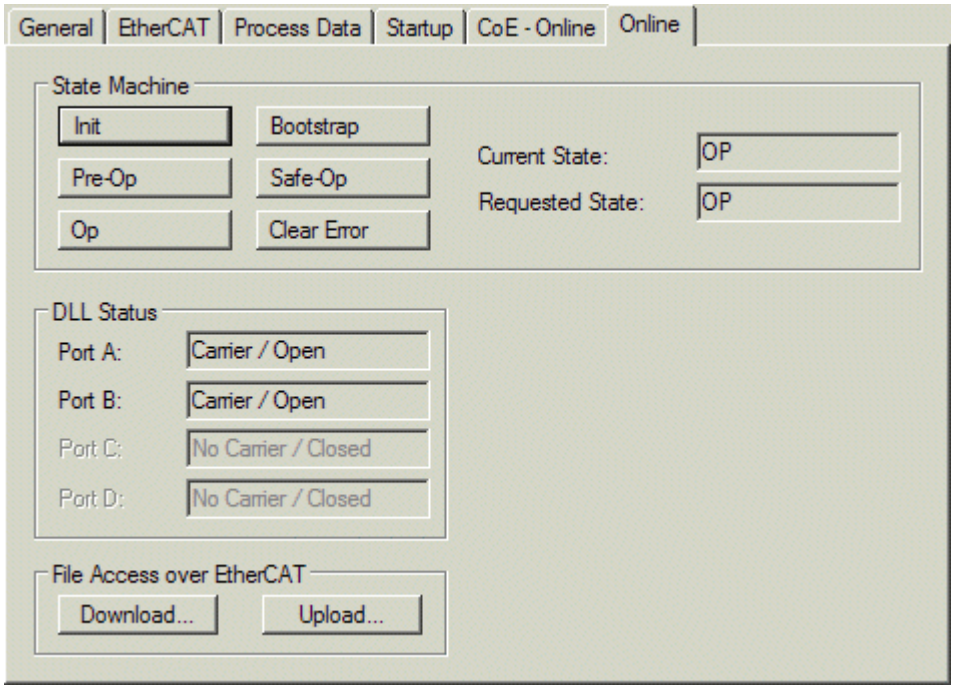


Fig. 135: “Online” tab

State Machine

Init	This button attempts to set the EtherCAT device to the *Init* state.
Pre-Op	This button attempts to set the EtherCAT device to the *pre-operational* state.
Op	This button attempts to set the EtherCAT device to the *operational* state.
Bootstrap	This button attempts to set the EtherCAT device to the *Bootstrap* state.
Safe-Op	This button attempts to set the EtherCAT device to the *safe-operational* state.
Clear Error	This button attempts to delete the fault display. If an EtherCAT slave fails during change of state it sets an error flag. Example: An EtherCAT slave is in PREOP state (pre-operational). The master now requests the SAFEOP state (safe-operational). If the slave fails during change of state it sets the error flag. The current state is now displayed as ERR PREOP. When the *Clear Error* button is pressed the error flag is cleared, and the current state is displayed as PREOP again.
Current State	Indicates the current state of the EtherCAT device.
Requested State	Indicates the state requested for the EtherCAT device.

DLL Status

Indicates the DLL status (data link layer status) of the individual ports of the EtherCAT slave. The DLL status can have four different states:

Status	Description
No Carrier / Open	No carrier signal is available at the port, but the port is open.
No Carrier / Closed	No carrier signal is available at the port, and the port is closed.
Carrier / Open	A carrier signal is available at the port, and the port is open.
Carrier / Closed	A carrier signal is available at the port, but the port is closed.

File Access over EtherCAT

Download	With this button a file can be written to the EtherCAT device.
Upload	With this button a file can be read from the EtherCAT device.

“DC” tab (Distributed Clocks)

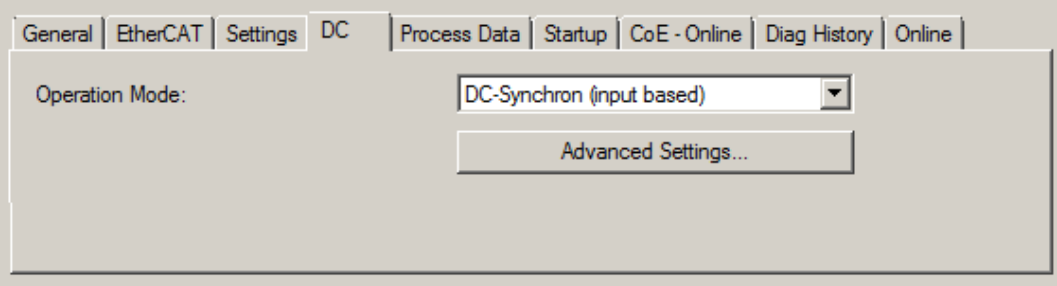


Fig. 136: “DC” tab (Distributed Clocks)

Operation Mode	Options (optional): • FreeRun • SM-Synchron • DC-Synchron (Input based) • DC-Synchron
Advanced Settings…	Advanced settings for readjustment of the real time determinant TwinCAT-clock

Detailed information to Distributed Clocks is specified on http://infosys.beckhoff.com:

Fieldbus Components → EtherCAT Terminals → EtherCAT System documentation → EtherCAT basics → Distributed Clocks

5.1.2.7.1 Detailed description of Process Data tab

Sync Manager

Lists the configuration of the Sync Manager (SM).
If the EtherCAT device has a mailbox, SM0 is used for the mailbox output (MbxOut) and SM1 for the mailbox input (MbxIn).
SM2 is used for the output process data (outputs) and SM3 (inputs) for the input process data.

If an input is selected, the corresponding PDO assignment is displayed in the *PDO Assignment* list below.

PDO Assignment

PDO assignment of the selected Sync Manager. All PDOs defined for this Sync Manager type are listed here:

- If the output Sync Manager (outputs) is selected in the Sync Manager list, all RxPDOs are displayed.
- If the input Sync Manager (inputs) is selected in the Sync Manager list, all TxPDOs are displayed.

The selected entries are the PDOs involved in the process data transfer. In the tree diagram of the System Manager these PDOs are displayed as variables of the EtherCAT device. The name of the variable is identical to the *Name* parameter of the PDO, as displayed in the PDO list. If an entry in the PDO assignment list is deactivated (not selected and greyed out), this indicates that the input is excluded from the PDO assignment. In order to be able to select a greyed out PDO, the currently selected PDO has to be deselected first.

Activation of PDO assignment

✓ If you have changed the PDO assignment, in order to activate the new PDO assignment,

a) the EtherCAT slave has to run through the PS status transition cycle (from pre-operational to safe-operational) once (see Online tab [▶ 114]),

b) and the System Manager has to reload the EtherCAT slaves

( button for TwinCAT 2 or  button for TwinCAT 3)

PDO list

List of all PDOs supported by this EtherCAT device. The content of the selected PDOs is displayed in the *PDO Content* list. The PDO configuration can be modified by double-clicking on an entry.

Column	Description	
Index	PDO index.	
Size	Size of the PDO in bytes.	
Name	Name of the PDO. If this PDO is assigned to a Sync Manager, it appears as a variable of the slave with this parameter as the name.	
Flags	F	Fixed content: The content of this PDO is fixed and cannot be changed by the System Manager.
	M	Mandatory PDO. This PDO is mandatory and must therefore be assigned to a Sync Manager! Consequently, this PDO cannot be deleted from the *PDO Assignment* list
SM	Sync Manager to which this PDO is assigned. If this entry is empty, this PDO does not take part in the process data traffic.	
SU	Sync unit to which this PDO is assigned.	

PDO Content

Indicates the content of the PDO. If flag F (fixed content) of the PDO is not set the content can be modified.

Download

If the device is intelligent and has a mailbox, the configuration of the PDO and the PDO assignments can be downloaded to the device. This is an optional feature that is not supported by all EtherCAT slaves.

PDO Assignment

If this check box is selected, the PDO assignment that is configured in the PDO Assignment list is downloaded to the device on startup. The required commands to be sent to the device can be viewed in the Startup [▶ 111] tab.

PDO Configuration

If this check box is selected, the configuration of the respective PDOs (as shown in the PDO list and the PDO Content display) is downloaded to the EtherCAT slave.

5.1.2.8 Import/Export of EtherCAT devices with SCI and XTI

SCI and XTI Export/Import – Handling of user-defined modified EtherCAT slaves

5.1.2.8.1 Basic principles

An EtherCAT slave is basically parameterized through the following elements:

- Cyclic process data (PDO)
- Synchronization (Distributed Clocks, FreeRun, SM-Synchron)
- CoE parameters (acyclic object dictionary)

Note: Not all three elements may be present, depending on the slave.

For a better understanding of the export/import function, let's consider the usual procedure for IO configuration:

- The user/programmer processes the IO configuration in the TwinCAT system environment. This involves all input/output devices such as drives that are connected to the fieldbuses used.
  Note: In the following sections, only EtherCAT configurations in the TwinCAT system environment are considered.
- For example, the user manually adds devices to a configuration or performs a scan on the online system.
- This results in the IO system configuration.
- On insertion, the slave appears in the system configuration in the default configuration provided by the vendor, consisting of default PDO, default synchronization method and CoE StartUp parameter as defined in the ESI (XML device description).
- If necessary, elements of the slave configuration can be changed, e.g. the PDO configuration or the synchronization method, based on the respective device documentation.

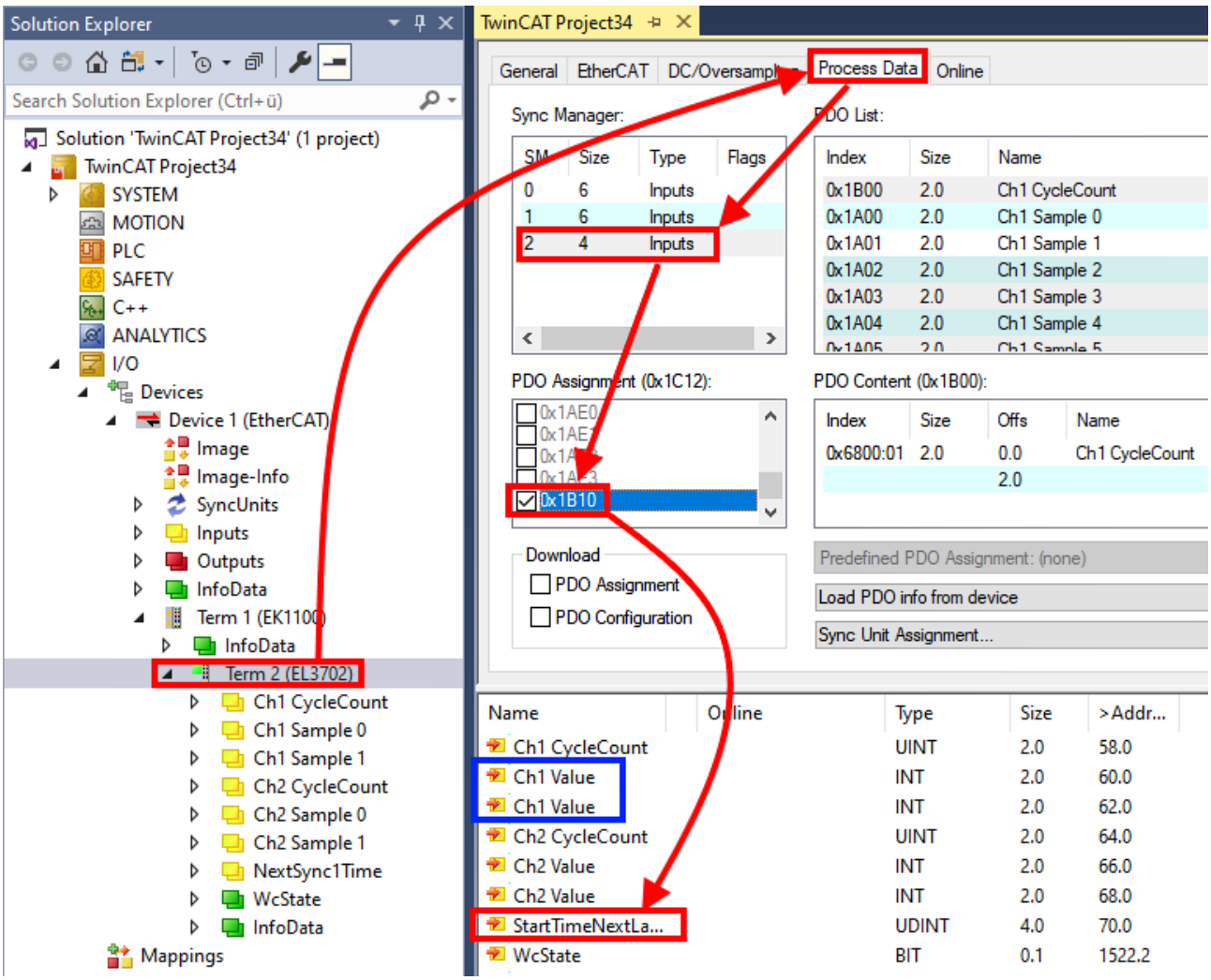
It may become necessary to reuse the modified slave in other projects in this way, without having to make equivalent configuration changes to the slave again. To accomplish this, proceed as follows:

- Export the slave configuration from the project,
- Store and transport as a file,
- Import into another EtherCAT project.

TwinCAT offers two methods for this purpose:

- within the TwinCAT environment: Export/Import as **xti** file or
- outside, i.e. beyond the TwinCAT limits: Export/Import as **sci** file.

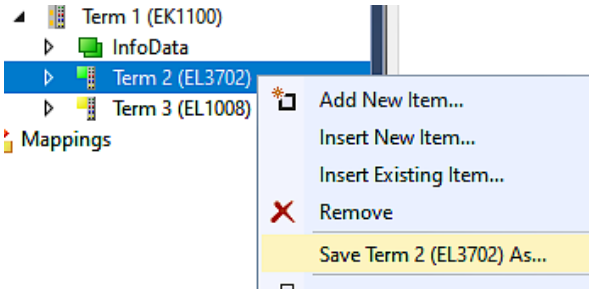
An example is provided below for illustration purposes: an EL3702 terminal with standard setting is switched to 2-fold oversampling (blue) and the optional PDO "StartTimeNextLatch" is added (red):



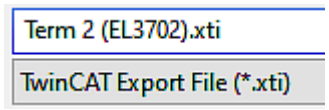
The two methods for exporting and importing the modified terminal referred to above are demonstrated below.

5.1.2.8.2 Procedure within TwinCAT with xti files

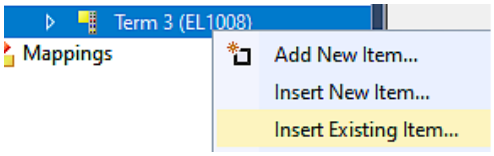
Each IO device can be exported/saved individually:



The xti file can be stored:



and imported again in another TwinCAT system via "Insert Existing item":



5.1.2.8.3 Procedure within and outside TwinCAT with sci file

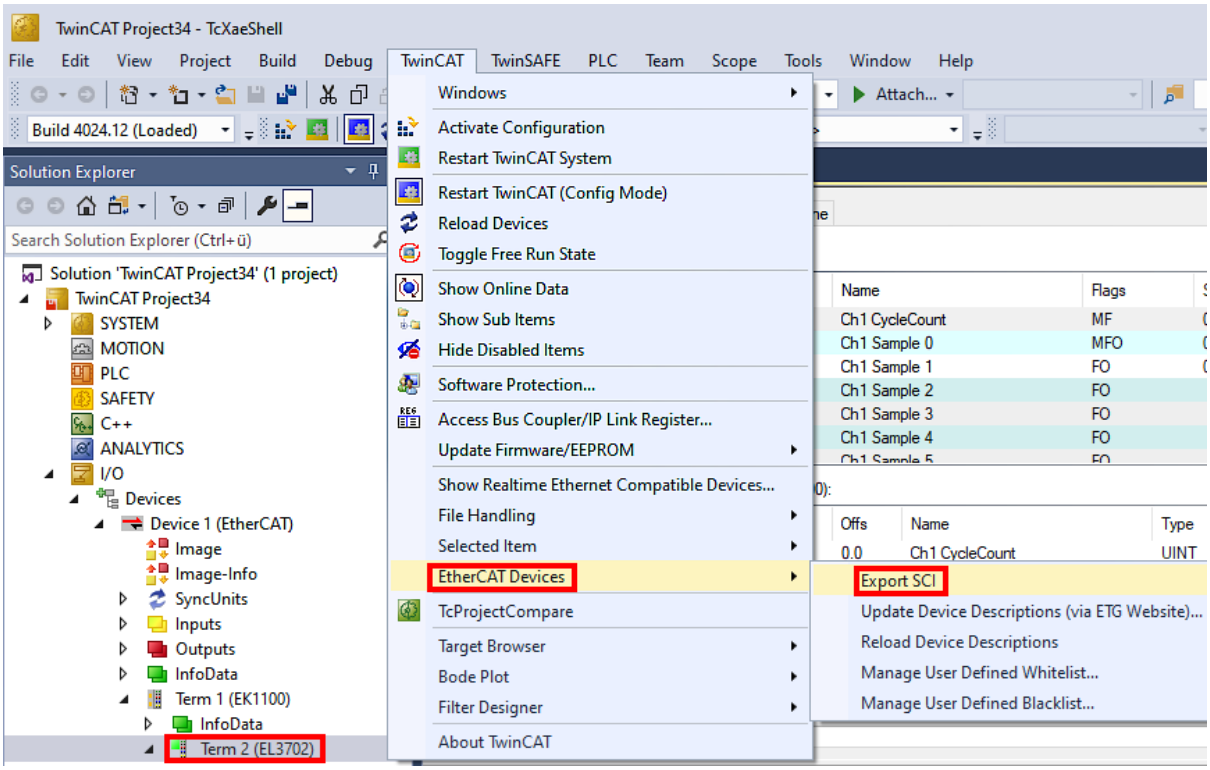
Note regarding availability (2021/01)

The SCI method is available from TwinCAT 3.1 build 4024.14.

The Slave Configuration Information (SCI) describes a specific complete configuration for an EtherCAT slave (terminal, box, drive...) based on the setting options of the device description file (ESI, EtherCAT Slave Information). That is, it includes PDO, CoE, synchronization.

Export:

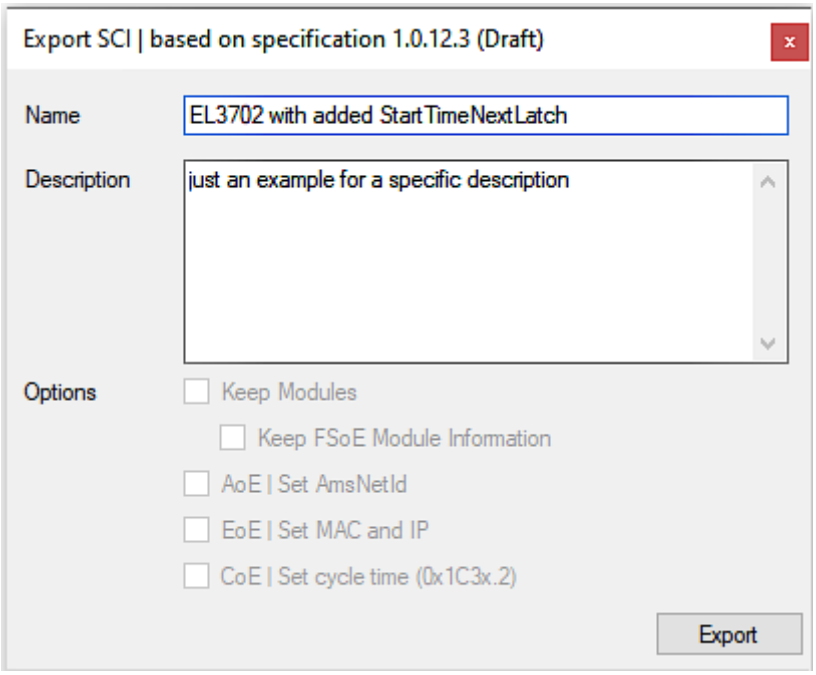
- select a single device via the menu (multiple selection is also possible):
  TwinCAT → EtherCAT Devices → Export SCI.



- If TwinCAT is offline (i.e. if there is no connection to an actual running controller) a warning message may appear, because after executing the function the system attempts to reload the EtherCAT segment. However, in this case this is not relevant for the result and can be acknowledged by clicking OK:



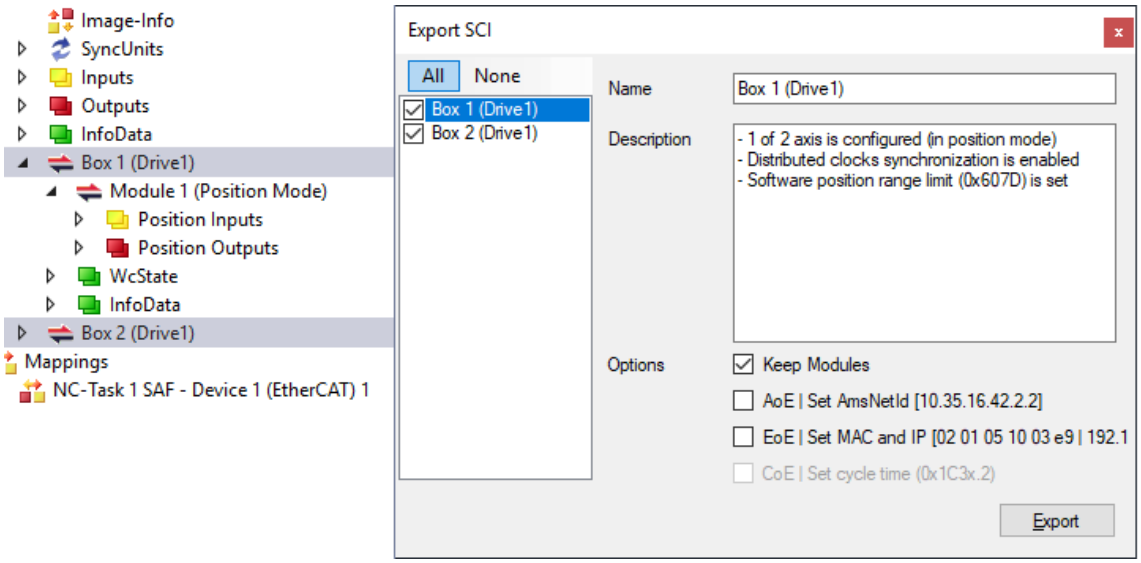
- A description may also be provided:



- Explanation of the dialog box:

Name		Name of the SCI, assigned by the user.
Description		Description of the slave configuration for the use case, assigned by the user.
Options	Keep modules	If a slave supports modules/slots, the user can decide whether these are to be exported or whether the module and device data are to be combined during export.
	AoE | Set AmsNetId	The configured AmsNetId is exported. Usually this is network-dependent and cannot always be determined in advance.
	EoE | Set MAC and IP	The configured virtual MAC and IP addresses are stored in the SCI. Usually these are network-dependent and cannot always be determined in advance.
	CoE | Set cycle time(0x1C3x.2)	The configured cycle time is exported. Usually this is network-dependent and cannot always be determined in advance.
ESI		Reference to the original ESI file.
Export		Save SCI file.

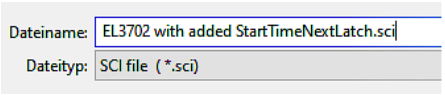
- A list view is available for multiple selections (*Export multiple SCI files)*:



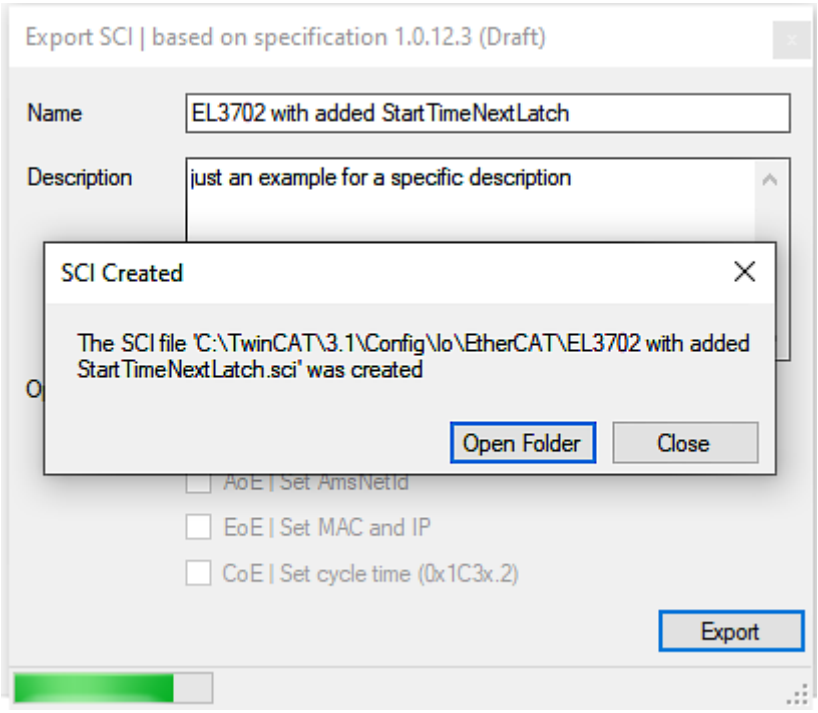
- Selection of the slaves to be exported:
  - All:
    All slaves are selected for export.

- None:
  All slaves are deselected.

- The sci file can be saved locally:

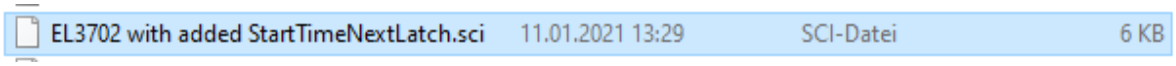


- The export takes place:

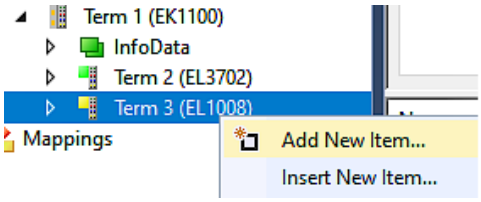


Import

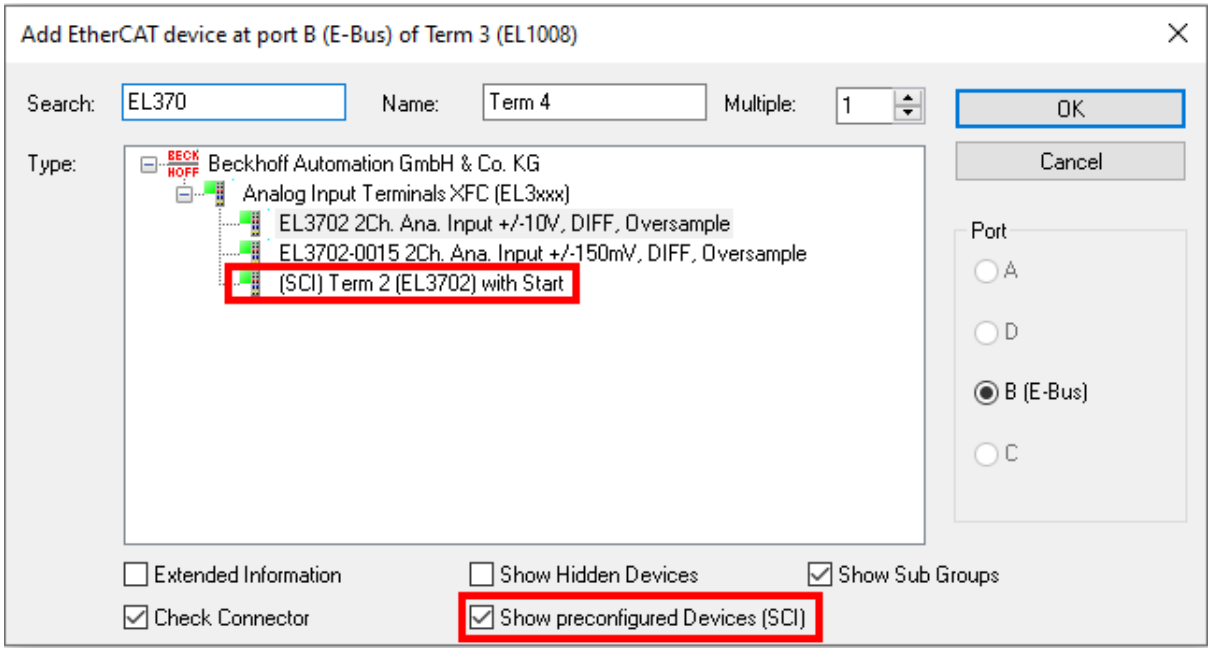
- An sci description can be inserted manually into the TwinCAT configuration like any normal Beckhoff device description.
- The sci file must be located in the TwinCAT ESI path, usually under:
  C:\TwinCAT\3.1\Config\Io\EtherCAT



- Open the selection dialog:

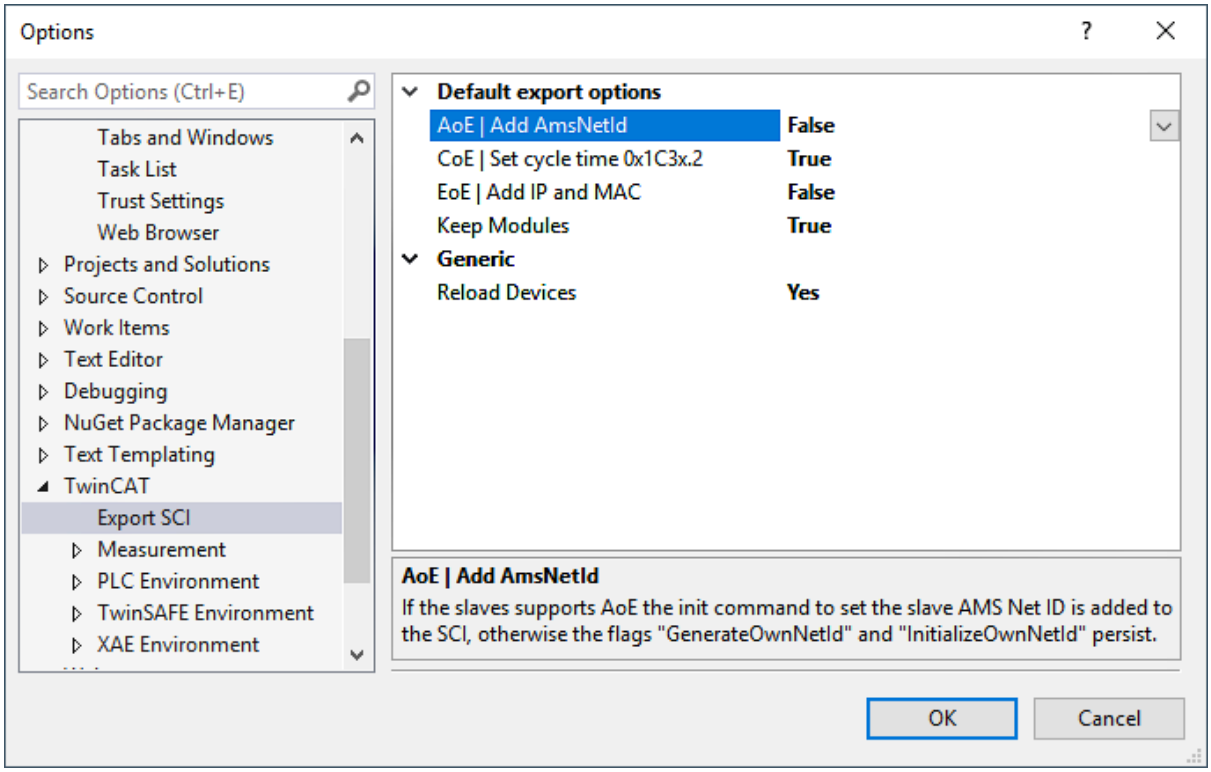


- Display SCI devices and select and insert the desired device:



Additional Notes

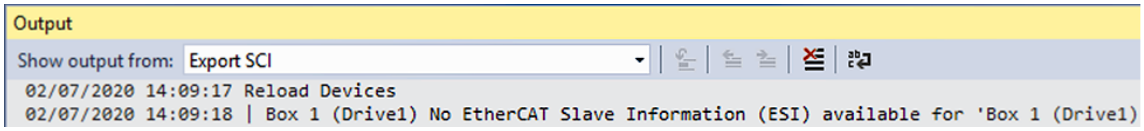
- Settings for the SCI function can be made via the general Options dialog (Tools → Options → TwinCAT → Export SCI):



Explanation of the settings:

Default export options	AoE | Set AmsNetId	Default setting whether the configured AmsNetId is exported.
	CoE | Set cycle time(0x1C3x.2)	Default setting whether the configured cycle time is exported.
	EoE | Set MAC and IP	Default setting whether the configured MAC and IP addresses are exported.
	Keep modules	Default setting whether the modules persist.
Generic	Reload Devices	Setting whether the Reload Devices command is executed before the SCI export. This is strongly recommended to ensure a consistent slave configuration.

SCI error messages are displayed in the TwinCAT logger output window if required:



5.1.3 General Commissioning Instructions for an EtherCAT Slave

This summary briefly deals with a number of aspects of EtherCAT Slave operation under TwinCAT. More detailed information on this may be found in the corresponding sections of, for instance, the EtherCAT System Documentation.

Diagnosis in real time: WorkingCounter, EtherCAT State and Status

Generally speaking an EtherCAT Slave provides a variety of diagnostic information that can be used by the controlling task.

This diagnostic information relates to differing levels of communication. It therefore has a variety of sources, and is also updated at various times.

Any application that relies on I/O data from a fieldbus being correct and up to date must make diagnostic access to the corresponding underlying layers. EtherCAT and the TwinCAT System Manager offer comprehensive diagnostic elements of this kind. Those diagnostic elements that are helpful to the controlling task for diagnosis that is accurate for the current cycle when in operation (not during commissioning) are discussed below.

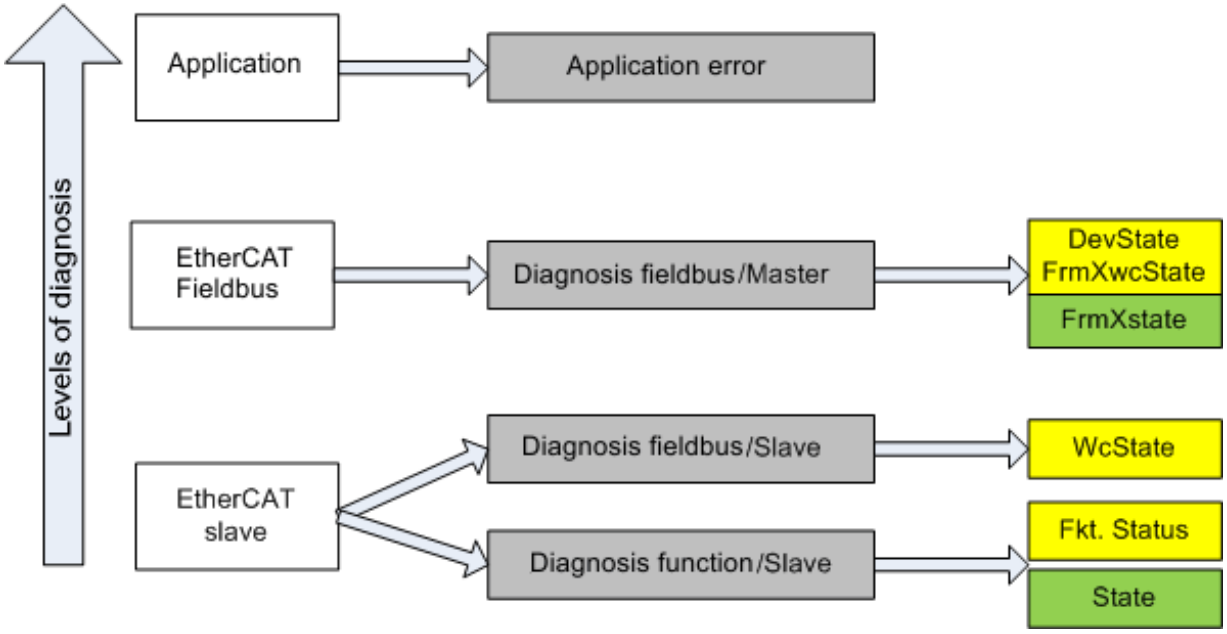


Fig. 137: Selection of the diagnostic information of an EtherCAT Slave

In general, an EtherCAT Slave offers

- communication diagnosis typical for a slave (diagnosis of successful participation in the exchange of process data, and correct operating mode)
  This diagnosis is the same for all slaves.

  as well as
- function diagnosis typical for a channel (device-dependent)
  See the corresponding device documentation

The colors in Fig. *Selection of the diagnostic information of an EtherCAT Slave* also correspond to the variable colors in the System Manager, see Fig. *Basic EtherCAT Slave Diagnosis in the PLC*.

Colour	Meaning
yellow	Input variables from the Slave to the EtherCAT Master, updated in every cycle
red	Output variables from the Slave to the EtherCAT Master, updated in every cycle
green	Information variables for the EtherCAT Master that are updated acyclically. This means that it is possible that in any particular cycle they do not represent the latest possible status. It is therefore useful to read such variables through ADS.

Fig. *Basic EtherCAT Slave Diagnosis in the PLC* shows an example of an implementation of basic EtherCAT Slave Diagnosis. A Beckhoff EL3102 (2-channel analogue input terminal) is used here, as it offers both the communication diagnosis typical of a slave and the functional diagnosis that is specific to a channel. Structures are created as input variables in the PLC, each corresponding to the process image.

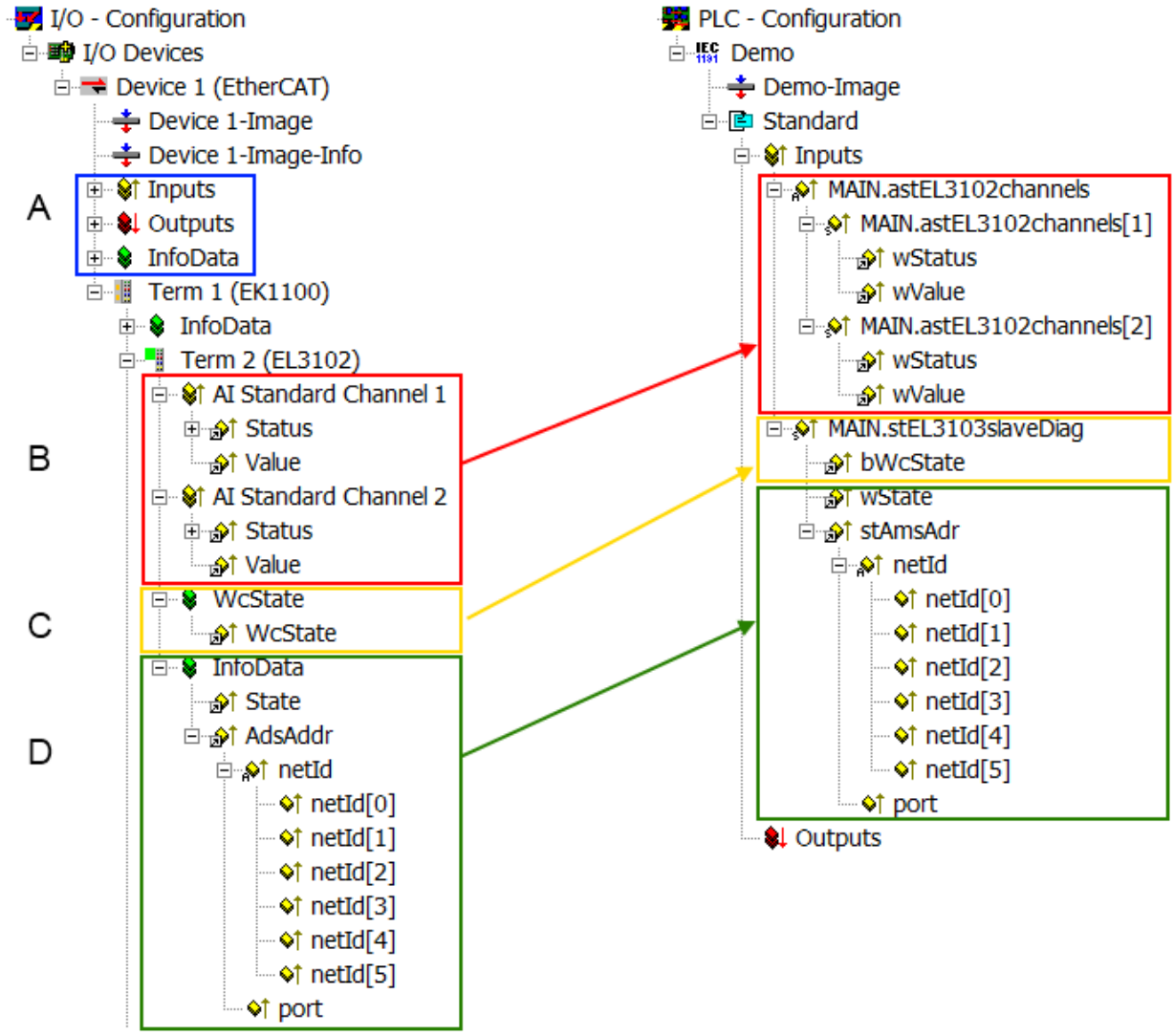


Fig. 138: Basic EtherCAT Slave Diagnosis in the PLC

The following aspects are covered here:

Code	Function	Implementation	Application/evaluation
A	The EtherCAT Master's diagnostic information updated cyclically (yellow) or provided acyclically (green).		At least the DevState is to be evaluated for the most recent cycle in the PLC. The EtherCAT Master's diagnostic information offers many more possibilities than are treated in the EtherCAT System Documentation. A few keywords: • CoE in the Master for communication with/through the Slaves • Functions from *TcEtherCAT.lib* • Perform an OnlineScan
B	In the example chosen (EL3102) the EL3102 comprises two analogue input channels that transmit a single function status for the most recent cycle.	Status • the bit significations may be found in the device documentation • other devices may supply more information, or none that is typical of a slave	In order for the higher-level PLC task (or corresponding control applications) to be able to rely on correct data, the function status must be evaluated there. Such information is therefore provided with the process data for the most recent cycle.
C	For every EtherCAT Slave that has cyclic process data, the Master displays, using what is known as a WorkingCounter, whether the slave is participating successfully and without error in the cyclic exchange of process data. This important, elementary information is therefore provided for the most recent cycle in the System Manager 1. at the EtherCAT Slave, and, with identical contents 2. as a collective variable at the EtherCAT Master (see Point A) for linking.	WcState (Working Counter) 0: valid real-time communication in the last cycle 1: invalid real-time communication This may possibly have effects on the process data of other Slaves that are located in the same SyncUnit	In order for the higher-level PLC task (or corresponding control applications) to be able to rely on correct data, the communication status of the EtherCAT Slave must be evaluated there. Such information is therefore provided with the process data for the most recent cycle.
D	Diagnostic information of the EtherCAT Master which, while it is represented at the slave for linking, is actually determined by the Master for the Slave concerned and represented there. This information cannot be characterized as real-time, because it • is only rarely/never changed, except when the system starts up • is itself determined acyclically (e.g. EtherCAT Status)	State current Status (INIT..OP) of the Slave. The Slave must be in OP (=8) when operating normally. *AdsAddr* The ADS address is useful for communicating from the PLC/task via ADS with the EtherCAT Slave, e.g. for reading/writing to the CoE. The AMS-NetID of a slave corresponds to the AMS-NetID of the EtherCAT Master; communication with the individual Slave is possible via the *port* (= EtherCAT address).	Information variables for the EtherCAT Master that are updated acyclically. This means that it is possible that in any particular cycle they do not represent the latest possible status. It is therefore possible to read such variables through ADS.

NOTICE

Diagnostic information

It is strongly recommended that the diagnostic information made available is evaluated so that the application can react accordingly.

CoE Parameter Directory

The CoE parameter directory (CanOpen-over-EtherCAT) is used to manage the set values for the slave concerned. Changes may, in some circumstances, have to be made here when commissioning a relatively complex EtherCAT Slave. It can be accessed through the TwinCAT System Manager, see Fig. *EL3102, CoE directory*:

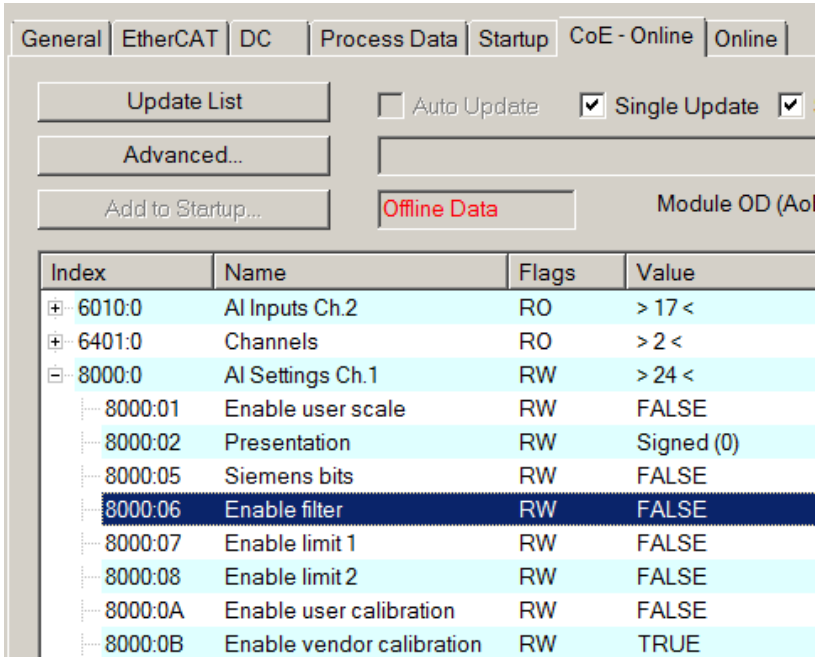


Fig. 139: EL3102, CoE directory

EtherCAT System Documentation

The comprehensive description in the EtherCAT System Documentation (EtherCAT Basics --> CoE Interface) must be observed!

A few brief extracts:

- Whether changes in the online directory are saved locally in the slave depends on the device. EL terminals (except the EL66xx) are able to save in this way.
- The user must manage the changes to the StartUp list.

Commissioning aid in the TwinCAT System Manager

Commissioning interfaces are being introduced as part of an ongoing process for EL/EP EtherCAT devices. These are available in TwinCAT System Managers from TwinCAT 2.11R2 and above. They are integrated into the System Manager through appropriately extended ESI configuration files.

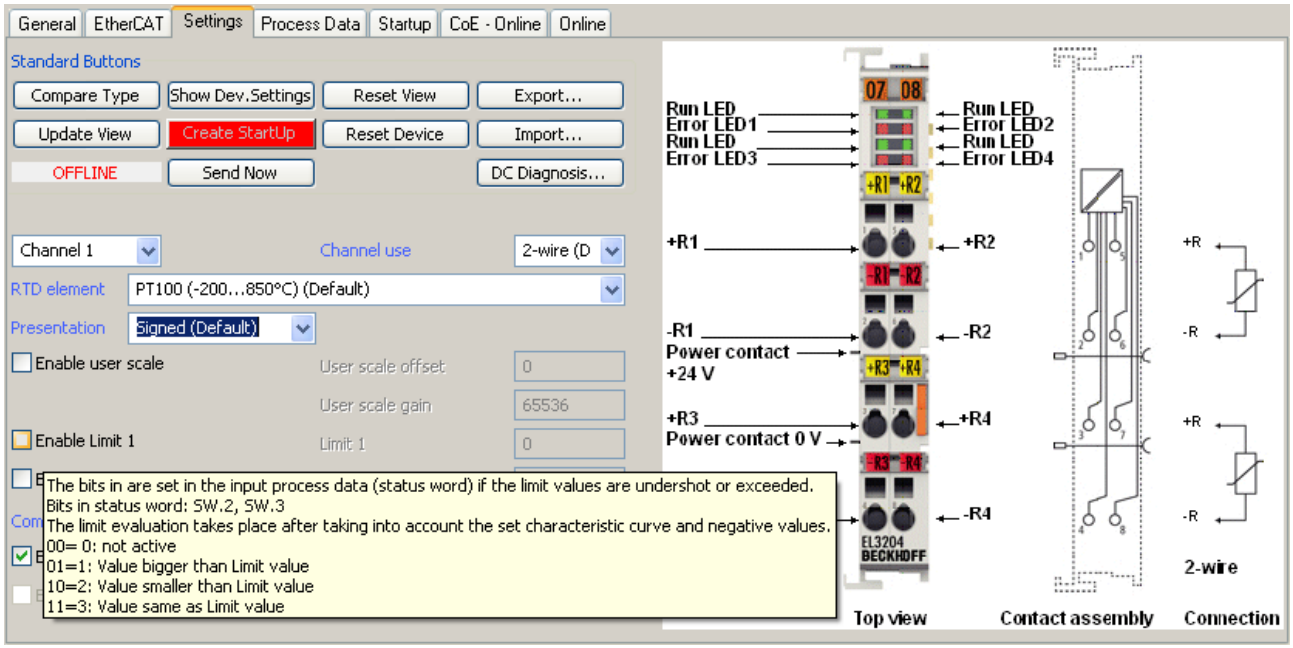


Fig. 140: Example of commissioning aid for a EL3204

This commissioning process simultaneously manages

- CoE Parameter Directory
- DC/FreeRun mode
- the available process data records (PDO)

Although the “Process Data”, “DC”, “Startup” and “CoE-Online” that used to be necessary for this are still displayed, it is recommended that, if the commissioning aid is used, the automatically generated settings are not changed by it.

The commissioning tool does not cover every possible application of an EL/EP device. If the available setting options are not adequate, the user can make the DC, PDO and CoE settings manually, as in the past.

EtherCAT State: automatic default behaviour of the TwinCAT System Manager and manual operation

After the operating power is switched on, an EtherCAT Slave must go through the following statuses

- INIT
- PREOP
- SAFEOP
- OP

to ensure sound operation. The EtherCAT Master directs these statuses in accordance with the initialization routines that are defined for commissioning the device by the ES/XML and user settings (Distributed Clocks (DC), PDO, CoE). See also the section on "Principles of Communication, EtherCAT State Machine [▶ 33]" in this connection. Depending how much configuration has to be done, and on the overall communication, booting can take up to a few seconds.

The EtherCAT Master itself must go through these routines when starting, until it has reached at least the OP target state.

The target state wanted by the user, and which is brought about automatically at start-up by TwinCAT, can be set in the System Manager. As soon as TwinCAT reaches the status RUN, the TwinCAT EtherCAT Master will approach the target states.

Standard setting

The advanced settings of the EtherCAT Master are set as standard:

- EtherCAT Master: OP
- Slaves: OP
  This setting applies equally to all Slaves.

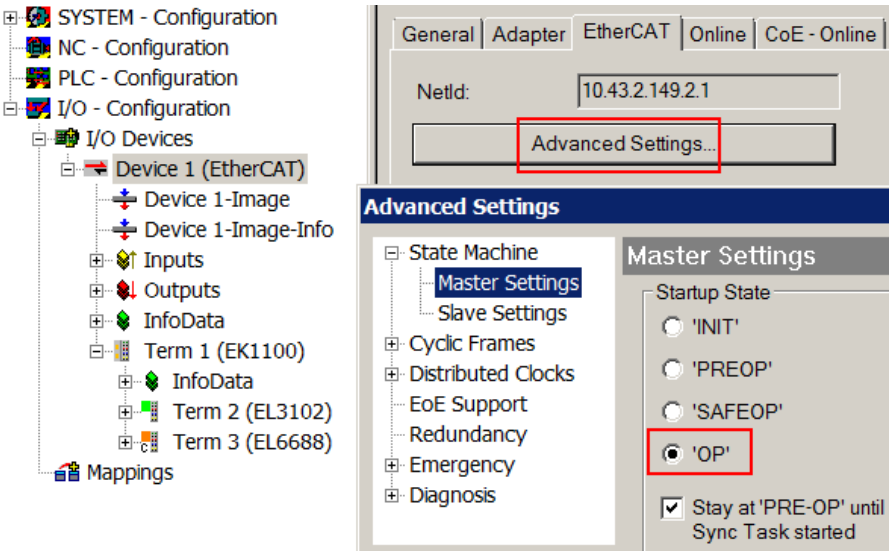


Fig. 141: Default behaviour of the System Manager

In addition, the target state of any particular Slave can be set in the “Advanced Settings” dialogue; the standard setting is again OP.

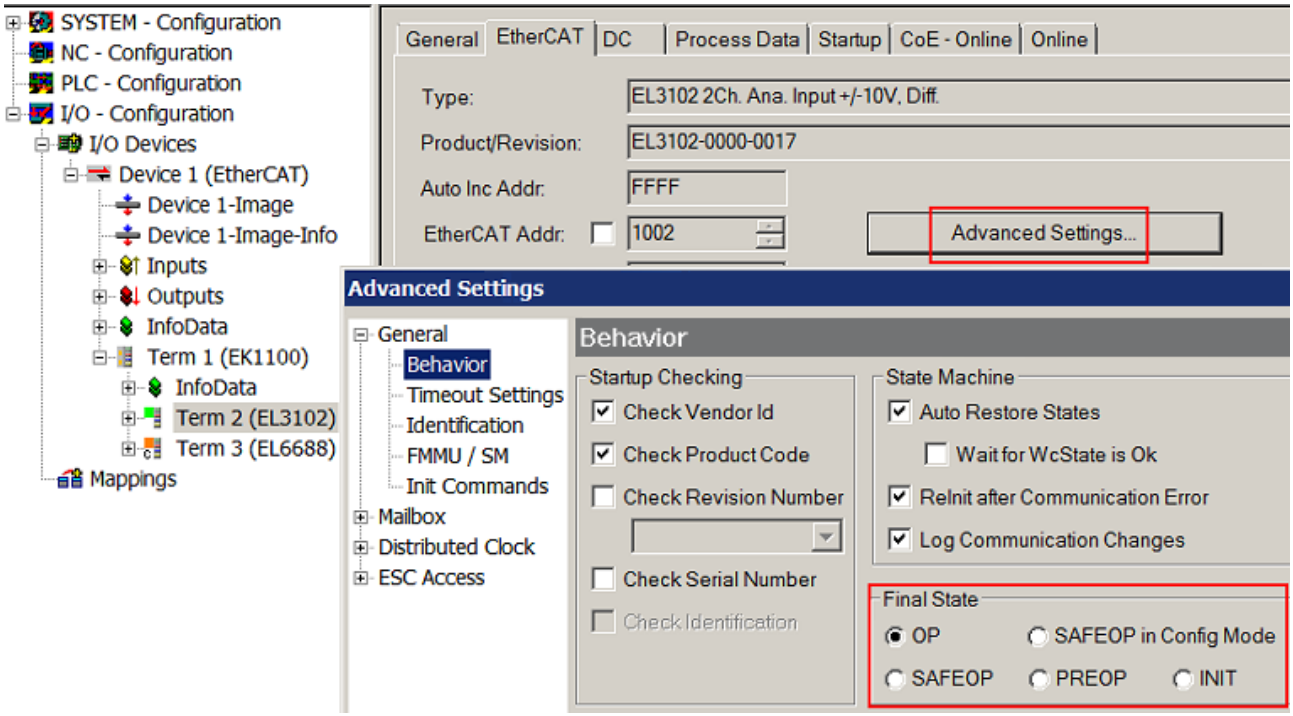


Fig. 142: Default target state in the Slave

Manual Control

There are particular reasons why it may be appropriate to control the states from the application/task/PLC. For instance:

- for diagnostic reasons
- to induce a controlled restart of axes
- because a change in the times involved in starting is desirable

In that case it is appropriate in the PLC application to use the PLC function blocks from the *TcEtherCAT.lib*, which is available as standard, and to work through the states in a controlled manner using, for instance, *FB_EcSetMasterState*.

It is then useful to put the settings in the EtherCAT Master to INIT for master and slave.

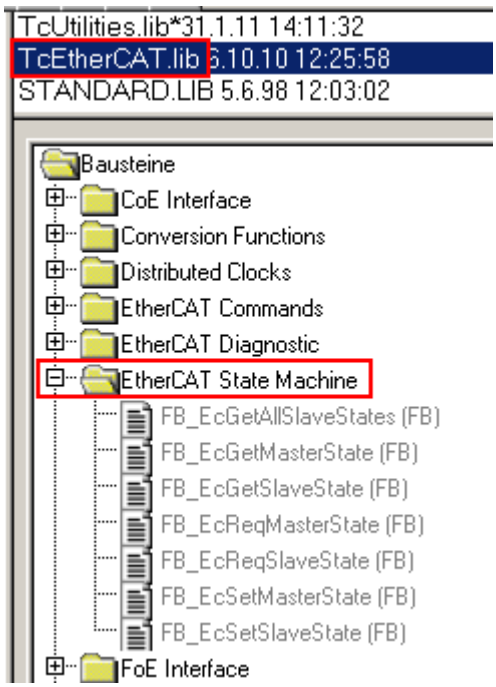


Fig. 143: PLC function blocks

Note regarding E-Bus current

EL/ES terminals are placed on the DIN rail at a coupler on the terminal strand. A Bus Coupler can supply the EL terminals added to it with the E-bus system voltage of 5 V; a coupler is thereby loadable up to 2 A as a rule. Information on how much current each EL terminal requires from the E-bus supply is available online and in the catalogue. If the added terminals require more current than the coupler can supply, then power feed terminals (e.g. EL9410) must be inserted at appropriate places in the terminal strand.

The pre-calculated theoretical maximum E-Bus current is displayed in the TwinCAT System Manager as a column value. A shortfall is marked by a negative total amount and an exclamation mark; a power feed terminal is to be placed before such a position.

General | Adapter | EtherCAT | Online | CoE - Online

NetId: 10.43.2.149.2.1 Advanced Settings...

Number	Box Name	Address	Type	In Size	Out S...	E-Bus (..
1	Term 1 (EK1100)	1001	EK1100			
2	Term 2 (EL3102)	1002	EL3102	8.0		1830
3	Term 4 (EL2004)	1003	EL2004		0.4	1730
4	Term 5 (EL2004)	1004	EL2004		0.4	1630
5	Term 6 (EL7031)	1005	EL7031	8.0	8.0	1510
6	Term 7 (EL2808)	1006	EL2808		1.0	1400
7	Term 8 (EL3602)	1007	EL3602	12.0		1210
8	Term 9 (EL3602)	1008	EL3602	12.0		1020
9	Term 10 (EL3602)	1009	EL3602	12.0		830
10	Term 11 (EL3602)	1010	EL3602	12.0		640
11	Term 12 (EL3602)	1011	EL3602	12.0		450
12	Term 13 (EL3602)	1012	EL3602	12.0		260
13	Term 14 (EL3602)	1013	EL3602	12.0		70
14	Term 3 (EL6688)	1014	EL6688	22.0		-240 !

Fig. 144: Illegally exceeding the E-Bus current

From TwinCAT 2.11 and above, a warning message “E-Bus Power of Terminal...” is output in the logger window when such a configuration is activated:

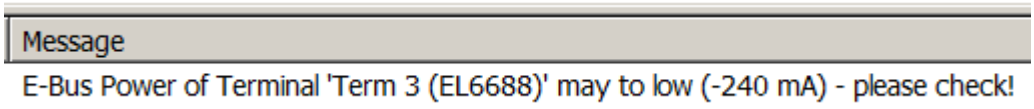


Fig. 145: Warning message for exceeding E-Bus current

NOTICE

Caution! Malfunction possible!

The same ground potential must be used for the E-Bus supply of all EtherCAT terminals in a terminal block!

5.2 Basic Function Principles

The basic function and possible settings of the EL407x family are described below.

Superordinate notes can be found in the system description.

5.2.1 Commissioning EL407x

5.2.1.1 General information on commissioning the EL407x

- The EL407x terminals EL4072, EL4074, EL4078 are functionally identical except for the number of channels; the EL4072 is used as an example below.
- The output channels are set by default to Interface ±10 V, Floating Point, Extended Range.

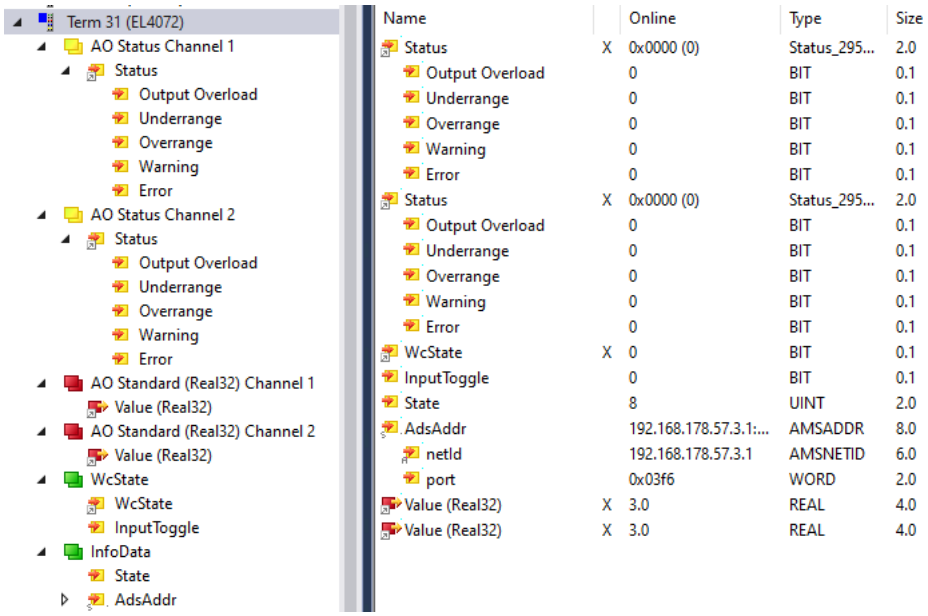


Fig. 146: TwinCAT tree EL4072, online values as an example

- Temporal operating behavior of the EL407x
  The terminals support the following EtherCAT operation modes
  - FreeRun: no
  - SM-synchron (frame triggered): yes
  - DistributedClocks: no

The terminals support EtherCAT cycle times of up to 50 µs, but the following minimum cycle times are recommended for precise value output:
- EL4072: >= 150 µs
- EL4074: >= 250 µs
- EL4078: >= 500 µs

If these values are not reached, it cannot be guaranteed that analog value output will begin in the corresponding output cycle. This may result in delayed output in the order of several cycles. The fact that the internal analog value output can no longer follow the cycle reliably can be recognized by the incrementing of the local “SM Output” error counter CoE 0x1C32:0C.

Example EL4078:

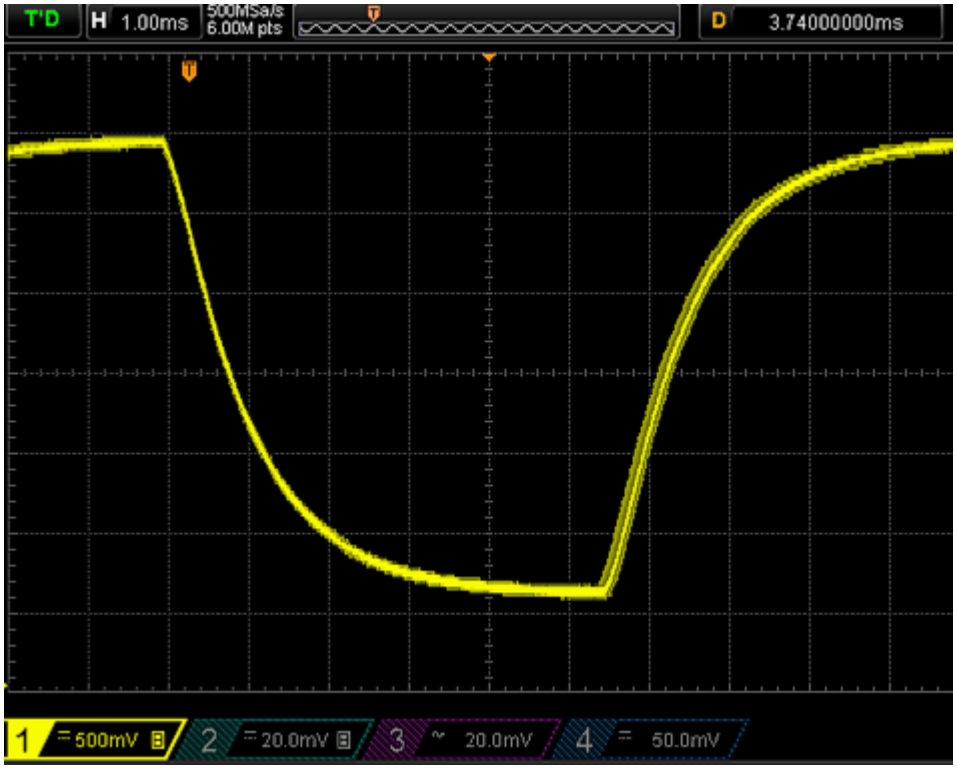


Fig. 147: EL4078 with 500 µs EtherCAT cycle time, output jitter according to theory ±0.5 cycles



Fig. 148: EL4078 with 100 µs EtherCAT cycle time, output jitter > 1 ms

If such time-precise analog value output is required, switch to the EL41xx or EL47xx series.

- The terminal contains several functions that are controlled
  - by enabling/disabling cyclic process data (PDO) at commissioning time in the EtherCAT master
  - by setting/changing so-called CoE settings in the device, this can be done during runtime

  The functions of this terminal are explained in the following commissioning chapter.
- for CoE parameterization:
  - CoE registers are named below, the "n" in the index stands for the channel number in the terminal: Channel 1: n=0, channel 2: n=1, ...
  - The CoE values cannot be read in real-time; they are updated in the online display in TwinCAT approx. every second, or can be get (or written, depending on the type) from the controller via ADS with an update rate of up to a few ms (depending on the EtherCAT cycle time).

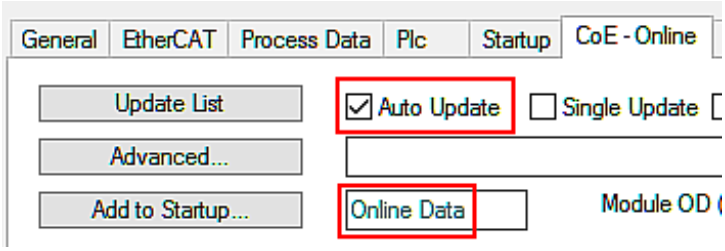


Fig. 149: CoE online tab

  - Before initial commissioning, it may be useful to reset the CoE parameters of the terminal using CoE reset by entering the value after 0x1011:01; this covers all channels
    (see chapter "Restoring the delivery state [▶ 204]")
- "Default" or "Default setting" means that the setting named in this way is the basic setting ex factory or CoE reset.
  -

5.2.1.2 Device diagnostic functions

The following EL407x device information can be read from the CoE:

Index	Name	Meaning
0xF900:01	Power Good	FALSE: There is approx. < 20.4 V at the power contacts. Operation of the device below this limit is not recommended. Overvoltage is not monitored.
0xF900:11	Operating Time	operating time of the device in [min], cannot be deleted
0xF900:12	Device Temperature	current internal terminal temperature in [°C]. Note: this value depends on the installation position, it is usually well above the ambient temperature.
0xF900:13	Min. Device Temperature	minimum value ever observed by the terminal in [°C], cannot be deleted
0xF900:14	Max. Device Temperature	maximum value ever observed by the terminal in [°C], cannot be deleted

The status of the 5 terminal LEDs can be read electronically as follows:

Index	Name	Meaning
0xF915:01	RUN	RUN-LED
0xF915:09	Error Ch.1	LED channel 1 (AI)
0xF915:0A	Error Ch.2	LED channel 2 (AI)
0xF915:0B	Error Ch.3	LED channel 3 (AO) (if present)
0xF915:0C	Error Ch.4	LED channel 4 (AO) (if present)
0xF915:0D	Error Ch.5	LED channel 5 (AO) (if present)
0xF915:0E	Error Ch.6	LED channel 6 (AO) (if present)
0xF915:0F	Error Ch.7	LED channel 7 (AO) (if present)
0xF915:10	Error Ch.8	LED channel 8 (AO) (if present)

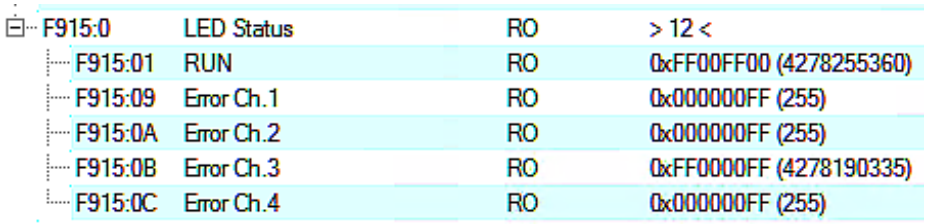


Fig. 150: Subindices Index F915, example values EL4074

The status of the optical displays (LEDs) in the device can be read out electronically in CoE 0xF915 LED Status, e.g. for simultaneous LED display in the visualization.

These are four bytes that describe the RGB value and the light status:

```
0x 12 34 56 78
   |  |  |  \---- Red
   |  |  \------- Green
   |  \---------- Blue
   \------------- Blink
```

- Byte 1 (from left to right): Flashing/lighting code
  - 0x00: Off/ not available
  - 0x01…0x14: 1..20 Hz
  - 0x80: EtherCAT PreOp
  - 0x81: EtherCAT SafeOp
  - 0x82: EtherCAT Boot
  - 0xFF: On/ available

- Byte 2..4:
  - 0x00: Off
  - 0xFF: On

Examples:

- 0x 00 00 00 00: LED not present
- 0x FF 00 00 00 : LED is on, RGB =0, i.e. not illuminated, meaning: LED is present

```
0x 00 00 00 FF : LED off (Red)
0x 00 00 FF 00 : LED off (Green)
0x 00 FF 00 00 : LED off (Blue)
0x 00 00 FF FF : LED off (Yellow)
0x 00 FF FF FF : LED off (White)
```

```
0x FF 00 00 FF : LED on  (Red)
0x FF 00 FF 00 : LED on  (Green)
0x FF FF 00 00 : LED on  (Blue)
0x FF 00 FF FF : LED on  (Yellow)
0x FF FF FF FF : LED on  (White)
```

Fig. 151: Examples LED status

5.2.1.3 Diag-Messages

The terminal provides the following diagnostic messages:

Diag-Code	Meaning	TwinCAT Message	Troubleshooting
#x0001	no error	no error	-
#x1180	Supply voltage ok	Supply voltage ok	-
#x1707	Channel is not in overload anymore	Channel is not in overload anymore	-
#x1708	Channel %d is not in overload anymore	Channel %d is not in overload anymore	-
#x170E	No overtemperature anymore	No overtemperature anymore	-
#x170F	No overtemperature on channel %d anymore	No overtemperature on channel %d anymore	-
#x1710	Full voltage range is available again	Full voltage range is available again	-
#x4101	Terminal-Overtemperature	Terminal-Overtemperature	Device is too warm internally, but continues to work, ensure cooling
#x4102	Overtemperature on channel %d	Overtemperature on channel %d	Device is too warm internally, but continues to work, ensure cooling
#x4710	More than four channels configured for current output, voltage output is limited to ±5 V	More than four channels configured for current output, voltage output is limited to ±5 V	If > ±5 V is required on the U channels, other outputs/devices must be used to output the current values.
#x8144	Hardware fault %d	Hardware fault %d	Check load, measure output function externally, contact support if necessary
#x8601	Supply voltage too low	Supply voltage too low	Check voltage of power contacts Up
#x8707	Channel overload (Channel %d)	Channel overload (Channel %d)	Check wiring and load, possibly short circuit/ OpenLoad?
#x870A	Range error of channel %d	Range error of channel %d	External voltage applied to the channel?

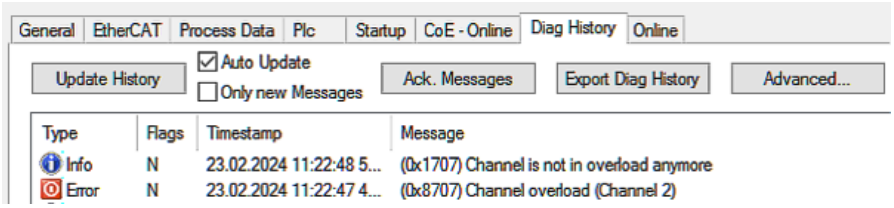


Fig. 152: Display in TwinCAT 3.1

The error states are expressed as follows

Error	PDO Status.Overrange Status.Underrange	PDO Status.Error (channel by channel)	PDO Status.Warning (channel by channel)	DiagMsg	Error-LED (channel by channel)
Supply UP (power contacts) too low/too high	No	No	No	Yes	All channels
Output channel overtemperature	No	No	Yes	Yes	No
Output channel overload [1)]	No	Yes	No	Yes	Yes
Output channel above/below limiter	Yes	Yes	Yes	No	Yes
Device overtemperature	No	no	All channels	yes	No

[1)] probably short circuit at voltage output and setpoint <> 0 V, probably wire break at current output and setpoint <> 0 mA

5.2.2 Commissioning analog output

5.2.2.1 Instructions for use

Parallel connection in current mode

The current outputs of the EL407x are single-ended and can be connected in parallel to add their currents to the load. Ohm's law must be observed, the internally available output voltage does not increase!

Examples

- EL4072, max. 750 ohm load see technical data -> 15 V output voltage is available internally -> if 100% output (20 mA each) is required for all channels and thus ∑= 40 mA, the load must be < 375 ohm.
- EL4074, max. 500 ohm load see technical data -> 10 V output voltage is available internally -> if 100% output (20 mA each) is required for all channels and thus ∑= 80 mA, the load must be < 125 ohm.
- EL4078, max. 300 ohm load see technical data -> 6 V output voltage is available internally -> if 100% output (20 mA each) is required for all channels and thus ∑= 160 mA, the load must be < 37 ohm.

Output behavior, rise time

You can expect approx. 3 ms for 0->90% and approx. 6 ms for 0->100%

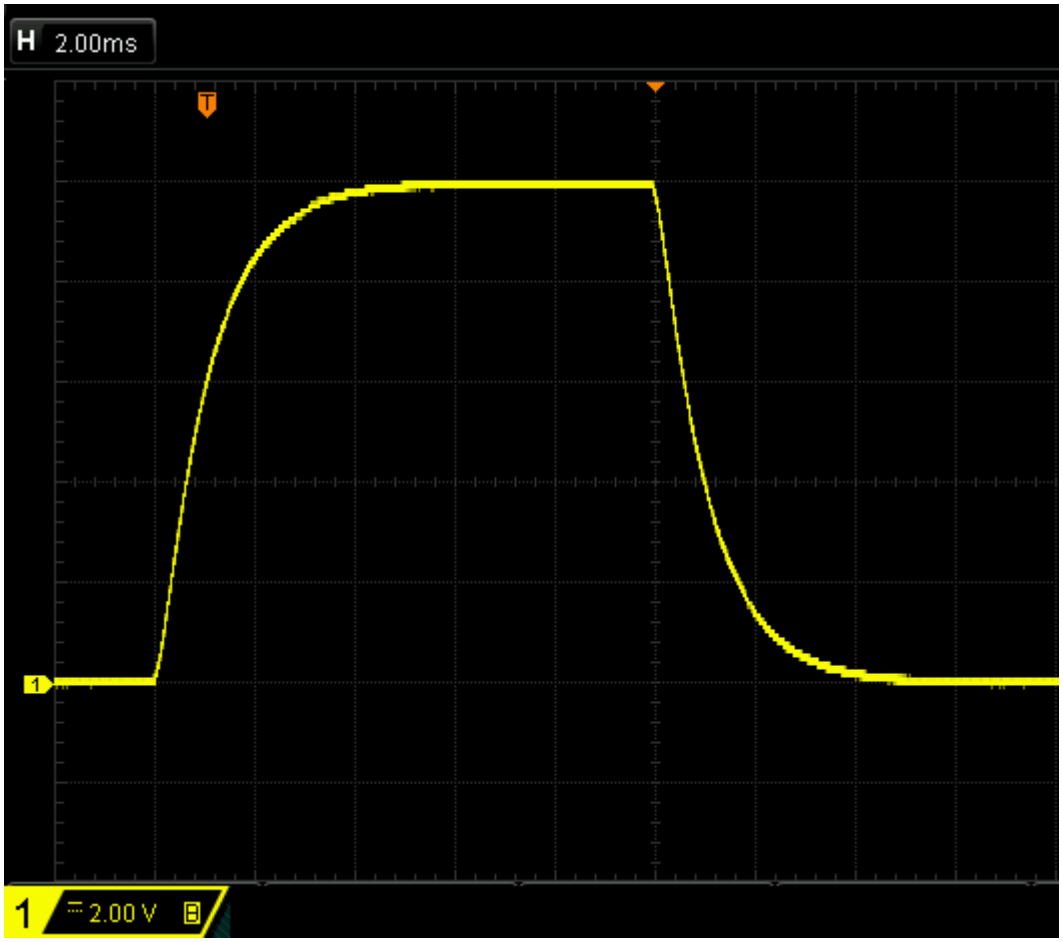


Fig. 153: Rise time representation

Output behavior, dynamics

The output behavior is designed for a dynamic range of 2 ksps/500 µs (setpoint specification). This means that low dynamic signals are output. The output terminals of the EL41xx series are recommended for the output of frequencies above 200 Hz.

Spontaneous output when switching on/off ("glitch")

When the Us supply (E-bus) is switched on or spontaneously switched off and on again, there may be brief, independent value output in the range of a few ±10% of the FSV for a few ms.

When the Up supply (power contacts) is switched on or spontaneously switched off and on again, there may be brief, independent value output in the range of a few ±% of the FSV for a few ms.

It is recommended that sensitive signal receivers only be switched to receive mode when the terminal/ channel

- is in the OP state
- does not report a warning or error in the PDO state

If a glitch-free output is required, output terminals from the EL417x series can be used.

5.2.2.2 Fast commissioning

The analog channel is set ready for operation ex factory with default settings for the 10 V output and FloatingPoint PDO (Real32). Experienced users can read immediately by:

- Scan the terminal in TwinCAT (or add it manually in the configuration, paying attention to the EtherCAT Revision!)
- Reload in TwinCAT

Put into operation and output a value via OnlineWrite/OnlineForce or linked to the PLC:

Name	Online	Type
Value (Real32)	4.256	REAL
Value (Real32)	0.0	REAL

Fig. 154: Output of 4.256 V on channel 1

To gain a deeper understanding of the capabilities of this product, it is recommended that you read the following sections.

5.2.2.3 Commissioning of the analog output

5.2.2.3.1 Data flow AO (Analog Output)

The signal acquisition and data processing of the analog output of this product is as follows:

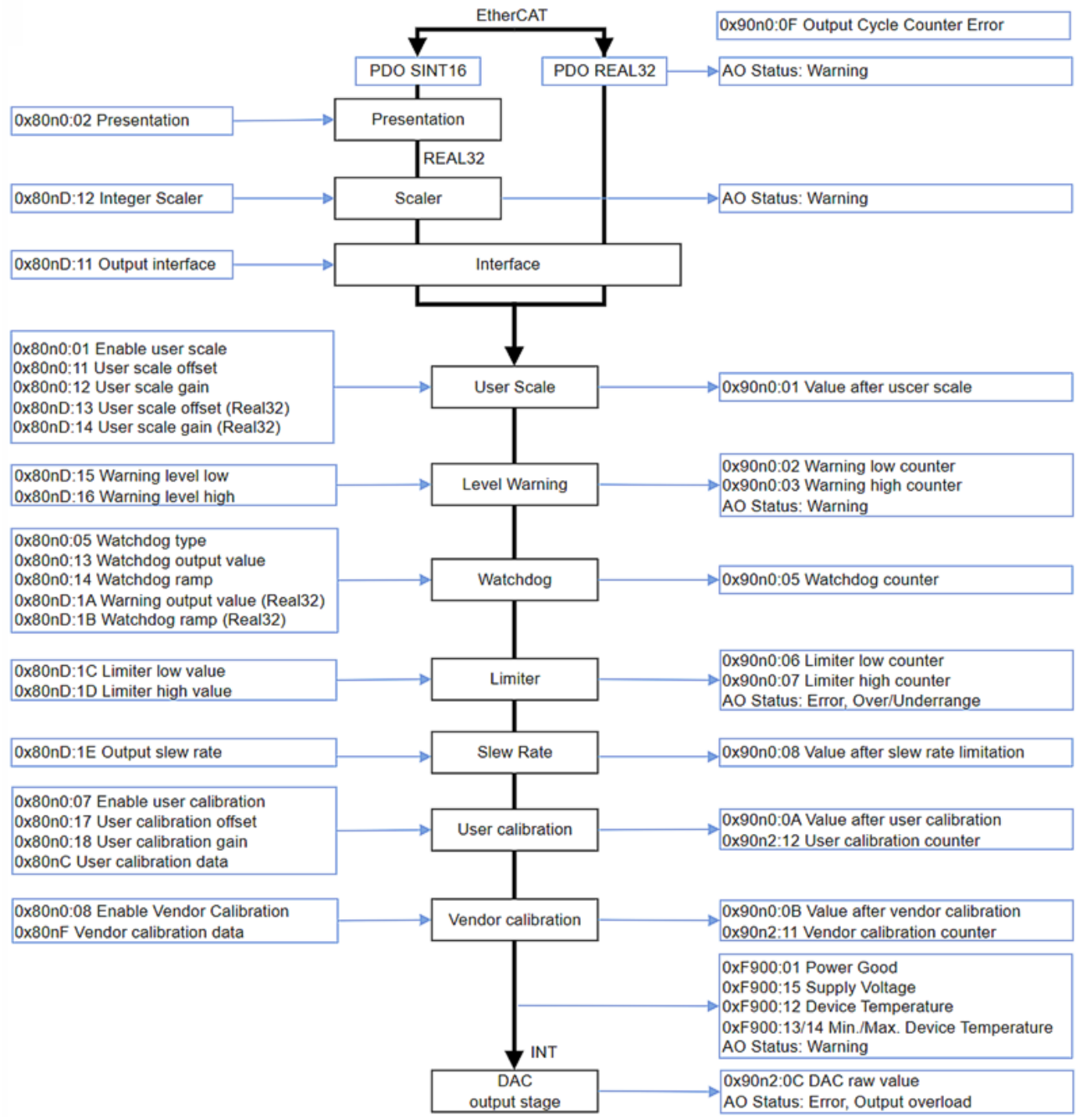


Fig. 155: Data flow of the analog output

Data flow diagram legend	
Left column	Changeable parameters (CoE settings or status PDO) that influence processing
Middle column	Functional units
Right column	Intermediate values and results, displayed in the CoE or status PDO

This terminal calculates internally exclusively in floating point, as shown in the data flow. This considerably simplifies and shortens the commissioning of the analog channel, which minimizes errors in understanding. In addition, intermediate values along the data calculation can be easily displayed in the CoE.

The Real32 and INT16 values are defined in the CoE without units. However, the unit is determined by the context and should, wherever possible, be regarded as an SI unit. For example, the voltage is measured in volts, the current in A (even with 20 mA input!), the resistance in ohms and the ratio in V/V....

Note: Individual functional units (see data flow) have already been introduced in earlier analog devices based on INT16 (integer) and are controlled by these INT-based parameters. Such INT parameters are still supported for compatibility reasons. For example, existing code in the controller should access the CoE via ADS. This means that parameters of functional units are either

- only available as REAL32 types in the CoE if the functional unit was newly introduced with the FloatingPoint data flow, or
- are present in the CoE both as INT type and as REAL32 type with the same meaning, recognizable by the name suffix "(Real32)". The values are automatically mirrored by the firmware when they are changed or taken into account one after the other.

When re-implementing the analog function, it is recommended to use the Real32 parameters.

Commissioning of the analog output in TwinCAT should follow this data flow and is described below.

5.2.2.3.2 Process data format (PDO)

The PDO (Process Data Objects) are the cyclic data of the EtherCAT Subdevice transmitted in real-time, i.e. measured values and status for analog channels, but not parameters.

5.2.2.3.2.1 Process data configuration

The PDOs that can be used for this output channel (A) are described in TwinCAT in the PDO list (B):

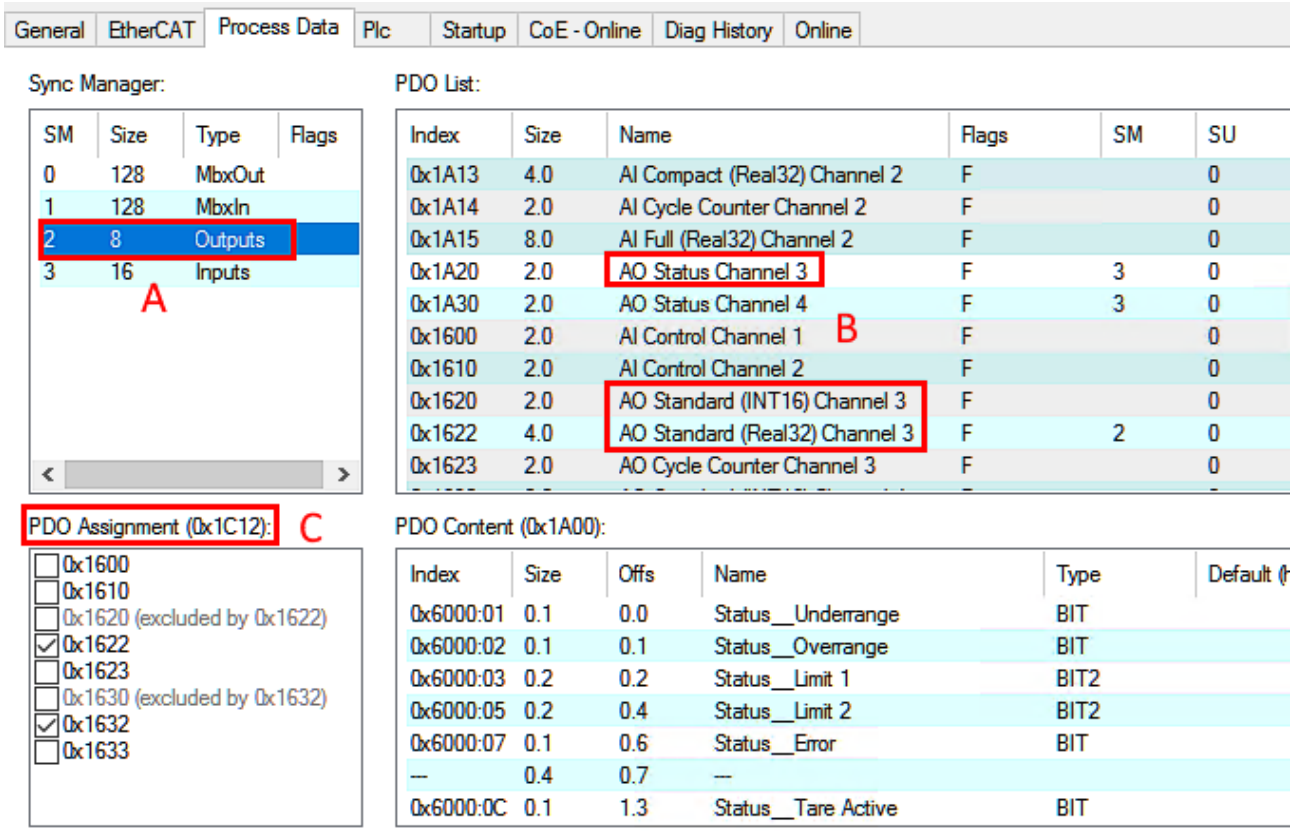


Fig. 156: Tab "Process data"

The desired PDOs can be enabled in (C), exclusions are displayed.

Each channel can be set to one of the above-mentioned process data formats, e.g. 0x1622 must be enabled for Real32 transmission on channel 1 and the control PDO 0x1621 can be added.

Alternatively, the suggested data sets can be selected from the Predefined PDO list:

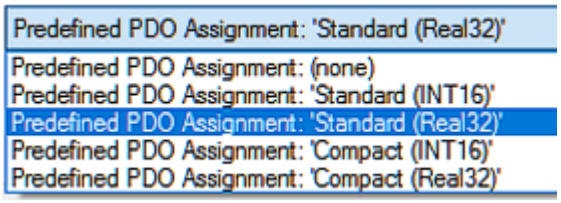


Fig. 157: Predefined PDO

This list summarizes frequently used PDO compositions for convenient choice. It should be noted that these then affect all channels at the same time.

- Standard (INT16) -> see above
- Standard (REAL32) -> see above.
- Compact (INT16) -> see above
- Compact (REAL32) -> see above.

During the PDO changeover, other functional units in the data flow may be reset to the default setting! Therefore, the PDO decision must be made at the beginning; a change requires an *ActivateConfiguration*.

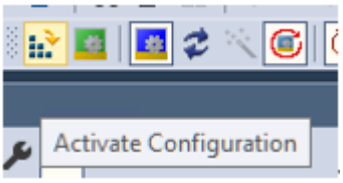


Fig. 158: Button ActivateConfiguration/ReloadDevices

5.2.2.3.2.2 AO Setpoint transport

The following chapters describe the operation of the PDO *Value* (setpoint specification for the analog output channel, Analog Output = AO).

Floating point output (Real32, default setting of the channel)

The channel expects its analog setpoint as a plain text-readable floating point value, both readable in the TwinCAT configuration

Name		Online	Type	Size
Value (Real32)	X	2.3242424	REAL	4.0

Fig. 159: Value (floating point value), TwinCAT

as well as in the PLC Online View:

Expression	Type	Value	Prepared value	Address
rOut1	REAL	7.310994		%Q*

Fig. 160: Value (REAL) in PLC

The Real32 PDO can simply be linked to a REAL variable in PLC:

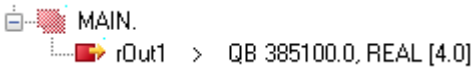


Fig. 161: Linking with REAL variable

This type of transmission avoids scaling errors, as the channel itself takes into account the output range (including any range changes), commissioning and troubleshooting are considerably simplified.

Even if no unit (V, A, Ω, ..) is formally transmitted, the SI unit corresponding to the context must be used, i.e. [A] and not [mA] for a 20 mA input.

If a setpoint outside AEW_{techn} (output end value) is requested by the terminal, it outputs the respective maximum value and displays PDO "AO Status Overrange/Underrange + Warning":

State	Output
Setpoint > AEW_{techn}	"Overange" + "Warning", output AEW_{techn}
Setpoint < AEW_{techn}	"Underrange" + "Warning", output -AEW_{techn}

Integer output (fixed point, INT16 or SINT16)

The channel expects its setpoint as a 16-bit fixed-point value (default incl. sign, signed integer), related to AEW (output end value):

Name		Online	Type	Size
Value	X	19460 <5.939>	INT	2.0

Fig. 162: Value (fixed-point value, "INT)"

The value range extends over -32767 ...0 ... 32768, knowledge of the output range is required for interpretation and transformation on the control side, e.g. 10V ~ x7FFF = 32767 in legacy presentation

If the channel is to be linked with existing PLC code, it can be converted to this INT16 format. Otherwise, the default setting "Real32" is recommended.

As no oversized values can be specified in INT16 format, no warning is evaluated.

5.2.2.3.2.3 AO Status

The output channel also has optional diagnostic data that can be activated as a status word:

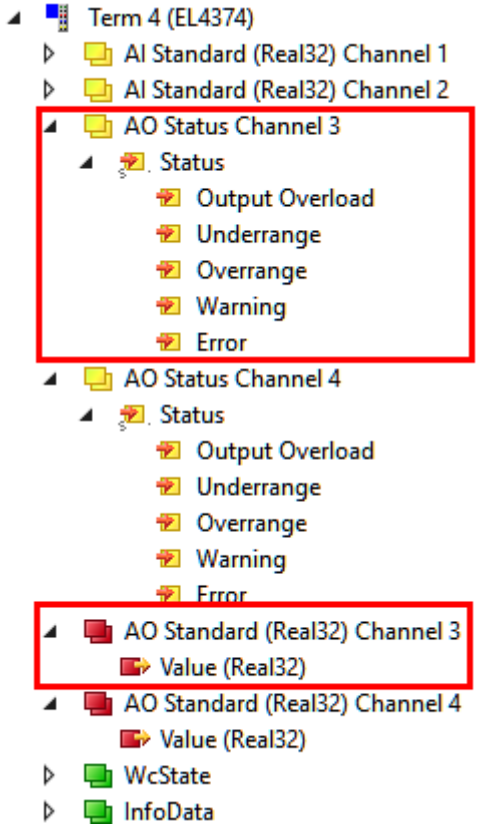


Fig. 163: Status word analog output

Interpretation:

Function [Type]	Output Overload [Bool]	Under-range [Bool]	Over-range [Bool]	Warning [Bool]	Error + Error-LED [Bool]	DiagMessage	Meaning
Bit position (from 0)	SW.1	SW.2	SW.3	SW.5	SW.6		
	x				x	x	Analog output overload: wire break at I output, short circuit at U output
		x			x		Limiter Low limit undershot [▶ 161]
			x		x		Limiter High limit exceeded [▶ 161]
		x		x			Real32 setpoint < AEW_{techn}
			x	x			Real32 setpoint > AEW_{techn}
					x		DAC error
				x		x	Warning Level Low/High exceeded [▶ 156] Internal overtemperature detected -> cooling required Voltage at the power contacts too low (see 0xF900:01/15 Device information: Output operation not protected!)

5.2.2.3.2.4 AO Continuity counter

The output channel should cyclically receive its current setpoint from the controller via EtherCAT, see *PDO Value*.

During the commissioning phase and possibly also during ongoing operation, it may be useful to monitor the regular and timely arrival of the setpoint at the device. For this purpose, the 16-bit counter *Output Cycle Counter* can be enabled channel by channel in the PDO assignment:

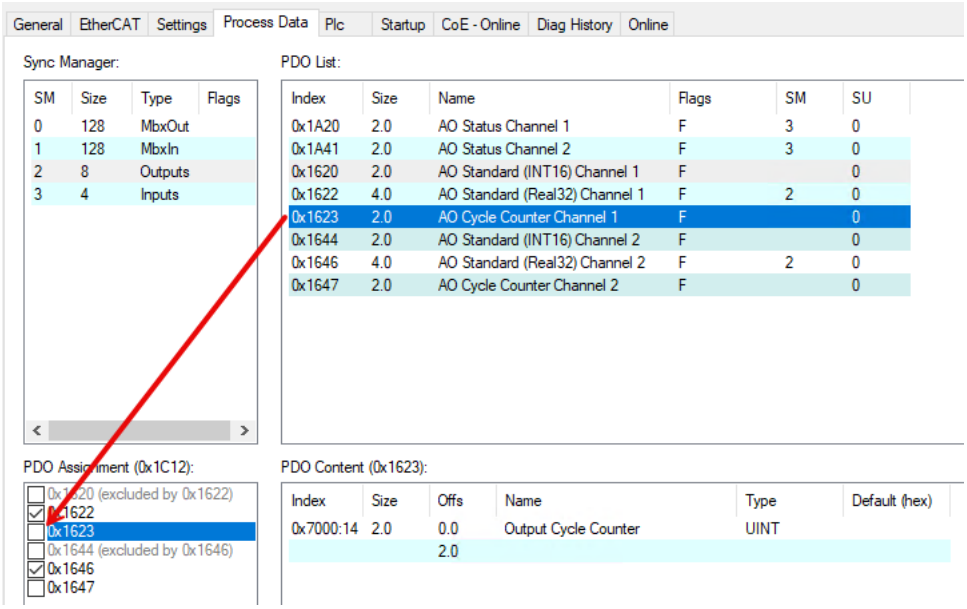
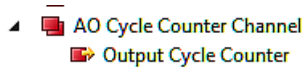


Fig. 164: Enabling 16-bit counter via PDO assignment 0x1623

The respective PDO is then offered for linking in the variable list:



If enabled, the variable is to be operated cyclically with +1 from the controller.
The channel now monitors the counter; if a change > +1 is observed, the channel increments the error counter in CoE 0x90n0:0F by +1. The overflow in the 16-bit value is taken into account. The error counter must be regularly read from the control via ADS and checked.

5.2.2.3.3 Integer scaler (only when using PDO SINT16)

It may be useful to output analog values slightly beyond the nominal output range AEW_{nom}, e.g. to compensate for power losses or to transport diagnostic information. For this reason, the optional Extended Range "107%" has been introduced in Beckhoff analog channels (support depends on the device). The definition for 16 bits is as follows:

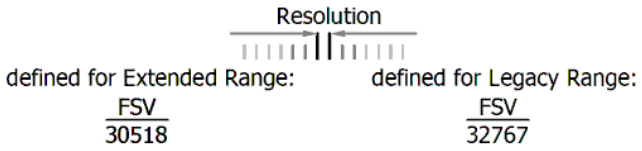


Fig. 165: Defined resolution, 16-bit

Setting:

- Index 80nD:12 = Extended Range Range (default setting)
  The channel operates up to the technical output range AEW_{techn}, which is approx. 107% of the nominal output range.
  For the Extended Range, 16-bit SINT PDO (16 bits + sign) is the nominal. AEW= 100%, the PDO value ±30518 (0x7736) has been set. Accordingly, the displayable output range now extends to 0x7FFF = 32767 ~ 107.37% of the nominal output range.
- Index 80nD:12 = Legacy Range
  The channel operates up to 100% of the nominal output range.
  Accordingly, 0x7FFF = 32767 is to be interpreted as 100% of the nominal AEW.

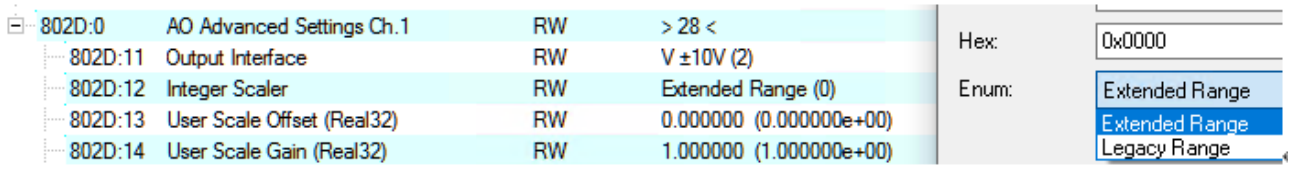


Fig. 166: Setting Index 80nD:12, Legacy Range, Extended Range

Depending on the interface, this means the conversion SINT16 -> Real32 in the controller:

Output range ±10 V (bipolar)

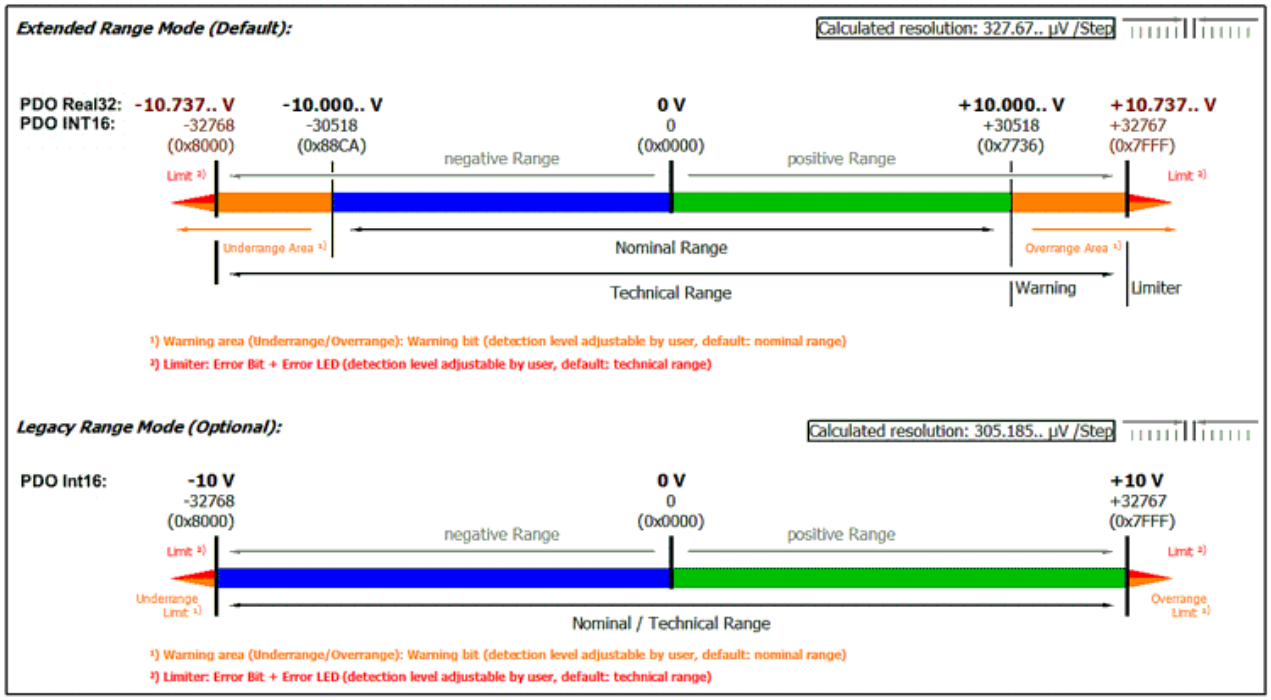


Fig. 167: Output range ±10 V (bipolar)

Output range 0...10 V (unipolar)

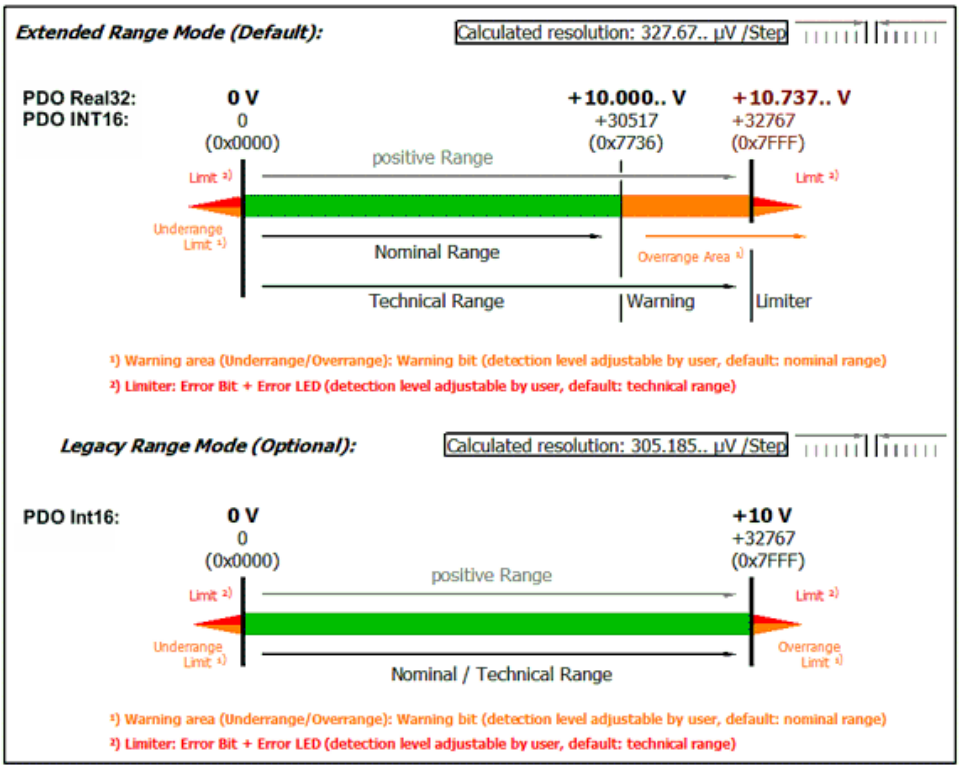


Fig. 168: Output range 0 - 10 V (unipolar)

Output range ±20 mA (bipolar)

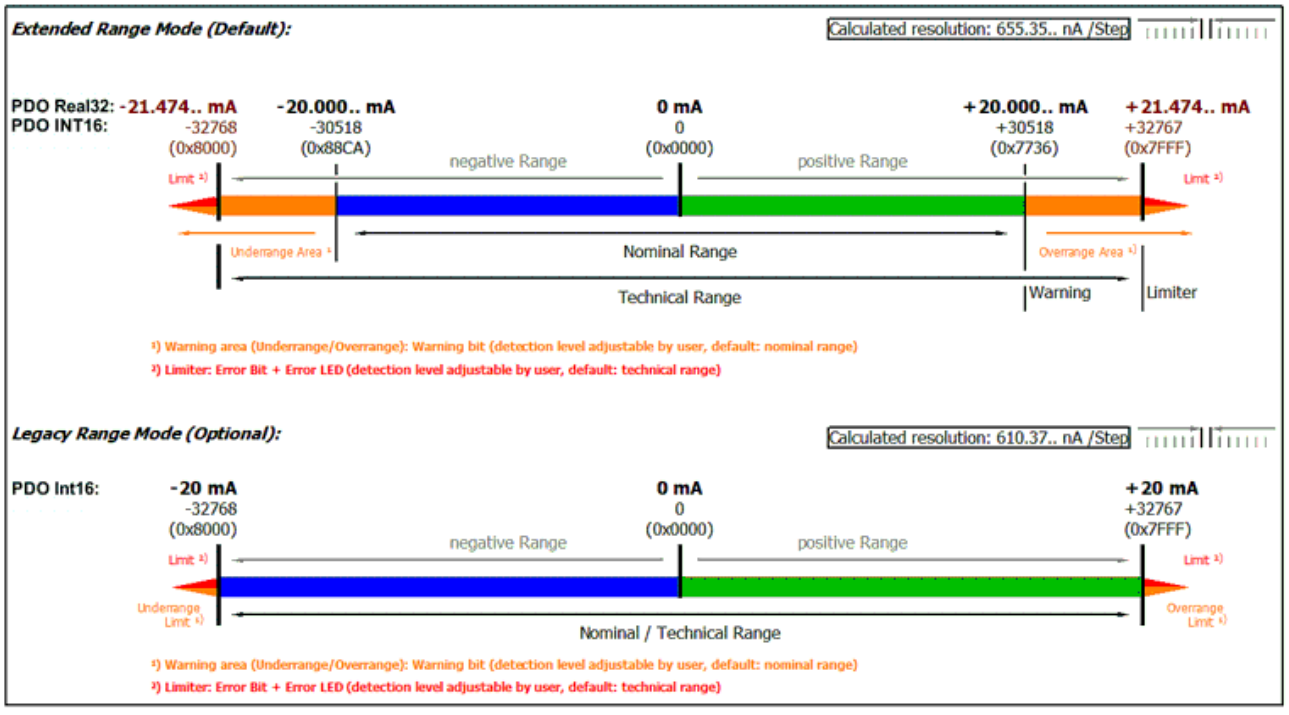


Fig. 169: Output range ±20 mA (bipolar)

Output range 0...20 mA (current loop)

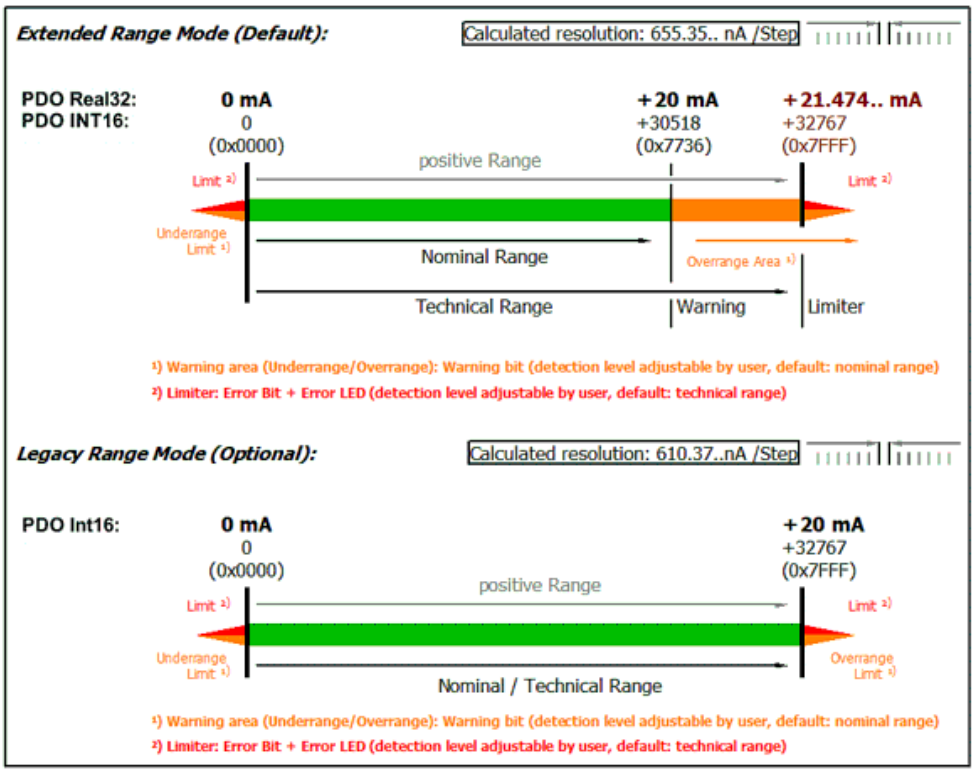


Fig. 170: Output range 0...20 mA (current loop)

Output range 4...20 mA (current loop)

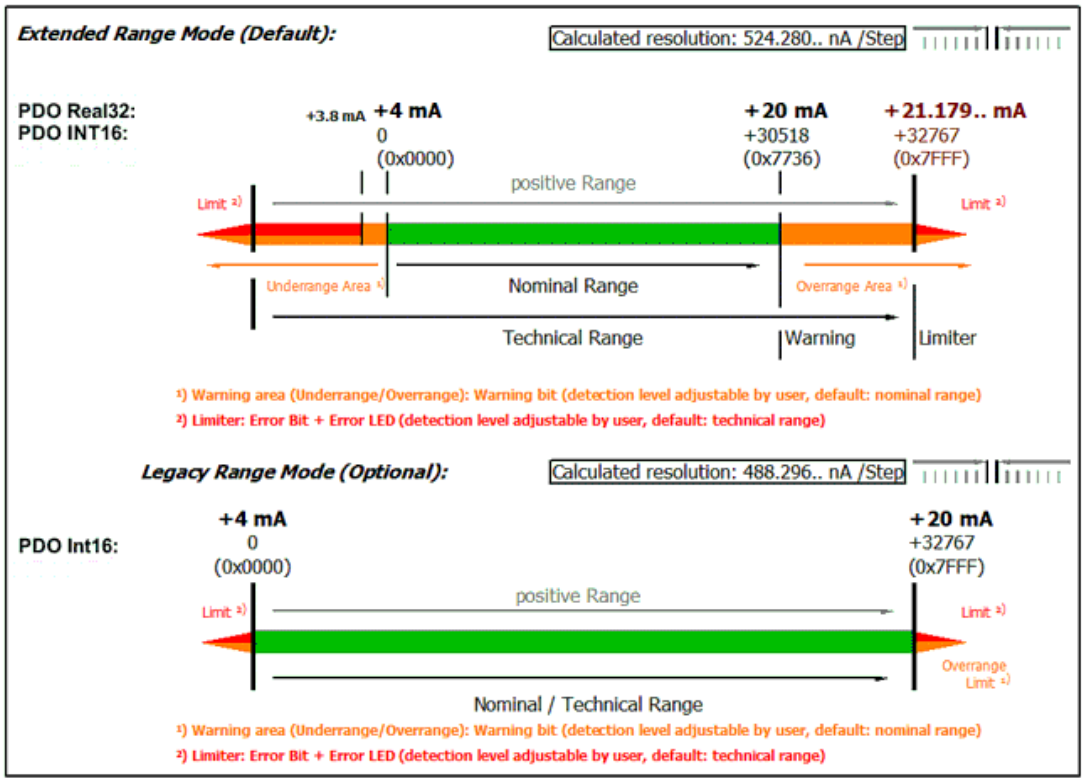


Fig. 171: Output range 4...20 mA (current loop)

5.2.2.3.4 Presentation (only when using PDO SINT16)

For historical reasons, there are various formats in which the 16 bits of the SINT PDO (Signed Integer Process Data Object) can be interpreted.
The format can be set in the index 80n0:02 .

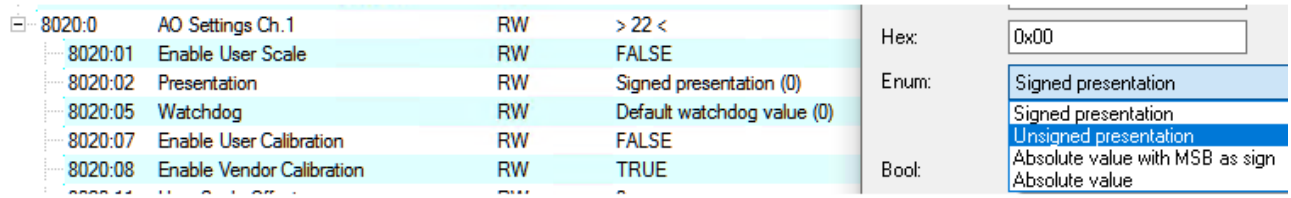


Fig. 172: PDO 80n0:02, "Presentation"

This analog channel supports:

- "Signed" (default): top/highest/0. Bit is sign, negative number in 2's complement in bit 1..15
- "Unsigned": all 16 bits are used for the amount of the analog value, resulting in double resolution for positive analog values. No transmission of negative values possible.
- "Absolute Value with sign": top/highest/0. Bit is sign, bits 1..15 carry the amount of the analog value
- "Absolute Value": the sign of the analog value is ignored, only the (positive) amount in bit 1..15 is transmitted

Legacy Range	Extended Range	Representation (values dec. / values hex.)			
		unsigned integer		Abs. value w. MSB as sign	
		Dec	Hex	Dec	Hex
100%	107.37%	32767	0x7FFF	32767	0x7FFF
-	100%	30518	0x7736	30518	0x7736
0%	0%	0	0x0000	0	0x0000
-	-100%	30518	0x7736	[-30518]	0xF736
-100%	-107.37%	32767	0x7FFF	[-32767]	0xFFFF

Presentation types

The presentation types "Unsigned integer" and "Absolute value with MSB as sign" have no function for unipolar terminals. There is no change in the presentation in the positive range.

5.2.2.3.5 Interface

The interface setting is fundamental for operation as an electrical output.

Setting: CoE 0x80nD:11 Output interface

Setting	Measuring range
None	-
V	±10 V
V	0-10 V
I	±20 mA
I	0-20 mA
I	4-20 mA

Note: When the interface is changed, the following CoE parameters of UserScale, Warning Level, Limiter, Output SlewRate, Watchdog are reset to the default setting.

5.2.2.3.6 Setpoint processing, User Scale

The digital setpoint sent from the controller to the analog output channel must or can be changed in the device in order to

- reinterpret the measured value on the application side (e.g. convert the electrical 0..10 V signal of a pressure sensor into a pressure value)
- compensate for hardware dependency (keyword: calibration)

The output value / setpoint can be changed in 3 functional units, all 3 can be active at the same time:

- User Scale
- User calibration [▶ 164]
- Vendor calibration [▶ 164]

The User Scale functional unit is intended for reinterpretations/transformations of the setpoint, so "50 kg" of the controller can become "10 V" with Gain=0.2. It is implemented as a linear transformation with gain/offset.

Parameter:

Index	Name	Data type	Meaning
80n0:01	Enable User Scale	BOOL	disabled by default, calculation only takes place if TRUE
80n0:11	User Scale Offset	SINT16	is added directly in digits.
80n0:12	User Scale Gain	UINT16	1 bit corresponds to 2^{-16}, so "1" corresponds to $x7FFF/32767_{dec}$
80nD:1D	User Scale Offset (Real32)	REAL32	-
80nD:1E	User Scale Gain (Real32)	REAL32	-

The intermediate value after this functional unit can be viewed in index 90n0:01.

NOTICE

Changing the interface

When changing the interface, the gain and offset are reset to 1 and 0 respectively!

5.2.2.3.7 Level Warning

The output channel makes it possible to return a warning to the controller if a permissible value range is exceeded. This can be used, for example, to detect rare or invalid output values. There is no limit to the output value. The Limiter should be used for this function.

The limits are set to the output range limits by default and after changing the interface.

Parameter:

Index	Designation
80nD:15	Warning Level Low
80nD:16	Warning Level High

Results:

PDO AO Status -> Warning		
Index	**Designation**	**Meaning**
90n0:02	Warning Low Counter	counts +1 when falling below x80nD:15, stored secured against power failure
90n0:03	Warning High Counter	counts +1 when exceeding x80nD:16, stored secured against power failure

The counters are reset by

- an interface change
- or the command x401n after index FB00:01 (channel 1: n=0, channel 2: n=1, ...), the success is displayed with "255" in index FB00:03 Response.
- and the command x4001 "Reset all counters"

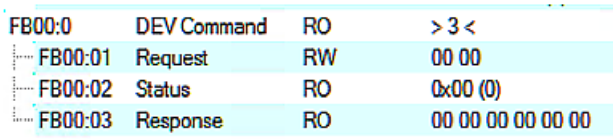


Fig. 173: CoE Index FB00, DEV Command

During command execution, "Status" 255 "busy" is displayed in index FB00:02, "0" means "successfully completed"
The firmware responds to an unknown command with

'Term 5 (EL4374)' (1002): CoE ('InitDown' 0xfb00:01) - SDO Abort ('General parameter incompatibility reason.', 0x06040043).

Fig. 174: General parameter incompatibility reason, 0x06040043

5.2.2.3.8 Watchdog

This output channel is equipped with a safety device (watchdog). This moves the output to a predefined setpoint if process data traffic to the output device is interrupted.

Setting the watchdog time

The watchdog time, i.e. the time at which the watchdog case is triggered, is set via the general TwinCAT dialog "Advanced Settings" -> General -> Behavior -> Watchdog -> "Set Multiplier" and "SM Watchdog" (SM = SyncManager).

Notes on settings

- The setting will only take effect after activating and restarting TwinCAT!
- This setting applies to the entire device (all channels).

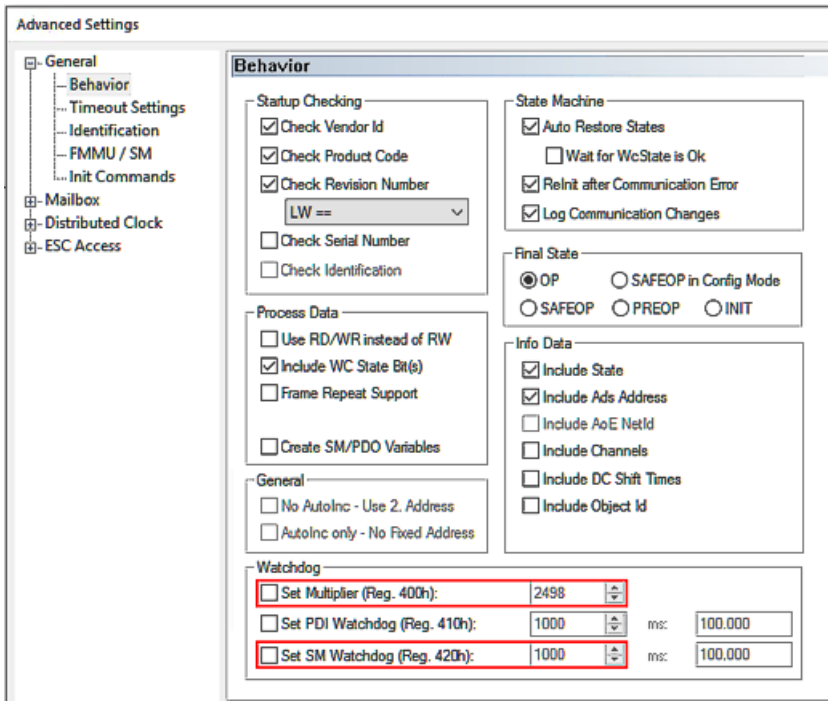


Fig. 175: Watchdog settings, here: 100 ms

Watchdog time [ms] = Multiplier * SM Watchdog [ms]

A maximum watchdog time of 65 s is possible. Larger values are calculated modulo 65, for example 70 s would be shortened to 5 s.

NOTICE

General notes on watchdog settings

Observe the general notes on the watchdog settings.

Sequence of the watchdog case

The sequence is as follows:

- As long as the channel is properly and regularly supplied with EtherCAT process data, they are output. The watchdog observes this without further action; they say "it is being brought up".
- As soon as the data no longer arrives (e.g. due to a wire break, EtherCAT master stopped, etc.), the output value remains at the last value. The watchdog now starts to run down. If data arrive again in time, the watchdog returns to the start value.
  The EtherCAT device remains in the OP state, even if it cannot be reached from the master.
- If the watchdog has expired, i.e. the time set as above has elapsed without new data arriving, the set substitute value is output.
  The EtherCAT devices returns to the Safe-OP state (recognizable by the slow flashing of the EtherCAT RUN-LED if present).
- As soon as new data arrives and the EtherCAT SubDevice has been reset to OP mode by the master (TwinCAT), it is output again and the watchdog resumes monitoring.
- The watchdog value is also output permanently and immediately (without waiting time) if the device leaves the OP state for other reasons.

Setting the watchdog behavior

The following settings can be made for the watchdog, starting from index 0x80n0:05 "Watchdog Type":

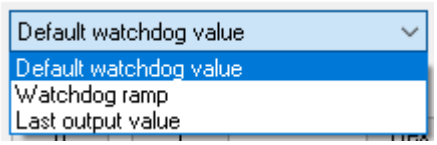


Fig. 176: Selection "Watchdog Type"

Index	Name	Flags	Value	Unit
8020:0	AO Settings Ch.1	RW	> 22 <	
8020:01	Enable User Scale	RW	FALSE	
8020:02	Presentation	RW	Signed presentation (0)	
8020:05	Watchdog Type	RW	Default watchdog value (0)	
8020:07	Enable User Calibration	RW	FALSE	
8020:08	Enable Vendor Calibration	RW	TRUE	
8020:11	User Scale Offset	RW	0	
8020:12	User Scale Gain	RW	65536	
8020:13	Watchdog Output Value	RW	0	
8020:14	Watchdog Ramp	RW	0xFFFF (65535)	
8020:15	User Calibration Offset	RW	0	
8020:16	User Calibration Gain	RW	0x7FFF (32767)	
802C:0	AO User Calibration Data Ch.1	RW	> 13 <	
802D:0	AO Advanced Settings Ch.1	RW	> 30 <	
802D:11	Output Interface	RW	V ±10V (2)	
802D:12	Integer Scaler	RW	Extended Range (0)	
802D:13	User Scale Offset (Real32)	RW	0.000000 (0.000000e+00)	
802D:14	User Scale Gain (Real32)	RW	1.000000 (1.000000e+00)	
802D:15	Warning Level Low	RW	-10.737420 (-1.073742e+01)	
802D:16	Warning Level High	RW	10.737420 (1.073742e+01)	
802D:1A	Watchdog Output Value (Real32)	RW	0.000000 (0.000000e+00)	
802D:1B	Watchdog Ramp (Real32)	RW	0.000000 (0.000000e+00)	s
802D:1C	Limiter Low Value	RW	-10.737420 (-1.073742e+01)	
802D:1D	Limiter High Value	RW	10.737420 (1.073742e+01)	
802D:1E	Output Slew Rate	RW	0.000000 (0.000000e+00)	s

Fig. 177: Indices for watchdog settings

Values Index 0x80nD, “Watchdog Value”	Meaning
Default watchdog value (default)	The analog output value is set without transition to the user-specific substitute value/setpoint according to index 0x80n0:13 or index 0x80nD:1A (default: 0)
Watchdog Ramp	Likewise, substitute value/setpoint according to index 0x80n0:13 or index 0x80nD:1B, but linear ramp to that point. The gradient of the ramp must be specified in relation to AEW_{nom} • per index 0x80n0:14 [digit/ms] • or time duration index 0x80nD:1B [sec]. If, for example, a gradient of 2 V/sec is required with AEW_{nom} = 10 V, the "Watchdog Ramp (Real32)" = 5 [sec] or (with "Extended Range" -> 327 µV/digit) "Watchdog Ramp" = 6 [digit/ms]. Default value: 0 (no ramp)
Last Output value	Last output value remains

Examples

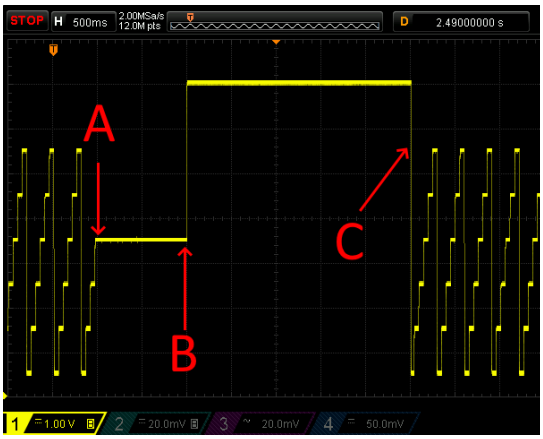
An EL4078 outputs a staircase signal, watchdog time set to 1 second, substitute value 7 V. Demonstration of various events:

1. Interruption of the EtherCAT connection

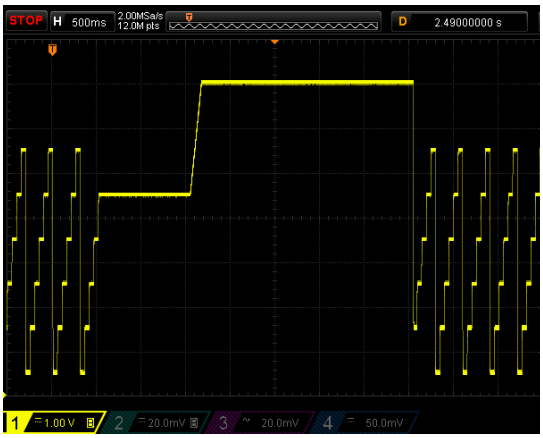
   A: Interruption, last value is output,
   Watchdog starts to run down

   B: immediate output of the substitute value after 1 second

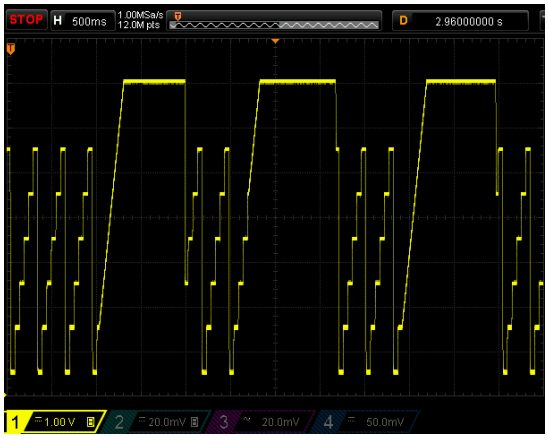
   C: EtherCAT connection restored, device in OP,
   new output data arrives



2. likewise, but with ramp 0.5 sec to the substitute value



3. Intentional EtherCAT status change
   OP -> SafeOP -> OP -> Pre-Op -> OP ->Init -> OP



Watchdog Counter

Each watchdog event is counted in index 0x90n0:05 “Watchdog Counter” (secured against power failure).

NOTICE

Note on resetting the watchdog counters

As the watchdog is a device property, it is displayed for each output channel but has the same value for all channels. When a watchdog counter is reset, all other channel-specific watchdog counters are also reset.

The counter is reset

- by the command x403n to index 0xFB00:01 (channel 1: n = 0, channel 2: n = 1, ...), the success is displayed with "255" in index 0xFB00:03 "Response".
- and by command x4001 "Reset all AO Counter"

FB00:0	DEV Command	RO	> 3 <
FB00:01	Request	RW	00 00
FB00:02	Status	RO	0x00 (0)
FB00:03	Response	RO	00 00 00 00 00 00

Fig. 178: CoE Index 0xFB00, „DEV Command“

During command execution, "Status" 255 "busy" is displayed in index 0xFB00:02, "0" means "successfully completed"
The firmware responds to an unknown command with

'Term 5 (EL4374)' (1002): CoE ('InitDown' 0xfb00:01) - SDO Abort ('General parameter incompatibility reason.', 0x06040043).

Fig. 179: General parameter incompatibility reason, 0x06040043

5.2.2.3.9 Limiter

The limiter makes it possible to limit the electrical output value to protect the connected signal sink.

Parameter:

Index	Designation
80nD:1C	Limiter Low Value
80nD:1D	Limiter High Value

Result:

A setpoint higher/lower than the limit values leads

- to increase +1 of the corresponding counter Index 90n0:06 "Limiter Low Counter" or Index 90n0:07 "Limiter High Counter" (stored secured against power failure)
- to display Error + Overrange/Underrange in the PDO "AO Status and Error-LED"
- to a limitation of the electrical output in the amount of the set amplitude

By default, the limiter is set to the maximum limits of the technical AEW and therefore has no effect.

The counter is reset

- by the command x402n to index FB00:01 (channel1: n=0, channel 2: n=1, ...), the success is displayed with "255" in index FB00:03 Response.
- or by the command x4001 "Reset all AO counters"
- or by changing the interface

FB00:0	DEV Command	RO	> 3 <
FB00:01	Request	RW	00 00
FB00:02	Status	RO	0x00 (0)
FB00:03	Response	RO	00 00 00 00 00 00

Fig. 180: CoE Index FB00, DEV Command

During command execution, "Status" 255 "busy" is displayed in index FB00:02, "0" means "successfully completed".

The firmware responds to an unknown command with

'Term 5 (EL4374)' (1002): CoE ('InitDown' 0xfb00:01) - SDO Abort ('General parameter incompatibility reason.', 0x06040043).

Fig. 181: General parameter incompatibility reason, 0x06040043

The limiter limits can be password protected, see section "Password protection for user data".

Example: Sawtooth 1..7 V is output and then a "Limiter High Value" of 5 V is set:

Index	Name	Flags	Value
802D:1C	Limiter Low Value	RW	-10.737420 (-1.073742e+01)
802D:1D	Limiter High Value	RW	5.000000 (5.000000e+00)

Fig. 182: Index 80nD set to 5 V

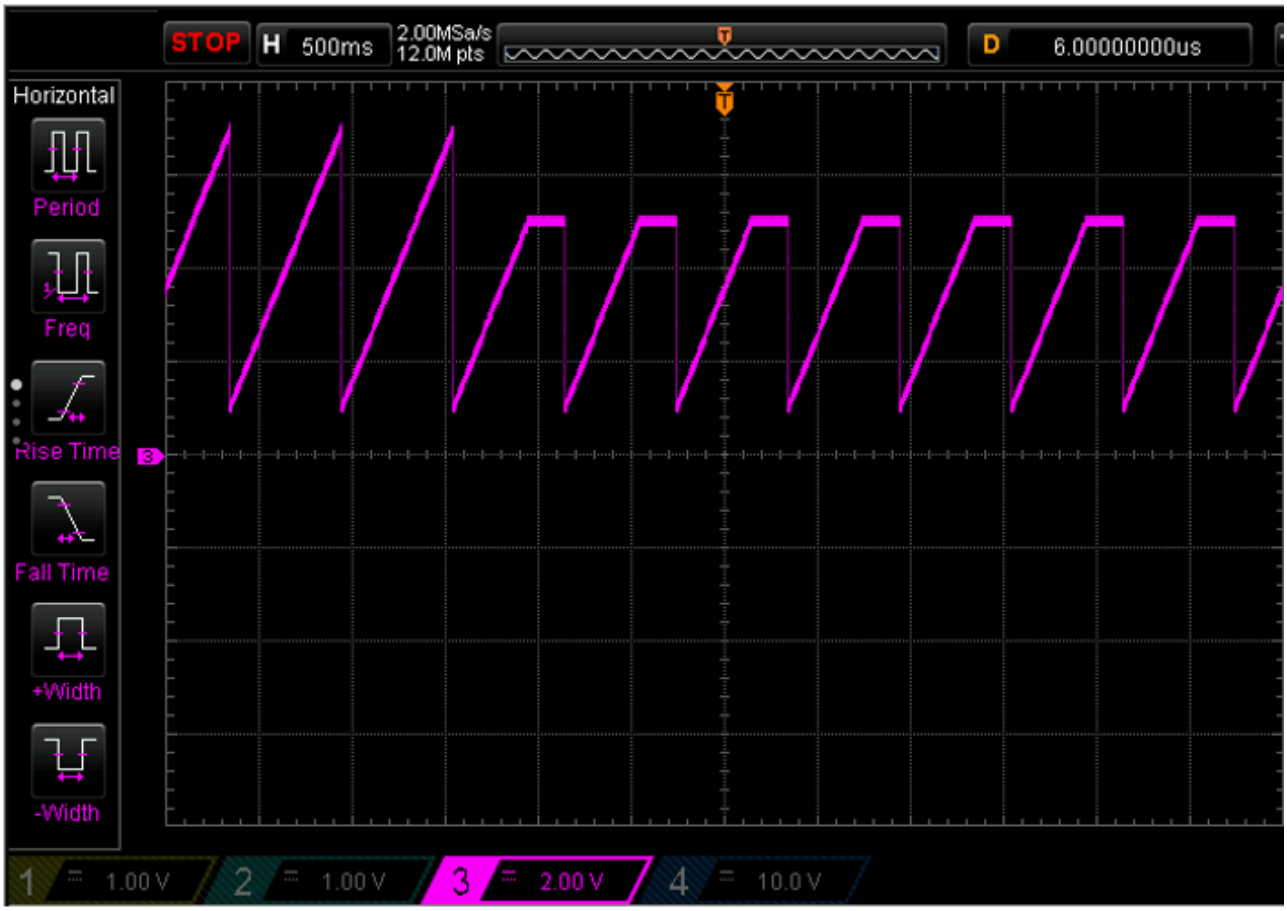


Fig. 183: Sawtooth output

The overrange is counted accordingly below:

9020:07	Limiter High Counter	RO	0x000000D8 (216)

Fig. 184: Counter overrange

Note:

⚠ CAUTION

Consider additional protective functions!

The limiter function protects the connected load from unintentionally high or low outputs, for example due to programming errors. However, it is possible that malfunctions may occur.

Depending on the expected level of damage, additional protective functions should therefore be provided. The function must not be used for functional safety purposes!

5.2.2.3.10 Slew Rate

The SlewRate function limits the slope with which the next setpoint is output electrically by the terminal. This serves, for example, to protect the connected signal sink or ultimately a mechanical load.

Parameter:

Index	Designation
80nD:1E	Output Slew Rate [s]

Result:

If the setpoint specification causes a higher electrical signal slope than permitted, the channel automatically reduces the setpoint change per internal cycle to the specified level. The value applies to positive and negative changes and is unsigned. The slope of the ramp must be specified in [sec] in relation to AEW_{nom}. If, for example, a maximum permitted slope of ±2 V/s at AEW_{nom} = 10 V is required, this results in an Output Slew Rate of 5 s. By default, the Slew Rate is unlimited (0).

The intermediate value after this functional unit can be viewed in x90n0:08.

Example: A rectangular output of 1 V / 6 V from the PLC is converted to a triangular output by "Output Slew Rate" = 1 s (corresponds to 10 V/s):

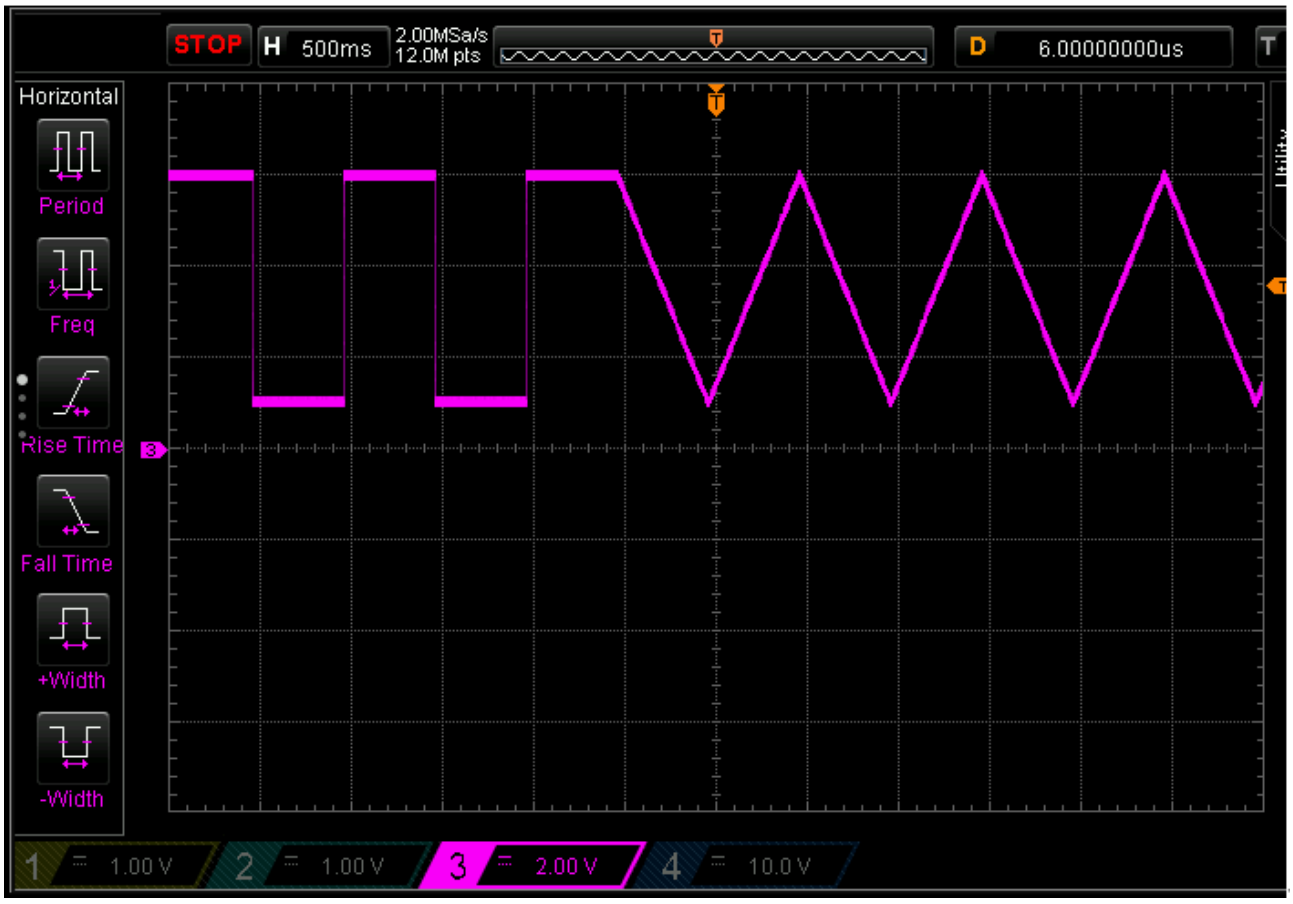


Fig. 185: Triangle output by reshaping

5.2.2.3.11 Setpoint processing, user/vendor calibration

The electrical set output value generated and digitized by the DAC must or can be changed in the device in order to

- compensate for hardware dependency (keyword: calibration)
- or to make application-specific changes

The output value can be changed in 3 functional units, all 3 can be active at the same time:

- User Scale [▶ 155]
- User calibration
- Vendor calibration

The "User/vendor calibration" functional unit is intended for correcting hardware-specific influences. They are implemented as 3rd order polynomial transformations (offset, gain, x^2, x^3).

Note: The term "calibration", which is historically based at Beckhoff, is used here in the CoE, even if it has nothing to do with deviation statements of a calibration certificate. Actually, this is a description of the vendor or customer calibration data/adjustment data used by the device during operation in order to maintain the assured analog accuracy.

- User Calibration

The "User calibration" functional unit can be used by the user if alternative, system-dependent correction values are to be used permanently.

To be able to work with both INT16-based gain/offset values and Real32 coefficients, processing in "User Calibration" (if Enable User calibration = 1) proceeds as follows:

- for setpoint >=0: "Value after User calibration" = S0 + "Value after Vendor calibration" * S1 +("Value after Vendor calibration")² * S2
- for setpoint <0: "Value after User calibration" = S0 + "Value after Vendor calibration" * S1n +("Value after Vendor calibration")² * S2

Parameter:

Index	Name	Data type	Meaning
80n0:07	Enable User calibration	BOOL	disabled by default, calculation only takes place if TRUE
80n0:17	User calibration offset	SINT16	1 bit = AEW_{nom} / 32767, default: 0
80n0:18	User calibration gain	UINT16	1 bit corresponds to 2-16, "1" therefore corresponds to $x7FFF/32767_{dec}$, default: 1
80nC:01	User calibration data	BYTE4	4 bytes of free memory space; here it is possible to store the calibration date in the form of 8 CHAR, for example
80nC:03..0D	User Scale Gain (Real32)	REAL32	Real32 coefficients S0/S1/S2/S3/S1n of the calculation polynomial, default: S0=0, S1=1, S1n = 1, S2=0

The intermediate value after this functional unit can be viewed in index 90n0:0A.

The number of setting changes in this functional unit is counted up in index 90n2:12 "User Calibration Counter" (cannot be deleted).

Procedure: the counter is incremented the first time any parameter in the data area index 80nC or index 80n0:17/18 is changed; further changes in the data area in the following 30 seconds are not taken into account for the counter. After this time has elapsed, a parameter change will increment the counter again.

- Vendor Calibration

The electrical channel is calibrated by Beckhoff in the Vendor Calibration functional unit for compliance with the given uncertainty specification (see Technical Data, previously: output error). The vendor calibration data from Beckhoff is available in this area.

Parameter:

Index	Name	Data type	Meaning
80n0:08	Enable vendor calibration	BOOL	enabled by default, the data is taken into account. Can be disabled on the application side if only the User Calibration Data is to be used for the calculation
80nF	Vendor calibration data	-	not intended for user modification

The intermediate value after this functional unit can be viewed in index 90n0:0B under 'Value after Vendor Calibration'.

The number of setting changes in this functional unit is counted up in index 90n2:12 as 'Vendor Calibration Counter' and cannot be deleted.

Procedure: the counter is incremented the first time any parameter in the data area is changed; further changes in the data area in the following 30 seconds are not taken into account for the counter. After this time has elapsed, a parameter change will increment the counter again.

Password protection for user data

Some user data are protected against unwanted or inadvertent writing by an additional password to be entered in CoE 0xF009:

- CoE write accesses by the user, PLC or startup entries in *Single* or *CompleteAccess* mode
- Overwrite the values by *RestoreDefaultParameter* Access to 0x80n0 (or 0x80nD, if available)

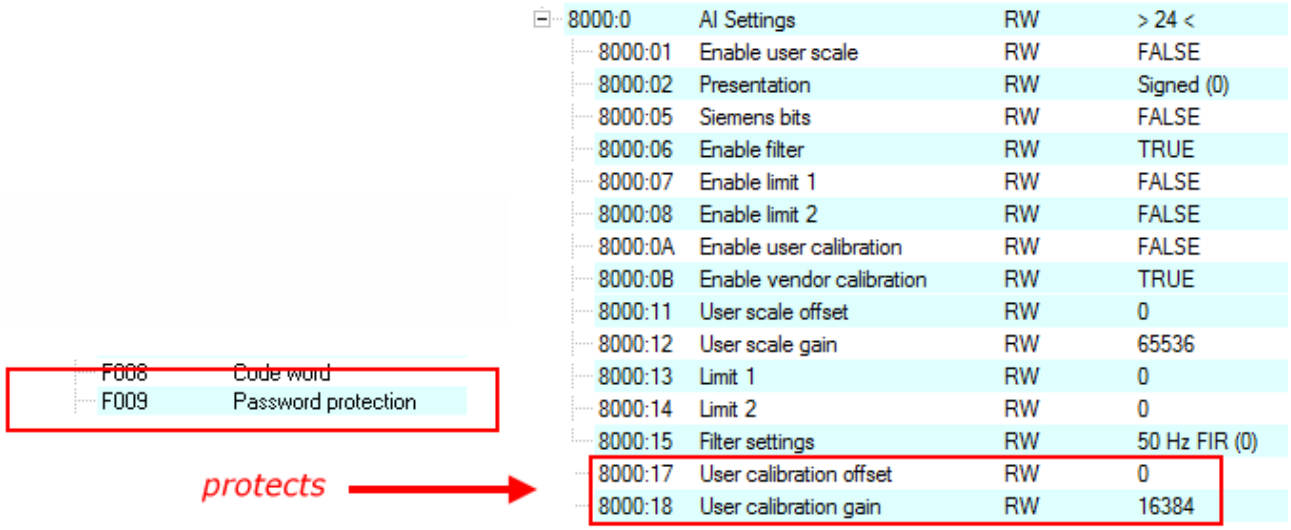


Fig. 186: Password protection for the 0x8000:17 and 0x8000:18 entries (example)

Use of CoE 0xF009

- Entering 0x12345678 enables the password protection → Object shows "1" (enabled)
  Protected objects can now no longer be changed, no error message occurs during a write access!
- Entering 0x11223344 disables password protection → Object displays "0" (disabled)

Password protection takes effect with the following AI settings:

Index	Designation
80n0:07	Enable User calibration
80n0:08	Enable Vendor calibration
80n0:17	User calibration offset
80n0:18	User calibration gain
80nC	User calibration data
80nD:1A	Limiter Low Value
80nD:1B	Limiter High Value

5.2.2.3.12 DAC output stage

The final processed setpoint is now transferred to the local DAC function block as an integer value and the electrical output is generated.

A case-by-case DAC error is output as PDO "AO Status" -> Error.

Notes:

- During "Power On" (U_s, U_p or both) or "Power Off", brief unregulated output (typically up to ±2 V for 200 ms) may occur at the output. If necessary, use an external signal isolator (relay) or switch to self-isolating analog output products.
- The output voltage in I mode can rise to >20 V, e.g. in the event of a wire break
- The output diagnosis wire break/short circuit should be seen as an informative aid; it cannot cover all error states. It indicates an overload of the analog output.
- In the time between "Power On" (power supply is available, local µC has started up) and EtherCATState = OP, the AO channel outputs the analog output value that is specified in the watchdog as a substitute value.

NOTICE

Note on overload behavior

In the event of an overcurrent, the channel interrupts the current flow periodically (so-called "auto-retry") and normal measuring operation is resumed as soon as the current falls below the limit threshold. The overcurrent event occurs at a signal current > approx. 40 mA for several seconds or also briefly at a significantly higher current, so both current and thermal monitoring takes place.

5.3 Overview of parameter objects (CoE)

EtherCAT XML Device Description

The display matches that of the CoE objects from the EtherCAT ESI Device Description (XML).We recommend downloading the latest XML file from the download area of the Beckhoff website and installing it according to installation instructions.

Parameterization via the CoE list (CAN over EtherCAT)

The EtherCAT device is parameterized via the CoE-Online tab [▶ 112] (double-click on the respective object) or via the Process Data tab [▶ 109](allocation of PDOs). Please note the following general CoE notes [▶ 35] when using/manipulating the CoE parameters:

- Keep a startup list if components have to be replaced
- Differentiation between online/offline dictionary, existence of current XML description
- use “CoE reload [▶ 204]” for resetting changes

5.3.1 Restore objects

Index 1011 Restore default parameters

Index (hex)	Name	Meaning	Data type	Flags	Default
1011:0	Restore default parameters	Restore default parameters	UINT8	RO	0x01 (1dec)
1011:01	SubIndex 001	If this object is set to "0x64616F6C" in the set value dialog, all backup objects are reset to their delivery state.	UINT32	RW	0x00000000 (0dec)

5.3.2 Configuration data

Index 80n0 AO settings*)

Index (hex)	Name	Meaning	Data type	Flags	Default
80n0:0	AO Settings	Max. Subindex	UINT8	RO	0x16 (22_{dec})
80n0:01	Enable User Scale [▶ 155]	User scaling is enabled. (see data stream flow chart [▶ 142])	BOOLEAN	RW	0x00 (0_{dec})
80n0:02	Presentation [▶ 153]	0: *Signed presentation* The measured value is displayed in two's complement. Maximum presentation range at 16 bits: -32768_{dec} ... $+32767_{dec}$ 1: *Unsigned presentation* Maximum presentation range for 16 bits: 0 ... $+65535_{dec}$ 2: *Absolute value with MSB as sign* The measured value is output in the signed amount representation. Maximum presentation range at 16 bits: $- 32768_{dec}$ .. $+32767_{dec}$ 3: *Absolute value* The negative number range is also output as positive.	BIT3	RW	0x00 (0_{dec})
80n0:05	Watchdog Type [▶ 157]	0: *Default watchdog value* The default value (80n0:13) is active. 1: *Watchdog ramp* The ramp (80n0:14) for moving to the default value is active. 2: *Last output value* The last process data is output when the watchdog drops.	BIT2	RW	0x00 (0_{dec})
80n0:07	Enable user calibration [▶ 164]	Enable user calibration (see data stream flow chart [▶ 142])	BOOLEAN	RW	0x00 (0_{dec})
80n0:08	Enable Vendor Calibration [▶ 164]	Enable vendor calibration (see data stream flow chart [▶ 142])	BOOLEAN	RW	0x01 (1_{dec})
80n0:11	User Scale Offset [▶ 155]	User scale offset	INT16	RW	0x0000 (0_{dec})
80n0:12	User Scale Gain [▶ 155]	User scale gain. The gain has a fixed-point representation with a factor of 2^{-16}. The value 1 corresponds to 65535 (0x00010000).	INT32	RW	0x00010000 (65536_{dec})
80n0:13	Watchdog Output Value [▶ 157]	Watchdog default output value	INT16	RW	0x0000 (0_{dec})
80n0:14	Watchdog Ramp [▶ 157]	Ramps to the default value Value in digits/ms.	UINT16	RW	0xFFFF (65535_{dec})
80n0:15	User Calibration Offset [▶ 164]	User calibration offset	INT16	RW	0x0000 (0_{dec})
80n0:16	User Calibration Gain [▶ 164]	User calibration gain	UINT16	RW	0x7FFF (32767_{dec})

*) EL4072 (Ch.1 - 2): $0 \leq n \leq 1$; EL4074 (Ch.1 – 4): $0 \leq n \leq 3$; EL4078 (Ch.1 – 8): $0 \leq n \leq 7$

Index 80nC AO User Calibration Data*)

Index (hex)	Name	Meaning	Data type	Flags	Default
80nC:0	AO User Calibration Data	Max. Subindex	UINT8	RO	0x06 (6_{dec})
80nC:01	Calibration Data	4 bytes of free memory space; here it is possible to store the calibration date in the form of 8 CHAR, for example	OCTET-STRING[4]	RW	{0}
80nC:03	S0	Real32 coefficient S0	REAL32	RW	0x00000000 (0_{dec})
80nC:04	S1	Real32 coefficient S1	REAL32	RW	0x00000000 (0_{dec})
80nC:05	S2	Real32 coefficient S2	REAL32	RW	0x00000000 (0_{dec})
80nC:06	S3	Real32 coefficient S3	REAL32	RW	0x00000000 (0_{dec})

*) EL4072 (Ch.1 - 2): *0 ≤ n ≤ 1*; EL4074 (Ch.1 – 4): *0 ≤ n ≤ 3*; EL4078 (Ch.1 – 8): *0 ≤ n ≤ 7*

Index 80nD AO Advanced Settings*)

Index (hex)	Name	Meaning	Data type	Flags	Default
80nD:0	AI Advanced Settings	Max. Subindex	UINT8	RO	0x1E (30_{dec})
80nD:11	Output Interface [▶ 154]	Values: 0 – None 2 - V ±10V 14 - V 0-10V 17 - I ±20mA 18 - I 0-20mA 19 - I 4-20mA	UINT16	RW	0x0002 (2_{dec})
80nD:12	Integer Scaler [▶ 150]	Values: 0 - Extended Range 3 - Legacy Range	UINT16	RW	0x0000 (0_{dec})
80nD:13	User Scale Offset (Real32) [▶ 155]	User scale offset	REAL32	RW	0x00000000 (0.0)
80nD:14	User Scale Gain (Real32) [▶ 155]	User scale gain.	REAL32	RW	0x3F800000 (1.0)
80nD:15	Warning Level Low [▶ 156]	Warning if output value falls below permitted value	REAL32	RW	0xC1200000 (-10.0)
80nD:16	Warning Level High [▶ 156]	Warning if permissible output value is exceeded	REAL32	RW	0x41200000 (10.0)
80nD:1A	Watchdog Output Value (Real32) [▶ 157]	User-specific setpoint Watchdog (Real32)	REAL32	RW	0x00000000 (0.0)
80nD:1B	Watchdog Ramp (Real32) [▶ 157]	Time of the ramp from maximum value (10 V / 20 mA) to zero. The end value is defined via watchdog value 0x18. [s]	REAL32	RW	0x00000000 (0.0)
80nD:1C	Limiter Low Value [▶ 161]	Output value limit (lowest value)	REAL32	RW	0x412BCC79 (10.7374201)
80nD:1D	Limiter High Value [▶ 161]	Output value limit (highest value)	REAL32	RW	0x412BCC79 (10.7374201)
80nD:1E	Output Slew Rate [▶ 163]	Limitation of the setpoint gradient [s]	REAL32	RW	0x00000000 (0.0)

*) EL4072 (Ch.1 - 2): *0 ≤ n ≤ 1*; EL4074 (Ch.1 – 4): *0 ≤ n ≤ 3*; EL4078 (Ch.1 – 8): *0 ≤ n ≤ 7*

Index 80nF AO Vendor Calibration Data*)

Index (hex)	Name	Meaning	Data type	Flags	Default
80nF:0	AO Vendor Calibration Data	Max. Subindex	UINT8	RO	0x08 (08_{dec})
80nF:01	Calibration Date [▶ 164]	4 bytes of free memory space; here it is possible to store the calibration date in the form of 8 CHAR, for example	OCTET-STRING[4]	RW	{0}
80nF:03	S0	Real32 coefficient S0	REAL32	RW	0x00000000 (0_{dec})
80nF:04	S1	Real32 coefficient S1	REAL32	RW	0x00000000 (0_{dec})
80nF:05	S2	Real32 coefficient S2	REAL32	RW	0x00000000 (0_{dec})
80nF:06	S3	Real32 coefficient S3	REAL32	RW	0x00000000 (0_{dec})
800F:07	T1	Real32 coefficient T1	REAL32	RW	0x00000000 (0_{dec})
800F:08	T1S1	Real32 coefficient T1S1	REAL32	RW	0x00000000 (0_{dec})

*) EL4072 (Ch.1 - 2): $0 \leq n \leq 1$; EL4074 (Ch.1 – 4): $0 \leq n \leq 3$; EL4078 (Ch.1 – 8): $0 \leq n \leq 7$

5.3.3 Input data

Index 60n0 AO Inputs*)

Index (hex)	Name	Meaning	Data type	Flags	Default
60n0:0	AO Inputs	Maximum subindex	UINT8	RO	0x07 (7_{dec})
60n0:02	Output Overload [▶ 148]	Overload on the analog output.	BOOLEAN	RO	0x00 (0_{dec})
60n0:03	Underrange [▶ 148]	The analog output is smaller than the technical output range or than the set limit value.	BOOLEAN	RO	0x00 (0_{dec})
60n0:04	Overrange [▶ 148]	The analog output is greater than the technical output range or than the set limit value.	BOOLEAN	RO	0x00 (0_{dec})
60n0:06	Warning [▶ 148]	Is set if "AO Output value" is outside the technical output range or the supply voltage is too low or the terminal temperature is too high.	BOOLEAN	RO	0x00 (0_{dec})
60n0:07	Error [▶ 148]	Is set if the limit values of the output range are exceeded or there is a hardware error.	BOOLEAN	RO	0x00 (0_{dec})

*) EL4072 (Ch.1 - 2): $0 \leq n \leq 1$; EL4074 (Ch.1 – 4): $0 \leq n \leq 3$; EL4078 (Ch.1 – 8): $0 \leq n \leq 7$

5.3.4 Output data

Index 70n0 AO Outputs*)

Index (hex)	Name	Meaning	Data type	Flags	Default
70n0:0	AI Outputs	Maximum subindex	UINT8	RO	0x14 (20_{dec})
70n0:11	Value [▶ 146]	Analog output value	INT16	RO	0x0000 (0_{dec})
70n0:13	Value (Real32) [▶ 146]	Analog output value (Real32)	REAL32	RO	0x00000000 (0_{dec})
70n0:14	Output Cycle Counter [▶ 149]	The counter is incremented each time the output data in the process image is updated.	UINT16	RO	0x0000 (0_{dec})

*) EL4072 (Ch.1 - 2): $0 \leq n \leq 1$; EL4074 (Ch.1 – 4): $0 \leq n \leq 3$; EL4078 (Ch.1 – 8): $0 \leq n \leq 7$

5.3.5 Information and diagnostic data

Index 90n0 AO Internal Data*)

Index (hex)	Name	Meaning	Data type	Flags	Default
90n0:0	AO Internal Data	Max. Subindex	UINT8	RO	0x0F (15_{dec})
90n0:01	Value after User Scale	Current output value after UserScale, see data flow, chapter "Analog output commissioning [▶ 142]"	REAL32	RO	0x00000000 (0_{dec})
90n0:02	Warning Low Counter	Counter for "Warning Low" events	UINT32	RO	0x00000000 (0_{dec})
90n0:03	Warning High Counter	Counter for "Warning High" events	UINT32	RO	0x00000000 (0_{dec})
90n0:05	Watchdog Counter	Counter for "Watchdog" events	UINT32	RO	0x00000000 (0_{dec})
90n0:06	Limiter Low Counter	Counter for "Limiter Low" events	UINT32	RO	0x00000000 (0_{dec})
90n0:07	Limiter High Counter	Counter for "Limiter High" events	UINT32	RO	0x00000000 (0_{dec})
90n0:08	Value After Slew Rate Limitation	Current output value after SlewRateLimitation, see data flow, chapter "Analog output commissioning [▶ 142]"	REAL32	RO	0x00000000 (0_{dec})
90n0:0A	Value After User Calibration	Current output value after user adjustment, see data flow, chapter "Commissioning analog output [▶ 142]"	REAL32	RO	0x00000000 (0_{dec})
90n0:0B	Value After Vendor Calibration	Current output value after vendor adjustment, see data flow, chapter "Analog output commissioning [▶ 142]"	REAL32	RO	0x00000000 (0_{dec})
90n0:0C	DAC Raw Value	Output value to the DAC	INT32	RO	0x00000000 (0_{dec})
90n0:0F	Output Cycle Counter Error	Counter for "Output Cycle Counter Error" events	UINT32	RO	0x00000000 (0_{dec})

*) EL4072 (Ch.1 - 2): $0 \leq n \leq 1$; EL4074 (Ch.1 – 4): $0 \leq n \leq 3$; EL4078 (Ch.1 – 8): $0 \leq n \leq 7$

Index 90n2 AO Info Data*)

Index (hex)	Name	Meaning	Data type	Flags	Default
90n2:0	AO Info Data	Max. Subindex	UINT8	RO	0x12 (18_{dec})
90n2:11	Vendor Calibration Counter [▶ 165]	Counter for changes to the vendor adjustment data	UINT32	RO	0x00000000 (0_{dec})
90n2:12	User Calibration Counter [▶ 164]	Counter for changes to the user calibration data	UINT32	RO	0x00000000 (0_{dec})

*) EL4072 (Ch.1 - 2): $0 \leq n \leq 1$; EL4074 (Ch.1 – 4): $0 \leq n \leq 3$; EL4078 (Ch.1 – 8): $0 \leq n \leq 7$

Index F000 Modular device profile

Index (hex)	Name	Meaning	Data type	Flags	Default
F000:0	Modular device profile	General information for the Modular Device Profiles (MDP) Organizational information on the profiles used in the device and listed in 0xF010	UINT8	RO	0x02 (2_{dec})
F000:01	Module index distance	Index distance of the objects of the individual channels	UINT16	RO	0x0010 (16_{dec})
F000:02	Maximum number of modules	Number of channels	UINT16	RO	0x0008 (8_{dec})

Index F010 Module Profile List

Index (hex)	Name	Meaning	Data type	Flags	Default
F010:0	Module Profile List	Maximum subindex	UINT8	RO	0x04 (4_{dec})
F010:01	Subindex 001	Profile 300	INT32	RO	0x0000012C (300_{dec})
F010:02	Subindex 002	Profile 300	INT32	RO	0x0000012C (300_{dec})
F010:03	Subindex 003	Profile 400	INT32	RO	0x00000190 (400_{dec})
F010:04	Subindex 004	Profile 400	INT32	RO	0x00000190 (400_{dec})

Index F900 DEV Info Data

Index (hex)	Name	Meaning	Data type	Flags	Default
F900:0	DEV Info Data	Largest subindex of this object	UINT8	RO	0x16 (22_{dec})
F900:01	Power Good	see chapter "Device diagnostic functions"	BOOLEAN	RO	0x00 (0_{dec})
F900:11	Operating Time		UINT32	RO	0x00000000 (0_{dec})
F900:12	Device Temperature		REAL32	RO	0x00000000 (0_{dec})
F900:13	Min. Device Temperature		REAL32	RO	0x00000000 (0_{dec})
F900:14	Max. Device Temperature		REAL32	RO	0x00000000 (0_{dec})

Index F915 LED Status

Index (hex)	Name	Meaning	Data type	Flags	Default
F915:0	LED Status	Largest subindex of this object	UINT8	RO	0x10 (16_{dec})
F915:01	RUN	see chapter "Device diagnostic functions"	UINT32	RO	0x00000000 (0_{dec})
F915:09	Error Ch.1		UINT32	RO	0x00000000 (0_{dec})
F915:0A	Error Ch.2		UINT32	RO	0x00000000 (0_{dec})
F915:0B	Error Ch.3		UINT32	RO	0x00000000 (0_{dec})
F915:0C	Error Ch.4		UINT32	RO	0x00000000 (0_{dec})
F915:0D	Error Ch.5		UINT32	RO	0x00000000 (0_{dec})
F915:0E	Error Ch.6		UINT32	RO	0x00000000 (0_{dec})
F915:0F	Error Ch.7		UINT32	RO	0x00000000 (0_{dec})
F915:10	Error Ch.8		UINT32	RO	0x00000000 (0_{dec})

Index FB00 DEV Command

The command object is used for triggering an action in the terminal. The command is started by writing subindex 1 (request). Write access is disabled until the current command is completed.

Index (hex)	Name	Meaning	Data type	Flags	Default
FB00:0	DEV Command	Largest subindex of this object	UINT8	RO	0x03 (3_{dec})
FB00:01	Request	Command value, for use see corresponding application chapter	OCTET-STRING [2]	RW	0x0000 (0_{dec})
FB00:02	Status	Command status, for use see corresponding application chapter	UINT8	RW	0x00 (0_{dec})
FB00:03	Response	Command response, for use see corresponding application chapter	OCTET-STRING [2]	RW	0x00000000 (0_{dec})

5.3.6 Standard objects

Index 1000 Device type

Index (hex)	Name	Meaning	Data type	Flags	Default
1000:0	Device type	Device type of the EtherCAT slave: the Lo-Word contains the used CoE profile (5001). The Hi-Word contains the module profile according to the modular device profile.	UINT32	RO	0x01901389 (26219401_{dec})

Index 1008 Device name

Index (hex)	Name	Meaning	Data type	Flags	Default
1008:0	Device name	Device name of the EtherCAT slave	STRING	RO	EL407x

Index 1009 Hardware version

Index (hex)	Name	Meaning	Data type	Flags	Default
1009:0	Hardware version	Hardware version of the EtherCAT slave	STRING	RO	-

Index 100A Software version

Index (hex)	Name	Meaning	Data type	Flags	Default
100A:0	Software version	Firmware version of the EtherCAT slave	STRING	RO	00

Index 100B Bootloader version

Index (hex)	Name	Meaning	Data type	Flags	Default
100B:0	Bootloader version	Bootloader version	STRING	RO	n/a

Index 1018 Identity

Index (hex)	Name	Meaning	Data type	Flags	Default
1018:0	Identity	Information for identifying the slave	UINT8	RO	0x04 (4_{dec})
1018:01	Vendor ID	Vendor ID of the EtherCAT slave	UINT32	RO	0x00000002 (2_{dec})
1018:02	Product code	Product code of the EtherCAT slave	UINT32	RO	0x0FEE3052 (267268178_{dec})
1018:03	Revision	Revision number of the EtherCAT slave; the Low Word (bit 0-15) indicates the special terminal number, the High Word (bit 16-31) refers to the device description	UINT32	RO	0x00000000 (0_{dec})
1018:04	Serial number	Serial number of the EtherCAT slave; the Low Byte (bit 0-7) of the Low Word contains the year of production, the High Byte (bit 8-15) of the Low Word contains the week of production, the High Word (bit 16-31) is 0	UINT32	RO	0x00000000 (0_{dec})

Index 10E2 Manufacturer-specific Identification Code*)

Index (hex)	Name	Meaning	Data type	Flags	Default
10E2:0	Manufacturer-specific Identification Code	Manufacturer specific Identification Code	UINT8	RO	0x01 (1_{dec})
10E2:01	SubIndex 001	reserved	STRING	RO	

Index 10F0 Backup parameter handling

Index (hex)	Name	Meaning	Data type	Flags	Default
10F0:0	Backup parameter handling	Information for standardized loading and saving of backup entries	UINT8	RO	0x01 (1_{dec})
10F0:01	Checksum	Checksum across all backup entries of the EtherCAT slave	UINT32	RO	0x00000000 (0_{dec})

Index 10F3 Diagnosis History

Index (hex)	Name	Meaning	Data type	Flags	Default
10F3:0	Diagnosis History	Maximum subindex	UINT8	RO	0x15 (21_{dec})
10F3:01	Maximum Messages	Maximum number of stored messages A maximum of 16 messages can be stored	UINT8	RO	0x00 (0_{dec})
10F3:02	Newest Message	Subindex of the latest message	UINT8	RO	0x00 (0_{dec})
10F3:03	Newest Acknowledged Message	Subindex of the last confirmed message	UINT8	RW	0x00 (0_{dec})
10F3:04	New Messages Available	Indicates that a new message is available	BOOLEAN	RO	0x00 (0_{dec})
10F3:05	Flags	not used	UINT16	RW	0x0000 (0_{dec})
10F3:06	Diagnosis Message 001	Message 1	OCTET-STRING[24]	RO	{0}
…	…	….	…	…	…
10F3:15	Diagnosis Message 016	Message 16	OCTET-STRING[24]	RO	{0}

Index 10F8 Timestamp Object

Index (hex)	Name	Meaning	Data type	Flags	Default
10F8:0	Timestamp Object	Timestamp Object [ns] For SM synchronous operation: Time since power-on For DC synchronous operation: copy of the DC time Time can be used by the device e.g. for timestamps of the DiagMessage	UINT64	RO	

Index 1420 AO RxPDO-Par Standard (INT16) Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1420:0	AO RxPDO-Par Standard (INT16) Ch.1	PDO Parameter RxPDO 33	UINT8	RO	0x06 (6_{dec})
1420:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 33	OCTET-STRING[2]	RO	22 16

Index 1422 AO RxPDO-Par Standard (Real32) Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1422:0	AO RxPDO-Par Standard (Real32) Ch. 1	PDO Parameter RxPDO 35	UINT8	RO	0x06 (6_{dec})
1422:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 35	OCTET-STRING[2]	RO	20 16

Index 1444 AO RxPDO-Par Standard (INT16) Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1444:0	AO RxPDO-Par Standard (INT16) Ch.2	PDO Parameter RxPDO 69	UINT8	RO	0x06 (6_{dec})
1444:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 69	OCTET-STRING[2]	RO	46 16

Index 1446 AO RxPDO-Par Standard (Real32) Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1446:0	AO RxPDO-Par Standard (Real32) Ch.2	PDO Parameter RxPDO 71	UINT8	RO	0x06 (6_{dec})
1446:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 71	OCTET-STRING[2]	RO	44 16

Index 1468 AO RxPDO-Par Standard (INT16) Ch.3*)

Index (hex)	Name	Meaning	Data type	Flags	Default
1468:0	AO RxPDO-Par Standard (INT16) Ch.3	PDO Parameter RxPDO 105	UINT8	RO	0x06 (6_{dec})
1468:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 105	OCTET-STRING[2]	RO	6A 16

*) EL4074, EL4078 only

Index 146A AO RxPDO-Par Standard (Real32) Ch.3*)

Index (hex)	Name	Meaning	Data type	Flags	Default
146A:0	AO RxPDO-Par Standard (Real32) Ch.3	PDO Parameter RxPDO 107	UINT8	RO	0x06 (6_{dec})
146A:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 107	OCTET-STRING[2]	RO	68 16

*) EL4074, EL4078 only

Index 148C AO RxPDO-Par Standard (INT16) Ch.4*)

Index (hex)	Name	Meaning	Data type	Flags	Default
148C:0	AO RxPDO-Par Standard (INT16) Ch.4	PDO Parameter RxPDO 141	UINT8	RO	0x06 (6_{dec})
148C:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 141	OCTET-STRING[2]	RO	8E 16

*) EL4074, EL4078 only

Index 148E AO RxPDO-Par Standard (Real32) Ch.4*)

Index (hex)	Name	Meaning	Data type	Flags	Default
148E:0	AO RxPDO-Par Standard (Real32) Ch.4	PDO Parameter RxPDO 143	UINT8	RO	0x06 (6_{dec})
148E:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 143	OCTET-STRING[2]	RO	8C 16

*) EL4074, EL4078 only

Index 14B0 AO RxPDO-Par Standard (INT16) Ch.5*)

Index (hex)	Name	Meaning	Data type	Flags	Default
14B0:0	AO RxPDO-Par Standard (INT16) Ch.5	PDO Parameter RxPDO 177	UINT8	RO	0x06 (6_{dec})
14B0:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 177	OCTET-STRING[2]	RO	B2 16

*) EL4078 only

Index 14B2 AO RxPDO-Par Standard (Real32) Ch.5*)

Index (hex)	Name	Meaning	Data type	Flags	Default
14B2:0	AO RxPDO-Par Standard (Real32) Ch.5	PDO Parameter RxPDO 179	UINT8	RO	0x06 (6_{dec})
14B2:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 179	OCTET-STRING[2]	RO	B0 16

*) EL4078 only

Index 14D4 AO RxPDO-Par Standard (INT16) Ch.6*)

Index (hex)	Name	Meaning	Data type	Flags	Default
14D4:0	AO RxPDO-Par Standard (INT16) Ch.6	PDO Parameter RxPDO 213	UINT8	RO	0x06 (6_{dec})
14D4:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 213	OCTET-STRING[2]	RO	D6 16

*) EL4078 only

Index 14D6 AO RxPDO-Par Standard (Real32) Ch.6*)

Index (hex)	Name	Meaning	Data type	Flags	Default
14D6:0	AO RxPDO-Par Standard (Real32) Ch.6	PDO Parameter RxPDO 215	UINT8	RO	0x06 (6_{dec})
14D6:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 215	OCTET-STRING[2]	RO	D4 16

*) EL4078 only

Index 14F8 AO RxPDO-Par Standard (INT16) Ch.7*)

Index (hex)	Name	Meaning	Data type	Flags	Default
14F8:0	AO RxPDO-Par Standard (INT16) Ch.7	PDO Parameter RxPDO 249	UINT8	RO	0x06 (6_{dec})
14F8:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 249	OCTET-STRING[2]	RO	FA 16

*) EL4078 only

Index 14FA AO RxPDO-Par Standard (Real32) Ch.7*)

Index (hex)	Name	Meaning	Data type	Flags	Default
14FA:0	AO RxPDO-Par Standard (Real32) Ch.7	PDO Parameter RxPDO 251	UINT8	RO	0x06 (6_{dec})
14FA:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 251	OCTET-STRING[2]	RO	F8 16

*) EL4078 only

Index 151C AO RxPDO-Par Standard (INT16) Ch.8*)

Index (hex)	Name	Meaning	Data type	Flags	Default
151C:0	AO RxPDO-Par Standard (INT16) Ch.8	PDO Parameter RxPDO 285	UINT8	RO	0x06 (6_{dec})
151C:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 285	OCTET-STRING[2]	RO	1E 17

*) EL4078 only

Index 151E AO RxPDO-Par Standard (Real32) Ch.8*)

Index (hex)	Name	Meaning	Data type	Flags	Default
151E:0	AO RxPDO-Par Standard (Real32) Ch.8	PDO Parameter RxPDO 287	UINT8	RO	0x06 (6_{dec})
151E:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 287	OCTET-STRING[2]	RO	1C 17

*) EL4078 only

Index 1620 AO RxPDO-Map Standard (INT16) Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1620:0	AO RxPDO-Map Standard (INT16) Ch.1	PDO Mapping RxPDO 33	UINT8	RO	0x01 (1_{dec})
1620:01	SubIndex 001	1. PDO Mapping entry (object 0x7000 (AO Outputs Ch.1), entry 0x11 (Value))	UINT32	RO	0x7000:11, 16

Index 1622 AO RxPDO-Map Standard (Real32) Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1622:0	AO RxPDO-Map Standard (Real32) Ch.1	PDO Mapping RxPDO 35	UINT8	RO	0x01 (1_{dec})
1622:01	SubIndex 001	1. PDO Mapping entry (object 0x7000 (AO Outputs Ch.1), entry 0x13 (Value (Real32)))	UINT32	RO	0x7000:13, 32

Index 1623 AO RxPDO-Map Cycle Counter Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1623:0	AO RxPDO-Map Cycle Counter Ch.1	PDO Mapping RxPDO 36	UINT8	RO	0x01 (1_{dec})
1623:01	SubIndex 001	1. PDO Mapping entry (object 0x7000 (AO Outputs Ch.1), entry 0x14 (Output Cycle Counter))	UINT32	RO	0x7000:14, 16

Index 1644 AO RxPDO-Map Standard (INT16) Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1644:0	AO RxPDO-Map Standard (INT16) Ch.2	PDO Mapping RxPDO 69	UINT8	RO	0x01 (1_{dec})
1644:01	SubIndex 001	1. PDO Mapping entry (object 0x7010 (AO Outputs Ch.2), entry 0x11 (Value))	UINT32	RO	0x7010:11, 16

Index 1646 AO RxPDO-Map Standard (Real32) Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1646:0	AO RxPDO-Map Standard (Real32) Ch.2	PDO Mapping RxPDO 71	UINT8	RO	0x01 (1_{dec})
1646:01	SubIndex 001	1. PDO Mapping entry (object 0x7010 (AO Outputs Ch.2), entry 0x13 (Value (Real32)))	UINT32	RO	0x7010:13, 32

Index 1647 AO RxPDO-Map Cycle Counter Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1647:0	AO RxPDO-Map Cycle Counter Ch.2	PDO Mapping RxPDO 72	UINT8	RO	0x01 (1_{dec})
1647:01	SubIndex 001	1. PDO Mapping entry (object 0x7010 (AO Outputs Ch.2), entry 0x14 (Output Cycle Counter))	UINT32	RO	0x7010:14, 16

Index 1668 AO RxPDO-Map Standard (INT16) Ch.3*)

Index (hex)	Name	Meaning	Data type	Flags	Default
1668:0	AO RxPDO-Map Standard (INT16) Ch.3	PDO Mapping RxPDO 105	UINT8	RO	0x01 (1_{dec})
1668:01	SubIndex 001	1. PDO Mapping entry (object 0x7020 (AO Outputs Ch.3), entry 0x11 (Value))	UINT32	RO	0x7020:11, 16

*) EL4074, EL4078 only

Index 166A AO RxPDO-Map Standard (Real32) Ch.3*)

Index (hex)	Name	Meaning	Data type	Flags	Default
166A:0	AO RxPDO-Map Standard (Real32) Ch.3	PDO Mapping RxPDO 107	UINT8	RO	0x01 (1_{dec})
166A:01	SubIndex 001	1. PDO Mapping entry (object 0x7020 (AO Outputs Ch.3), entry 0x13 (Value (Real32)))	UINT32	RO	0x7020:13, 32

*) EL4074, EL4078 only

Index 166B AO RxPDO-Map Cycle Counter Ch.3*)

Index (hex)	Name	Meaning	Data type	Flags	Default
166B:0	AO RxPDO-Map Cycle Counter Ch.3	PDO Mapping RxPDO 108	UINT8	RO	0x01 (1_{dec})
166B:01	SubIndex 001	1. PDO Mapping entry (object 0x7020 (AO Outputs Ch.3), entry 0x14 (Output Cycle Counter))	UINT32	RO	0x7020:14, 16

*) EL4074, EL4078 only

Index 168C AO RxPDO-Map Standard (INT16) Ch.4*)

Index (hex)	Name	Meaning	Data type	Flags	Default
168C:0	AO RxPDO-Map Standard (INT16) Ch.4	PDO Mapping RxPDO 141	UINT8	RO	0x01 (1_{dec})
168C:01	SubIndex 001	1. PDO Mapping entry (object 0x7030 (AO Outputs Ch.4), entry 0x11 (Value))	UINT32	RO	0x7030:11, 16

*) EL4074, EL4078 only

Index 168E AO RxPDO-Map Standard (Real32) Ch.4*)

Index (hex)	Name	Meaning	Data type	Flags	Default
168E:0	AO RxPDO-Map Standard (Real32) Ch.4	PDO Mapping RxPDO 143	UINT8	RO	0x01 (1_{dec})
168E:01	SubIndex 001	1. PDO Mapping entry (object 0x7030 (AO Outputs Ch.4), entry 0x13 (Value (Real32)))	UINT32	RO	0x7030:13, 32

*) EL4074, EL4078 only

Index 168F AO RxPDO-Map Cycle Counter Ch.4*)

Index (hex)	Name	Meaning	Data type	Flags	Default
168F:0	AO RxPDO-Map Cycle Counter Ch.4	PDO Mapping RxPDO 144	UINT8	RO	0x01 (1_{dec})
168F:01	SubIndex 001	1. PDO Mapping entry (object 0x7030 (AO Outputs Ch.4), entry 0x14 (Output Cycle Counter))	UINT32	RO	0x7030:14, 16

*) EL4074, EL4078 only

Index 16B0 AO RxPDO-Map Standard (INT16) Ch.5*)

Index (hex)	Name	Meaning	Data type	Flags	Default
16B0:0	AO RxPDO-Map Standard (INT16) Ch.5	PDO Mapping RxPDO 177	UINT8	RO	0x01 (1_{dec})
16B0:01	SubIndex 001	1. PDO Mapping entry (object 0x7040 (AO Outputs Ch.5), entry 0x11 (Value))	UINT32	RO	0x7040:11, 16

*) EL4078 only

Index 16B2 AO RxPDO-Map Standard (Real32) Ch.5*)

Index (hex)	Name	Meaning	Data type	Flags	Default
16B2:0	AO RxPDO-Map Standard (Real32) Ch.5	PDO Mapping RxPDO 179	UINT8	RO	0x01 (1_{dec})
16B2:01	SubIndex 001	1. PDO Mapping entry (object 0x7040 (AO Outputs Ch.5), entry 0x13 (Value (Real32)))	UINT32	RO	0x7040:13, 32

*) EL4078 only

Index 16B3 AO RxPDO-Map Cycle Counter Ch.5*)

Index (hex)	Name	Meaning	Data type	Flags	Default
16B3:0	AO RxPDO-Map Cycle Counter Ch.5	PDO Mapping RxPDO 180	UINT8	RO	0x01 (1_{dec})
16B3:01	SubIndex 001	1. PDO Mapping entry (object 0x7040 (AO Outputs Ch.5), entry 0x14 (Output Cycle Counter))	UINT32	RO	0x7040:14, 16

*) EL4078 only

Index 16D4 AO RxPDO-Map Standard (INT16) Ch.6*)

Index (hex)	Name	Meaning	Data type	Flags	Default
16D4:0	AO RxPDO-Map Standard (INT16) Ch.6	PDO Mapping RxPDO 213	UINT8	RO	0x01 (1_{dec})
16D4:01	SubIndex 001	1. PDO Mapping entry (object 0x7050 (AO Outputs Ch.6), entry 0x11 (Value))	UINT32	RO	0x7050:11, 16

*) EL4078 only

Index 16D6 AO RxPDO-Map Standard (Real32) Ch.6*)

Index (hex)	Name	Meaning	Data type	Flags	Default
16D6:0	AO RxPDO-Map Standard (Real32) Ch.6	PDO Mapping RxPDO 215	UINT8	RO	0x01 (1_{dec})
16D6:01	SubIndex 001	1. PDO Mapping entry (object 0x7050 (AO Outputs Ch.6), entry 0x13 (Value (Real32)))	UINT32	RO	0x7050:13, 32

*) EL4078 only

Index 16D7 AO RxPDO-Map Cycle Counter Ch.6*)

Index (hex)	Name	Meaning	Data type	Flags	Default
16D7:0	AO RxPDO-Map Cycle Counter Ch.6	PDO Mapping RxPDO 216	UINT8	RO	0x01 (1_{dec})
16D7:01	SubIndex 001	1. PDO Mapping entry (object 0x7050 (AO Outputs Ch.6), entry 0x14 (Output Cycle Counter))	UINT32	RO	0x7050:14, 16

*) EL4078 only

Index 16F8 AO RxPDO-Map Standard (INT16) Ch.7*)

Index (hex)	Name	Meaning	Data type	Flags	Default
16F8:0	AO RxPDO-Map Standard (INT16) Ch.7	PDO Mapping RxPDO 249	UINT8	RO	0x01 (1_{dec})
16F8:01	SubIndex 001	1. PDO Mapping entry (object 0x7060 (AO Outputs Ch.7), entry 0x11 (Value))	UINT32	RO	0x7060:11, 16

*) EL4078 only

Index 16FA AO RxPDO-Map Standard (Real32) Ch.7*)

Index (hex)	Name	Meaning	Data type	Flags	Default
16FA:0	AO RxPDO-Map Standard (Real32) Ch.7	PDO Mapping RxPDO 251	UINT8	RO	0x01 (1_{dec})
16FA:01	SubIndex 001	1. PDO Mapping entry (object 0x7060 (AO Outputs Ch.7), entry 0x13 (Value (Real32)))	UINT32	RO	0x7060:13, 32

*) EL4078 only

Index 16FB AO RxPDO-Map Cycle Counter Ch.7*)

Index (hex)	Name	Meaning	Data type	Flags	Default
16FB:0	AO RxPDO-Map Cycle Counter Ch.7	PDO Mapping RxPDO 252	UINT8	RO	0x01 (1_{dec})
16FB:01	SubIndex 001	1. PDO Mapping entry (object 0x7060 (AO Outputs Ch.7), entry 0x14 (Output Cycle Counter))	UINT32	RO	0x7060:14, 16

*) EL4078 only

Index 171C AO RxPDO-Map Standard (INT16) Ch.8*)

Index (hex)	Name	Meaning	Data type	Flags	Default
171C:0	AO RxPDO-Map Standard (INT16) Ch.8	PDO Mapping RxPDO 285	UINT8	RO	0x01 (1_{dec})
171C:01	SubIndex 001	1. PDO Mapping entry (object 0x7070 (AO Outputs Ch.8), entry 0x11 (Value))	UINT32	RO	0x7070:11, 16

*) EL4078 only

Index 171E AO RxPDO-Map Standard (Real32) Ch.8*)

Index (hex)	Name	Meaning	Data type	Flags	Default
171E:0	AO RxPDO-Map Standard (Real32) Ch.8	PDO Mapping RxPDO 287	UINT8	RO	0x01 (1_{dec})
171E:01	SubIndex 001	1. PDO Mapping entry (object 0x7070 (AO Outputs Ch.8), entry 0x13 (Value (Real32)))	UINT32	RO	0x7070:13, 32

*) EL4078 only

Index 171F AO RxPDO-Map Cycle Counter Ch.8*)

Index (hex)	Name	Meaning	Data type	Flags	Default
171F:0	AO RxPDO-Map Cycle Counter Ch.8	PDO Mapping RxPDO 288	UINT8	RO	0x01 (1_{dec})
171F:01	SubIndex 001	1. PDO Mapping entry (object 0x7070 (AO Outputs Ch.8), entry 0x14 (Output Cycle Counter))	UINT32	RO	0x7070:14, 16

*) EL4078 only

Index 1A20 AO TxPDO-Map Status Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1A20:0	AO TxPDO-Map Status Ch.1	PDO Mapping TxPDO 33	UINT8	RO	0x08 (8_{dec})
1A20:01	SubIndex 001	1. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1A20:02	SubIndex 002	2. PDO Mapping entry (object 0x6000 (AO Inputs Ch.1), entry 0x02 (Output Overload))	UINT32	RO	0x6000:02, 1
1A20:03	SubIndex 003	3. PDO Mapping entry (object 0x6000 (AO Inputs Ch.1), entry 0x03 (Underrange))	UINT32	RO	0x6000:03, 1
1A20:04	SubIndex 004	4. PDO Mapping entry (object 0x6000 (AO Inputs Ch.1), entry 0x04 (Overrange))	UINT32	RO	0x6000:04, 1
1A20:05	SubIndex 005	5. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1A20:06	SubIndex 006	6. PDO Mapping entry (object 0x6000 (AO Inputs Ch.1), entry 0x06 (Warning))	UINT32	RO	0x6000:06, 1
1A20:07	SubIndex 007	7. PDO Mapping entry (object 0x6000 (AO Inputs Ch.1), entry 0x07 (Error))	UINT32	RO	0x6000:07, 1
1A20:08	SubIndex 008	8. PDO Mapping entry (9 bits align)	UINT32	RO	0x0000:00, 9

Index 1A41 AO TxPDO-Map Status Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1A41:0	AO TxPDO-Map Status Ch.2	PDO Mapping TxPDO 66	UINT8	RO	0x08 (8_{dec})
1A41:01	SubIndex 001	1. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1A41:02	SubIndex 002	2. PDO Mapping entry (object 0x6010 (AO Inputs Ch.2), entry 0x02 (Output Overload))	UINT32	RO	0x6010:02, 1
1A41:03	SubIndex 003	3. PDO Mapping entry (object 0x6010 (AO Inputs Ch.2), entry 0x03 (Underrange))	UINT32	RO	0x6010:03, 1
1A41:04	SubIndex 004	4. PDO Mapping entry (object 0x6010 (AO Inputs Ch.2), entry 0x04 (Overrange))	UINT32	RO	0x6010:04, 1
1A41:05	SubIndex 005	5. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1A41:06	SubIndex 006	6. PDO Mapping entry (object 0x6010 (AO Inputs Ch.2), entry 0x06 (Warning))	UINT32	RO	0x6010:06, 1
1A41:07	SubIndex 007	7. PDO Mapping entry (object 0x6010 (AO Inputs Ch.2), entry 0x07 (Error))	UINT32	RO	0x6010:07, 1
1A41:08	SubIndex 008	8. PDO Mapping entry (9 bits align)	UINT32	RO	0x0000:00, 9

Index 1A62 AO TxPDO-Map Status Ch.3*)

Index (hex)	Name	Meaning	Data type	Flags	Default
1A62:0	AO TxPDO-Map Status Ch.3	PDO Mapping TxPDO 99	UINT8	RO	0x08 (8_{dec})
1A62:01	SubIndex 001	1. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1A62:02	SubIndex 002	2. PDO Mapping entry (object 0x6020 (AO Inputs Ch.3), entry 0x02 (Output Overload))	UINT32	RO	0x6020:02, 1
1A62:03	SubIndex 003	3. PDO Mapping entry (object 0x6020 (AO Inputs Ch.3), entry 0x03 (Underrange))	UINT32	RO	0x6020:03, 1
1A62:04	SubIndex 004	4. PDO Mapping entry (object 0x6020 (AO Inputs Ch.3), entry 0x04 (Overrange))	UINT32	RO	0x6020:04, 1
1A62:05	SubIndex 005	5. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1A62:06	SubIndex 006	6. PDO Mapping entry (object 0x6020 (AO Inputs Ch.3), entry 0x06 (Warning))	UINT32	RO	0x6020:06, 1
1A62:07	SubIndex 007	7. PDO Mapping entry (object 0x6020 (AO Inputs Ch.3), entry 0x07 (Error))	UINT32	RO	0x6020:07, 1
1A62:08	SubIndex 008	8. PDO Mapping entry (9 bits align)	UINT32	RO	0x0000:00, 9

*) EL4074, EL4078 only

Index 1A83 AO TxPDO-Map Status Ch.4*)

Index (hex)	Name	Meaning	Data type	Flags	Default
1A83:0	AO TxPDO-Map Status Ch.4	PDO Mapping TxPDO 132	UINT8	RO	0x08 (8_{dec})
1A83:01	SubIndex 001	1. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1A83:02	SubIndex 002	2. PDO Mapping entry (object 0x6030 (AO Inputs Ch.4), entry 0x02 (Output Overload))	UINT32	RO	0x6030:02, 1
1A83:03	SubIndex 003	3. PDO Mapping entry (object 0x6030 (AO Inputs Ch.4), entry 0x03 (Underrange))	UINT32	RO	0x6030:03, 1
1A83:04	SubIndex 004	4. PDO Mapping entry (object 0x6030 (AO Inputs Ch.4), entry 0x04 (Overrange))	UINT32	RO	0x6030:04, 1
1A83:05	SubIndex 005	5. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1A83:06	SubIndex 006	6. PDO Mapping entry (object 0x6030 (AO Inputs Ch.4), entry 0x06 (Warning))	UINT32	RO	0x6030:06, 1
1A83:07	SubIndex 007	7. PDO Mapping entry (object 0x6030 (AO Inputs Ch.4), entry 0x07 (Error))	UINT32	RO	0x6030:07, 1
1A83:08	SubIndex 008	8. PDO Mapping entry (9 bits align)	UINT32	RO	0x0000:00, 9

*) EL4074, EL4078 only

Index 1AA4 AO TxPDO-Map Status Ch.5*)

Index (hex)	Name	Meaning	Data type	Flags	Default
1AA4:0	AO TxPDO-Map Status Ch.5	PDO Mapping TxPDO 165	UINT8	RO	0x08 (8_{dec})
1AA4:01	SubIndex 001	1. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1AA4:02	SubIndex 002	2. PDO Mapping entry (object 0x6040 (AO Inputs Ch.5), entry 0x02 (Output Overload))	UINT32	RO	0x6040:02, 1
1AA4:03	SubIndex 003	3. PDO Mapping entry (object 0x6040 (AO Inputs Ch.5), entry 0x03 (Underrange))	UINT32	RO	0x6040:03, 1
1AA4:04	SubIndex 004	4. PDO Mapping entry (object 0x6040 (AO Inputs Ch.5), entry 0x04 (Overrange))	UINT32	RO	0x6040:04, 1
1AA4:05	SubIndex 005	5. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1AA4:06	SubIndex 006	6. PDO Mapping entry (object 0x6040 (AO Inputs Ch.5), entry 0x06 (Warning))	UINT32	RO	0x6040:06, 1
1AA4:07	SubIndex 007	7. PDO Mapping entry (object 0x6040 (AO Inputs Ch.5), entry 0x07 (Error))	UINT32	RO	0x6040:07, 1
1AA4:08	SubIndex 008	8. PDO Mapping entry (9 bits align)	UINT32	RO	0x0000:00, 9

*) EL4078 only

Index 1AC5 AO TxPDO-Map Status Ch.6*)

Index (hex)	Name	Meaning	Data type	Flags	Default
1AC5:0	AO TxPDO-Map Status Ch.6	PDO Mapping TxPDO 198	UINT8	RO	0x08 (8_{dec})
1AC5:01	SubIndex 001	1. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1AC5:02	SubIndex 002	2. PDO Mapping entry (object 0x6050 (AO Inputs Ch.6), entry 0x02 (Output Overload))	UINT32	RO	0x6050:02, 1
1AC5:03	SubIndex 003	3. PDO Mapping entry (object 0x6050 (AO Inputs Ch.6), entry 0x03 (Underrange))	UINT32	RO	0x6050:03, 1
1AC5:04	SubIndex 004	4. PDO Mapping entry (object 0x6050 (AO Inputs Ch.6), entry 0x04 (Overrange))	UINT32	RO	0x6050:04, 1
1AC5:05	SubIndex 005	5. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1AC5:06	SubIndex 006	6. PDO Mapping entry (object 0x6050 (AO Inputs Ch.6), entry 0x06 (Warning))	UINT32	RO	0x6050:06, 1
1AC5:07	SubIndex 007	7. PDO Mapping entry (object 0x6050 (AO Inputs Ch.6), entry 0x07 (Error))	UINT32	RO	0x6050:07, 1
1AC5:08	SubIndex 008	8. PDO Mapping entry (9 bits align)	UINT32	RO	0x0000:00, 9

*) EL4078 only

Index 1AE6 AO TxPDO-Map Status Ch.7*)

Index (hex)	Name	Meaning	Data type	Flags	Default
1AE6:0	AO TxPDO-Map Status Ch.7	PDO Mapping TxPDO 231	UINT8	RO	0x08 (8_{dec})
1AE6:01	SubIndex 001	1. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1AE6:02	SubIndex 002	2. PDO Mapping entry (object 0x6060 (AO Inputs Ch.7), entry 0x02 (Output Overload))	UINT32	RO	0x6060:02, 1
1AE6:03	SubIndex 003	3. PDO Mapping entry (object 0x6060 (AO Inputs Ch.7), entry 0x03 (Underrange))	UINT32	RO	0x6060:03, 1
1AE6:04	SubIndex 004	4. PDO Mapping entry (object 0x6060 (AO Inputs Ch.7), entry 0x04 (Overrange))	UINT32	RO	0x6060:04, 1
1AE6:05	SubIndex 005	5. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1AE6:06	SubIndex 006	6. PDO Mapping entry (object 0x6060 (AO Inputs Ch.9), entry 0x06 (Warning))	UINT32	RO	0x6060:06, 1
1AE6:07	SubIndex 007	7. PDO Mapping entry (object 0x6060 (AO Inputs Ch.7), entry 0x07 (Error))	UINT32	RO	0x6060:07, 1
1AE6:08	SubIndex 008	8. PDO Mapping entry (9 bits align)	UINT32	RO	0x0000:00, 9

*) EL4078 only

Index 1B07 AO TxPDO-Map Status Ch.8*)

Index (hex)	Name	Meaning	Data type	Flags	Default
1B07:0	AO TxPDO-Map Status Ch.8	PDO Mapping TxPDO 264	UINT8	RO	0x08 (8_{dec})
1B07:01	SubIndex 001	1. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1B07:02	SubIndex 002	2. PDO Mapping entry (object 0x6070 (AO Inputs Ch.8), entry 0x02 (Output Overload))	UINT32	RO	0x6070:02, 1
1B07:03	SubIndex 003	3. PDO Mapping entry (object 0x6070 (AO Inputs Ch.8), entry 0x03 (Underrange))	UINT32	RO	0x6070:03, 1
1B07:04	SubIndex 004	4. PDO Mapping entry (object 0x6070 (AO Inputs Ch.8), entry 0x04 (Overrange))	UINT32	RO	0x6070:04, 1
1B07:05	SubIndex 005	5. PDO Mapping entry (1 bits align)	UINT32	RO	0x0000:00, 1
1B07:06	SubIndex 006	6. PDO Mapping entry (object 0x6070 (AO Inputs Ch.8), entry 0x06 (Warning))	UINT32	RO	0x6070:06, 1
1B07:07	SubIndex 007	7. PDO Mapping entry (object 0x6070 (AO Inputs Ch.8), entry 0x07 (Error))	UINT32	RO	0x6070:07, 1
1B07:08	SubIndex 008	8. PDO Mapping entry (9 bits align)	UINT32	RO	0x0000:00, 9

*) EL4078 only

Index 1C00 Sync manager type

Index (hex)	Name	Meaning	Data type	Flags	Default
1C00:0	Sync manager type	Using the sync managers	UINT8	RO	0x04 (4_{dec})
1C00:01	SubIndex 001	Sync-Manager Type Channel 1: Mailbox Write	UINT8	RO	0x01 (1_{dec})
1C00:02	SubIndex 002	Sync-Manager Type Channel 2: Mailbox Read	UINT8	RO	0x02 (2_{dec})
1C00:03	SubIndex 003	Sync-Manager Type Channel 3: Process Data Write (Outputs)	UINT8	RO	0x03 (3_{dec})
1C00:04	SubIndex 004	Sync-Manager Type Channel 4: Process Data Read (Inputs)	UINT8	RO	0x04 (4_{dec})

Index 1C12 RxPDO assign

Index (hex)	Name	Meaning	Data type	Flags	Default
1C12:0	RxPDO assign	PDO Assign Outputs	UINT8	RW	0x08 (8_{dec})
1C12:01	Subindex 001	1. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x1622 (5666_{dec})
1C12:02	Subindex 002	2. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x1646 (5702_{dec})
1C12:03	Subindex 003	3. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x166A (5738_{dec})
1C12:04	Subindex 004	4. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x168E (5774_{dec})
1C12:05	Subindex 005	5. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x16B2 (5810_{dec})
1C12:06	Subindex 006	6. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x16D6 (5846_{dec})
1C12:07	Subindex 007	7. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x16FA (5882_{dec})
1C12:08	Subindex 008	8. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x171E (5918_{dec})
1C12:09	Subindex 009	9. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x0000 (0_{dec})
1C12:0A	Subindex 010	10. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x0000 (0_{dec})
1C12:0B	Subindex 011	11. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x0000 (0_{dec})
1C12:0C	Subindex 012	12. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x0000 (0_{dec})
1C12:0D	Subindex 013	13. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x0000 (0_{dec})
1C12:0E	Subindex 014	14. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x0000 (0_{dec})
1C12:0F	Subindex 015	15. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x0000 (0_{dec})
1C12:10	Subindex 016	16. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x0000 (0_{dec})

Index 1C13 TxPDO assign

Index (hex)	Name	Meaning	Data type	Flags	Default
1C13:0	TxPDO assign	PDO Assign Inputs	UINT8	RW	0x08 (8_{dec})
1C13:01	Subindex 001	1. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A20 (6688_{dec})
1C13:02	Subindex 002	2. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A41 (6721_{dec})
1C13:03	Subindex 003	3. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A62 (6754_{dec})
1C13:04	Subindex 004	4. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A83 (6787_{dec})
1C13:05	Subindex 005	5. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1AA4 (6820_{dec})
1C13:06	Subindex 006	6. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1AC5 (6853_{dec})
1C13:07	Subindex 007	7. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1AE6 (6886_{dec})
1C13:08	Subindex 008	8. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1B07 (6919_{dec})

Index 1C32 SM output parameter

Index (hex)	Name	Meaning	Data type	Flags	Default
1C32:0	SM output parameter	Synchronization parameters for the outputs	UINT8	RO	0x20 (32_{dec})
1C32:01	Sync mode	Current synchronization mode: • 0: Free Run • 1: Synchron with SM 2 Event • 2: DC-Mode - Synchron with SYNC0 Event • 3: DC-Mode - Synchron with SYNC1 Event	UINT16	RW	0x0001 (1_{dec})
1C32:02	Cycle time	Cycle time (in ns): • Free Run: cycle time of the local timer • Synchron with SM 2 Event: cycle time of the master • DC-Mode: SYNC0/SYNC1 Cycle Time	UINT32	RW	0x000F4240 (1000000_{dec})
1C32:03	Shift time	Time between SYNC0 event and output of the outputs (in ns, DC mode only)	UINT32	RO	0x00000000 (0_{dec})
1C32:04	Sync modes supported	Sync modes supported: • Bit 0 = 1: Free Run is supported • Bit 1 = 1: Synchron with SM 2 Event is supported • Bit 2-3 = 01: DC-Mode is supported • Bit 4-5 = 10: Output Shift with SYNC1 Event (only DC mode) • Bit 14 = 1: dynamic times (measurement through writing of 1C32:08)	UINT16	RO	0x0002 (2_{dec})
1C32:05	Minimum cycle time	Minimum cycle time (in ns)	UINT32	RO	0x000F4240 (1000000_{dec})
1C32:06	Calc and copy time	Minimum time between SYNC0 and SYNC1 event (in ns, DC Mode only)	UINT32	RO	0x00000000 (0_{dec})
1C32:09	Maximum delay time	Time between SYNC1 event and output of the outputs (in ns, DC mode only)	UINT32	RO	0x00000000 (0_{dec})
1C32:0B	SM event missed counter	Number of missed SM events in OPERATIONAL (DC Mode only)	UINT16	RO	0x0000 (0_{dec})
1C32:0C	Cycle exceeded counter	Number of occasions the cycle time was exceeded in OPERATIONAL (cycle was not completed in time or the next cycle began too early)	UINT16	RO	0x0000 (0_{dec})
1C32:0D	Shift too short counter	Number of intervals between SYNC0 and SYNC1 events that are too short (DC Mode only)	UINT16	RO	0x0000 (0_{dec})
1C32:20	Sync error	The synchronization was not correct in the last cycle (outputs were output too late; DC Mode only)	BOOLEAN	RO	0x00 (0_{dec})

Index 1C33 SM input parameter

Index (hex)	Name	Meaning	Data type	Flags	Default
1C33:0	SM input parameter	Synchronization parameters for the inputs	UINT8	RO	0x20 (32_{dec})
1C33:01	Sync mode	Current synchronization mode: • 0: Free Run • 1: Synchron with SM 3 Event (no outputs available) • 2: DC - Synchron with SYNC0 Event • 3: DC - Synchron with SYNC1 Event • 34: Synchron with SM 2 Event (outputs available)	UINT16	RW	0x0022 (34_{dec})
1C33:02	Cycle time	• as 1C32:02	UINT32	RW	0x000F4240 (1000000_{dec})
1C33:03	Shift time	Time between SYNC0 event and reading of the inputs (in ns, DC Mode only)	UINT32	RO	0x00000000 (0_{dec})
1C33:04	Sync modes supported	Sync modes supported: • Bit 0: Free Run is supported • Bit 1: Synchron with SM 2 Event is supported (outputs available) • Bit 1: Synchron with SM 3 Event is supported (no outputs available) • Bit 2-3 = 01: DC-Mode is supported • Bit 4-5 = 01: Input shift through local event (outputs available) • Bit 4-5 = 10: Input shift with SYNC1 event (no outputs available) • Bit 14 = 1: dynamic times (measurement through writing of 1C32:08 or 1C33:08)	UINT16	RO	0x0002 (2_{dec})
1C33:05	Minimum cycle time	as 1C32:05	UINT32	RO	0x000F4240 (1000000_{dec})
1C33:06	Calc and copy time	Time between reading of the inputs and the inputs being available for the master (in ns, DC Mode only)	UINT32	RO	0x00000000 (0_{dec})
1C33:07	Minimum delay time		UINT32	RO	0x00000000 (0_{dec})
1C33:09	Maximum delay time	Time between SYNC1 event and reading of the inputs (in ns, DC Mode only)	UINT32	RO	0x00000000 (0_{dec})
1C33:0B	SM event missed counter	as 1C32:11	UINT16	RO	0x0000 (0_{dec})
1C33:0C	Cycle exceeded counter	as 1C32:12	UINT16	RO	0x0000 (0_{dec})
1C33:0D	Shift too short counter	as 1C32:13	UINT16	RO	0x0000 (0_{dec})
1C33:20	Sync error	as 1C32:32	BOOLEAN	RO	0x00 (0_{dec})

Index F008 Code word

Index (hex)	Name	Meaning	Data type	Flags	Default
F008:0	Code word	Code word (currently reserved)	UINT32	RW	0x00000000 (0_{dec})

Index F009 Password protection

Index (hex)	Name	Meaning	Data type	Flags	Default
F009:0	Password protection	Password protection user calibration	UINT32	RW	0x00000000 (0_{dec})

Index F081 Download revision

Index (hex)	Name	Meaning	Data type	Flags	Default
F081:0	Download revision	Max. Subindex	UINT8	RO	0x01 (1_{dec})
F081:01	Download revision	The subindex 0xF081:01 (Download revision) describes the revision level of the module.	UINT32	RW	0x00000000 (0_{dec})

6 Appendix

6.1 EtherCAT AL Status Codes

For detailed information please refer to the EtherCAT system description.

6.2 Firmware compatibility

Beckhoff EtherCAT devices are delivered with the latest available firmware version. Compatibility of firmware and hardware is mandatory; not every combination ensures compatibility. The overview below shows the hardware versions on which a firmware can be operated.

Note

- It is recommended to use the newest possible firmware for the respective hardware
- Beckhoff is not under any obligation to provide customers with free firmware updates for delivered products.

NOTICE

Risk of damage to the device!

Pay attention to the instructions for firmware updates on the separate page [▶ 192].
If a device is placed in BOOTSTRAP mode for a firmware update, it does not check when downloading whether the new firmware is suitable.
This can result in damage to the device! Therefore, always make sure that the firmware is suitable for the hardware version!

EL4072			
Hardware (HW)	**Firmware**	**Revision no.**	**Release date**
00 – 01*	01*	EL4072-0000-0016	2025/07

EL4074			
Hardware (HW)	**Firmware**	**Revision no.**	**Release date**
00 – 01*	01*	EL4074-0000-0016	2025/07

EL4078			
Hardware (HW)	**Firmware**	**Revision no.**	**Release date**
00 - 01*	01*	EL4374-0000-0016	2025/07

*) This is the current compatible firmware/hardware version at the time of the preparing this documentation. Check on the Beckhoff web page whether more up-to-date documentation is available.

6.3 Firmware Update

This section describes the device update for Beckhoff EtherCAT slaves from the ED/EF, EL/ES, ELM, EM, EK, EP, EPP and ERP series. A firmware update should only be carried out after consultation with Beckhoff support.

NOTICE

Only use TwinCAT 3 software!

A firmware update of Beckhoff IO devices must only be performed with a TwinCAT 3 installation. It is recommended to build as up-to-date as possible, available for free download on the Beckhoff website.

To update the firmware, TwinCAT can be operated in the so-called FreeRun mode, a paid license is not required.

The device to be updated can usually remain in the installation location, but TwinCAT has to be operated in the FreeRun. Please make sure that EtherCAT communication is trouble-free (no LostFrames etc.).

Other EtherCAT master software, such as the EtherCAT Configurator, should not be used, as they may not support the complexities of updating firmware, EEPROM and other device components.

Storage locations

An EtherCAT slave stores operating data in up to three locations:

- Each EtherCAT slave has a device description, consisting of identity (name, product code), timing specifications, communication settings, etc.
  This device description (ESI; EtherCAT Slave Information) can be downloaded from the Beckhoff website in the download area as a zip file and used in EtherCAT masters for offline configuration, e.g. in TwinCAT.
  Above all, each EtherCAT slave carries its device description (ESI) electronically readable in a local memory chip, the so-called **ESI EEPROM**. When the slave is switched on, this description is loaded locally in the slave and informs it of its communication configuration; on the other hand, the EtherCAT master can identify the slave in this way and, among other things, set up the EtherCAT communication accordingly.

NOTICE

Application-specific writing of the ESI-EEPROM

The ESI is developed by the device manufacturer according to ETG standard and released for the corresponding product.
- Meaning for the ESI file: Modification on the application side (i.e. by the user) is not permitted.
- Meaning for the ESI EEPROM: Even if a writeability is technically given, the ESI parts in the EEPROM and possibly still existing free memory areas must not be changed beyond the normal update process. Especially for cyclic memory processes (operating hours counter etc.), dedicated memory products such as EL6080 or IPC's own NOVRAM must be used.

- Depending on functionality and performance EtherCAT slaves have one or several local controllers for processing I/O data. The corresponding program is the so-called **firmware** in *.efw format.
- In some EtherCAT slaves the EtherCAT communication may also be integrated in these controllers. In this case the controller is usually a so-called **FPGA** chip with *.rbf firmware.

Customers can access the data via the EtherCAT fieldbus and its communication mechanisms. Acyclic mailbox communication or register access to the ESC is used for updating or reading of these data.

The TwinCAT System Manager offers mechanisms for programming all three parts with new data, if the slave is set up for this purpose. Generally the slave does not check whether the new data are suitable, i.e. it may no longer be able to operate if the data are unsuitable.

Simplified update by bundle firmware

The update using so-called **bundle firmware** is more convenient: in this case the controller firmware and the ESI description are combined in a *.efw file; during the update both the firmware and the ESI are changed in the terminal. For this to happen it is necessary

- for the firmware to be in a packed format: recognizable by the file name, which also contains the revision number, e.g. ELxxxx-xxxx_REV0016_SW01.efw
- for password=1 to be entered in the download dialog. If password=0 (default setting) only the firmware update is carried out, without an ESI update.
- for the device to support this function. The function usually cannot be retrofitted; it is a component of many new developments from year of manufacture 2016.

Following the update, its success should be verified

- ESI/Revision: e.g. by means of an online scan in TwinCAT ConfigMode/FreeRun – this is a convenient way to determine the revision
- Firmware: e.g. by looking in the online CoE of the device

NOTICE

Risk of damage to the device!

✓ Note the following when downloading new device files

a) Firmware downloads to an EtherCAT device must not be interrupted

b) Flawless EtherCAT communication must be ensured. CRC errors or LostFrames must be avoided.

c) The power supply must adequately dimensioned. The signal level must meet the specification.

⇨ In the event of malfunctions during the update process the EtherCAT device may become unusable and require re-commissioning by the manufacturer.

6.3.1 Device description ESI file/XML

NOTICE

Attention regarding update of the ESI description/EEPROM

Some slaves have stored calibration and configuration data from the production in the EEPROM. These are irretrievably overwritten during an update.

The ESI device description is stored locally on the slave and loaded on start-up. Each device description has a unique identifier consisting of slave name (9 characters/digits) and a revision number (4 digits). Each slave configured in the System Manager shows its identifier in the EtherCAT tab:

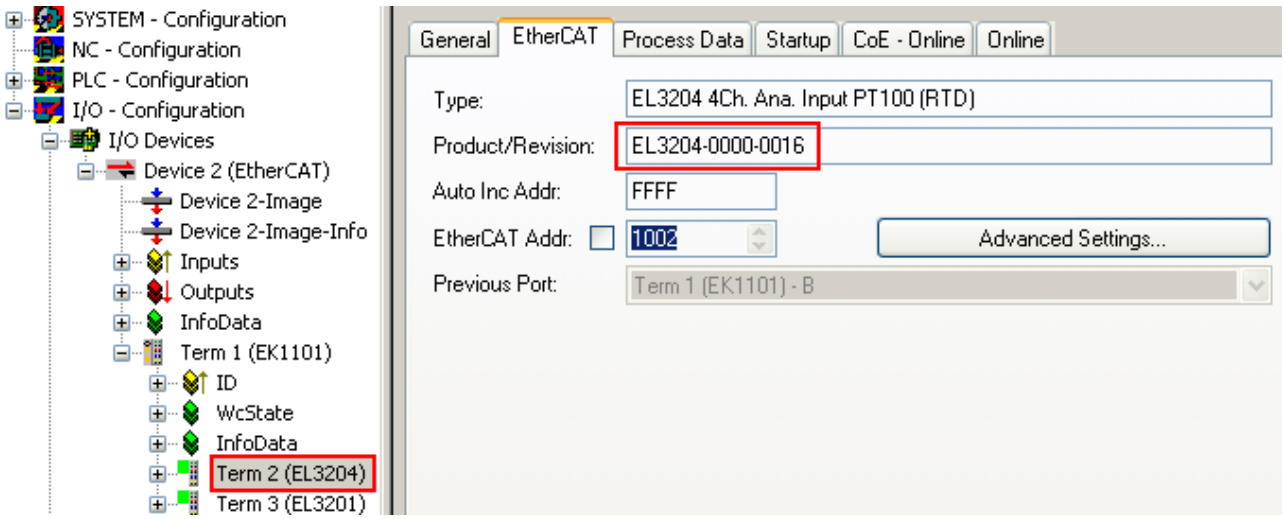


Fig. 187: Device identifier consisting of name EL3204-0000 and revision -0016

The configured identifier must be compatible with the actual device description used as hardware, i.e. the description which the slave has loaded on start-up (in this case EL3204). Normally the configured revision must be the same or lower than that actually present in the terminal network.

For further information on this, please refer to the EtherCAT system documentation.

Update of XML/ESI description

The device revision is closely linked to the firmware and hardware used. Incompatible combinations lead to malfunctions or even final shutdown of the device. Corresponding updates should only be carried out in consultation with Beckhoff support.

Display of ESI slave identifier

The simplest way to ascertain compliance of configured and actual device description is to scan the EtherCAT boxes in TwinCAT mode Config/FreeRun:

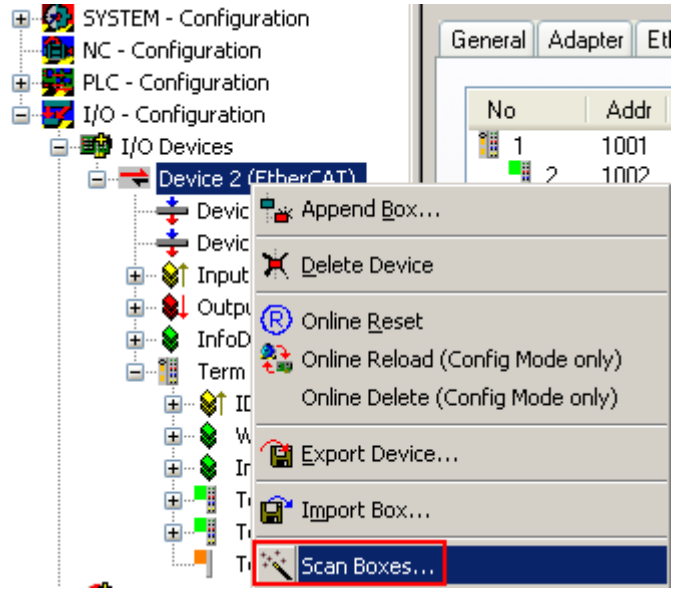


Fig. 188: Scan the subordinate field by right-clicking on the EtherCAT device

If the found field matches the configured field, the display shows



Fig. 189: Configuration is identical

otherwise a change dialog appears for entering the actual data in the configuration.

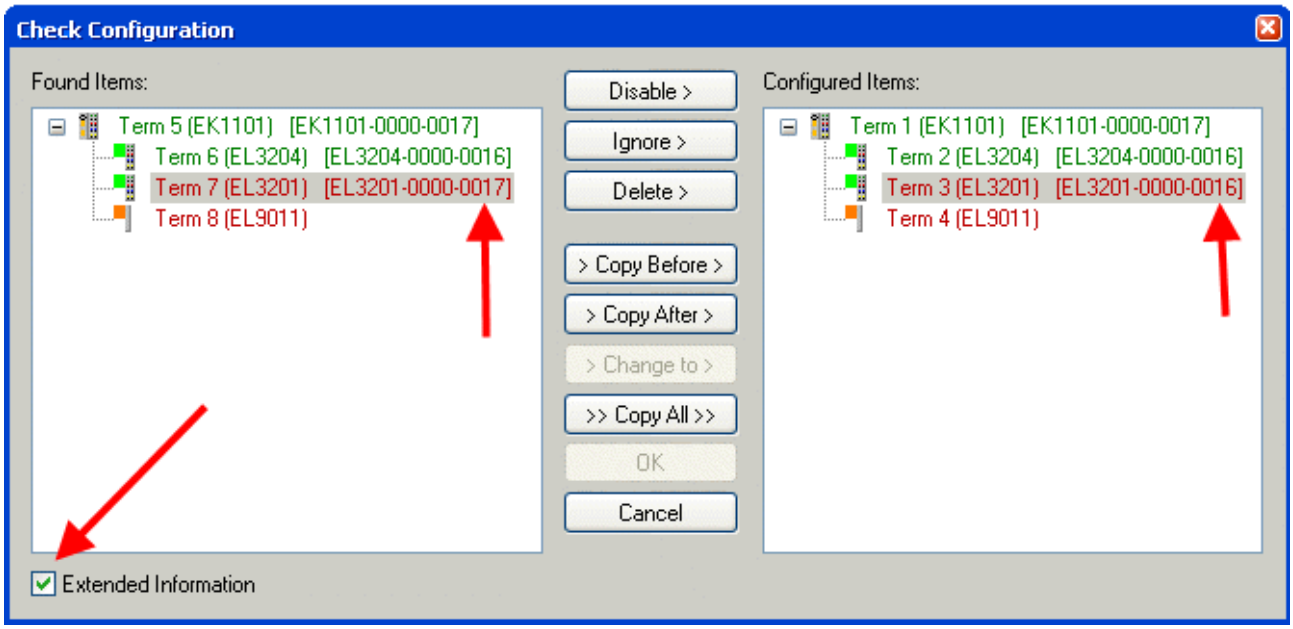


Fig. 190: Change dialog

In this example in Fig. *Change dialog*, an EL3201-0000-**0017** was found, while an EL3201-0000-**0016** was configured. In this case the configuration can be adapted with the *Copy Before* button. The *Extended Information* checkbox must be set in order to display the revision.

Changing the ESI slave identifier

The ESI/EEPROM identifier can be updated as follows under TwinCAT:

- Trouble-free EtherCAT communication must be established with the slave.
- The state of the slave is irrelevant.
- Right-clicking on the slave in the online display opens the *EEPROM Update* dialog, Fig. *EEPROM Update*

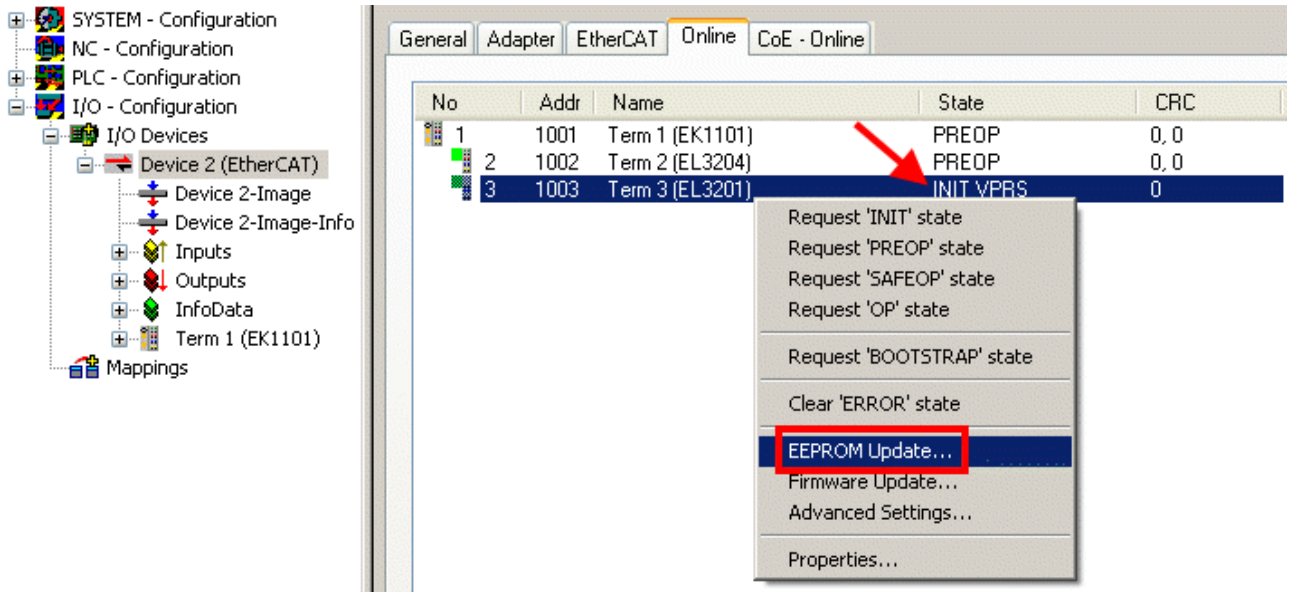


Fig. 191: EEPROM Update

The new ESI description is selected in the following dialog, see Fig. *Selecting the new ESI.* The checkbox *Show Hidden Devices* also displays older, normally hidden versions of a slave.

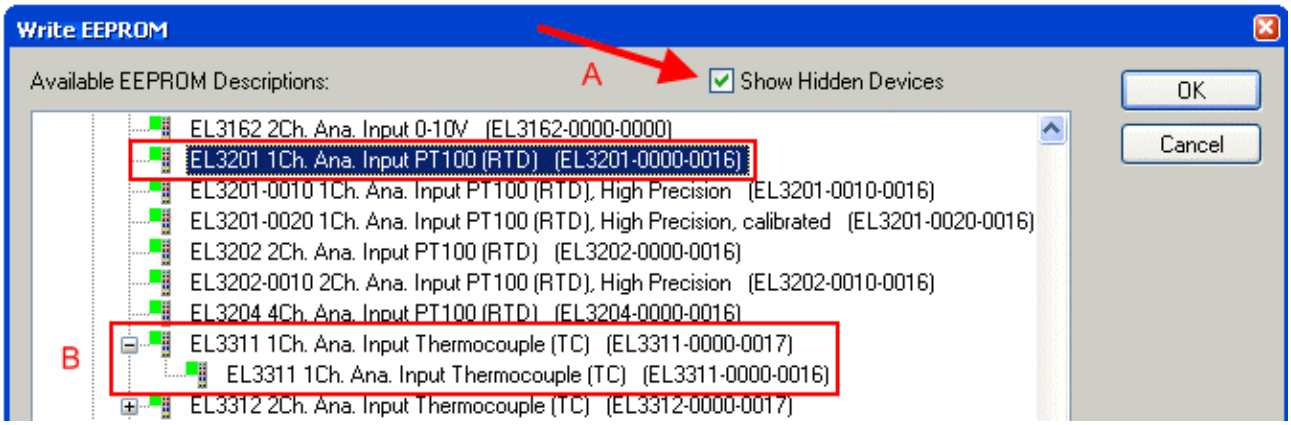


Fig. 192: Selecting the new ESI

A progress bar in the System Manager shows the progress. Data are first written, then verified.

The change only takes effect after a restart.

Most EtherCAT devices read a modified ESI description immediately or after startup from the INIT. Some communication settings such as distributed clocks are only read during power-on. The EtherCAT slave therefore has to be switched off briefly in order for the change to take effect.

6.3.2 Firmware explanation

Determining the firmware version

Determining the version via the TwinCAT System Manager

The TwinCAT System Manager shows the version of the controller firmware if the master can access the slave online. Click on the E-Bus Terminal whose controller firmware you want to check (in the example terminal 2 (EL3204)) and select the tab *CoE Online* (CAN over EtherCAT).

CoE Online and Offline CoE

Two CoE directories are available:
• **online**: This is offered in the EtherCAT slave by the controller, if the EtherCAT slave supports this. This CoE directory can only be displayed if a slave is connected and operational.
• **offline**: The EtherCAT Slave Information ESI/XML may contain the default content of the CoE. This CoE directory can only be displayed if it is included in the ESI (e.g. “Beckhoff EL5xxx.xml”).

The Advanced button must be used for switching between the two views.

In Fig. *Display of EL3204 firmware version* the firmware version of the selected EL3204 is shown as 03 in CoE entry 0x100A.

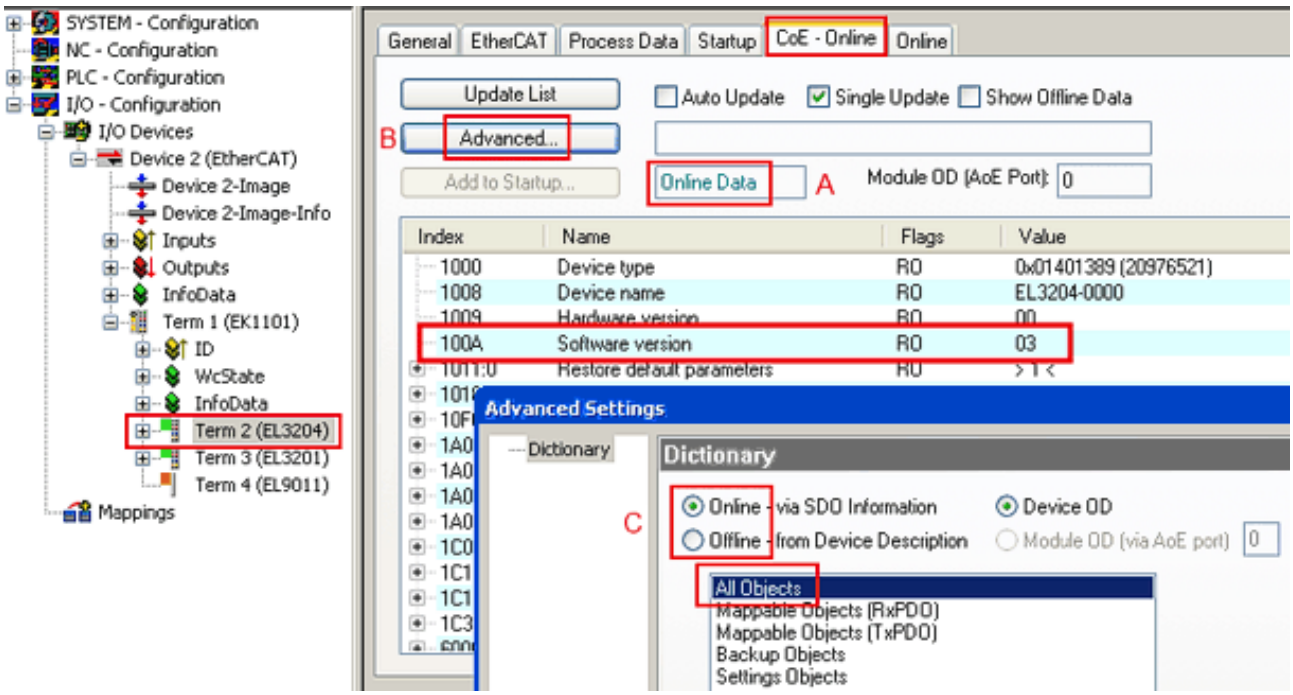


Fig. 193: Display of EL3204 firmware version

In (A) TwinCAT 2.11 shows that the Online CoE directory is currently displayed. If this is not the case, the Online directory can be loaded via the *Online* option in Advanced Settings (B) and double-clicking on *AllObjects*.

6.3.3 Updating controller firmware *.efw

CoE directory

The Online CoE directory is managed by the controller and stored in a dedicated EEPROM, which is generally not changed during a firmware update.

Switch to the *Online* tab to update the controller firmware of a slave, see Fig. *Firmware Update.*

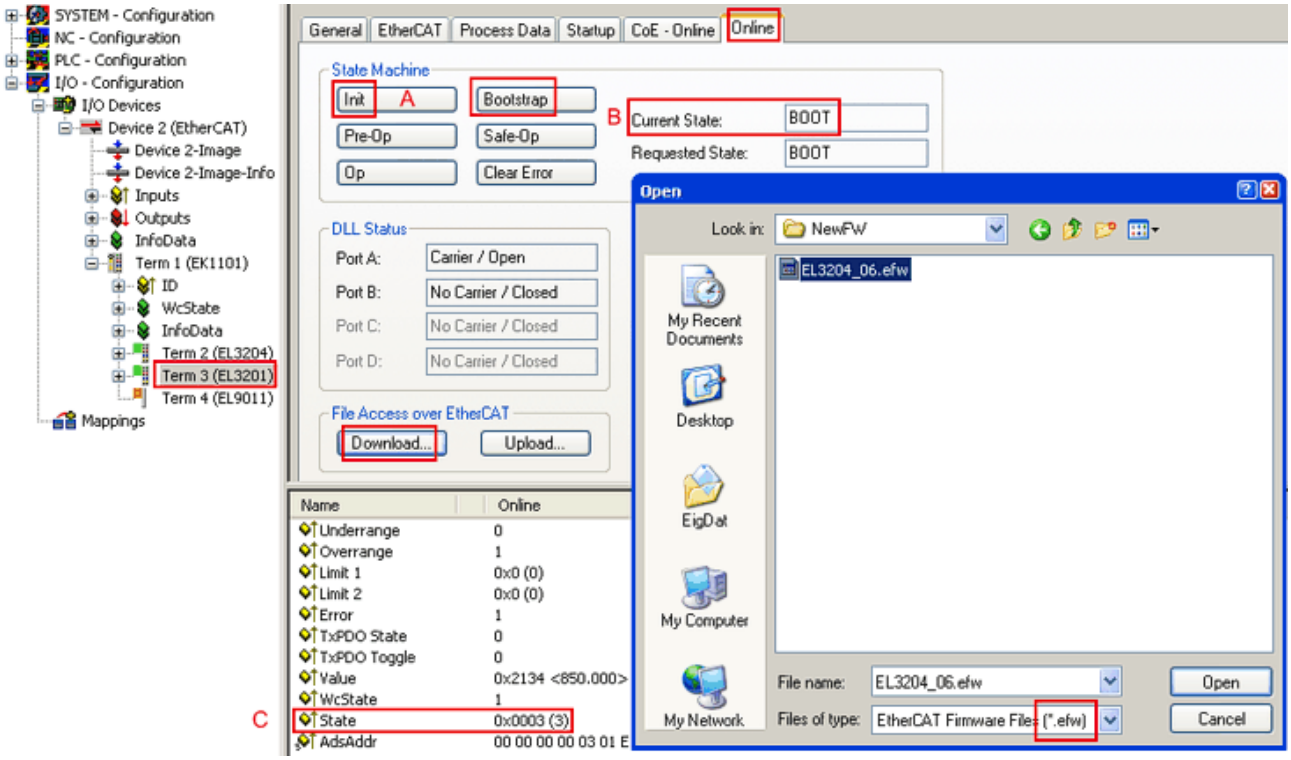
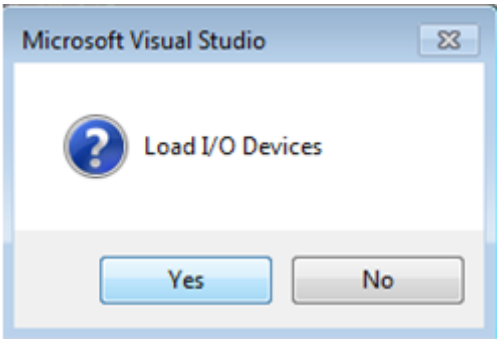
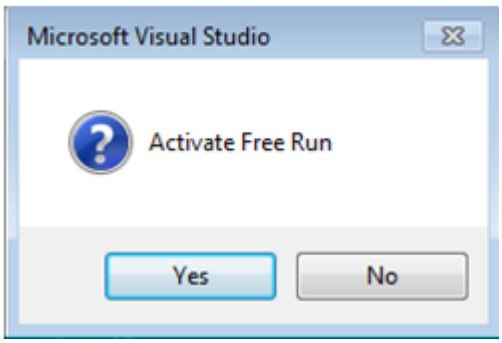


Fig. 194: Firmware Update

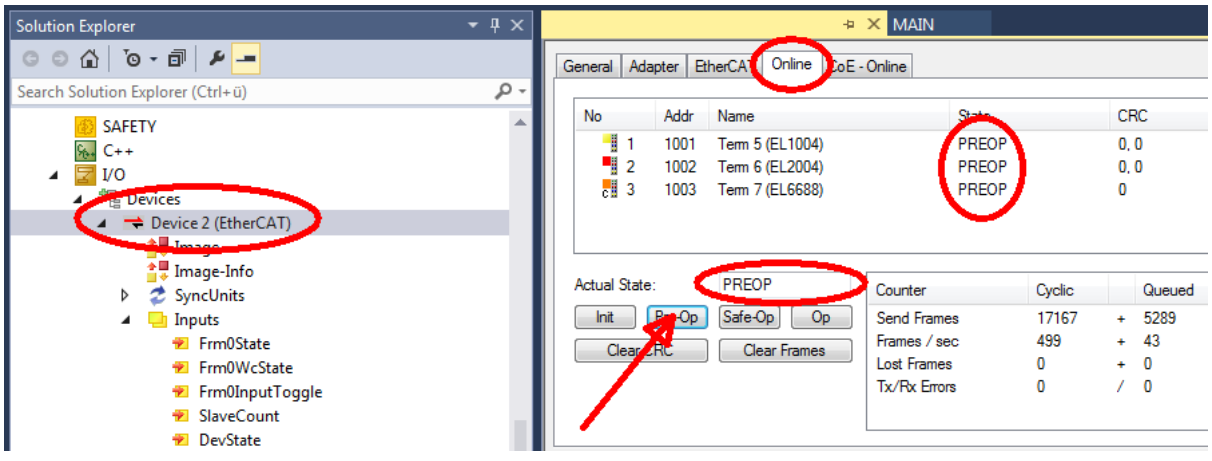
Proceed as follows, unless instructed otherwise by Beckhoff support. Valid for TwinCAT 2 and 3 as EtherCAT master.

- Switch TwinCAT system to ConfigMode/FreeRun with cycle time >= 1 ms (default in ConfigMode is 4 ms). A FW-Update during real time operation is not recommended.



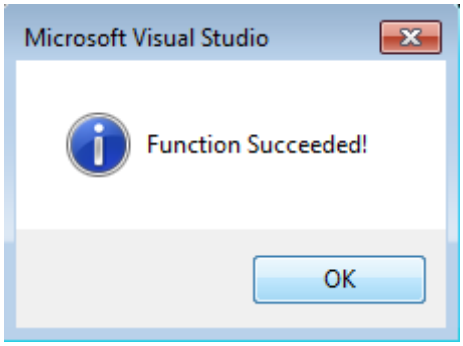


- Switch EtherCAT Master to PreOP



- Switch slave to INIT (A)
- Switch slave to BOOTSTRAP

- Check the current status (B, C)
- Download the new *efw file (wait until it ends). A password will not be necessary usually.



- After the download switch to INIT, then PreOP
- Switch off the slave briefly (don't pull under voltage!)
- Check within CoE 0x100A, if the FW status was correctly overtaken.

6.3.4 FPGA firmware *.rbf

If an FPGA chip deals with the EtherCAT communication an update may be accomplished via an *.rbf file.

- Controller firmware for processing I/O signals
- FPGA firmware for EtherCAT communication (only for terminals with FPGA)

The firmware version number included in the terminal serial number contains both firmware components. If one of these firmware components is modified this version number is updated.

Determining the version via the TwinCAT System Manager

The TwinCAT System Manager indicates the FPGA firmware version. Click on the Ethernet card of your EtherCAT strand (Device 2 in the example) and select the *Online* tab.

The *Reg:0002* column indicates the firmware version of the individual EtherCAT devices in hexadecimal and decimal representation.

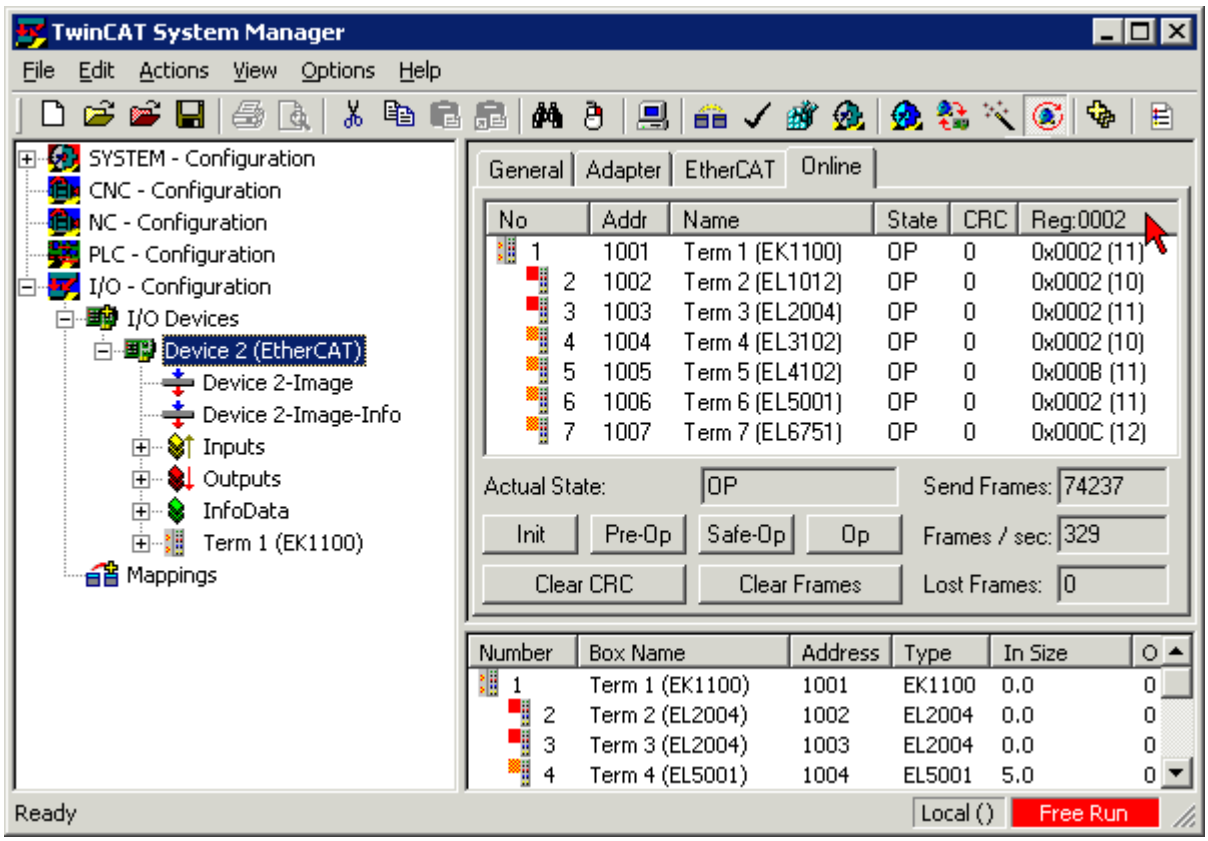


Fig. 195: FPGA firmware version definition

If the column *Reg:0002* is not displayed, right-click the table header and select *Properties* in the context menu.

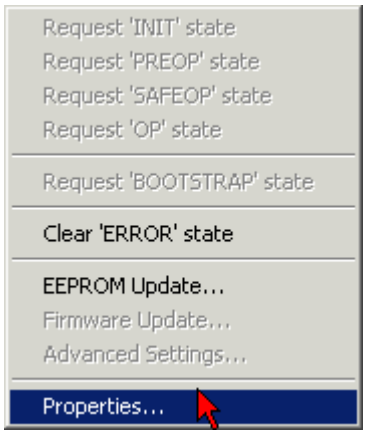


Fig. 196: Context menu *Properties*

The *Advanced Settings* dialog appears where the columns to be displayed can be selected. Under *Diagnosis/***Online View** select the *'0002 ETxxxx Build'* check box in order to activate the FPGA firmware version display.

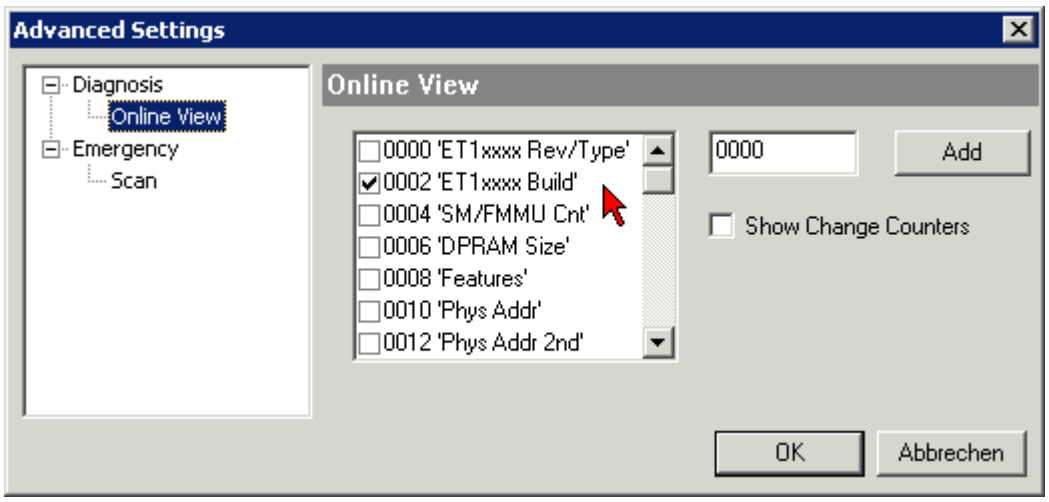


Fig. 197: Dialog *Advanced Settings*

Update

For updating the FPGA firmware

- of an EtherCAT coupler the coupler must have FPGA firmware version 11 or higher;
- of an E-Bus Terminal the terminal must have FPGA firmware version 10 or higher.

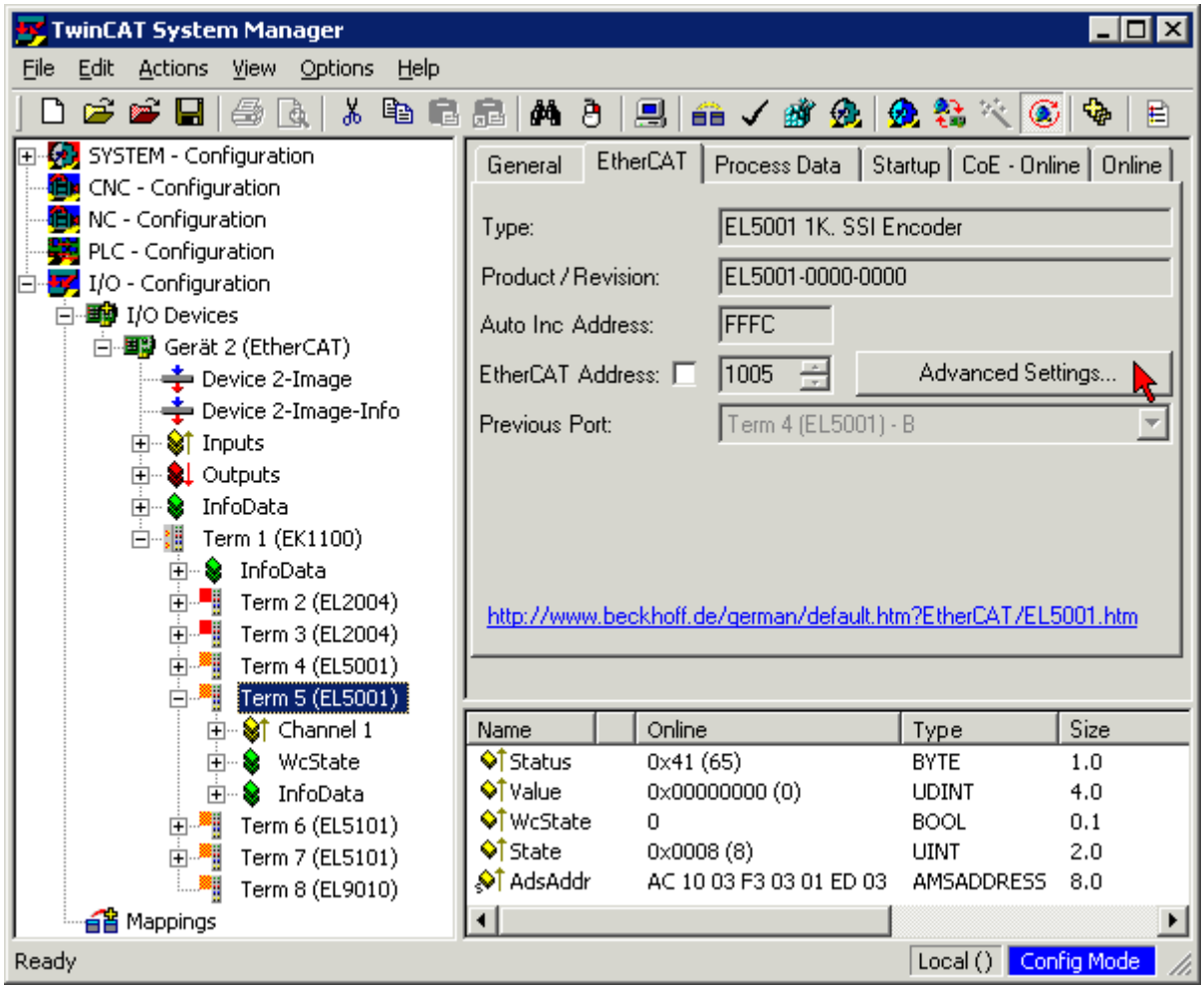
Older firmware versions can only be updated by the manufacturer!

Updating an EtherCAT device

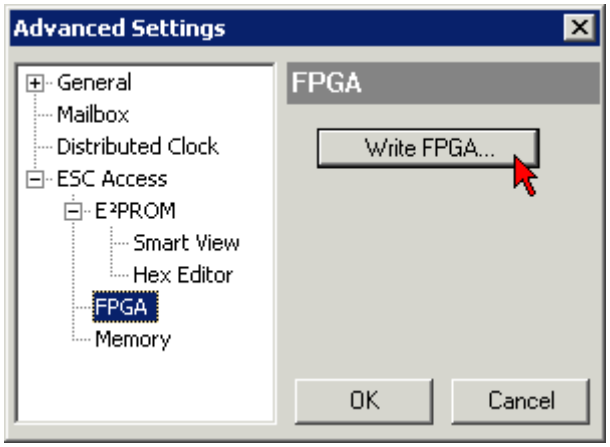
The following sequence order have to be met if no other specifications are given (e.g. by the Beckhoff support):

- Switch TwinCAT system to ConfigMode/FreeRun with cycle time >= 1 ms (default in ConfigMode is 4 ms). A FW-Update during real time operation is not recommended.

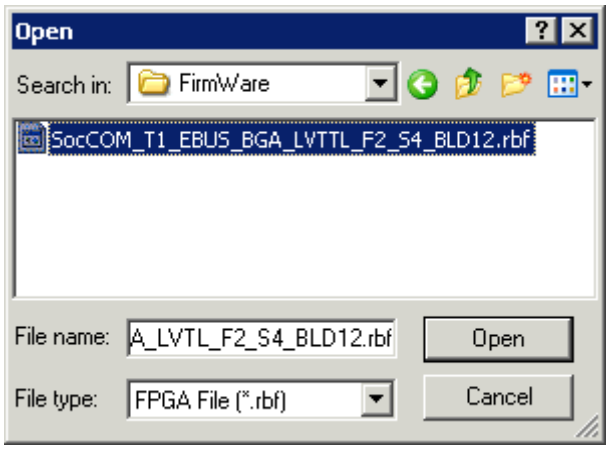
- In the TwinCAT System Manager select the terminal for which the FPGA firmware is to be updated (in the example: Terminal 5: EL5001) and
  click the *Advanced Settings* button in the *EtherCAT* tab:



- The *Advanced Settings* dialog appears. Under *ESC Access/E²PROM*/FPGA click on *Write FPGA* button:



- Select the file (*.rbf) with the new FPGA firmware, and transfer it to the EtherCAT device:



- Wait until download ends
- Switch slave current less for a short time (don't pull under voltage!). In order to activate the new FPGA firmware a restart (switching the power supply off and on again) of the EtherCAT device is required.
- Check the new FPGA status

NOTICE

Risk of damage to the device!

A download of firmware to an EtherCAT device must not be interrupted in any case! If you interrupt this process by switching off power supply or disconnecting the Ethernet link, the EtherCAT device can only be recommissioned by the manufacturer!

6.3.5 Simultaneous updating of several EtherCAT devices

The firmware and ESI descriptions of several devices can be updated simultaneously, provided the devices have the same firmware file/ESI.

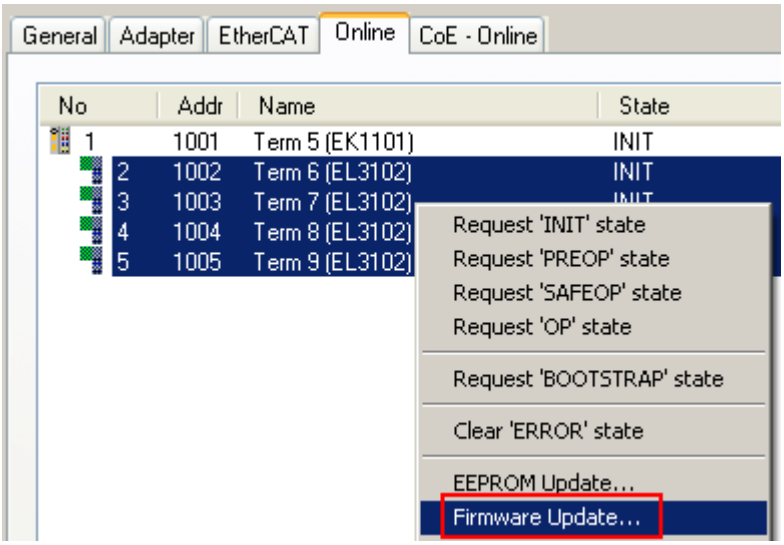


Fig. 198: Multiple selection and firmware update

Select the required slaves and carry out the firmware update in BOOTSTRAP mode as described above.

6.4 Restoring the delivery state

To restore the delivery state (factory settings) of CoE objects (object directory) for EtherCAT devices ("slaves"), the CoE object *Restore default parameters*, SubIndex 001 can be used via EtherCAT master (e.g. TwinCAT) (see Fig. *Selecting the Restore default parameters PDO*).

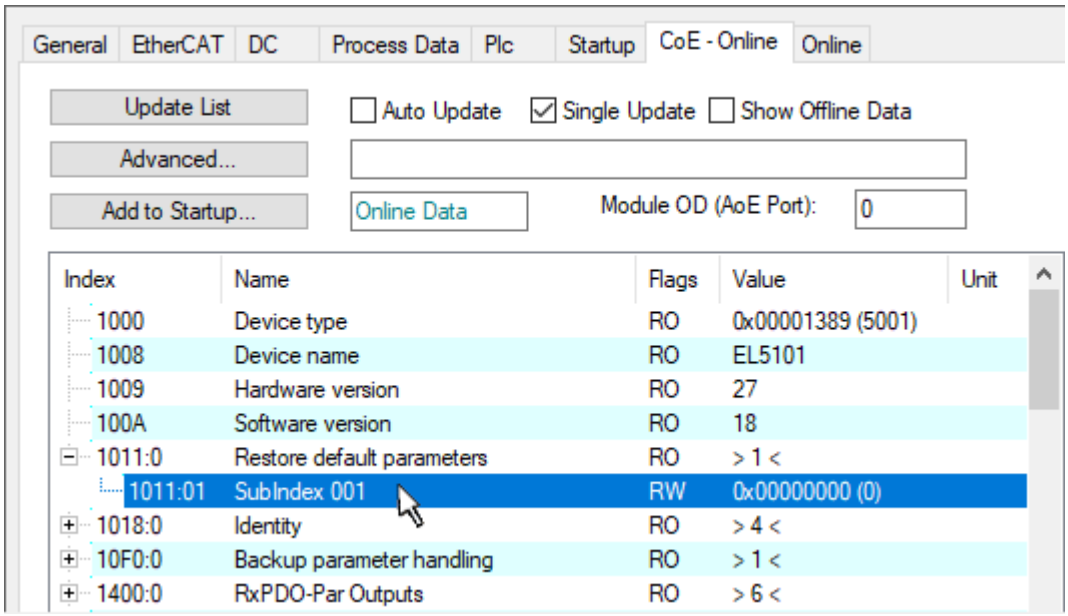


Fig. 199: Selecting the *Restore default parameters* PDO

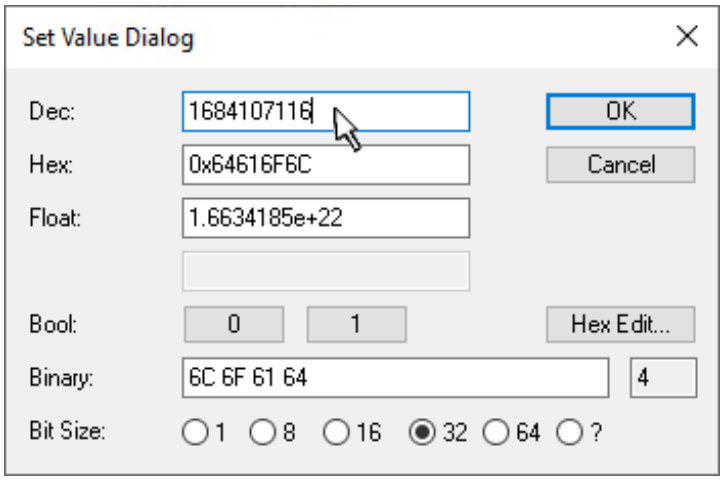


Fig. 200: Entering a restore value in the Set Value dialog

Double-click on *SubIndex 001* to enter the Set Value dialog. Enter the reset value **1684107116** in field *Dec* or the value **0x64616F6C** in field *Hex* (ASCII: "load") and confirm with *OK* (Fig. *Entering a restore value in the Set Value dialog*).

- All changeable entries in the slave are reset to the default values.
  Exception: objects write-protected via password
- Depending on the size of the object directory, this process can take from a few ms to > 1 second.
- The values can only be successfully restored if the reset is directly applied to the online CoE, i.e. to the slave. No values can be changed in the offline CoE.
- TwinCAT must be in the RUN or CONFIG/Freerun state for this; that means EtherCAT data exchange takes place. Ensure error-free EtherCAT transmission.
- No separate confirmation takes place due to the reset. A changeable object can be manipulated beforehand for the purposes of checking.
- This reset procedure can also be adopted as the first entry in the startup list of the slave, e.g. in the state transition PREOP->SAFEOP or, as in Fig. *CoE reset as a startup entry*, in SAFEOP->OP.

All backup objects are reset to the delivery state.

Alternative restore value

In some older terminals (FW creation approx. before 2007) the backup objects can be switched with an alternative restore value: Decimal value: 1819238756, Hexadecimal value: 0x6C6F6164.

An incorrect entry for the restore value has no effect.

6.5 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on her internet pages: www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Support

The Beckhoff Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline:	+49 5246 963 157
e-mail:	support@beckhoff.com
web:	www.beckhoff.com/support

Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline:	+49 5246 963 460
e-mail:	service@beckhoff.com
web:	www.beckhoff.com/service

Headquarters Germany

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Germany

Phone:	+49 5246 963 0
e-mail:	info@beckhoff.com
web:	www.beckhoff.com

Trademark statements

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar® and XTS® are registered and licensed trademarks of Beckhoff Automation GmbH.

Third-party trademark statements

CANopen and CANopen FD are registered trademarks of CAN in AUTOMATION - International Users and Manufacturers Group e.V.

DeviceNet and EtherNet/IP are trademarks of ODVA, Inc.

DSP System Toolbox, Embedded Coder, MATLAB, MATLAB Coder, MATLAB Compiler, MathWorks, Predictive Maintenance Toolbox, Simscape, Simscape™ Multibody™, Simulink, Simulink Coder, Stateflow and ThingSpeak are registered trademarks of The MathWorks, Inc.

Excel, IntelliSense, Microsoft, Microsoft Azure, Microsoft Edge, PowerShell, Visual Studio, Windows and Xbox are trademarks of the Microsoft group of companies.

More Information:
www.beckhoff.com/EL4xxx

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com