BECKHOFF New Automation Technology

Documentation | EN

EL5032, EL5032-0090

2 channel EnDat2.2-Interface



2025-07-02 | Version: 3.1.0

Table of contents

1 Foreword 7
1.1 Two-channel ENDAT 2.2 interface - product overview 7
1.2 Notes on the documentation 7
1.3 Safety instructions 8
1.4 Guide through documentation 9
1.5 Documentation issue status 10
1.6 Version identification of EtherCAT devices 11
1.6.1 General notes on marking 11
1.6.2 Version identification of EL terminals 12
1.6.3 Beckhoff Identification Code (BIC) 12
1.6.4 Electronic access to the BIC (eBIC) 14
2 EL5032 - Product description 16
2.1 Introduction 16
2.2 Technical data 17
2.3 Start 18
3 EL5032-0090 - Product description 19
3.1 Introduction 19
3.2 Technical data 20
3.3 Start 21
4 EL503x series overview 22
5 Basics communication 23
5.1 EtherCAT basics 23
5.2 EtherCAT cabling – wire-bound 23
5.3 General notes for setting the watchdog 25
5.4 EtherCAT State Machine 26
5.5 CoE Interface 28
5.6 Distributed Clock 33
6 Basics of EnDat 2.2 technology 34
6.1 EnDat 2.2 - Bidirectional interface for position measuring devices 34
6.2 Connection technology 36
6.3 Compatibility 37
6.4 Functional description 38
6.5 Data transfer 39
7 Mounting and wiring 40
7.1 Instructions for ESD protection 40
7.2 Installation on mounting rails 41
7.3 Installation instructions for enhanced mechanical load capacity 44
7.4 Connection 44
7.4.1 Connection system 44
7.4.2 Wiring 47
7.4.3 Shielding 48
7.5 Note - power supply 48
7.6 Installation positions 49

7.7 Positioning of passive Terminals ... 51
7.8 UL notice ... 52
7.9 LEDs and connection ... 53
7.10 Disposal ... 55

8 Commissioning ... 56
8.1 TwinCAT Quick Start ... 56
8.1.1 TwinCAT 2 ... 59
8.1.2 TwinCAT 3 ... 69
8.2 TwinCAT Development Environment ... 82
8.2.1 Installation of the TwinCAT real-time driver ... 83
8.2.2 Notes regarding ESI device description ... 89
8.2.3 TwinCAT ESI Updater ... 93
8.2.4 Distinction between Online and Offline ... 93
8.2.5 OFFLINE configuration creation ... 94
8.2.6 ONLINE configuration creation ... 99
8.2.7 EtherCAT subscriber configuration ... 107
8.2.8 Import/Export of EtherCAT devices with SCI and XTI ... 117
8.3 General Commissioning Instructions for an EtherCAT Slave ... 123

9 EL5032-00x0 - Commissioning ... 131
9.1 Quick start ... 131
9.2 Process data ... 132
9.2.1 Sync Manager (SM) ... 132
9.2.2 PDO Assignment ... 132
9.2.3 Predefined PDO Assignment ... 134
9.3 Velocity measurement ... 135
9.4 Filter operation ... 136
9.5 Overview of commands and samples ... 137
9.5.1 Command 0x1000 - Delete Diag History ... 138
9.5.2 Command 0x1001 - Diag History Test ... 138
9.5.3 Command 0x8001 - EnDat Reset ... 139
9.5.4 Command 0x8006 - EnDat read Pos2 ... 139
9.5.5 Command 0x8008 - Suppress TxPDO State (EL5032) ... 140
9.5.6 Command 0x9000 - Read EnDat Word ... 141
9.5.7 Command 0x9001 - Write EnDat Word ... 142
9.6 EL5032-00x0 - Object description and parameterization ... 143
9.6.1 Restore object ... 143
9.6.2 Configuration data ... 143
9.6.3 Command object ... 145
9.6.4 Input data ... 145
9.6.5 Information data ... 146
9.6.6 Diagnostic data ... 147
9.6.7 Standard objects ... 149
9.7 EL5032-0090 - TwinSAFE SC ... 157
9.7.1 TwinSAFE SC ... 157
9.7.2 EL5032-0090 - TwinSAFE SC process data ... 162

9.7.3 EL5032-0090 - TwinSAFE Single Channel objects ... 163

10 Diagnostics ... 165

10.1 Diagnostics - basic principles of diag messages ... 165

11 Appendix ... 175

11.1 Firmware compatibility ... 175

11.2 Firmware Update EL/ES/EM/ELM/EP/EPP/ERPxxxx ... 175

11.2.1 Device description ESI file/XML ... 177

11.2.2 Firmware explanation ... 180

11.2.3 Updating controller firmware *.efw ... 180

11.2.4 FPGA firmware *.rbf ... 182

11.2.5 Simultaneous updating of several EtherCAT devices ... 186

11.3 Restoring the delivery state ... 187

11.4 Support and Service ... 189

1 Foreword

1.1 Two-channel ENDAT 2.2 interface - product overview

EL5032 [▶ 16]	2-channel ENDAT 2.2 interface
EL5032-0090 [▶ 19]	2-channel ENDAT 2.2 interface (TwinSAFE Single Channel)

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 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.

1.4 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
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

1.5 Documentation issue status

Version	Comment
3.1.0	• Update chapter “Introduction” and “LEDs and Connection”
3.0.0	• Chapter “EL503x series overview” added • Update structure
2.9	• Update Technical data • Chapter "Note on power supply" added • Update chapter "Process data" • Chapter "Filter operation" and "Velocity measurement" added • Update chapter "Overview of the commands and samples" • Update chapter "EL5032-00x0 – Object description and parameterization" • Update chapter "EL5032-0090 – TwinSAFE SC" • Update revision status • Structural update
2.8	• Update chapter "Product description • • Chapter "Disposal" added • • Chapter "TwinCAT Quick-Start" added • • Structural update
2.7	• Update chapter "Basics of EnDat 2.2 technology" • Update chapter "Object description and parameterization" • Structural update
2.6	• Update chapter "Technical data" • Update chapter "LEDs and connection" • Structural update
2.5	• Update chapter "Technical data" • Update revision status • Structural update
2.4	• Update chapter "Basics of EnDat 2.2 technology" • Structural update
2.3	• EL5032-0090 added • Structural update • Update revision status
2.2	• Update chapter "Basics of EnDat 2.2 technology" • Update revision status • Structural update
2.1	• Update chapter "Notes on the documentation" • Update chapter "Technical data" • Note on ESD protection added • Chapter "Installation instructions for enhanced mechanical load capacity" added • Chapter "ATEX - special conditions" replaced by chapter "ATEX - special conditions (standard temperature range)" • Update chapter "TwinCAT 2.1x" -> "TwinCAT Development Environment" • Update chapter "Diagnosis" • Update revision status
2.0	• Migration
1.2	• Structural update • Update chapter "Technical data" • Revision status updated
1.0.1	• Complements, corrections
1.0	• Complements, corrections, 1st public issue
0.3	• Complements, corrections
0.2	• Complements, corrections
0.1	• Provisional documentation for EL5032

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.

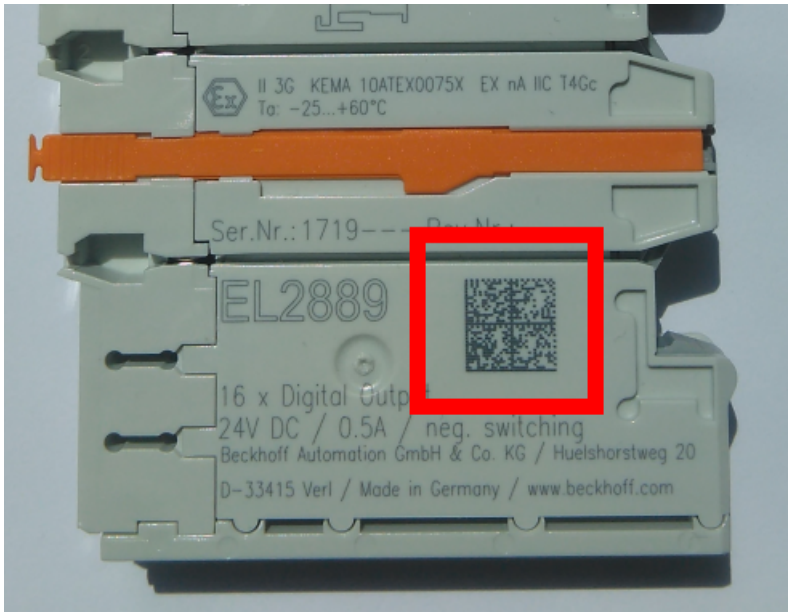


Fig. 2: BIC as data matrix code (DMC, code scheme ECC200)

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.

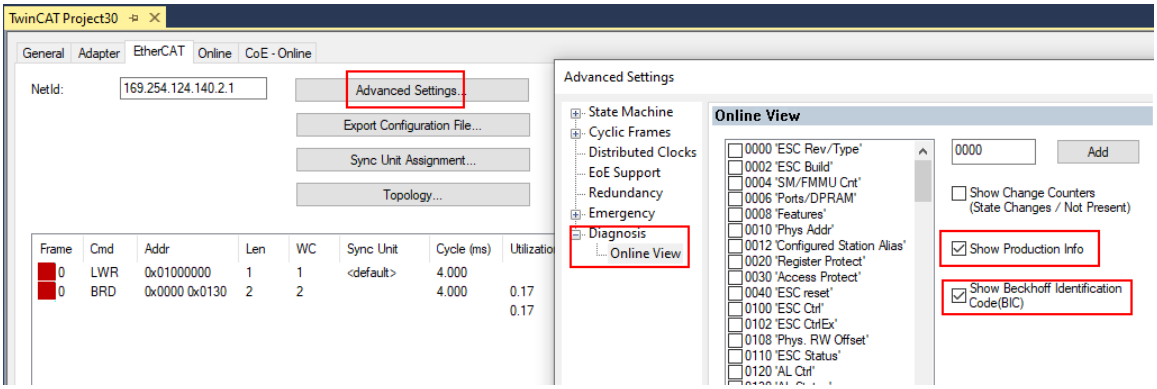
EtherCAT devices (IP20, IP67)

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.

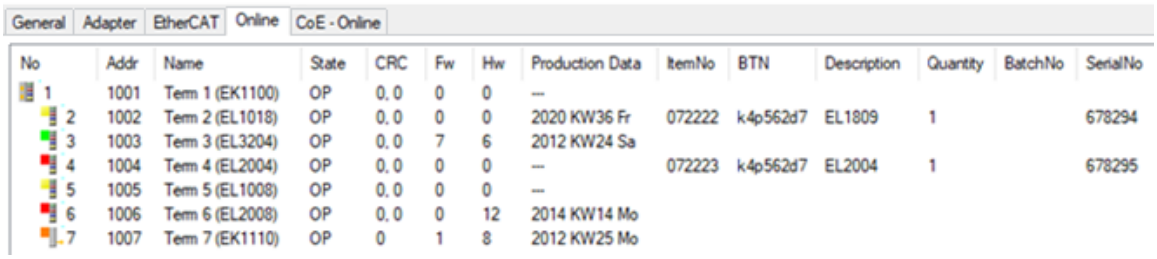
Beckhoff also stores the eBIC in the ESI-EEPROM. The eBIC was introduced into Beckhoff IO production (terminals, box modules) in 2020; as of 2023, implementation is largely complete.

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:

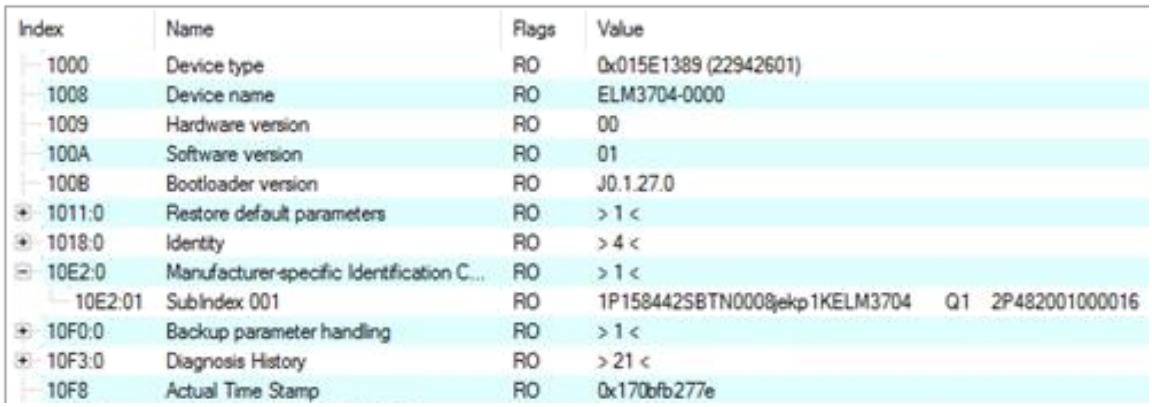


  - The BTN and its contents are then displayed:



  - 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 EL5032 - Product description

2.1 Introduction

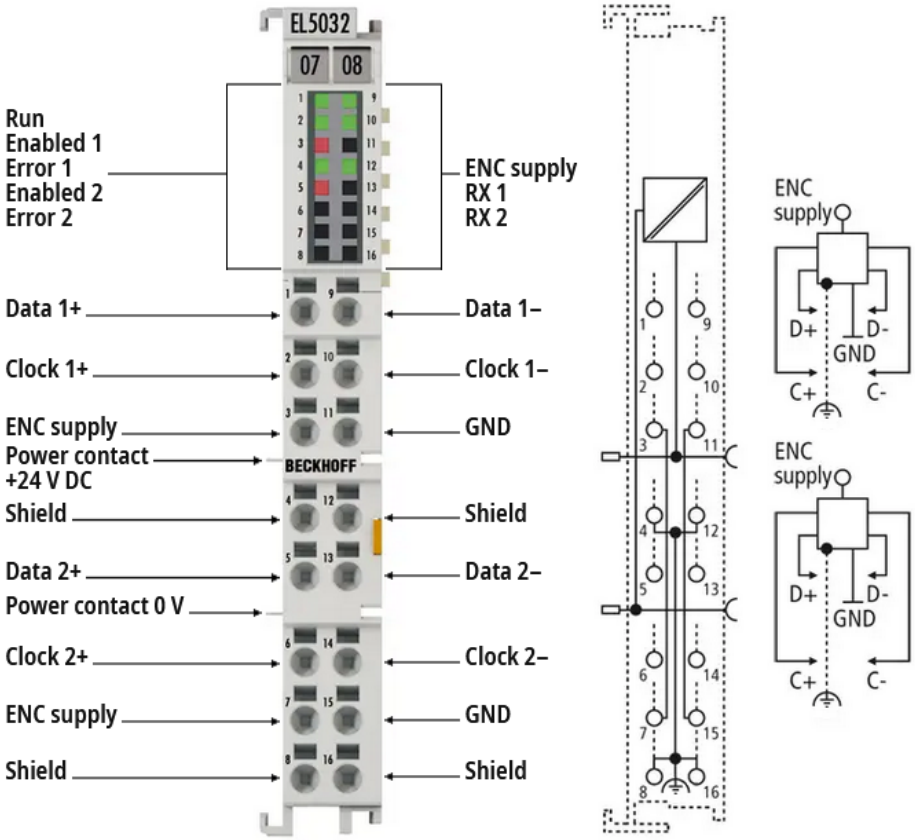


Fig. 4: EL5032

Two-channel EnDat2.2 interface

The EL5032 EnDat 2.2 EtherCAT Terminal is used for direct connection of two encoders with EnDat2.2 interface.
The EL5032 enables reading of position values, diagnosis encoder data, internal and external temperature values and the electronic type plate. With the electronic type plate all measuring device-specific information is directly available. In addition, the EL5032 provides an encoder supply of 5 V or optionally 9 V.

In addition, user-defined data can be stored in the encoder. This enables cost-effective and quicker commissioning. The position value is output with up to 48 bits, depending on the resolution of the connected measuring device. In addition to the position value, further information such as status information, addresses and data can be transferred.
A list of additional information supported by the encoder is stored in the parameters.

The EL5032 features distributed clocks, which means that the position value can be read in exact synchrony with the system. If the distributed clock function is deactivated, the EL5032 cycles synchronous with the EtherCAT cycle.

Quick links

- Basic Function Principles [▶ 34]
- Quick start [▶ 131]
- EL5032-00x0 - object description and parameterization [▶ 143]

2.2 Technical data

EnDat-2.2 interface	EL5032
Technology	EnDat 2.2 interface (order designation EnDat22)
Number of channels	2
Encoder connection, encoder clock frequency	D+, D-, C+, C-; 8 MHz
Commands	• Read position values incl. additional information selectable via MRS code (Memory Range Select), • Read and write parameters, • Reset functions
Resolution	max. 48 bit for position value

Function and communication	EL5032
Distributed Clocks	• yes
Special features	• Storage of the zero offset shift • Electronic nameplate • Diagnosis • Warning, incl. cable length compensation up to 100 m • Reading the encoder temperature values • Integrated velocity calculation (from firmware 16, revision 28)
Configuration	via EtherCAT master/CoE

Supply and potentials	EL5032
Power supply	24 V via power contacts
Nominal voltage	24 V at power contact, encoder supply included in the installation, sum current max. 0.5 A for both channels
Current consumption power contacts	typ. 150 mA
Encoder operating voltage/ Encoder supply	optionally 5 V_{DC} or 9 V_{DC} (generated from the 24 V_{DC} power contacts)
Electrical isolation	500 V (E-bus/field voltage)
Current consumption via E-bus	120 mA typ.

Environmental conditions	EL5032
Permissible ambient temperature range during operation	0°C... +55°C
Permissible ambient temperature range during storage	-25°C... +85°C
Permissible relative air humidity	95%, no condensation

General data	EL5032
Weight	app. 50 g
Dimensions (W x H x D)	approx. 15 mm x 100 mm x 70 mm (width aligned: 12 mm)
Installation	on 35 mm mounting rail, conforms to EN 60715
Installation position	variable

Standards and approvals	EL5032
Enhanced mechanical load capacity	yes, see also Installation instructions for enhanced mechanical load capacity [▶ 44]
Vibration / shock resistance	conforms to EN 60068-2-6 / EN 60068-2-27
EMC immunity / emission	conforms to EN 61000-6-2 / EN 61000-6-4
Protection rating	IP20
Approvals/markings	CE, EAC, UKCA cULus [▶ 52]

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

2.3 Start

For commissioning:

- mount the EL5032 as explained in the chapter Mounting and wiring [▶ 40]
- configure the EL5032 in TwinCAT as described in the chapters Commissioning [▶ 131] and EL5032-00x0 - Commissioning [▶ 131].

For fast commissioning please refer to chapter Commissioning -> Quick start [▶ 131].

3 EL5032-0090 - Product description

3.1 Introduction

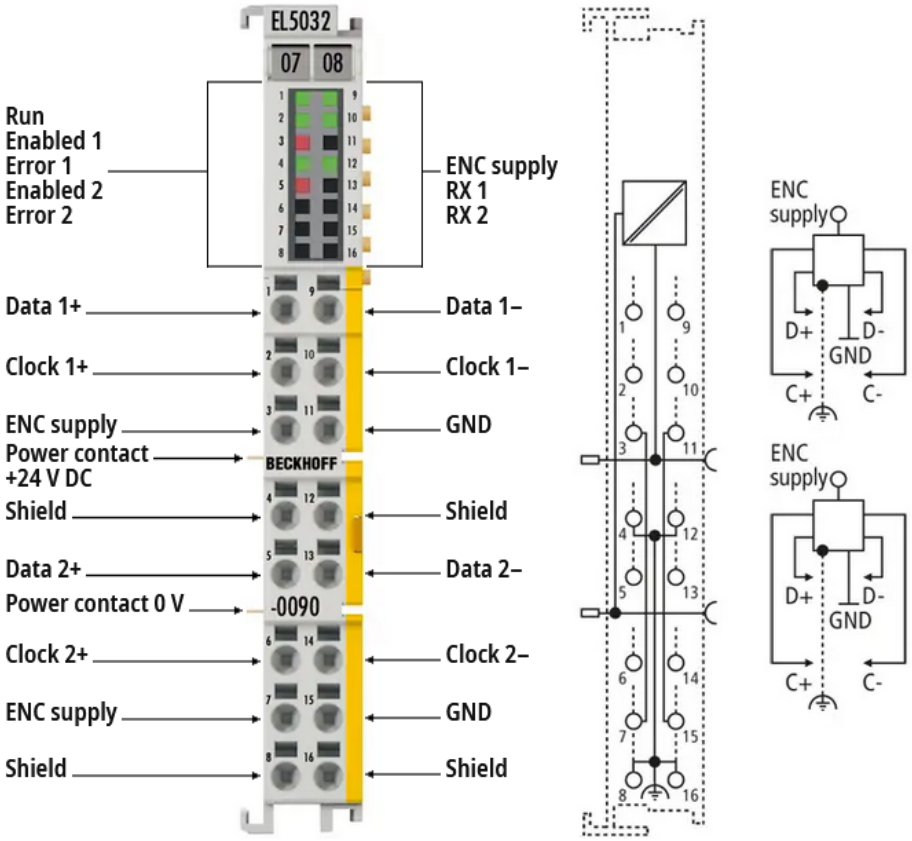


Fig. 5: EL5032-0090

Two-channel EnDat2.2 interface, TwinSAFE SC

The EL5032-0090 EnDat 2.2 EtherCAT Terminal is used for direct connection of two encoders with EnDat2.2 interface.
The EL5032-0090 enables reading of position values, diagnosis encoder data, internal and external temperature values and the electronic type plate. With the electronic type plate all measuring device-specific information is directly available. In addition, the EL5032-0090 provides an encoder supply of 5 V or optionally 9 V.

In addition, user-defined data can be stored in the encoder. This enables cost-effective and quicker commissioning. The position value is output with up to 48 bits, depending on the resolution of the connected measuring device. In addition to the position value, further information such as status information, addresses and data can be transferred.
A list of additional information supported by the encoder is stored in the parameters.

The EL5032-0090 features distributed clocks, which means that the position value can be read in exact synchrony with the system. If the distributed clock function is deactivated, the EL5032-0090 cycles synchronous with the EtherCAT cycle.

In addition to the full functionality of the EL5032, the EL5032-0090 supports TwinSAFE SC (Single Channel) technology. This enables the use of standard signals for safety tasks in any networks of fieldbuses.

Quick links

- Basic Function Principles [▶ 34]
- Quick start [▶ 131]
- EL5032-00x0 - object description and parameterization [▶ 143]
- EL5032-0090 - TwinSAFE Single Channel objects [▶ 163]
- EL5032-0090 - TwinSAFE process data [▶ 162]

3.2 Technical data

EnDat-2.2 interface	EL5032-0090
Technology	EnDat 2.2 interface (order designation EnDat22)
Number of channels	2
Encoder connection, encoder clock frequency	D+, D-, C+, C-; 8 MHz
Commands	• Read position values incl. additional information selectable via MRS code (Memory Range Select), • Read and write parameters, • Reset functions
Resolution	max. 48 bit for position value

Function and communication	EL5032-0090
Distributed Clocks	yes
Special features	• Storage of the zero offset shift • Electronic nameplate • Diagnosis, Warning, incl. cable length compensation up to 100 m • Reading the encoder temperature values • Integrated velocity calculation (from firmware 16, revision 17) • TwinSAFE SC
Configuration	via EtherCAT master/CoE

Supply and potentials	EL5032-0090
Power supply	24 V via power contacts
Nominal voltage	24 V at power contact, encoder supply included in the installation, sum current max. 0.5 A for both channels
Current consumption power contacts	typ. 150 mA
Encoder operating voltage/ Encoder supply	optionally 5 V_{DC} or 9 V_{DC} (generated from the 24 V_{DC} power contacts)
Electrical isolation	500 V (E-bus/field voltage)
Current consumption via E-bus	120 mA typ.

Environmental conditions	EL5032-0090
Permissible ambient temperature range during operation	0°C... +55°C
Permissible ambient temperature range during storage	-25°C... +85°C
Permissible relative air humidity	95%, no condensation

General data	EL5032-0090
MTBF (+55 °C)	> 790,000 h
Weight	app. 50 g
Dimensions (W x H x D)	approx. 15 mm x 100 mm x 70 mm (width aligned: 12 mm)
Installation	on 35 mm mounting rail, conforms to EN 60715
Installation position	variable

Standards and approvals	EL5032-0090
Enhanced mechanical load capacity	yes, see also Installation instructions for enhanced mechanical load capacity [▶ 44]
Vibration / shock resistance	conforms to EN 60068-2-6 / EN 60068-2-27
EMC immunity / emission	conforms to EN 61000-6-2 / EN 61000-6-4
Protection rating	IP20
Approvals/markings*	CE, EAC, UKCA cULus [▶ 52]

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

3.3 Start

For commissioning:

- mount the EL5032 as explained in the chapter Mounting and wiring [▶ 40]
- configure the EL5032 in TwinCAT as described in the chapters Commissioning [▶ 131] and EL5032-00x0 - Commissioning [▶ 131].

For fast commissioning please refer to chapter Commissioning -> Quick start [▶ 131].

4 EL503x series overview

Technical data	EL5031-0011	EL5032	EL5032-0090
Technology	EnDat 2.2 interface		
Number of channels	1	2	2
Power supply (encoder)	5 V_{DC}, 9 V_{DC}		
Commands	• Read position values incl. additional information selectable via MRS code (Memory Range Select) • Read and write parameters • Reset functions		
Functions	• Storage of the zero offset shift • Electronic nameplate • Diagnosis • Warning, incl. cable length compensation up to 100 m • Reading the encoder temperature values		
MTBF (+55 °C)	-	-	> 790,000 h
Special features	• Oversampling	• Integrated velocity calculation	• Integrated velocity calculation • TwinSAFE SC

5 Basics communication

5.1 EtherCAT basics

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

5.2 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

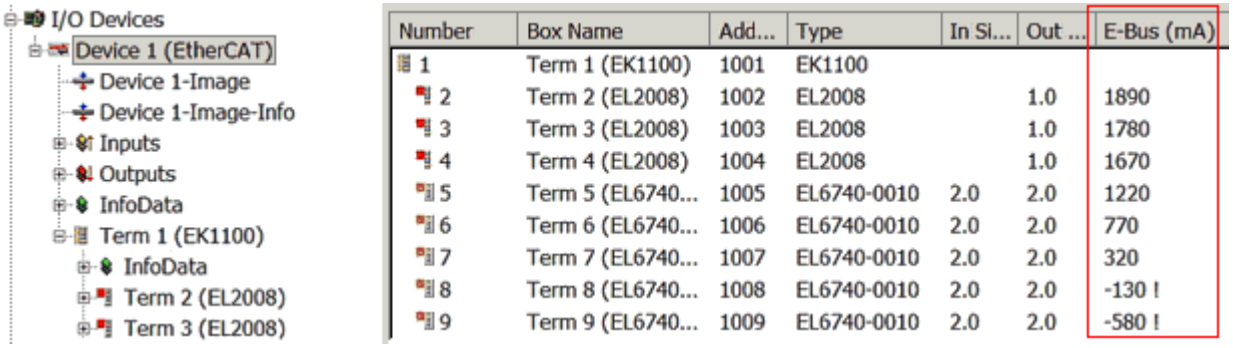
It is recommended to use the appropriate Beckhoff components e.g.
- cable sets ZK1090-9191-xxxx respectively
- RJ45 connector, field assembly ZS1090-0005
- EtherCAT cable, field assembly ZB9010, ZB9020

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.



Number	Box Name	Add...	Type	In Si...	Out ...	E-Bus (mA)
1	Term 1 (EK1100)	1001	EK1100			
2	Term 2 (EL2008)	1002	EL2008		1.0	1890
3	Term 3 (EL2008)	1003	EL2008		1.0	1780
4	Term 4 (EL2008)	1004	EL2008		1.0	1670
5	Term 5 (EL6740...	1005	EL6740-0010	2.0	2.0	1220
6	Term 6 (EL6740...	1006	EL6740-0010	2.0	2.0	770
7	Term 7 (EL6740...	1007	EL6740-0010	2.0	2.0	320
8	Term 8 (EL6740...	1008	EL6740-0010	2.0	2.0	-130 !
9	Term 9 (EL6740...	1009	EL6740-0010	2.0	2.0	-580 !

Fig. 6: 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!

5.3 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:

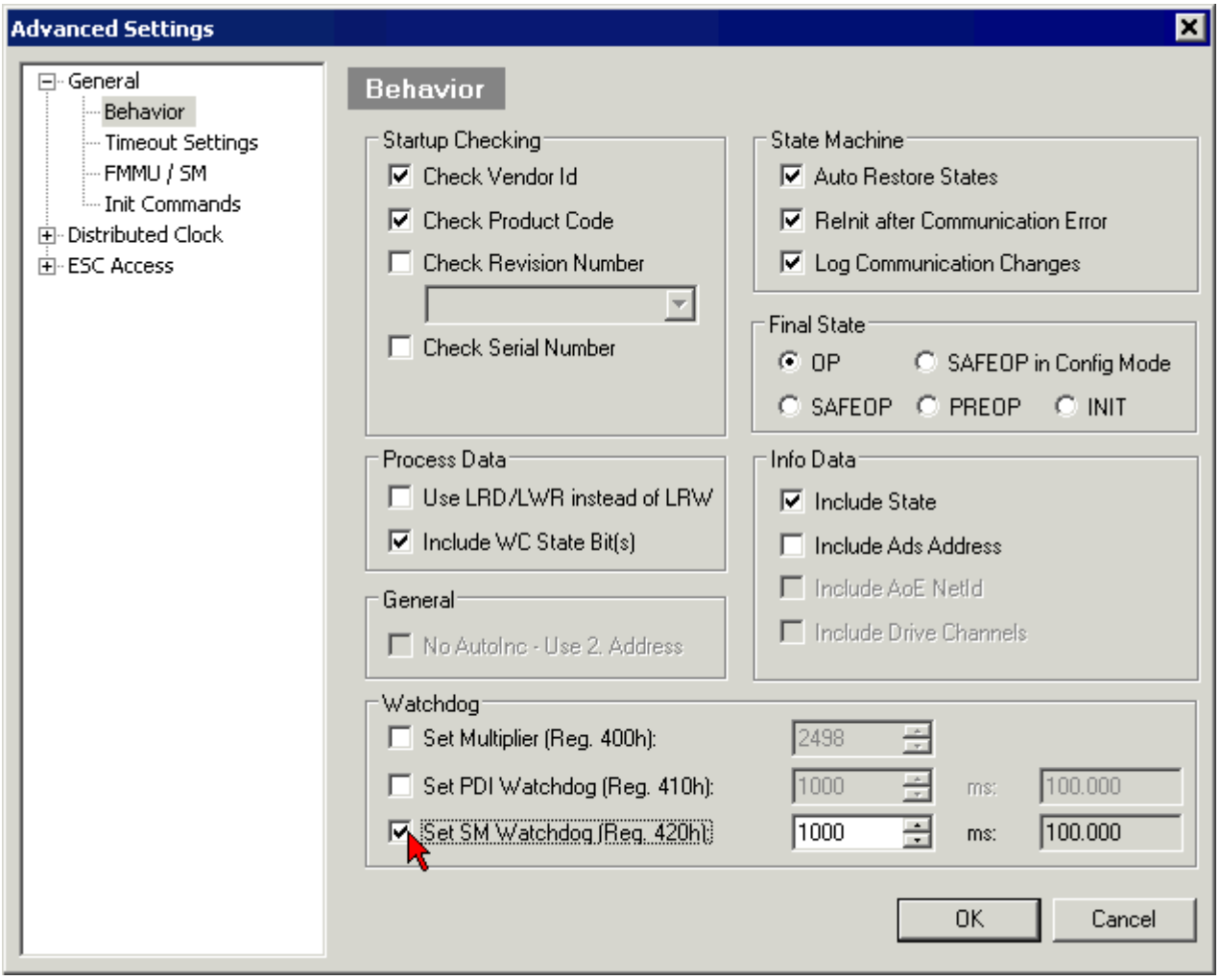


Fig. 7: eEtherCAT tab -> Advanced Settings -> Behavior -> Watchdog

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.

5.4 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.

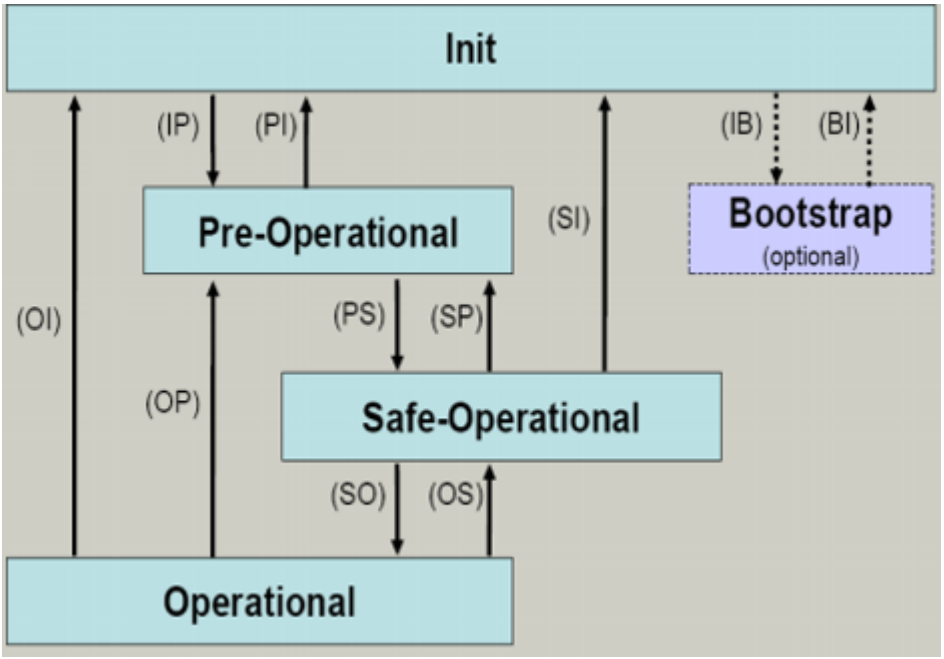


Fig. 8: States of the EtherCAT State Machine

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.

5.5 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:

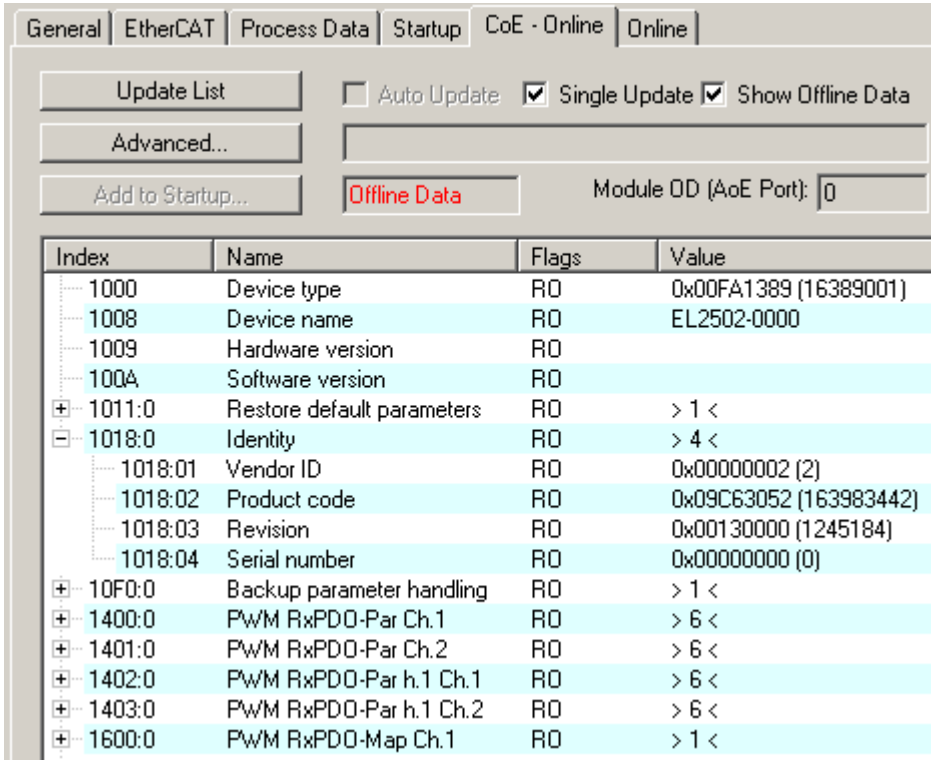


Fig. 9: “CoE Online” tab

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.

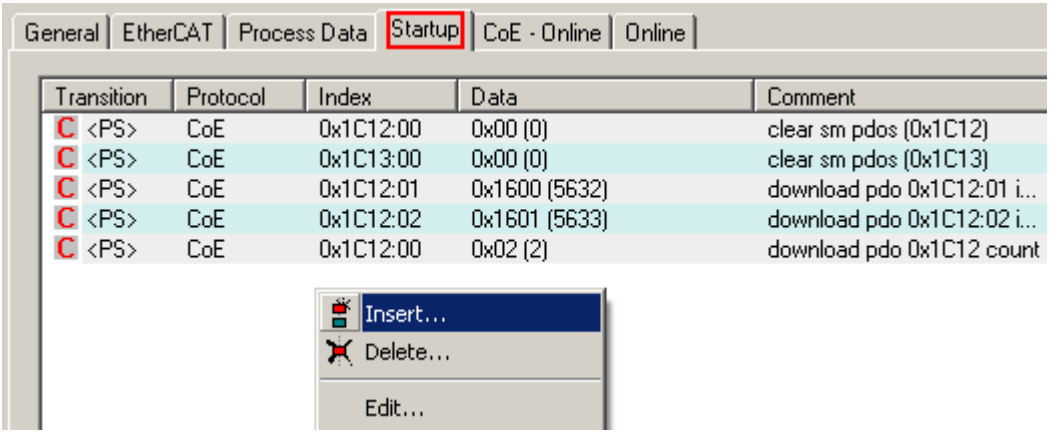


Fig. 10: Startup list in the TwinCAT System Manager

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.

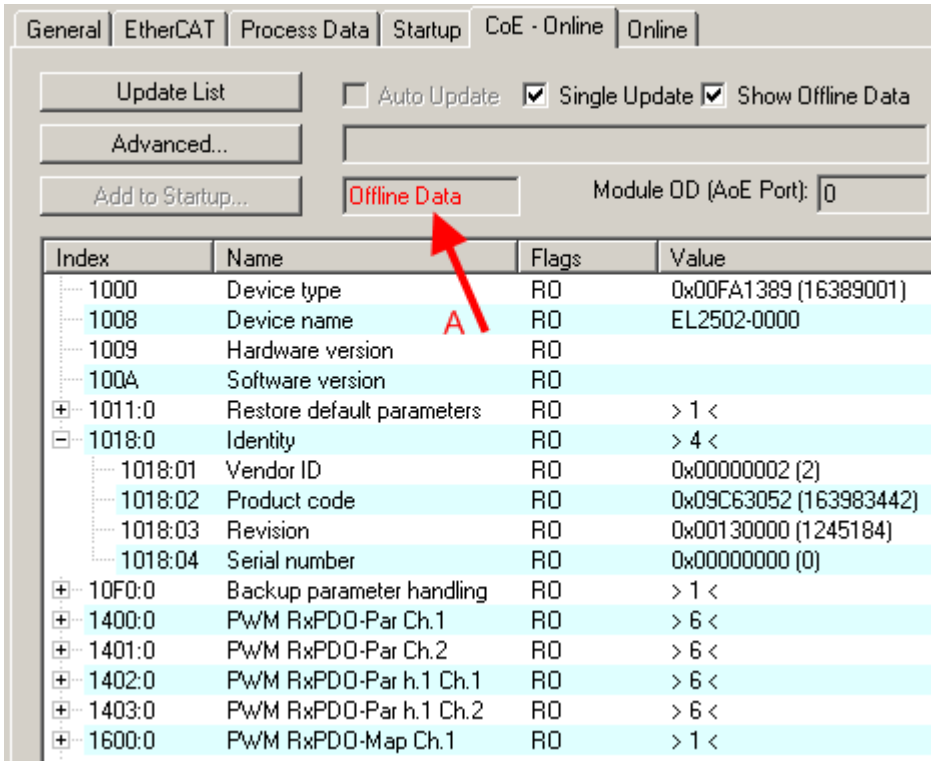


Fig. 11: Offline list

- 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.

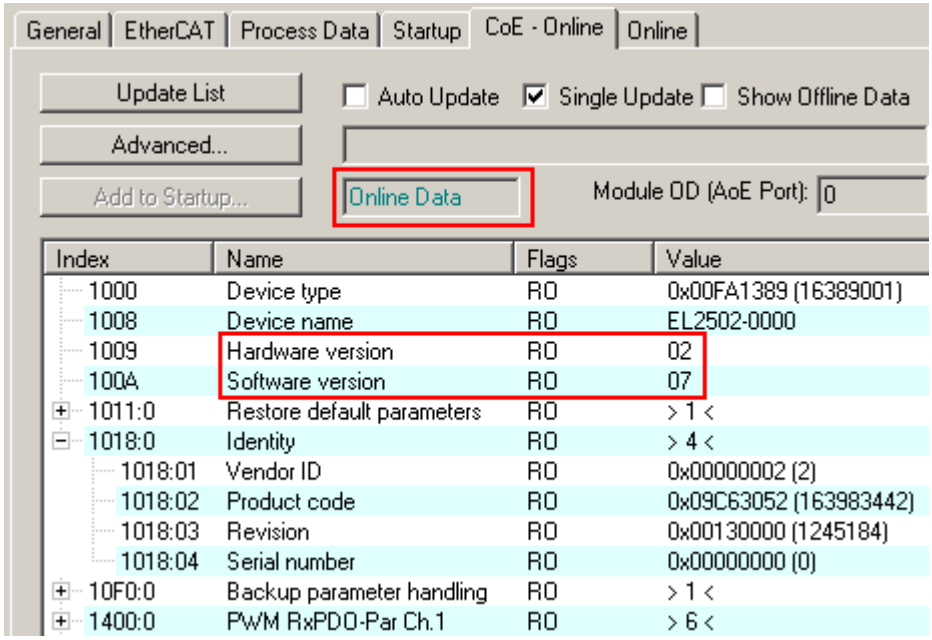


Fig. 12: Online list

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.

5.6 Distributed Clock

The distributed clock represents a local clock in the EtherCAT slave controller (ESC) with the following characteristics:

- Unit *1 ns*
- Zero point *1.1.2000 00:00*
- Size *64 bit* (sufficient for the next 584 years; however, some EtherCAT slaves only offer 32-bit support, i.e. the variable overflows after approx. 4.2 seconds)
- The EtherCAT master automatically synchronizes the local clock with the master clock in the EtherCAT bus with a precision of < 100 ns.

For detailed information please refer to the EtherCAT system description.

6 Basics of EnDat 2.2 technology

Basic information on the technological field of "EnDat interface" is to be given below. It is up to users to check to what extent this information refers to their application.

6.1 EnDat 2.2 - Bidirectional interface for position measuring devices

Digital drive systems and position control loops with position measuring devices for data sampling require measuring equipment offering fast data transfer with high transmission integrity.
In addition, further data such as drive-specific characteristic values, correction tables etc. should be provided. To ensure high system reliability, the measuring devices should be integrated in error detection routines and offer diagnostic capabilities.

The EnDat 2.2 interface is a digital, bidirectional interface for measuring devices. It is able to output position values of incremental and absolute measuring devices and read and update information stored in the measuring device, as well as store new information. four signal cables are sufficient, due to the fact that the data transfer is serial. The data is transferred synchronous with the clock signal specified by the downstream electronics. The transmission type (position values, parameters, diagnostics ...) is selected via mode commands, received by the measuring device from the downstream electronics. As a purely serial interface, EnDat 2.2 is also suitable for safety applications.

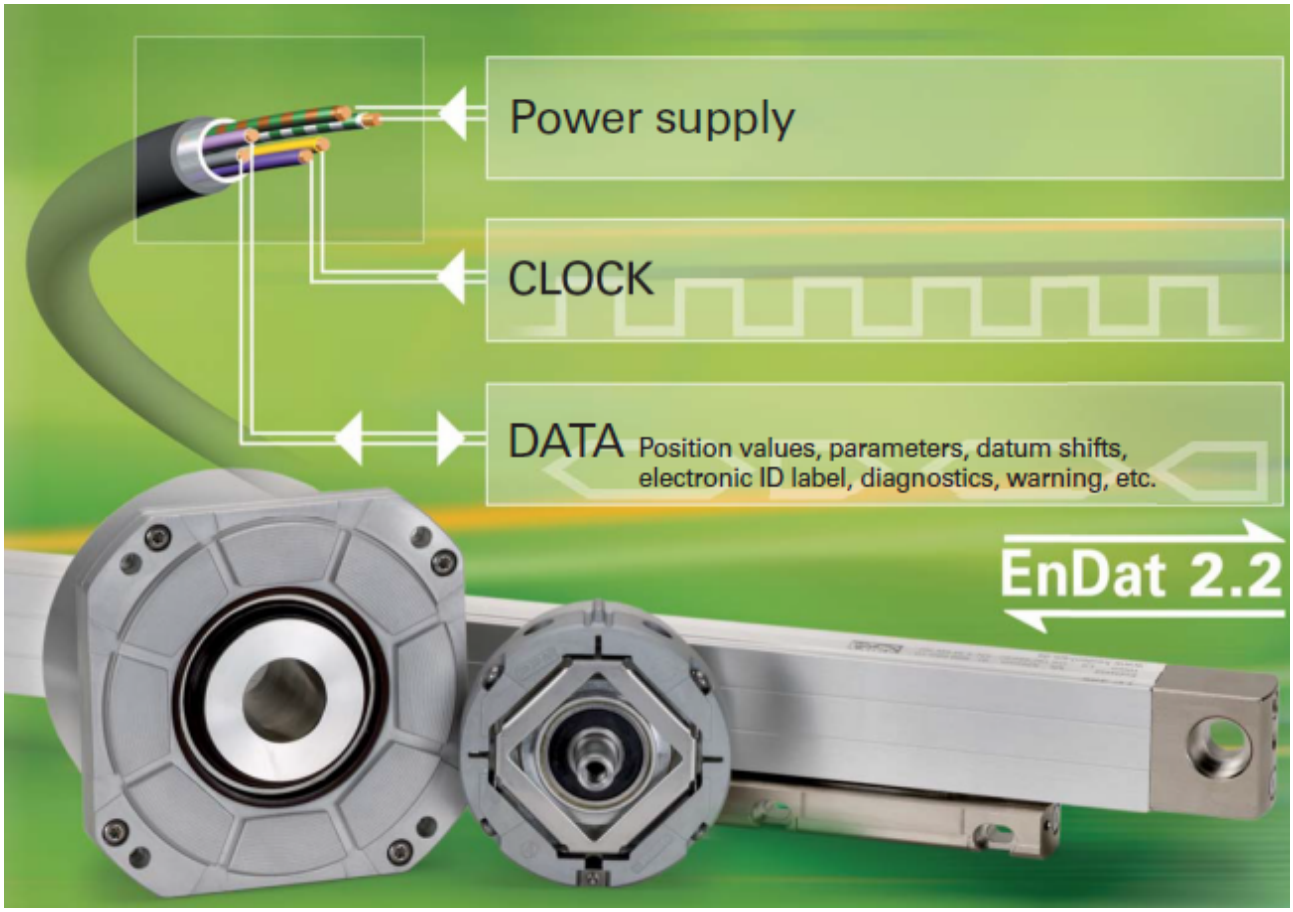


Fig. 13: Signal cables and power supply for the EnDat 2.2 interface**

Incremental signals and ordering designation (HEIDENHAIN EnDat interfaces)**

Some encoders also provide incremental signals. These are usually used to increase the resolution of the position value, or to serve a second subsequent electronics unit.
Current generations of encoders have a high internal resolution, and therefore no longer need to provide incremental signals.

The order designation indicates whether an encoder outputs incremental signals:

- EnDat 01 with 1 V_{PP} incremental signals
- EnDat H with HTL incremental signals
- EnDat T with TTL incremental signals
- EnDat 21 without incremental signals
- EnDat 02 with 1 V_{PP} incremental signals
- EnDat 22 without incremental signals

**) Source: Dr. Johannes HEIDENHAIN GmbH

Connection of HEIDENHAIN EnDat-interfaces to terminals of the EL503x series

Terminals of the EL503x series [▶ 22] can only be connected to measuring instruments that support a clock frequency of 8 MHz. This corresponds to the order designation EnDat22.

Encoders with the ordering designation EnDat02 can be operated at 8 MHz on terminals of the EL503x series without incremental signals using an appropriate adapter cable.

6.2 Connection technology

EnDat-02 encoder connection**

EnDat-02 encoders can have a plug-in cable module (e.g. LC 415 [Heidenhain]). The customer decides on the design of the adapter cable whether the device is operated with incremental signals [▶ 35] or without incremental signals.**

If an adapter cable without incremental signals is selected for the EnDat-02 encoder, the clock frequency of 8 MHz can be realized and the terminals of the EL503x series can be used.

**) Source: Dr. Johannes HEIDENHAIN GmbH

Data transmission from incremental or absolute measuring transducers to terminals of the EL503x series can be carried out with a single shielded 8-wire cable (see following figure).

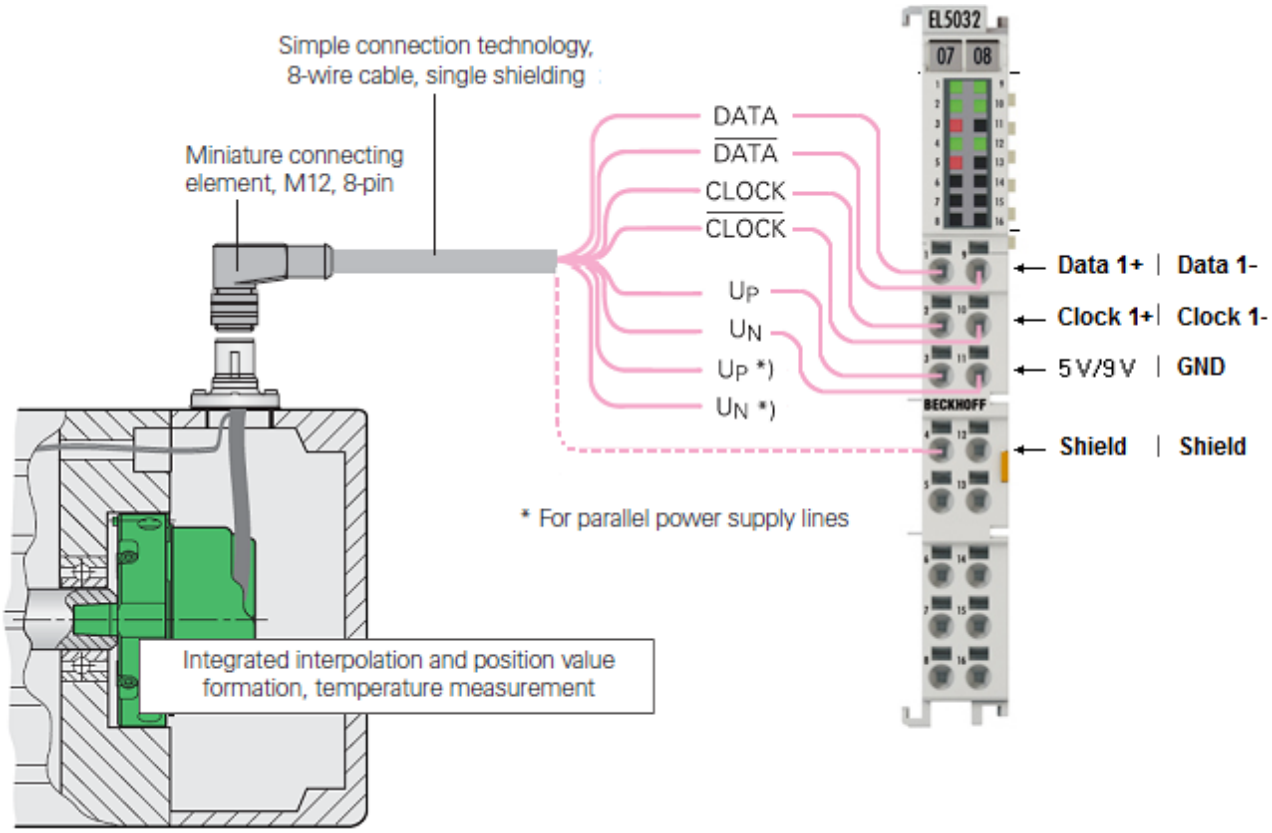


Fig. 14: Connection of measuring transducer and e.g. EL5032

6.3 Compatibility

The extended interface version EnDat 2.2 is compatible with the previous version 2.1 in terms of communication, command sets and time conditions, but offers significant advantages. For example, it is possible to send additional information with the position value without having to start a separate query. To this end the interface protocol has been extended and the time conditions optimized as follows:

- Increased clock frequency (CLOCK)
- Optimized computing time (determination of position value within 5 µs)
- Minimize recovery time (1.25 to 3.75 µs)
- Extended voltage supply range (U_P = 3.6 to 5.25 V or 3.6 to 14 V at the encoder)

EnDat 2.2 with command set (includes EnDat 2.1 command set)

- Position values for incremental and absolute encoders
- Additional information on the position value
  - Diagnosis, test values
  - Absolute position values after referencing incremental encoders
  - Sending and receiving parameters
  - Commutation
  - Acceleration
  - Limit position signal
  - Position value 2 for safety-related applications or incremental encoders

EnDat with command set 2.1

- Absolute position values
- Sending and receiving parameters
- Reset
- Test command
- Test values

6.4 Functional description

The EnDat interface transmits position values or physical additional variables in a temporally unambiguous order and is used for reading from, and writing to the internal memory of the measuring device.

Position values

These can be transmitted with or without additional information. The additional information itself can be selected via the MRS code (**M**emory **R**ange **S**elect). Together with the position value, other functions can be called after a previous memory area selection, e.g. Read and write parameters. Simultaneous transmission with the position value means that additional information can also be requested from axes in the control loop and functions executed.

Read and write parameters

Read and write parameters is possible both as a separate function and also in conjunction with the position value. Parameters can be read or written after selection of the memory area.

Reading or writing values of the EndDat encoder

If the configuration and information data access specific memory areas of the Endat encoder, make sure that the encoder is connected and supplied with voltage to read or write to these memory areas.
If no communication can be established with the EnDat encoder, an error message "Data cannot be transferred or stored to the application" is displayed.

Reset functions

The reset functions serve to reset the measuring device in case of a malfunction. A reset is possible in place of or during the transmission of the position value.

Extended diagnosis from FW15

From FW15, the EL5032 offers extended diagnosis for the EnDat error message with multi-turn rotary encoders with battery buffer. The battery operating state error sources are not automatically reset; special error handling may be necessary here.

Special error handling with multi-turn rotary encoders with battery buffer

The absolute position value loses its validity if the battery of a battery-buffered multi-turn rotary encoder is replaced. EnDat 2.2 allows error and warning messages to be read and temporarily stored.

After the battery is replaced, the following objects are set to TRUE:

- 0x9008:23 [▶ 146] "Position error supported" and
- 0x9008:34 [▶ 146] "Battery warning supported".

This activates the following diagnosis messages:

- 0xA008:01 [▶ 148] "Warning",
- 0xA008:02 [▶ 148] "Error" 1,
- 0xA008:23 [▶ 148] "Position error" and
- 0xA008:23 [▶ 148] "Battery warning".

These errors can be reset using the command 0x8001 "EnDat Reset". An overview of the commands and samples of their use can be found in the chapter "Overview of commands and samples [▶ 137]".

6.5 Data transfer

A clock is provided by the subsequent electronics for the synchronization of the data transmission. In the idle state the clock line is at the HIGH level.

Clock frequency – cable length

With the runtime compensation used in the terminals of the EL503x series, a maximum cable length of 100 m is possible at a clock frequency of 8 MHz. Appropriate cables and connectors must be used to ensure proper operation.

Further information on EnDat technology

Further notes and information on EnDat technology from HEIDENHAIN can be found at http://www.heidenhain.de

**) Source: Dr. Johannes HEIDENHAIN GmbH

7 Mounting and wiring

7.1 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.

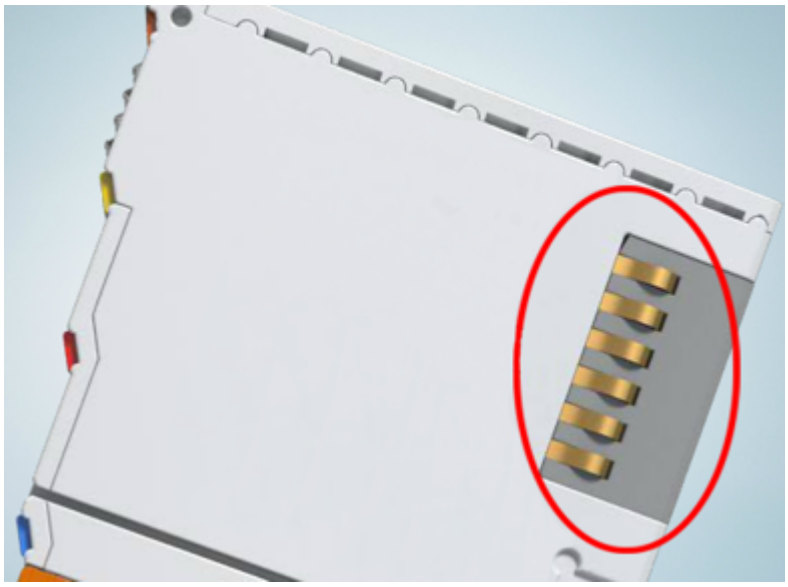


Fig. 15: Spring contacts of the Beckhoff I/O components

7.2 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

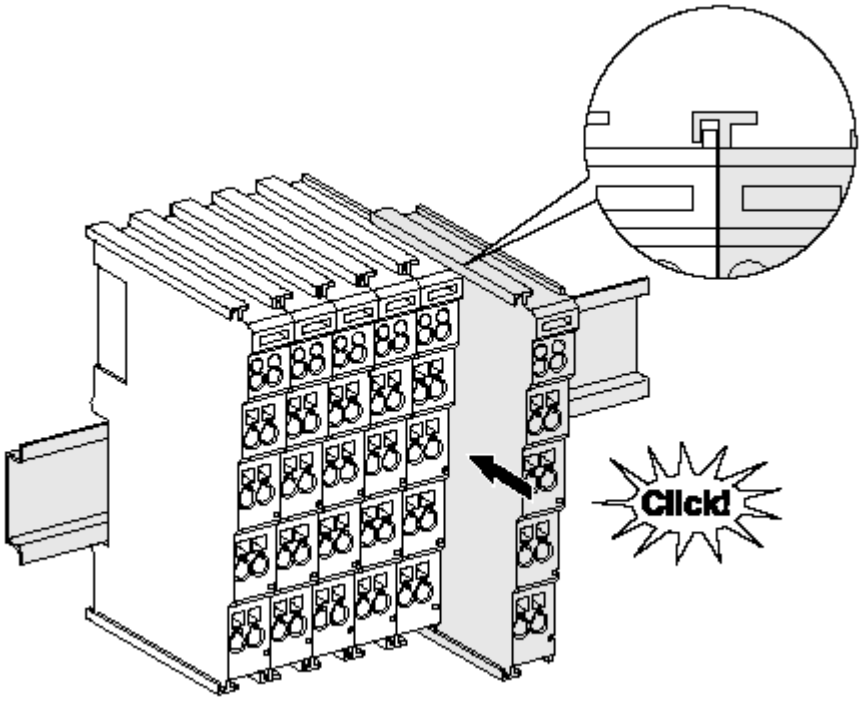


Fig. 16: Attaching on mounting rail

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).

Disassembly

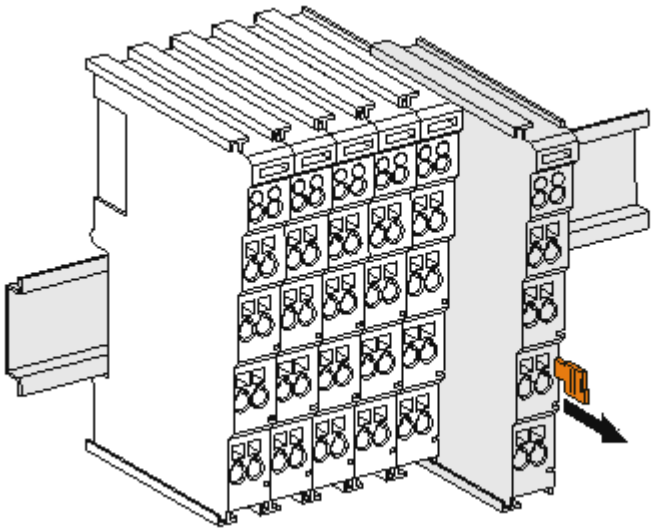


Fig. 17: Disassembling of terminal

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.

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 K-Bus/E-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 (KL91xx, KL92xx or EL91xx, EL92xx) interrupt the power contacts and thus represent the start of a new supply rail.

PE power contact

The power contact labeled PE can be used as a protective earth. For safety reasons this contact mates first when plugging together, and can ground short-circuit currents of up to 125 A.

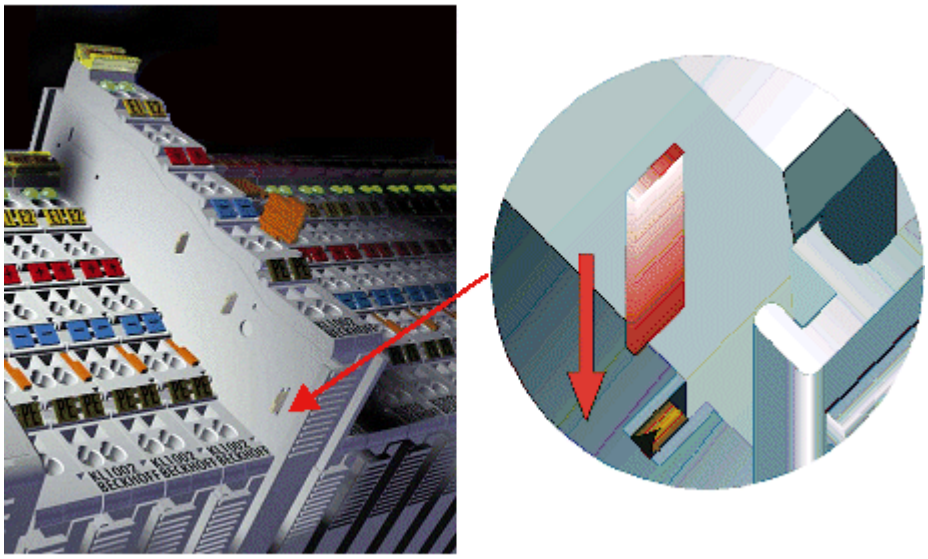


Fig. 18: Power contact on left side

NOTICE

Possible damage of the device

Note that, for reasons of electromagnetic compatibility, the PE 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 PE line during insulation testing of a consumer with a nominal voltage of 230 V). For insulation testing, disconnect the PE 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.

⚠ WARNING

Risk of electric shock!

The PE power contact must not be used for other potentials!

7.3 Installation instructions for enhanced mechanical load capacity

⚠ WARNING

Risk of injury through electric shock and damage to the device!

Bring the Bus Terminal system into a safe, de-energized state before starting mounting, disassembly or wiring of the Bus Terminals!

Additional checks

The terminals have undergone the following additional tests:

Verification	Explanation
Vibration	10 frequency runs in 3 axes
	6 Hz < f < 60 Hz displacement 0.35 mm, constant amplitude
	60.1 Hz < f < 500 Hz acceleration 5 *g*, constant amplitude
Shocks	1000 shocks in each direction, in 3 axes
	25 *g*, 6 ms

Additional installation instructions and notes

For terminals with enhanced mechanical load capacity, the following additional installation instructions and notes apply:

- The enhanced mechanical load capacity is valid for all permissible installation positions.
- Use a mounting rail according to EN 60715 TH35-15.
- Fix the terminal segment on both sides of the mounting rail with a mechanical fixture, e.g. an earth terminal or reinforced end clamp.
- The maximum total extension of the terminal segment (without coupler) is:
  64 terminals (12 mm mounting width) or 32 terminals (24 mm mounting width)
- Avoid deformation, twisting, crushing and bending of the mounting rail during edging and installation of the rail.
- The mounting points of the mounting rail must be set at 5 cm intervals.
- Use countersunk head screws to fasten the mounting rail.
- The free length between the strain relief and the wire connection should be kept as short as possible. A distance of approx. 10 cm should be maintained to the cable duct.

7.4 Connection

7.4.1 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. 19: Standard wiring

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

Pluggable wiring (ESxxxx / KSxxxx)

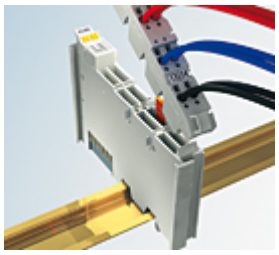


Fig. 20: Pluggable wiring

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. 21: 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 [▶ 48]!

7.4.2 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

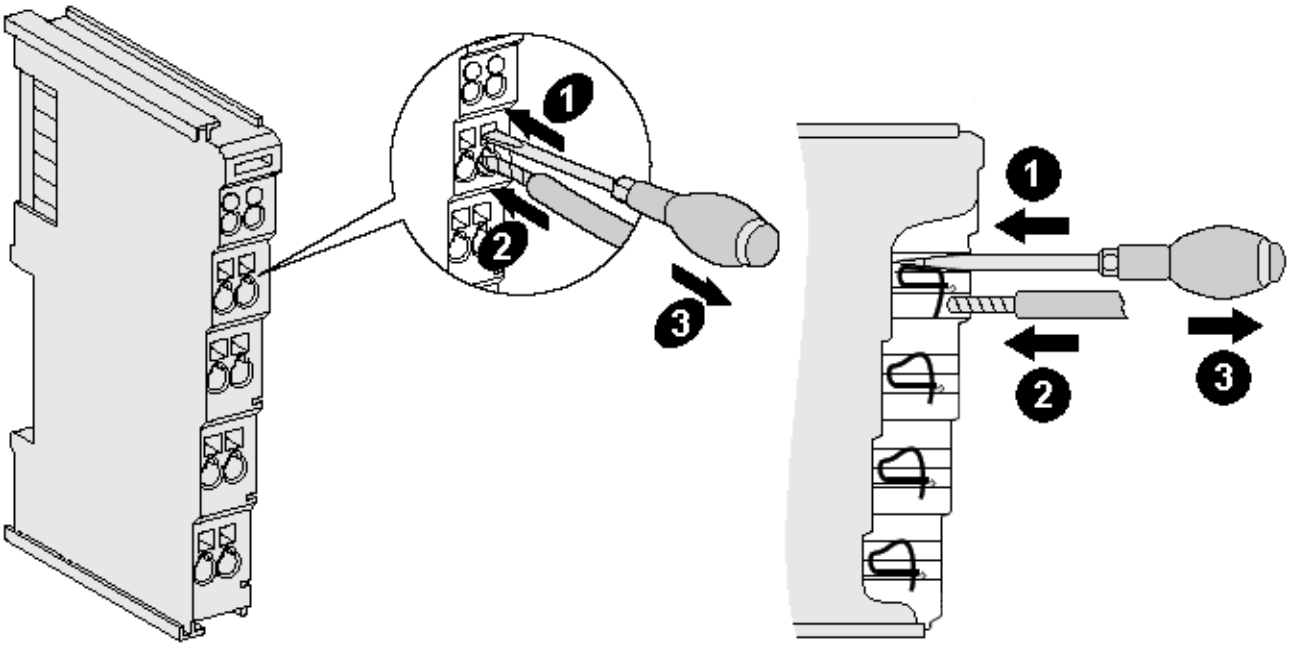


Fig. 22: Connecting a cable on a terminal point

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 [▶ 45]) 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 [▶ 46])
Wire stripping length	8 ... 9 mm

7.4.3 Shielding

Shielding

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

7.5 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.

7.6 Installation positions

NOTICE

Constraints regarding installation position and operating temperature range

Please refer to the technical data for a terminal to ascertain whether any restrictions regarding the installation position and/or the operating temperature range have been specified. When installing high power dissipation terminals ensure that an adequate spacing is maintained between other components above and below the terminal in order to guarantee adequate ventilation!

Optimum installation position (standard)

The optimum installation position requires the mounting rail to be installed horizontally and the connection surfaces of the EL- / KL terminals to face forward (see Fig. “Recommended distances for standard installation position”). The terminals are ventilated from below, which enables optimum cooling of the electronics through convection. “From below” is relative to the acceleration of gravity.

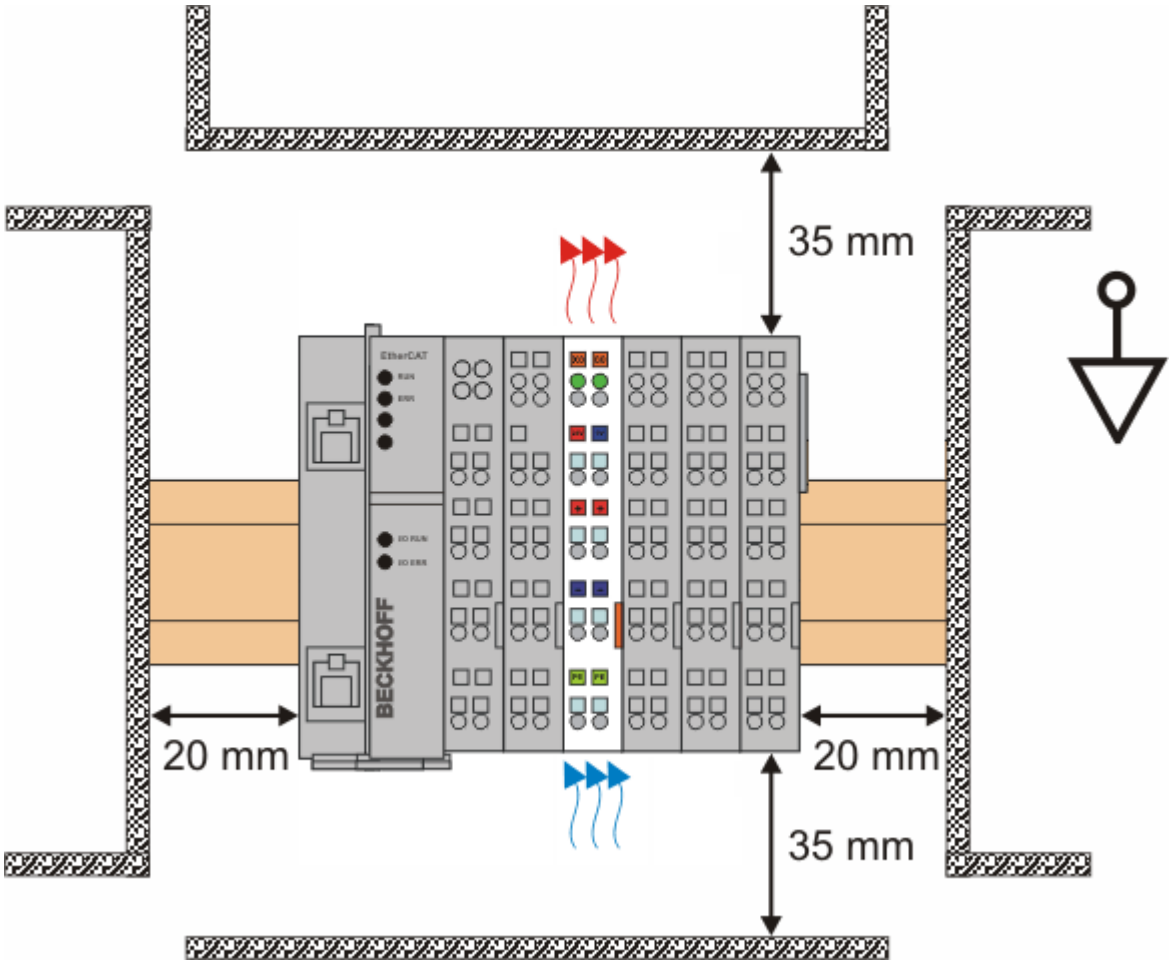


Fig. 23: Recommended distances for standard installation position

Compliance with the distances shown in Fig. “Recommended distances for standard installation position” is recommended.

Other installation positions

All other installation positions are characterized by different spatial arrangement of the mounting rail - see Fig “Other installation positions”.

The minimum distances to ambient specified above also apply to these installation positions.

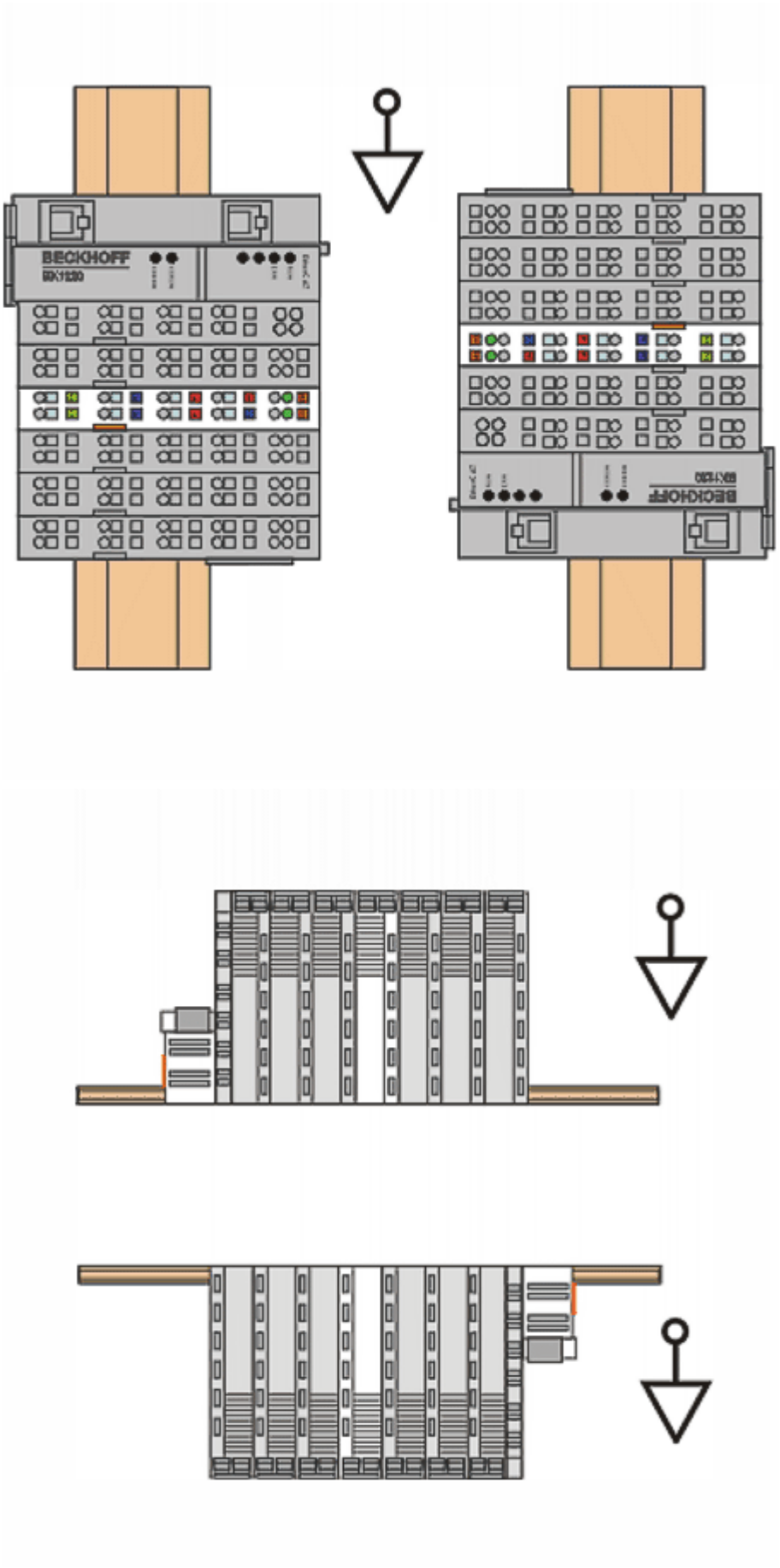


Fig. 24: Other installation positions

7.7 Positioning of passive Terminals

Hint for positioning of passive terminals in the bus terminal block

EtherCAT Terminals (ELxxxx / ESxxxx), 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)

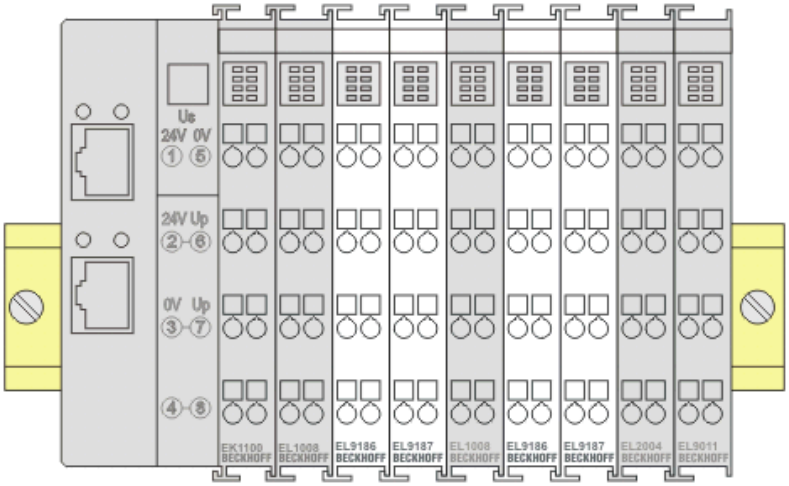


Fig. 25: Correct positioning

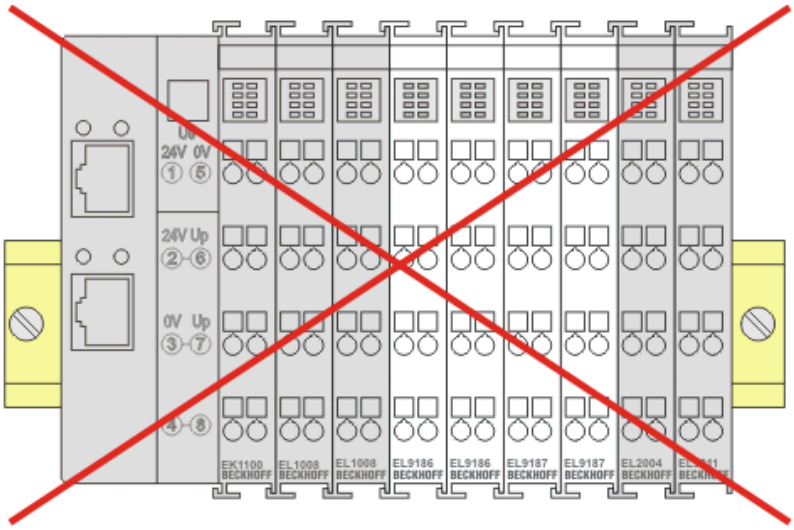


Fig. 26: Incorrect positioning

7.8 UL notice

⚠ CAUTION	
c UL us	**Application** Beckhoff EtherCAT modules are intended for use with Beckhoff's UL Listed EtherCAT System only.

⚠ CAUTION	
c UL us	**Examination** For cULus examination, the Beckhoff I/O System has only been investigated for risk of fire and electrical shock (in accordance with UL508 and CSA C22.2 No. 142).

⚠ CAUTION	
c UL us	**For devices with Ethernet connectors** Not for connection to telecommunication circuits.

Basic principles

UL certification according to UL508. Devices with this kind of certification are marked by this sign:



7.9 LEDs and connection

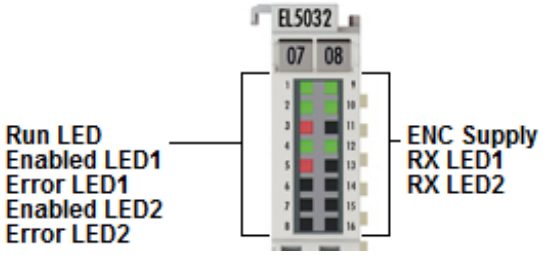


Fig. 27: LEDs

NOTICE

Otherwise components may be damaged: Note encoder supply voltage!

Note the limit values for the supply voltage specified in the data sheets of the encoder manufacturers. The encoder supply voltage may have to be adjusted in object 0x80p8:13 [▶ 144] (5 V or 9 V)!

LEDs

LED	Color	Meaning	
RUN (1)	green	This LED indicates the terminal's operating state:	
		off	State of the EtherCAT State Machine: **INIT** = initialization of the terminal
		flashing	State of the EtherCAT State Machine: **PREOP** = function for mailbox communication and different standard-settings set
		single flash	State of the EtherCAT State Machine: **SAFEOP** = verification of the sync manager 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 terminal firmware updates
ENABLED 1 (2) ENABLED 2 (4)	green	ON	Connected encoder for the corresponding channel initialized and ready for operation
		OFF	Connected encoder for the corresponding channel not ready for operation
ERROR 1 (3) ERROR 2 (5)	red	ON	No encoder connected to the corresponding channel, or position values invalid (CRC error, not referenced, EnDat error bit set)
		FLASHES	Error during initialization (see diag history [▶ 165])
		OFF	No error
ENC SUPPLY (9)	green	ON	Encoder voltage available
		OFF	24 V field voltage missing or encoder voltage overload
RX 1 (10) RX 2 (12)	green	FLASHES	Terminal receives position values at the corresponding channel

Connection

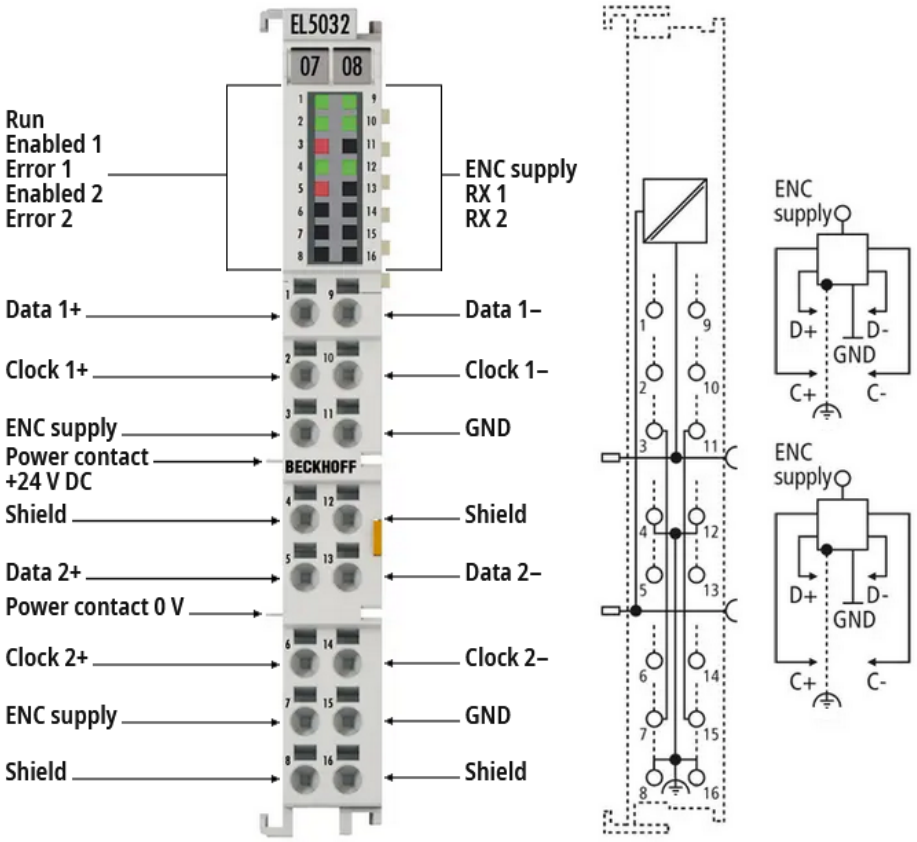


Fig. 28: Connection

NOTICE

Encoder supply via the terminal

The encoder supply voltage (5 V or 9 V), which can be set in object 0x80p8:13 [▶ 144], can be taken from the terminal points 3 (channel 1) and 7 (channel 2).

Terminal point		Description
Name	No.	
Data 1+	1	Data + input (channel 1)
Clock 1+	2	Clock + input (channel 1)
ENC supply	3	Supply voltage for encoder (+5 V / +9 V)
Shield	4	Shield
Data 2+	5	Data + input (channel 2)
Clock 2+	6	Clock + input (channel 2)
ENC supply	7	Supply voltage for encoder (+5 V / +9 V)
Shield	8	Shield
Data 1-	9	Data - input (channel 1)
Clock 1-	10	Clock - input (channel 1)
GND	11	Ground
Shield	12	Shield
Data 2-	13	Data - input (channel 2)
Clock 2-	14	Clock - input (channel 2)
GND	15	Ground
Shield	16	Shield

Further connection information

Further connection information can be found in the chapter "Technology [▶ 34]".

7.10 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.

8 Commissioning

8.1 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:

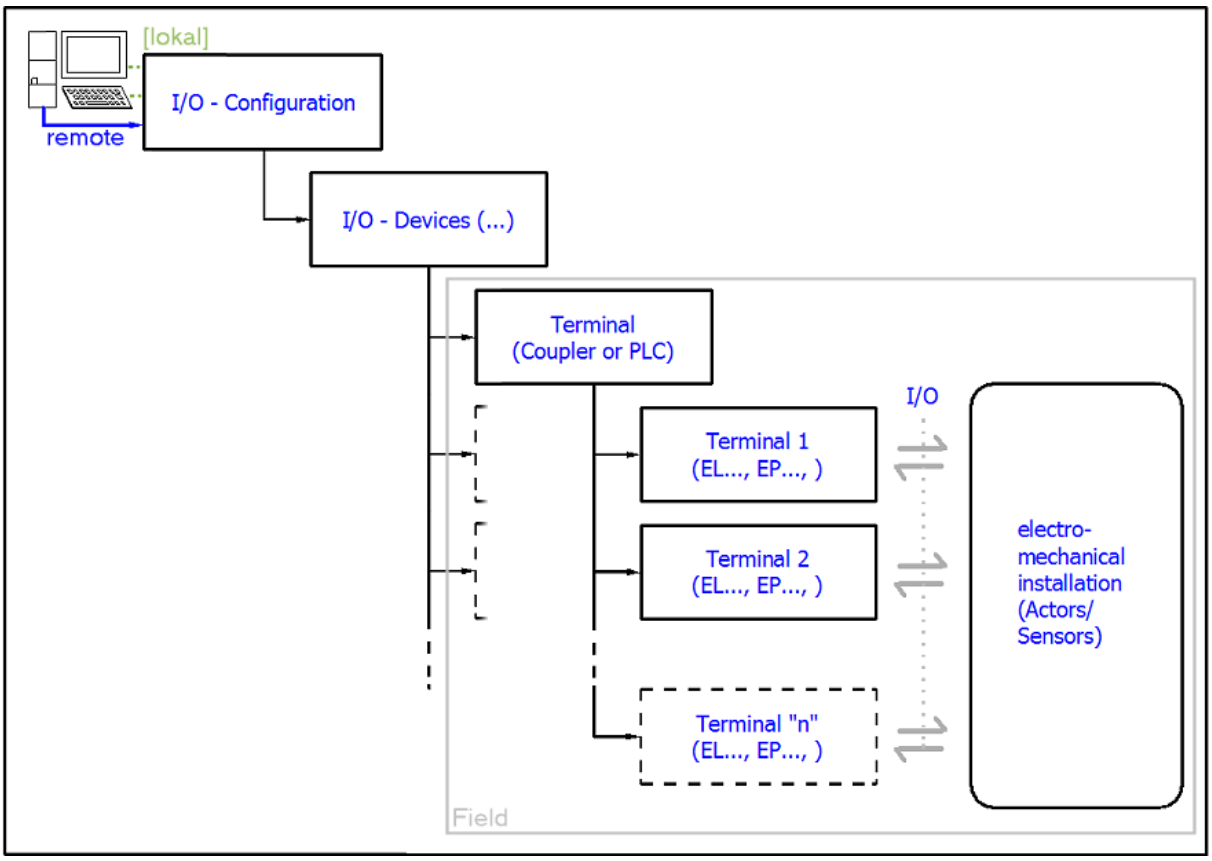


Fig. 29: Relationship between user side (commissioning) and installation

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)

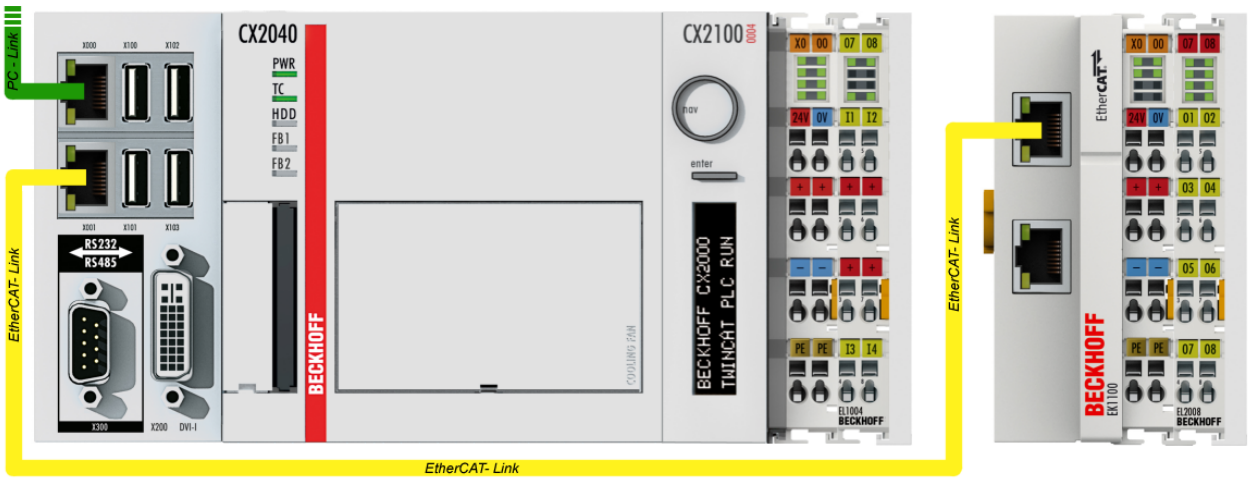


Fig. 30: Control configuration with Embedded PC, input (EL1004) and output (EL2008)

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.

8.1.1 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:

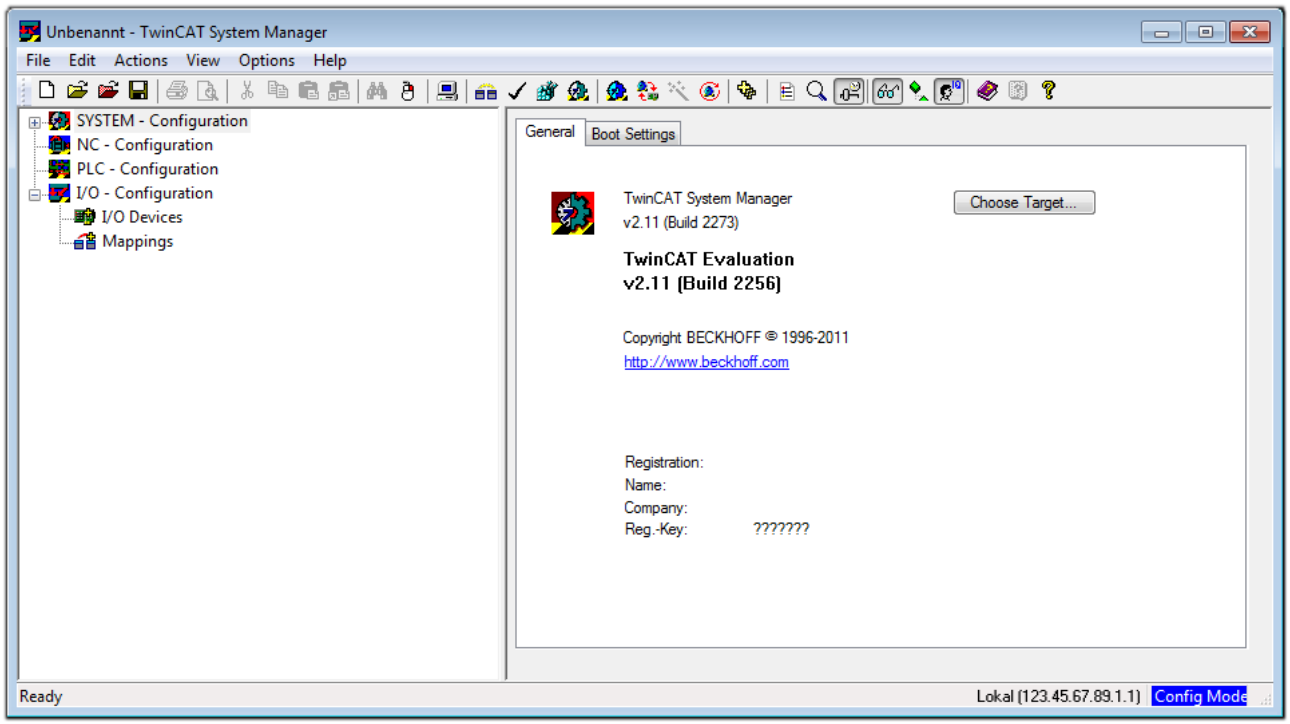


Fig. 31: Initial TwinCAT 2 user interface

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 [▶ 61]”.

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:

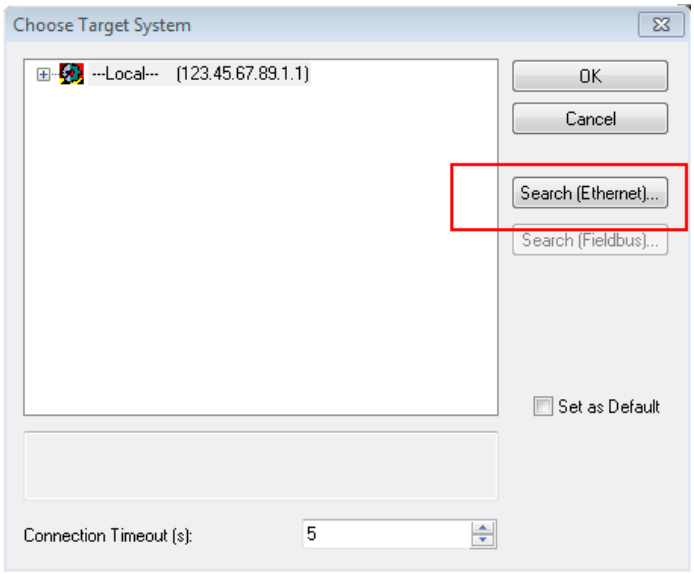


Fig. 32: Selection of the target system

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

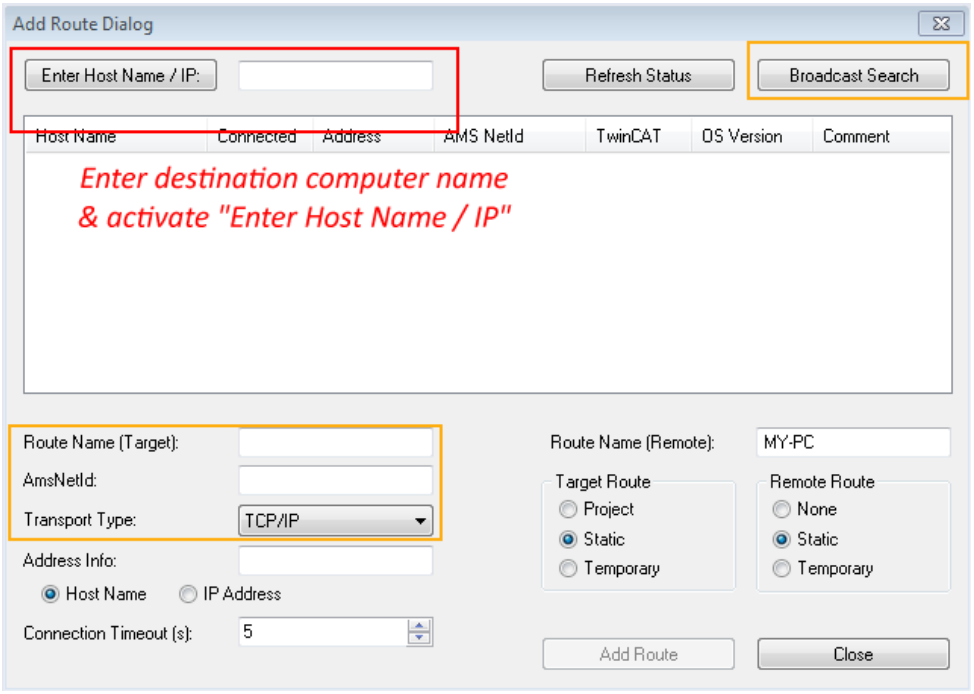
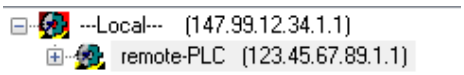


Fig. 33: specify the PLC for access by the TwinCAT System Manager: selection of the target system

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).

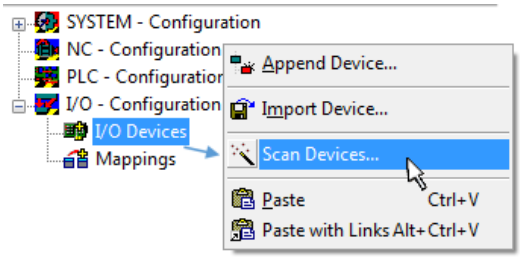


Fig. 34: Select “Scan Devices...”

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

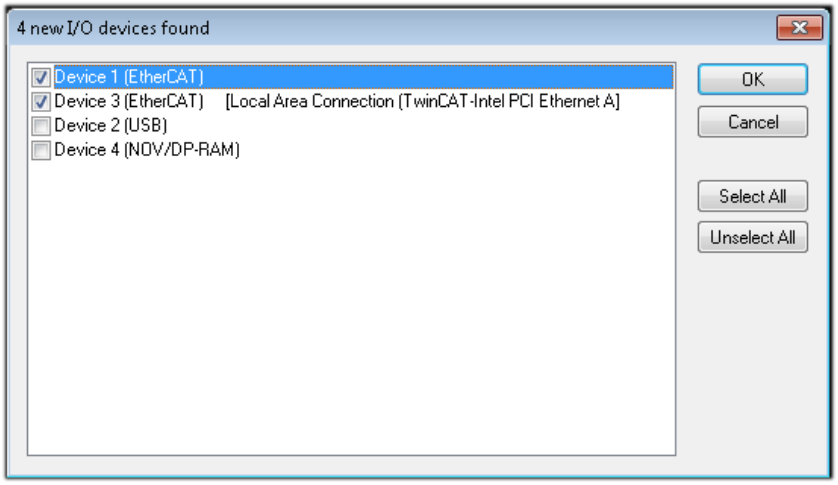


Fig. 35: Automatic detection of I/O devices: selection of the devices to be integrated

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 [▶ 57] described at the beginning of this section, the result is as follows:

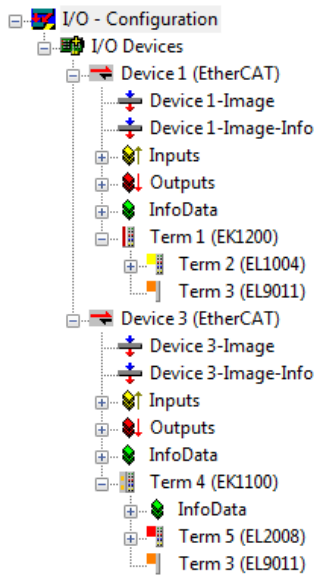


Fig. 36: Mapping of the configuration in the TwinCAT 2 System Manager

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:

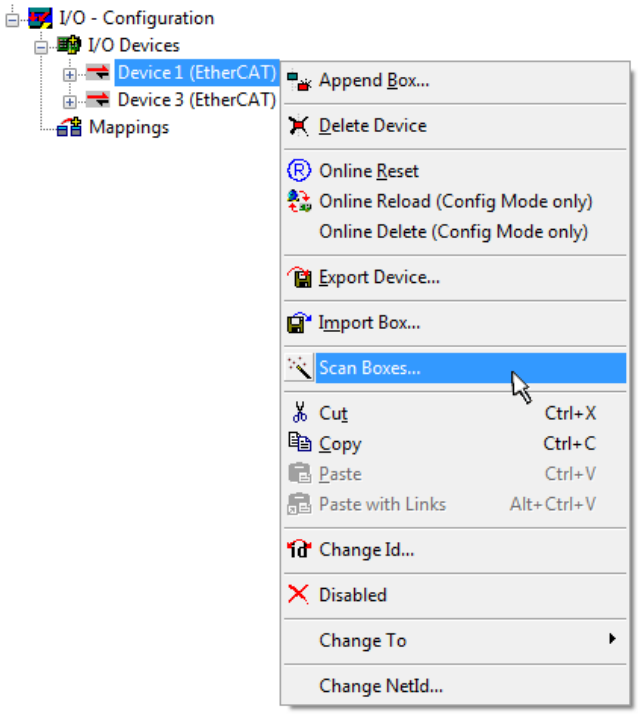


Fig. 37: Reading of individual terminals connected to a device

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:

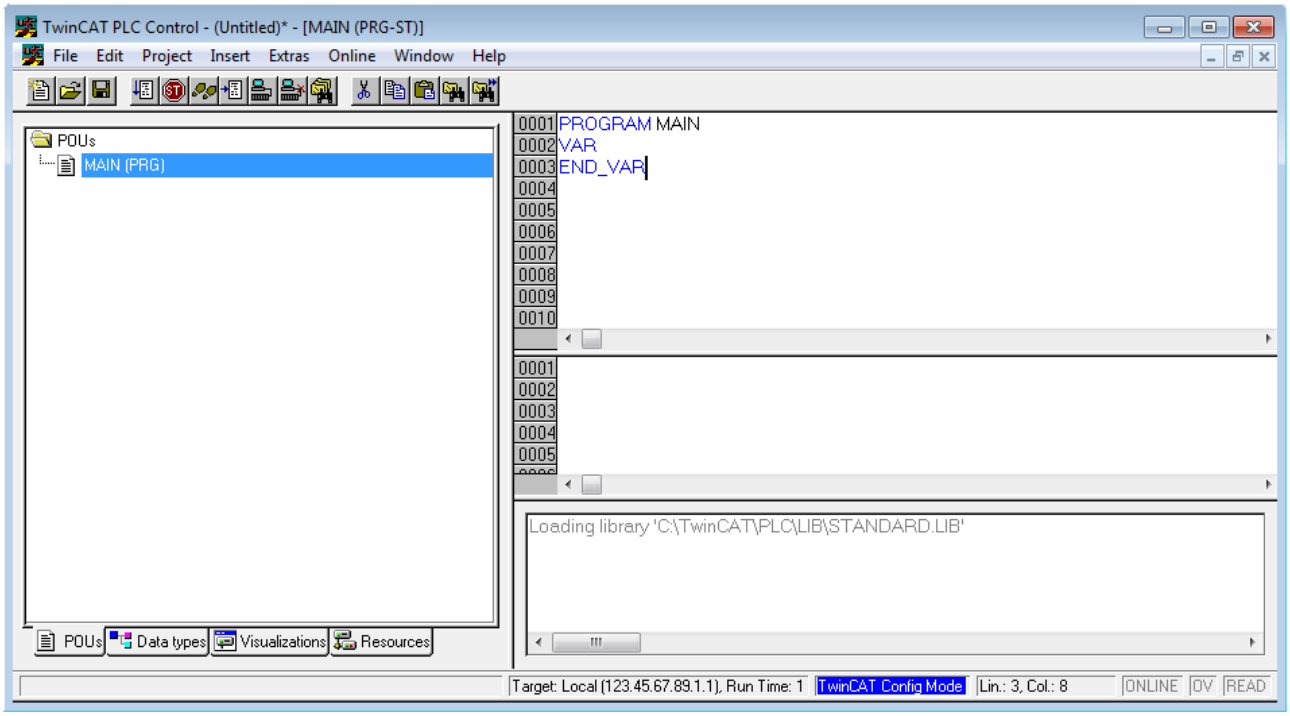


Fig. 38: TwinCAT PLC Control after startup

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

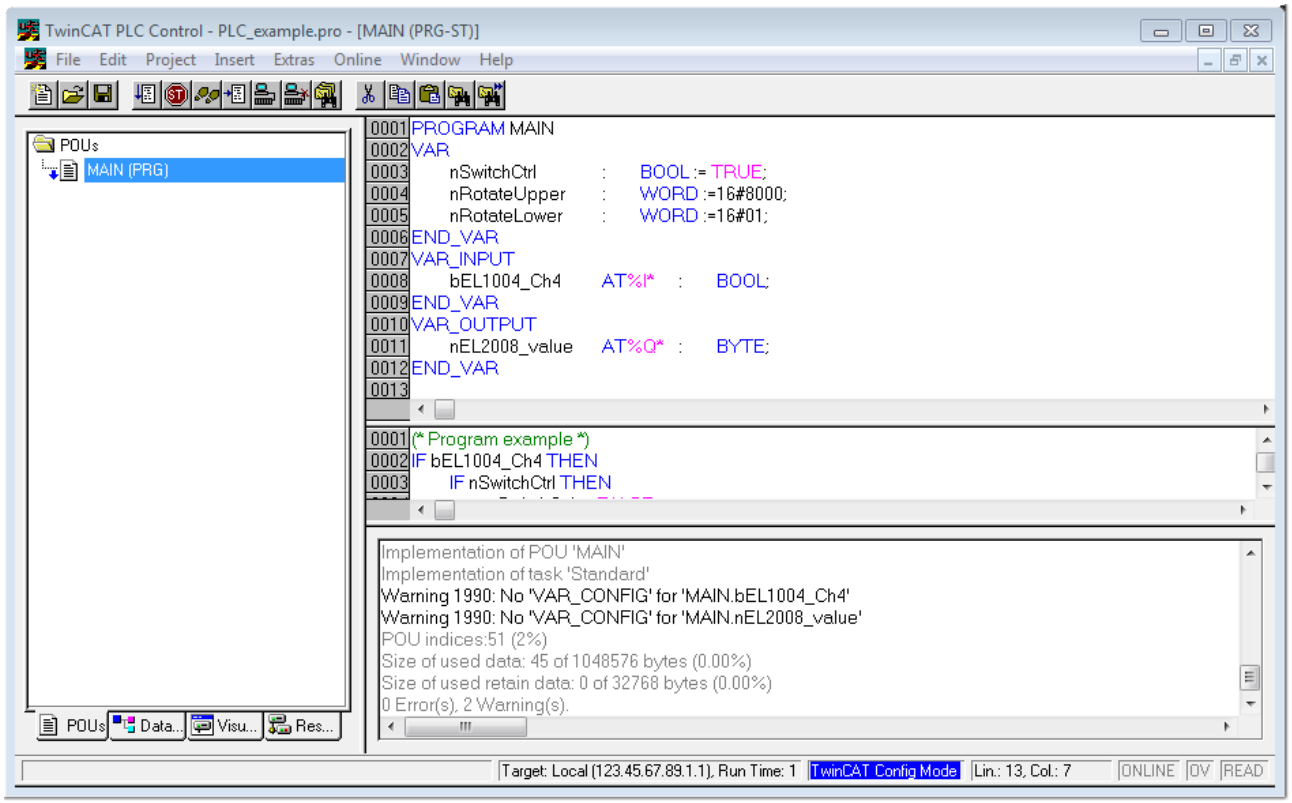


Fig. 39: Example program with variables after a compile process (without variable integration)

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…”:

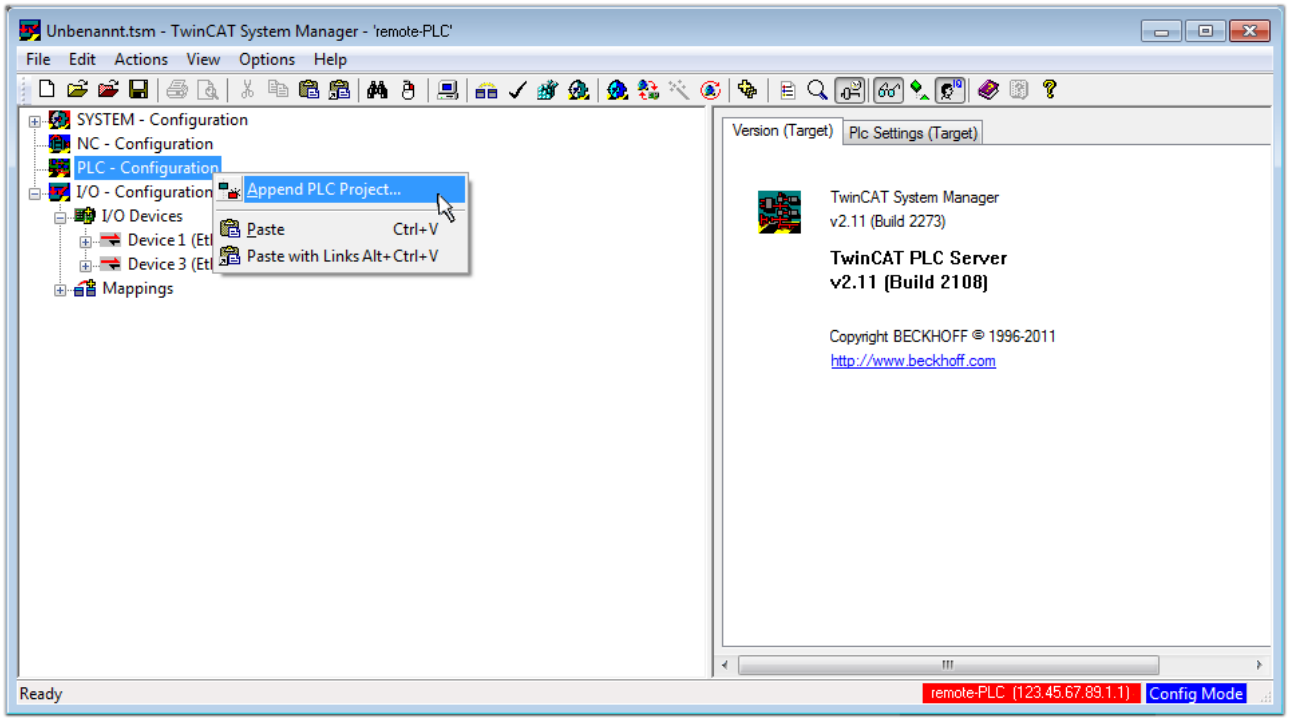


Fig. 40: Appending the TwinCAT PLC Control project

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:

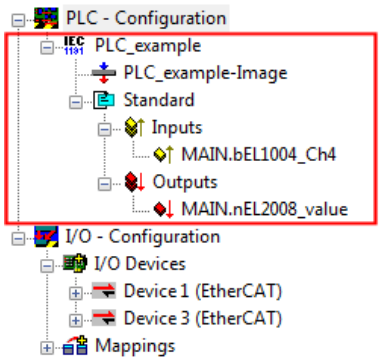


Fig. 41: PLC project integrated in the PLC configuration of the System Manager

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”:

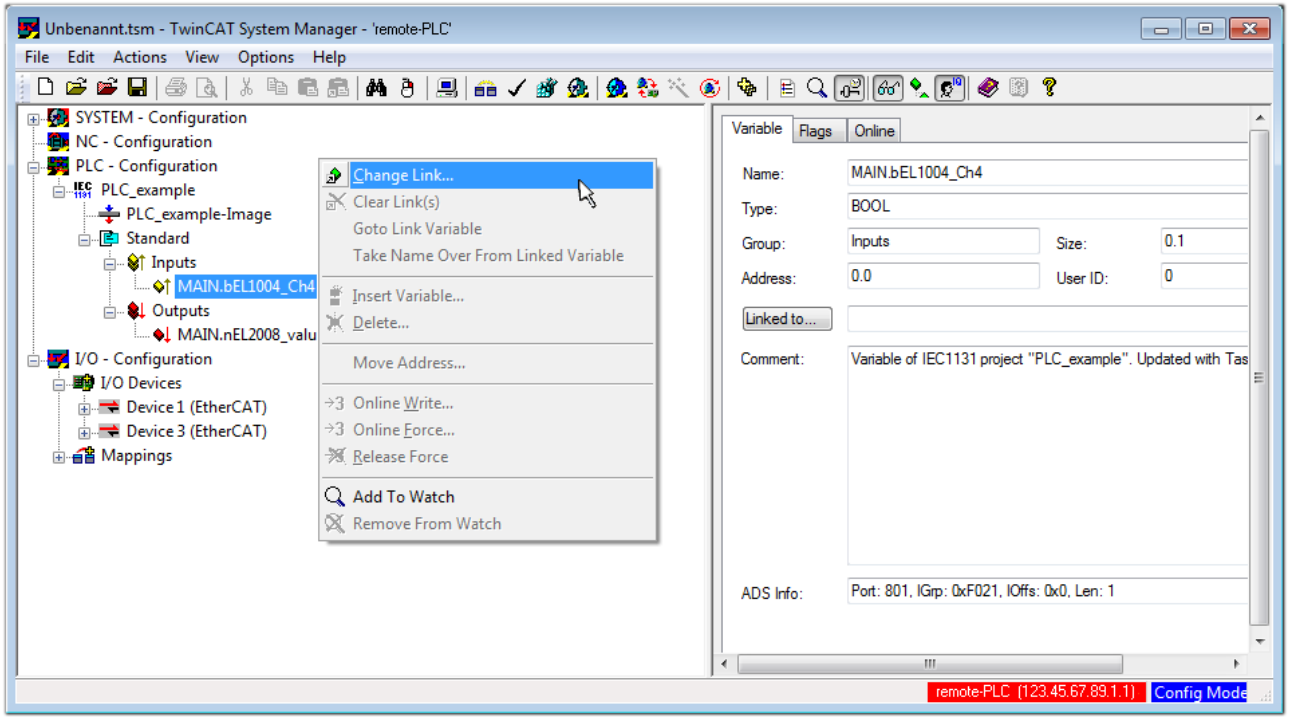


Fig. 42: Creating the links between PLC variables and process objects

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

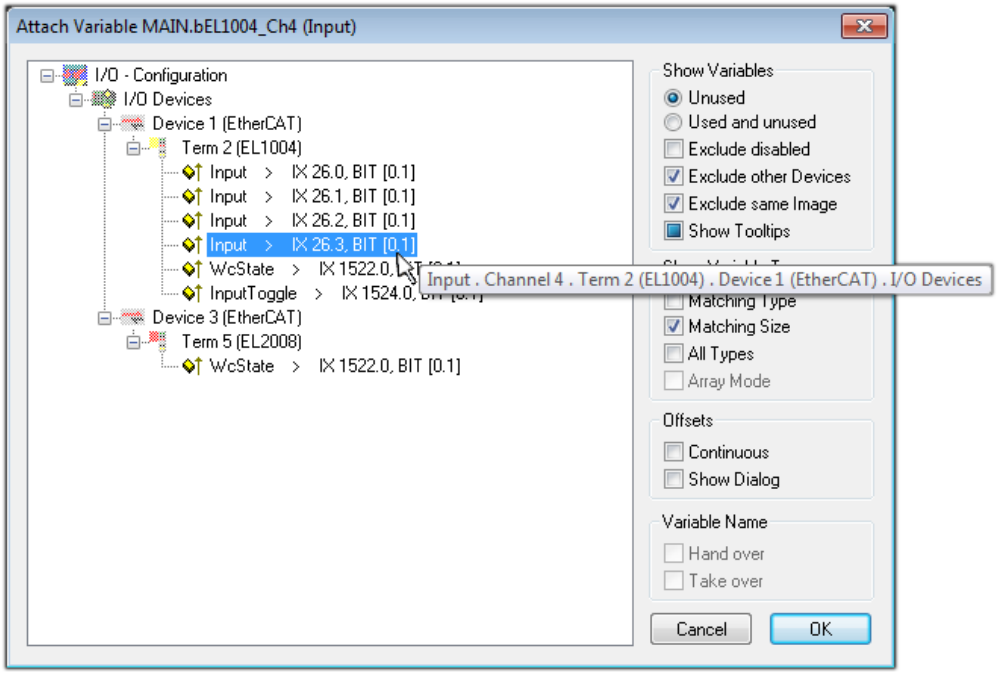


Fig. 43: Selecting BOOL-type PDO

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:

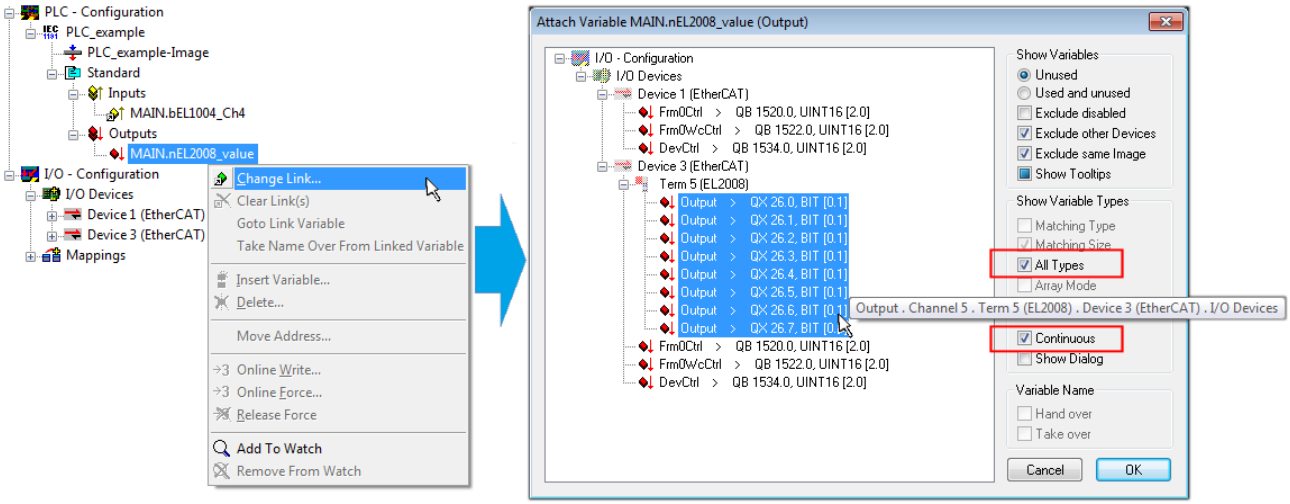


Fig. 44: Selecting several PDOs simultaneously: activate “Continuous” and “All types”

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:

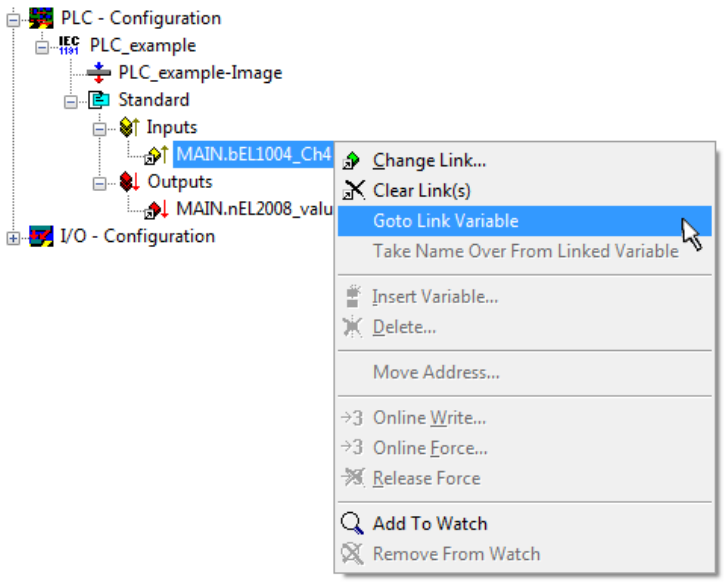
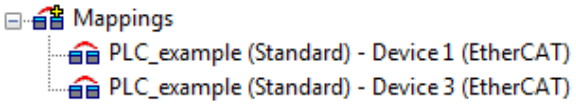


Fig. 45: Application of a “Goto Link Variable”, using “MAIN.bEL1004_Ch4” as an example

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…”:

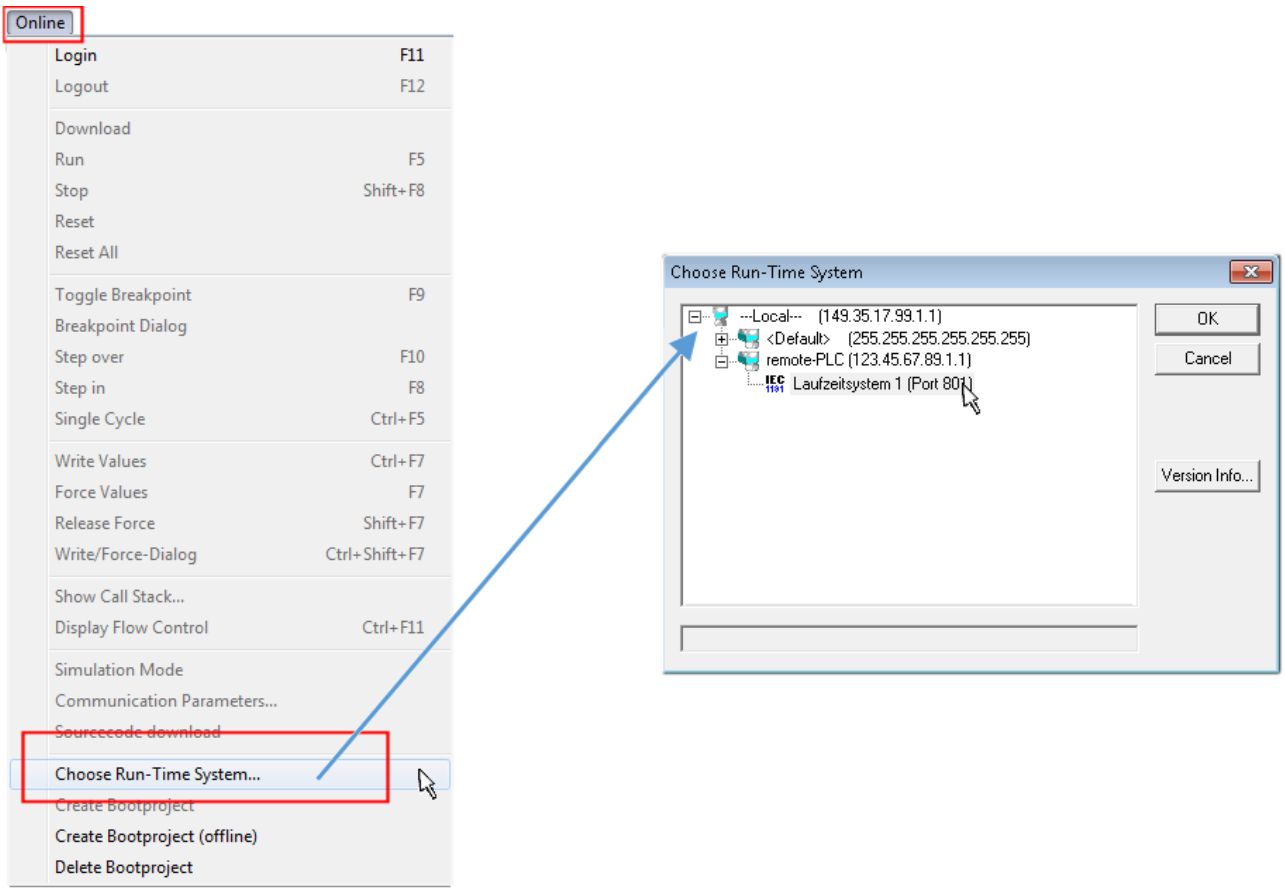


Fig. 46: Choose target system (remote)

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:

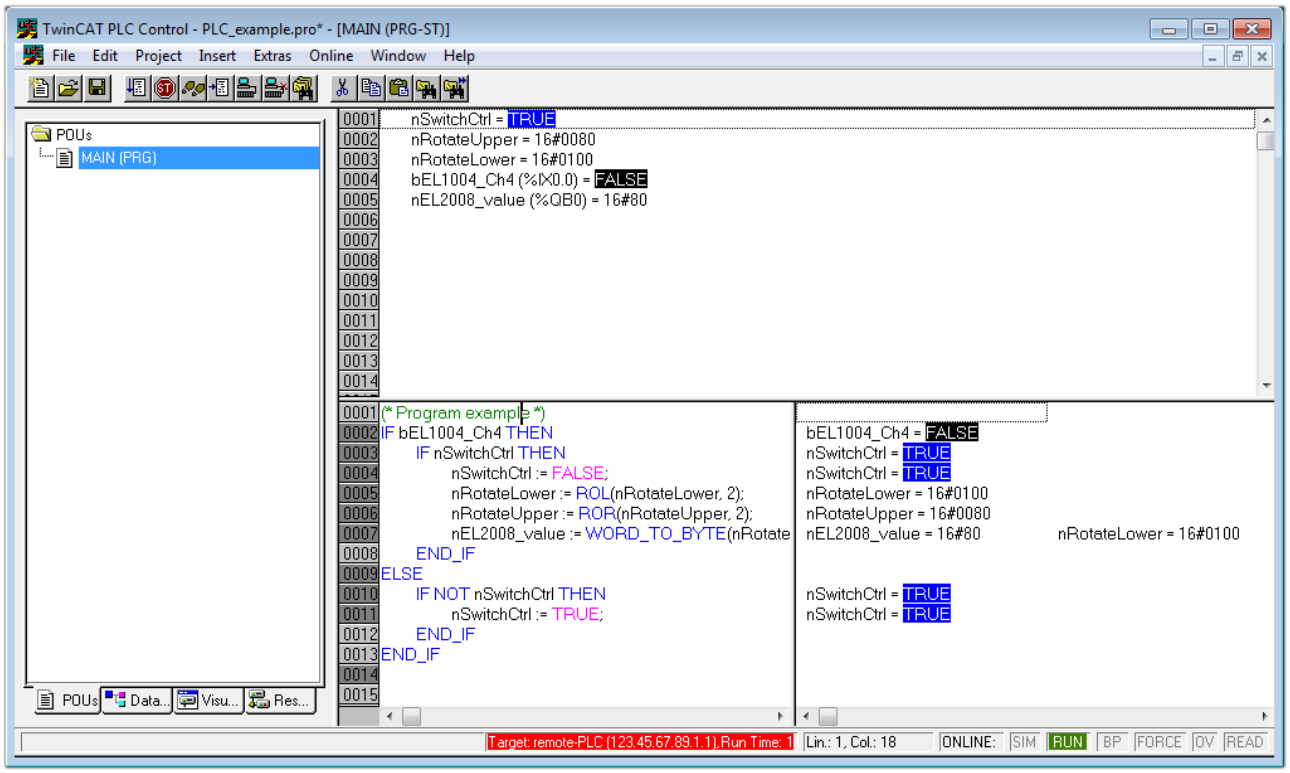


Fig. 47: PLC Control logged in, ready for program startup

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

8.1.2 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:

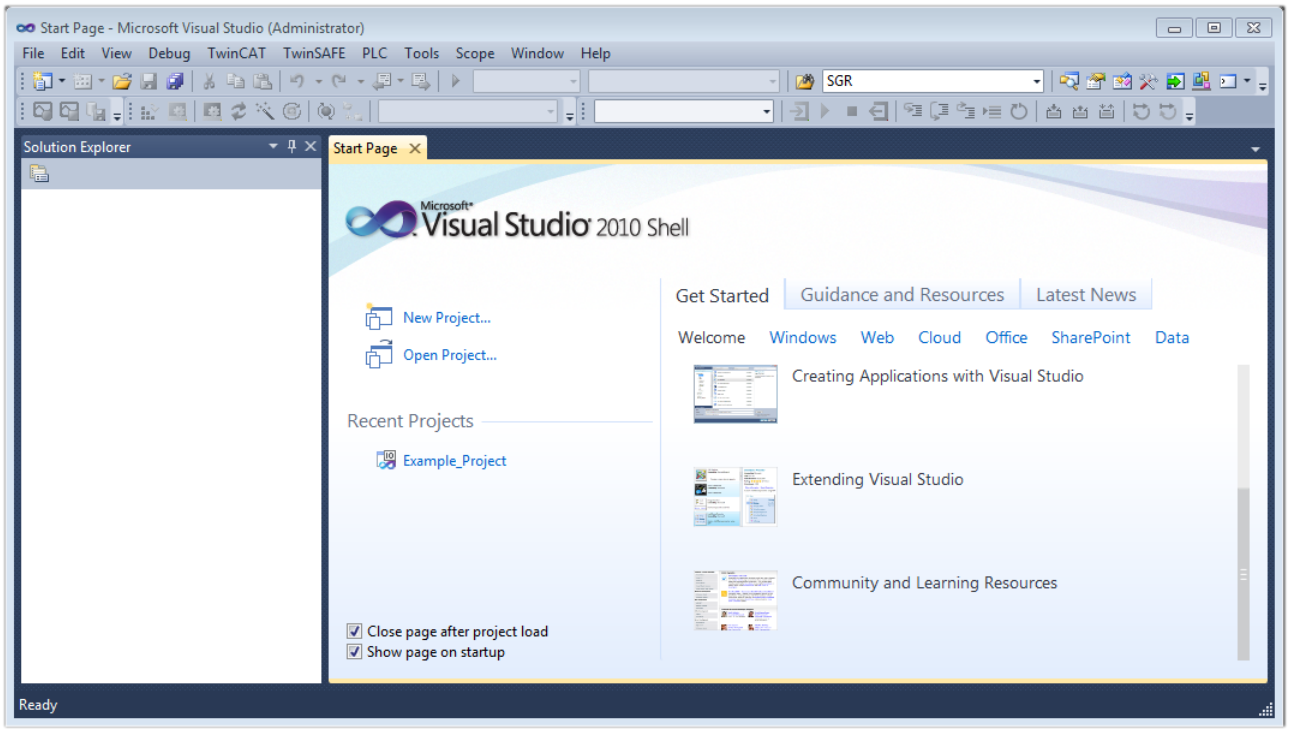


Fig. 48: Initial TwinCAT 3 user interface

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):

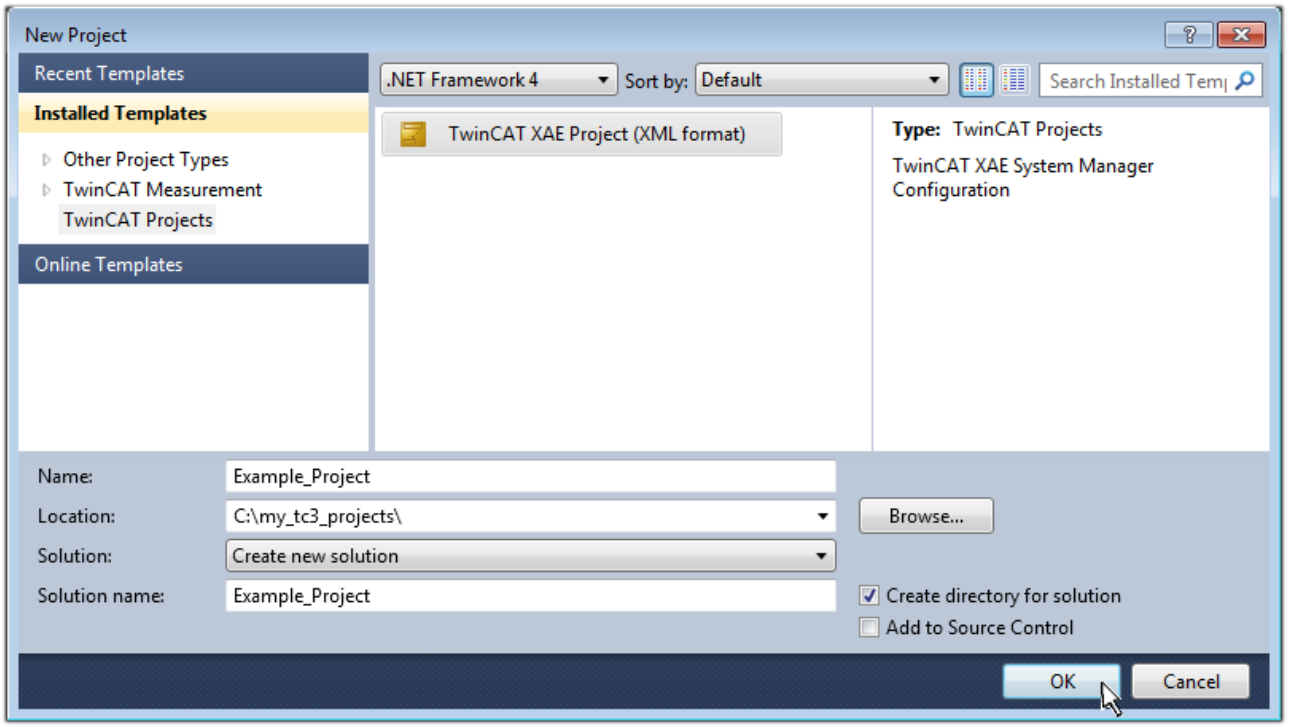


Fig. 49: Create new TwinCAT 3 project

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

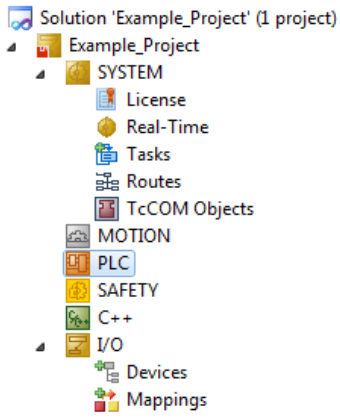
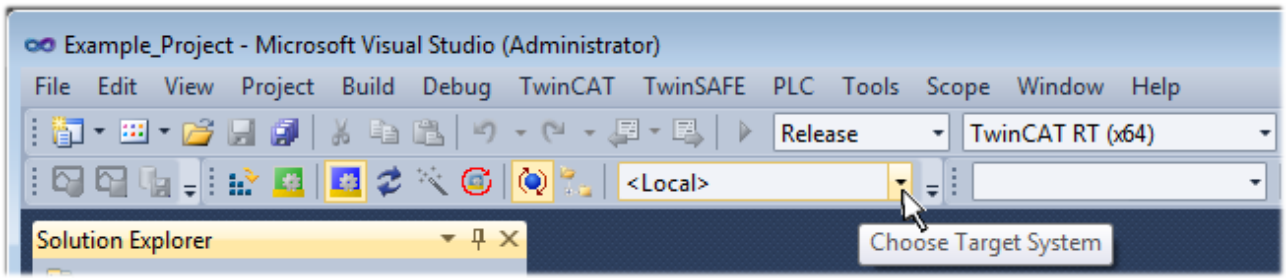


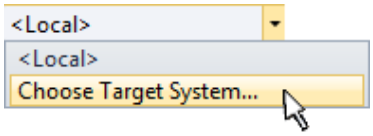
Fig. 50: New TwinCAT 3 project in the project folder explorer

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 [▶ 72]".

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:

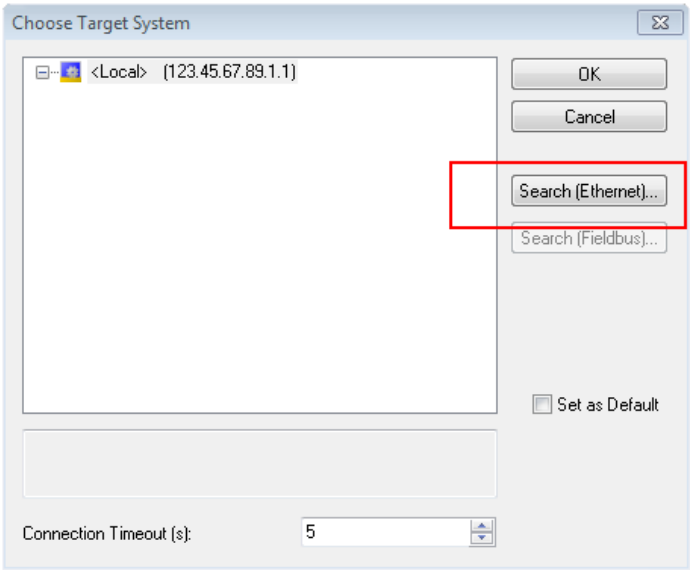


Fig. 51: Selection dialog: Choose the target system

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

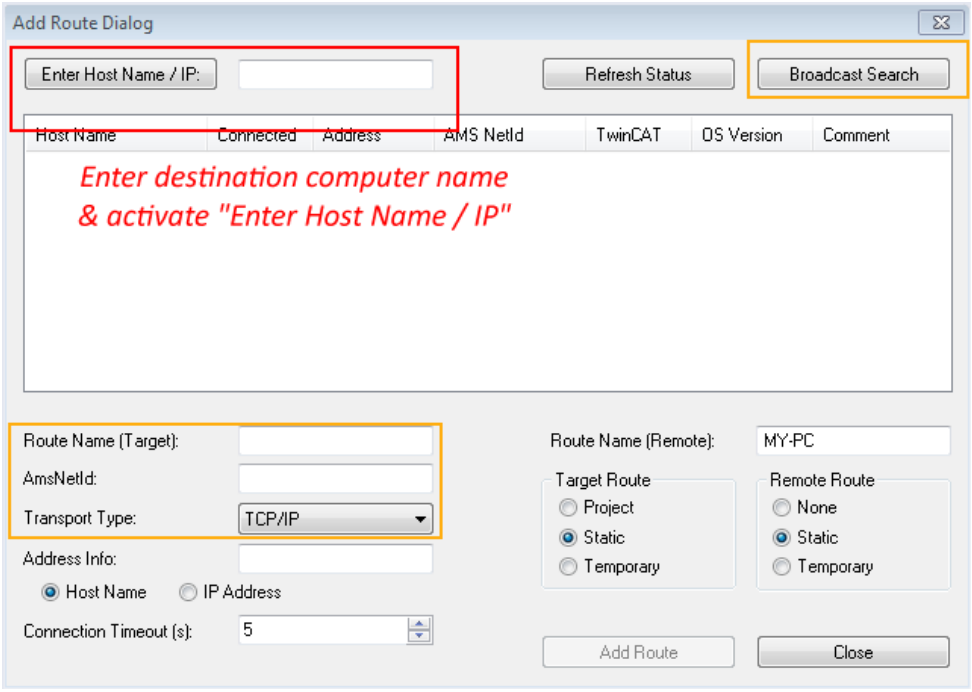
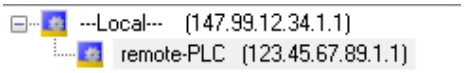


Fig. 52: specify the PLC for access by the TwinCAT System Manager: selection of the target system

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)”.

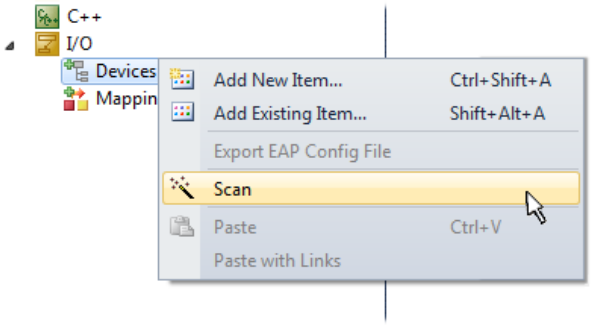


Fig. 53: Select “Scan”

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

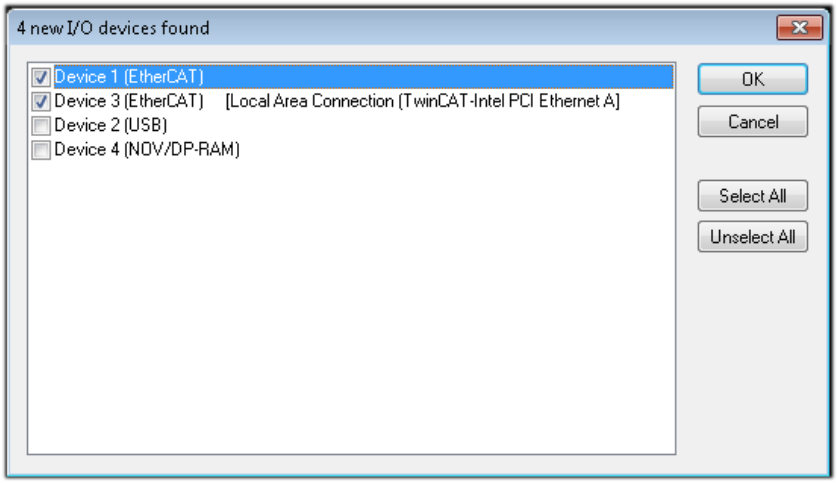


Fig. 54: Automatic detection of I/O devices: selection of the devices to be integrated

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 [▶ 57] described at the beginning of this section, the result is as follows:

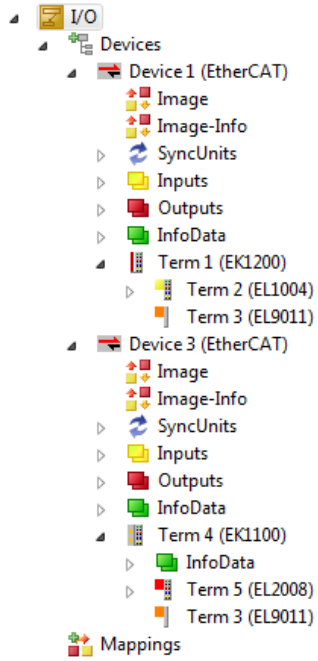


Fig. 55: Mapping of the configuration in VS shell of the TwinCAT 3 environment

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:

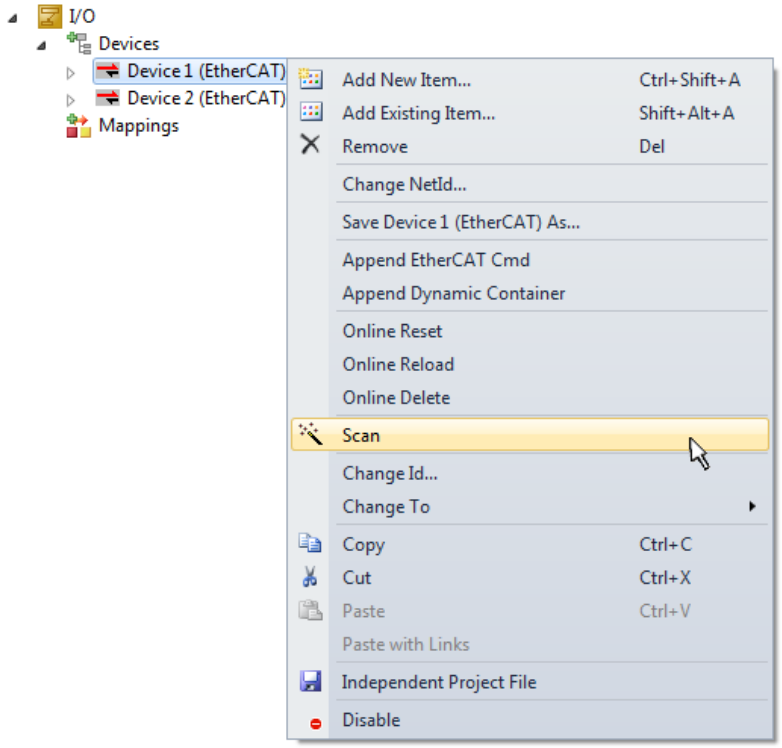


Fig. 56: Reading of individual terminals connected to a device

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….”:

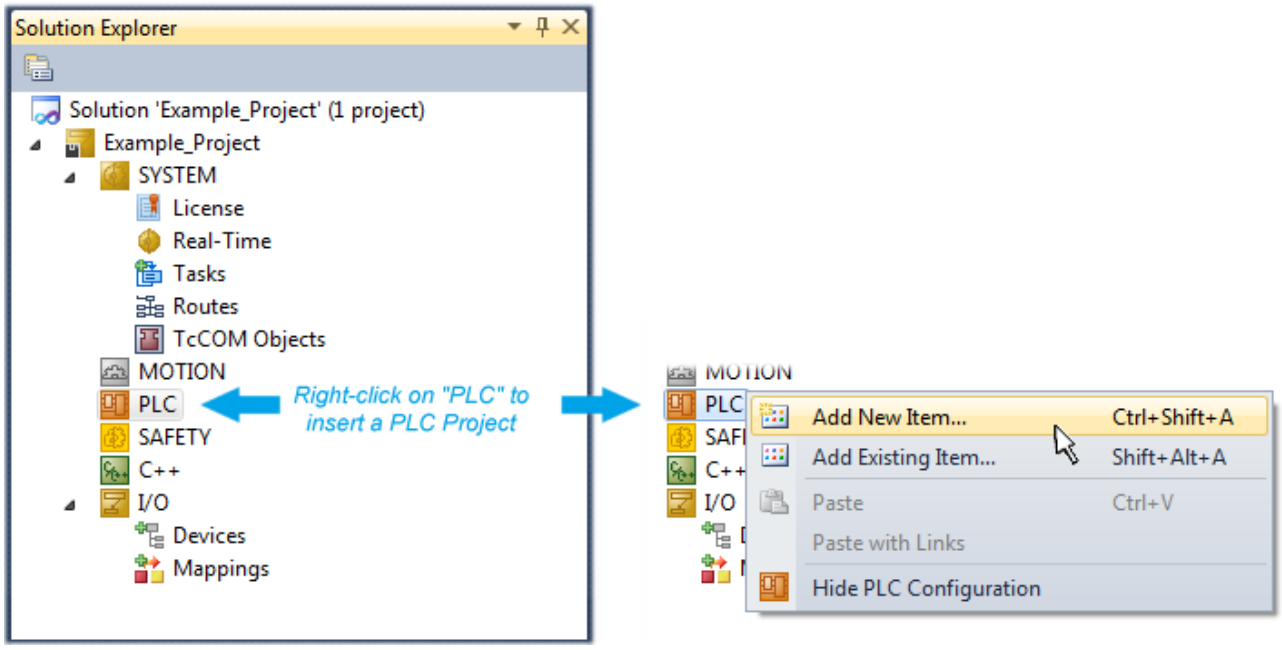


Fig. 57: Adding the programming environment in “PLC”

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

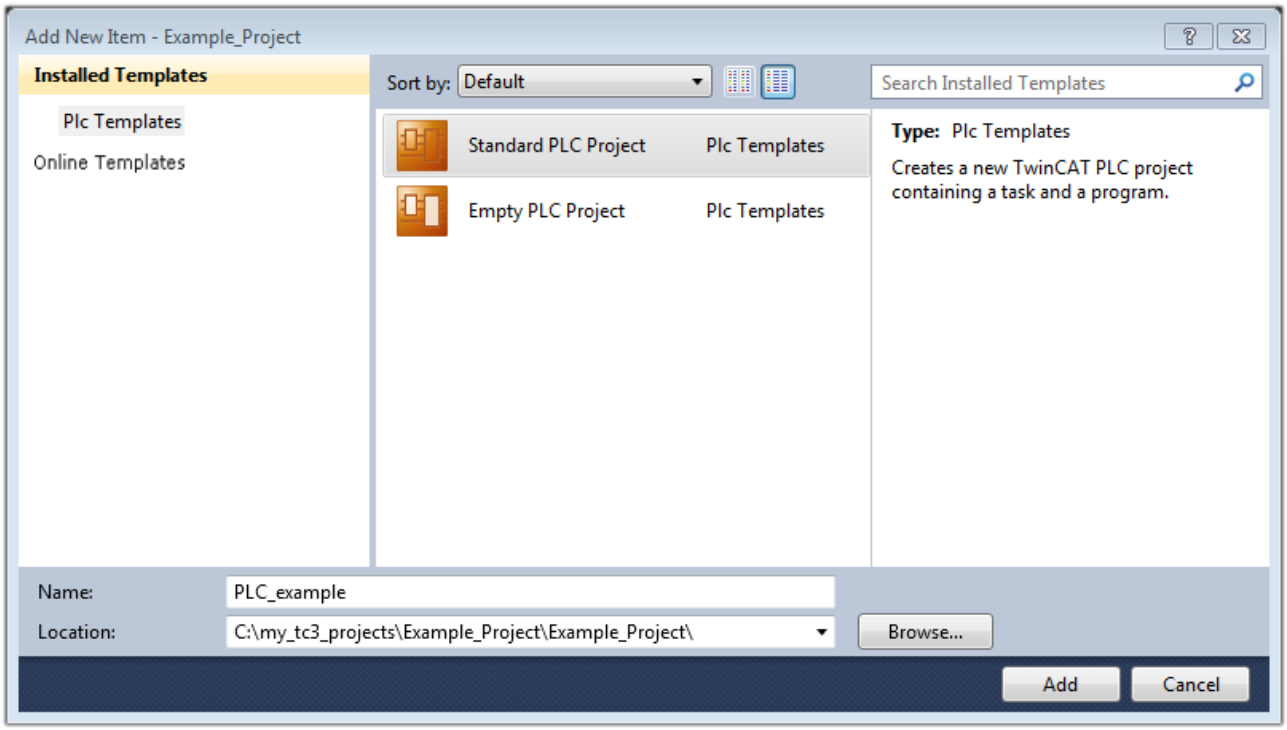


Fig. 58: Specifying the name and directory for the PLC programming environment

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:

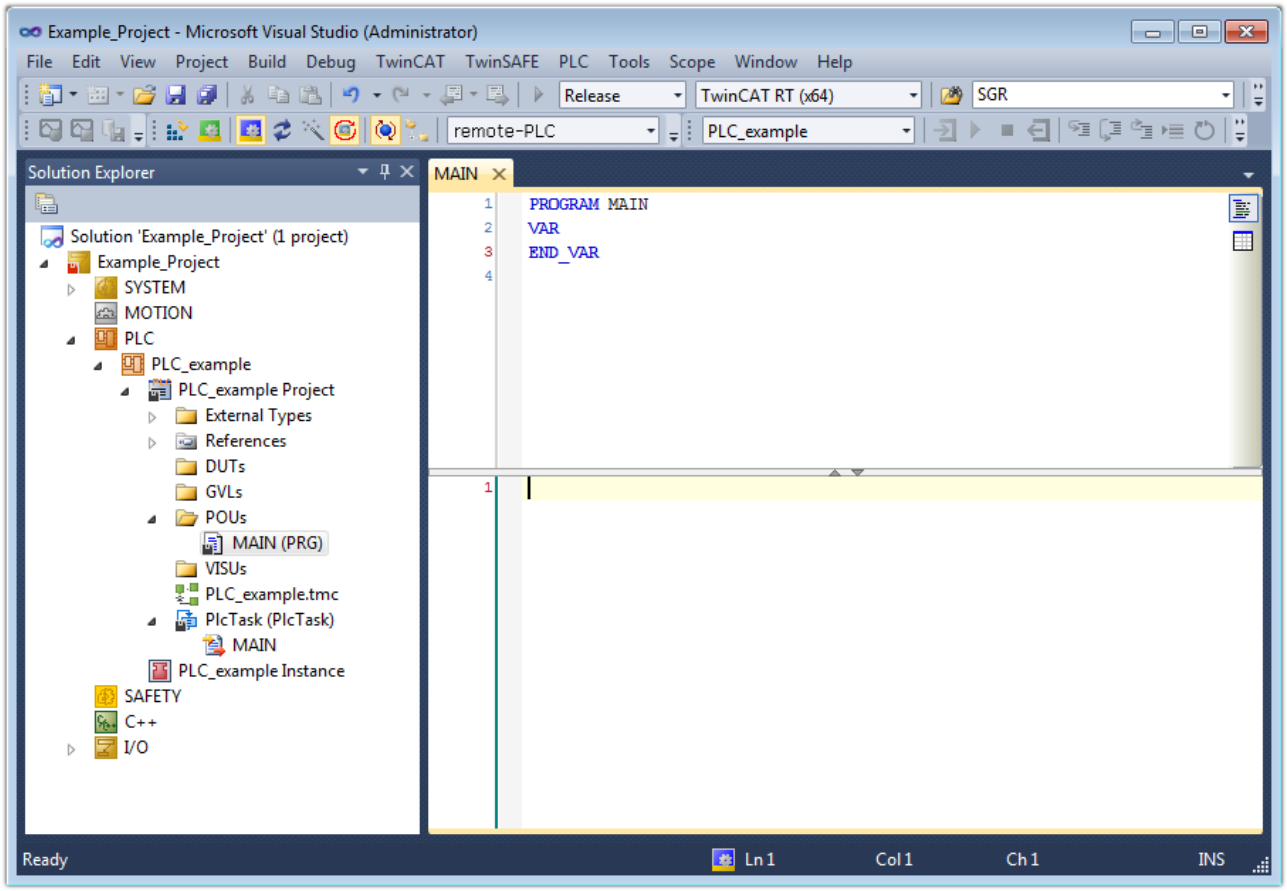


Fig. 59: Initial “Main” program for the standard PLC project

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

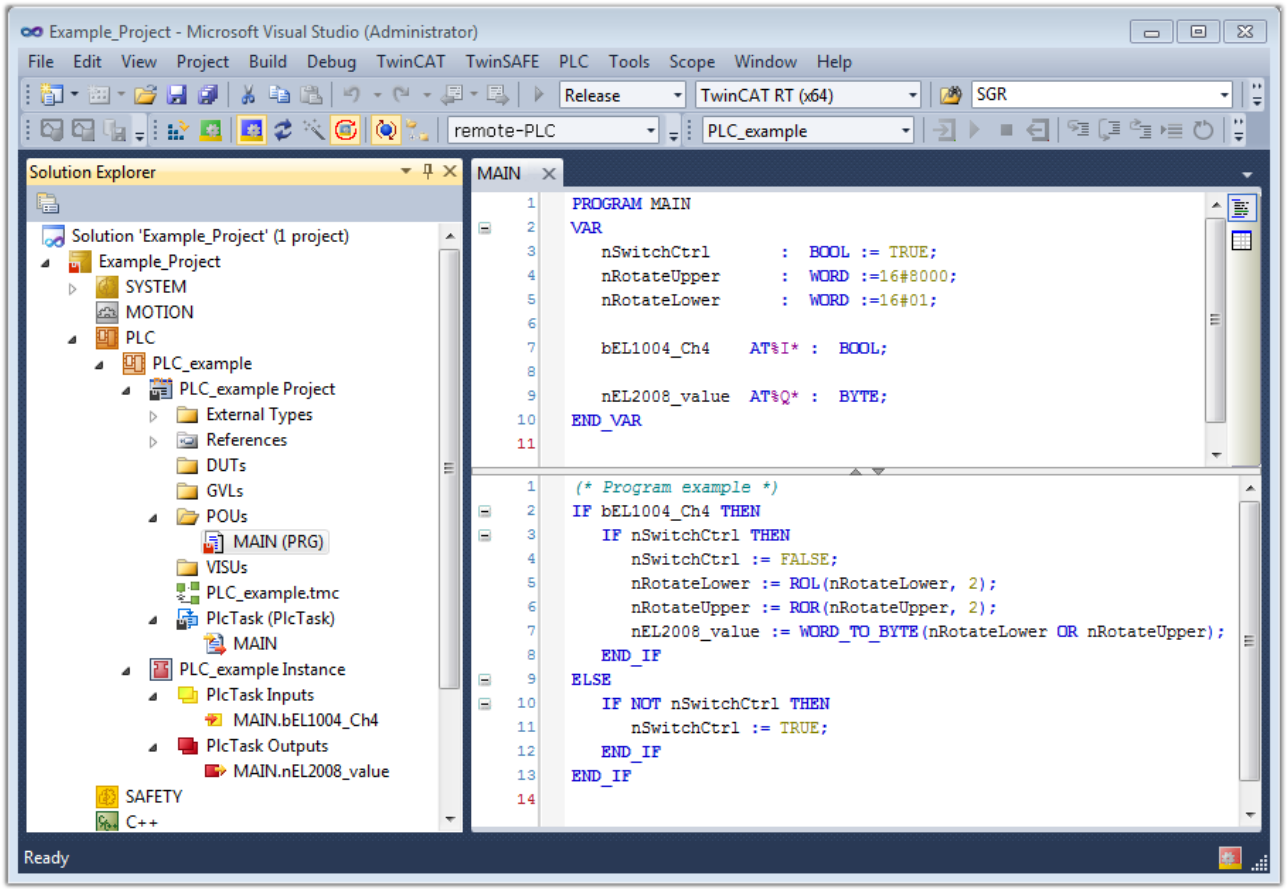


Fig. 60: Example program with variables after a compile process (without variable integration)

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

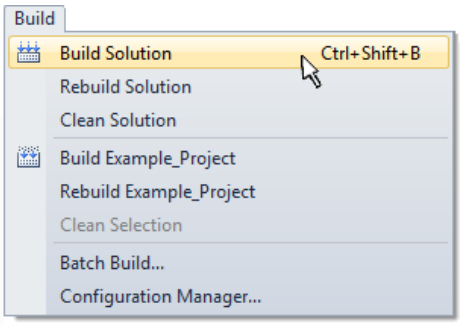
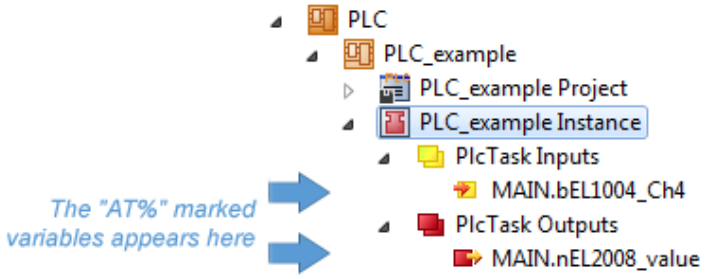


Fig. 61: Start program compilation

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:

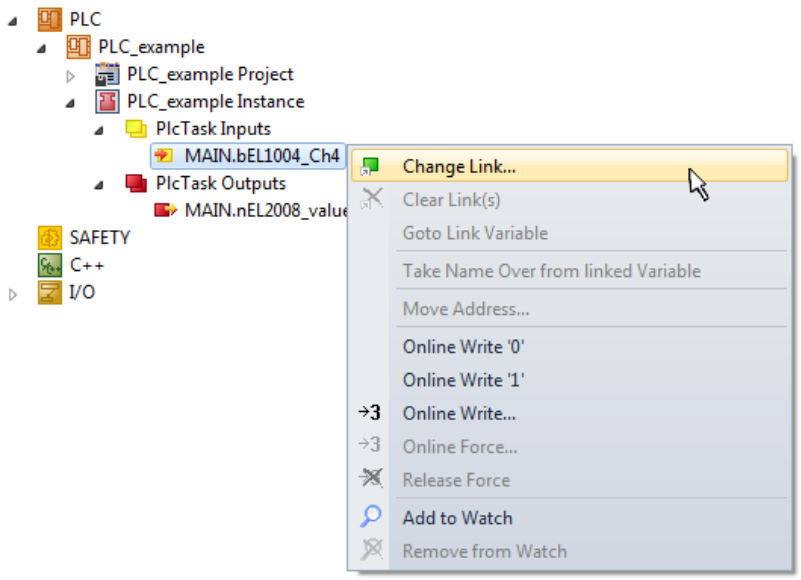


Fig. 62: Creating the links between PLC variables and process objects

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

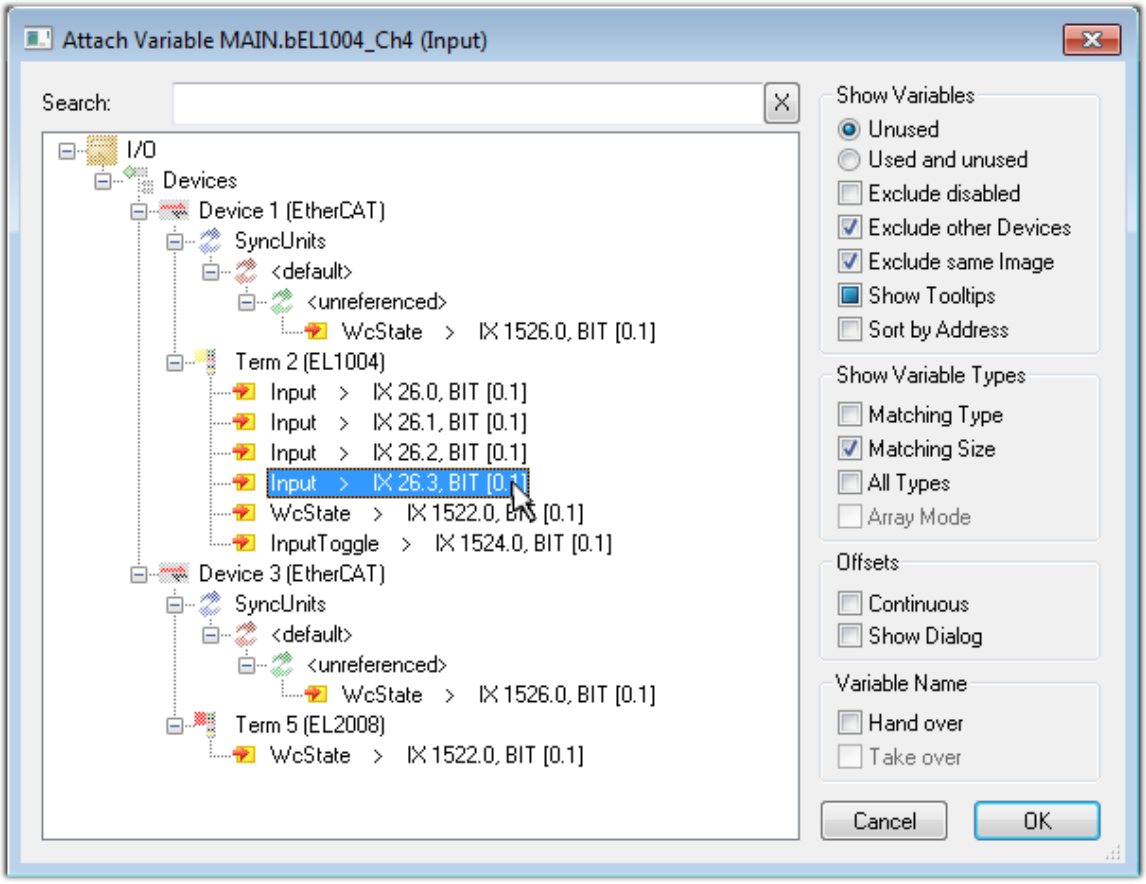


Fig. 63: Selecting BOOL-type PDO

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:

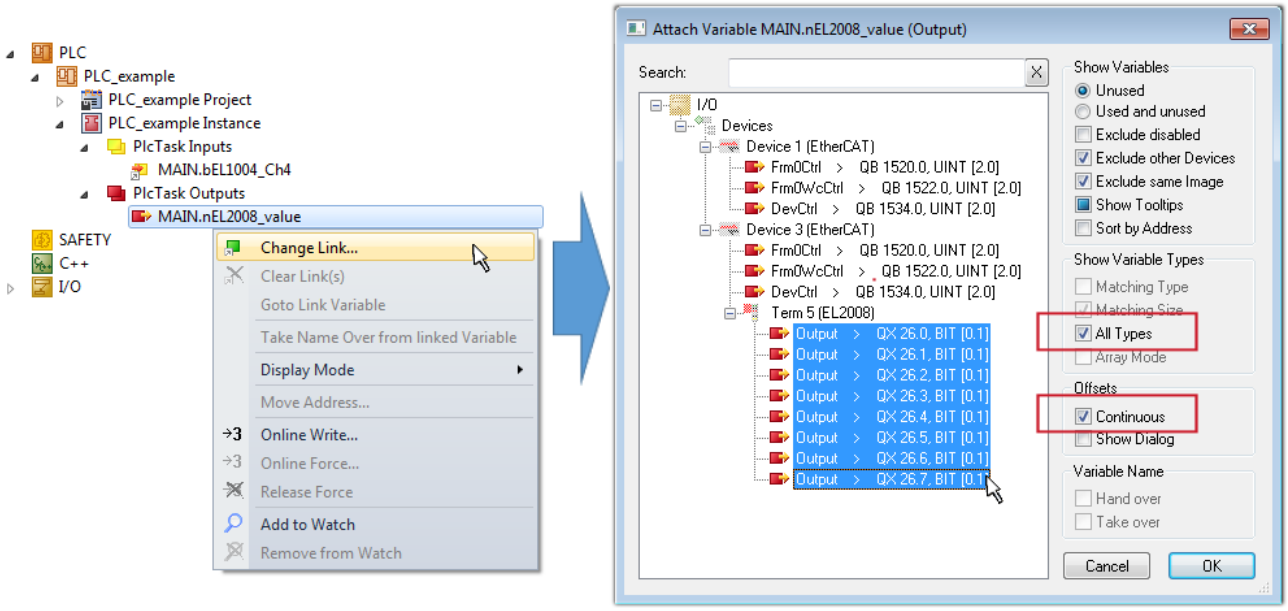


Fig. 64: Selecting several PDOs simultaneously: activate “Continuous” and “All types”

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:

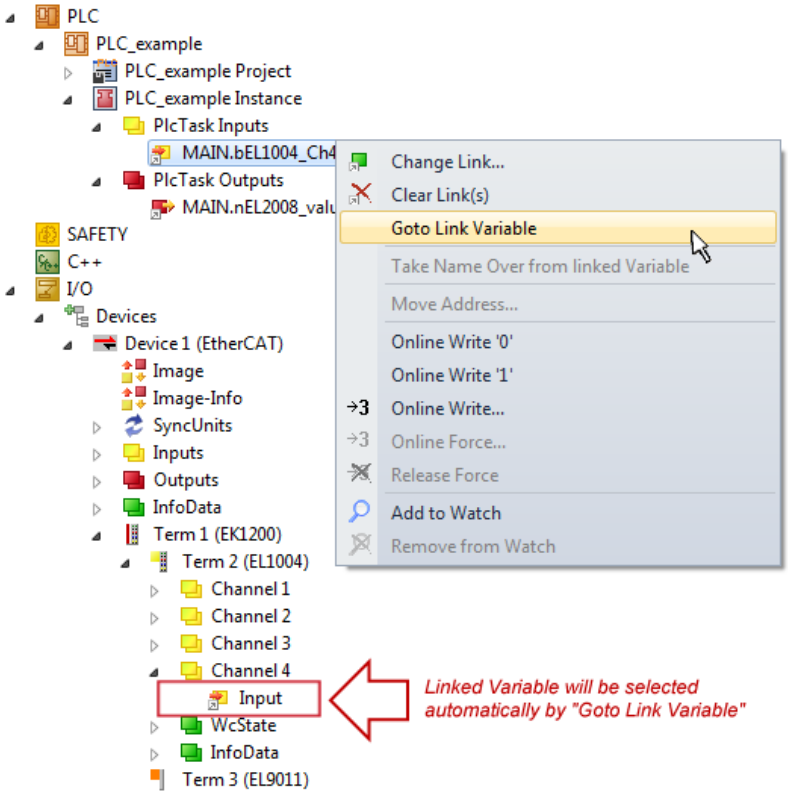


Fig. 65: Application of a “Goto Link Variable”, using “MAIN.bEL1004_Ch4” as an example

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.

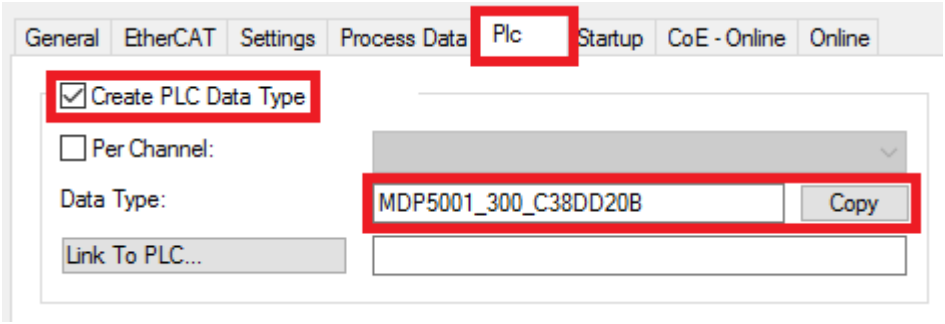


Fig. 66: Creating a PLC data type

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

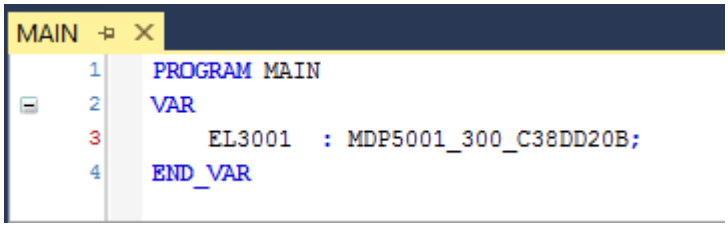


Fig. 67: Instance_of_struct

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.

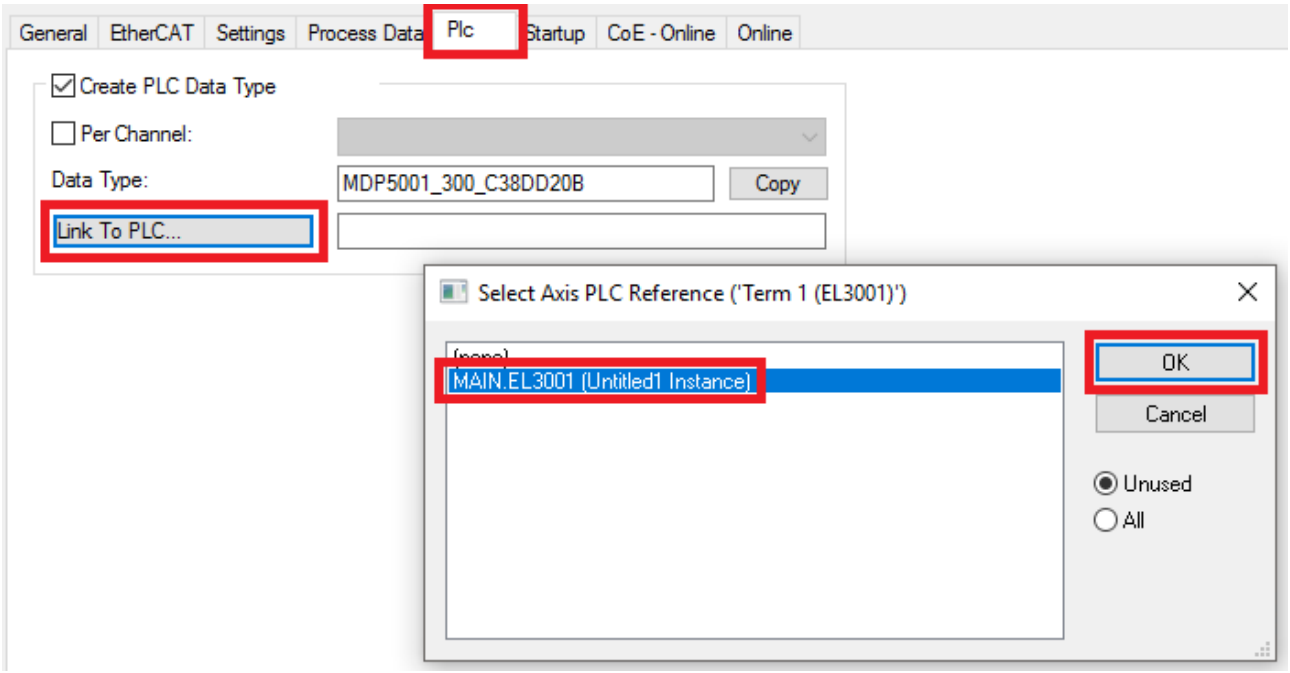


Fig. 68: Linking the structure

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

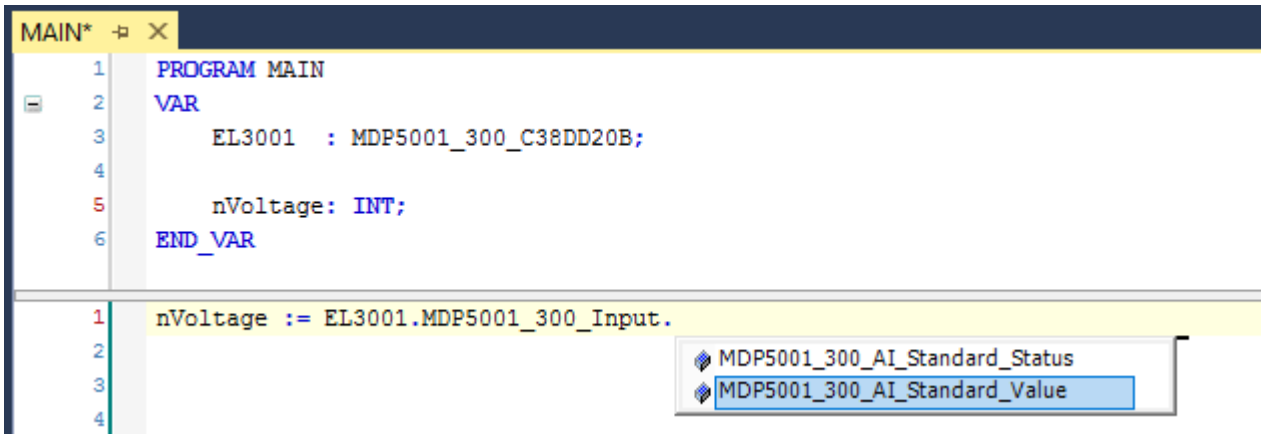
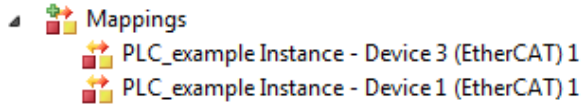


Fig. 69: Reading a variable from the structure of the process data

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:

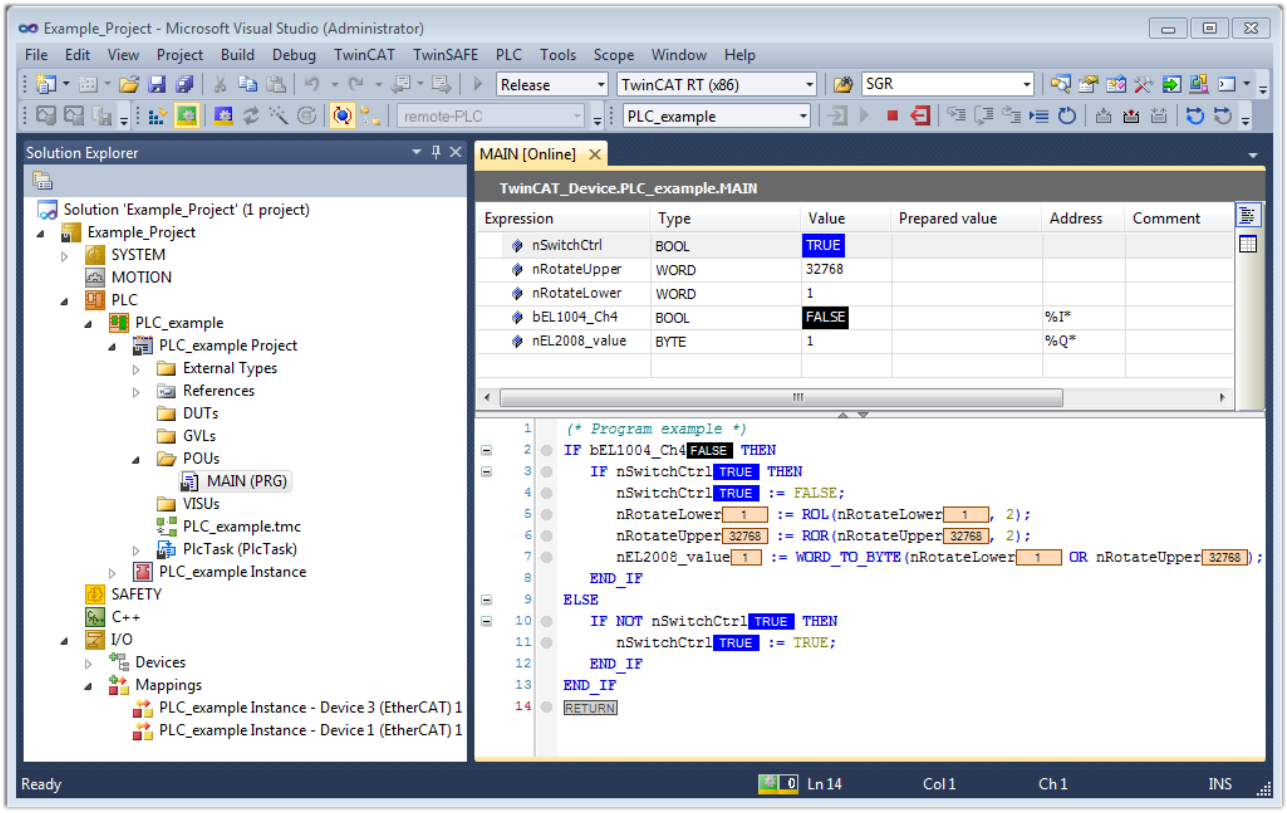


Fig. 70: TwinCAT 3 development environment (VS shell): logged-in, after program startup

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).

8.2 TwinCAT Development Environment

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

- TwinCAT 2: System Manager (Configuration) & 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.

8.2.1 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.

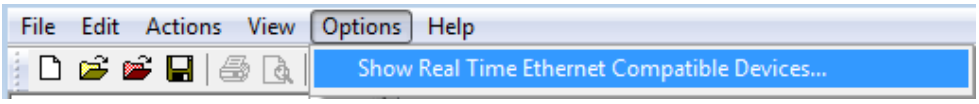


Fig. 71: System Manager “Options” (TwinCAT 2)

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

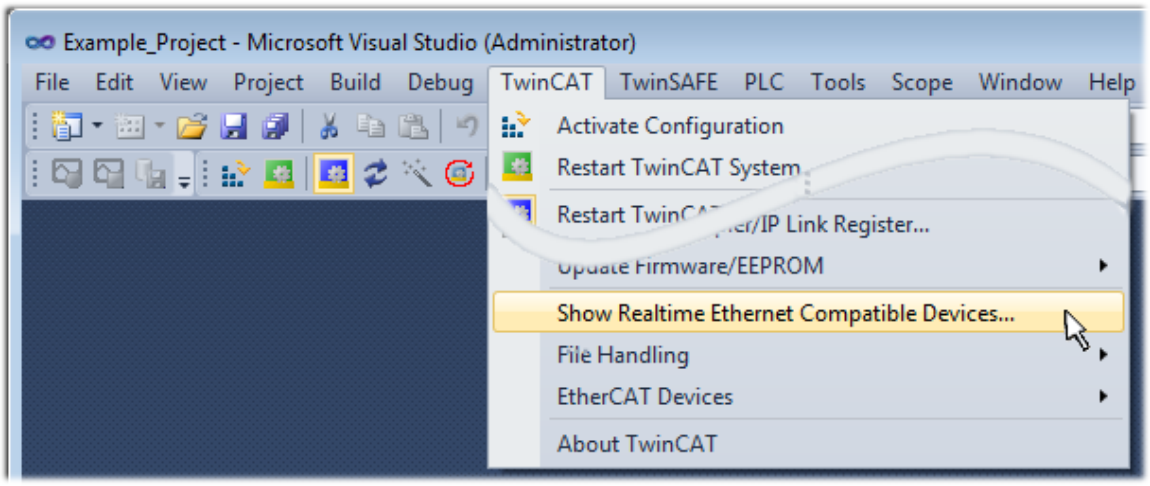


Fig. 72: Call up under VS Shell (TwinCAT 3)

B: Via TcRteInstall.exe in the TwinCAT directory

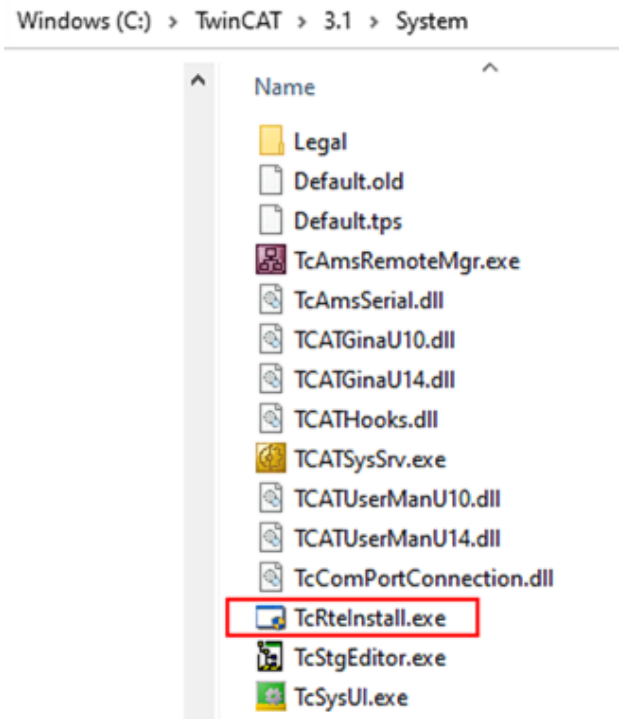


Fig. 73: TcRteInstall in the TwinCAT directory

In both cases, the following dialog appears:

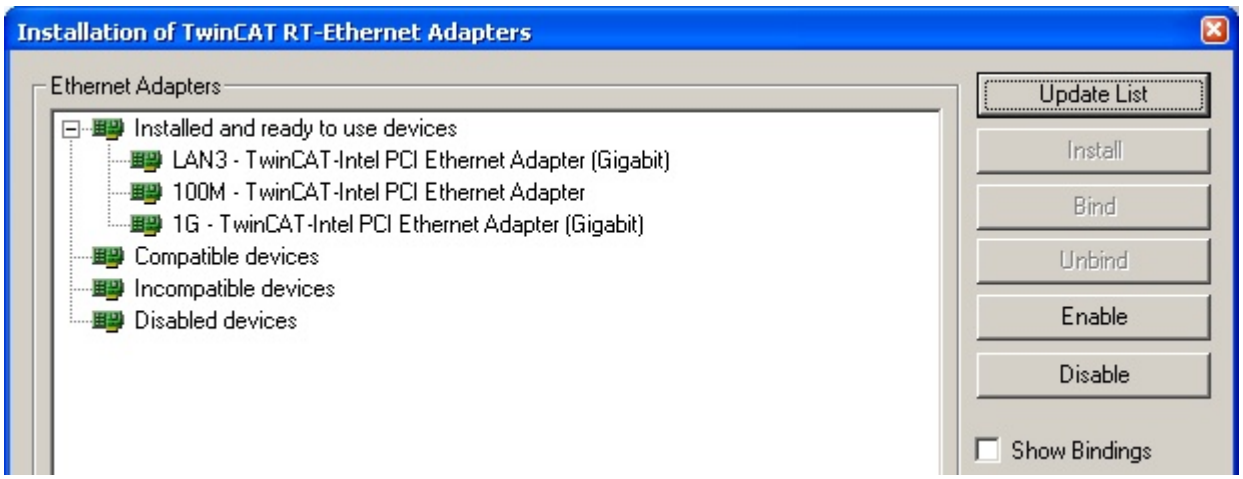


Fig. 74: Overview of network interfaces

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

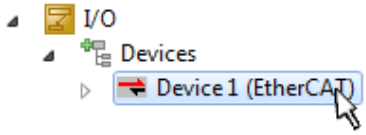
A Windows warning regarding the unsigned driver can be ignored.

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. 75: 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)

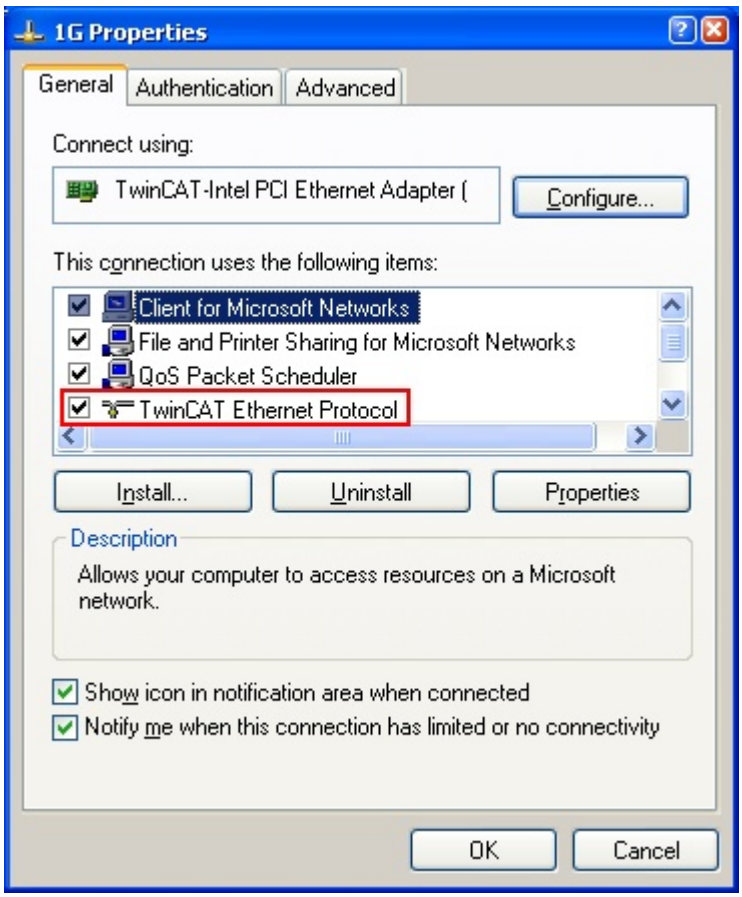


Fig. 76: Windows properties of the network interface

A correct setting of the driver could be:

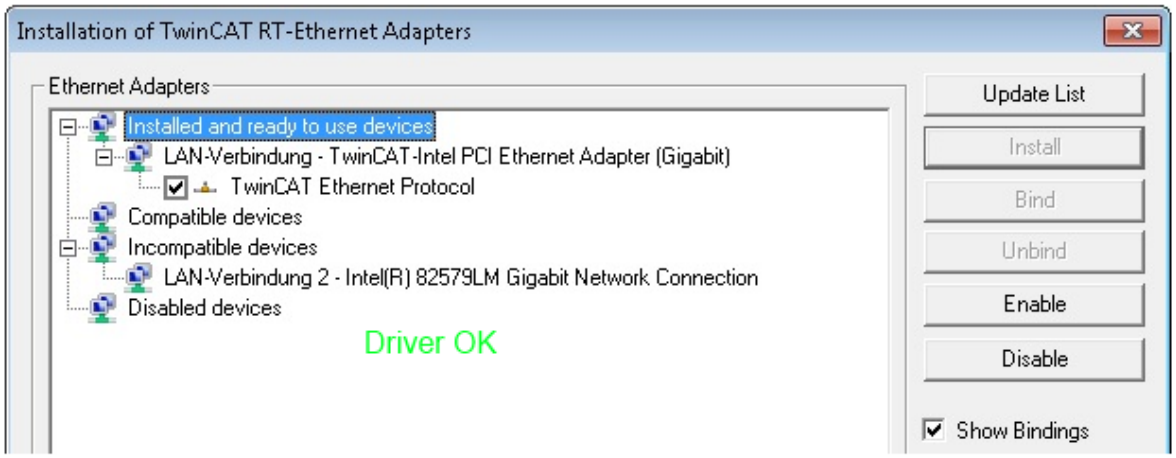


Fig. 77: Exemplary correct driver setting for the Ethernet port

Other possible settings have to be avoided:

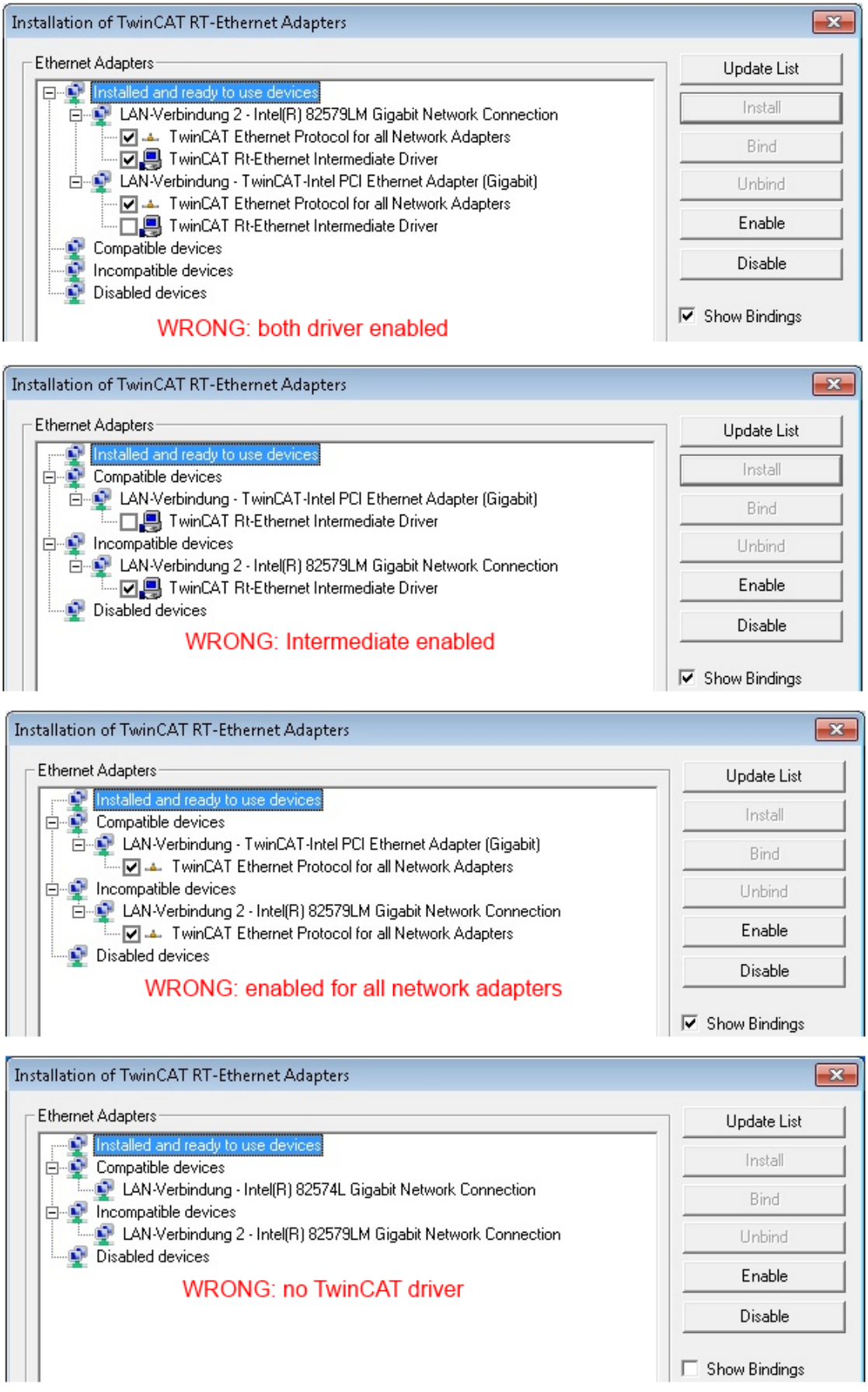


Fig. 78: Incorrect driver settings for the Ethernet port

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.

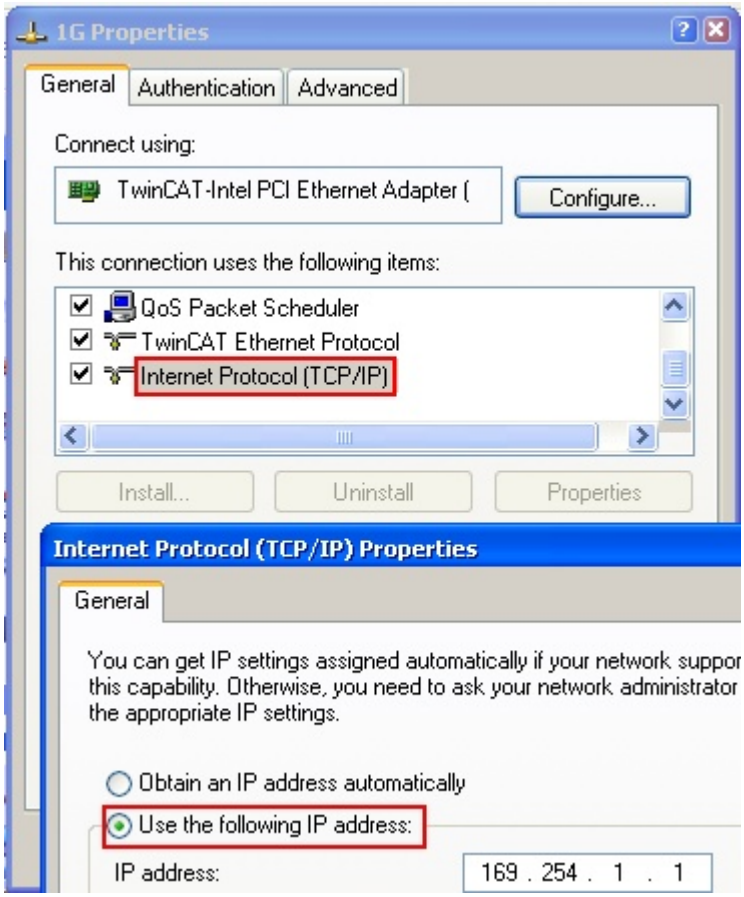


Fig. 79: TCP/IP setting for the Ethernet port

8.2.2 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 [▶ 93] 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”

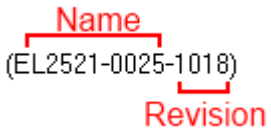


Fig. 80: Identifier structure

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 [▶ 11].

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.

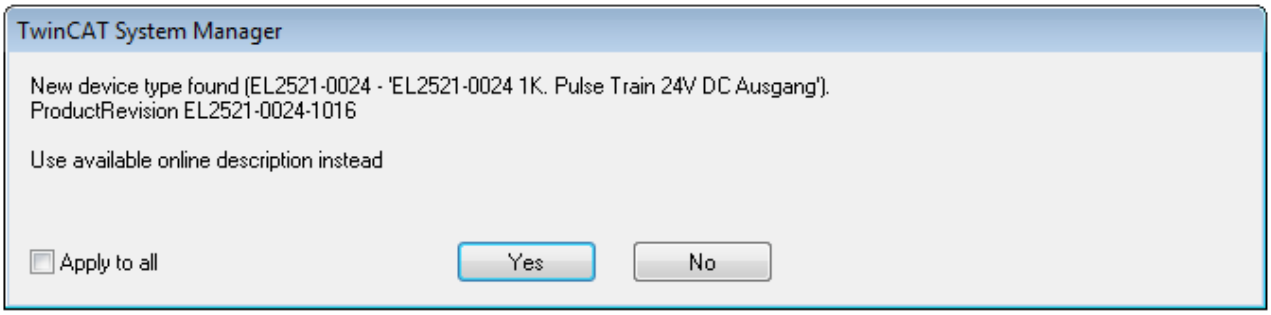


Fig. 81: OnlineDescription information window (TwinCAT 2)

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

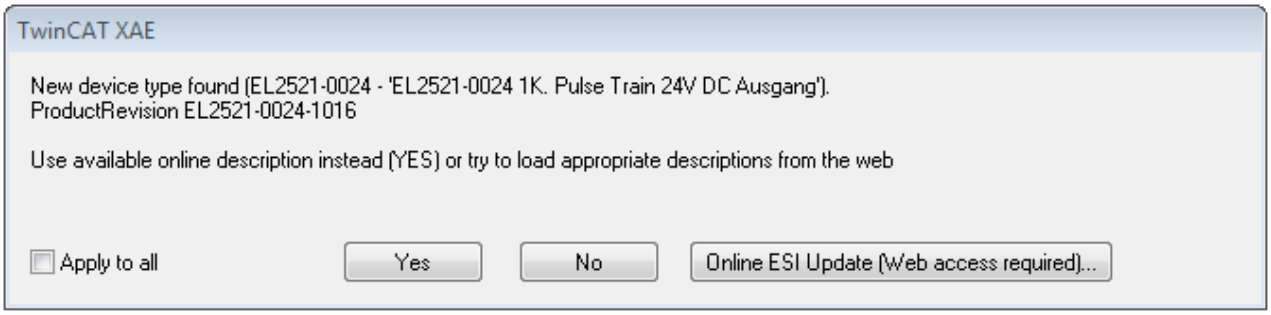


Fig. 82: Information window OnlineDescription (TwinCAT 3)

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. 83: 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)*.

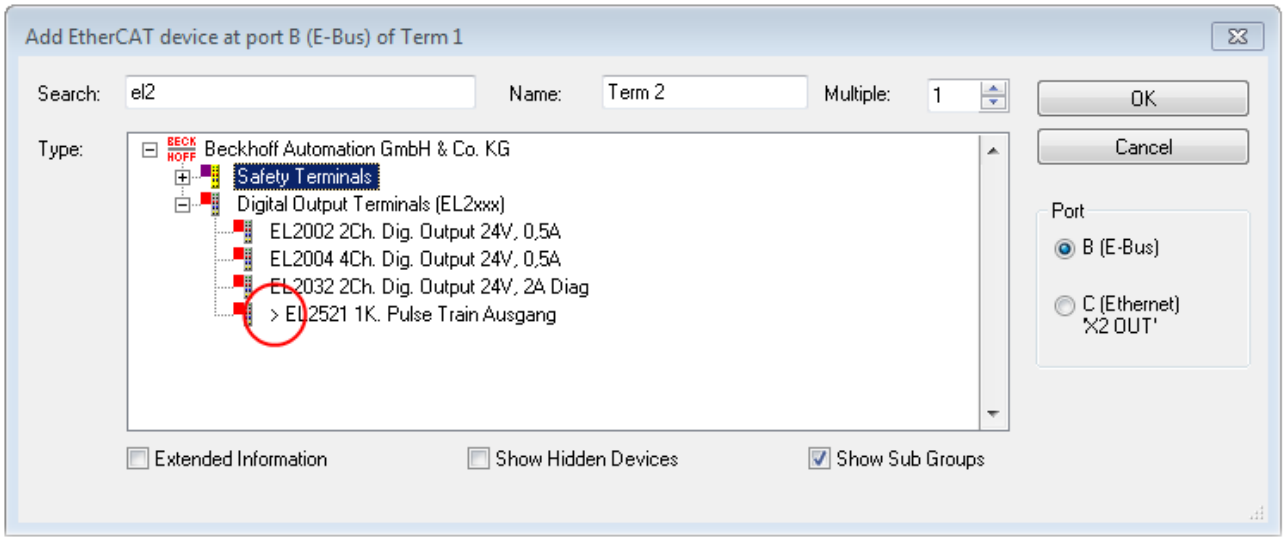


Fig. 84: Indication of an online recorded ESI of EL2521 as an example

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.

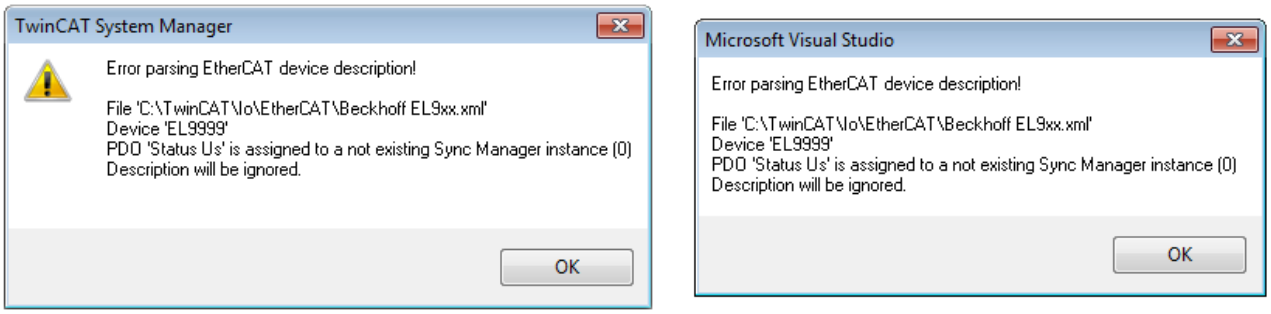


Fig. 85: Information window for faulty ESI file (left: TwinCAT 2; right: TwinCAT 3)

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

8.2.3 TwinCAT ESI Updater

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

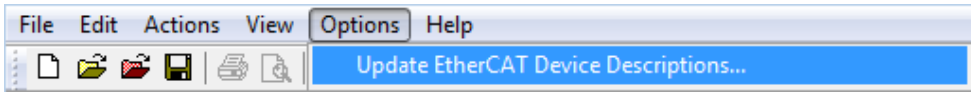


Fig. 86: Using the ESI Updater (>= TwinCAT 2.11)

The call up takes place under:
“Options” → “Update EtherCAT Device Descriptions”

Selection under TwinCAT 3:

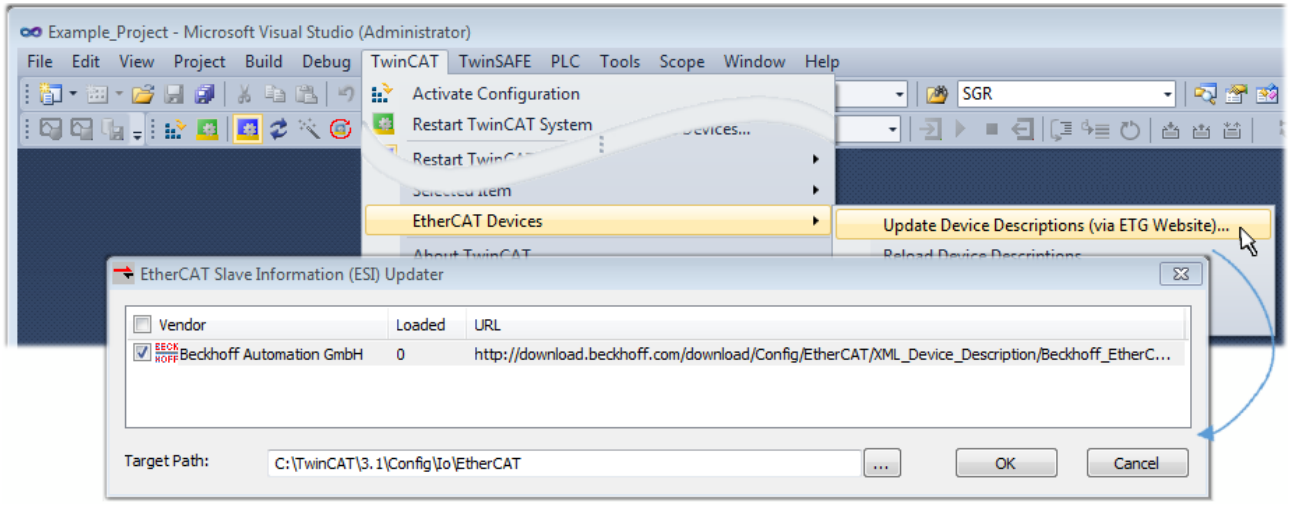


Fig. 87: Using the ESI Updater (TwinCAT 3)

The ESI Updater (TwinCAT 3) 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.

The call up takes place under:
“TwinCAT” → “EtherCAT Devices” → “Update Device Description (via ETG Website)…”.

8.2.4 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” [▶ 89].

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 [▶ 99] (Ethernet port at the IPC)
- detecting the connected EtherCAT devices [▶ 100]. This step can be carried out independent of the preceding step
- troubleshooting [▶ 103]

The scan with existing configuration [▶ 104] can also be carried out for comparison.

8.2.5 OFFLINE configuration creation

Creating the EtherCAT device

Create an EtherCAT device in an empty System Manager window.

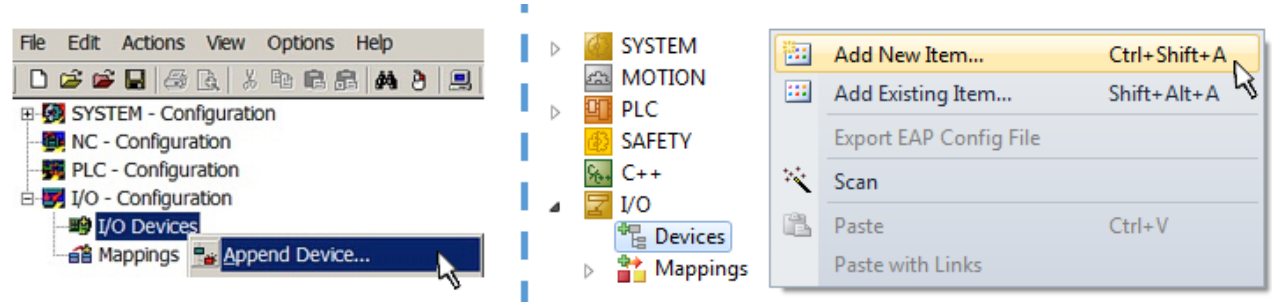


Fig. 88: Append EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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”.

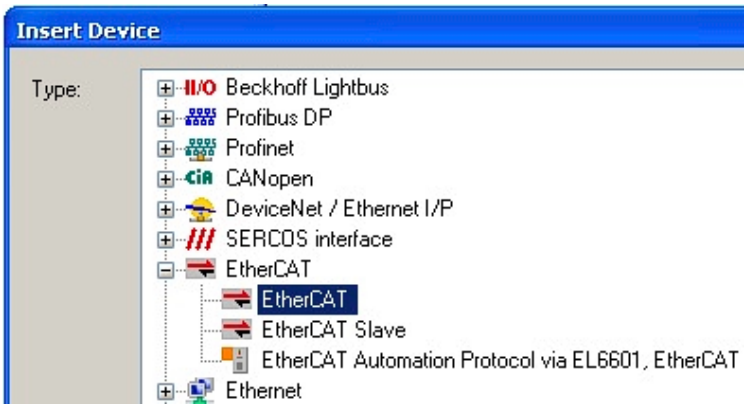


Fig. 89: Selecting the EtherCAT connection (TwinCAT 2.11, TwinCAT 3)

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

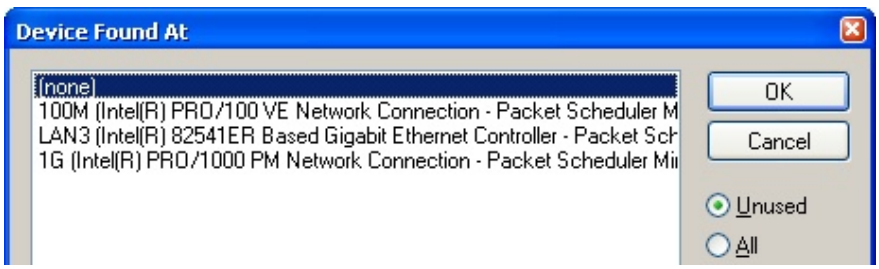


Fig. 90: Selecting the Ethernet port

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)”.

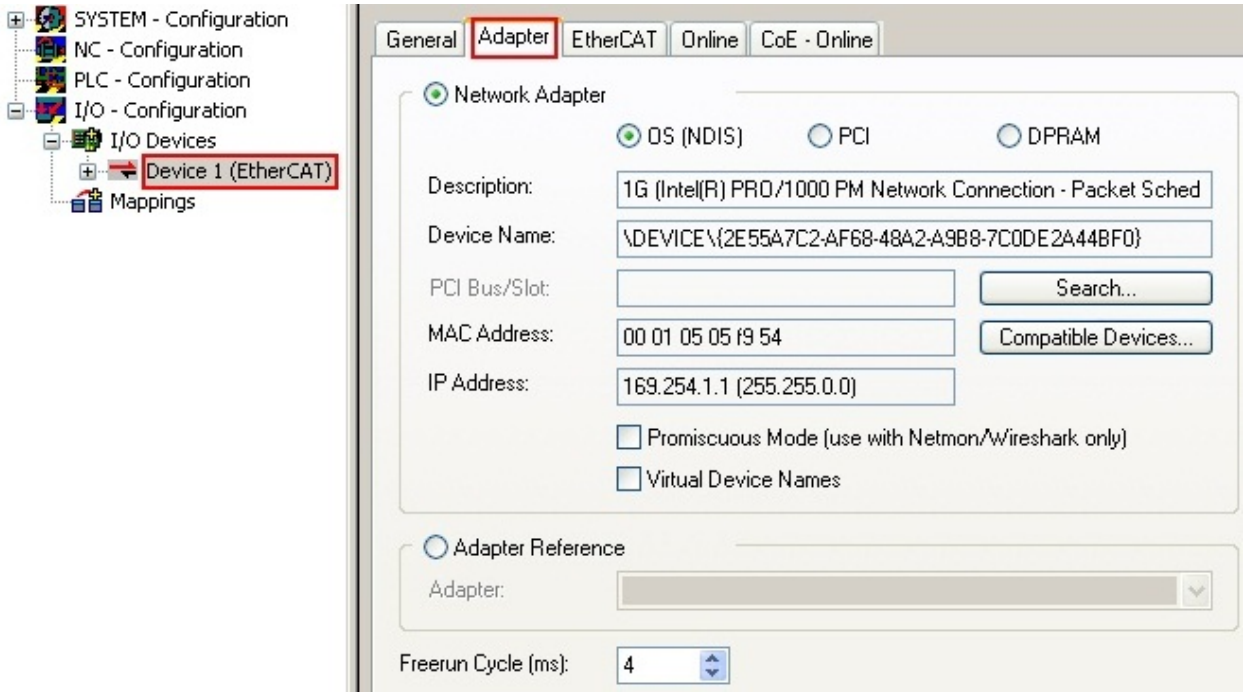
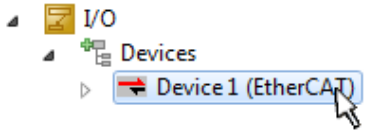


Fig. 91: EtherCAT device properties (TwinCAT 2)

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 [▶ 83].

Defining EtherCAT slaves

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

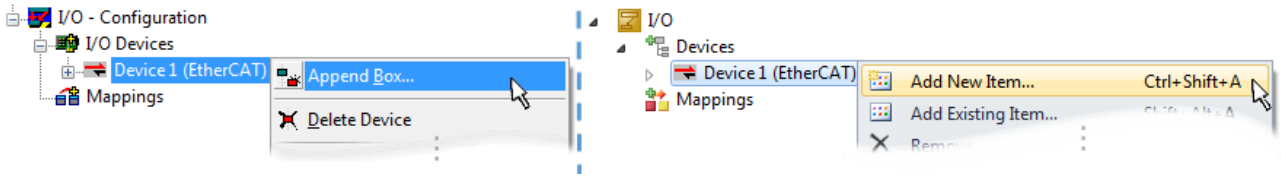


Fig. 92: Appending EtherCAT devices (left: TwinCAT 2; right: TwinCAT 3)

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 (since TwinCAT 2.11 or TwinCAT 3).

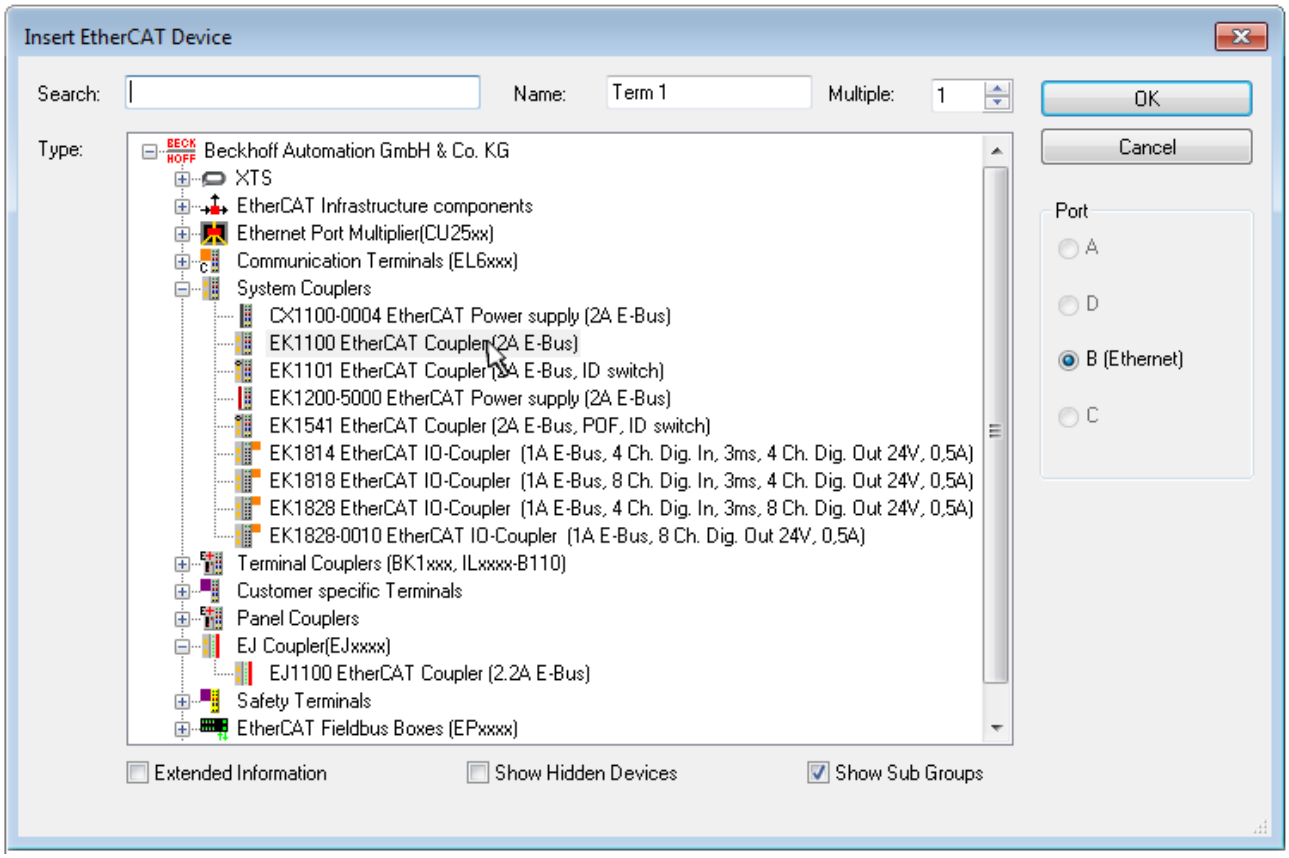


Fig. 93: Selection dialog for new EtherCAT device

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”.

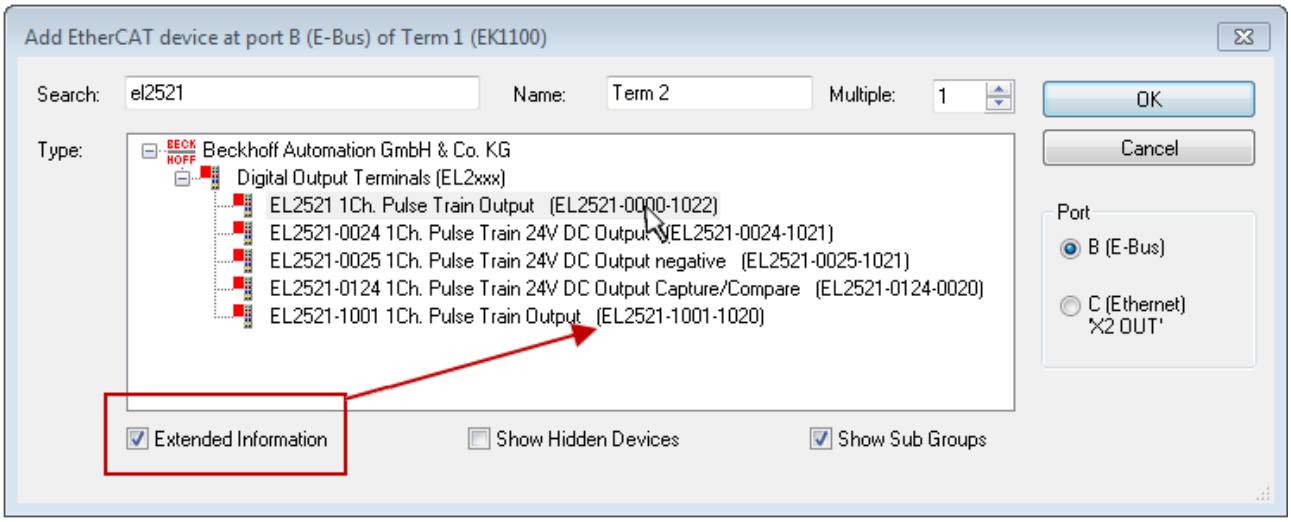


Fig. 94: Display of device revision

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”.

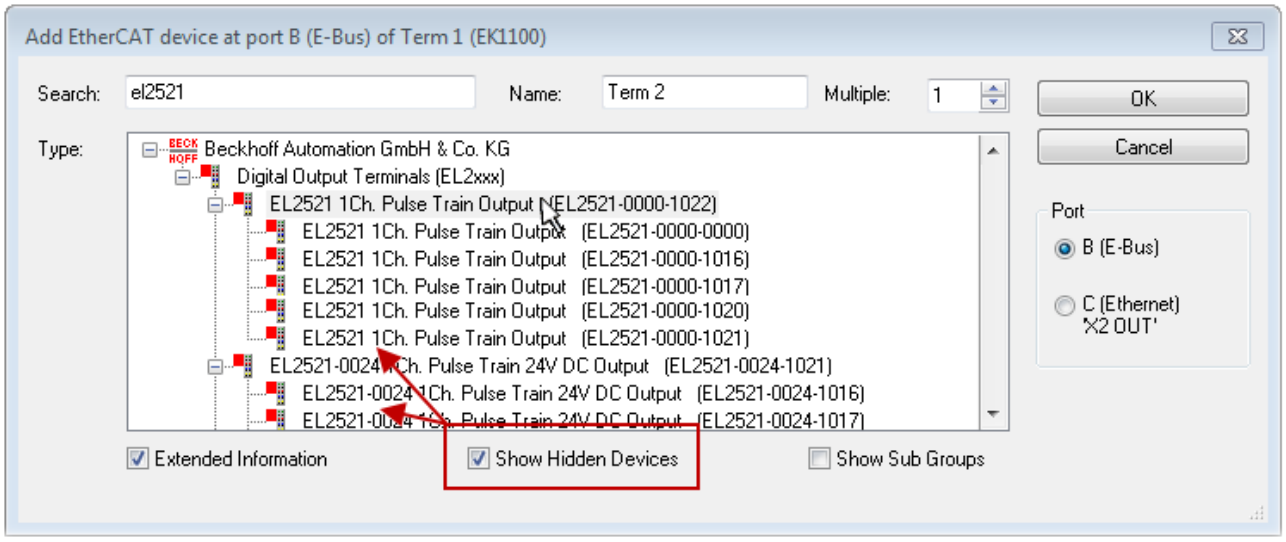


Fig. 95: Display of previous revisions

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.

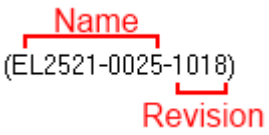


Fig. 96: Name/revision of the terminal

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, ...

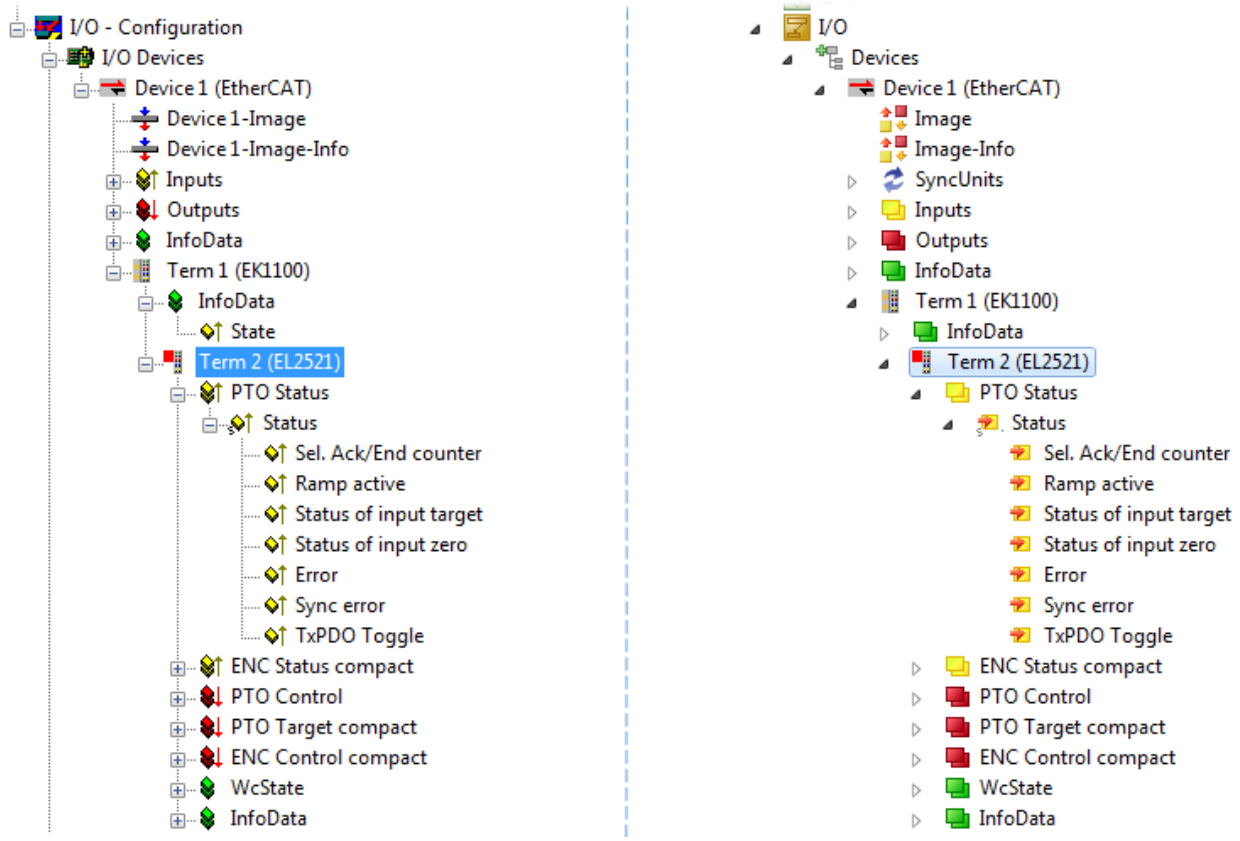


Fig. 97: EtherCAT terminal in the TwinCAT tree (left: TwinCAT 2; right: TwinCAT 3)

8.2.6 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.

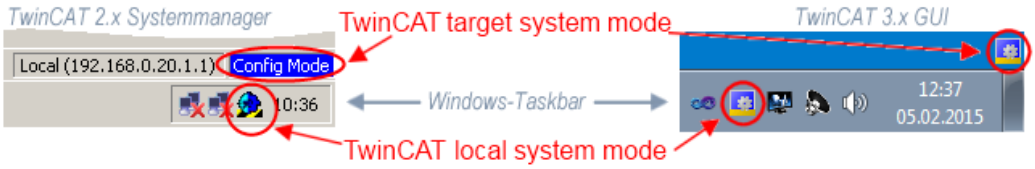


Fig. 98: Differentiation local/target system (left: TwinCAT 2; right: TwinCAT 3)

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

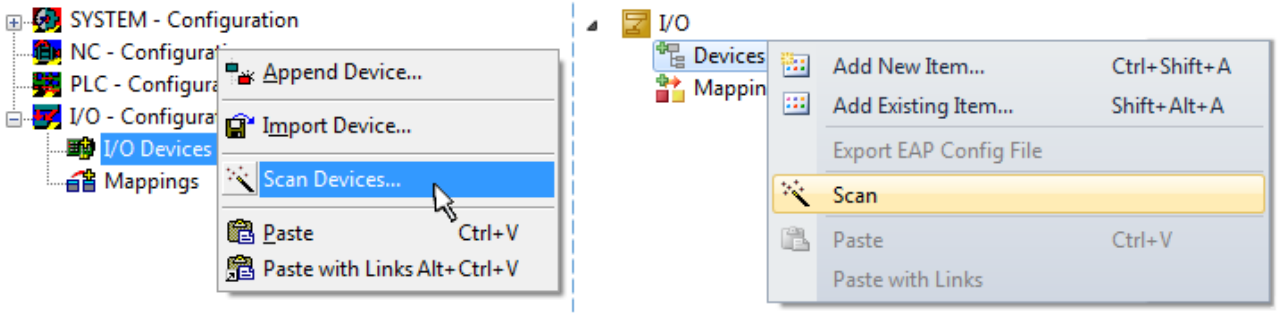


Fig. 99: Scan Devices (left: TwinCAT 2; right: TwinCAT 3)

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.

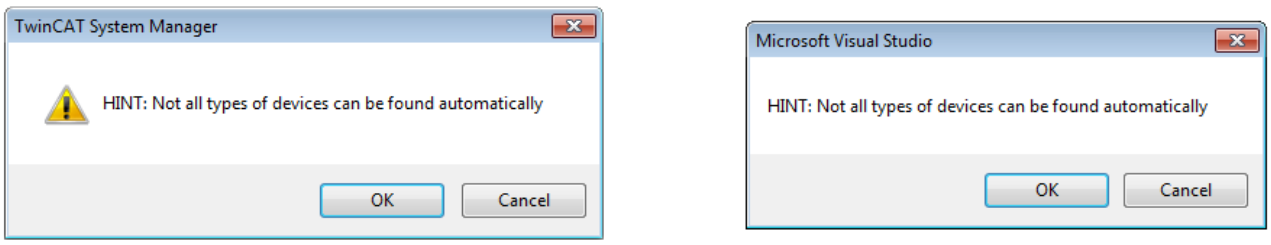


Fig. 100: Note for automatic device scan (left: TwinCAT 2; right: TwinCAT 3)

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” .

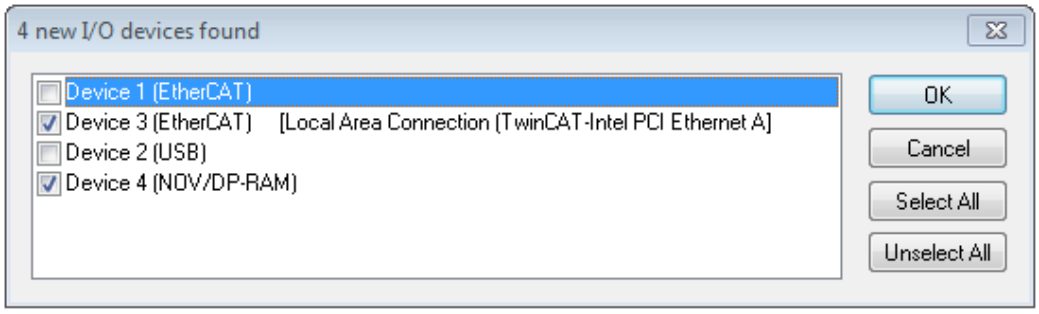


Fig. 101: Detected Ethernet devices

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 [▶ 83].

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.

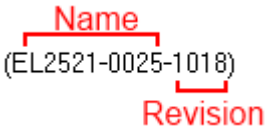


Fig. 102: Example default state

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 [▶ 104] 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:

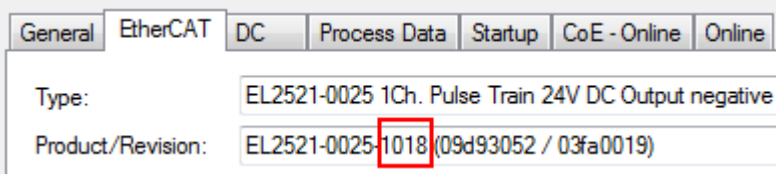


Fig. 103: Installing EthetCAT terminal with revision -1018

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 [▶ 104] 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:

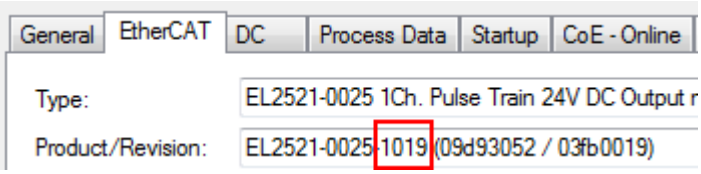


Fig. 104: Detection of EtherCAT terminal with revision -1019

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

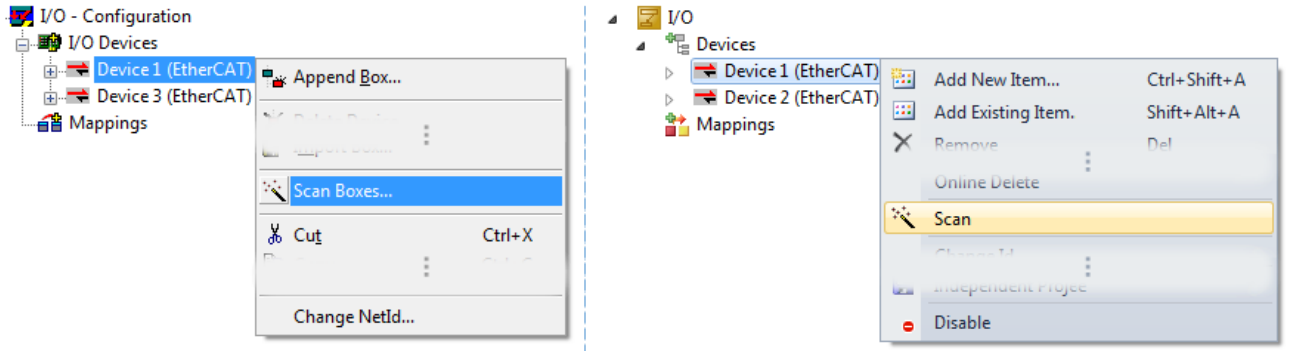


Fig. 106: Manual scanning for devices on a specified EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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.

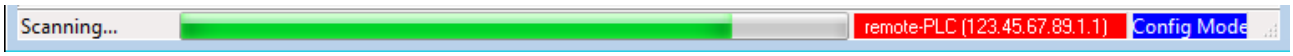


Fig. 107: *Scan progressexemplary by TwinCAT 2*

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



Fig. 108: 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).

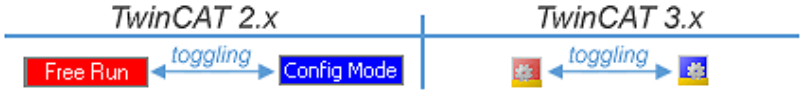


Fig. 109: Displaying of “Free Run” and “Config Mode” toggling right below in the status bar



Fig. 110: 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*.

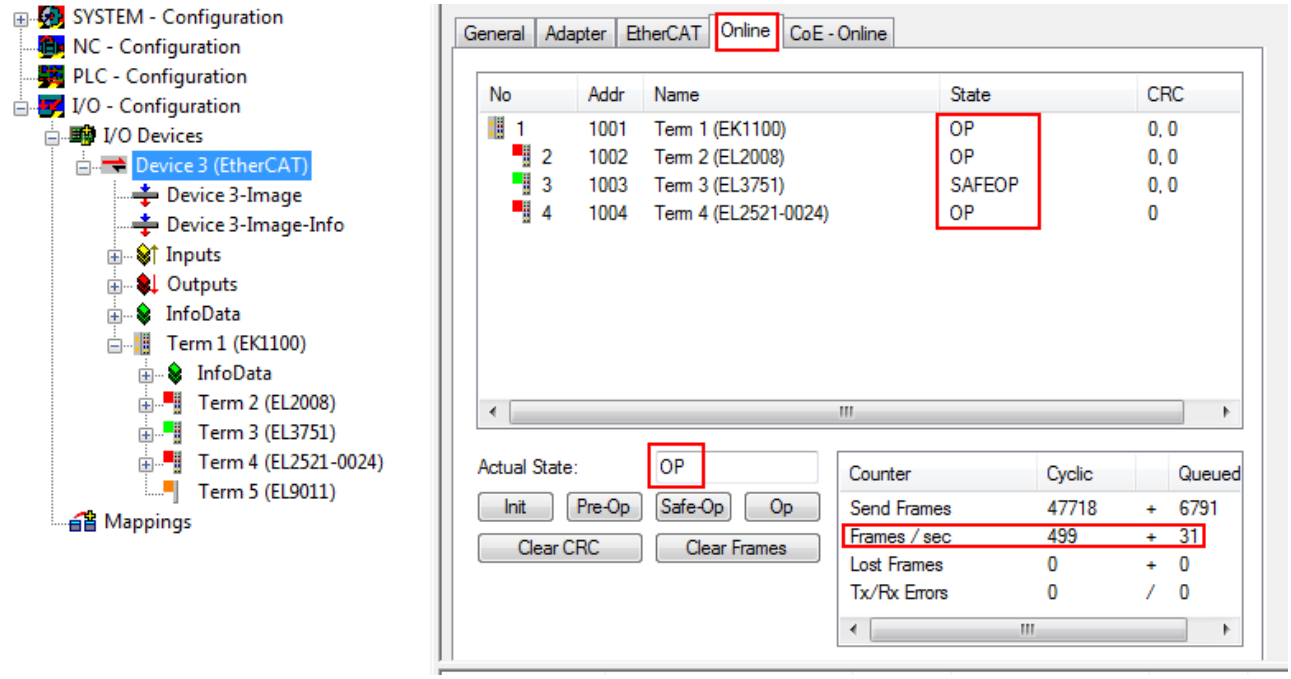


Fig. 111: Online display example

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.

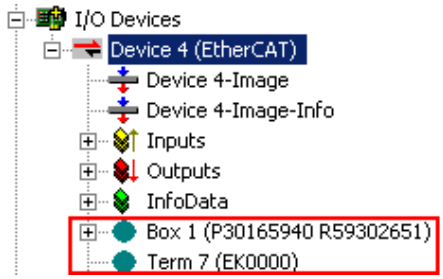


Fig. 112: Faulty identification

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. 113: 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.

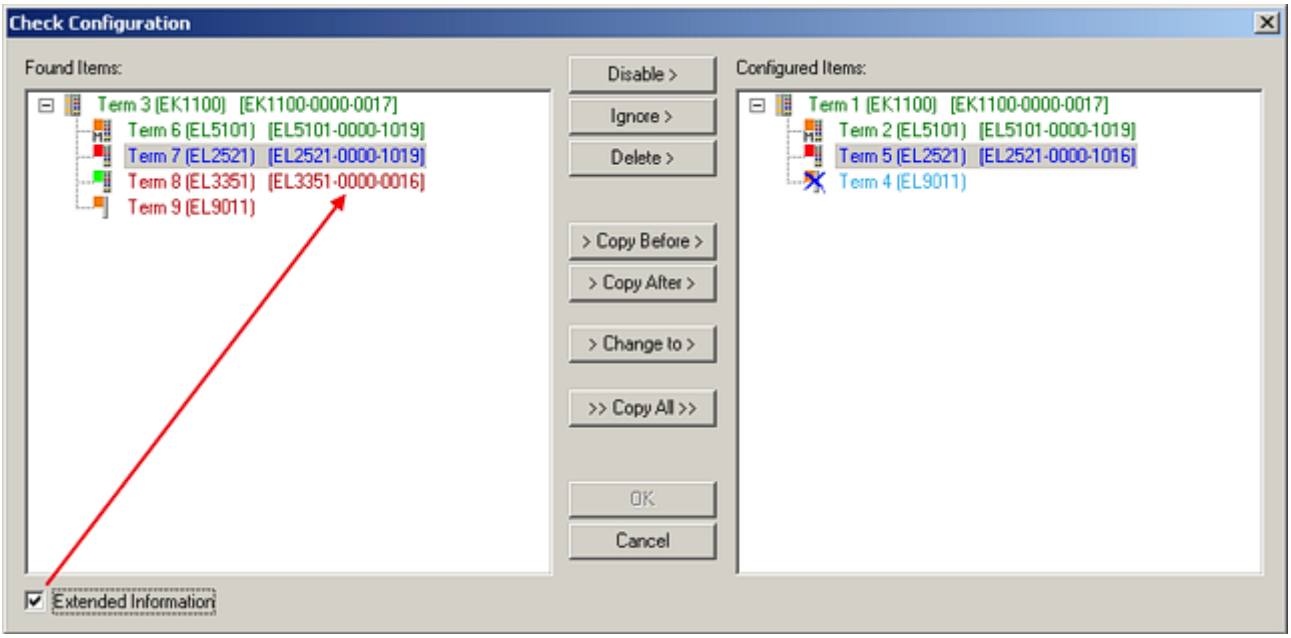


Fig. 114: Correction dialog

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.

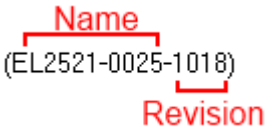


Fig. 115: Name/revision of the terminal

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, ...

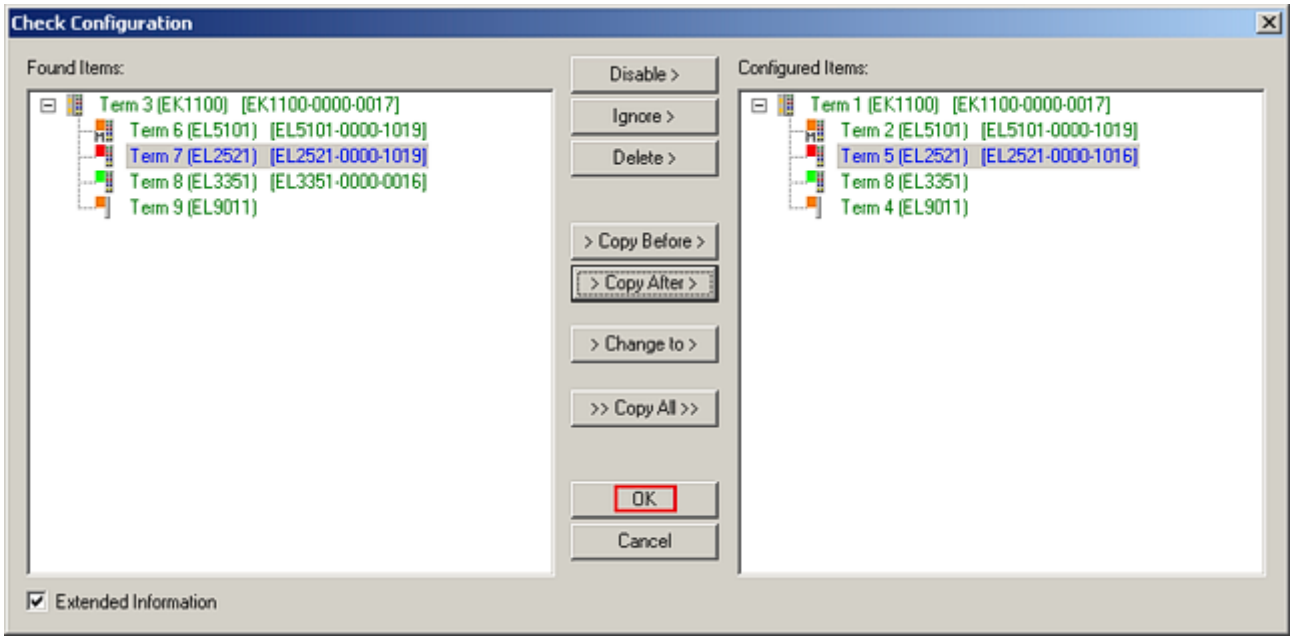


Fig. 116: Correction dialog with modifications

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.

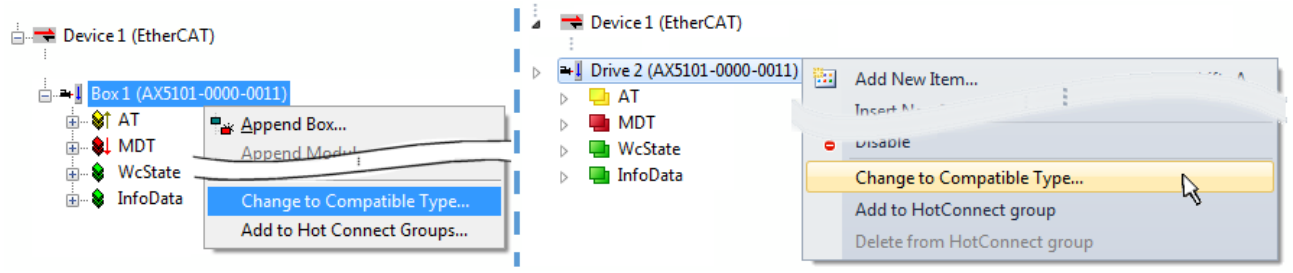


Fig. 117: Dialog “Change to Compatible Type…” (left: TwinCAT 2; right: TwinCAT 3)

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

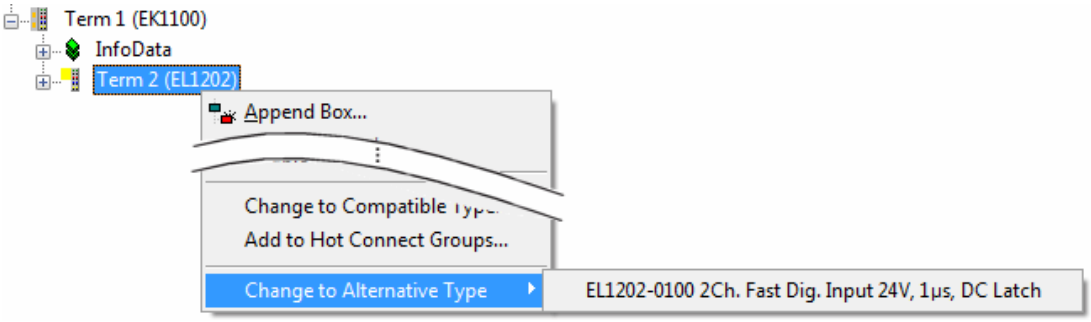


Fig. 118: TwinCAT 2 Dialog Change to Alternative Type

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).

8.2.7 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).

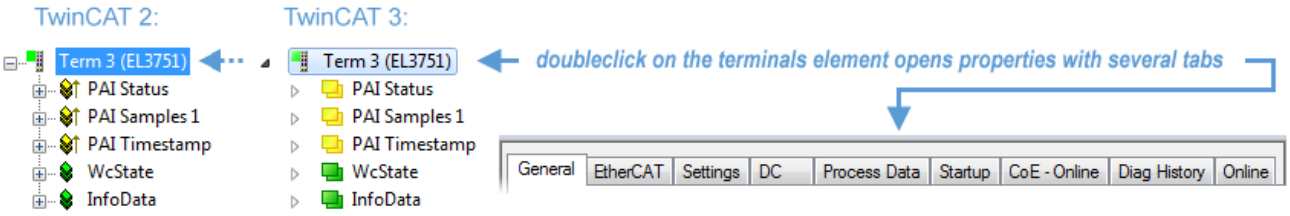


Fig. 119: Branch element as terminal EL3751

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

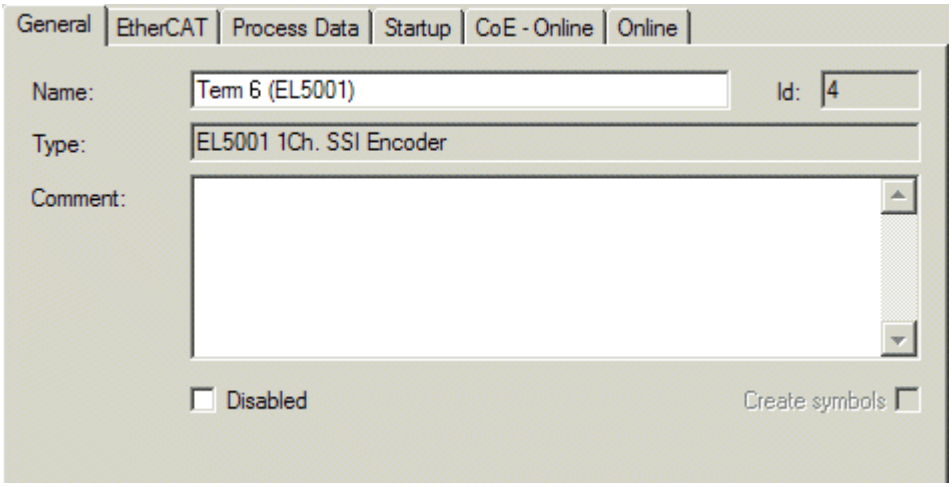


Fig. 120: “General” tab

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

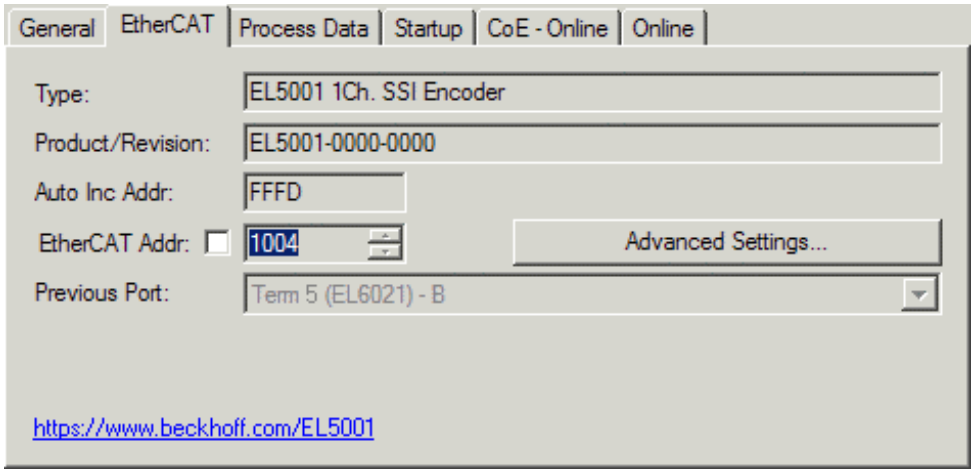


Fig. 121: “EtherCAT” tab

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.

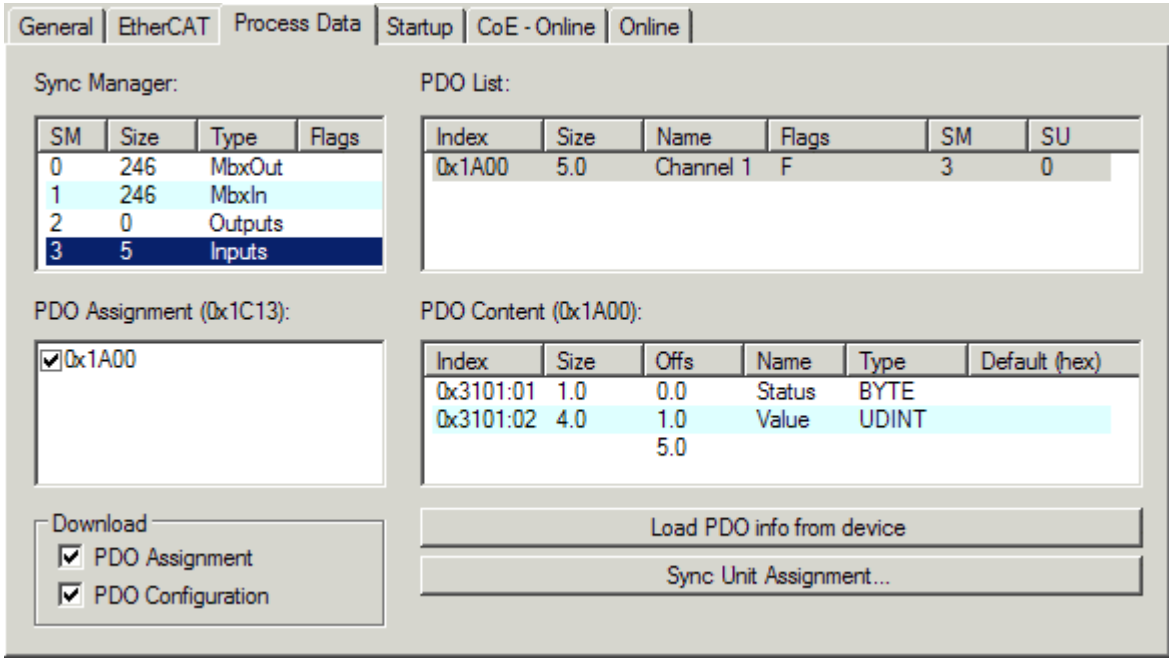


Fig. 122: “Process Data” tab

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”).

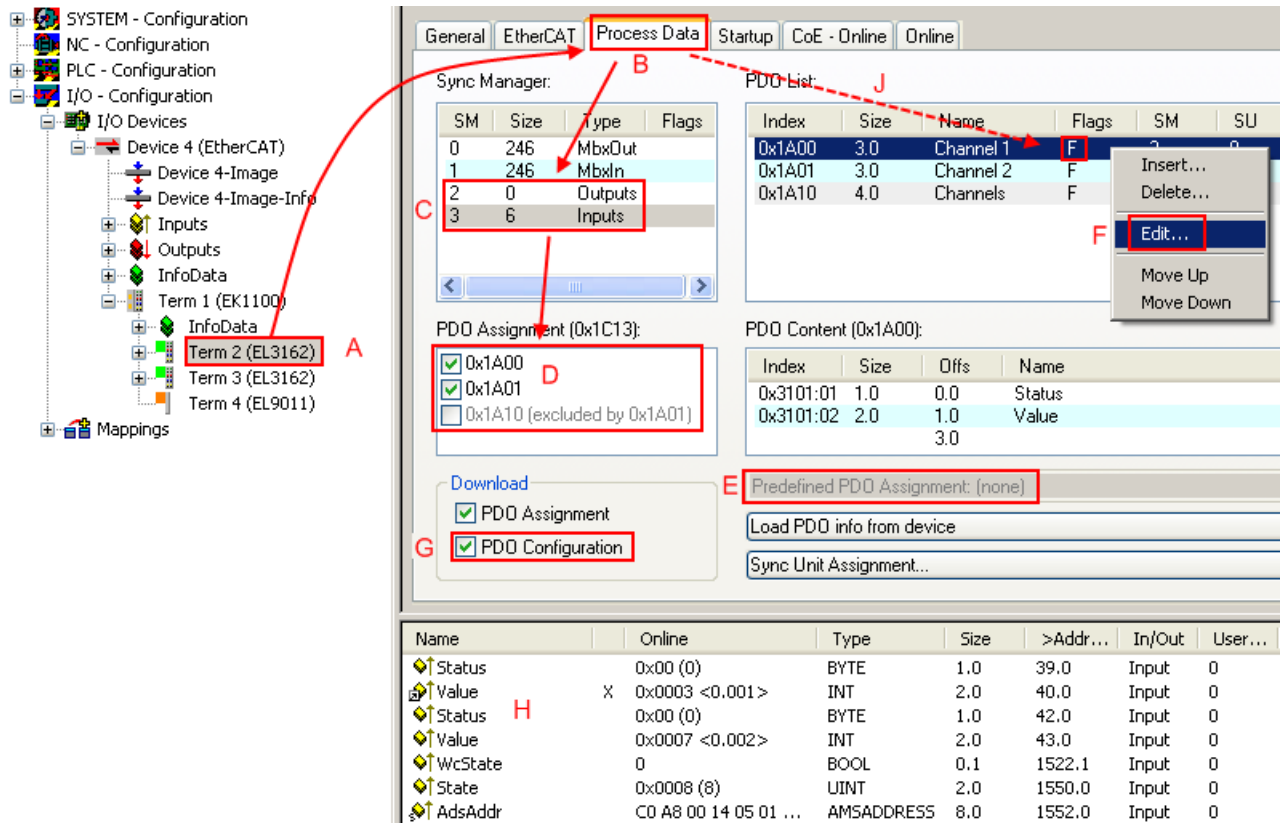


Fig. 123: Configuring the process data

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 [▶ 115] 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.

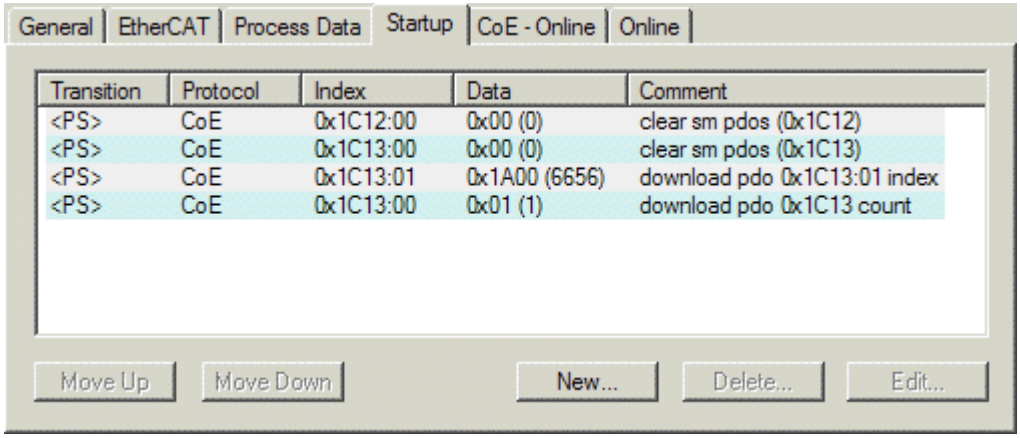


Fig. 124: “Startup” tab

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.

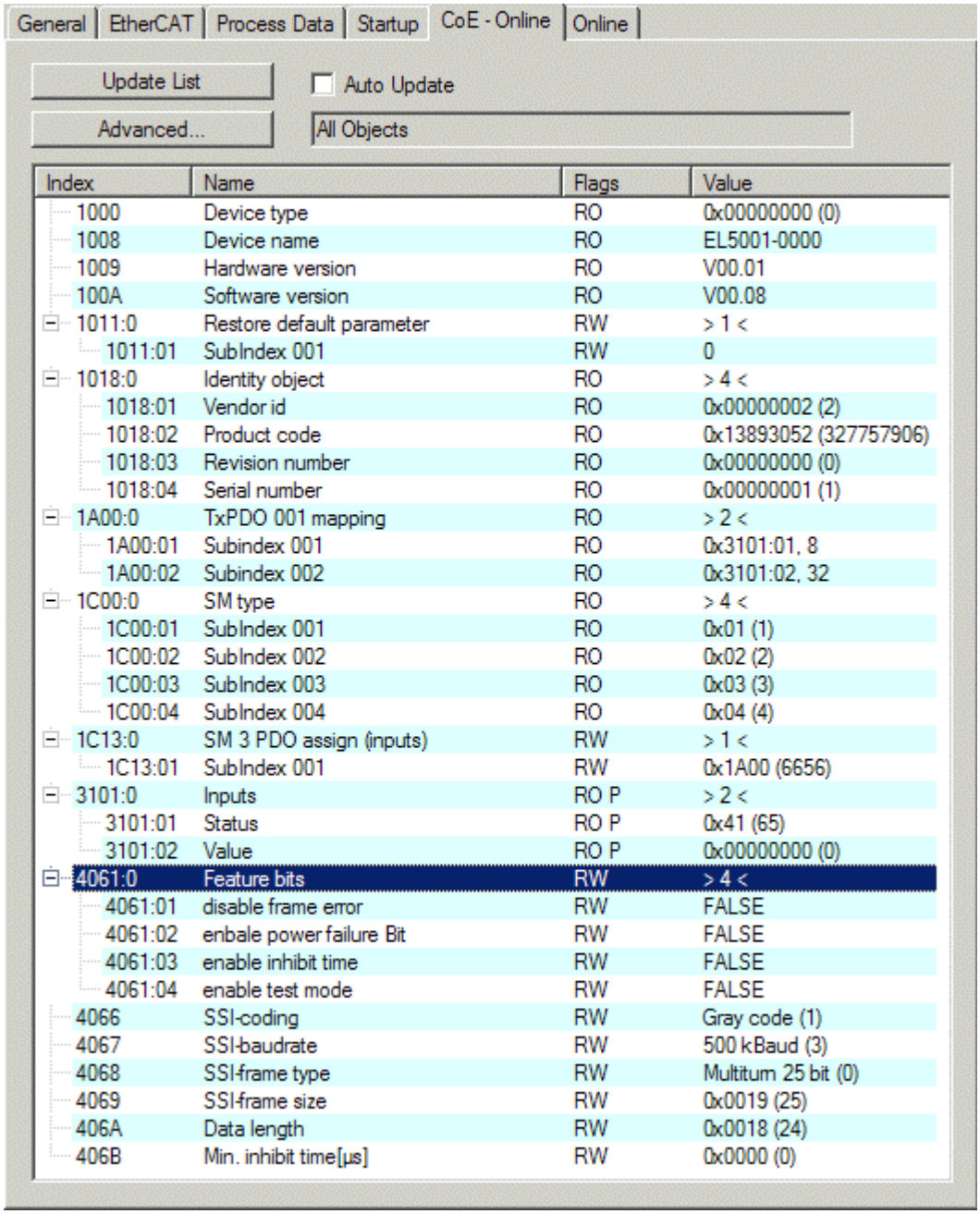


Fig. 125: “CoE - Online” tab

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.

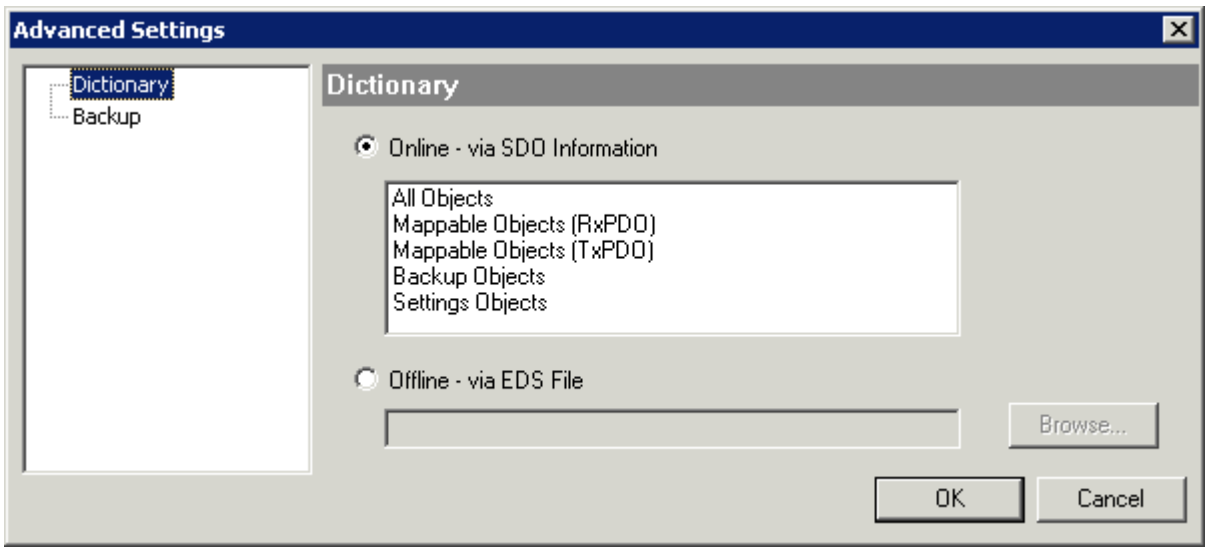


Fig. 126: Dialog “Advanced settings”

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

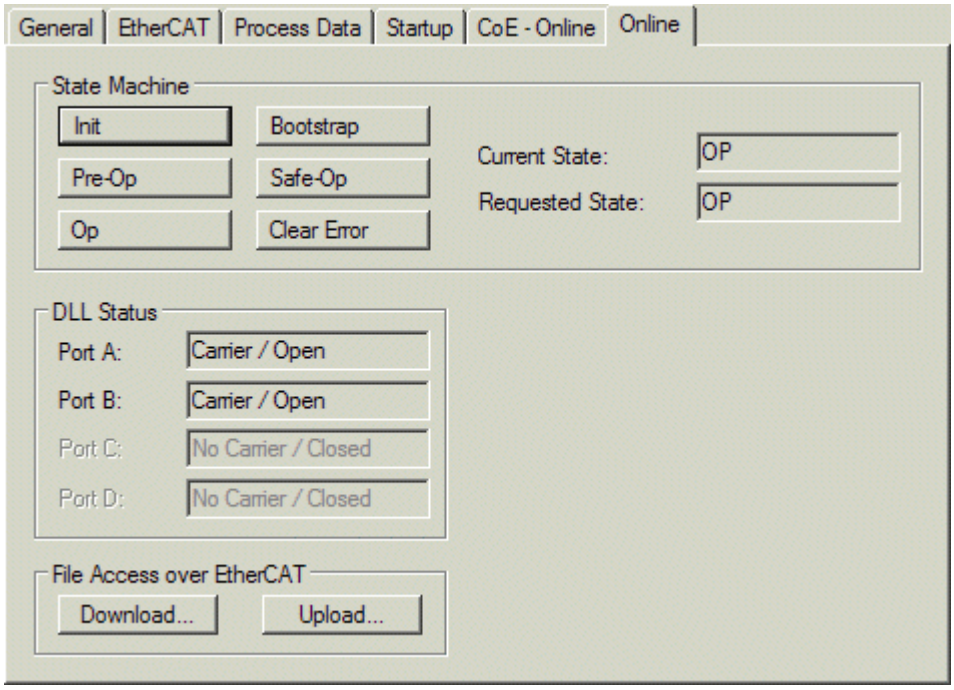


Fig. 127: “Online” tab

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)

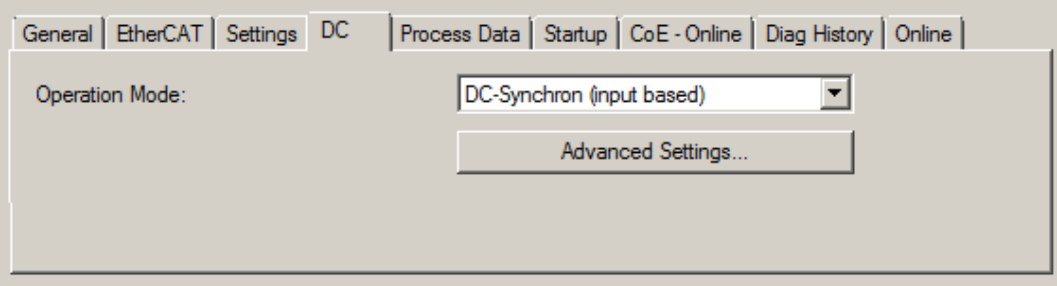


Fig. 128: “DC” tab (Distributed Clocks)

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

8.2.7.1 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 [▶ 113]),

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 [▶ 110] 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.

8.2.8 Import/Export of EtherCAT devices with SCI and XTI

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

8.2.8.1 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.

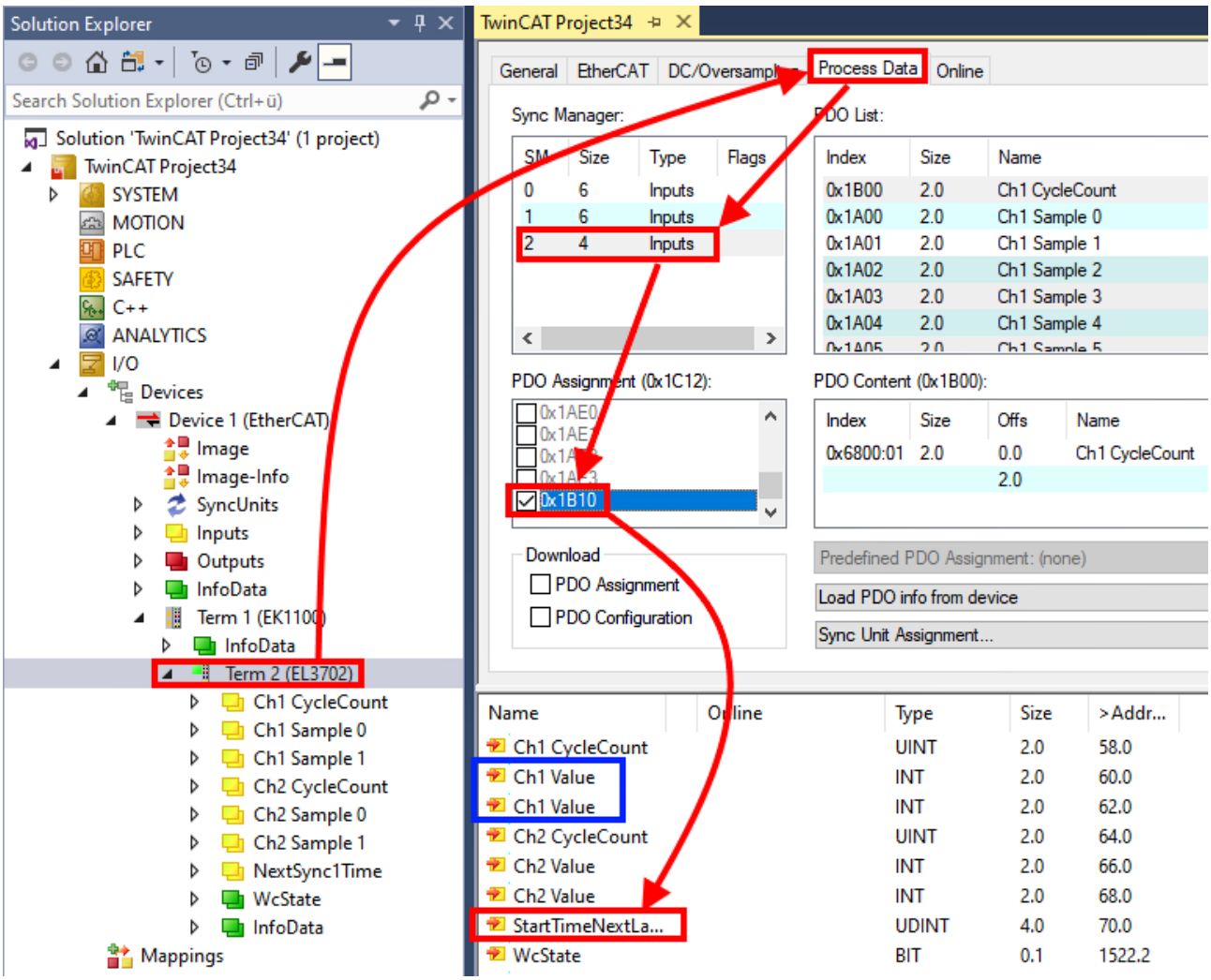
It may become necessary to reuse the modified slave in other projects in this way, without having to make equivalent configuration changes to the slave again. To accomplish this, proceed as follows:

- 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.

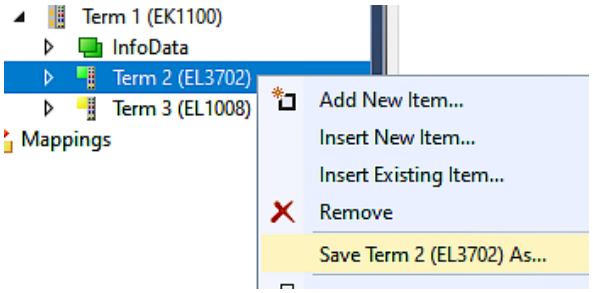
An example is provided below for illustration purposes: an EL3702 terminal with standard setting is switched to 2-fold oversampling (blue) and the optional PDO "StartTimeNextLatch" is added (red):



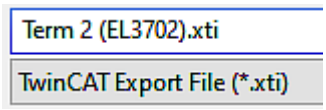
The two methods for exporting and importing the modified terminal referred to above are demonstrated below.

8.2.8.2 Procedure within TwinCAT with xti files

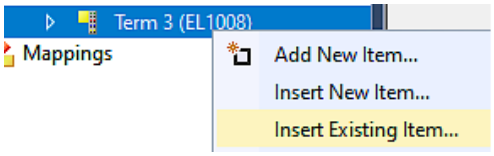
Each IO device can be exported/saved individually:



The xti file can be stored:



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



8.2.8.3 Procedure within and outside TwinCAT with sci file

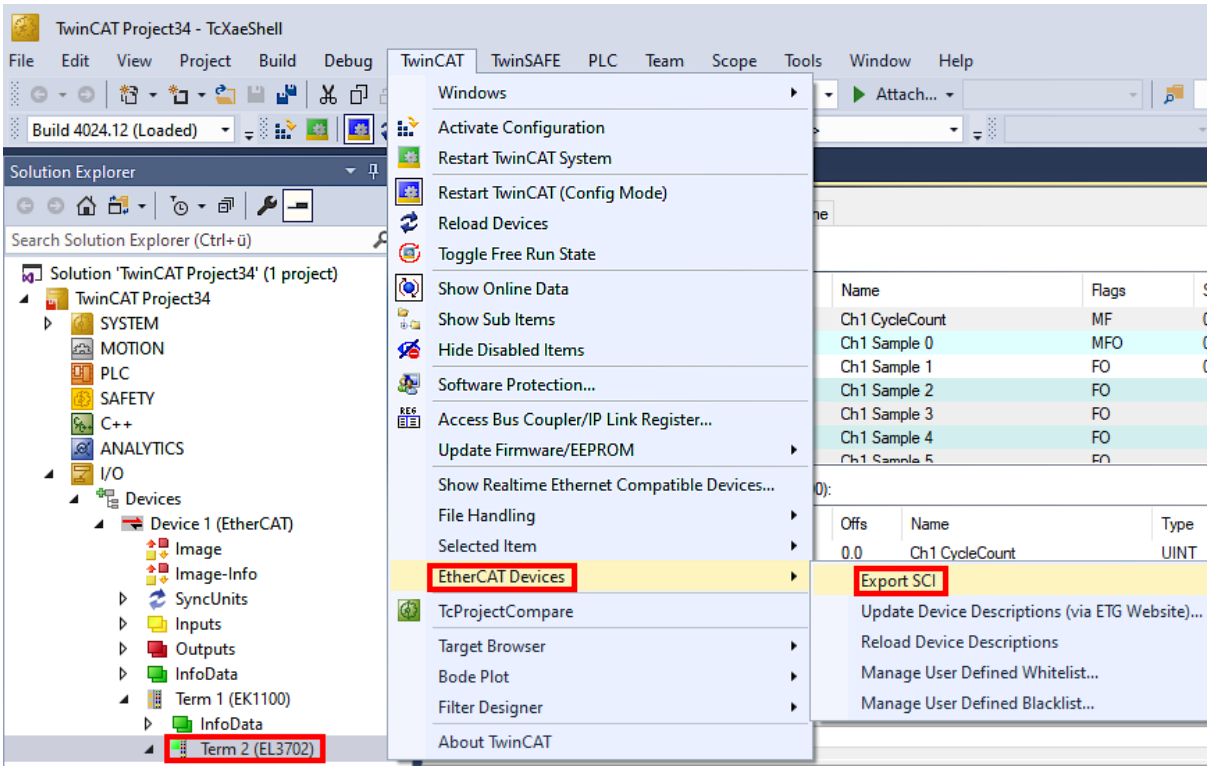
Note regarding availability (2021/01)

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:

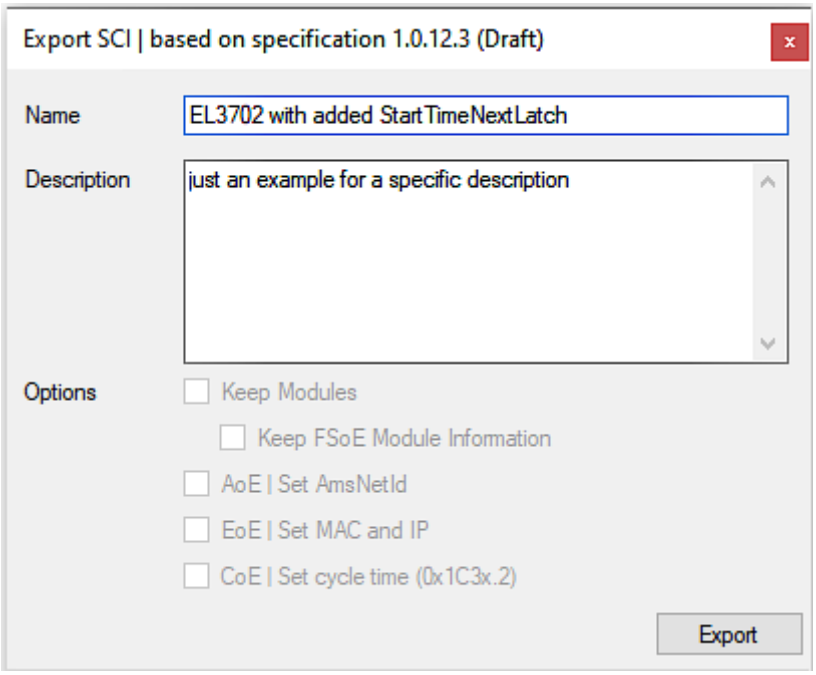
- select a single device via the menu (multiple selection is also possible):
  TwinCAT → EtherCAT Devices → Export SCI.



- 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:



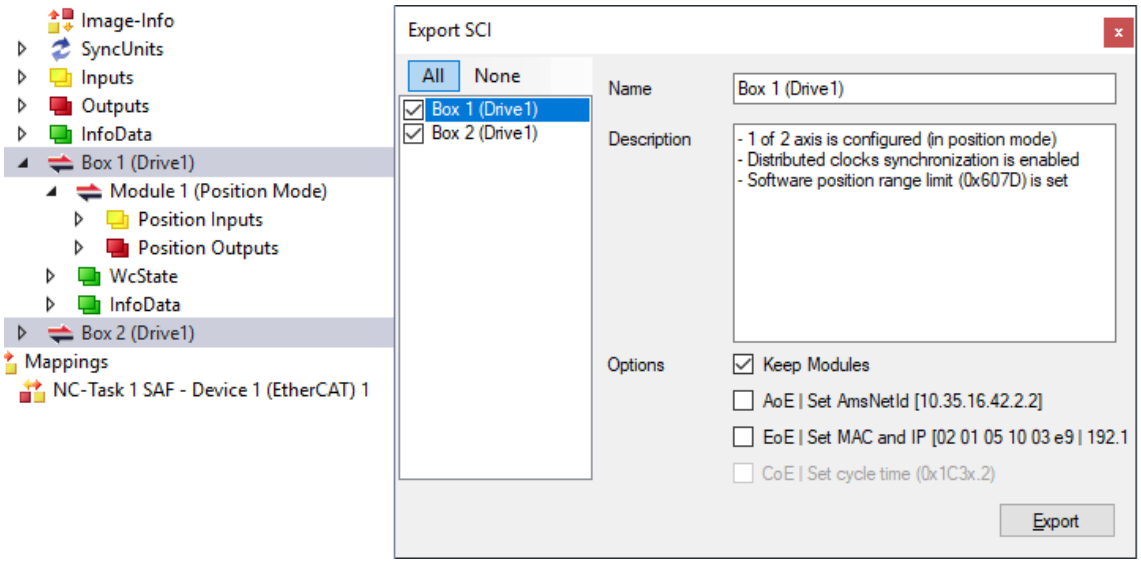
- A description may also be provided:



- 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.

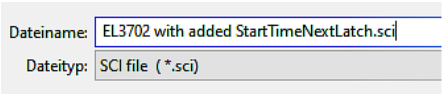
- A list view is available for multiple selections (*Export multiple SCI files)*:



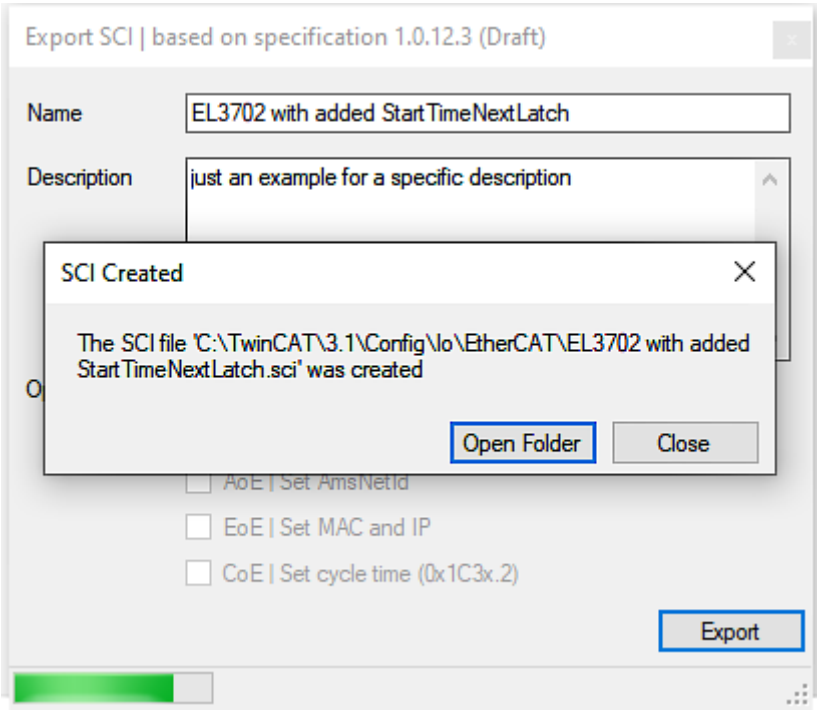
- Selection of the slaves to be exported:
  - All:
    All slaves are selected for export.

- None:
  All slaves are deselected.

- The sci file can be saved locally:

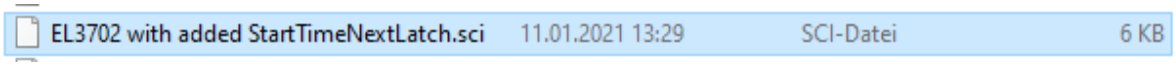


- The export takes place:

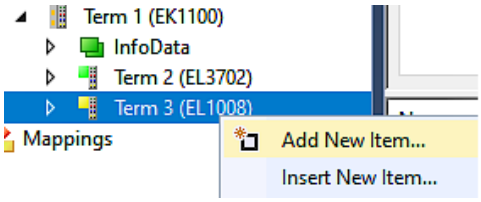


Import

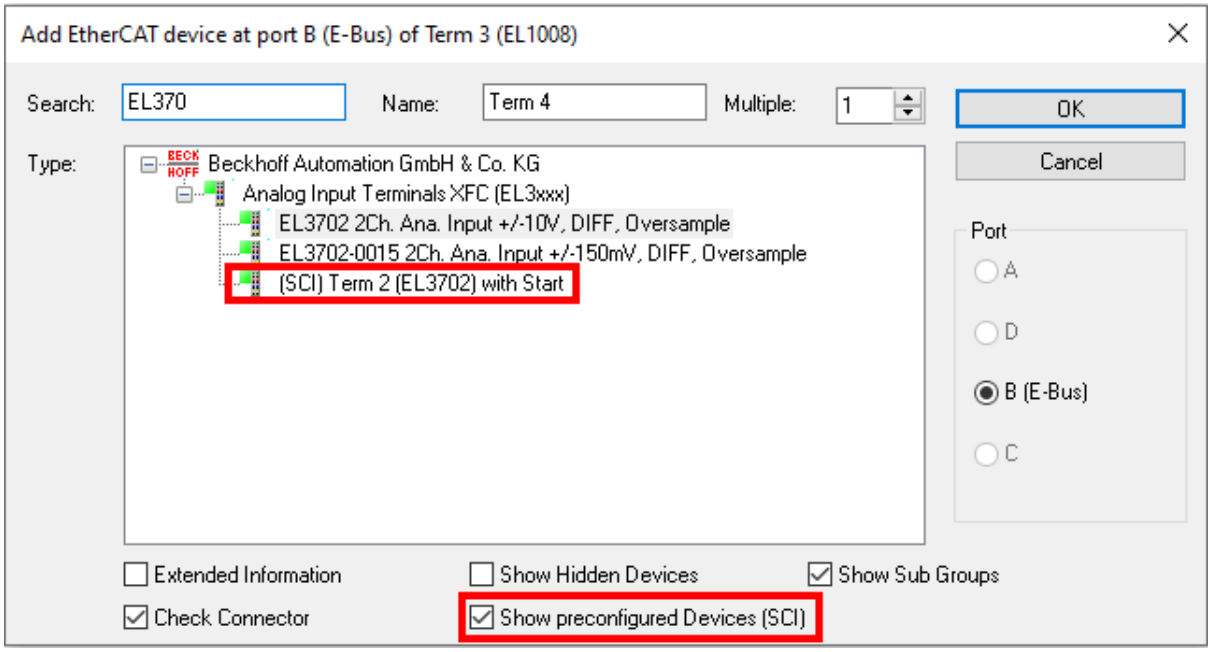
- An sci description can be inserted manually into the TwinCAT configuration like any normal Beckhoff device description.
- The sci file must be located in the TwinCAT ESI path, usually under:
  C:\TwinCAT\3.1\Config\Io\EtherCAT



- Open the selection dialog:

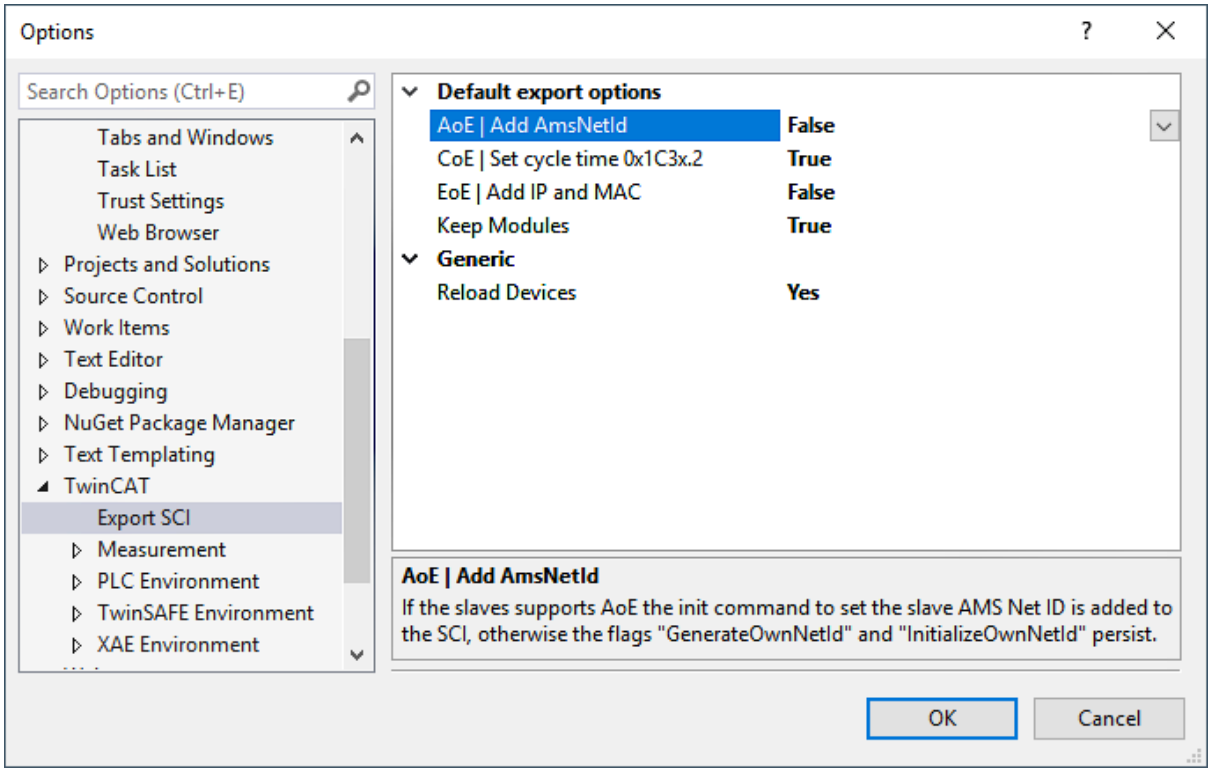


- Display SCI devices and select and insert the desired device:



Additional Notes

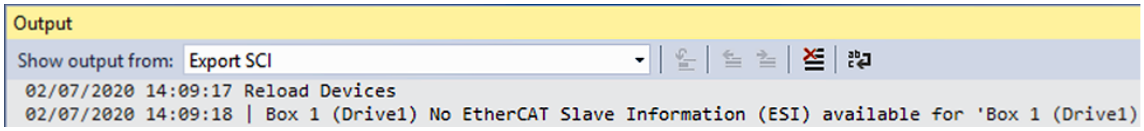
- Settings for the SCI function can be made via the general Options dialog (Tools → Options → TwinCAT → Export SCI):



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:



8.3 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.

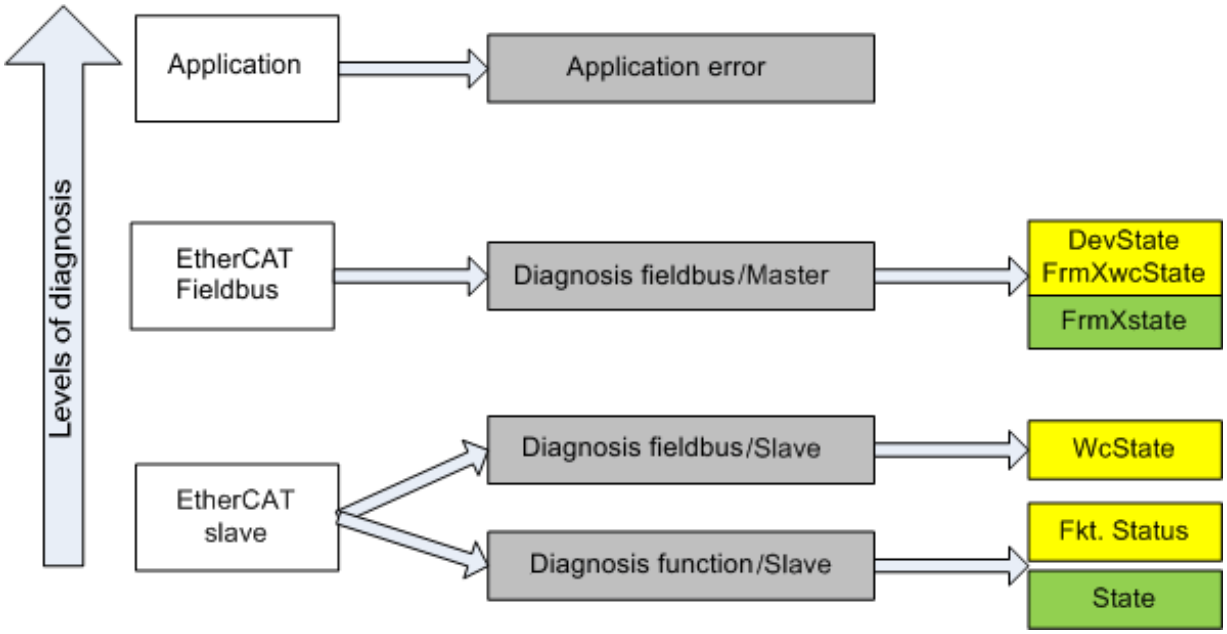


Fig. 129: Selection of the diagnostic information of an EtherCAT Slave

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.

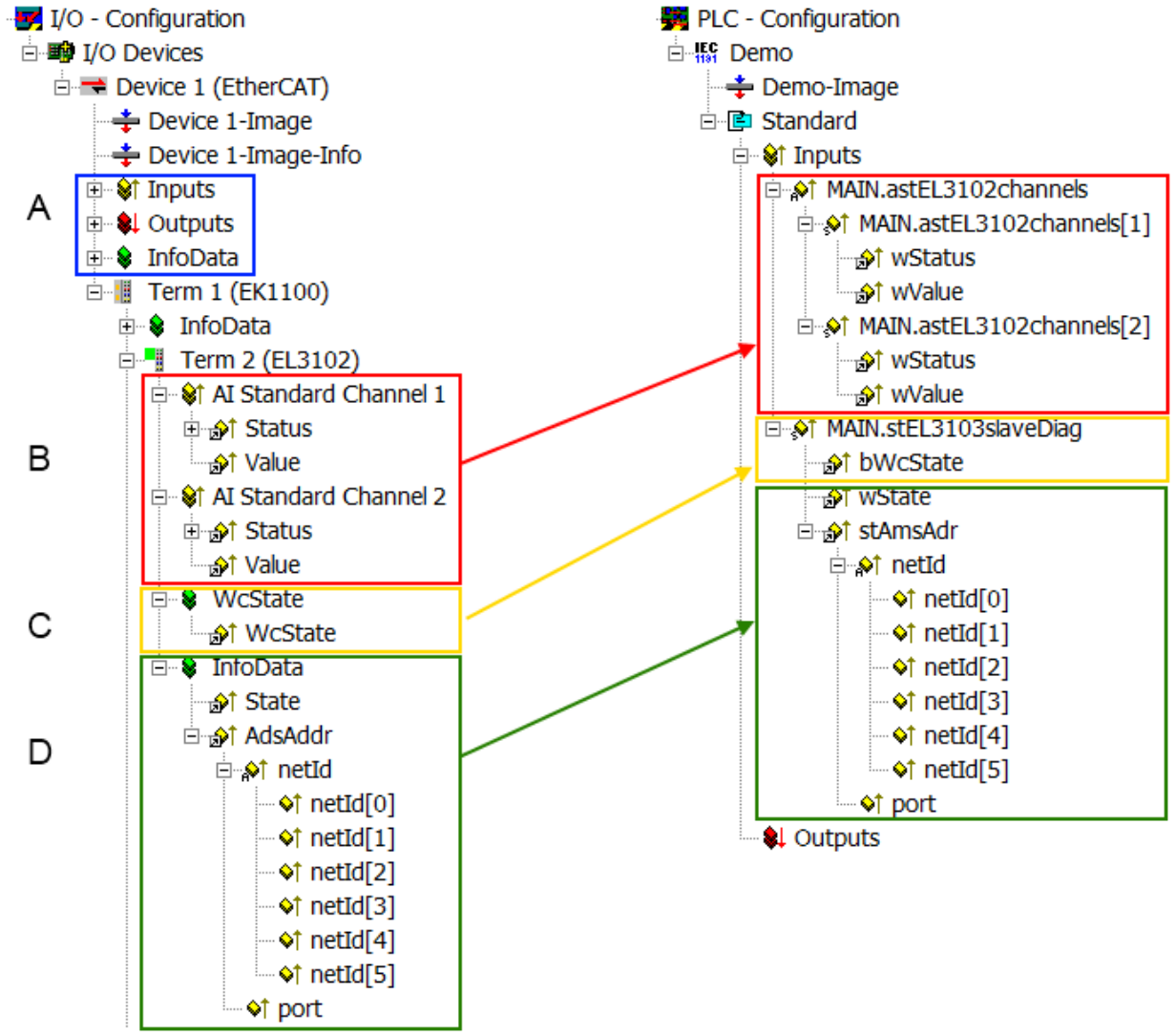


Fig. 130: Basic EtherCAT Slave Diagnosis in the PLC

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*:

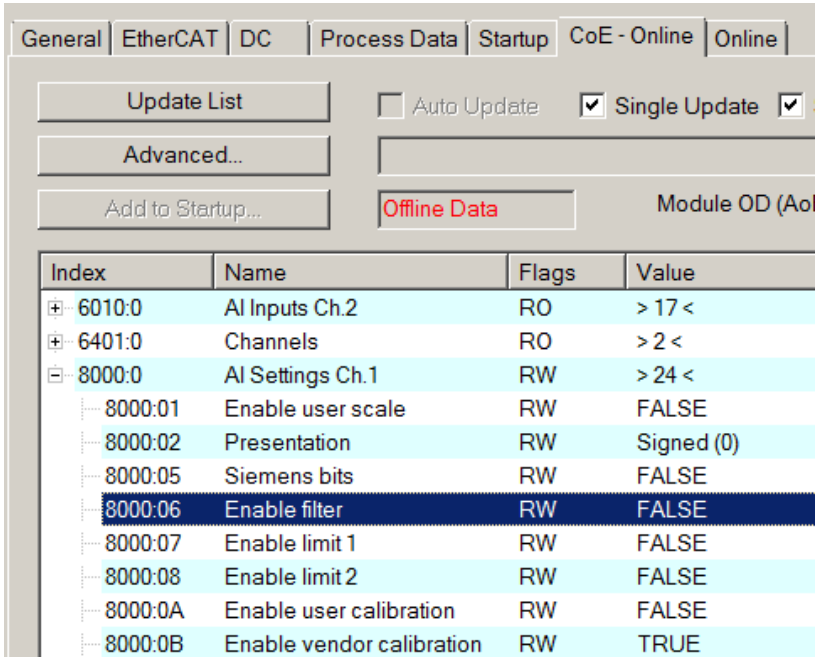


Fig. 131: EL3102, CoE directory

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.

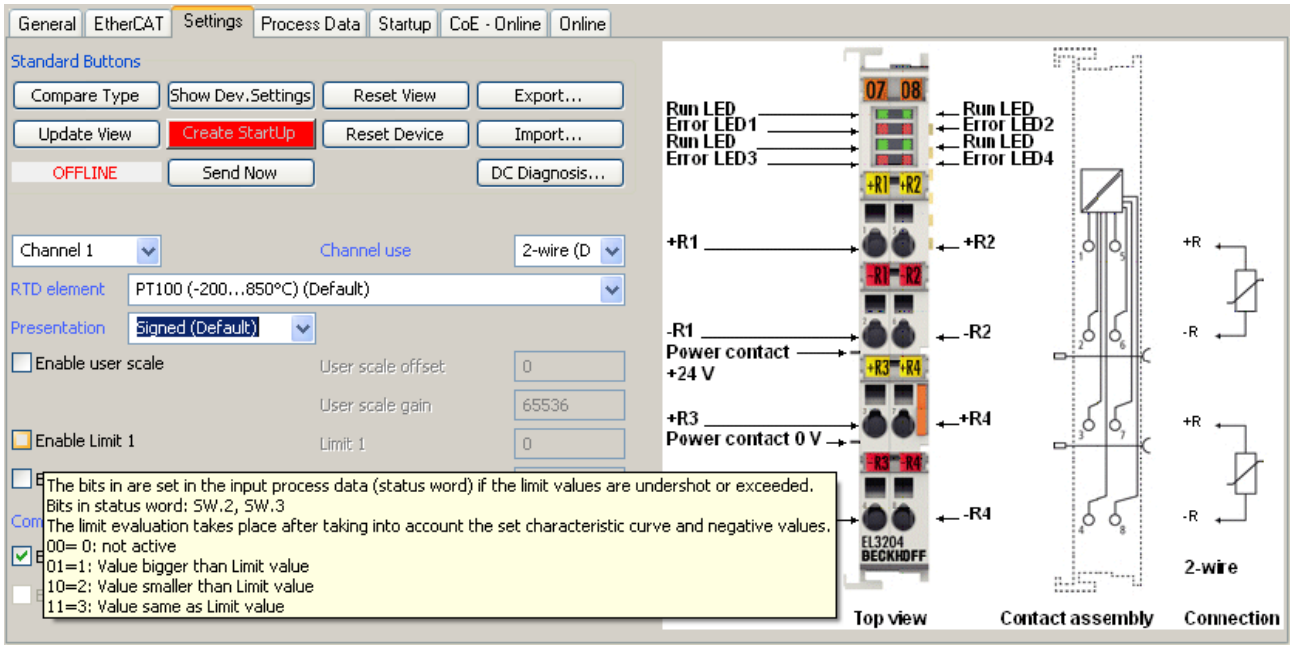


Fig. 132: Example of commissioning aid for a EL3204

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 [▶ 26]" 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.

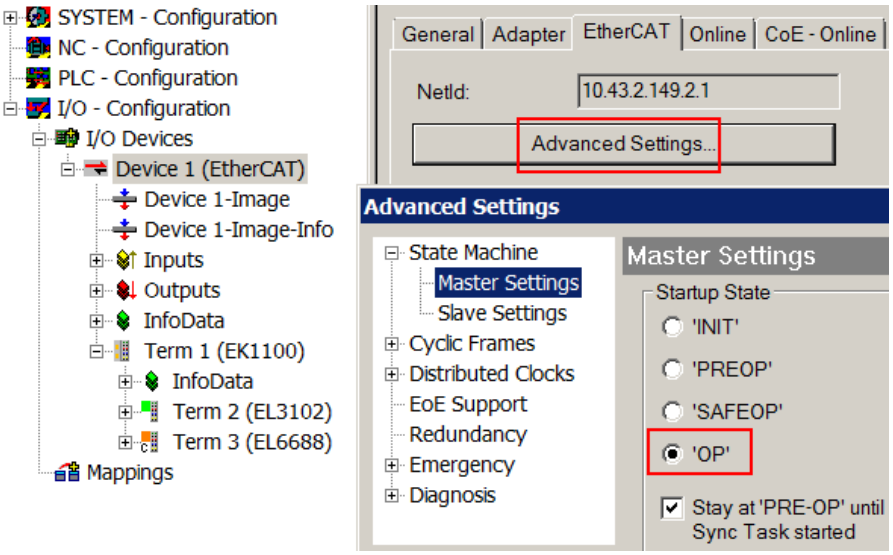


Fig. 133: Default behaviour of the System Manager

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

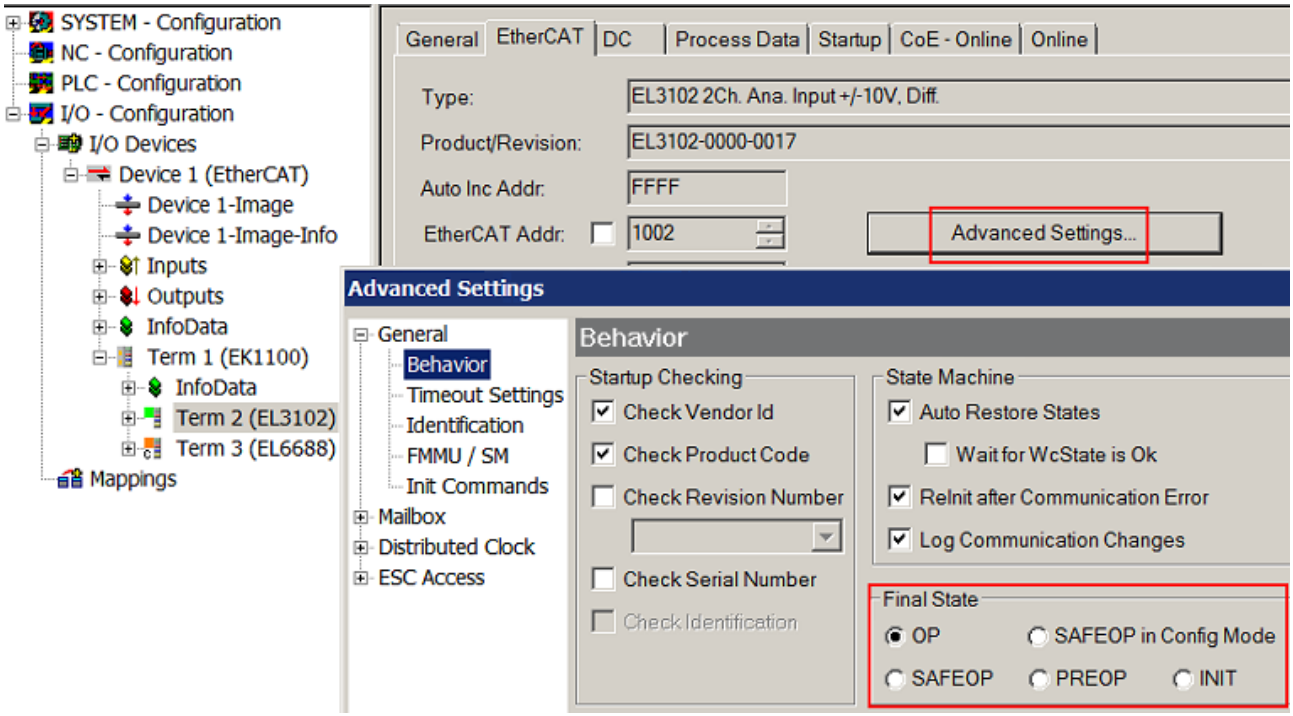


Fig. 134: Default target state in the Slave

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.

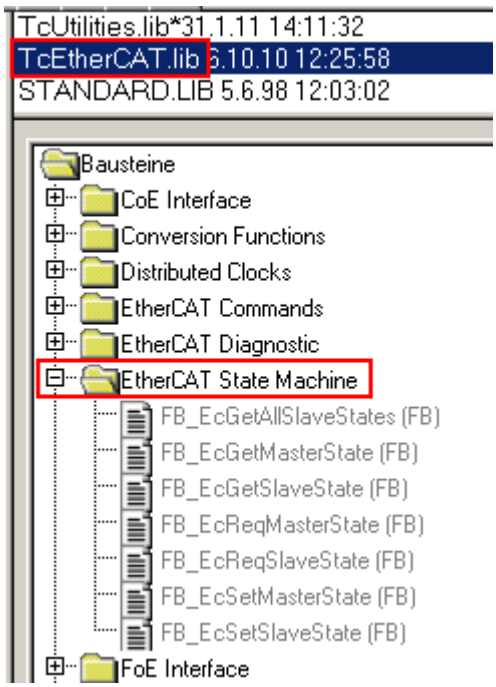
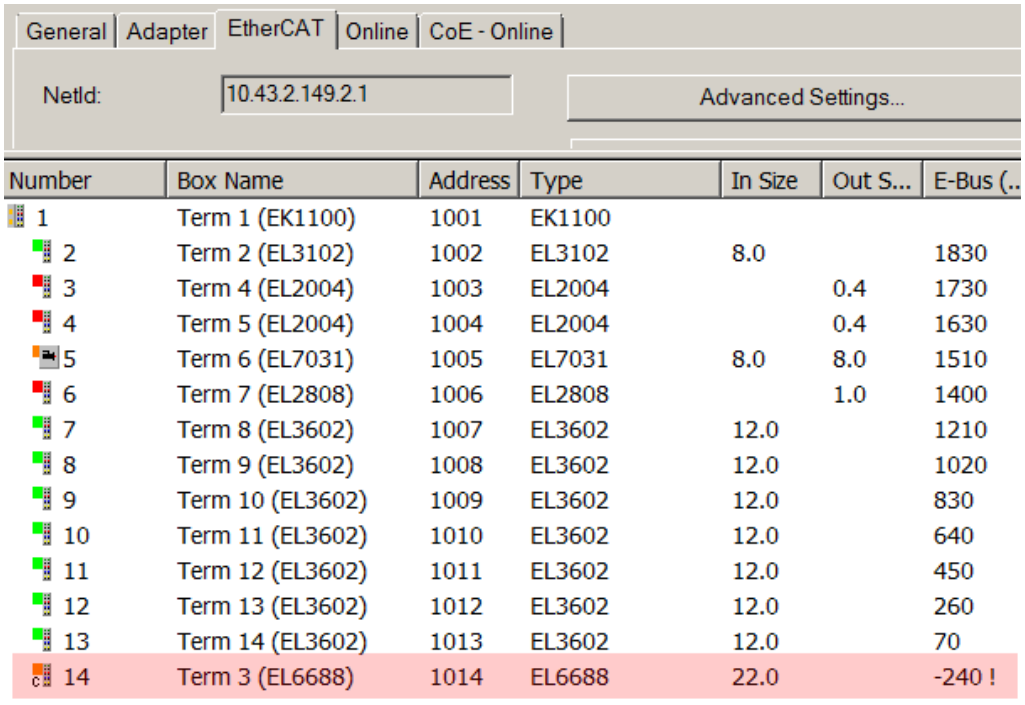


Fig. 135: PLC function blocks

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.



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. 136: 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:

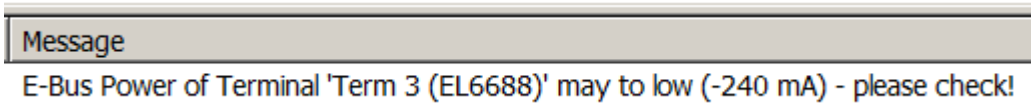


Fig. 137: Warning message for exceeding E-Bus current

NOTICE

Caution! Malfunction possible!

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

9 EL5032-00x0 - Commissioning

9.1 Quick start

Proceed as follows for standard commissioning of the EL5032 with EnDat 2.2 encoders.

1. Install the EL5032 in the E-bus terminal strand on an EtherCAT coupler, e.g. EK1100 or EK1501.
2. Connect the EnDat 2.2 encoder(s) according to the connection diagram (Data(+/-), Clock(+/-) and supply voltage).
   The encoders must support EnDat 2.2 and a clock frequency >= 8 MHz.
3. Set up a correct EtherCAT configuration with the terminal.
   Since the device is present and is electrically reachable, the simplest way of accomplishing this is by scanning the devices [▶ 99].
4. Activate the EtherCAT master and start the terminal in OP state.
   In the input variables the EL5032 must deliver State=8 and WC=0.
5. Parameterize the CoE settings of the EL5032 according to the EnDat2.2 encoder data sheet.
   - Reverse any previous parameter changes by means of a CoE reset: enter **0x64616F6C** in object 0x1011:01 [▶ 143].
   - If an encoder voltage of, for sample, 9 V is set in object 0x8008:13 [▶ 144], ensure before connecting that this is supported by both encoders.
6. The data can now be read via the process data [▶ 143].

9.2 Process data

9.2.1 Sync Manager (SM)

The scope of process data offered can be modified via the "Process Data" tab (see Fig. *Process Data tab SM3, EL5032 (default)* below).

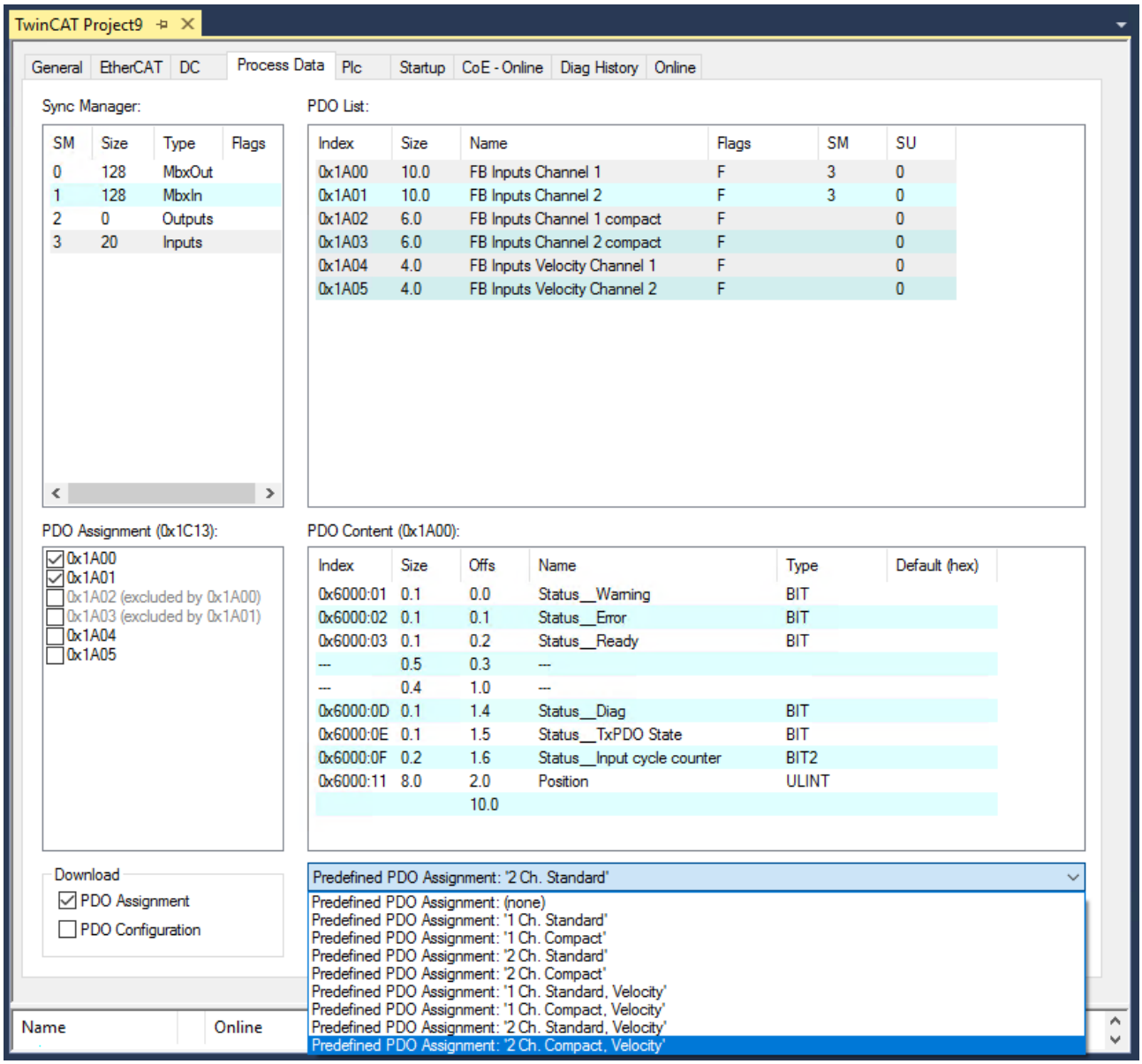


Fig. 138: Process data tab SM3, EL5032-0000-0028 (default)

9.2.2 PDO Assignment

To configure the process data, proceed as follows:

- Select the required Sync Manager (SM2 + SM3 can be edited here) in the "Sync Manager" box at the top left (see fig. *Process Data tab SM3, EL5032-0000-0028 (default)*).
- The process data assigned to this Sync Manager can then be switched on or off in the "PDO Assignment" box underneath.
- Restarting the EtherCAT system, or reloading the configuration in Config mode (F4), causes the EtherCAT communication to restart, and the process data is transferred from the terminal.

PDO assignment of Sync Managers, EL5032

SM3, PDO Assignment 0x1C13				
Index	**Size (byte.bit)**	**Name**	**PDO content**	**Size (byte.bit)**
0x1A00 (default) FB Inputs Channel 1	10.0	Status_Warning	Index 0x6000:01 [▶ 145]	0.1
		Status_Error	Index 0x6000:02 [▶ 145]	0.1
		Status_Ready	Index 0x6000:03 [▶ 145]	0.1
		[Offset]	-	0.5
		[Offset]	-	0.4
		Status_Diag	Index 0x6000:0D [▶ 145]	0.1
		Status_TxPDO State	Index 0x6000:0E [▶ 145]	0.1
		Status_Input Cycle counter	Index 0x6000:0F [▶ 145]	0.2
		Position	Index 0x6000:11 [▶ 145]	8.0
0x1A01 (default) FB Inputs Channel 2	10.0	Status_Warning	Index 0x6010:01 [▶ 145]	0.1
		Status_Error	Index 0x6010:02 [▶ 145]	0.1
		Status_Ready	Index 0x6010:03 [▶ 145]	0.1
		[Offset]	-	0.5
		[Offset]	-	0.4
		Status_Diag	Index 0x6010:0D [▶ 145]	0.1
		Status_TxPDO State	Index 0x6010:0E [▶ 145]	0.1
		Status_Input Cycle counter	Index 0x6010:0F [▶ 145]	0.2
		Position	Index 0x6010:11 [▶ 145]	8.0
0x1A02 FB Inputs Channel 1 compact	6.0	Status_Warning	Index 0x6000:01 [▶ 145]	0.1
		Status_Error	Index 0x6000:02 [▶ 145]	0.1
		Status_Ready	Index 0x6000:03 [▶ 145]	0.1
		[Offset]	-	0.5
		[Offset]	-	0.4
		Status_Diag	Index 0x6000:0D [▶ 145]	0.1
		Status_TxPDO State	Index 0x6000:0E [▶ 145]	0.1
		Status_Input Cycle counter	Index 0x6000:0F [▶ 145]	0.2
		Position	Index 0x6000:11 [▶ 145]	4.0
0x1A03 FB Inputs Channel 2 compact	6.0	Status_Warning	Index 0x6010:01 [▶ 145]	0.1
		Status_Error	Index 0x6010:02 [▶ 145]	0.1
		Status_Ready	Index 0x6010:03 [▶ 145]	0.1
		[Offset]	-	0.5
		[Offset]	-	0.4
		Status_Diag	Index 0x6010:0D [▶ 145]	0.1
		Status_TxPDO State	Index 0x6010:0E [▶ 145]	4.0
		Status_Input Cycle counter	Index 0x6010:0F [▶ 145]	4.0
		Position	Index 0x6010:11 [▶ 145]	4.0
0x1A04*) FB Velocity Inputs Channel 1	4.0	Velocity	Index 0x6008:12 [▶ 145]	4.0
0x1A05*) FB Velocity Inputs Channel 2	4.0	Velocity	Index 0x6018:12 [▶ 145]	4.0
*) EL5032 from Firmware 16, Revision 28 /EL5032-0090 from Revision 17				

9.2.3 Predefined PDO Assignment

The "Predefined PDO Assignment" enables a simplified selection of the process data. The desired function is selected on the lower part of the Process Data tab. As a result, all necessary PDOs are automatically enabled and the unnecessary PDOs are disabled.

Depending on the firmware/revision status, the following PDO assignments are available for selection:

Name	SM3, PDO assignment
1 Ch. Standard	0x1A00 [▶ 151] (FB Inputs Channel 1)
1 Ch. Compact	0x1A02 [▶ 152] (FB Inputs Channel 1 compact)
2 Ch. Standard	0x1A00 [▶ 151] (FB Inputs Channel 1) 0x1A01 [▶ 151] (FB Inputs Channel 2)
2 Ch. Compact	0x1A02 [▶ 152] (FB Inputs Channel 1 compact) 0x1A03 [▶ 152] (FB Inputs Channel 2 compact)
1 Ch. Standard, Velocity*)	0x1A00 [▶ 151] (FB Inputs Channel 1) 0x1A04 [▶ 152] (FB Inputs Velocity Channel 1)
1 Ch. Compact, Velocity*)	0x1A02 [▶ 152] (FB Inputs Channel 1 compact) 0x1A04 [▶ 152] (FB Inputs Velocity Channel 1)
2 Ch. Standard, Velocity*)	0x1A00 [▶ 151] (FB Inputs Channel 1) 0x1A01 [▶ 151] (FB Inputs Channel 2) 0x1A04 [▶ 152] (FB Inputs Velocity Channel 1) 0x1A05 [▶ 152] (FB Inputs Velocity Channel 2)
2 Ch. Compact, Velocity*)	0x1A02 [▶ 152] (FB Inputs Channel 1 compact) 0x1A03 [▶ 152] (FB Inputs Channel 2 compact) 0x1A04 [▶ 152] (FB Inputs Velocity Channel 1) 0x1A05 [▶ 152] (FB Inputs Velocity Channel 2)
*) EL5032 from Firmware 16 , Revision 28 / EL5032-0090 from Revision 17	

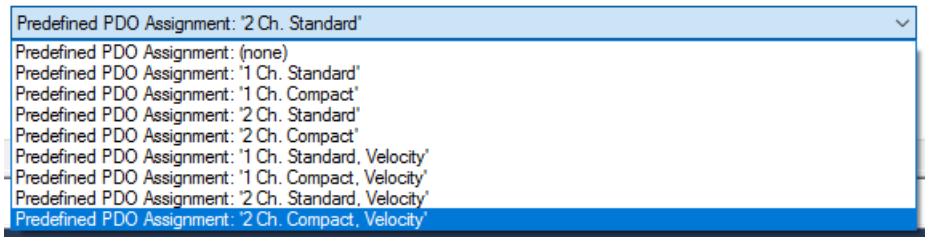


Fig. 139: Process data tab Predefined PDO Assignment, EL5032-0000-0028

9.3 Velocity measurement

In addition to the pure position value, a velocity value can also be calculated and output (EL5032: from firmware 16, Revision 28 /EL5032-0090: from Revision 17).

Procedure of velocity measurement

The velocity is calculated from the change of the position value in a defined time interval.
The result is output via the process data either in index 0x60n8:12 "Velocity" or in index 0x60n8:13 "Velocity (int16)" (n = 0 for channel 1, n = 1 for channel 2).

- Selection of the velocity window in index 0x80n8:17 "Velocity window"
  - Default: 10 ms
  - Follow the instructions below to select the correct timeframe.
- The velocity value can be normalized over the two indices:
  - 0x80n8:15 "Velocity numerator"
  - 0x80n8:16 "Velocity denominator".
- Additionally, a digital filter can be set (see chapter "Filter operation [▶ 136]")
- The calculated velocity value is output:
  - either via 0x60n8:12 "Velocity"
  - or via 0x60n8:13 "Velocity (int16)"
- The calculation is carried out acyclically and without reference to the distributed clock system and is therefore independent of the operation mode.

Correct selection of the timeframe

The timeframe for measuring the velocity is set to 10 ms in the delivery state. It is always a multiple of the cycle time, and is rounded up to the next larger value. This is due to technical reasons, because the position value is internally available only once per PLC cycle.

The accuracy of the determined velocity depends on the size of the timeframe (0x80n8:17 "Velocity window"). This must be selected according to the application.

- At constant velocities, select a larger timeframe so that the best possible averaging can take place.
- In situations where frequent positive or negative accelerations are encountered, select a smaller timeframe in order to be able to respond more quickly to changing position values.

9.4 Filter operation

Velocity Filter (index 0x80p8:18 [▶ 143])

The EL5032 (from FW16 [▶ 175]) and EL5032-0090 terminals are equipped with a digital filter that can assume the characteristics of an *I*nfinite *I*mpulse *R*esponse filter (*IIR filter)*. The filter is disabled by default (value: "0" (None)).

IIR filter

The filter with IIR characteristics is a discrete time, linear, time invariant filter that can be set to eight levels (level 1 = weak recursive filter, up to level 8 = strong recursive filter).
The IIR can be understood as a sliding average value calculation after a low-pass filter.
Here, the IIR filters work cycle-synchronously and are thus updated depending on the cycle time.

IIR filter

Difference equation: $Y_n = X_n * a_0 + Y_{n-1} * b_1$
with $a_0 + b_1 = 1$
a_0 = (see table)
$b_1 = 1 - a_0$

Velocity Filter 0x80p8:18	Value	PDO update time	Filter property	Comment	Rise time 10-90% [samples] (typ.)
0	None	-	-	Filter disabled	-
1	IIR 1	Cycle-synchronous (up to min. 100 µs)	Low-pass filter	$a_0 = 1/2^1 = 0.5$	3
2	IIR 2			$a_0 = 1/2^2 = 0.25$	8
3	IIR 3			$a_0 = 1/2^3 = 0.125$	17
4	IIR 4			$a_0 = 1/2^4 = 0.0625$	34
5	IIR 5			$a_0 = 1/2^5 = 0.03125$	69
6	IIR 6			$a_0 = 1/2^6 = 0.015625$	140
7	IIR 7			$a_0 = 1/2^7 = 0.0078125$	280
8	IIR 8			$a_0 = 1/2^8 = 0.00390625$	562

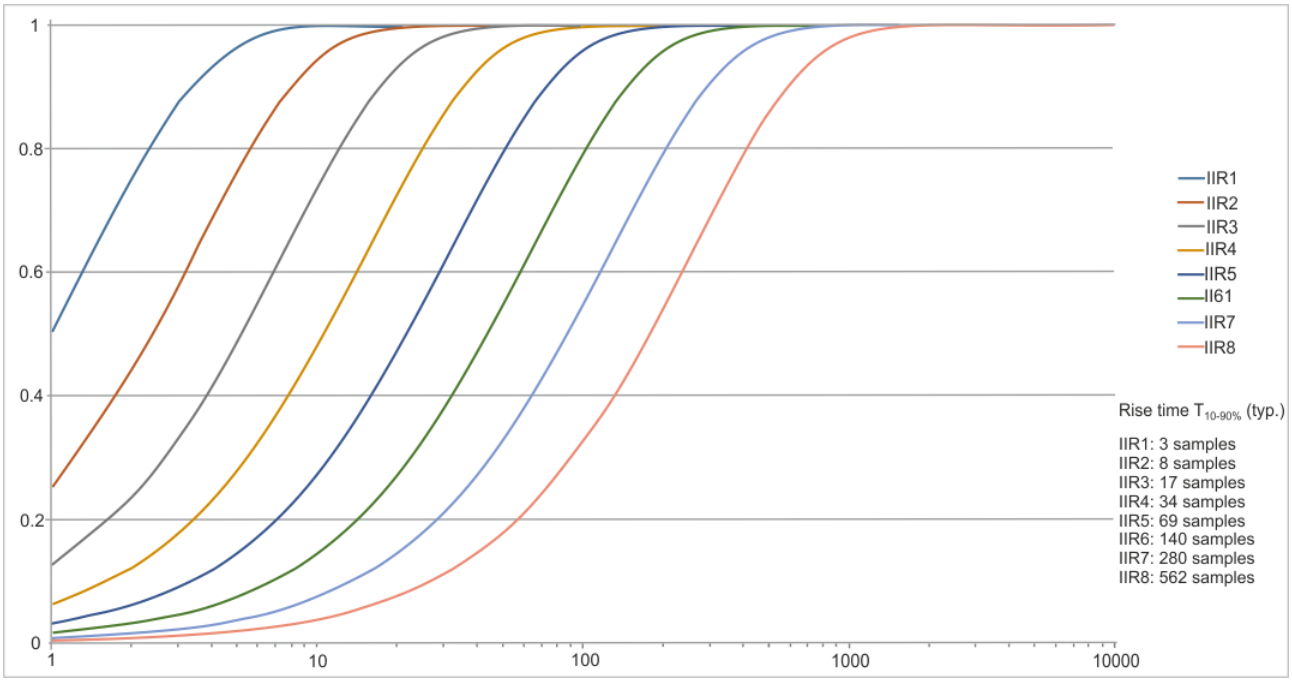


Fig. 140: Rise time $T_{10-90\%}$ (typ.)

9.5 Overview of commands and samples

The FB EnDat Command object, index 0xB0n8:01 [▶ 145] "Request" can be used to trigger various actions such as reading or writing to a specific memory area of the EnDat encoder (depending on the number of channels with n = 0 for channel 1 and n = 1 for channel 2).
The command is entered via the System Manager or the PLC (see following note)

Command entry via the System Manager

- If the command input is written via the System Manager, the value must be entered in the "Binary" field.
  - ⇨ LOBYTE first, (see following fig.).
  - ⇨ Space between all bytes of the request
    If entered without a space, the command is not accepted correctly.
- If the command is initiated via the PLC, then this takes place automatically.

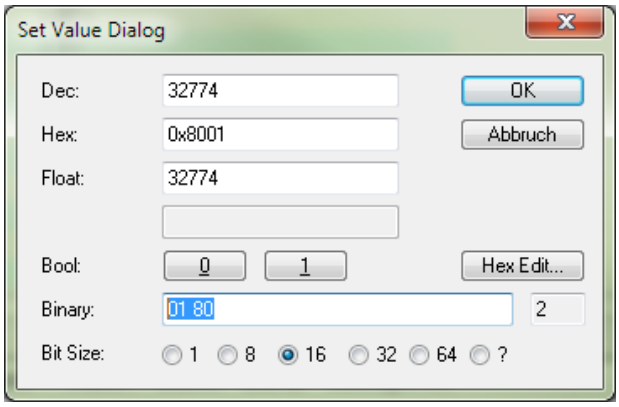


Fig. 141: Input using the example of command 0x8001 in the Set Value Dialog in the System Manager

Overview of commands

The following table provides an overview of the available commands, with detailed descriptions and examples in the following chapters.

Command	Description
0x1000 [▶ 138]	Clearing the Diag History (see note above)
0x1001 [▶ 138]	Diag History Test All implemented Diag messages are sent channel-dependent with this command.
0x8001 [▶ 139]	EnDat Reset (sends the mode command “measuring device send position values and receive error reset”)
0x8006 [▶ 139]	EnDat read Pos2, calculate and write offset (only for incremental encoders with reference mark).
0x8008 [▶ 140]	Suppress TxPDOState (only for EL5032 from FW 17) (suppresses the "TxPDO State", e.g. for encoders with distance-coded reference marks)
0x9000 [▶ 141]	Read EnDat Word
0x9001 [▶ 142]	Write EnDat Word

9.5.1 Command 0x1000 - Delete Diag History

To delete the Diag History, proceed as follows:

1. Enter the command 0x1000 via the FB EnDat Command object, index 0xB0p8:01 [▶ 145] "Request". (dependent on the number of channels with p = 0 for channel 1 and p = 1 for channel 2).
   ⇨ Please observe the note on Command Entry via the System Manager [▶ 137].

CoE Entry / Example for command 0x1000

B008:0	FB EnDat Command Ch.1	RO	> 3 <
B008:01	Request	RW	00 10
B008:02	Status	RO	0x00 (0)
B008:03	Response	RO	00 00 00 00

Fig. 142: Example command 0x1000, delete Diag History for channel 1

9.5.2 Command 0x1001 - Diag History Test

All implemented Diag messages are sent channel-dependent with this message.

Proceed as follows:

1. Enter the command 0x1001 via the FB EnDat Command object, index 0xB0p8:01 [▶ 145] "Request". (dependent on the number of channels with p = 0 for channel 1 and p = 1 for channel 2).
   ⇨ Please observe the note on Command Entry via the System Manager [▶ 137].

CoE Entry / Example for command 0x1001

B008:0	FB EnDat Command Ch.1	RO	> 3 <
B008:01	Request	RW	01 10
B008:02	Status	RO	0x00 (0)
B008:03	Response	RO	00 00 00

Fig. 143: Example command 0x1001, Diag History test for channel 1

9.5.3 Command 0x8001 - EnDat Reset

The command 0x8001 "EnDat Reset" sends the mode command "Encoder sends position values and receives error reset".

With this command the words 0 and 1 (error messages and warnings) in the operating state memory area are also reset.

With battery-buffered multi-turn rotary encoders the battery operating state error must be reset with this command after replacing the battery.

Proceed as follows:

1. Enter the command 0x1001 via the FB EnDat Command object, index 0xB0p8:01 [▶ 145] "Request". (dependent on the number of channels with p = 0 for channel 1 and p = 1 for channel 2).
   ⇨ Please observe the note on Command Entry via the System Manager [▶ 137].

CoE Entry / Example for command 0x8001

B008:0	FB EnDat Command Ch.1	RO	> 3 <
B008:01	Request	RW	01 80
B008:02	Status	RO	0x01 (1)
B008:03	Response	RO	01 80

Fig. 144: Example command 0x8001, EnDat reset for channel 1

9.5.4 Command 0x8006 - EnDat read Pos2

The command 0x8006 "EnDat read Pos2",
calculate and write offset (only for incremental encoders with reference mark).

The command calculates the difference between Position1 and Position2.

✓ A prerequisite is that the RM bit (Reference Mark) (0xA0p8:04 [▶ 148]) is set.

1. Enter the command 0x8006 via the FB EnDat Command object, index 0xB0p8:01 [▶ 145] "Request". (dependent on the number of channels with p = 0 for channel 1 and p = 1 for channel 2).
   ⇨ Please observe the note on Command Entry via the System Manager [▶ 137].

The offset is then automatically written to 0x80p8:12 [▶ 144] “Offset value”. The “Position raw value” (0xA0p8:43 [▶ 148]) is then absolute.

CoE Entry / Example for command 0x8006

B008:0	FB EnDat Command Ch.1	RO	> 3 <
B008:01	Request	RW	06 80
B008:02	Status	RO	0x01 (1)
B008:03	Response	RO	06 80 00 80 84 00 00 00 00 00

Fig. 145: Example command 0x8006, EnDat read Pos2 for channel 1

9.5.5 Command 0x8008 - Suppress TxPDO State (EL5032)

The command 0x8008 "Suppress TxPDO State" suppresses the "TxPDOState" (only EL5032 from FW 17).

This command suppresses the "TxPDO State" until the reference marks have been reached, e.g. for encoders with distance-coded reference marks, and the bit in 0xA0p8:04 "Reference Mark" = TRUE. The output position is then absolute.

1. Enter the command 0x8008 via the FB EnDat Command object, index 0xB0p8:01 [▶ 145] "Request". (dependent on the number of channels with p = 0 for channel 1 and p = 1 for channel 2).
   ⇨ Please observe the note on Command Entry via the System Manager [▶ 137].

CoE Entry / Example for command 0x8008

Example of commissioning encoders with distance-coded reference marks

E.g. HEIDENHAIN encoders from the EIB 1500 series External interface box, signal converter for high-accuracy angular measurement do not initially provide an absolute position value after switch-on. An absolute position value is only available after passing the distance-coded reference marks.

1. Initial state:
   No absolute position value is available.
   ⇨ The following is therefore reported back via the process data:
   0x60p0:01 [▶ 145] "Warning" = TRUE and
   0x60p0:0E [▶ 145] "TxPDO State" = TRUE
   The axis cannot be moved via the NC as the position data is invalid.
2. Set command 0x8008 "Suppress TxPDO State".
   The TxPDO State is suppressed until the distance-coded reference marks have been passed. If the axis is released, it can now be moved with the NC.
   ⇨ If "Reference mark" = TRUE is reported in the "FB EnDat Diag data Ch. p" object index 0xA0p8:04 [▶ 148], homing is complete.
3. Command 0x8001 [▶ 139] "Encoder reset" is used to reset the encoder. The "Warning flag" in the process data 0x60xp0:01 "Warning" is also reset.
4. Command 0x8006 [▶ 139] "EnDat read Pos2, calculate and write offset" is used to write the calculated offset to object "EnDat Settings Ch. p" Index 0x80p8:12 [▶ 144] "Offset value".
   ⇨ As soon as the status of the currently executed command object in index 0xB0p:02 [▶ 145] "Status" = 1, the command has been accepted without errors.

⇨ The position value in index 0x60p:11 "Position" is now absolute.

9.5.6 Command 0x9000 - Read EnDat Word

Proceed as follows for the command 0x9000 "Read EnDat Word":

1. **Write request**
   Enter the 5 byte header via the FB EnDat Command object, index 0xB0p8:01 [▶ 145] "Request" as follows. (dependent on the number of channels with p = 0 for channel 1 and p = 1 for channel 2).
   0: LOBYTE command
   1: HIBYTE command
   2: MRS code
   3: Offset
   4: Length in Words
2. **Read status 0xB0p8:02**:
   A valid response is present if this is 0x01.
3. **Read response**:
   (5 byte Header + 1 to 32 Word data)
   0: Acknowledgement LOBYTE command
   1: Acknowledgement HIBYTE command
   2: Acknowledgement MRS code
   3: Acknowledgement offset
   4: Acknowledgement length in Words
   5: LOBYTE Word 0
   6: HIBYTE Word 0
   ....
   5+(n*2) LOBYTE Word n
   5+(n*2) HIBYTE Word n

CoE Entry / Example for command 0x9000

Example read encoder designation

Command read: 0x9000
MRS code: "Parameter of the encoder manufacturer for EnDat2.2": 0xBD
Offset Word 17: 0x11
Length: 0x04

1. Write request 0xB008:01: "00 90 BD 11 04"

B008:0	FB EnDat Command Ch.1	RO	> 3 <
B008:01	Request	RW	00 90 BD 11 04
B008:02	Status	RO	0x01 (1)
B008:03	Response	RO	00 90 BD 11 04 45 43 4E 20 31 31 32 33

Fig. 146: Example command 0x9000 Read encoder designation

2. Read status 0xB008:02: A valid response is present if this is 0x01.
3. Read Response
   5 byte acknowledgement Header + 4 Word data
   0x45 = ASCII "E"
   0x43 = ASCII "C"
   0x4E = ASCII "N"
   0x20 = ASCII " "
   0x31 = ASCII "1"
   0x31 = ASCII "1"
   0x32 = ASCII "2"
   0x33 = ASCII "3"

⇨ The encoder designation is "ECN 1123"

9.5.7 Command 0x9001 - Write EnDat Word

Proceed as follows for the command 0x9001 "Write EnDat Word":

1. **Write request**
   Enter the 5 byte Header and 1 to 32 Word data via the FB EnDat Command object, index 0xB0p8:01 [▶ 145] "Request" as follows. (dependent on the number of channels with p = 0 for channel 1 and p = 1 for channel 2).
   0: LOBYTE Command
   1: HIBYTE Command
   2: MRS Code
   3: Offset
   4: Length in Words
   5: LOBYTE Word 0
   6: HIBYTE Word 0
   ...
   5+(n*2) LOBYTE Word n
   5+(n*2) HIBYTE Word n
2. **Read status 0xB0p8:02**:
   The data have been successfully written if this is 0x01.
3. **Read response**:
   (5 byte Header + 1 to 32 Word data)
   0: Acknowledgement LOBYTE command
   1: Acknowledgement HIBYTE command
   2: Acknowledgement MRS code
   3: Acknowledgement Offset
   4: Acknowledgement Length in Words

CoE Entry / Example for command 0x9001

Example: Write encoder address

Write command: 0x9001
MRS code: "Operating parameters": 0xA7
Offset Word 04: 0x04
Length: 0x01
New encoder address: 0x1234

1. Write request 0xB008:01: "01 90 A7 04 01 34 12"

B008:0	FB EnDat Command Ch.1	RO	> 3 <
B008:01	Request	RW	01 90 A7 04 01 34 12
B008:02	Status	RO	0x01 (1)
B008:03	Response	RO	01 90 A7 04 01 34 12

Fig. 147: Example command 0x9001 Write encoder address

2. Read status 0xB008:02: If this is 0x01, the data was written successfully.
3. Read response
   5 byte acknowledgement Header + 1 Word data
   0x34 = LOBYTE of "0x1234"
   0x12 = HIBYTE of "0x1234"

9.6 EL5032-00x0 - Object description and parameterization

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 [▶ 111] (double-click on the respective object) or via the Process Data tab [▶ 108](allocation of PDOs). Please note the following general CoE notes [▶ 28] 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 [▶ 187]” for resetting changes

9.6.1 Restore object

Index 1011 Restore default parameters

Index (hex)	Name	Meaning	Data type	Flags	Default
1011:0	Restore default parameters [▶ 187]	Restore default parameters	UINT8	RO	0x01 (1_{dec})
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 (0_{dec})

9.6.2 Configuration data

Index 80p0 FB Settings (dependent on the number of channels for Ch.1, p = 0; Ch.2, p = 1)

Index (hex)	Name	Meaning	Data type	Flags	Default
80p0:0	FB Settings	Max. subindex	UINT8	RO	> 17 <
80p0:01	Invert feedback direction	TRUE: Negates the 64-bit position value	BOOLEAN	RW	FALSE
80p0:11	Device type	02: EnDat	UINT32	RW	0x00000002 (2_{dec})

Index 80p8 FB EnDat settings (dependent on the number of channels for Ch.1, p = 0; Ch.2, p = 1)

Index (hex)	Name	Meaning	Data type	Flags	Default
80p8:0	FB EnDat settings	Max. subindex	UINT8	RO	> 24 <
80p8:01	Invert feedback direction	TRUE: Negates the 64-bit position value	BOOLEAN	RW	FALSE
80p8:12	Offset value	Value that will be added to the EnDat position value	INT64	RW	00 00 00 00 00 00 00 00
80p8:13	Supply voltage	Encoder supply voltage: 50: 5.0 V 90: 9.0 V (see note below)	UINT8	RW	5.0 V (50)
80p8:15*)	Velocity numerator	Velocity numerator value Velocity scaling	UINT32	RW	0x00000064 (100_{dec})
80p8:16*)	Velocity denominator	Velocity denominator value Velocity scaling	UINT32	RW	0x00000001 (1_{dec})
80p8:17*)	Velocity window	Minimum time over which the velocity is determined. Default value: 10 ms [resolution 1 µs]. Multiples of the cycle time are rounded up to the next larger value	UINT32	RW	0x00002710 (10000_{dec})
80p8:18*)	Velocity filter	0: None 1: IIR1 2: IIR2 3: IIR3 4: IIR4 5: IIR5 6: IIR6 7: IIR7 8: IIR8			
*) EL5032: from Firmware 16, Revision 28 / EL5032-0090 from Revision 17					

Reading or writing values of the EndDat encoder

If the configuration and information data access specific memory areas of the Endat encoder, make sure that the encoder is connected and supplied with voltage to read or write to these memory areas.
If no communication can be established with the EnDat encoder, an error message "Data cannot be transferred or stored to the application" is displayed.

Setting the encoder supply voltage

To write 0x8008:13 "Supply Voltage", the value 0x72657375 (ASCII:"user") must be set in 0xF008 [▶ 149] "Code word". Specification in 0.1 V.
Only the values 50 (5.0 V) and 90 (9.0 V) are permissible. This setting applies to both channels; therefore, before switching to 9.0 V, it must be ensured that both EnDat encoders support the extended voltage range! The setting is only adopted in the "INIT" state.
The encoder supply voltage is set for both channels in object 0x80p8!

9.6.3 Command object

Index B0p8 FB ENDat Command (dependent on the number of channels for Ch.1, p = 0; Ch.2, p = 1)

Index (hex)	Name	Meaning	Data type	Flags	Default
B0p8:0	FB ENDat Command	Max. subindex	UINT8	RO	> 3 <
B0p8:01	Request	The Object command can be used to initiate various actions, such as the reading or writing of a particular memory area of the EnDat encoder. Request: up to 69 bytes see Commands [▶ 132]	OCTET-STRING[69]	RW	
B0p8:02	Status	Status of the command currently being executed 0: Command executed without error and without response 1: Command executed without error and with response 2: Command executed with error and without response 3: Command executed with error and with response 255: Command is being executed	UINT8	RO	0x00 (0_{dec})
B0p8:03	Response	Optional return value of the command Response: up to 69 bytes	OCTET-STRING[69]	RO	

9.6.4 Input data

Index 60p0 FB Inputs (dependent on the number of channels for Ch.1, p = 0; Ch.2, p = 1)

Index (hex)	Name	Meaning	Data type	Flags	Default
60p0:0	FB Inputs	Max. subindex	UINT8	RO	> 18 <
60p0:01	STATUS_Warning	Warning bit of the EnDat protocol	BOOLEAN	RO	FALSE
60p0:02	STATUS_Error	Error collecting bit of the EnDat protocol (error message 1/ error message 2)	BOOLEAN	RO	FALSE
60p0:03	STATUS_Ready	Ready for use (initialization of the encoder completed)	BOOLEAN	RO	FALSE
60p0:0D	STATUS_Diag	Indicates that a new message is available in the "Diag History"	BOOLEAN	RO	TRUE
60p0:0E	STATUS_TxPDO State	Validity of the data of the associated TxPDO (0 = valid, 1 = invalid)	BOOLEAN	RO	TRUE
60p0:0F	STATUS_Input cycle counter	2-bit counter for synchronization (incremented only if a new value is present)	BIT2	RO	0
60p0:11	Position	EnDat position value EL5032:0090: ULINT values cannot be processed in the TwinSAFE logic, therefore there is no possibility to transmit the position value 0x60n0:11 [▶ 145] "Position" to the TwinSAFE logic. In this case the process data must be parameterized under 0x8021 [▶ 164] "TSC Process Data settings".	UINT64	RO	00 00 00 00 00 00 00 00
60p0:12	Position (uint32)	EnDat position value (unit32)	UINT32	RO	00 00 00 00

Index 60p8 FB Inputs (dependent on the number of channels for Ch.1, p = 0; Ch.2, p = 1) (EL5032 from FW 16, Rev.28 / EL5032-0090 from Revision 17)

Index (hex)	Name	Meaning	Data type	Flags	Default
60p8:0	FB ENDat Inputs Ch(n+1)	Max. subindex	UINT8	RO	> 19 <
60p8:12	Velocity	Velocity	DINT32	RO	0x00000000 (0_{dec})
60p8:13	Velocity (int 16)	Velocity	DINT16	RO	0x0000 (0_{dec})

Also see about this

- EL5032-0090 - TwinSAFE Single Channel objects [▶ 163]

9.6.5 Information data

(Electronic type plate and list of the supported errors and warnings)

Index 90p8 FB EnDat Info data (dependent on the number of channels for Ch.1, p = 0; Ch.2, p = 1)

Index (hex)	Name	Meaning	Data type	Flags	Default
90p8:0	FB ENDat Info data	Max. subindex	UINT8	RO	> 81 <
90p8:01	Encoder type	0: "Linear encoder" 1: "Rotary/angle encoder" 2: "EIB" (External interface Box)	BIT2	RO	Linear encoder (0_{dec})
90p8:03	Reference system	0: "Incremental" 1: "Absolute"	BIT1	RO	Incremental (0_{dec})
90p8:11	Encoder designation	Encoder designation, e.g.: LC181	STRING	RO	
90p8:12	Encoder ident number	Encoder ident number	STRING	RO	
90p8:13	Encoder serial number	Encoder serial number	STRING	RO	
90p8:14	Multiturn resolution	Number of distinguishable revolutions e.g: 4096 (corresponds to 12 bit)	UINT32	RO	0x00000000 (0_{dec})
90p8:15	Singleturn resolution	“Rotary/angle encoder” only: number of measuring steps per revolution	UINT32	RO	0x00000000 (0_{dec})
90p8:16	Measuring step	“Linear encoder” only: step size in nm	UINT32	RO	0x00000000 (0_{dec})
90p8:21	Light source error supported	Corresponding diagnostic entry in 0xA0p8 [▶ 148] FALSE: not supported TRUE: supported	BOOLEAN	RO	FALSE
90p8:22	Signal amplitude error supported				
90p8:23	Position error supported				
90p8:24	Overvoltage error supported				
90p8:25	Undervoltage error supported				
90p8:26	Overcurrent error supported				
90p8:27	Battery error supported				
90p8:31	Frequency warning supported				
90p8:32	Temperature warning supported				
90p8:33	Light source warning supported				
90p8:34	Battery warning supported				
90p8:35	Reference Mark warning supported				
90p8:36	Acyclic mode warning supported				
90p8:37	Limit position warning supported				
90p8:38	Not ready warning supported				
90p8:39	Diagnostic warning supported				
90p8:43	Temperature 1 (extern) supported				
90p8:44	Temperature 2 (intern) supported				
90p8:51	Clock pulse periods	Number of position bits	UINT16	RO	0x0000 (0_{dec})

Reading or writing values of the EndDat encoder

If the configuration and information data access specific memory areas of the Endat encoder, make sure that the encoder is connected and supplied with voltage to read or write to these memory areas.
If no communication can be established with the EnDat encoder, an error message "Data cannot be transferred or stored to the application" is displayed.

9.6.6 Diagnostic data

Index 10F3 Diagnosis History

Index (hex)	Name	Meaning	Data type	Flags	Default
10F3:0	Diagnosis History	Maximum subindex	UINT8	RO	> 55 <
10F3:01	Maximum Messages	Maximum number of stored messages A maximum of 50 messages can be stored	UINT8	RO	0x32 (50_{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	FALSE
10F3:05	Flags	not used	UINT16	RW	0x0000 (0_{dec})
10F3:06	Diagnosis Message 001	Message 1	OCTET-STRING[28]	RO	
...	...	...			
10F3:37	Diagnosis Message 050	Message 50	OCTET-STRING[28]	RO	

Index 10F8 Actual Time Stamp

Index (hex)	Name	Meaning	Data type	Flags	Default
10F8:0	Actual Time Stamp	Time stamp	UINT64	RO	

Index A0p8 FB EnDat Diag data (dependent on the number of channels for Ch.1, p = 0; Ch.2, p = 1)

Index (hex)	Name	Meaning	Data type	Flags	Default
A0p8:0	FB EnDat Diag data	Max. subindex	UINT8	RO	> 67 <
A0p8:01	Warning	Warning bit of the EnDat protocol • FALSE: no warning • TRUE: indicates that a warning has occurred (collective message)	BOOLEAN	RO	FALSE
A0p8:02	Error 1	Error 2 bit of the EnDat protocol • FALSE: no error message 1 • TRUE: indicates that an error message 1 has occurred (collective message)	BOOLEAN	RO	FALSE
A0p8:03	Error 2 (active low)	Error 2 bit of the EnDat protocol • FALSE: no error message 2 • TRUE: indicates that an error message 2 has occurred (collective message)	BOOLEAN	RO	FALSE
A0p8:04	Reference Mark	• FALSE: homing not completed • TRUE: homing completed. Already set when switching on in case of absolute encoders	BOOLEAN	RO	FALSE
A0p8:11	CRC Error Position counter	Incremented with each "CRC Error Position" Reset on reinitialization	UINT16	RW	0x0000 (0_{dec})
A0p8:12	CRC Error Z1 counter	Incremented with each "CRC error additional information 1" Reset on reinitialization	UINT16	RW	0x0000 (0_{dec})
A0p8:13	CRC Error Z2 counter	Incremented with each "CRC error additional information 2" Reset on reinitialization	UINT16	RW	0x0000 (0_{dec})
A0p8:21	Light source error	Lighting • FALSE: OK • TRUE: failure	BOOLEAN	RO	FALSE
A0p8:22	Signal amplitude error	Signal amplitude • FALSE: OK • TRUE: error	BOOLEAN	RO	FALSE
A0p8:23	Position error	Position • FALSE: OK • TRUE: error	BOOLEAN	RO	FALSE
A0p8:24	Overvoltage error	Overvoltage • FALSE: none • TRUE: yes	BOOLEAN	RO	FALSE
A0p8:25	Undervoltage error	Undervoltage • FALSE: none • TRUE: supply undervoltage	BOOLEAN	RO	FALSE
A0p8:26	Overcurrent error	Overcurrent • FALSE: no • TRUE: yes	BOOLEAN	RO	FALSE
A0p8:27	Battery error	Battery • FALSE: OK • TRUE: change required	BOOLEAN	RO	FALSE
A0p8:31	Frequency warning	Frequency collision • FALSE: none • TRUE: yes	BOOLEAN	RO	FALSE
A0p8:32	Temperature warning	Overtemperature • FALSE: none • TRUE: yes	BOOLEAN	RO	FALSE
A0p8:33	Light source warning	Lighting control reserve • FALSE: not achieved • TRUE: achieved	BOOLEAN	RO	FALSE

Index (hex)	Name	Meaning	Data type	Flags	Default
A0p8:34	Battery warning	Battery charge level • FALSE: OK • TRUE: too low	BOOLEAN	RO	FALSE
A0p8:35	Reference mark warning	Reference point • FALSE: not achieved • TRUE: achieved	BOOLEAN	RO	FALSE
A0p8:36	Acyclic mode warning	Cyclic operation • FALSE: cyclic • TRUE: acyclic	BOOLEAN	RO	FALSE
A0p8:37	Limit position warning	Limit position • FALSE: not achieved • TRUE: achieved	BOOLEAN	RO	FALSE
A0p8:38	Not ready warning	Readiness • FALSE: yes • TRUE: none	BOOLEAN	RO	FALSE
A0p8:39	Diagnostic warning	Diagnosis • FALSE: OK • TRUE: threshold value exceeded	BOOLEAN	RO	FALSE
A0p8:41	Temperature sensor 1 (extern)	Temperature sensor 1 (0x8000 is output if not supported) Unit: 0.1 °C	INT16	RO	0x8000 (-32768_{dec})
A0p8:42	Temperature sensor 2 (extern)	Temperature sensor 1 (0x8000 is output if not supported) Unit: 0.1 °C	INT16	RO	0x8000 (-32768_{dec})
A0p8:43	Position raw value	Position value without inversion and offset	UINT64	RO	00 00 00 00 00 00 00 00

9.6.7 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 CoE profile used (5001). The Hi-Word contains the module profile according to the modular device profile.	UINT32	RO	EL5032: 0x02011389 (5001_{dec}) EL5032-0090: 0x00001389 (33624969_{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	EL5032 EL5032-0090

Index 1009 Hardware version

Index (hex)	Name	Meaning	Data type	Flags	Default
1009:0	Hardware version	Hardware version of the EtherCAT slave	STRING	RO	01

Index 100A Software version

Index (hex)	Name	Meaning	Data type	Flags	Default
100A:0	Software version	Firmware version of the EtherCAT slave	STRING	RO	09

Index 1018 Identity

Index (hex)	Name	Meaning	Data type	Flags	Default
1018:0	Identity	Information for identifying the slave	UINT8	RO	> 4 <
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	0x13A83052 (329789522_{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	0x00130000 (1245184_{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 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	> 1 <
10F0:01	Checksum	Checksum across all backup entries of the EtherCAT slave	UINT32	RO	0x00000D06 (3334_{dec})

Index 1800 FB TxPDO-Par Inputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1800:0	FB TxPDO-Par Inputs Ch.1	PDO parameter TxPDO 1	UINT8	RO	> 6 <
1800:06	Exclude TxPDOs	Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 1	OCTET-STRING[2]	RO	02 1A

Index 1801 FB TxPDO-Par Inputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1801:0	FB TxPDO-Par Inputs Ch.2	PDO parameter TxPDO 2	UINT8	RO	> 6 <
1801:06	Exclude TxPDOs	Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 2	OCTET-STRING[2]	RO	03 1A

Index 1802 FB TxPDO-Par Inputs Ch.1 compact

Index (hex)	Name	Meaning	Data type	Flags	Default
1802:0	FB TxPDO-Par Inputs Ch.1 compact	PDO parameter TxPDO 3	UINT8	RO	> 6 <
1802:06	Exclude TxPDOs	Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 3	OCTET-STRING[2]	RO	00 1A

Index 1803 FB TxPDO-Par Inputs Ch.2 compact

Index (hex)	Name	Meaning	Data type	Flags	Default
1803:0	FB TxPDO-Par Inputs Ch.2 compact	PDO parameter TxPDO 4	UINT8	RO	> 6 <
1803:06	Exclude TxPDOs	Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 4	OCTET-STRING[2]	RO	01 1A

Index 1A00 FB TxPDO-Map Inputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1A00:0	FB TxPDO-Map Inputs Ch.1	PDO Mapping TxPDO 1	UINT8	RO	> 9 <
1A00:01	SubIndex 001	1. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x01 (Warning))	UINT32	RO	0x6000:01 [▶ 145], 1
1A00:02	SubIndex 002	2. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x02 (Error))	UINT32	RO	0x6000:02 [▶ 145], 1
1A00:03	SubIndex 003	3. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x03 (Ready))	UINT32	RO	0x6000:03 [▶ 145], 1
1A00:04	SubIndex 004	4. PDO Mapping entry (5 bits align)	UINT32	RO	0x0000:00, 5
1A00:05	SubIndex 005	5. PDO Mapping entry (4 bits align)	UINT32	RO	0x0000:00, 4
1A00:06	SubIndex 006	6. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x0D (Diag))	UINT32	RO	0x6000:0D [▶ 145], 1
1A00:07	SubIndex 007	7. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x0E (TxPDO State))	UINT32	RO	0x6000:0E [▶ 145], 1
1A00:08	SubIndex 008	8. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x0F (Input cycle counter))	UINT32	RO	0x6000:0F [▶ 145], 2
1A00:09	SubIndex 009	9. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x11 (Position))	UINT32	RO	0x6000:11 [▶ 145], 64

Index 1A01 FB TxPDO-Map Inputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1A01:0	FB TxPDO-Map Inputs Ch.2	PDO Mapping TxPDO 2	UINT8	RO	> 9 <
1A01:01	SubIndex 001	1. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x01 (Warning))	UINT32	RO	0x6010:01 [▶ 145], 1
1A01:02	SubIndex 002	2. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x02 (Error))	UINT32	RO	0x6010:02 [▶ 145], 1
1A01:03	SubIndex 003	3. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x03 (Ready))	UINT32	RO	0x6010:03 [▶ 145], 1
1A01:04	SubIndex 004	4. PDO Mapping entry (5 bits align)	UINT32	RO	0x0000:00, 5
1A01:05	SubIndex 005	5. PDO Mapping entry (4 bits align)	UINT32	RO	0x0000:00, 4
1A01:06	SubIndex 006	6. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x0D (Diag))	UINT32	RO	0x6010:0D [▶ 145], 1
1A01:07	SubIndex 007	7. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x0E (TxPDO State))	UINT32	RO	0x6010:0E [▶ 145], 1
1A01:08	SubIndex 008	8. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x0F (Input cycle counter))	UINT32	RO	0x6010:0F [▶ 145], 2
1A01:09	SubIndex 009	9. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x11 (Position))	UINT32	RO	0x6010:11 [▶ 145], 64

Index 1A02 FB TxPDO-Map Inputs Ch.1 compact

Index (hex)	Name	Meaning	Data type	Flags	Default
1A02:0	FB TxPDO-Map Inputs Ch.1 compact	PDO Mapping TxPDO 3	UINT8	RO	> 9 <
1A02:01	SubIndex 001	1. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x01 (Warning))	UINT32	RO	0x6000:01 [▶ 145], 1
1A02:02	SubIndex 002	2. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x02 (Error))	UINT32	RO	0x6000:02 [▶ 145], 1
1A02:03	SubIndex 003	3. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x03 (Ready))	UINT32	RO	0x6000:03 [▶ 145], 1
1A02:04	SubIndex 004	4. PDO Mapping entry (5 bits align)	UINT32	RO	0x0000:00, 5
1A02:05	SubIndex 005	5. PDO Mapping entry (4 bits align)	UINT32	RO	0x0000:00, 4
1A02:06	SubIndex 006	6. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x0D (Diag))	UINT32	RO	0x6000:0D [▶ 145], 1
1A02:07	SubIndex 007	7. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x0E (TxPDO State))	UINT32	RO	0x6000:0E [▶ 145], 1
1A02:08	SubIndex 008	8. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x0F (Input cycle counter))	UINT32	RO	0x6000:0F [▶ 145], 2
1A02:09	SubIndex 009	9. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x11 (Position))	UINT32	RO	0x6000:11 [▶ 145], 32

Index 1A03 FB TxPDO-Map Inputs Ch.2 compact

Index (hex)	Name	Meaning	Data type	Flags	Default
1A03:0	FB TxPDO-Map Inputs Ch.2 compact	PDO Mapping TxPDO 4	UINT8	RO	> 9 <
1A03:01	SubIndex 001	1. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x01 (Warning))	UINT32	RO	0x6010:01 [▶ 145], 1
1A03:02	SubIndex 002	2. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x02 (Error))	UINT32	RO	0x6010:02 [▶ 145], 1
1A03:03	SubIndex 003	3. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x03 (Ready))	UINT32	RO	0x6010:03 [▶ 145], 1
1A03:04	SubIndex 004	4. PDO Mapping entry (5 bits align)	UINT32	RO	0x0000:00, 5
1A03:05	SubIndex 005	5. PDO Mapping entry (4 bits align)	UINT32	RO	0x0000:00, 4
1A03:06	SubIndex 006	6. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x0D (Diag))	UINT32	RO	0x6010:0D [▶ 145], 1
1A03:07	SubIndex 007	7. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x0E (TxPDO State))	UINT32	RO	0x6010:0E [▶ 145], 1
1A03:08	SubIndex 008	8. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x0F (Input cycle counter))	UINT32	RO	0x6010:0F [▶ 145], 2
1A03:09	SubIndex 009	9. PDO Mapping entry (object 0x6010 (FB Inputs Ch.2), entry 0x11 (Position))	UINT32	RO	0x6010:11 [▶ 145], 32

Index 1A04 FB TxPDO-Map Velocity Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1A04:0	FB TxPDO-Map Velocity Ch.1	PDO Mapping TxPDO 5	UINT8	RO	> 1 <
1A04:01	SubIndex 001	1. PDO Mapping entry (object 0x6008 (FB EnDat Inputs Ch.1), entry 0x12 (Velocity))	UINT32	RO	0x6008:12, 32

Index 1A05 FB TxPDO-Map Velocity Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1A05:0	FB TxPDO-Map Velocity Ch.2	PDO Mapping TxPDO 6	UINT8	RO	> 1 <
1A05:01	SubIndex 001	1. PDO Mapping entry (object 0x6018 (FB EnDat Inputs Ch.2), entry 0x12 (Velocity))	UINT32	RO	0x6018:12, 32

Index 1C00 Sync manager type

Index (hex)	Name	Meaning	Data type	Flags	Default
1C00:0	Sync manager type	Using the sync managers	UINT8	RO	> 4 <
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	> 0 <

Index 1C12 RxPDO assign (EL5032-0090)

Index (hex)	Name	Meaning	Data type	Flags	Default
1C12:0	RxPDO assign	PDO Assign Outputs	UINT8	RW	> 1 <
1C12:1	SubIndex 001	1. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x1610 (5648)

Index 1C13 TxPDO assign

Index (hex)	Name	Meaning	Data type	Flags	Default
1C13:0	TxPDO assign	PDO Assign Inputs	UINT8	RW	> 2 <
1C13:01	Subindex 001	1. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A00 (6656_{dec})
1C13:02	Subindex 002	2. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A01 (6657_{dec})
1C13:03*	Subindex 003	3. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A10 (6672_{dec})

*) EL5032-0090 only

Index 1C32 SM output parameter (EL5032-0090 only)

Index (hex)	Name	Meaning	Data type	Flags	Default
1C32:0	SM output parameter	Synchronization parameters for the outputs	UINT8	RO	> 32 <
1C32:01	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	0x0001 (1_{dec})
1C32:02	Cycle time	Cycle time (in ns): • Free Run: Cycle time of the local timer • Synchron with SM 2 Event: Master cycle time • DC mode: SYNC0/SYNC1 Cycle Time	UINT32	RW	0x000F4240 (1000000_{dec})
1C32:03	Shift time	Time between SYNC0 event and reading of the inputs (in ns, only DC mode)	UINT32	RO	0x00000000 (0_{dec})
1C32:04	Sync modes supported	Supported synchronization modes: • 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 0x1C32:08 [▶ 154])	UINT16	RO	0x0807 (2055_{dec})
1C32:05	Minimum cycle time	Minimum cycle time (in ns)	UINT32	RO	0x000186A0 (100000_{dec})
1C32:06	Calc and copy time	Time between reading of the inputs and availability of the inputs for the master (in ns, only DC mode)	UINT32	RO	0x00000000 (0_{dec})
1C32:07	Minimum delay time		UINT32	RO	0x00000000 (0_{dec})
1C32:08	Command	• 0: Measurement of the local cycle time is stopped • 1: Measurement of the local cycle time is started The entries 0x1C32:03, 0x1C33:06, 0x1C33:09 are updated with the maximum measured values. For a subsequent measurement the measured values are reset	UINT16	RW	0x0000 (0_{dec})
1C32:09	Maximum delay time	Time between SYNC1 event and reading of the inputs (in ns, only DC mode)	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 occasions that the interval between SYNC0 and SYNC1 event was 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	> 32 <
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	Cycle time (in ns): • Free Run: Cycle time of the local timer • Synchron with SM 2 Event: Master cycle time • DC mode: SYNC0/SYNC1 Cycle Time	UINT32	RW	0x000F4240 (1000000_{dec})
1C33:03	Shift time	Time between SYNC0 event and reading of the inputs (in ns, only DC mode)	UINT32	RO	0x00000000 (0_{dec})
1C33:04	Sync modes supported	Supported synchronization modes: • 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 0x1C33:08 [▶ 155])	UINT16	RO	0x0807 (2055_{dec})
1C33:05	Minimum cycle time	Minimum cycle time (in ns)	UINT32	RO	0x000186A0 (100000_{dec})
1C33:06	Calc and copy time	Time between reading of the inputs and availability of the inputs for the master (in ns, only DC mode)	UINT32	RO	0x00000000 (0_{dec})
1C33:07	Minimum delay time		UINT32	RO	0x00000000 (0_{dec})
1C33:08	Command	• 0: Measurement of the local cycle time is stopped • 1: Measurement of the local cycle time is started The entries 0x1C33:03, 0x1C33:06, 0x1C33:09 are updated with the maximum measured values. For a subsequent measurement the measured values are reset	UINT16	RW	0x0000 (0_{dec})
1C33:09	Maximum delay time	Time between SYNC1 event and reading of the inputs (in ns, only DC mode)	UINT32	RO	0x00000000 (0_{dec})
1C33:0B	SM event missed counter	Number of missed SM events in OPERATIONAL (DC mode only)	UINT16	RO	0x0000 (0_{dec})
1C33:0C	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})
1C33:0D	Shift too short counter	Number of occasions that the interval between SYNC0 and SYNC1 event was too short (DC mode only)	UINT16	RO	0x0000 (0_{dec})
1C33:20	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 F000 Modular device profile

Index (hex)	Name	Meaning	Data type	Flags	Default
F000:0	Modular device profile	General information for the modular device profile	UINT8	RO	> 2 <
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	EL5032: 0x0002 (2_{dec}) EL5032-0090: 0x0003 (3_{dec})

Index F008 Code word

Index (hex)	Name	Meaning	Data type	Flags	Default
F008:0	Code word	see object 0x80p8:13 [▶ 144]	UINT32	RW	0x00000000 (0_{dec})

Index F010 Module list

Index (hex)	Name	Meaning	Data type	Flags	Default
F010:0	Module list	Max. subindex	UINT8	RW	> 3 <
F010:01	SubIndex 001	MDP FB profiles 513	UINT32	RW	0x00000201 (513_{dec})
F010:02	SubIndex 002	MDP FB profiles 513	UINT32	RW	0x00000201 (513_{dec})
F010:03*	SubIndex 003	MDP FB profiles 950	UINT32	RW	0x000003B6 (950_{dec})

*) EL5032-0090 only

Index FB40 Memory interface

Index (hex)	Name	Meaning	Data type	Flags	Default
FB40:0	Memory interface	Max. subindex	UINT8	RO	> 3 <
FB40:01	Address	reserved	UINT32	RW	0x00000000 (0_{dec})
FB40:02	Length	reserved	UINT16	RW	0x0000 (0_{dec})
FB40:03	Data	reserved	OCTET-STRING[8]	RW	{0}

9.7 EL5032-0090 - TwinSAFE SC

The EL5032-0090 supports the full functionality of the EL5032 (please refer to chapter EL5032-00x0 - Commissioning [▶ 131]).

In addition, the EL5032-0090 supports TwinSAFE SC technology (TwinSAFE Single Channel)

9.7.1 TwinSAFE SC

9.7.1.1 TwinSAFE SC - operating principle

The TwinSAFE SC (Single Channel) technology enables the use of standard signals for safety tasks in any networks of fieldbuses. To do this, EtherCAT Terminals from the areas of analog input, angle/displacement measurement or communication (4…20 mA, incremental encoder, IO-Link, etc.) are extended by the TwinSAFE SC function. The typical signal characteristics and standard functionalities of the I/O components are retained. TwinSAFE SC I/Os have a yellow strip at the front of the housing to distinguish them from standard I/Os.

The TwinSAFE SC technology enables communication via a TwinSAFE protocol. These connections can be distinguished from the usual safe communication via Safety over EtherCAT.

The data of the TwinSAFE SC components are transferred via a TwinSAFE protocol to the TwinSAFE logic, where they can be used in the context of safety-relevant applications. Detailed examples for the correct application of the TwinSAFE SC components and the respective normative classification, which were confirmed/calculated by TÜV SÜD, can be found in the TwinSAFE application manual.

9.7.1.2 TwinSAFE SC - configuration

The TwinSAFE SC technology enables communication with standard EtherCAT terminals via the Safety over EtherCAT protocol. These connections use another checksum, in order to be able to distinguish between TwinSAFE SC and TwinSAFE. Eight fixed CRCs can be selected, or a free CRC can be entered by the user.

By default the TwinSAFE SC communication channel of the respective TwinSAFE SC component is not enabled. In order to be able to use the data transfer, the corresponding TwinSAFE SC module must first be added under the Slots tab. Only then is it possible to link to a corresponding alias device.

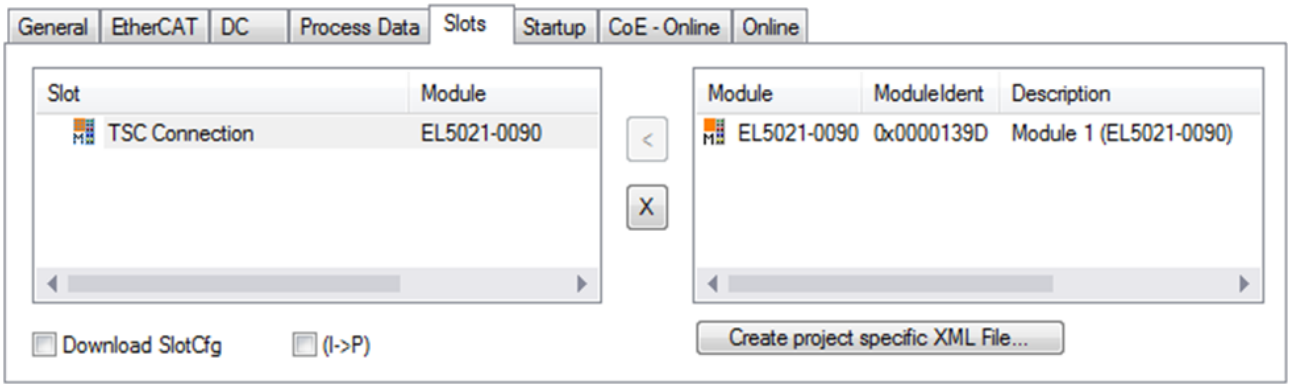


Fig. 148: Adding the TwinSAFE SC process data under the component, e.g. EL5021-0090

Additional process data with the ID TSC Inputs, TSC Outputs are generated (TSC - TwinSAFE Single Channel).

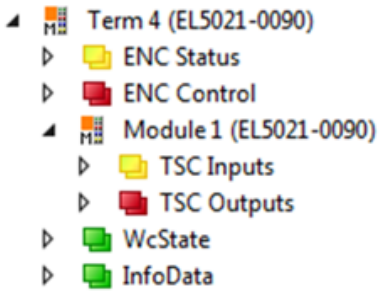


Fig. 149: TwinSAFE SC component process data, example EL5021-0090

A TwinSAFE SC connection is added by adding an alias devices in the safety project and selecting TSC (*TwinSAFE Single Channel*)

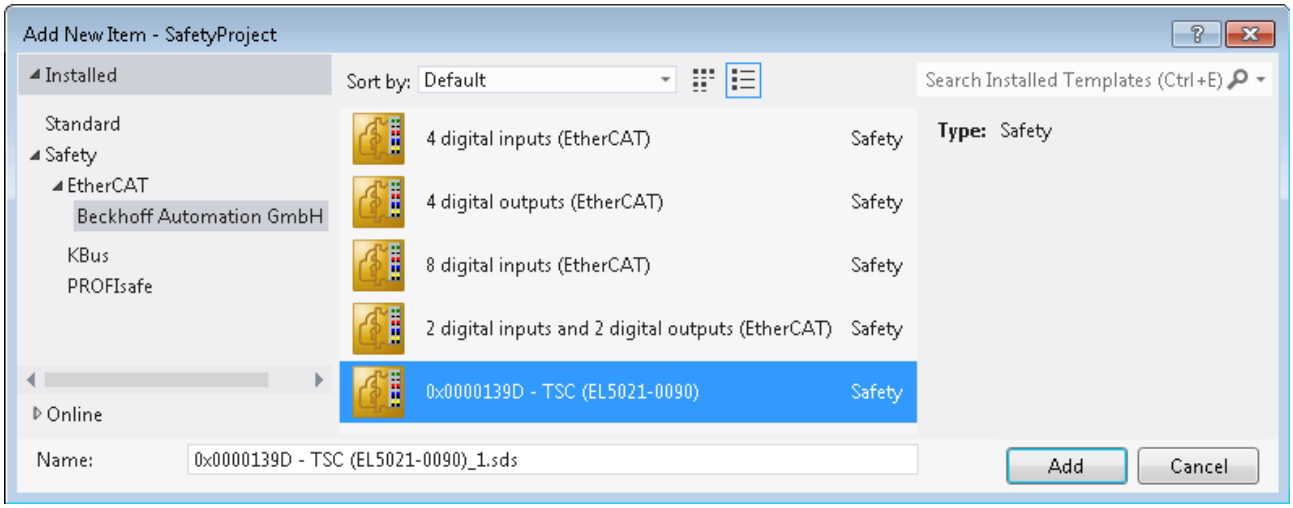


Fig. 150: Adding a TwinSAFE SC connection

After opening the alias device by double-clicking, select the Link button  next to *Physical Device,* in order to create the link to a TwinSAFE SC terminal. Only suitable TwinSAFE SC terminals are offered in the selection dialog.

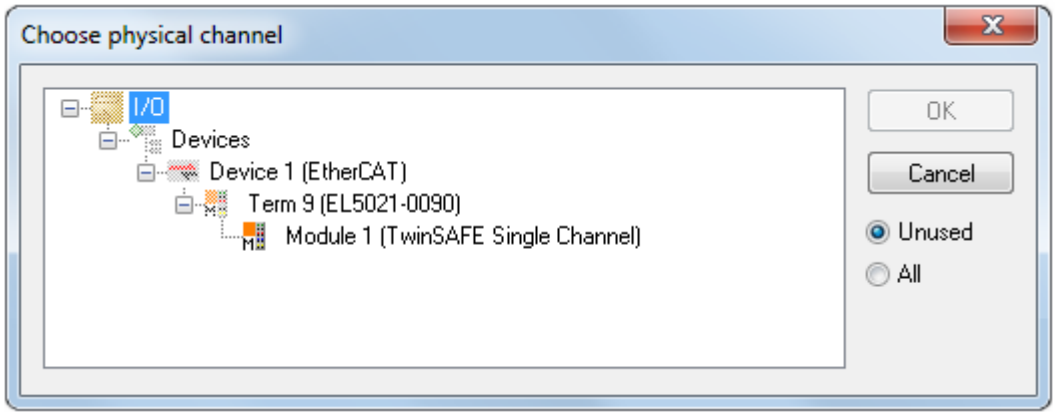


Fig. 151: Creating a link to TwinSAFE SC terminal

The CRC to be used can be selected or a free CRC can be entered under the Connection tab of the alias device.

Entry Mode	Used CRCs
TwinSAFE SC CRC 1 master	0x17B0F
TwinSAFE SC CRC 2 master	0x1571F
TwinSAFE SC CRC 3 master	0x11F95
TwinSAFE SC CRC 4 master	0x153F1
TwinSAFE SC CRC 5 master	0x1F1D5
TwinSAFE SC CRC 6 master	0x1663B
TwinSAFE SC CRC 7 master	0x1B8CD
TwinSAFE SC CRC 8 master	0x1E1BD

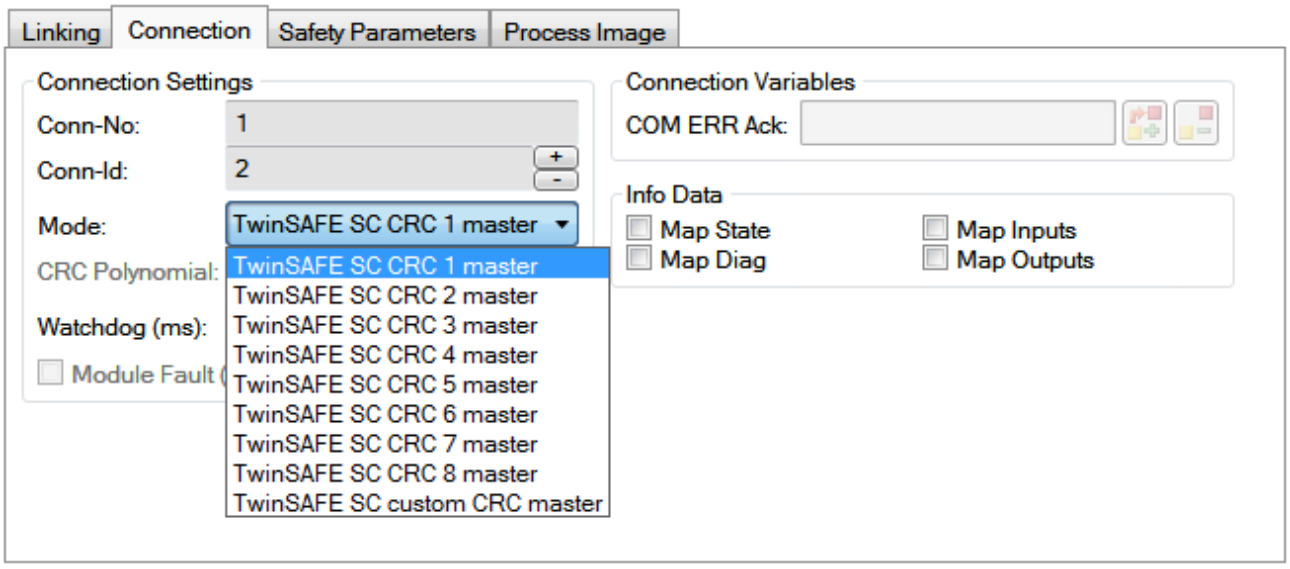


Fig. 152: Selecting a free CRC

These settings must match the settings in the CoE objects of the TwinSAFE SC component.
The TwinSAFE SC component initially makes all available process data available. The *Safety Parameters* tab typically contains no parameters. The process data size and the process data themselves can be selected under the *Process Image* tab.

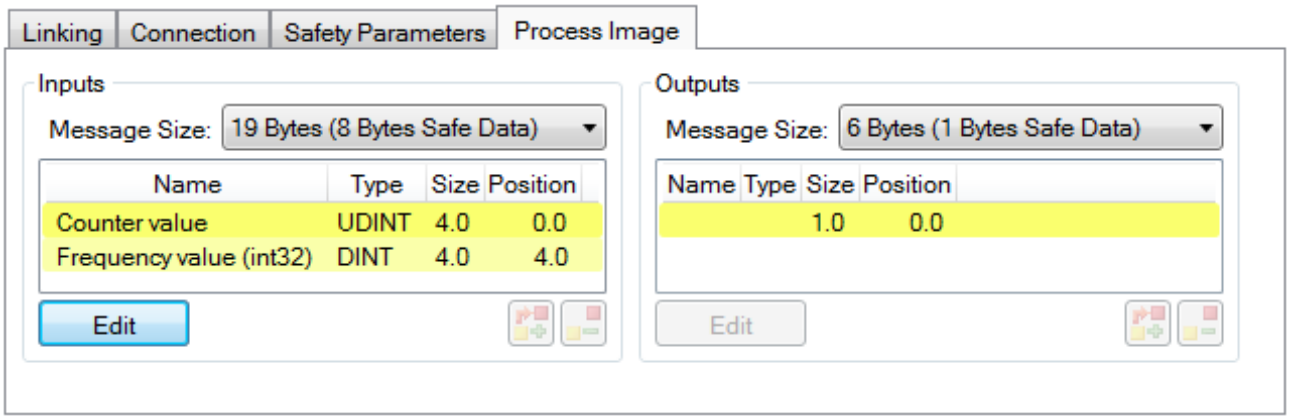


Fig. 153: Selecting the process data size and the process data

The process data (defined in the ESI file) can be adjusted to user requirements by selecting the *Edit* button in the dialog *Configure I/O element(s)*.

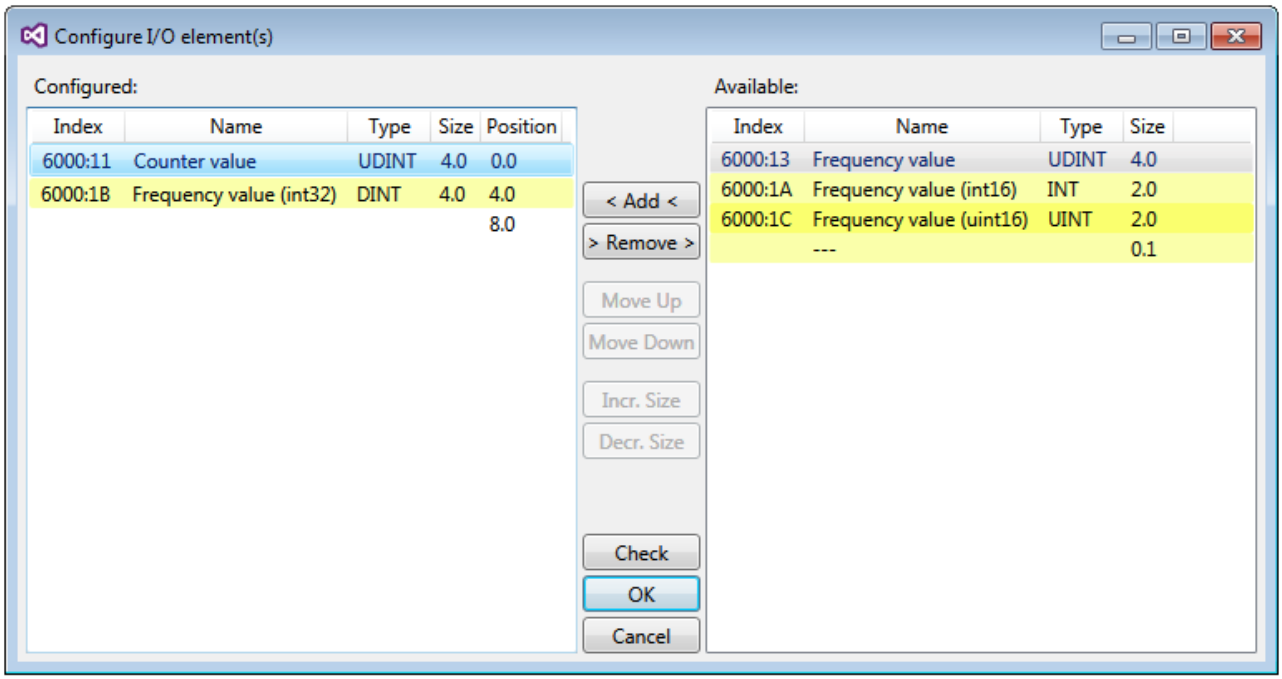


Fig. 154: Selection of the process data

The safety address together with the CRC must be entered on the TwinSAFE SC slave side. This is done via the CoE objects under *TSC settings* of the corresponding TwinSAFE SC component (here, for example, EL5021-0090, 0x8010: 01 and 0x8010: 02). The address set here must also be set in the *alias device* as *FSoE* address under the *Linking* tab.

Under the object 0x80n0:02 Connection Mode the CRC to be used is selected or a free CRC is entered. A total of 8 CRCs are available. A free CRC must start with 0x00ff in the high word.

8010:0	TSC Settings	RW	> 2 <
8010:01	Address	RW	0x0000 (0)
8010:02	Connection Mode	RW	TwinSAFE SC CRC1 master (97039)

Fig. 155: CoE objects 0x8010:01 and 0x8010:02

Object *TSC Settings*

Depending on the terminal, the index designation of the configuration object *TSC Settings* can vary.
Example:
- EL3214-0090 and EL3314-0090, TSC Settings, Index 8040
- EL5021-0090, TSC Settings, Index 8010
- EL6224-0090, TSC Settings, Index 800F

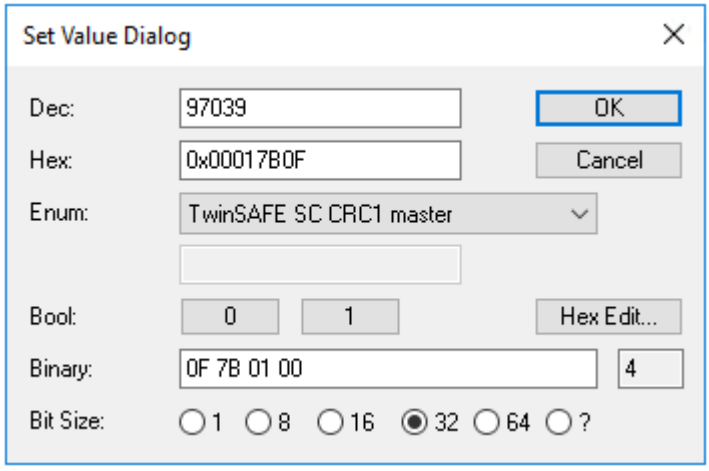


Fig. 156: Entering the safety address and the CRC

TwinSAFE SC connections

If several TwinSAFE SC connections are used within a configuration, a different CRC must be selected for each TwinSAFE SC connection.

9.7.2 EL5032-0090 - TwinSAFE SC process data

The EL5032-0090 transmits the following process data to the TwinSAFE logic:

ModuleIdent	Index (hex)	Name	Type	Size
0x000013A8 "2Ch. EnDat Encoder, TSC"	6000:12	Position (uint32)	UDINT	4.0
	6010:12	Position (uint32)	UDINT	4.0
0x010013A8 "2Ch. EnDat Velocitiy, TSC" (from Firmware 16, Revision 17)	6000:12	Position (uint32)	UDINT	4.0
	6010:12	Position (uint32)	UDINT	4.0
	6008:12	Velocity	UDINT	4.0
	6008:13	Velocity (int16)	INT	2.0
	6018:12	Velocity	UDINT	4.0
	6018:13	Velocity (int16)	INT	2.0

The position value "Position (uint32)" (0x60n0:12) of channels 1 and 2 is thereby transmitted by default.

The process data for the individual channels can be deselected on the "Process Image" tab in the Safety Editor.

If the ModuleIdent 0x010013A8 is selected, the "Velocity" can additionally be selected in the Safety Editor in the required data type.

Use of EnDat 2.2 encoders with position values >32 bits

ULINT values cannot be processed in the TwinSAFE logic, therefore there is no possibility to transmit the position value 0x60n0:11 [▶ 145] "Position" to the TwinSAFE logic.

- In this case the process data must be parameterized under 0x8021 [▶ 164] "TSC Process Data settings".

Parameterization of the TSC process data

Since ULINT values cannot be processed in the TwinSAFE logic, it may be necessary to parameterize the TwinSAFE SC process data when using EnDat 2.2 encoders that transmit a position value of > 32 bits. The parameterization of the process data and their effect must be taken into account accordingly in the safety application, especially with regard to counter/position value overflow.

The parameterization is done under 0x8021 [▶ 164] "TSC Process Data Settings". In the process, the number of position bits is automatically read from the EnDat protocol and can be taken from the object 0x90n8:51 [▶ 146] "Clock pulse periods".

- 0x8021:0n [▶ 164]: Auto align to MSB Ch.n
  - Default: the bit is not set.
    The position value is shifted to the right by the number of bits in 0x8021:1n "Shift right data Ch.n".
  - If the bit is set the position value is automatically shifted to the right to the max. MSB.
- 0x8021:1n [▶ 164]: Shift right data Ch.n
  - Default: 0
    The position value is shifted by 0 bits to the right.
  - An individual number of bits can be specified by which the process data is to be shifted to the right.

TwinSAFE SC objects

The overview of the TwinSAFE SC objects of the EL5032-0090 can be found in the chapter EL5032-0090 - TwinSAFE Single Channel objects [▶ 163].

9.7.3 EL5032-0090 - TwinSAFE Single Channel objects

Index 1610 TSC Master Message

Index (hex)	Name	Meaning	Data type	Flags	Default
1610:0	TSC RxPDO-Map Master Message	PDO Mapping RxPDO	UINT8	RO	0x04 (4_{dec})
1610:01	SubIndex 001	1. PDO Mapping entry (object 0x7020 (TSC Master Frame Elements), entry 0x01 (TSC__Master Cmd))	UINT32	RO	0x7020:01, 8
1610:02	SubIndex 002	2. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8
1610:03	SubIndex 003	3. PDO Mapping entry (object 0x7020 (TSC Master Frame Elements), entry 0x03 (TSC__Master CRC_0))	UINT32	RO	0x7020:03, 16
1610:04	SubIndex 004	4. PDO Mapping entry (object 0x7020 (TSC Master Frame Elements), entry 0x02 (TSC__Master ConnID))	UINT32	RO	0x7020:02, 16

Index 1A10 TSC Slave Message

Index (hex)	Name	Meaning	Data type	Flags	Default
1A10:0	TSC TxPDO-Map Slave Message	PDO Mapping TxPDO	UINT8	RW	0x0A (10_{dec})
1A10:01	SubIndex 001	1. PDO Mapping entry (object 0x6020 (TSC Slave Frame Elements), entry 0x01 (TSC__Slave Cmd))	UINT32	RW	0x6020:01, 8
1A10:02	SubIndex 002	2. PDO Mapping entry (object 0x6000 (FB Inputs Ch.1), entry 0x12 (Position (uint32)))	UINT32	RW	0x6000:12, 16
1A10:03	SubIndex 003	3. PDO Mapping entry (object 0x6020 (TSC Slave Frame Elements), entry 0x03 (TSC__Slave CRC_0))	UINT32	RW	0x6020:03, 16
1A10:04	SubIndex 004	4. PDO Mapping entry (16 bits align)	UINT32	RW	0x0000:00, 16
1A10:05	SubIndex 005	5. PDO Mapping entry (object 0x6020 (TSC Slave Frame Elements), entry 0x04 (TSC__Slave CRC_1))	UINT32	RW	0x6020:04, 16
1A10:06	SubIndex 006	6. PDO Mapping entry (object 0x6010 (FB Inputs Ch.1), entry 0x12 (Position (uint32)))	UINT32	RW	0x6010:12, 16
1A10:07	SubIndex 007	7. PDO Mapping entry (object 0x6020 (TSC Slave Frame Elements), entry 0x05 (TSC__Slave CRC_2))	UINT32	RW	0x6020:05, 16
1A10:08	SubIndex 008	8. PDO Mapping entry (16 bits align)	UINT32	RW	0x0000:00, 16
1A10:09	SubIndex 009	9. PDO Mapping entry (object 0x6020 (TSC Slave Frame Elements), entry 0x06 (TSC__Slave CRC_3))	UINT32	RW	0x6020:06, 16
1A10:0A	SubIndex 010	10. PDO Mapping entry (object 0x6010 (TSC Slave Frame Elements), entry 0x02 (TSC__Slave ConnID))	UINT32	RW	0x6020:02, 16

Index 6020 TSC Slave Frame Elements

Index (hex)	Name	Meaning	Data type	Flags	Default
6020:0	TSC Slave Frame Elements	Max. Subindex	UINT8	RO	0x06 (6_{dec})
6020:01	TSC__Slave Cmd	reserved	UINT8	RO	0x00 (0_{dec})
6020:02	TSC__Slave ConnID	reserved	UINT16	RO	0x0000 (0_{dec})
6020:03	TSC__Slave CRC_0	reserved	UINT16	RO	0x0000 (0_{dec})
6020:04	TSC__Slave CRC_1	reserved	UINT16	RO	0x0000 (0_{dec})
6020:05	TSC__Slave CRC_2	reserved	UINT16	RO	0x0000 (0_{dec})
6020:06	TSC__Slave CRC_3	reserved	UINT16	RO	0x0000 (0_{dec})
6020:07*)	TSC__Slave CRC_4	reserved	UINT16	RO	0x0000 (0_{dec})
6020:08*)	TSC__Slave CRC_5	reserved	UINT16	RO	0x0000 (0_{dec})
6020:09*)	TSC__Slave CRC_6	reserved	UINT16	RO	0x0000 (0_{dec})
6020:0A*)	TSC__Slave CRC_7	reserved	UINT16	RO	0x0000 (0_{dec})
6020:0B*)	TSC__Slave CRC_8	reserved	UINT16	RO	0x0000 (0_{dec})
6020:0C*)	TSC__Slave CRC_9	reserved	UINT16	RO	0x0000 (0_{dec})

*) from Revision 17

Index 7020 TSC Master Frame Elements

Index (hex)	Name	Meaning	Data type	Flags	Default
7020:0	TSC Master Frame Elements	Maximum subindex	UINT8	RO	0x03 (3_{dec})
7020:01	TSC__Master Cmd	reserved	UINT8	RO	0x00 (0_{dec})
7020:02	TSC__Master ConnID	reserved	UINT16	RO	0x0000 (0_{dec})
7020:03	TSC__Master CRC_0	reserved	UINT16	RO	0x0000 (0_{dec})

Index 8020 TSC Settings

Index (hex)	Name	Meaning	Data type	Flags	Default
8020:0	TSC Settings	Maximum subindex	UINT8	RO	0x02 (2_{dec})
8020:01	Address	TwinSAFE SC address	UINT16	RO	0x0000 (0_{dec})
8020:02	Connection Mode	Selection of TwinSAFE SC CRC	UINT32	RO	0x00000000 (0_{dec})

Index 8021 TSC Process Data Settings

Index (hex)	Name	Meaning	Data type	Flags	Default
8021:0	TSC Process Data Settings	Maximum subindex	UINT8	RO	0x12 (18_{dec})
8021:01	Auto align to MSB Ch.1	0x8021:0n: Auto align to MSB Ch.n • Default: the bit is not set. The position value is shifted to the right by the number of bits in 0x8021:1n "Shift right data Ch.n". • If the bit is set the position value is automatically shifted to the right to the max. MSB.	BOOLEAN	RO	0x00 (0_{dec})
8021:02	Auto align to MSB Ch.2		BOOLEAN	RO	0x00 (0_{dec})
8021:11	Shift right data Ch.1	0x8021:1n: Shift right data Ch.n • Default: 0 The position value is shifted by 0 bits to the right. • An individual number of bits can be specified by which the process data is to be shifted to the right.	UINT8	RO	0x00 (0_{dec})
8021:12	Shift right data Ch.2		UINT8	RO	0x00 (0_{dec})

10 Diagnostics

10.1 Diagnostics - basic principles of diag messages

DiagMessages designates a system for the transmission of messages from the EtherCAT Slave to the EtherCAT Master/TwinCAT. The messages are stored by the device in its own CoE under 0x10F3 and can be read by the application or the System Manager. An error message referenced via a code is output for each event stored in the device (warning, error, status change).

Definition

The *DiagMessages* system is defined in the ETG (EtherCAT Technology Group) in the guideline ETG.1020, chapter 13 “Diagnosis handling”. It is used so that pre-defined or flexible diagnostic messages can be conveyed from the EtherCAT Slave to the Master. In accordance with the ETG, the process can therefore be implemented supplier-independently. Support is optional. The firmware can store up to 250 DiagMessages in its own CoE.

Each DiagMessage consists of

- Diag Code (4-byte)
- Flags (2-byte; info, warning or error)
- Text ID (2-byte; reference to explanatory text from the ESI/XML)
- Timestamp (8-byte, local slave time or 64-bit Distributed Clock time, if available)
- Dynamic parameters added by the firmware

The DiagMessages are explained in text form in the ESI/XML file belonging to the EtherCAT device: on the basis of the Text ID contained in the DiagMessage, the corresponding plain text message can be found in the languages contained in the ESI/XML. In the case of Beckhoff products these are usually German and English.

Via the entry *NewMessagesAvailable* the user receives information that new messages are available.

DiagMessages can be confirmed in the device: the last/latest unconfirmed message can be confirmed by the user.

In the CoE both the control entries and the history itself can be found in the CoE object 0x10F3:

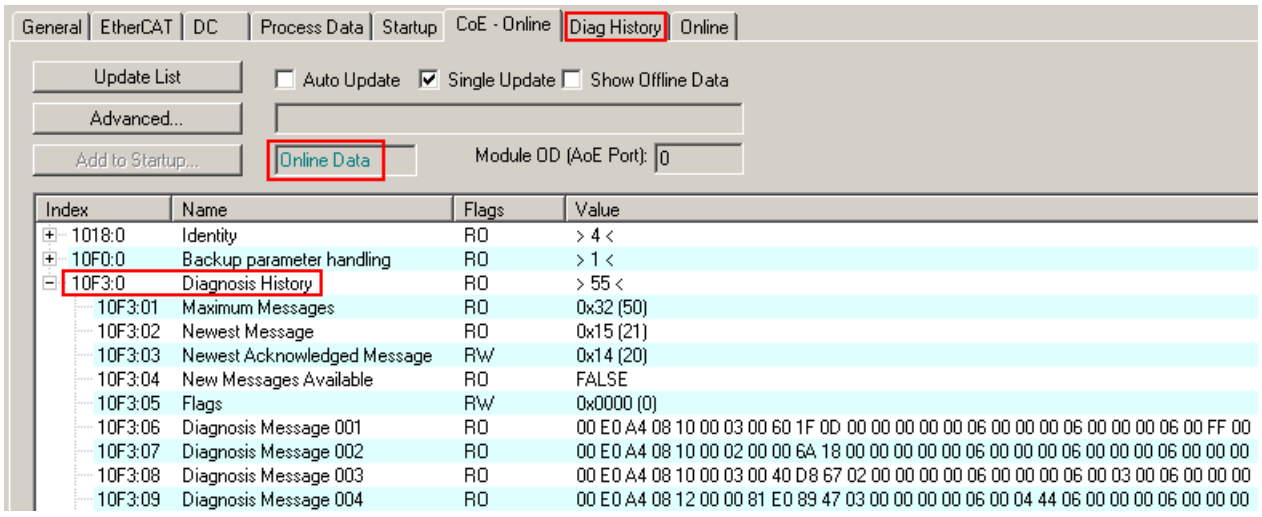


Fig. 157: DiagMessages in the CoE

The subindex of the latest *DiagMessage* can be read under 0x10F3:02.

Support for commissioning

The DiagMessages system is to be used above all during the commissioning of the plant. The diagnostic values e.g. in the StatusWord of the device (if available) are helpful for online diagnosis during the subsequent continuous operation.

TwinCAT System Manager implementation

From TwinCAT 2.11 DiagMessages, if available, are displayed in the device's own interface. Operation (collection, confirmation) also takes place via this interface.

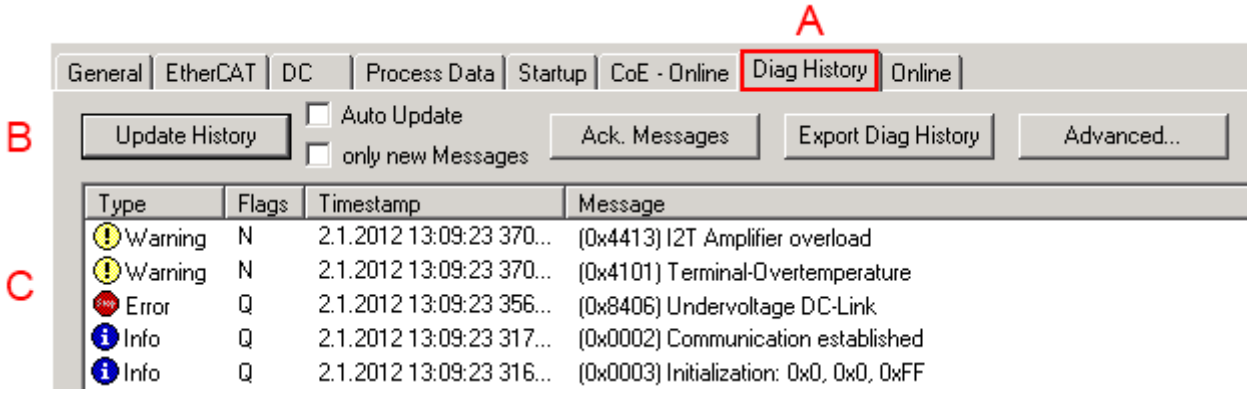


Fig. 158: Implementation of the DiagMessage system in the TwinCAT System Manager

The operating buttons (B) and the history read out (C) can be seen on the Diag History tab (A). The components of the message:

- Info/Warning/Error
- Acknowledge flag (N = unconfirmed, Q = confirmed)
- Time stamp
- Text ID
- Plain text message according to ESI/XML data

The meanings of the buttons are self-explanatory.

DiagMessages within the ADS Logger/Eventlogger

From TwinCAT 3.1 build 4022 onwards, DiagMessages sent by the terminal are shown by the TwinCAT ADS Logger. Given that DiagMessages are represented IO- comprehensive at one place, commissioning will be simplified. In addition, the logger output could be stored into a data file – hence DiagMessages are available long-term for analysis.

DiagMessages are actually only available locally in CoE 0x10F3 in the terminal and can be read out manually if required, e.g. via the DiagHistory mentioned above.

In the latest developments, the EtherCAT Terminals are set by default to report the presence of a DiagMessage as emergency via EtherCAT; the event logger can then retrieve the DiagMessage. The function is activated in the terminal via 0x10F3:05, so such terminals have the following entry in the StartUp list by default:

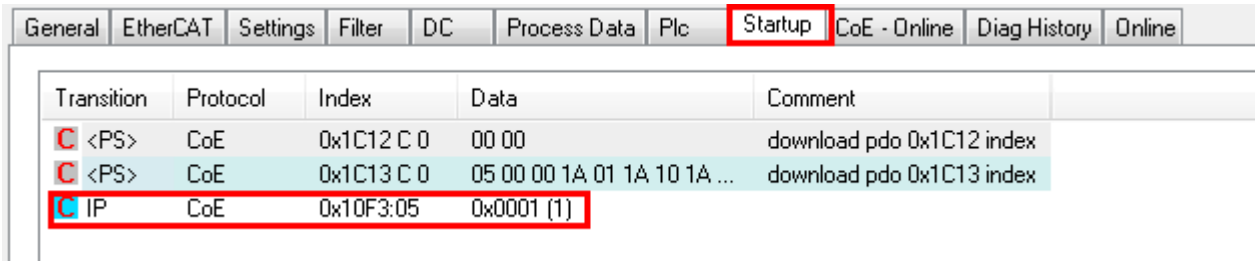


Fig. 159: Startup List

If the function is to be deactivated because, for example, many messages come in or the EventLogger is not used, the StartUp entry can be deleted or set to 0. The value can then be set back to 1 later from the PLC via CoE access if required.

Reading messages into the PLC

- In preparation -

Interpretation

Time stamp

The time stamp is obtained from the local clock of the terminal at the time of the event. The time is usually the distributed clock time (DC) from register x910.

Please note: When EtherCAT is started, the DC time in the reference clock is set to the same time as the local IPC/TwinCAT time. From this moment the DC time may differ from the IPC time, since the IPC time is not adjusted. Significant time differences may develop after several weeks of operation without a EtherCAT restart. As a remedy, external synchronization of the DC time can be used, or a manual correction calculation can be applied, as required: The current DC time can be determined via the EtherCAT master or from register x901 of the DC slave.

Structure of the Text ID

The structure of the MessageID is not subject to any standardization and can be supplier-specifically defined. In the case of Beckhoff EtherCAT devices (EL, EP) it usually reads according to **xyzz**:

x	y	zz
0: Systeminfo 2: reserved 1: Info 4: Warning 8: Error	0: System 1: General 2: Communication 3: Encoder 4: Drive 5: Inputs 6: I/O general 7: reserved	Error number

Example: Message 0x4413 --> Drive Warning Number 0x13

Overview of text IDs

Specific text IDs are listed in the device documentation.

Text ID	Type	Place	Text Message	Additional comment
0x0001	Information	System	No error	No error
0x0002	Information	System	Communication established	Connection established
0x0003	Information	System	Initialization: 0x%X, 0x%X, 0x%X	General information; parameters depend on event. See device documentation for interpretation.
0x1000	Information	System	Information: 0x%X, 0x%X, 0x%X	General information; parameters depend on event. See device documentation for interpretation.
0x1012	Information	System	EtherCAT state change Init - PreOp	
0x1021	Information	System	EtherCAT state change PreOp - Init	
0x1024	Information	System	EtherCAT state change PreOp - Safe-Op	
0x1042	Information	System	EtherCAT state change SafeOp - PreOp	
0x1048	Information	System	EtherCAT state change SafeOp - Op	
0x1084	Information	System	EtherCAT state change Op - SafeOp	
0x1100	Information	General	Detection of operation mode completed: 0x%X, %d	Detection of the mode of operation ended
0x1135	Information	General	Cycle time o.k.: %d	Cycle time OK
0x1157	Information	General	Data manually saved (Idx: 0x%X, SubIdx: 0x%X)	Data saved manually
0x1158	Information	General	Data automatically saved (Idx: 0x%X, SubIdx: 0x%X)	Data saved automatically
0x1159	Information	General	Data deleted (Idx: 0x%X, SubIdx: 0x%X)	Data deleted
0x117F	Information	General	Information: 0x%X, 0x%X, 0x%X	Information
0x1201	Information	Communication	Communication re-established	Communication to the field side restored This message appears, for example, if the voltage was removed from the power contacts and re-applied during operation.
0x1300	Information	Encoder	Position set: %d, %d	Position set - StartInputhandler
0x1303	Information	Encoder	Encoder Supply ok	Encoder power supply unit OK
0x1304	Information	Encoder	Encoder initialization successfully, channel: %X	Encoder initialization successfully completed
0x1305	Information	Encoder	Sent command encoder reset, channel: %X	Send encoder reset command
0x1400	Information	Drive	Drive is calibrated: %d, %d	Drive is calibrated
0x1401	Information	Drive	Actual drive state: 0x%X, %d	Current drive status
0x1705	Information		CPU usage returns in normal range (< 85%%)	Processor load is back in the normal range
0x1706	Information		Channel is not in saturation anymore	Channel is no longer in saturation
0x1707	Information		Channel is not in overload anymore	Channel is no longer overloaded
0x170A	Information		No channel range error anymore	A measuring range error is no longer active
0x170C	Information		Calibration data saved	Calibration data were saved
0x170D	Information		Calibration data will be applied and saved after sending the command “0x5AFE”	Calibration data are not applied and saved until the command "0x5AFE" is sent.

Text ID	Type	Place	Text Message	Additional comment
0x2000	Information	System	%s: %s	
0x2001	Information	System	%s: Network link lost	Network connection lost
0x2002	Information	System	%s: Network link detected	Network connection found
0x2003	Information	System	%s: no valid IP Configuration - Dhcp client started	Invalid IP configuration
0x2004	Information	System	%s: valid IP Configuration (IP: %d.%d.%d.%d) assigned by Dhcp server %d.%d.%d.%d	Valid IP configuration, assigned by the DHCP server
0x2005	Information	System	%s: Dhcp client timed out	DHCP client timeout
0x2006	Information	System	%s: Duplicate IP Address detected (%d.%d.%d.%d)	Duplicate IP address found
0x2007	Information	System	%s: UDP handler initialized	UDP handler initialized
0x2008	Information	System	%s: TCP handler initialized	TCP handler initialized
0x2009	Information	System	%s: No more free TCP sockets available	No free TCP sockets available.

Text ID	Type	Place	Text Message	Additional comment
0x4000	Warning		Warning: 0x%X, 0x%X, 0x%X	General warning; parameters depend on event. See device documentation for interpretation.
0x4001	Warning	System	Warning: 0x%X, 0x%X, 0x%X	
0x4002	Warning	System	%s: %s Connection Open (IN:%d OUT:%d API:%dms) from %d.%d.%d.%d successful	
0x4003	Warning	System	%s: %s Connection Close (IN:%d OUT:%d) from %d.%d.%d.%d successful	
0x4004	Warning	System	%s: %s Connection (IN:%d OUT:%d) with %d.%d.%d.%d timed out	
0x4005	Warning	System	%s: %s Connection Open (IN:%d OUT:%d) from %d.%d.%d.%d denied (Error: %u)	
0x4006	Warning	System	%s: %s Connection Open (IN:%d OUT:%d) from %d.%d.%d.%d denied (Input Data Size expected: %d Byte(s) received: %d Byte(s))	
0x4007	Warning	System	%s: %s Connection Open (IN:%d OUT:%d) from %d.%d.%d.%d denied (Output Data Size expected: %d Byte(s) received: %d Byte(s))	
0x4008	Warning	System	%s: %s Connection Open (IN:%d OUT:%d) from %d.%d.%d.%d denied (RPI:%dms not supported -> API:%dms)	
0x4101	Warning	General	Terminal-Overtemperature	Overtemperature. The internal temperature of the terminal exceeds the parameterized warning threshold.
0x4102	Warning	General	Discrepancy in the PDO-Configuration	The selected PDOs do not match the set operating mode. Sample: Drive operates in velocity mode, but the velocity PDO is but not mapped in the PDOs.
0x417F	Warning	General	Warning: 0x%X, 0x%X, 0x%X	
0x428D	Warning	General	Challenge is not Random	
0x4300	Warning	Encoder	Subincrements deactivated: %d, %d	Sub-increments deactivated (despite activated configuration)
0x4301	Warning	Encoder	Encoder-Warning	General encoder error
0x4302	Warning	Encoder	Maximum frequency of the input signal is nearly reached (channel %d)	
0x4303	Warning	Encoder	Limit counter value was reduced because of the PDO configuration (channel %d)	
0x4304	Warning	Encoder	Reset counter value was reduced because of the PDO configuration (channel %d)	
0x4400	Warning	Drive	Drive is not calibrated: %d, %d	Drive is not calibrated
0x4401	Warning	Drive	Starttype not supported: 0x%X, %d	Start type is not supported
0x4402	Warning	Drive	Command rejected: %d, %d	Command rejected
0x4405	Warning	Drive	Invalid modulo subtype: %d, %d	Modulo sub-type invalid
0x4410	Warning	Drive	Target overrun: %d, %d	Target position exceeded
0x4411	Warning	Drive	DC-Link undervoltage (Warning)	The DC link voltage of the terminal is lower than the parameterized minimum voltage. Activation of the output stage is prevented.
0x4412	Warning	Drive	DC-Link overvoltage (Warning)	The DC link voltage of the terminal is higher than the parameterized maximum voltage. Activation of the output stage is prevented.
0x4413	Warning	Drive	I2T-Model Amplifier overload (Warning)	• The amplifier is being operated outside the specification. • The I2T-model of the amplifier is incorrectly parameterized.
0x4414	Warning	Drive	I2T-Model Motor overload (Warning)	• The motor is being operated outside the parameterized rated values.

Text ID	Type	Place	Text Message	Additional comment
				• The I2T-model of the motor is incorrectly parameterized.
0x4415	Warning	Drive	Speed limitation active	The maximum speed is limited by the parameterized objects (e.g. velocity limitation, motor speed limitation). This warning is output if the set velocity is higher than one of the parameterized limits.
0x4416	Warning	Drive	Step lost detected at position: 0x%X%X	Step loss detected
0x4417	Warning	Drive	Motor overtemperature	The internal temperature of the motor exceeds the parameterized warning threshold
0x4418	Warning	Drive	Limit: Current	Limit: current is limited
0x4419	Warning	Drive	Limit: Amplifier I2T-model exceeds 100%%	The threshold values for the maximum current were exceeded.
0x441A	Warning	Drive	Limit: Motor I2T-model exceeds 100%%	Limit: Motor I2T-model exceeds 100%
0x441B	Warning	Drive	Limit: Velocity limitation	The threshold values for the maximum speed were exceeded.
0x441C	Warning	Drive	STO while the axis was enabled	An attempt was made to activate the axis, despite the fact that no voltage is present at the STO input.
0x4600	Warning	General IO	Wrong supply voltage range	Supply voltage not in the correct range
0x4610	Warning	General IO	Wrong output voltage range	Output voltage not in the correct range
0x4705	Warning		Processor usage at %d %%	Processor load at %d %%
0x470A	Warning		EtherCAT Frame missed (change Settings or DC Operation Mode or Sync0 Shift Time)	EtherCAT frame missed (change DC Operation Mode or Sync0 Shift Time under Settings)

Text ID	Type	Place	Text Message	Additional comment
0x8000	Error	System	%s: %s	
0x8001	Error	System	Error: 0x%X, 0x%X, 0x%X	General error; parameters depend on event. See device documentation for interpretation.
0x8002	Error	System	Communication aborted	Communication aborted
0x8003	Error	System	Configuration error: 0x%X, 0x%X, 0x%X	General; parameters depend on event. See device documentation for interpretation.
0x8004	Error	System	%s: Unsuccessful FwdOpen-Response received from %d.%d.%d.%d (%s) (Error: %u)	
0x8005	Error	System	%s: FwdClose-Request sent to %d.%d.%d.%d (%s)	
0x8006	Error	System	%s: Unsuccessful FwdClose-Response received from %d.%d.%d.%d (%s) (Error: %u)	
0x8007	Error	System	%s: Connection with %d.%d.%d.%d (%s) closed	
0x8100	Error	General	Status word set: 0x%X, %d	Error bit set in the status word
0x8101	Error	General	Operation mode incompatible to PDO interface: 0x%X, %d	Mode of operation incompatible with the PDO interface
0x8102	Error	General	Invalid combination of Inputs and Outputs PDOs	Invalid combination of input and output PDOs
0x8103	Error	General	No variable linkage	No variables linked
0x8104	Error	General	Terminal-Overtemperature	The internal temperature of the terminal exceeds the parameterized error threshold. Activation of the terminal is prevented
0x8105	Error	General	PD-Watchdog	Communication between the fieldbus and the output stage is secured by a Watchdog. The axis is stopped automatically if the fieldbus communication is interrupted. • The EtherCAT connection was interrupted during operation. • The Master was switched to Config mode during operation.
0x8135	Error	General	Cycle time has to be a multiple of 125 µs	The IO or NC cycle time divided by 125 µs does not produce a whole number.
0x8136	Error	General	Configuration error: invalid sampling rate	Configuration error: Invalid sampling rate
0x8137	Error	General	Electronic type plate: CRC error	Content of the external name plate memory invalid.
0x8140	Error	General	Sync Error	Real-time violation
0x8141	Error	General	Sync%X Interrupt lost	Sync%X Interrupt lost
0x8142	Error	General	Sync Interrupt asynchronous	Sync Interrupt asynchronous
0x8143	Error	General	Jitter too big	Jitter limit violation
0x817F	Error	General	Error: 0x%X, 0x%X, 0x%X	
0x8200	Error	Communication	Write access error: %d, %d	Error while writing
0x8201	Error	Communication	No communication to field-side (Auxiliary voltage missing)	• There is no voltage applied to the power contacts. • A firmware update has failed.
0x8281	Error	Communication	Ownership failed: %X	
0x8282	Error	Communication	To many Keys founded	
0x8283	Error	Communication	Key Creation failed: %X	
0x8284	Error	Communication	Key loading failed	
0x8285	Error	Communication	Reading Public Key failed: %X	
0x8286	Error	Communication	Reading Public EK failed: %X	
0x8287	Error	Communication	Reading PCR Value failed: %X	
0x8288	Error	Communication	Reading Certificate EK failed: %X	
0x8289	Error	Communication	Challenge could not be hashed: %X	
0x828A	Error	Communication	Tickstamp Process failed	
0x828B	Error	Communication	PCR Process failed: %X	
0x828C	Error	Communication	Quote Process failed: %X	
0x82FF	Error	Communication	Bootmode not activated	Boot mode not activated
0x8300	Error	Encoder	Set position error: 0x%X, %d	Error while setting the position

Text ID	Type	Place	Text Message	Additional comment
0x8301	Error	Encoder	Encoder increments not configured: 0x%X, %d	Encoder increments not configured
0x8302	Error	Encoder	Encoder error	The amplitude of the resolver is too small
0x8303	Error	Encoder	Encoder power missing (channel %d)	
0x8304	Error	Encoder	Encoder communication error, channel: %X	Encoder communication error
0x8305	Error	Encoder	EnDat2.2 is not supported, channel: %X	EnDat2.2 is not supported
0x8306	Error	Encoder	Delay time, tolerance limit exceeded, 0x%X, channel: %X	Runtime measurement, tolerance exceeded
0x8307	Error	Encoder	Delay time, maximum value exceeded, 0x%X, channel: %X	Runtime measurement, maximum value exceeded
0x8308	Error	Encoder	Unsupported ordering designation, 0x%X, channel: %X (only 02 and 22 is supported)	Wrong EnDat order ID
0x8309	Error	Encoder	Encoder CRC error, channel: %X	Encoder CRC error
0x830A	Error	Encoder	Temperature %X could not be read, channel: %X	Temperature cannot be read
0x830C	Error	Encoder	Encoder Single-Cycle-Data Error, channel. %X	CRC error detected. Check the transmission path and the CRC polynomial
0x830D	Error	Encoder	Encoder Watchdog Error, channel. %X	The sensor has not responded within a predefined time period
0x8310	Error	Encoder	Initialisation error	
0x8311	Error	Encoder	Maximum frequency of the input signal is exceeded (channel %d)	
0x8312	Error	Encoder	Encoder plausibility error (channel %d)	
0x8313	Error	Encoder	Configuration error (channel %d)	
0x8314	Error	Encoder	Synchronisation error	
0x8315	Error	Encoder	Error status input (channel %d)	
0x8400	Error	Drive	Incorrect drive configuration: 0x%X, %d	Drive incorrectly configured
0x8401	Error	Drive	Limiting of calibration velocity: %d, %d	Limitation of the calibration velocity
0x8402	Error	Drive	Emergency stop activated: 0x%X, %d	Emergency stop activated
0x8403	Error	Drive	ADC Error	Error during current measurement in the ADC
0x8404	Error	Drive	Overcurrent	Overcurrent in phase U, V or W
0x8405	Error	Drive	Invalid modulo position: %d	Modulo position invalid
0x8406	Error	Drive	DC-Link undervoltage (Error)	The DC link voltage of the terminal is lower than the parameterized minimum voltage. Activation of the output stage is prevented.
0x8407	Error	Drive	DC-Link overvoltage (Error)	The DC link voltage of the terminal is higher than the parameterized maximum voltage. Activation of the output stage is prevented.
0x8408	Error	Drive	I2T-Model Amplifier overload (Error)	• The amplifier is being operated outside the specification. • The I2T-model of the amplifier is incorrectly parameterized.
0x8409	Error	Drive	I2T-Model motor overload (Error)	• The motor is being operated outside the parameterized rated values. • The I2T-model of the motor is incorrectly parameterized.
0x840A	Error	Drive	Overall current threshold exceeded	Total current exceeded
0x8415	Error	Drive	Invalid modulo factor: %d	Modulo factor invalid
0x8416	Error	Drive	Motor overtemperature	The internal temperature of the motor exceeds the parameterized error threshold. The motor stops immediately. Activation of the output stage is prevented.
0x8417	Error	Drive	Maximum rotating field velocity exceeded	Rotary field speed exceeds the value specified for dual use (EU 1382/2014).
0x841C	Error	Drive	STO while the axis was enabled	An attempt was made to activate the axis, despite the fact that no voltage is present at the STO input.

Text ID	Type	Place	Text Message	Additional comment
0x8550	Error	Inputs	Zero crossing phase %X missing	Zero crossing phase %X missing
0x8551	Error	Inputs	Phase sequence Error	Wrong direction of rotation
0x8552	Error	Inputs	Overcurrent phase %X	Overcurrent phase %X
0x8553	Error	Inputs	Overcurrent neutral wire	Overcurrent neutral wire
0x8581	Error	Inputs	Wire broken Ch %D	Wire broken Ch %d
0x8600	Error	General IO	Wrong supply voltage range	Supply voltage not in the correct range
0x8601	Error	General IO	Supply voltage to low	Supply voltage too low
0x8602	Error	General IO	Supply voltage to high	Supply voltage too high
0x8603	Error	General IO	Over current of supply voltage	Overcurrent of supply voltage
0x8610	Error	General IO	Wrong output voltage range	Output voltage not in the correct range
0x8611	Error	General IO	Output voltage to low	Output voltage too low
0x8612	Error	General IO	Output voltage to high	Output voltage too high
0x8613	Error	General IO	Over current of output voltage	Overcurrent of output voltage
0x8700	Error		Channel/Interface not calibrated	Channel/interface not synchronized
0x8701	Error		Operating time was manipulated	Operating time was manipulated
0x8702	Error		Oversampling setting is not possible	Oversampling setting not possible
0x8703	Error		No slave controller found	No slave controller found
0x8704	Error		Slave controller is not in Bootstrap	Slave controller is not in bootstrap
0x8705	Error		Processor usage to high (>= 100%%)	Processor load too high (>= 100%%)
0x8706	Error		Channel in saturation	Channel in saturation
0x8707	Error		Channel overload	Channel overload
0x8708	Error		Overloadtime was manipulated	Overload time was manipulated
0x8709	Error		Saturationtime was manipulated	Saturation time was manipulated
0x870A	Error		Channel range error	Measuring range error for the channel
0x870B	Error		no ADC clock	No ADC clock available
0xFFFF	Information		Debug: 0x%X, 0x%X, 0x%X	Debug: 0x%X, 0x%X, 0x%X

11 Appendix

11.1 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 [▶ 175].
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!

EL5032			
Hardware (HW)	**Firmware**	**Revision no.**	**Release date**
01 - 10*	06	EL5032-0000-0021	2012/11
	07		2013/01
		EL5032-0000/0023	2012/08
	08		2013/01
	09		2013/02
	11		2014/04
	12		2014/07
	13	EL5032-0000/0024	2015/06
	14	EL5032-0000/0025	2015/06
		EL5032-0000/0026	2016/04
	15	EL5032-0000/0027	2018/07
	16*	EL5032-0000/0028	2023/08

EL5032-0090			
Hardware (HW)	**Firmware**	**Revision no.**	**Release date**
07 - 10*	15	EL5032-0090/0016	2018/07
	16*	EL5032-0090/0017	2023/08

*) 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.

11.2 Firmware Update EL/ES/EM/ELM/EP/EPP/ERPxxxx

This section describes the device update for Beckhoff EtherCAT slaves from the 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.

11.2.1 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:

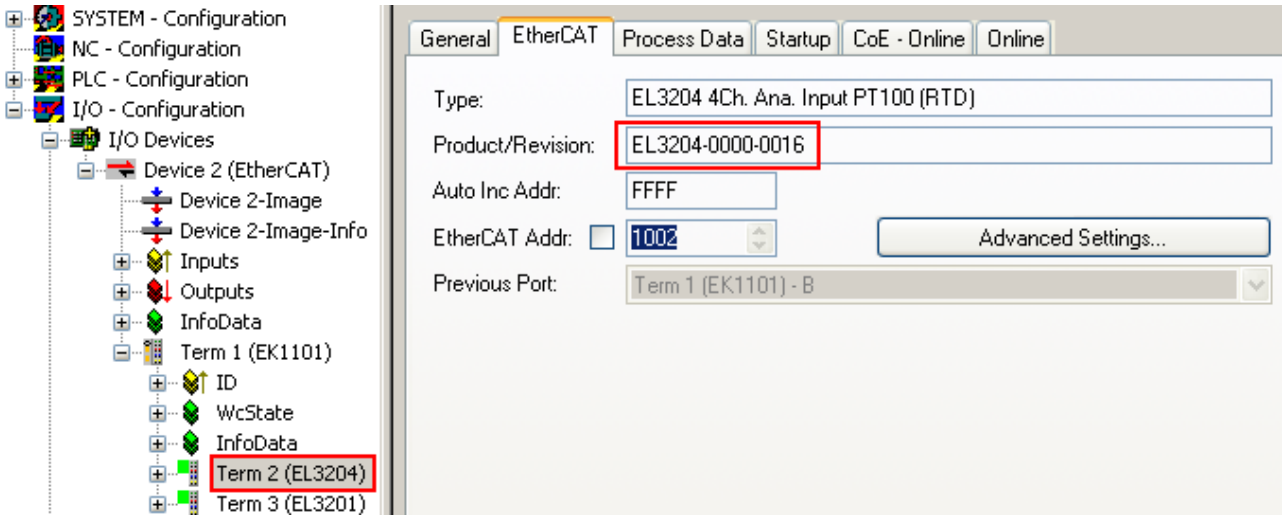


Fig. 160: Device identifier consisting of name EL3204-0000 and revision -0016

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:

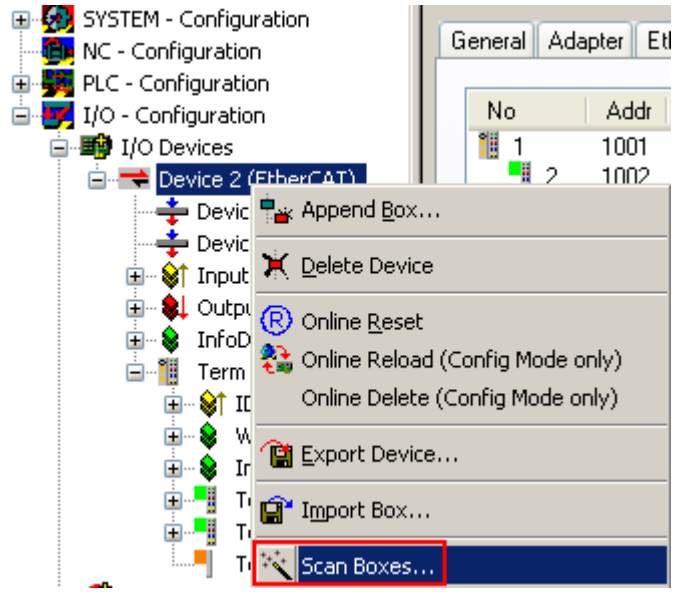


Fig. 161: Scan the subordinate field by right-clicking on the EtherCAT device

If the found field matches the configured field, the display shows



Fig. 162: Configuration is identical

otherwise a change dialog appears for entering the actual data in the configuration.

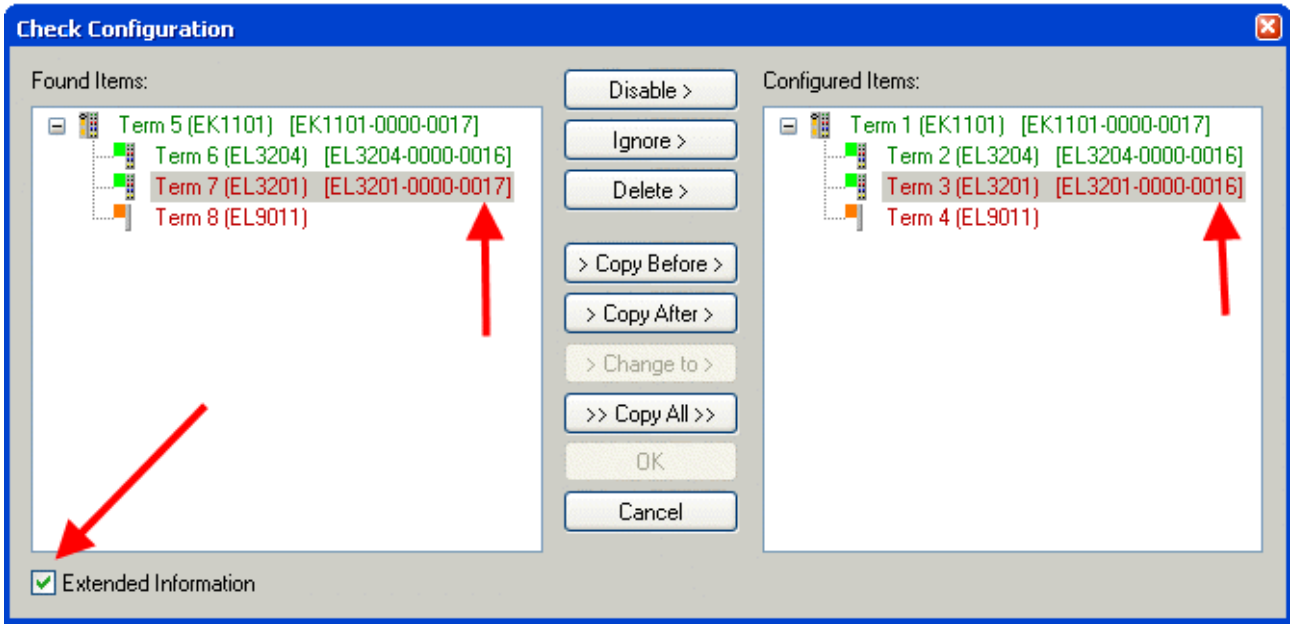


Fig. 163: Change dialog

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*

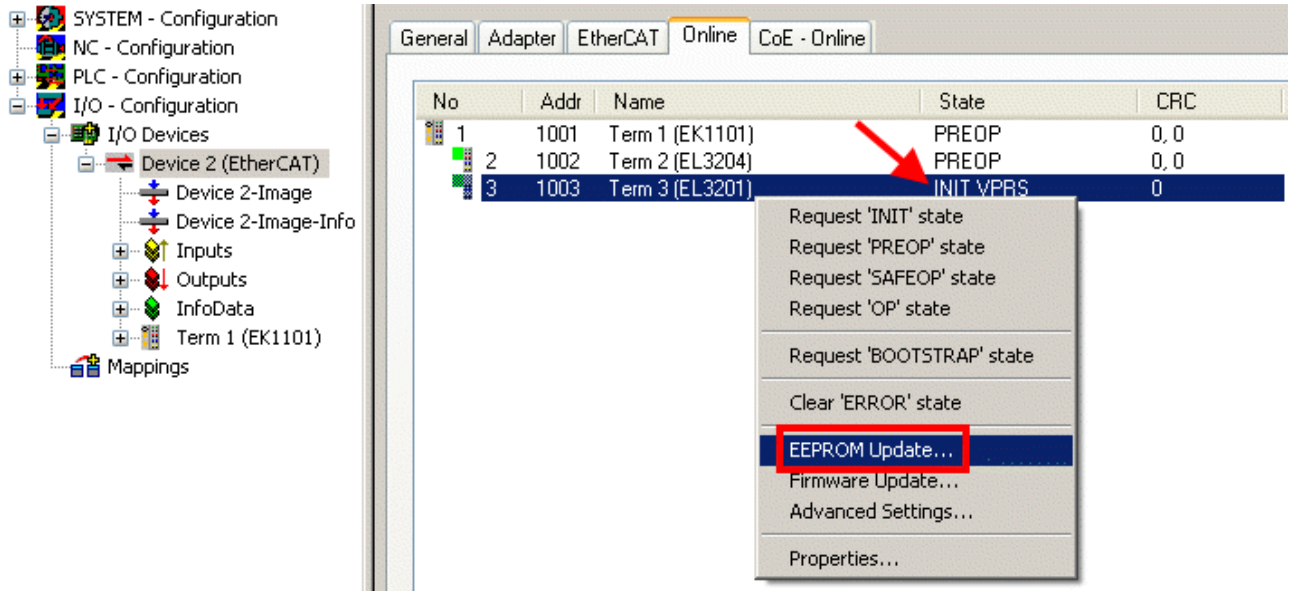


Fig. 164: EEPROM Update

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.

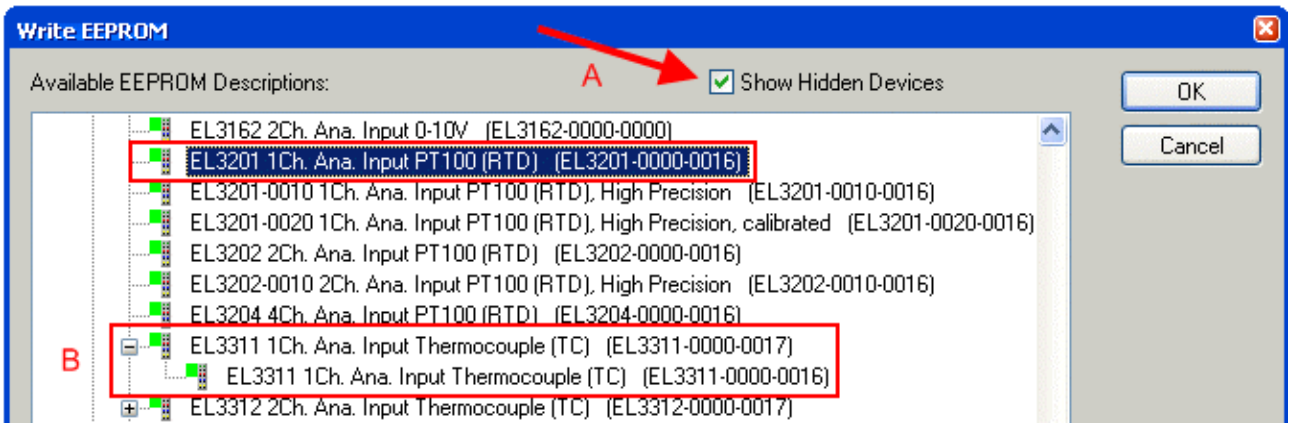


Fig. 165: Selecting the new ESI

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.

11.2.2 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.

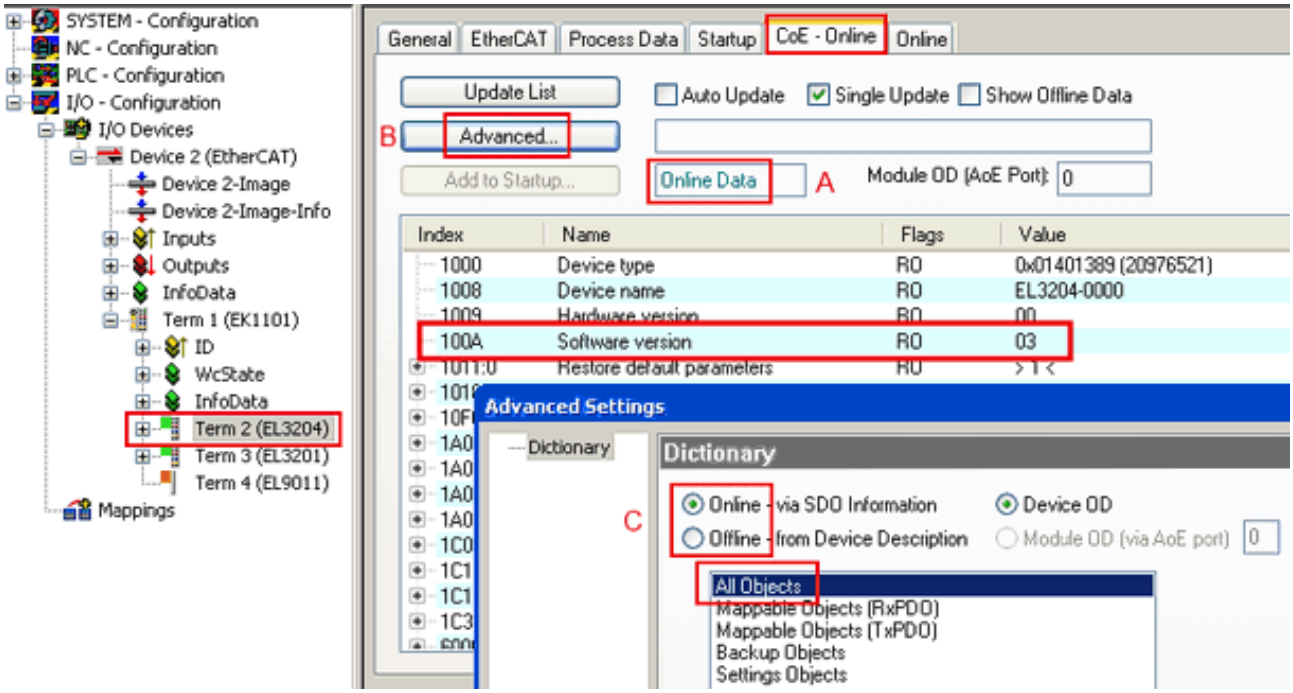


Fig. 166: Display of EL3204 firmware version

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*.

11.2.3 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.*

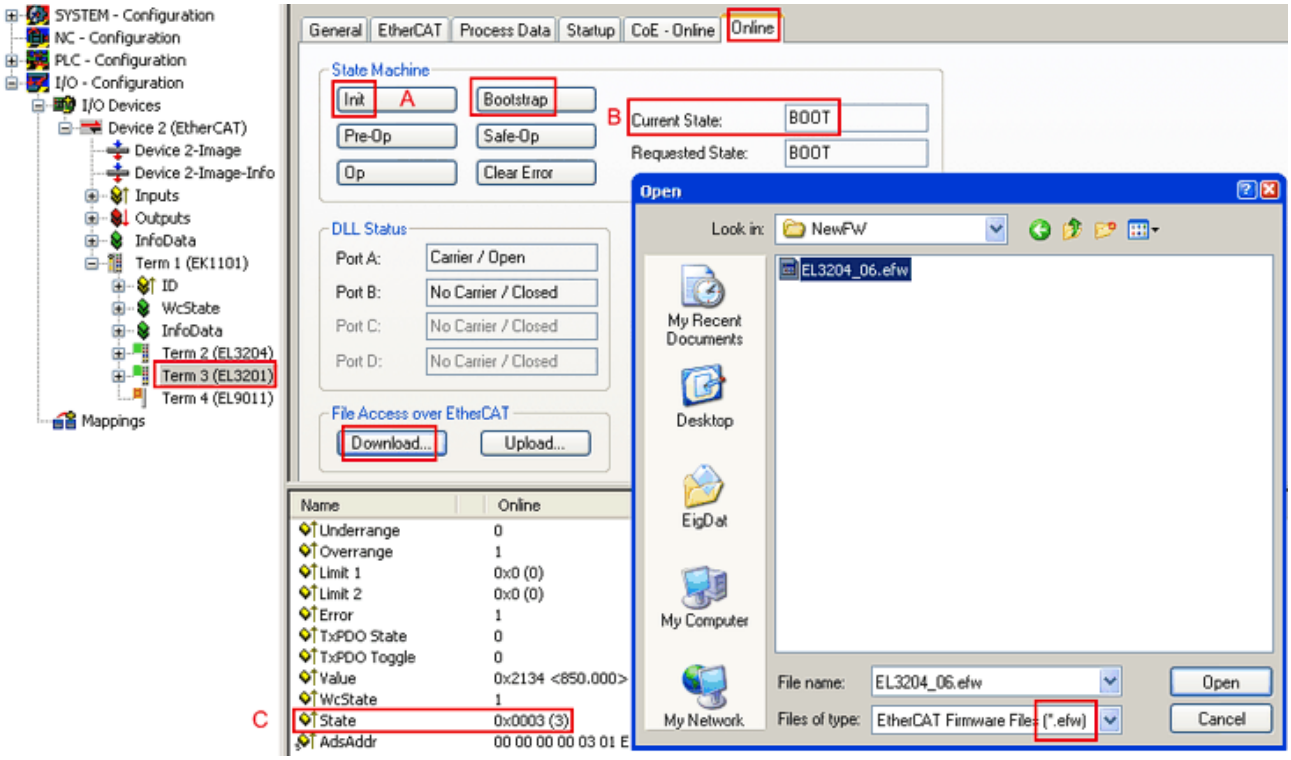
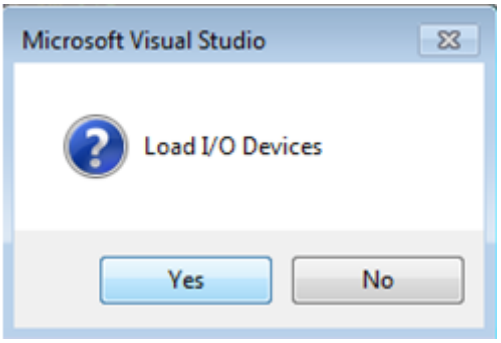
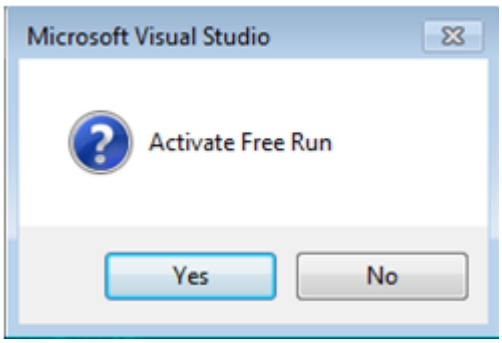


Fig. 167: Firmware Update

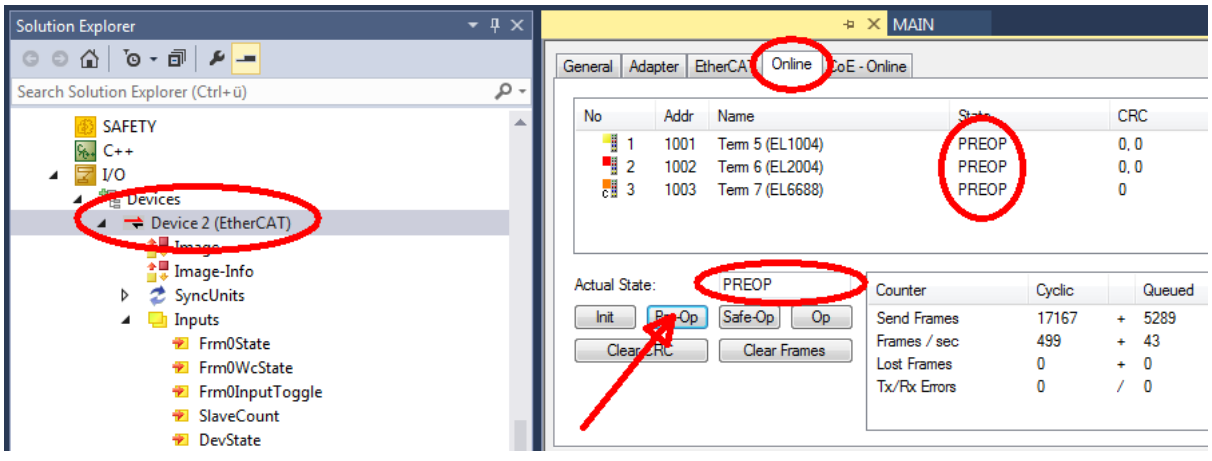
Proceed as follows, unless instructed otherwise by Beckhoff support. Valid for TwinCAT 2 and 3 as EtherCAT master.

- 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.



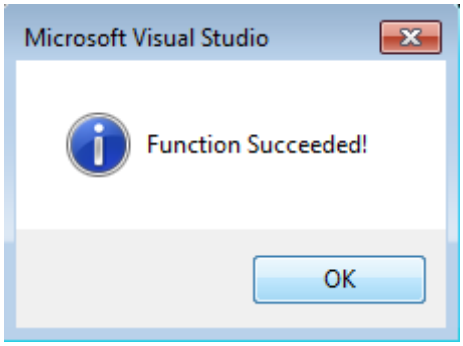


- Switch EtherCAT Master to PreOP



- 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.

11.2.4 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.

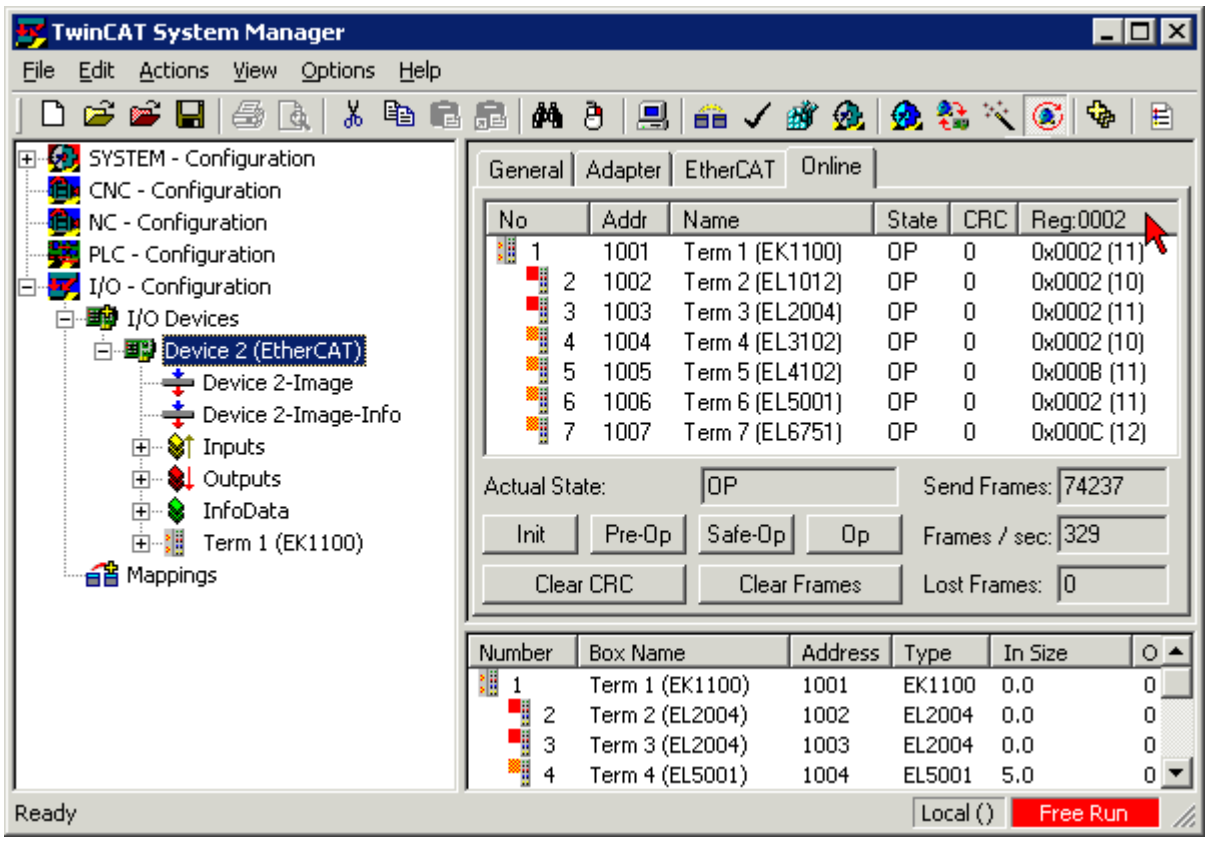


Fig. 168: FPGA firmware version definition

If the column *Reg:0002* is not displayed, right-click the table header and select *Properties* in the context menu.

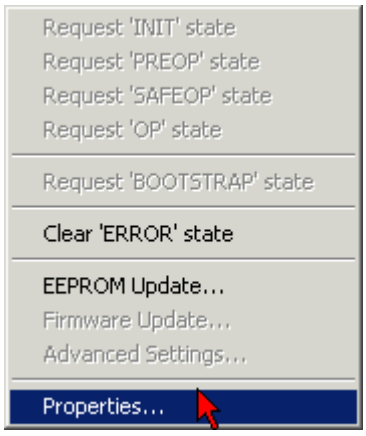


Fig. 169: Context menu *Properties*

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.

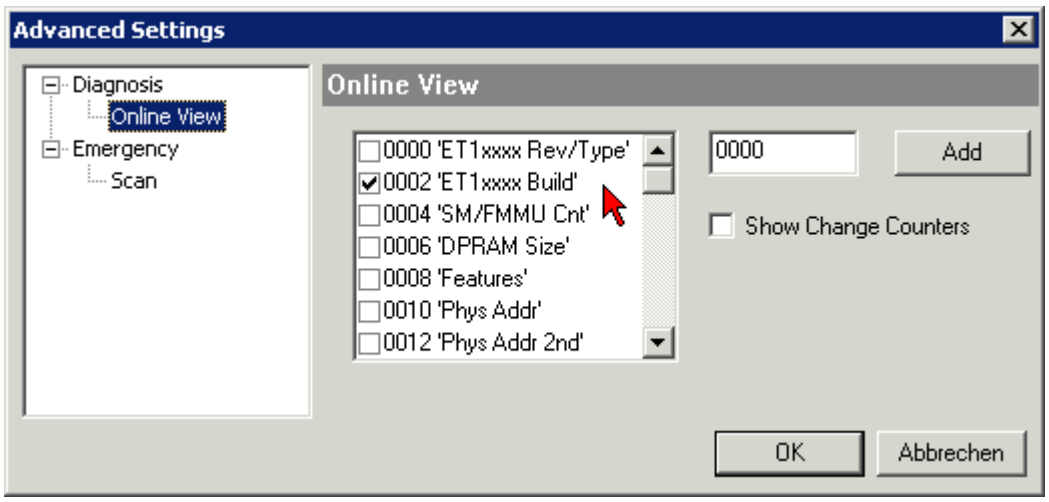


Fig. 170: Dialog *Advanced Settings*

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.

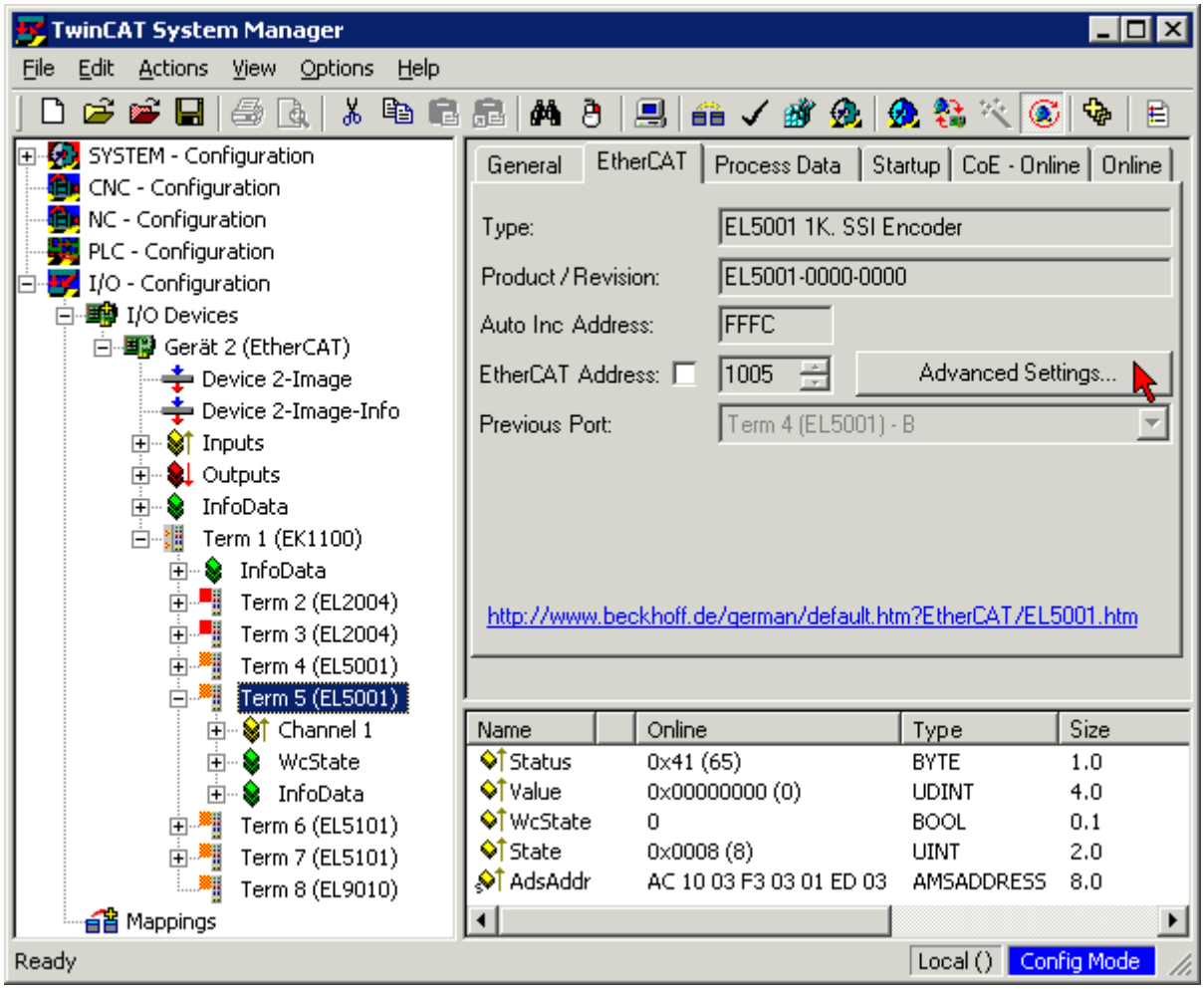
Older firmware versions can only be updated by the manufacturer!

Updating an EtherCAT device

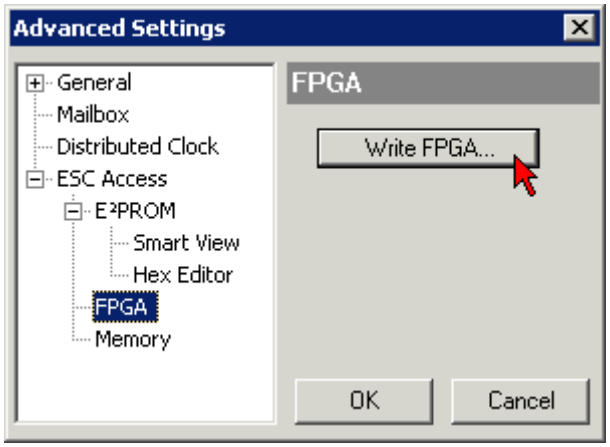
The following sequence order have to be met if no other specifications are given (e.g. by the Beckhoff support):

- 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.

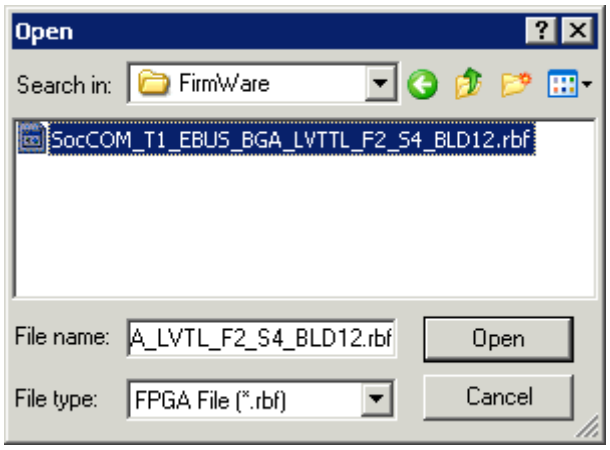
- In the TwinCAT System Manager select the terminal for which the FPGA firmware is to be updated (in the example: Terminal 5: EL5001) and
  click the *Advanced Settings* button in the *EtherCAT* tab:



- 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!

11.2.5 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.

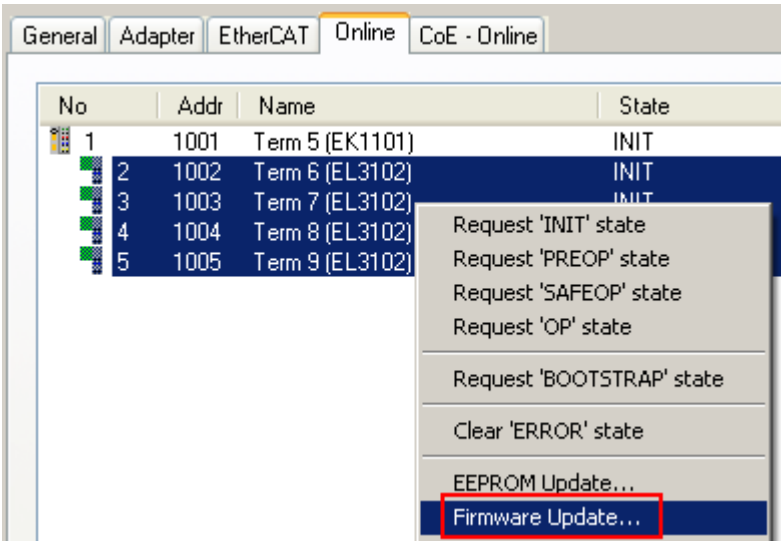


Fig. 171: Multiple selection and firmware update

Select the required slaves and carry out the firmware update in BOOTSTRAP mode as described above.

11.3 Restoring the delivery state

To restore the delivery state (factory settings) of CoE objects 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*).

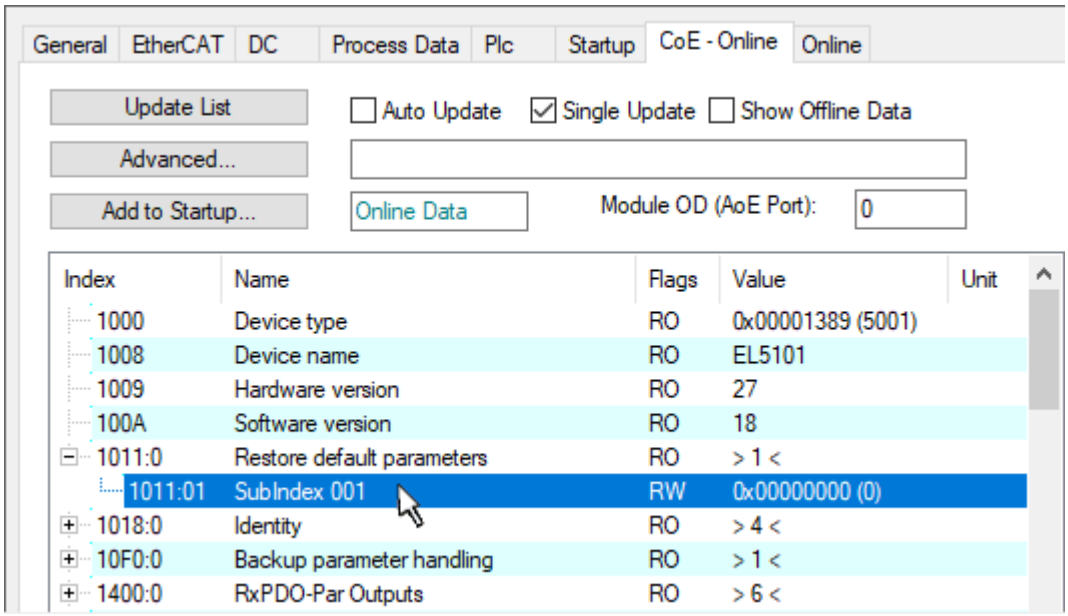


Fig. 172: Selecting the *Restore default parameters* PDO

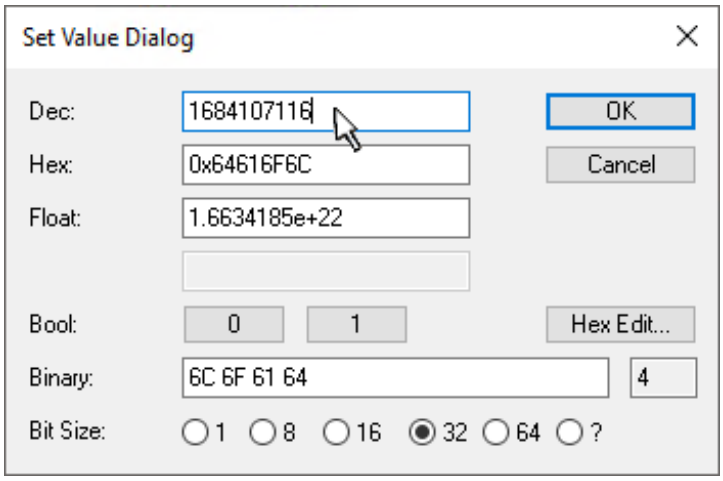


Fig. 173: Entering a restore value in the Set Value dialog

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.
- 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.

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

Third-party trademark statements

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.

EnDat is a trademark of Dr. Johannes Heidenhain GmbH.

Microsoft, Microsoft Azure, Microsoft Edge, PowerShell, Visual Studio, Windows and Xbox are trademarks of the Microsoft group of companies.

More Information:
www.beckhoff.com/EL5032/

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com