

Documentation | EN

EL6224-00x0

IO-Link Terminal



Table of contents

1 Foreword	7
1.1 IO-Link Terminals Product Overview	7
1.2 Notes on the documentation	8
1.3 Safety instructions	9
1.4 Guide through documentation	10
1.5 Documentation issue status	11
1.6 Version identification of EtherCAT devices	12
1.6.1 General notes on marking	12
1.6.2 Version identification of EL terminals	13
1.6.3 Beckhoff Identification Code (BIC)	14
1.6.4 Electronic access to the BIC (eBIC)	16
2 EL6224, EL6224-0090 - Product description	18
2.1 EL6224 - Introduction	18
2.2 EL6224-0090 - Introduction	19
2.3 EL6224, EL6224-0090 - Technical data	20
2.4 Start	21
3 IO-Link basics	22
3.1 IO-Link system layout	22
3.2 Establishment of IO Link communication	23
3.3 Device description IODD	24
3.4 Parameter server	24
3.5 Data transfer rate	25
4 Basics communication	26
4.1 EtherCAT basics	26
4.2 EtherCAT cabling – wire-bound	26
4.3 General notes for setting the watchdog	28
4.4 EtherCAT State Machine	29
4.5 CoE Interface	31
4.6 Distributed Clock	36
5 Mounting and wiring	37
5.1 Instructions for ESD protection	37
5.2 Explosion protection	38
5.2.1 ATEX - Special conditions (extended temperature range)	38
5.2.2 IECEx - Special conditions	39
5.2.3 Continuative documentation for ATEX and IECEx	39
5.3 UL notice	40
5.4 Notes on TwinSAFE SC	41
5.4.1 Safety instructions	41
5.4.2 Environmental conditions	41
5.4.3 Transport / storage	41
5.4.4 Control cabinet / terminal box	41
5.5 Installation on mounting rails	42
5.6 Installation instructions for enhanced mechanical load capacity	45

5.7	Connection	46
5.7.1	Connection system.....	46
5.7.2	Wiring	48
5.7.3	Shielding	49
5.8	Note - power supply	50
5.9	Positioning of passive Terminals.....	51
5.10	Installation positions	52
5.11	EL6224, EL6224-0090 - LEDs and connection	54
5.12	Disposal	56
6	IO link - Configuration and parameterizing.....	57
6.1	Configuration of the IO link master.....	57
6.2	Configuration of the IO-Link devices	58
6.2.1	Open the IO link configuration tool.....	58
6.2.2	Integrating IO-Link devices	59
6.2.3	Removal of IO-Link devices	68
6.2.4	Activating the configuration	69
6.3	Settings of the IO-Link devices	70
6.4	EPIxxxx, ERIxxxx - Setting of the IO-Link device parameters	72
6.5	Object description and parameterization.....	83
6.5.1	Objects for commissioning	83
6.5.2	Standard objects (0x1000-0x1FFF)	85
6.5.3	Profile-specific objects (0x6000-0xFFFF)	89
6.5.4	Objects TwinSAFE Single Channel (EL6224-0090).....	93
7	Access to IO-Link data	96
7.1	IO-Link system communication	96
7.2	PDO Assignment.....	97
7.3	Accessing IO-Link parameters	98
7.4	Parameter data exchange	99
7.5	ADS	100
7.6	Access to events	101
7.7	PLC library: Tc3_IoLink.....	101
8	TwinSAFE SC.....	102
8.1	TwinSAFE SC	102
8.1.1	TwinSAFE SC - operating principle.....	102
8.1.2	TwinSAFE SC - configuration	102
8.2	TwinSAFE SC process data EL6224-0090	106
9	Configuration with TwinCAT	108
9.1	TwinCAT Quick Start.....	108
9.1.1	TwinCAT 2	111
9.1.2	TwinCAT 3	121
9.2	TwinCAT Development Environment	134
9.2.1	Installation of the TwinCAT real-time driver	135
9.2.2	Notes regarding ESI device description	141
9.2.3	TwinCAT ESI Updater	145
9.2.4	Distinction between Online and Offline	145

9.2.5	OFFLINE configuration creation.....	146
9.2.6	ONLINE configuration creation	151
9.2.7	EtherCAT subscriber configuration	159
9.2.8	Import/Export of EtherCAT devices with SCI and XTI.....	169
9.3	General Commissioning Instructions for an EtherCAT Slave	175
10	Error handling and diagnostics	184
10.1	ADS error codes and further error diagnostics.....	184
11	Appendix	187
11.1	EtherCAT AL Status Codes	187
11.2	Firmware Update EL/ES/EM/ELM/EP/EPP/ERPxxxx	187
11.2.1	Device description ESI file/XML	188
11.2.2	Firmware explanation.....	191
11.2.3	Updating controller firmware *.efw	192
11.2.4	FPGA firmware *.rbf	194
11.2.5	Simultaneous updating of several EtherCAT devices	198
11.3	Firmware compatibility	199
11.4	Restoring the delivery state.....	200
11.5	Support and Service.....	201

1 Foreword

1.1 IO-Link Terminals Product Overview

[EL6224](#) [► 18]

IO-Link Terminal

[EL6224-0090](#) [► 19]

IO-Link Terminal, 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
Application guide TwinSAFE (PDF)	Examples for the calculation of safety parameters for safety functions
Explosion Protection for Terminal Systems (PDF)	Notes on the use of the Beckhoff terminal systems in hazardous areas according to ATEX and IECEx
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
3.5.0	• Update chapter "Technical data" • Update chapter "Explosion protection" • Update revision status • Update structure
3.4	• Chapter "PLC library: Tc3_IoLink" added • Update revision status • Update structure
3.3	• Update chapter "Product description" • Update chapter "Configuration of the IO link master" • Chapter "Integrating IO link devices" replaced by chapter "Configuration of the IO link devices" • Chapter "Parameterizing of the IO link devices" replaced by chapters "Settings of the IO link devices" and "EPIxxxx, ERIxxxx - Setting of the IO-Link device parameters" • Update chapter "PDO-Assignment" and "Object description and parameterizing" • Update revision status • Update structure
3.2	• Update chapter "Commissioning" • Update structure • Chapter "Note - Power supply" added
3.1	• Update chapter "Technical data" • Update chapter "Version identification of EtherCAT devices" • Update structure • Update Notes • Update revision status • Chapter Disposal added
3.0	• Update chapter " Configuration of the IO link master" • Update structure • Update revision status
0.1 – 2.9	• archived

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**

KK - week of production (CW, calendar week)

YY - year of production

FF - firmware version

HH - hardware version

Example with serial number 12 06 3A 02:

12 - production week 12

06 - production year 2006

3A - firmware version 3A

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.

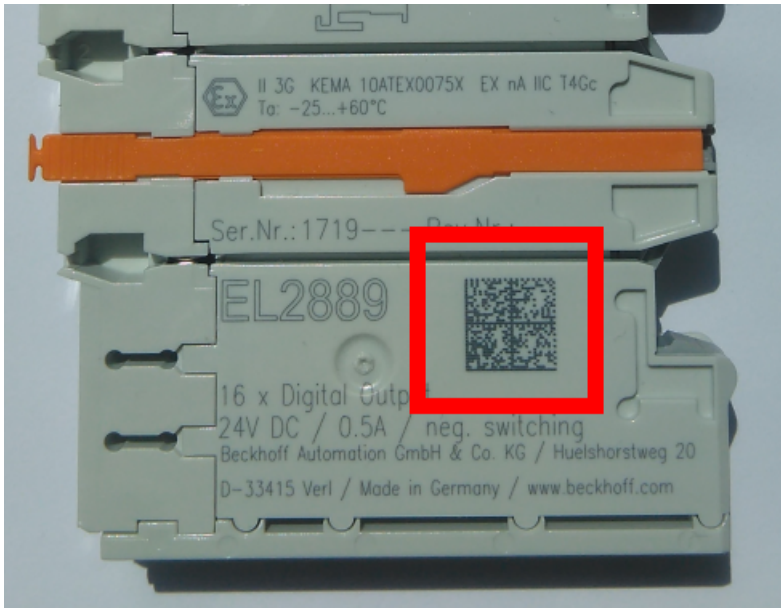


Fig. 2: BIC as data matrix code (DMC, code scheme ECC200)

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	Q1
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 **Q1** **51S**678294

Accordingly as DMC:



Fig. 3: Example DMC **1P**072222**SBTN**k4p562d7**1K**EL1809 **Q1** **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.

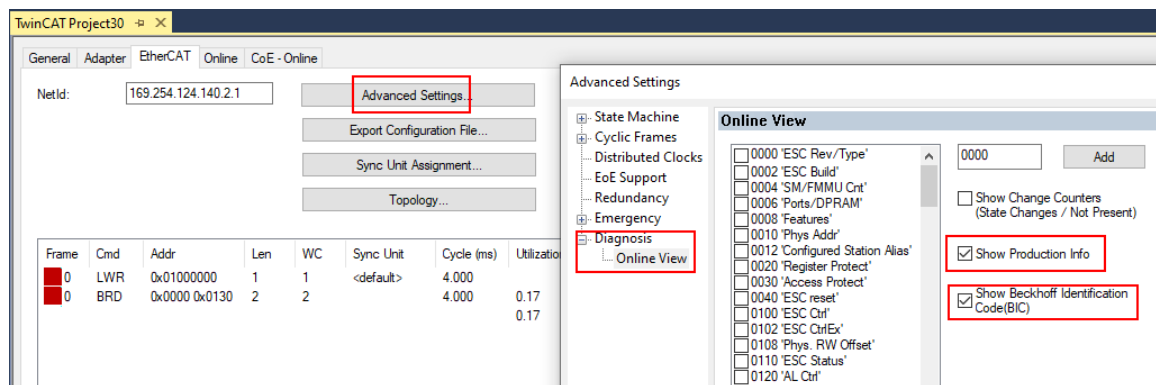
EtherCAT devices (IP20, IP67)

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:

No	Addr	Name	State	CRC	Fw	Hw	Production Data	ItemNo	BTN	Description	Quantity	BatchNo	SerialNo
1	1001	Term 1 (EK1100)	OP	0.0	0	0	---						
2	1002	Term 2 (EL1018)	OP	0.0	0	0	2020 KW36 Fr	072222	k4p562d7	EL1809	1		678294
3	1003	Term 3 (EL3204)	OP	0.0	7	6	2012 KW24 Sa						
4	1004	Term 4 (EL2004)	OP	0.0	0	0	---	072223	k4p562d7	EL2004	1		678295
5	1005	Term 5 (EL1008)	OP	0.0	0	0	---						
6	1006	Term 6 (EL2008)	OP	0.0	0	12	2014 KW14 Mo						
7	1007	Term 7 (EK1110)	OP	0	1	8	2012 KW25 Mo						

- 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	1P158442SBTN0008jckp1KELM3704 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_Uutilities* 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 EL6224, EL6224-0090 - Product description

2.1 EL6224 - Introduction

IO-Link Terminal

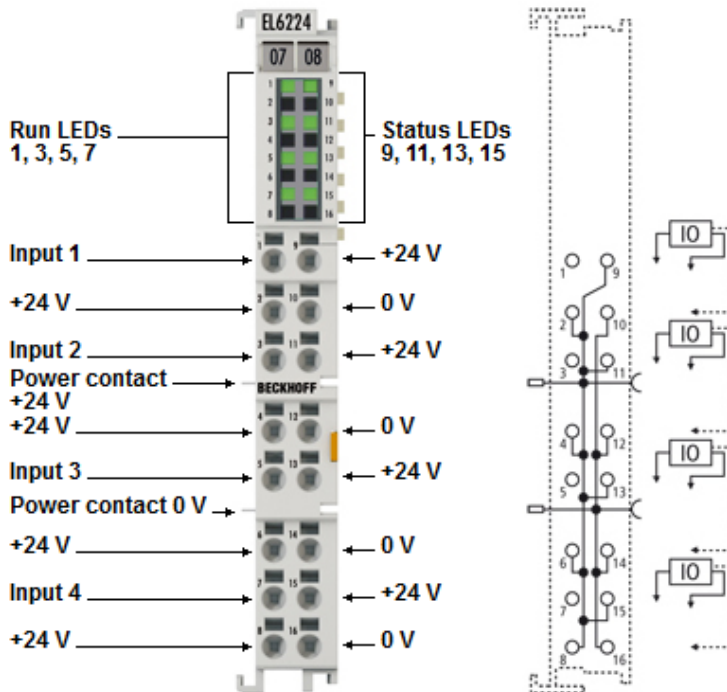


Fig. 4: EL6224

The EL6224 IO-Link terminal enables the connection of up to four IO-Link devices. These can be actuators, sensors or a combination of both. A point-to-point connection is used between the terminal and the device.

The terminal is parameterized via the EtherCAT master. IO-Link is designed as an intelligent link between the fieldbus level and the sensor, wherein parameterization information can be exchanged bidirectionally via the IO-Link connection.

The parameterization of the IO-Link devices with service data can be done from TwinCAT via ADS or via the integrated IO-Link configuration tool.

In the standard setting, the EL6224 functions as a 4-channel input terminal, 24 V_{DC}, which communicates if necessary with connected IO-Link devices, parameterizes them and, if necessary, changes their mode of operation.

Integration into the HD housing with 16 connection points enables each IO-Link device to be operated in 3-wire connection mode. The direct plug-in technique enables toolless construction.

Quick links

- [EtherCAT basics](#)
- [IO-Link basics \[► 22\]](#)
- [Object description and parameterizing \[► 83\]](#)
- [Configuration with TwinCAT \[► 108\]](#)
- [IO-Link - Configuration and parameterizing \[► 57\]](#)
- [ADS Error Codes \[► 184\]](#)

2.2 EL6224-0090 - Introduction

IO-Link Terminal, TwinSAFE Single Channel

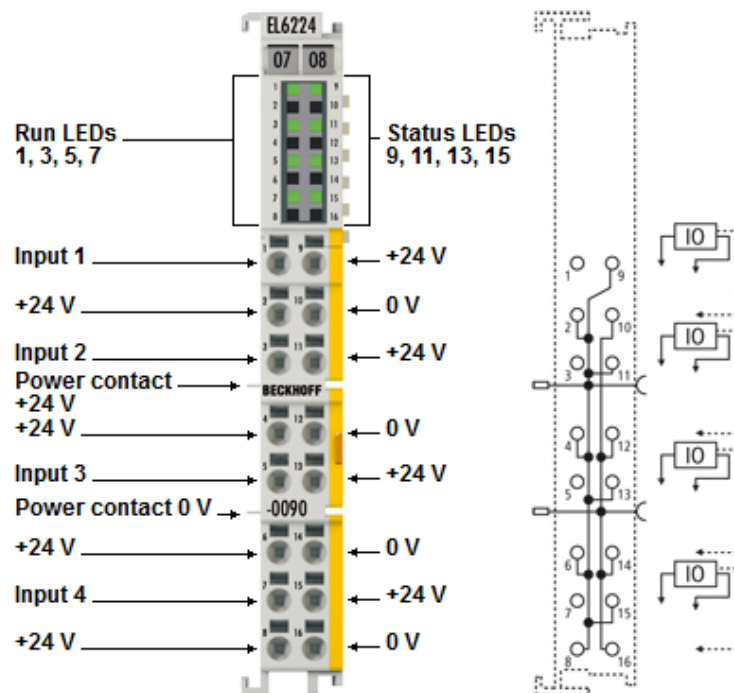


Fig. 5: EL6224-0090

The EL6224-0090 IO-Link terminal enables connection of up to four IO-Link devices, e.g. actuators, sensors or combinations of both. A point-to-point connection is used between the terminal and the device.

The terminal is parameterized via the EtherCAT master. IO-Link is designed as an intelligent link between the fieldbus level and the sensor, wherein parameterization information can be exchanged bidirectionally via the IO-Link connection.

The parameterization of the IO-Link devices with service data can be done from TwinCAT via ADS or via the integrated IO-Link configuration tool.

In the standard setting, the EL6224-0090 functions as a 4-channel input terminal, 24 V_{DC}, which communicates if necessary with connected IO-Link devices, parameterizes them and, if necessary, changes their mode of operation.

IO-Link devices can be operated in 3-wire connection mode. The direct plug-in technique enables toolless construction.

With the aid of the TwinSAFE SC technology [► 102] (TwinSAFE Single Channel) it is possible to make use of standard signals for safety tasks in any network or fieldbus. The standard functions and features of the I/Os remain available. The data from these TwinSAFE SC I/Os is fed to the TwinSAFE Logic, where they undergo safety-related multi-channel processing. In the Safety Logic the data originating from different sources is analyzed, checked for plausibility and submitted to a "voting". This is done by certified function blocks such as Scale, Compare/Voting (1oo2, 2oo3, 3oo5), Limit, etc. For safety reasons, however, at least one of the data sources must be a TwinSAFE SC component. The remainder of the data can originate from other standard Bus Terminals, drive controllers or measuring transducers.

With the aid of the TwinSAFE SC technology it is typically possible to achieve a safety level equivalent to PL d/Cat. 3 in accordance with EN ISO 13849-1 or SIL 2 in accordance with EN 62061.

Quick links

- [EtherCAT basics](#)
- [IO-Link basics](#) [► 22]
- [Object description and parameterization](#) [► 83]
- [Configuration with TwinCAT](#) [► 108]
- [IO-Link - Configuration and parameterization](#) [► 57]
- [ADS Error Codes](#) [► 184]
- [TwinCAT SC](#) [► 102]

2.3 EL6224, EL6224-0090 - Technical data

Technical data	EL6224	EL6224-0090
IO-Link interfaces	4	
Specification version	IO-Link V1.1	
Housing	HD housing with 16 connection points (from hardware version 04)	
Field voltage	24 V _{DC} via the power contacts	
Connection	3-wire	
Data transfer rates	4.8 kbaud, 38.4 kbaud and 230.4 kbaud	
Cable length between IO-Link Master and Device	max. 20 m	
MTBF (55°C)	-	>1.200.000 h
Supply voltage for electronics	via E-bus and power contacts	
Current consumption via E-bus	typ. 120 mA	
Electrical isolation	500 V (E-bus/field voltage)	
Supply current for devices	500 mA per device	
Configuration	via TwinCAT System Manager	
Weight	approx. 60 g	
Permissible ambient temperature range during operation	-25°C ... +60°C	
Permissible ambient temperature range during storage	-40°C ... +85°C	
Permissible relative humidity	95 %, no condensation	
Dimensions (W x H x D)	approx. 15 mm x 100 mm x 70 mm	
Mounting [► 42]	on 35 mm mounting rail conforms to EN 60715	
Increased mechanical load capacity	yes, see also Installation instructions [► 45] for terminals with increased mechanical load capacity	
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 class	IP20	
Installation position	variable	
Approvals/markings*	CE, UKCA, EAC, CCC cULus [► 40], ATEX [► 38], IECEx [► 39]	

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

Ex markings

Standard	Marking
ATEX	II 3 G Ex ec IIC T4 Gc
IECEx	Ex ec IIC T4 Gc

2.4 Start

For commissioning:

- mount the EL6224 as described in the chapter [Mounting and wiring \[► 37\]](#)
- configure the EL6224 in TwinCAT as described in the chapter [Configuration with TwinCAT \[► 108\]](#) and [IO-Link - Configuration and parameterizing \[► 57\]](#).

3 IO-Link basics

IO-Link is a communication system for connecting intelligent sensors and actuators to an automation system. The IEC 61131-9 standard specifies IO-Link under the designation "Single-drop digital communication interface for small sensors and actuators" (SDCI).

Both the electrical connection data and the communication protocol are standardized and summarized in the IO-Link Spec.

i IO-Link specification

The development of the Beckhoff EL6224/EJ6224 IO-Link master was subject to the IO-Link specification 1.1. At the time of the preparation of this documentation, the IO-Link specification is entering the IEC standardization and will be adopted in extended form as IEC 61131-9. The new designation SDCI will be introduced at the same time.

As a member of the respective committee, Beckhoff supports the development of IO-Link and reflects changes to the specification in its products.

3.1 IO-Link system layout

An IO-Link system consists of an IO-Link master and one or more IO-Link devices, i.e. sensors or actuators. The IO-Link master provides the interface to the higher-level controller and controls communication with the connected IO-Link devices.

The Beckhoff EL6224/EJ6224 IO-Link Master Terminal has four IO-Link ports. One IO-Link device can be connected to each of them. IO-Link is therefore not a fieldbus, but a point-to-point connection (see following figure).

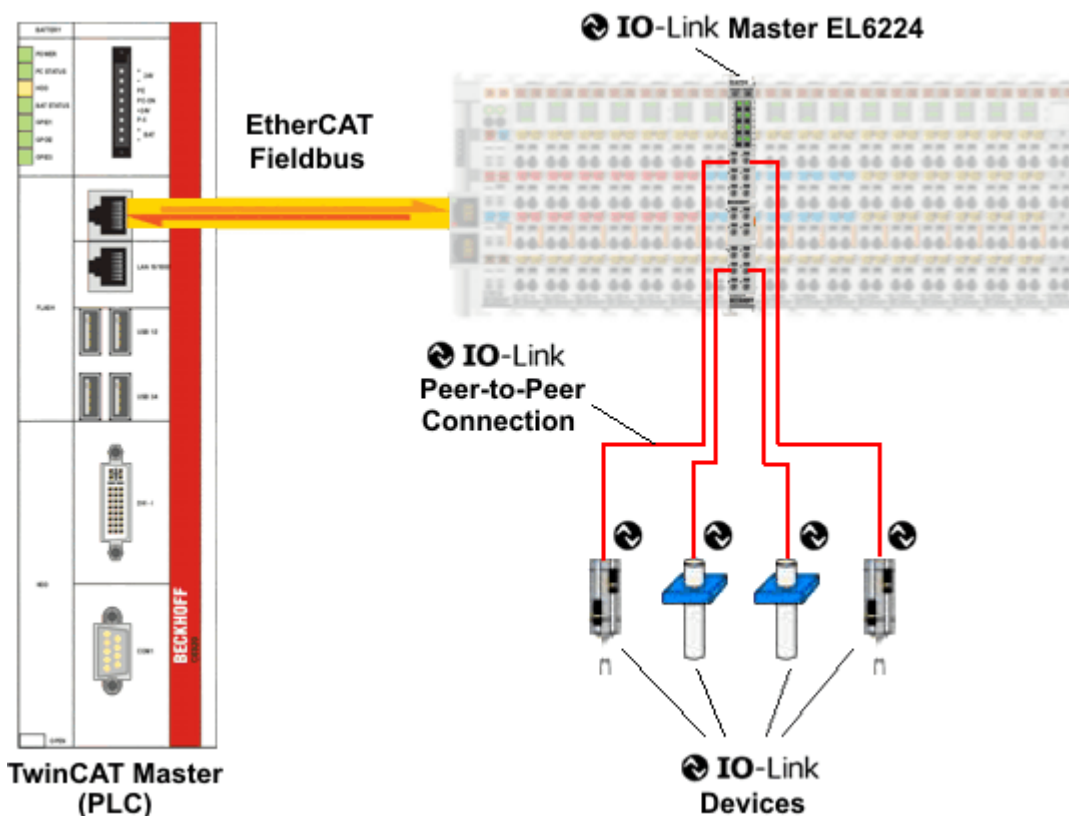


Fig. 6: IO-Link point-to-point communication using the example of the EL6224 IO-Link Master

⚠ CAUTION

Damage to the devices possible!

The IO-Link Devices must be powered from the 24 V supply of the EL6224/EJ6224, otherwise the IO-Link port may be damaged!

3.2 Establishment of IO Link communication

The establishment of the IO-Link communication is illustrated in Fig. *Establishment of IO-Link communication*. This illustrates in particular the sequence when automatically scanning [► 63] the IO-Link port.

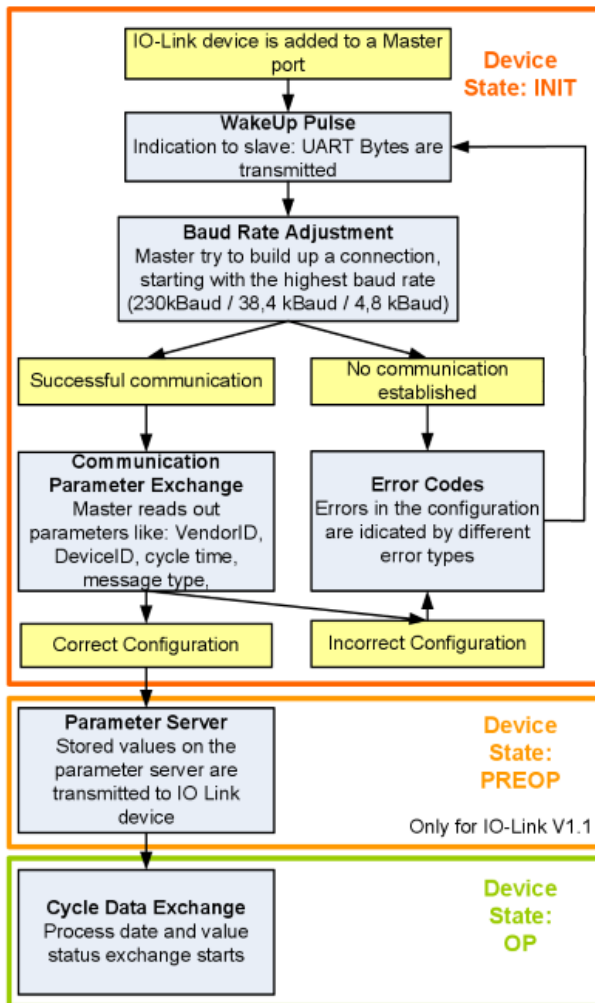


Fig. 7: Establishment of IO Link communication

- If an IO-Link device is connected to a master port, the master attempts to establish communication. A defined signal level, the **wake-up pulse**, signals to the device that UART bytes are to be sent from now on.
From this point on, all data will be interpreted by the IO-Link device as UART bytes.
- The master runs through all baud rates [► 25], starting with the fastest baud rate (COM3 = 230 kbaud). A successful connection has been established when the slave responds to the wake-up pulse.
- First of all the master reads the **basic parameters** (Vendor ID, Device ID, process data length, telegram type and cycle time) and compares them with the existing configuration.
- If no connection could be established to the device, or if the saved parameters differ from those read, the corresponding error is output.
- If the saved parameters differ from those read, the IO-Link device changes to the PREOP state.
If the IO-Link device specification is V1.1, the parameter server [► 24] is now executed. If the IO-Link device specification is V1.0, this step is omitted and the device changes directly to OP.
- Finally the cycle time is written and the device changes to OP. After that the master cyclically exchanges data with the device.

3.3 Device description IODD

IO-Link devices possess individual system information in the form of an IO device description (IODD), which contains:

- Communication features
- Device parameters with value range and default values
- Identification, process and diagnostic data
- Device data
- Text description
- Picture of the device
- Vendor's logo

If the IODD is imported, then the device data are automatically detected during [automatic scanning](#) [► 63] with TwinCAT and adopted in the System Manager.

3.4 Parameter server

In order to be able to use the functionality of the parameter server, both the IO-Link master and the IO-Link device must be specified to V1.1. The IO-Link revision of the device can be read for the individual port under [Settings](#) [► 70]. All IO-Link masters from Beckhoff with current firmware support the IO-Link specification V1.1.

- The parameter server in the IO-Link master contains parameter data that are saved in the IO-Link device. The memory capacity is max. 2 kbyte (including header).
If the IO-Link device is exchanged, then the data are loaded from the parameter server into the new device. The requirement for this is that the device is of the same type (VendorID and DeviceID must be the same).
- If a new IO-Link device is configured, then the IO-Link master loads the parameters from the IO-Link device into the parameter server when starting for the first time.
Data from other IO-Link devices that are already configured (VendorID and DeviceID do not correspond to the configured device) are overwritten.
- At each further start the IO-Link master uses a checksum to check whether the data in the parameter server correspond to those on the IO-Link device and if necessary downloads them to the device.
- If the parameters change during the device runtime, this can be reported to the Master via the [store button](#) [► 78] ([ParamDownloadStore](#) [► 79]). The master then starts the parameter server with an upload.
- By default the event is not set each time the parameters are written, therefore the end of the parameterization procedure has to be reported to the IO-Link device via the [store button](#) [► 78] ([ParamDownloadStore](#) [► 79]).
The IO-Link device then sends the corresponding event to the master. The data are loaded into the parameter server.
- In the case of a pre-programmed IO-Link device, no download takes place from the parameter server to the device.

3.5 Data transfer rate

An IO-Link master according to specification V1.1 supports all three transmission methods and automatically adjusts the baud rate to that of the IO-Link device.

An IO-Link device usually supports only one baud rate. IO-Link devices with different baud rates can be connected to the various ports of the master.

- COM1 = 4.8 kbaud
- COM2 = 38.4 kbaud
- COM3 = 230.4 kbaud

4 Basics communication

4.1 EtherCAT basics

Please refer to the [EtherCAT System Documentation](#) for the EtherCAT fieldbus basics.

4.2 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.

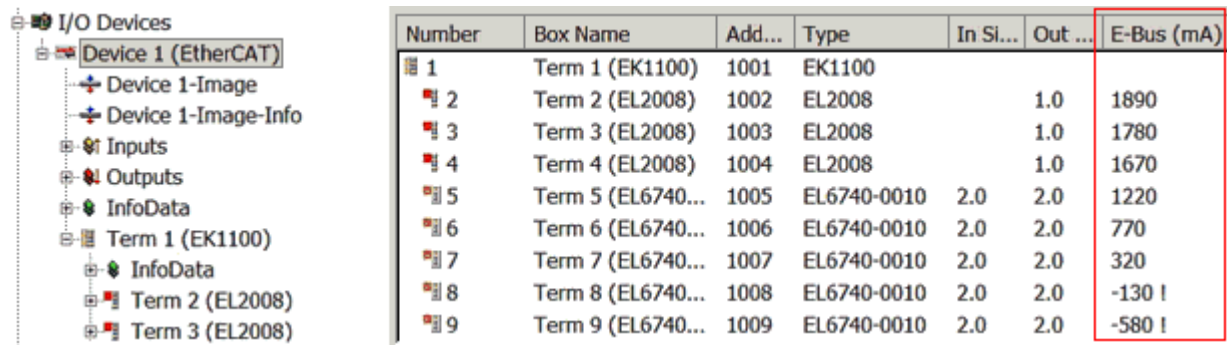
- cable sets ZK1090-9191-xxxx respectively
- RJ45 connector, field assembly ZS1090-0005
- EtherCAT cable, field assembly ZB9010, ZB9020

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.



The screenshot shows the 'I/O Devices' tree on the left with 'Device 1 (EtherCAT)' expanded, showing 'Term 1 (EK1100)', 'Term 2 (EL2008)', and 'Term 3 (EL2008)'. The main table on the right lists 9 terminals with their addresses, types, and E-Bus current values. The 'E-Bus (mA)' column is highlighted with a red box.

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 (EL6740-0010)	1005	EL6740-0010	2.0	2.0	1220
6	Term 6 (EL6740-0010)	1006	EL6740-0010	2.0	2.0	770
7	Term 7 (EL6740-0010)	1007	EL6740-0010	2.0	2.0	320
8	Term 8 (EL6740-0010)	1008	EL6740-0010	2.0	2.0	-130 I
9	Term 9 (EL6740-0010)	1009	EL6740-0010	2.0	2.0	-580 I

Fig. 8: 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!

4.3 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:

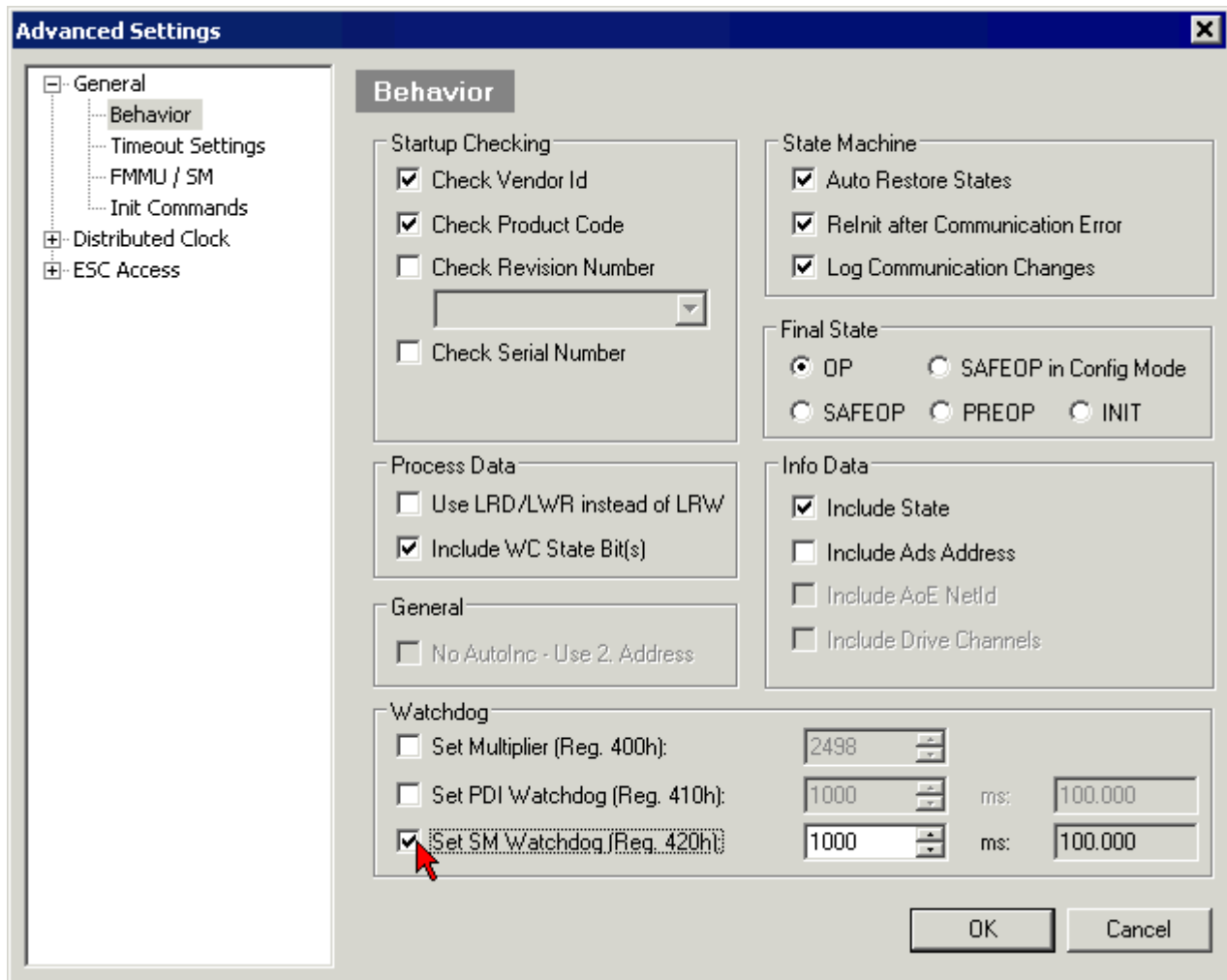


Fig. 9: eEtherCAT tab -> Advanced Settings -> Behavior -> Watchdog

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 = $[1/25 \text{ MHz} * (\text{Watchdog multiplier} + 2)] * \text{SM/PDI watchdog}$

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.

4.4 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.

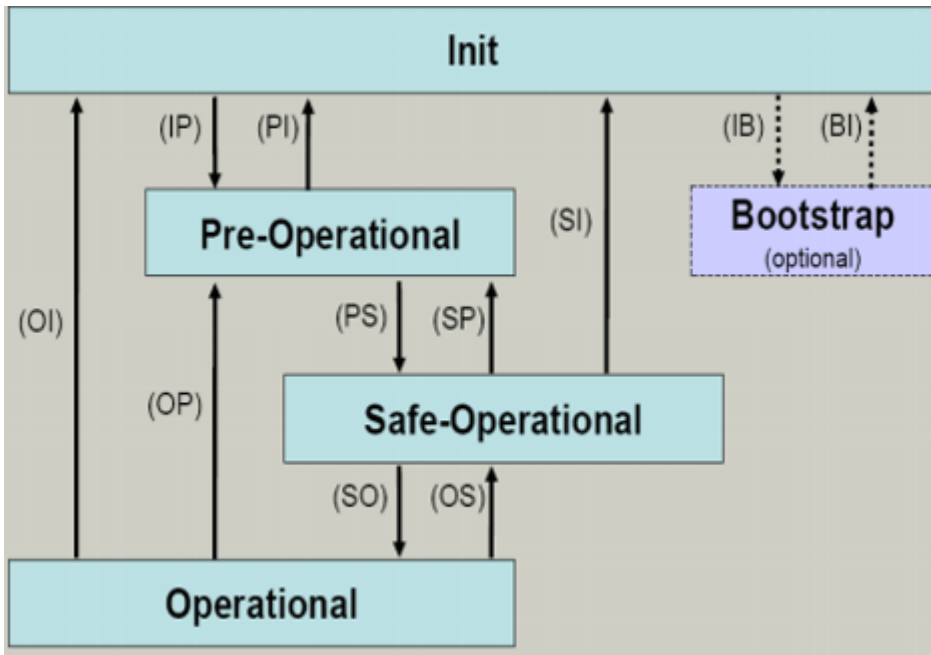


Fig. 10: States of the EtherCAT State Machine

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

I

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.

4.5 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:

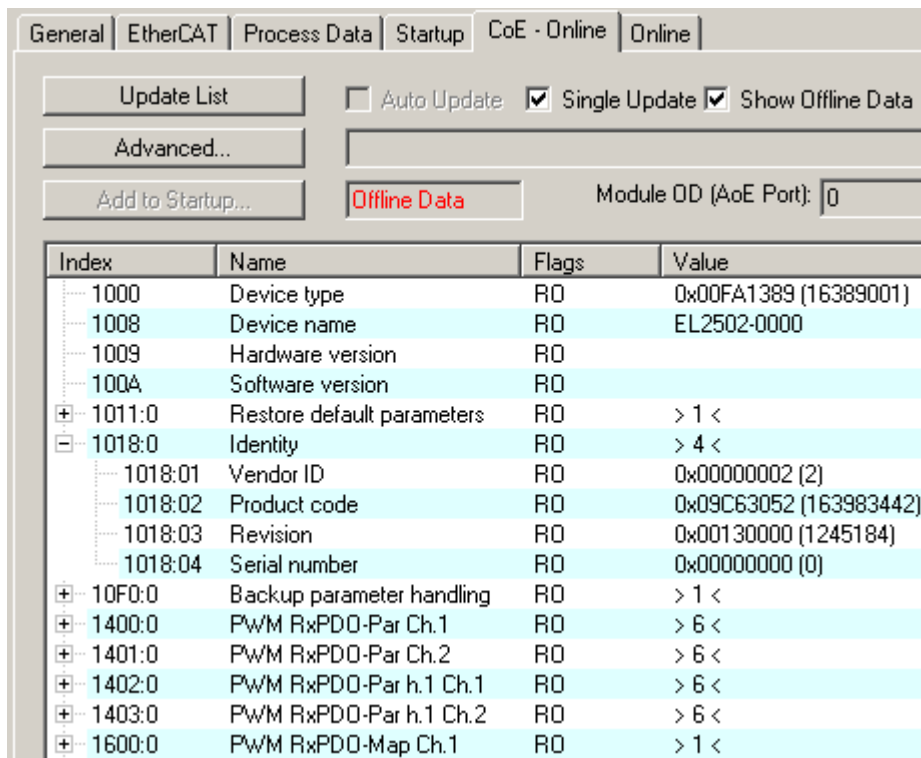


Fig. 11: "CoE Online" tab

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.

i 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.

i 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.

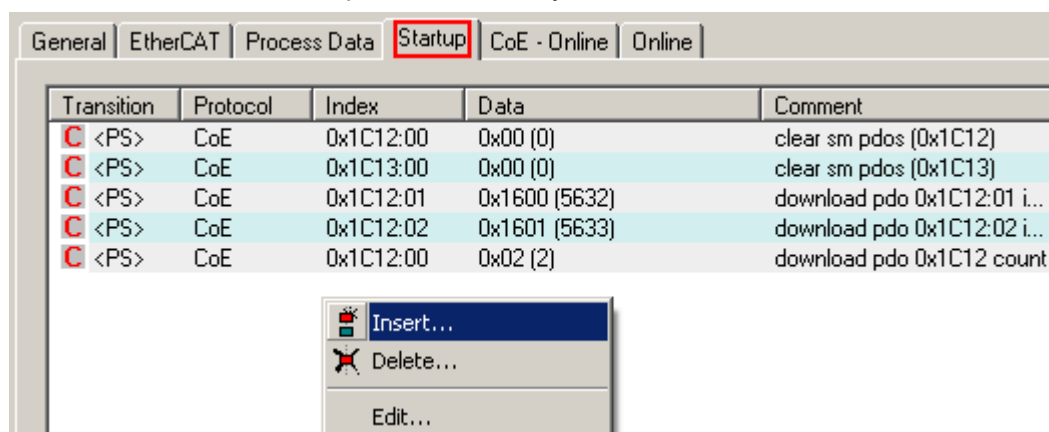


Fig. 12: Startup list in the TwinCAT System Manager

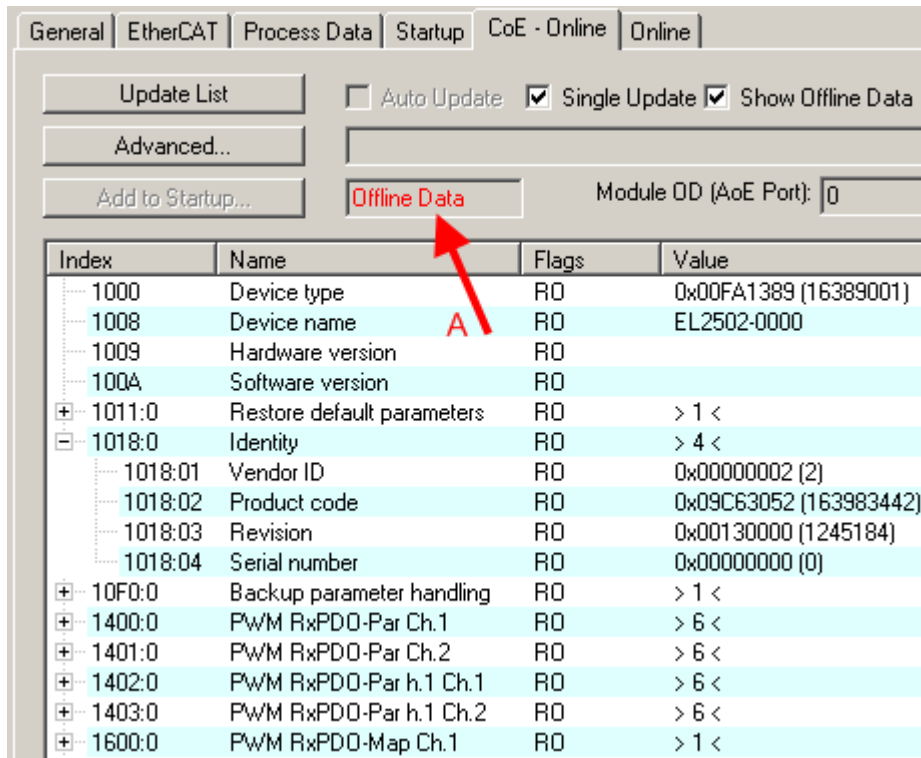
The Startup list may already contain values that were configured by the System Manager based on the ESI specifications. Additional application-specific entries can also be created.

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.



The screenshot shows the 'CoE - Online' tab in a software interface. The 'Offline Data' label is highlighted in red in the table header. A red arrow points to this label. The table lists various CoE objects with their indices, names, flags, and values. The 'Identity' section (1018:0) is expanded, showing details like Vendor ID, Product code, Revision, and Serial number.

Index	Name	Flags	Value
1000	Device type	RO	0x00FA1389 (16389001)
1008	Device name	RO	EL2502-0000
1009	Hardware version	RO	
100A	Software version	RO	
1011:0	Restore default parameters	RO	> 1 <
1018:0	Identity	RO	> 4 <
1018:01	Vendor ID	RO	0x00000002 (2)
1018:02	Product code	RO	0x09C63052 (163983442)
1018:03	Revision	RO	0x00130000 (1245184)
1018:04	Serial number	RO	0x00000000 (0)
10F0:0	Backup parameter handling	RO	> 1 <
1400:0	PWM RxD0-Par Ch.1	RO	> 6 <
1401:0	PWM RxD0-Par Ch.2	RO	> 6 <
1402:0	PWM RxD0-Par h.1 Ch.1	RO	> 6 <
1403:0	PWM RxD0-Par h.1 Ch.2	RO	> 6 <
1600:0	PWM RxD0-Map Ch.1	RO	> 1 <

Fig. 13: 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.

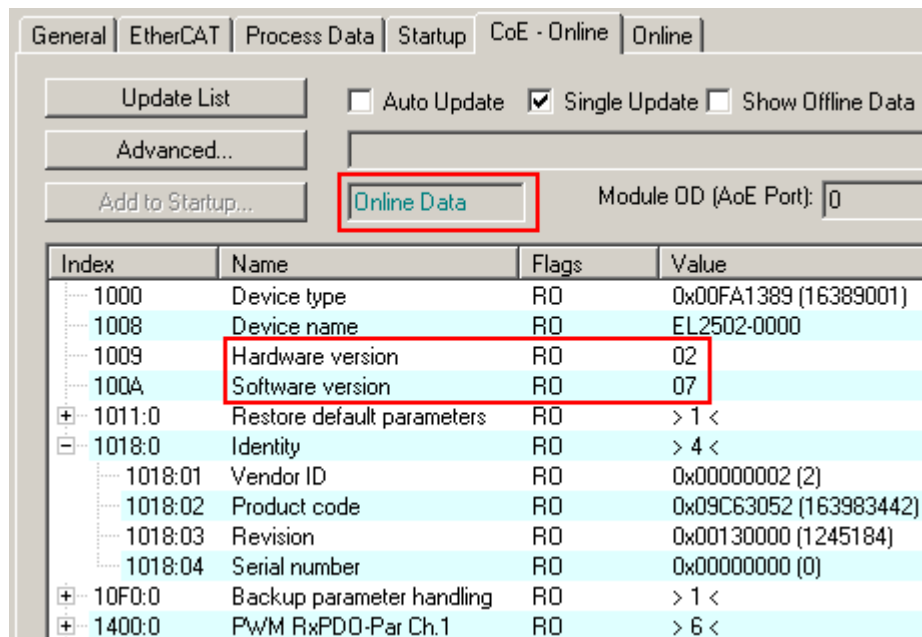


Fig. 14: Online list

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.

4.6 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](#).

5 Mounting and wiring

5.1 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.

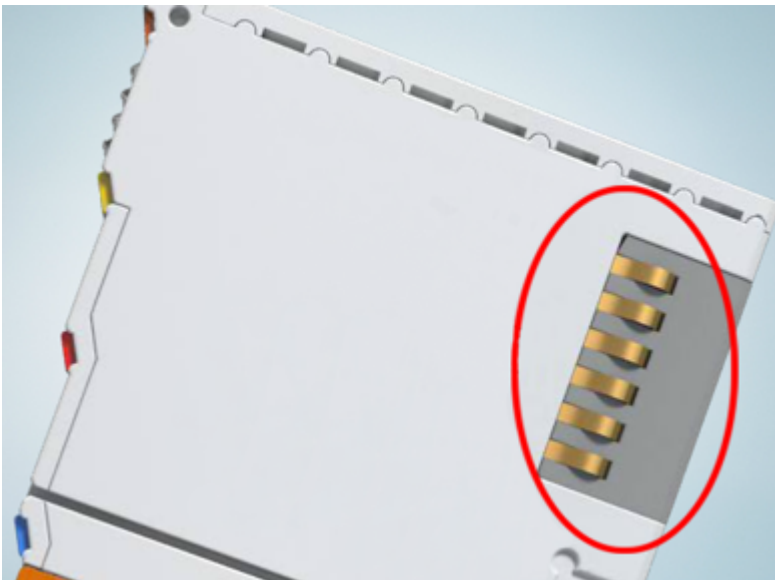


Fig. 15: Spring contacts of the Beckhoff I/O components

5.2 Explosion protection

5.2.1 ATEX - Special conditions (extended temperature range)

WARNING

Observe the special conditions for the intended use of Beckhoff fieldbus components with extended temperature range (ET) in potentially explosive areas!

- The certified components are to be installed in a suitable housing that guarantees a protection class of at least IP54 in accordance with EN 60079-7! The environmental conditions during use are thereby to be taken into account!
- If the temperatures during rated operation are higher than 70°C at the feed-in points of cables, lines or pipes, or higher than 80°C at the wire branching points, then cables must be selected whose temperature data correspond to the actual measured temperature values!
- Observe the permissible ambient temperature range of -25 to 60°C for the use of Beckhoff fieldbus components with extended temperature range (ET) in potentially explosive areas!
- Measures must be taken to protect against the rated operating voltage being exceeded by more than 40% due to short-term interference voltages!
- The individual terminals may only be unplugged or removed from the Bus Terminal system if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- The connections of the certified components may only be connected or disconnected if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- The fuses of the KL92xx/EL92xx power feed terminals may only be exchanged if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- Address selectors and ID switches may only be adjusted if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!

Standards

The fundamental health and safety requirements are fulfilled by compliance with the following standards:

- EN IEC 60079-0:2018
- EN 60079-7:2015+A1:2018

Marking

The Beckhoff fieldbus components with extended temperature range (ET) certified according to regulations for potentially explosive areas bear the following marking:



II 3 G Ex ec IIC T4 Gc
KEMA 10 ATEX0075 X
DEKRA 22UKEX6024X
Ta: -25 ... + 60°C

5.2.2 IECEx - Special conditions

⚠ WARNING

Observe the special conditions for the intended use of Beckhoff fieldbus components in potentially explosive areas!

- For gas: The equipment shall be installed in a suitable enclosure providing a degree of protection of IP54 according to IEC 60079-7, taking into account the environmental conditions under which the equipment is used!
- The equipment shall only be used in an area of at least pollution degree 2, as defined in IEC 60664-1!
- Provisions shall be made to prevent the rated voltage from being exceeded by transient disturbances of more than 119 V!
- If the temperatures during rated operation are higher than 70°C at the feed-in points of cables, lines or pipes, or higher than 80°C at the wire branching points, then cables must be selected whose temperature data correspond to the actual measured temperature values!
- Observe the permissible ambient temperature range for the use of Beckhoff fieldbus components in potentially explosive areas!
- The individual terminals may only be unplugged or removed from the Bus Terminal system if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- The connections of the certified components may only be connected or disconnected if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- Address selectors and ID switches may only be adjusted if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- The front hatch of certified units may only be opened if the supply voltage has been switched off or a non-explosive atmosphere is ensured!

Standards

The fundamental health and safety requirements are fulfilled by compliance with the following standards:

- IEC 60079-0:2017 (Edition 7.0)
- IEC 60079-7:2017 (Edition 5.1)

Marking

Beckhoff fieldbus components that are certified in accordance with IECEx for use in areas subject to an explosion hazard bear the following marking:

IECEx DEK 16.0078 X

Ex ec IIC T4 Gc

5.2.3 Continuative documentation for ATEX and IECEx

NOTICE



Continuative documentation about explosion protection according to ATEX and IECEx

Pay also attention to the continuative documentation

Ex. Protection for Terminal Systems

Notes on the use of the Beckhoff terminal systems in hazardous areas according to ATEX and IECEx,

that is available for [download](#) within the download area of your product on the Beckhoff homepage www.beckhoff.com!

5.3 UL notice

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



5.4 Notes on TwinSAFE SC

5.4.1 Safety instructions

Before installing and commissioning the TwinSAFE components please read the safety instructions in the foreword of this documentation.

5.4.2 Environmental conditions

Please ensure that the TwinSAFE components are only transported, stored and operated under the specified conditions (see technical data)!

WARNING

Risk of injury!

The TwinSAFE components must not be used under the following operating conditions.

- under the influence of ionizing radiation (that exceeds the level of the natural environmental radiation)
- in corrosive environments¹
- in an environment that leads to unacceptable soiling of the TwinSAFE component

¹ A corrosive environment exists when corrosion damage becomes apparent.

NOTICE

Electromagnetic compatibility

The TwinSAFE components comply with the current standards on electromagnetic compatibility with regard to spurious radiation and immunity to interference in particular.

However, in cases where devices such as mobile phones, radio equipment, transmitters or high-frequency systems that exceed the interference emissions limits specified in the standards are operated near TwinSAFE components, the function of the TwinSAFE components may be impaired.

5.4.3 Transport / storage

Use the original packaging in which the components were delivered for transporting and storing the TwinSAFE components.

CAUTION

Note the specified environmental conditions

Please ensure that the digital TwinSAFE components are only transported and stored under the specified environmental conditions (see technical data).

5.4.4 Control cabinet / terminal box

The TwinSAFE terminals must be installed in a control cabinet or terminal box with IP54 protection class according to IEC 60529 as a minimum.

5.5 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

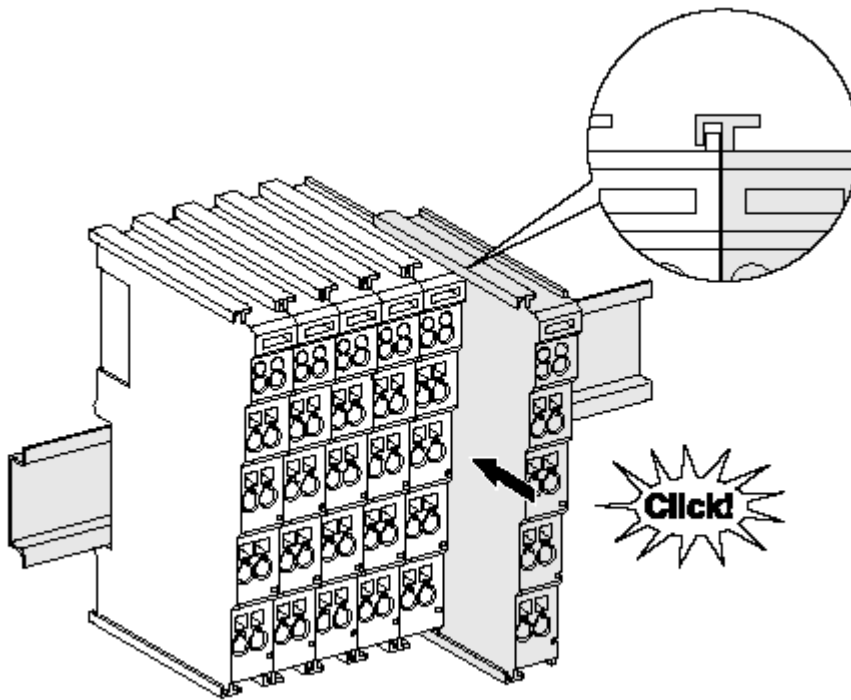


Fig. 16: Attaching on mounting rail

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

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

NOTICE

Ground the mounting rail!

Ensure that the mounting rail is sufficiently grounded.

Connections within a bus terminal block

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

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



Power Contacts

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

Power contact $\perp$

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

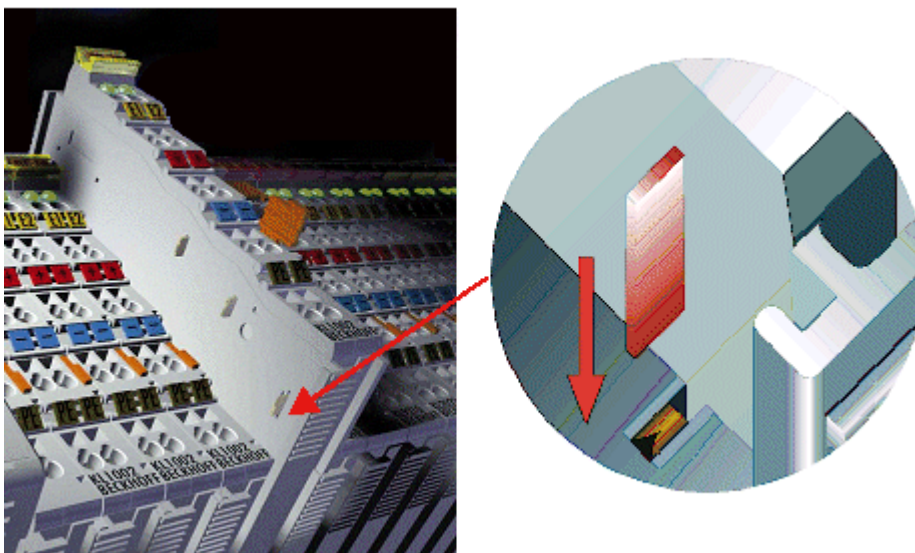


Fig. 17: Power contact on left side

⚠ WARNING

Risk of electric shock!

The power contact labeled $\perp$ must not be used for other potentials!

NOTICE

Possible damage of the device

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

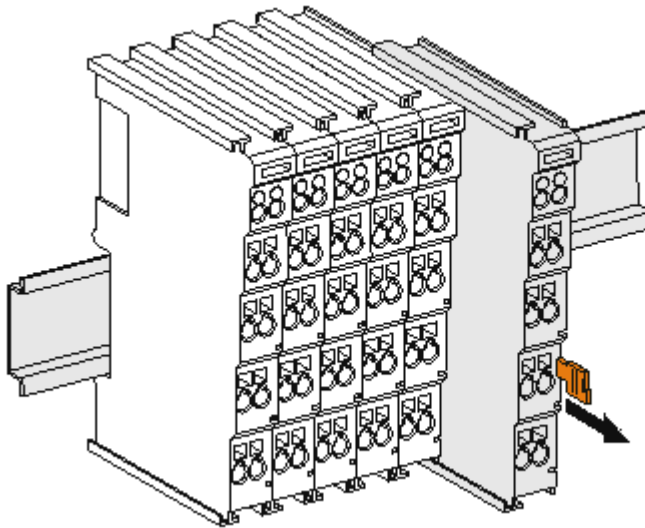
Disassembly

Fig. 18: 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.

5.6 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.

5.7 Connection

5.7.1 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 mm² and 2.5 mm² 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 [► 49]!

5.7.2 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

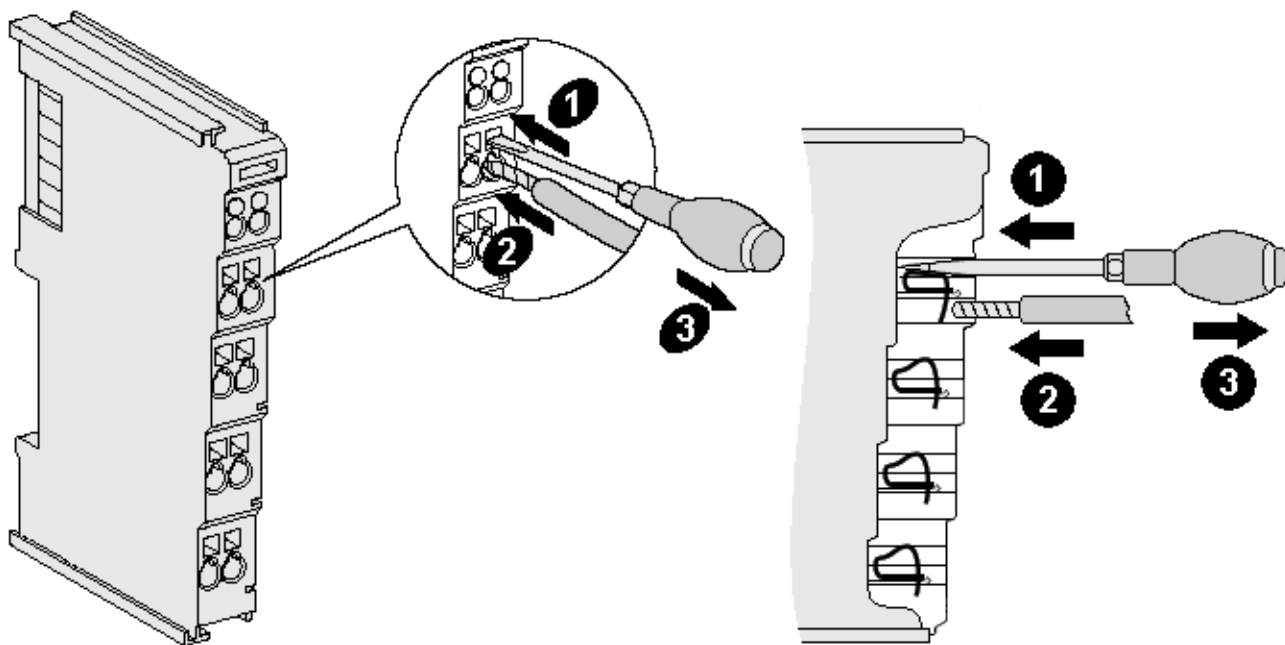


Fig. 22: Connecting a cable on a terminal point

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 mm ²	0.08 ... 2.5 mm ²
Wire size width (fine-wire conductors)	0.08 ... 2.5 mm ²	0.08 ... 2.5 mm ²
Wire size width (conductors with a wire end sleeve)	0.14 ... 1.5 mm ²	0.14 ... 1.5 mm ²
Wire stripping length	8 ... 9 mm	9 ... 10 mm

High Density Terminals (HD Terminals [► 47]) 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 mm ²
Wire size width (fine-wire conductors)	0.25 ... 1.5 mm ²
Wire size width (conductors with a wire end sleeve)	0.14 ... 0.75 mm ²
Wire size width (ultrasonically compacted [ultrasonically welded] strands)	only 1.5 mm ² (see <u>notice</u> [► 47])
Wire stripping length	8 ... 9 mm

5.7.3 Shielding**Shielding**

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

5.8 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.

5.9 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)

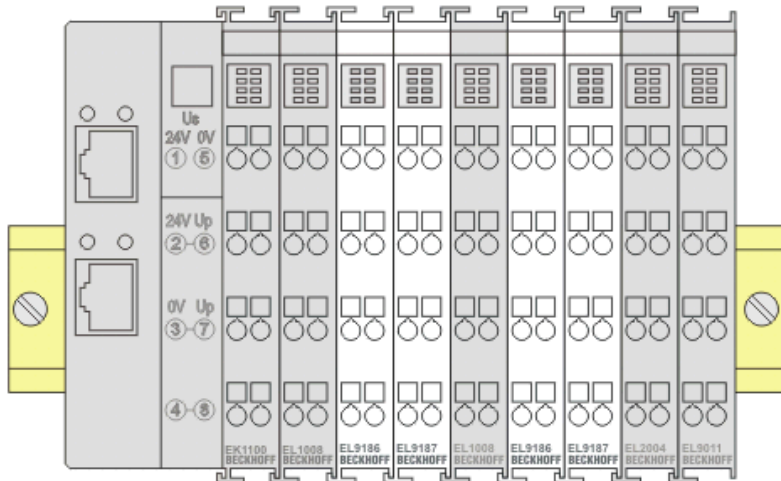


Fig. 23: Correct positioning

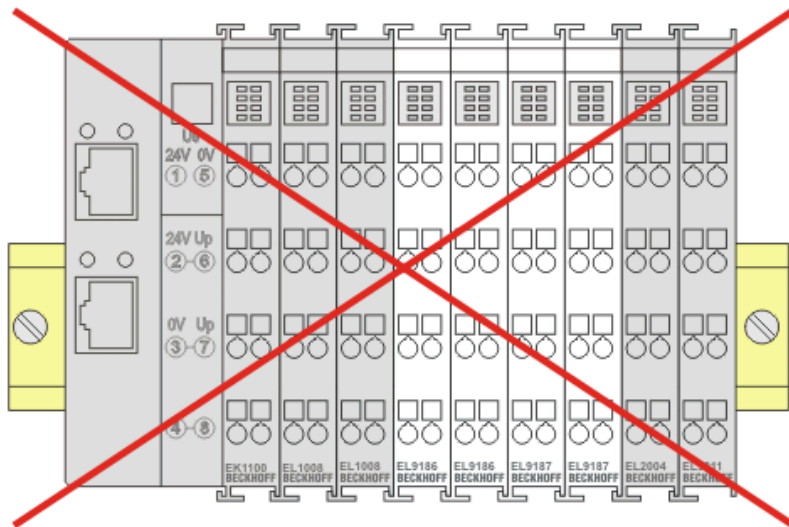


Fig. 24: Incorrect positioning

5.10 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.

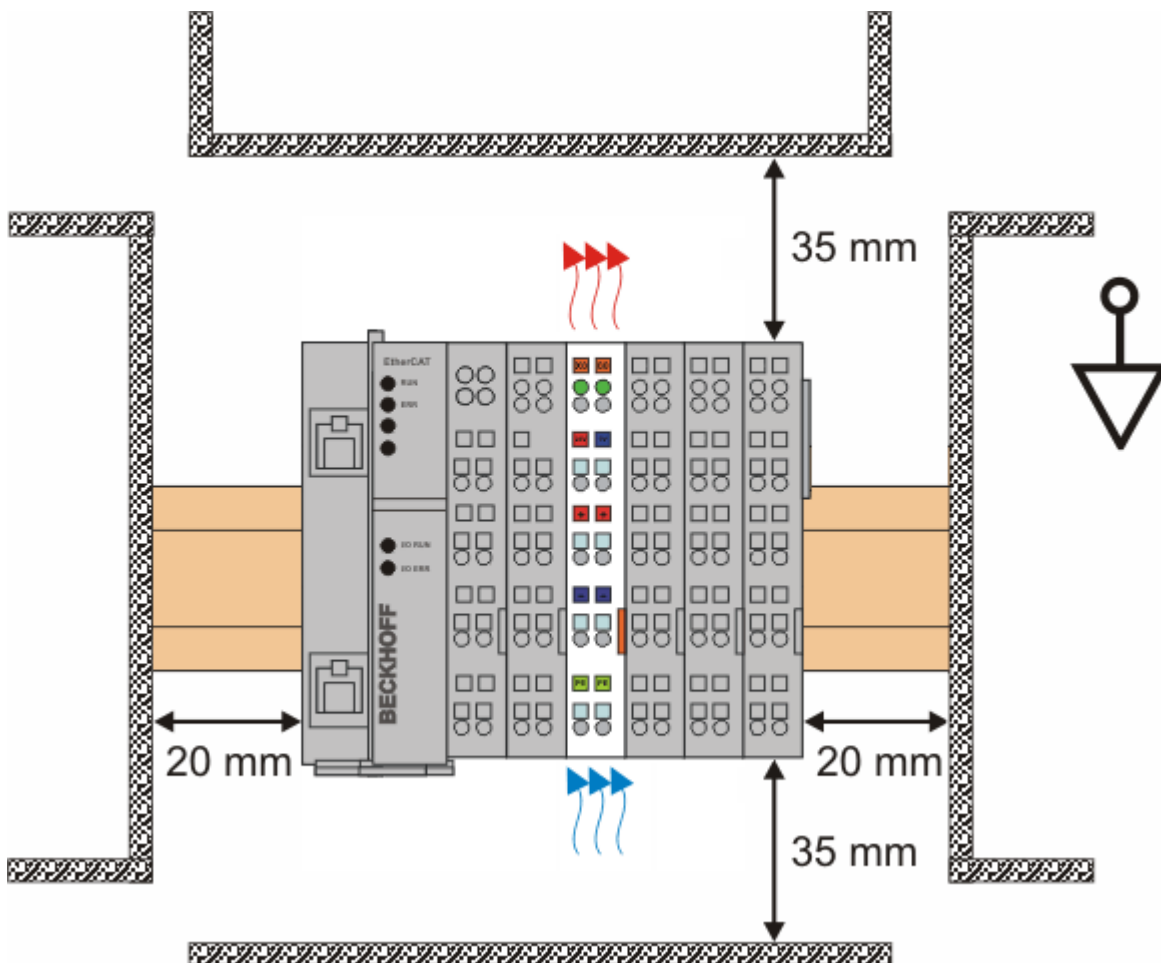


Fig. 25: Recommended distances for standard installation position

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.

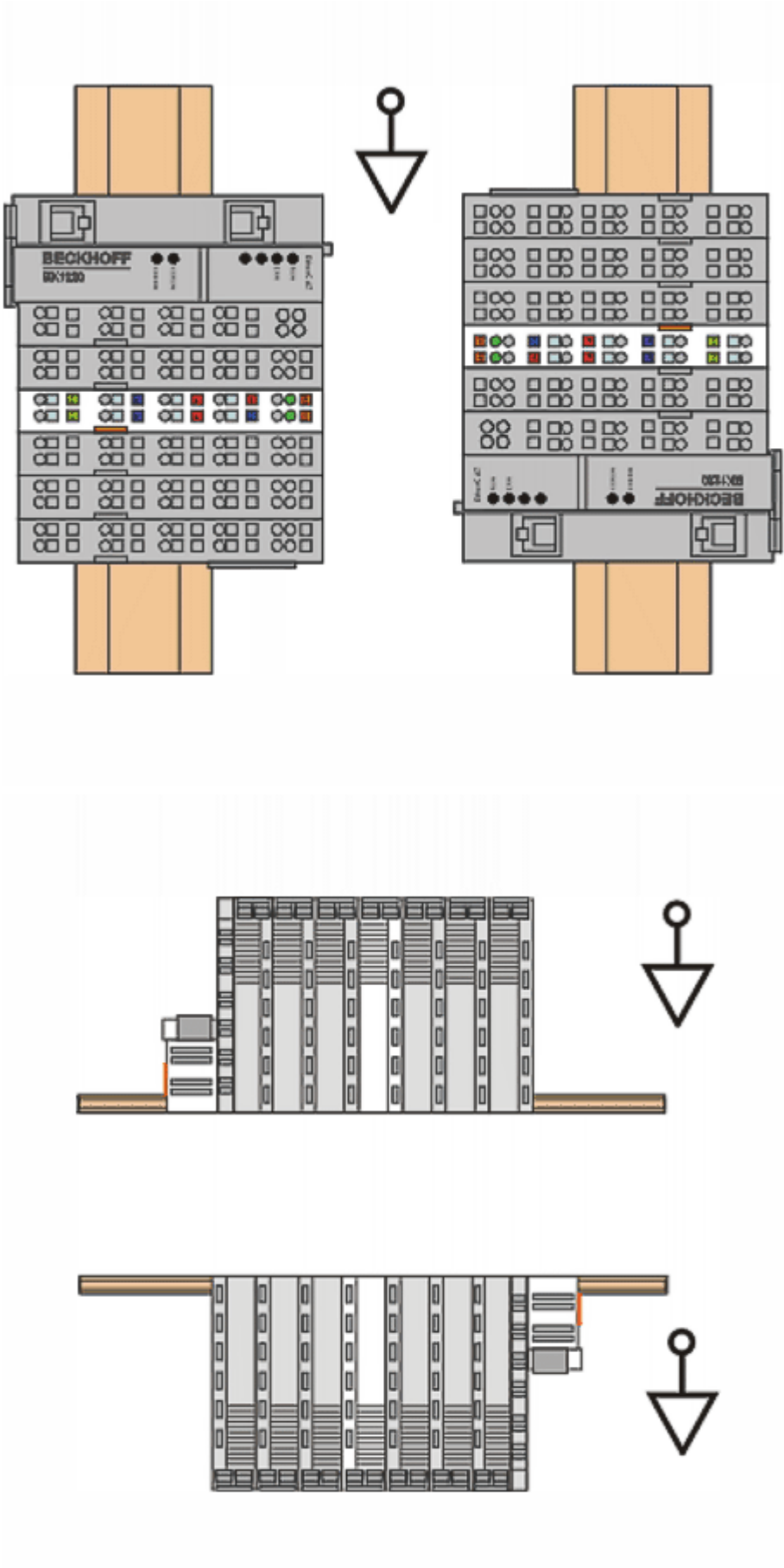


Fig. 26: Other installation positions

5.11 EL6224, EL6224-0090 - LEDs and connection

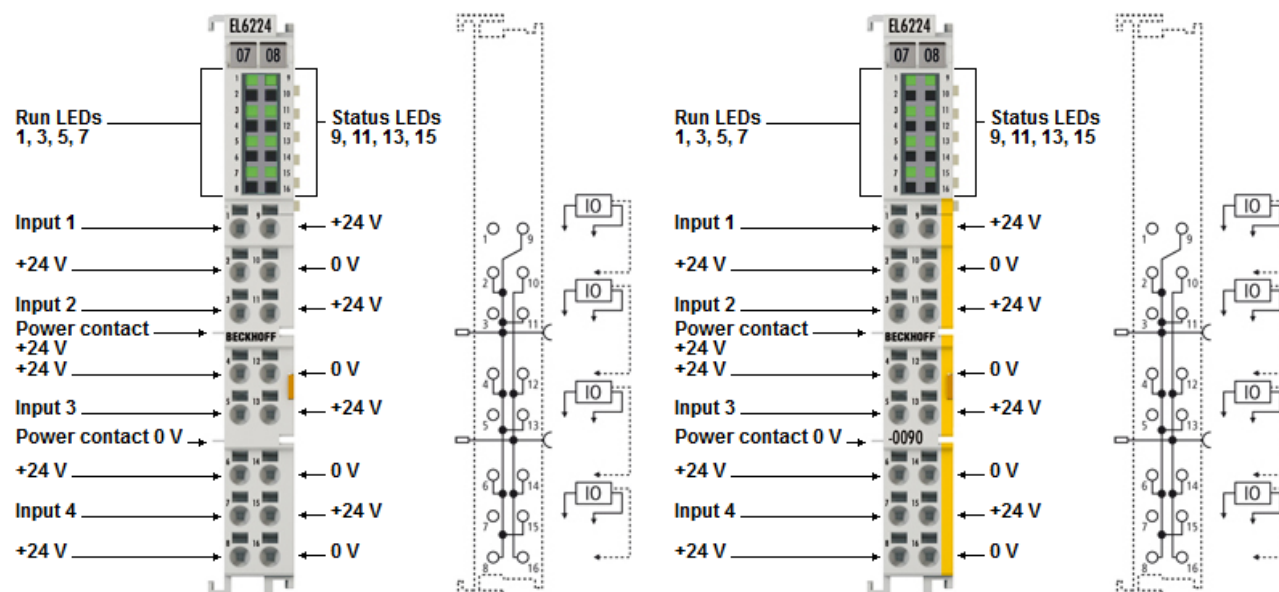


Fig. 27: EL6224, EL6224-0090 - LEDs and connection (HD housing)

Connection		
Terminal point		Description
Name	No.	
Input 1	1	Input 1
+ 24 V	2	+ 24 V
Input 2	3	Input 2
+ 24 V	4	+ 24 V
Input 3	5	Input 3
+ 24 V	6	+ 24 V
Input 4	7	Input 4
+ 24 V	8	+ 24 V
+ 24 V	9	+ 24 V
0 V	10	0 V
+ 24 V	11	+ 24 V
0 V	12	0 V
+ 24 V	13	+ 24 V
0 V	14	0 V
+ 24 V	15	+ 24 V
0 V	16	0 V

⚠ CAUTION

Damage to the devices possible!

The IO-Link Devices must be powered from the 24 V supply of the EL6224/EJ6224, otherwise the IO-Link port may be damaged!

LEDs			
LED	Color	Meaning	
RUN*	green	These LEDs indicate the terminal's operating state:	
		off	State of the EtherCAT State Machine [► 29]: INIT = initialization of the terminal or BOOTSTRAP = function for firmware updates [► 187] 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 [► 159] 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
State Ch. 1 - 4	green	on / off	Status of the signal line (if configured as STD in / out)
		flashes briefly twice	IO-Link communication is being established
		permanently flashing	IO-Link communication established and functioning
* If several RUN LEDs are present, all of them have the same function.			

5.12 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.

6 IO link - Configuration and parameterizing

6.1 Configuration of the IO link master

i EtherCAT XML device description and configuration files

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

When adding the IO-Link master in the TwinCAT System Manager (see chapter [Setup](#) in the [TwinCAT System Manager](#) of the EtherCAT system description), an additional tab called "IO-Link" is created (fig. *IO-Link tab*). A detailed description can be found in chapter [Configuration of the IO-Link devices](#) [► 58]

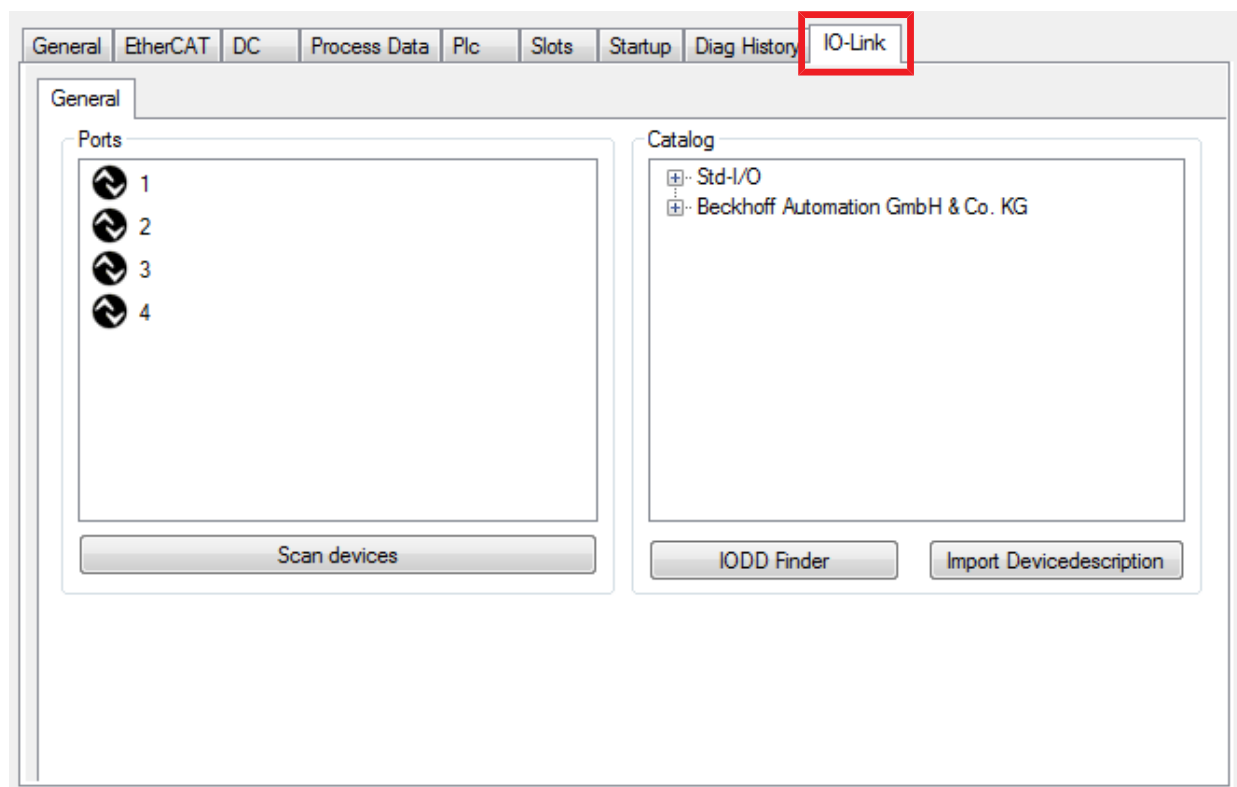


Fig. 28: "IO Link" tab

i IO-Link Extension

If the tab "IO-Link" is not displayed, the associated System Manager extension is missing. The System Manager extension for EL6224-00xx is required for TwinCAT version 2.10, build 1325 to 1330.

- If your System Manager version or TwinCAT3 does not yet provide this support, it can be installed later if necessary. Please contact our [Support](#) [► 201].

6.2 Configuration of the IO-Link devices

The configuration of the IO link devices is carried out in the IO link configuration tool. Configure the IO link device as described below.

✓ Requirement: an IO-Link master has been added in the Solution Explorer under the "I/O" entry.

1. Open the IO link configuration tool [► 58].
2. Import the IODD file of the IO link device [► 62].
3. Assign devices to ports.
 - ⇒ Assign a device to a port [► 59].
 - ⇒ Configure a port as digital in- or output [► 62].
4. Remove a device from a port [► 68].
5. Activate the IO link configuration [► 69], so that changes become effective.

6.2.1 Open the IO link configuration tool

✓ Requirement: an IO-Link master has been added in the Solution Explorer under the "I/O" entry.

1. Double-click on the IO-Link master.
 - ⇒ A device editor for the IO-Link master opens.
 2. Click on the "IO-Link" tab.
 - ⇒ The IO-Link configuration tool opens. The configuration tool contains two fields:
- „Ports“
The left-hand field "Ports" shows a list of the ports of the IO-Link master. If a device has been assigned to a port, the device designation is shown next to the port.
 - „Catalog“
The right-hand field "Catalog" shows the device catalog.
The device catalog contains an alphabetically sorted list of the IO-Link devices for which a device description (IODD) exists in the local TwinCAT installation.
The IODDs for the EPIxxxx, ERIxxxx IO-Link Box modules from Beckhoff can be downloaded via the [Download finder](#). The downloaded zip file contains the IODD device description files for the Beckhoff EPIxxxx, ERIxxxx IO-Link Box modules.

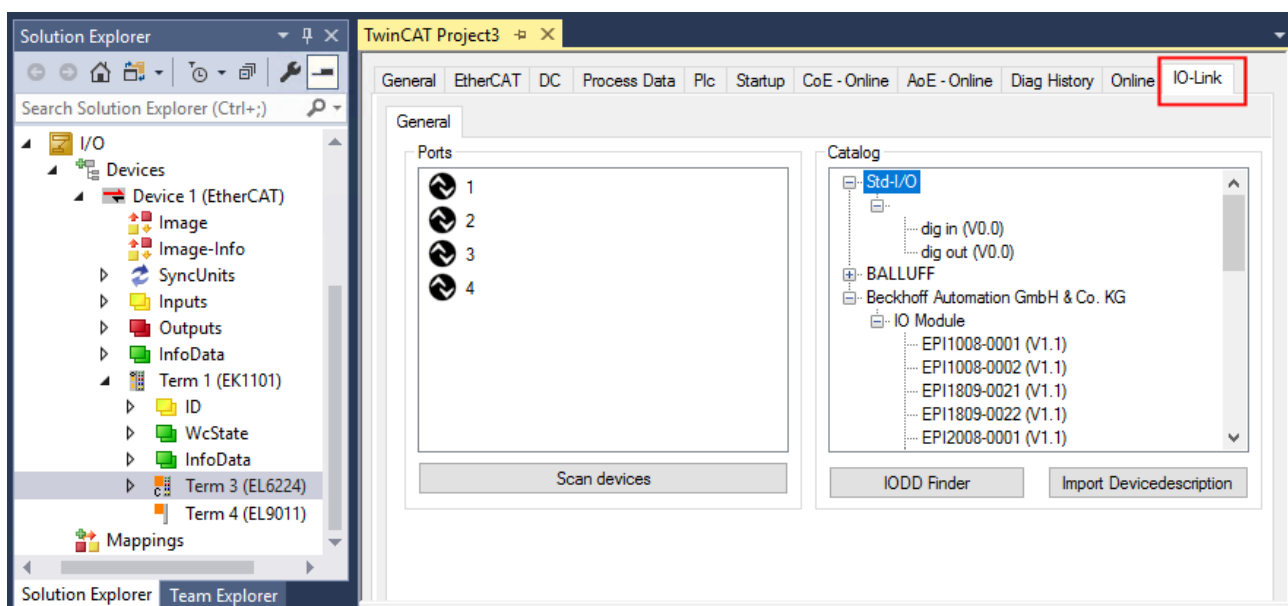


Fig. 29: IO-Link configuration tool

6.2.2 Integrating IO-Link devices

The integration of the IODD file should always be the first step, since this enables the breakdown of the individual process data of the IO-Link devices as well as the display of the parameters.

There are several ways of integrating an IO-Link device:

1. Importing the IODD file (offline and online) via
 - ⇒ button Import Devicedescription [► 60] (A) or
 - ⇒ button IODD Finder [► 60] (B)
2. Select the device in the "Catalog" field and assign it to a port [► 62].
3. Automatic scanning of the IO-Link ports (online) via
 - ⇒ button Scan devices [► 63] (C)
4. Manual insertion (offline and online) via
 - ⇒ menu Create Device [► 67] (D)

i

Application note

- If the IODD is not available, the IO-Link device should be integrated online by scanning.
- Manual integration of the IO-Link devices via "Create Device" should only be carried out if the IODD of the vendor and the IO-Link device are not available at the time of project creation.

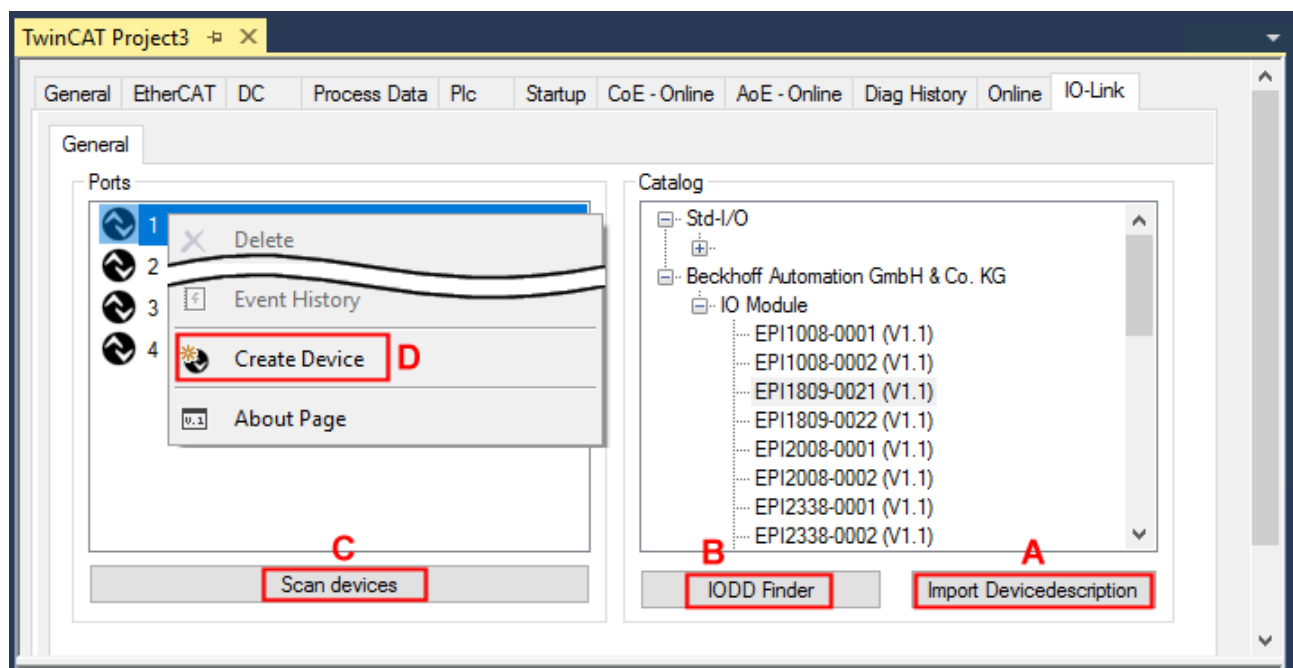


Fig. 30: Creating IO-Link devices

6.2.2.1 1. Importing the device description IODD

Importing the device description simplifies the integration of the IO-Link devices. The individual process data are broken down, enabling simple parameterization of the sensor. The IODD only needs to be imported during the initial commissioning of a new IO-Link device. The import is port-independent. Proceed as follows to import the IODD:

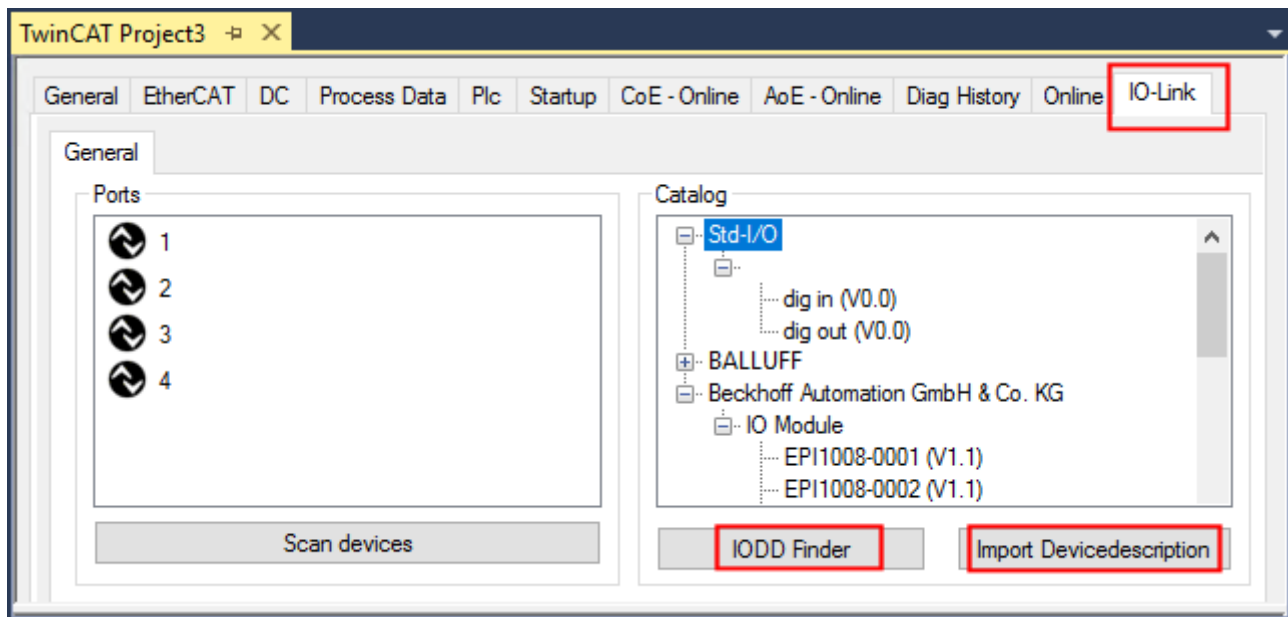


Fig. 31: Import of the IODD device description via "IODD Finder" or "Import Devicedescription"

Button "Import Devicedescription"

1. Press the "Import Devicedescription" button in the "IO-Link" tab
 2. Select the .xml file of the desired sensor.
 3. After pressing the Open button, the imported files are stored in the following folder:
 - for TwinCAT 2.x: \TwinCAT\IO\IOLink
 - for TwinCAT 3.x: \TwinCAT\3.X\Config\IO\IOLink.
- ⇒ The imported device descriptions are listed in a tree structure in the "Catalog" field, sorted by vendor.



No manual copying of the XML files

Do not copy the files directly into the folder; read them in via *Import Devicedescription* instead! Important checks will otherwise be bypassed!

Button "IODD Finder"

1. Press the "IODD Finder" button in the "IO-Link" tab
2. Searching for the desired IO-Link sensor/device by entering them in the search mask; see the figure below (1)
3. Selecting the desired IO-Link sensor/device. Move the mouse pointer over the figure of the desired IO-Link sensor/device. A blue download icon appears, see the following figure (2).

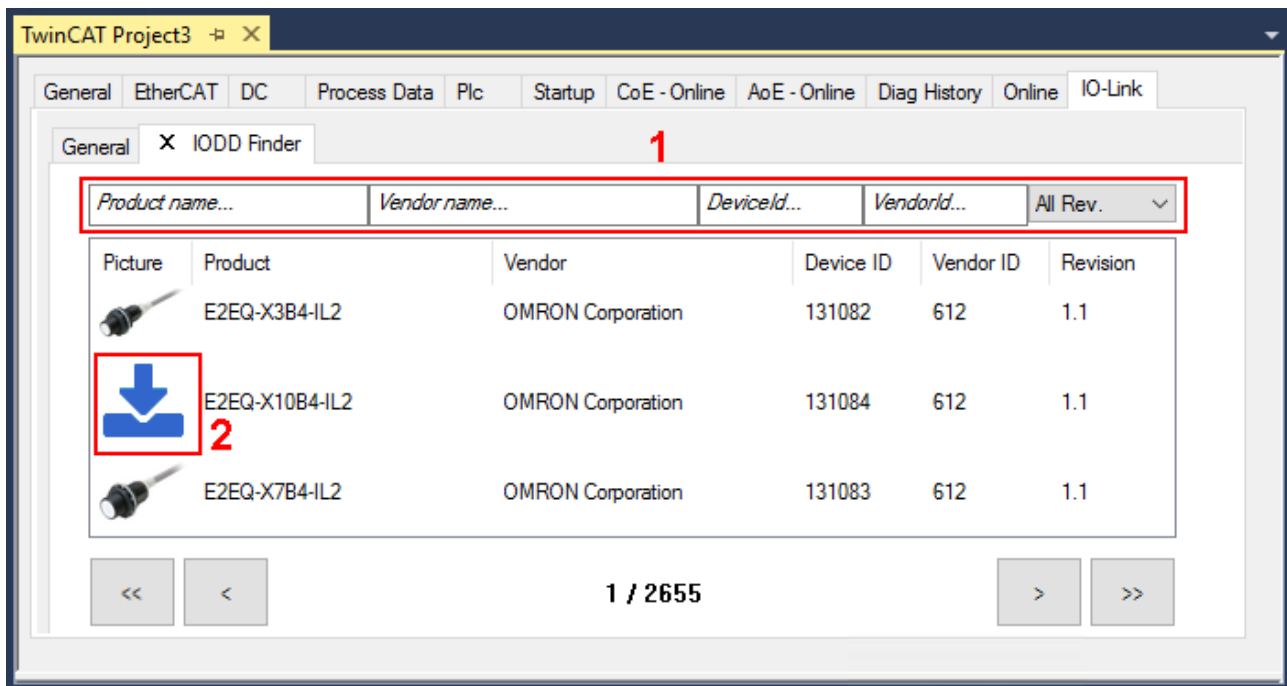


Fig. 32: IODD Finder, selection and import of the .xml-file

4. After clicking the download symbol, the .xml file of the selected IO-Link sensor/device is imported and stored in the following folder:
 - for TwinCAT 2.x: \TwinCAT\IO\IOLink
 - for TwinCAT 3.x: \TwinCAT\3.X\Config\IO\IOLink
5. When moving the mouse pointer over the IO-Link sensor/device, a green icon now indicates (see the following figure (3)) that the .xml file already exists.

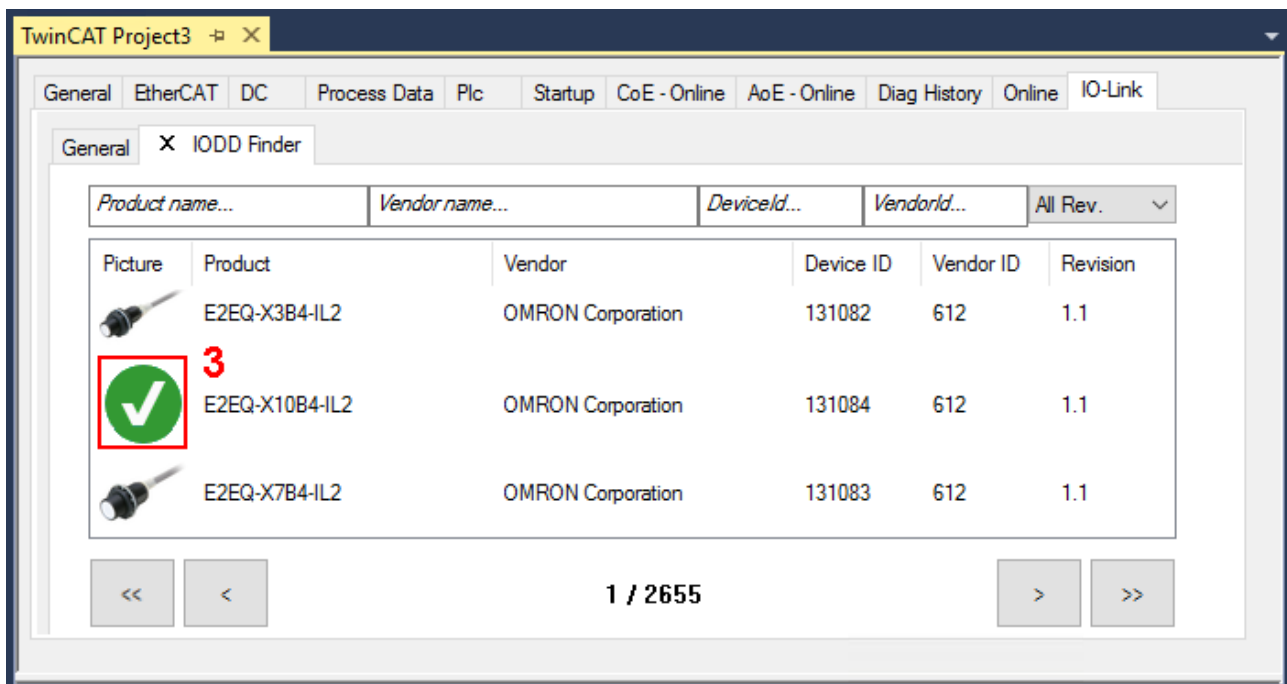


Fig. 33: IODD Finder, display of an already imported device description

- ⇒ The imported device descriptions are listed in a tree structure in the “Catalog” field of the IO-Link tab, sorted by vendor.

6.2.2.2 2. Assigning IO-Link device to port n

Online configuration

✓ Requirement: The IO-Link device is connected.

1. Press the button Scan devices (see chapter [Automatic scanning](#) [► 63])

⇒ The device is automatically detected and created with the corresponding parameters. If several devices are stored in the IODD file, the first entry is always selected here. Grouping in the IODD is usually carried out by the vendor if the process data are the same and there are only mechanical differences (e.g. other material).

Offline configuration

The *Catalog* field shows the IO-Link device catalog, which lists the already imported device descriptions in a tree structure, sorted by vendor.

1. Select the desired IO-Link device from the *Catalog* field
 - via drag and drop or
 - by right-clicking on the product with "Add to Port n".

Activating the configuration

2. [Activate the IO link configuration](#) [► 69], so that changes become effective.

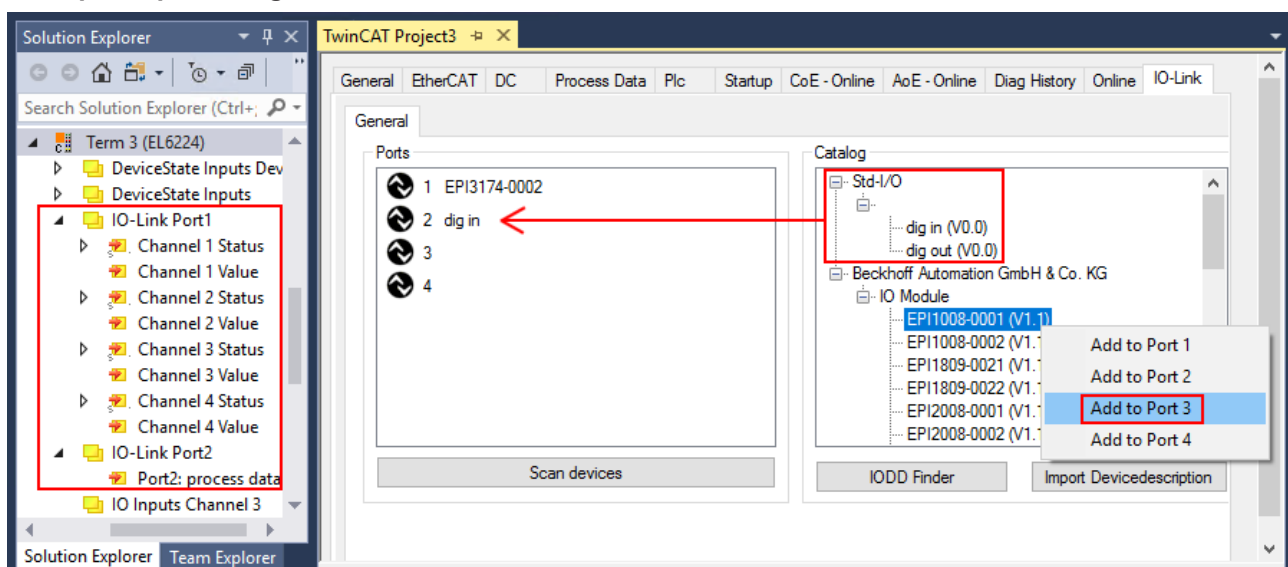
⇒ The IO-Link devices are displayed, and the process data are created. If an error is found when integrating the IO-Link device, e.g. wrong VendorID or no device connected, then this is indicated via the status of the port (object state Ch.n 0xF100:0n).

Configuration of the IO-Link ports as digital in- or output

IO-Link ports can also be configured as digital inputs or digital outputs. This allows digital sensors and actuators having no IO-Link functionality to be connected to IO-Link ports.

1. Expand the "Std-I/O" tree node in the "Catalog" field.
 - ⇒ The operating modes "dig in" and "dig out" appear.
2. Configure the desired port. There are two ways to do this:
 - Drag-and-drop: pull "dig in" or "dig out" onto the port in the "Ports" field or
 - Right-click on "dig in" or "dig out" and click on "Add to Port n".

Example of port assignment on the IO link master EL6224



Port1:
EPI3174-0002 is assigned

Port2:
is configured as digital input
Process data of Port1 and Port2 are displayed in the Solution Explorer.

Port3:
EPI1008-0001 will be assigned

6.2.2.3 3. Automatic scanning of the IO-Link ports

This part of the documentation describes the configuration of the physically available IO-Link devices in TwinCAT.

During automatic scanning of the IO-Link ports, the steps “WakeUp pulse”, “Baud rate setting”, “Reading of the communication parameters”, plus “Parameter server” and “Cyclic data exchange”, if applicable, are performed, see [Establishing the IO-Link communication](#) [► 23]. The corresponding IO-Link device must be connected to the IO-Link port for this.

The connected devices are automatically detected, configured and a search is performed for the associated IODD.

Finding connected IO-Link devices

✓ Requirement: the master and the devices are cabled and supplied with voltage.

1. Click on the “Scan devices” button (see the following figure).

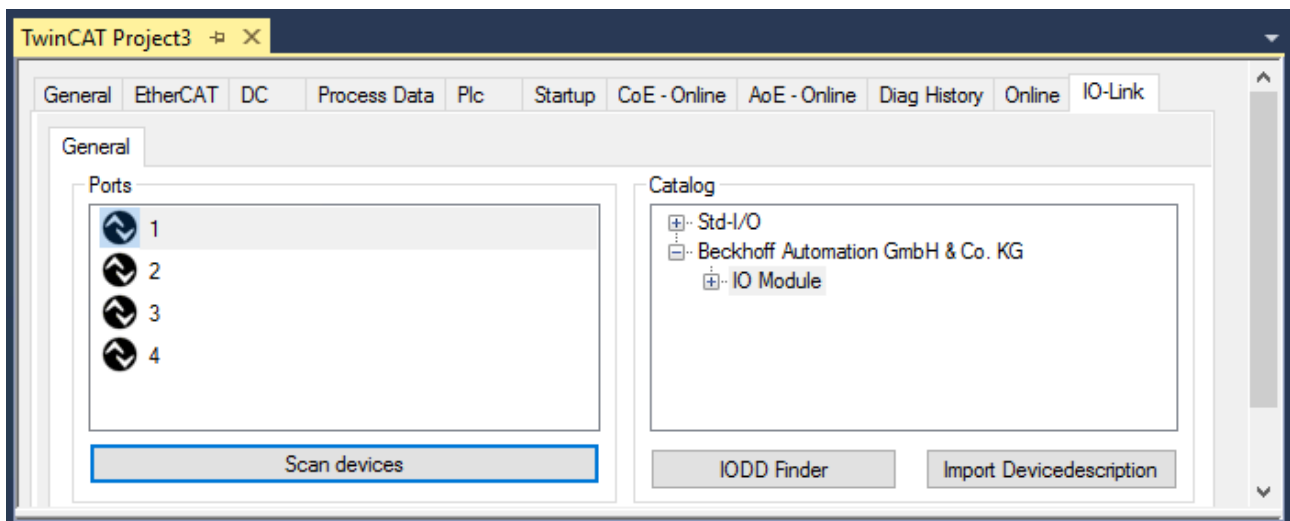


Fig. 34: Scan devices

- ⇒ The connected IO-Link devices can be found.
- ⇒ The information window lists the connected device for each of the four ports. Only port2 of the master is assigned an IO-Link device.
- ⇒ Confirm with the OK button.

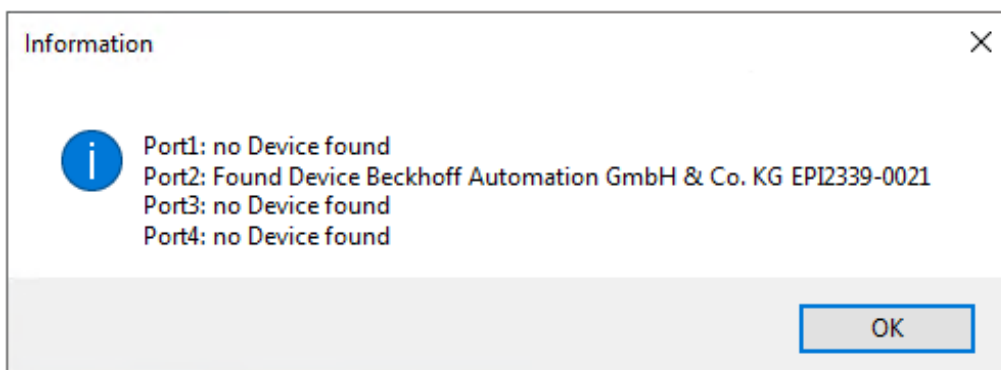


Fig. 35: Information “Scan devices”

1. To be able to work with the devices, the button “Reload Devices” must be clicked. 

The IO-Link devices are now entered in the *General* display. The Port2 “Details” field displays information about the connected device. Additionally the tabs Settings [► 65] and Parameter [► 66] can be opened.

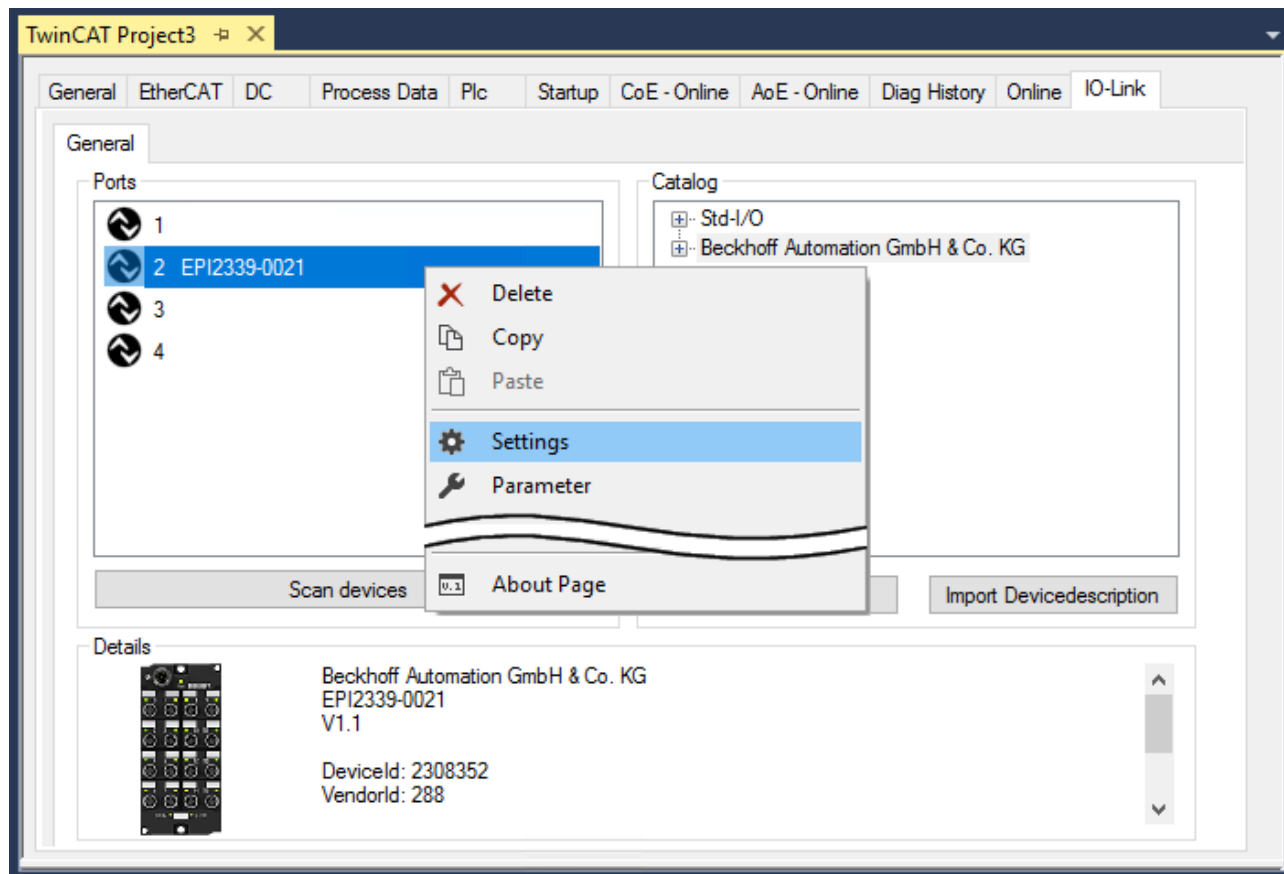


Fig. 36: Device at Port2, Display “Details”, open tabs “Settings” and “Parameter”

Show settings of the device

1. Right-click on port2, to display more details in dialog "Settings".
2. If necessary, change the settings as described in chapter [Settings of the IO-Link devices \[► 70\]](#).

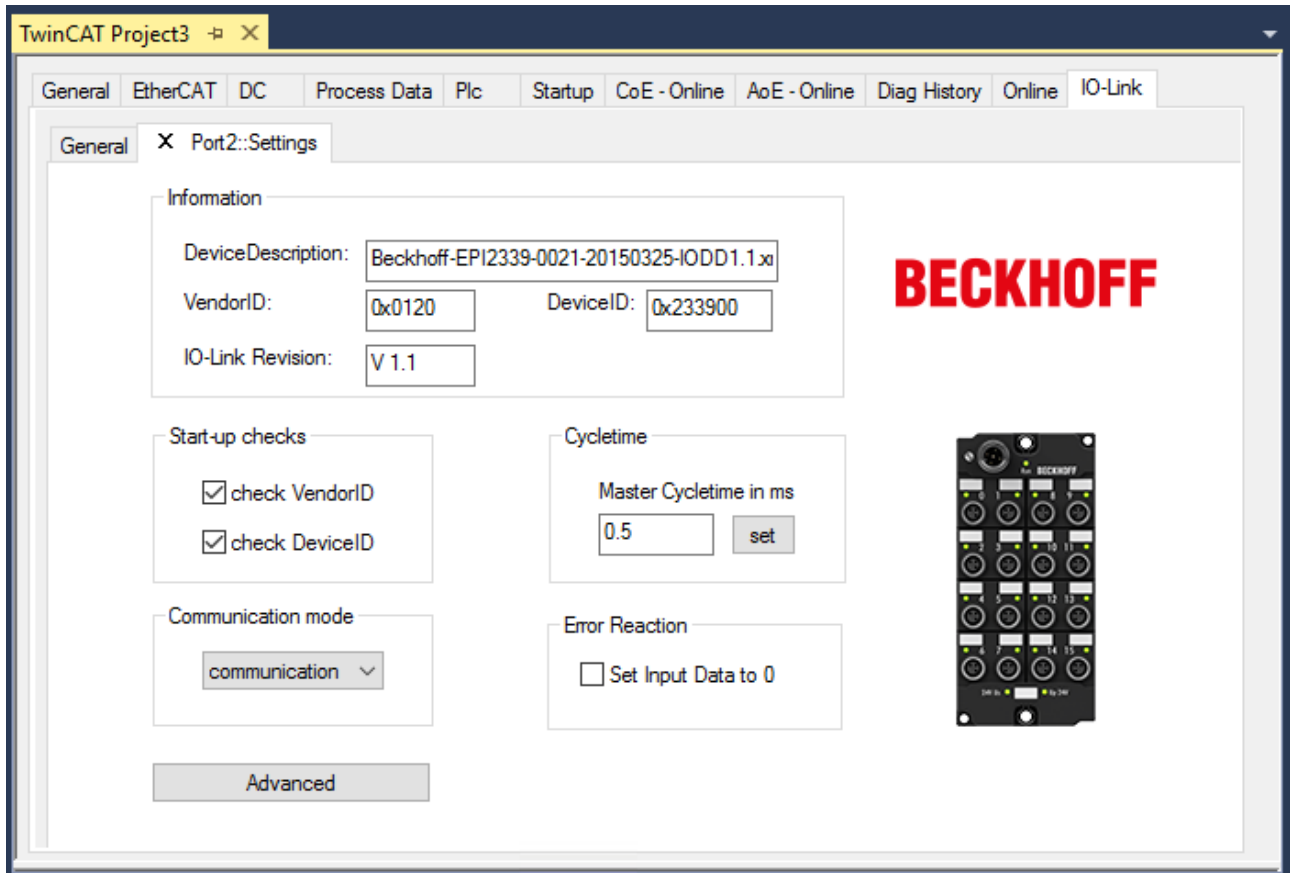


Fig. 37: Settings of the device assigned to port2

Show parameters of the device

1. Open the Parameter tab via
 - double-click on Port2 or
 - right-click on Port2 and select "Parameter" in the menu.
 ⇒ The Parameters of of the respective IO link device are listed.
2. Parameterize the device as described in chapter [EPIxxxx, ERIxxxx - Setting of the IO-Link device parameters](#) [[▶ 72](#)].

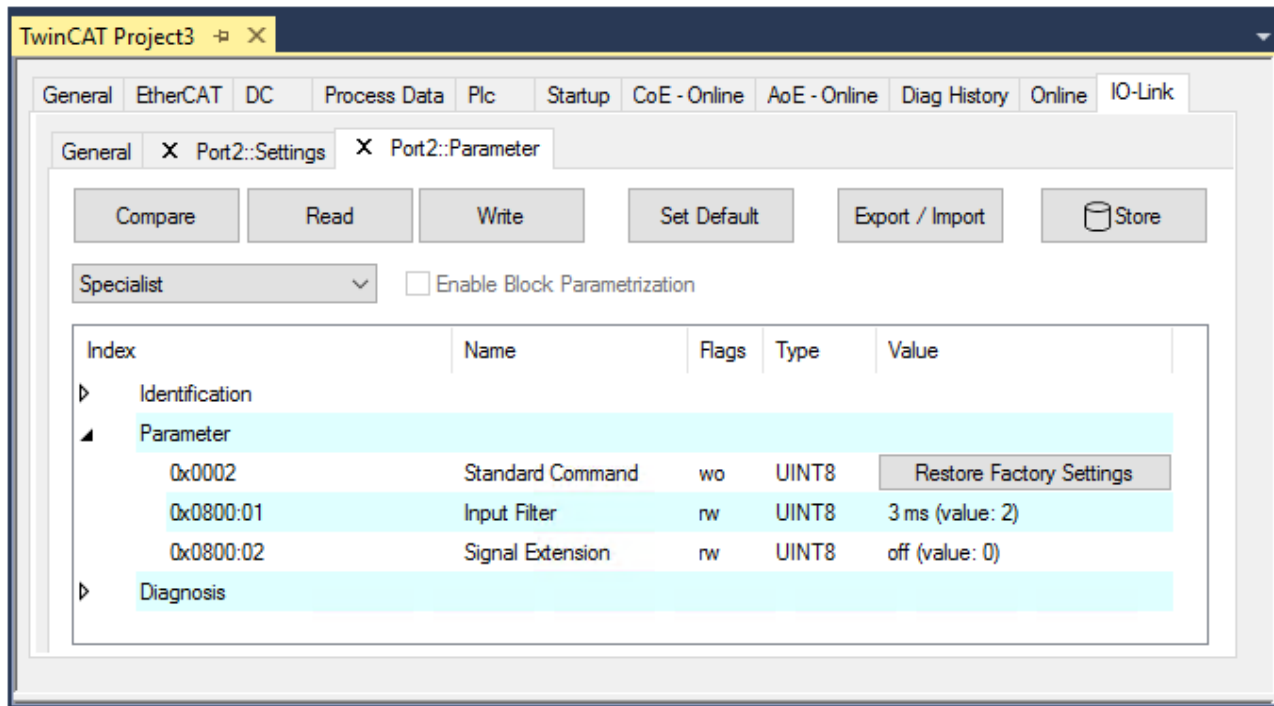


Fig. 38: Parameter of the device assigned to port2

6.2.2.4 4. Manual insertion via Create Device

This part of the documentation describes the manual configuration of the IO-Link devices in TwinCAT.

The manual insertion of the IO-Link device should only be carried out if the IODD from the vendor and the IO-Link device are not available. By saving the project, the settings for the individual ports are saved. The devices that were created are **not** stored in the "Catalog" (see the figure below (A)). To insert the IO-Link devices manually via "Create Device", proceed as follows:

1. The IODD of the IO link device is already available:
Select the respective device from the "Catalog" field sorted by manufacturer (see following figure (A)).
2. No IODD is available:
Add the device can be manually via "Create Device". These data are **not** saved in the "Catalog" field and must be manually entered for each port.
3. Right-click on the port to open the context menu (see the figure below (B)) and select "Create Device".
4. In the "Create Device" dialog an IO-Link device with the basic communication parameters can be created. The mandatory fields here are: For Vendor ID, Device ID and process data length see the figure below (C). The values VendorID and DeviceID can be entered both in hexadecimal notation (input format: 0xnnnn) and as decimal numbers (nnnn).
The communication parameters to be entered can be found in the information provided by the device vendor.
5. If the IO-Link device version is 1.1, then the parameter server is activated by the selection of the check box "Revision V1.1" (see following figure (D)).
6. Activate the IO link configuration [[▶ 69](#)], so that changes become effective.

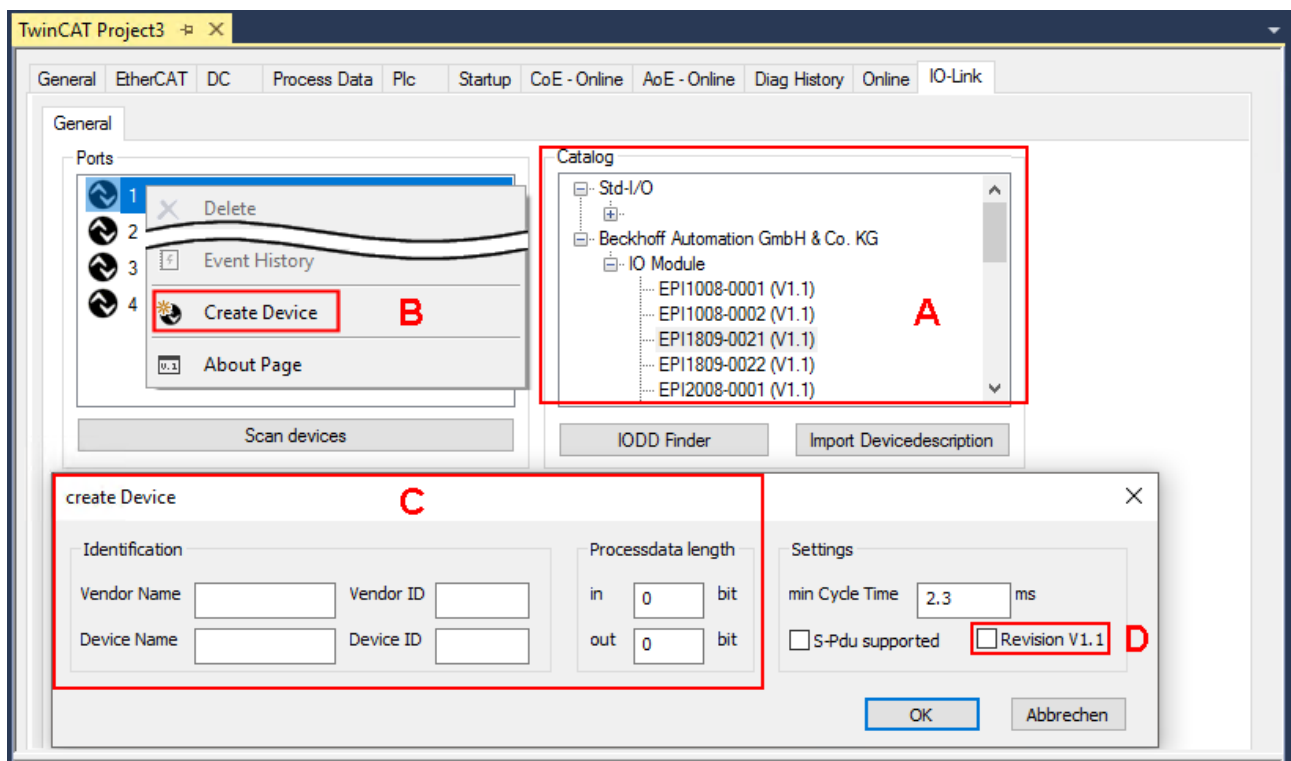


Fig. 39: Manual creation of an IO-Link device via the "Create Device" dialog (C)



Reading the IODD

Even when manually creating and scanning, the IODD should always be read in as well in order to display further sensor-specific information.

7. In the "Settings" tab of the IO link devices further settings can be made as described in chapter [Settings of the IO-Link devices](#) [[▶ 70](#)].

6.2.3 Removal of IO-Link devices

To remove a device that has already been inserted, proceed as follows.

1. Right-click on the port to open the context menu and select "Delete".

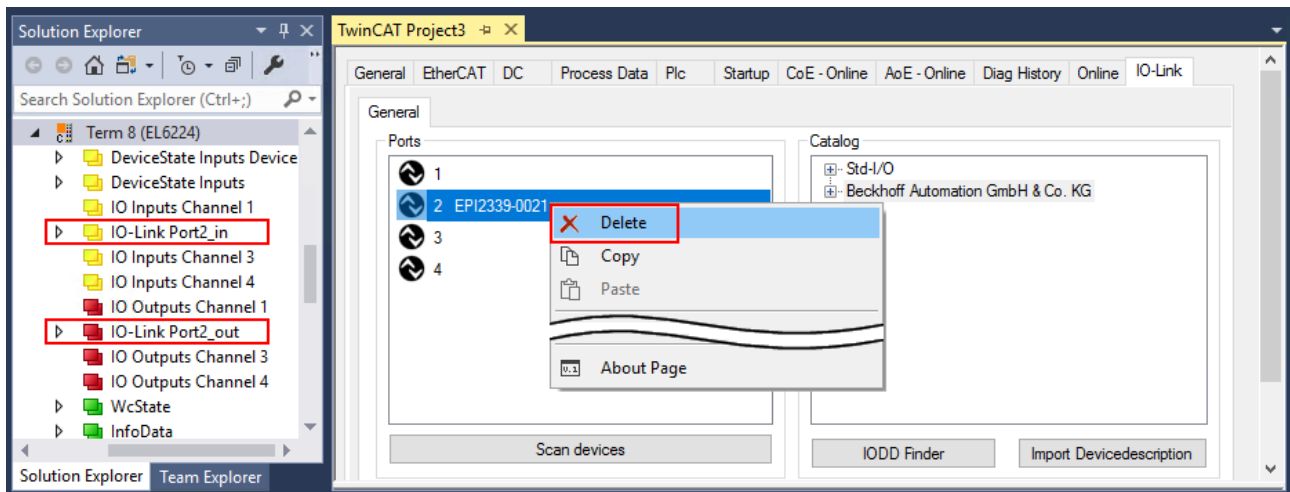


Fig. 40: Remove the device from port2

2. Activate the IO link configuration [► 69], so that changes become effective.

⇒ The already create process data are removed.

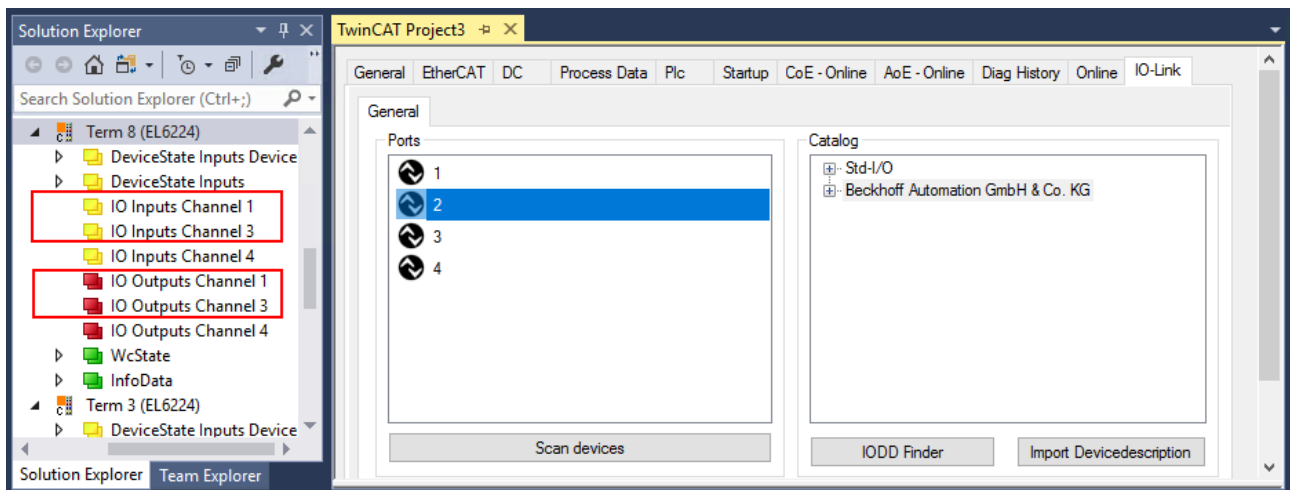


Fig. 41: The device was removed from the port2, the process data no longer displayed in the tree.

6.2.4 Activating the configuration

Changes in the IO-Link configuration tool only become effective when you activate the IO-Link configuration.

There are two ways to activate the IO-Link configuration:

- Click on the "Reload Devices" button



- Activate the TwinCAT configuration:
Click on the "Activate Configuration" button



6.3 Settings of the IO-Link devices

To find the basic settings of the devices for each port, proceed as follows.

1. right-click on the port to open the context menu and select "Settings".

⇒ A new tab "Portx:: Settings" opens where the settings described below can be made.

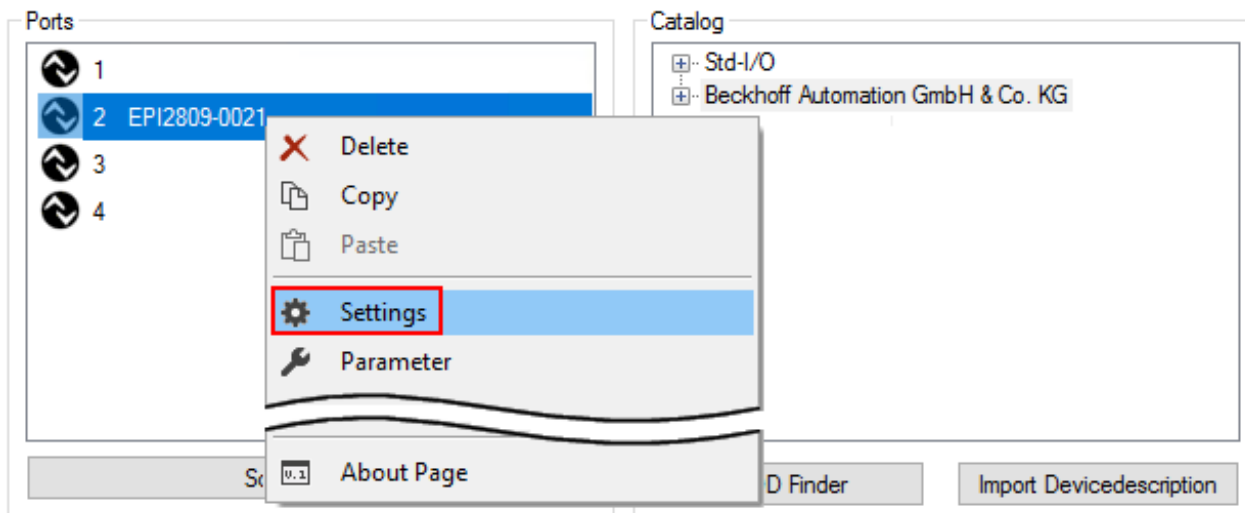


Fig. 42: Context menu - Settings

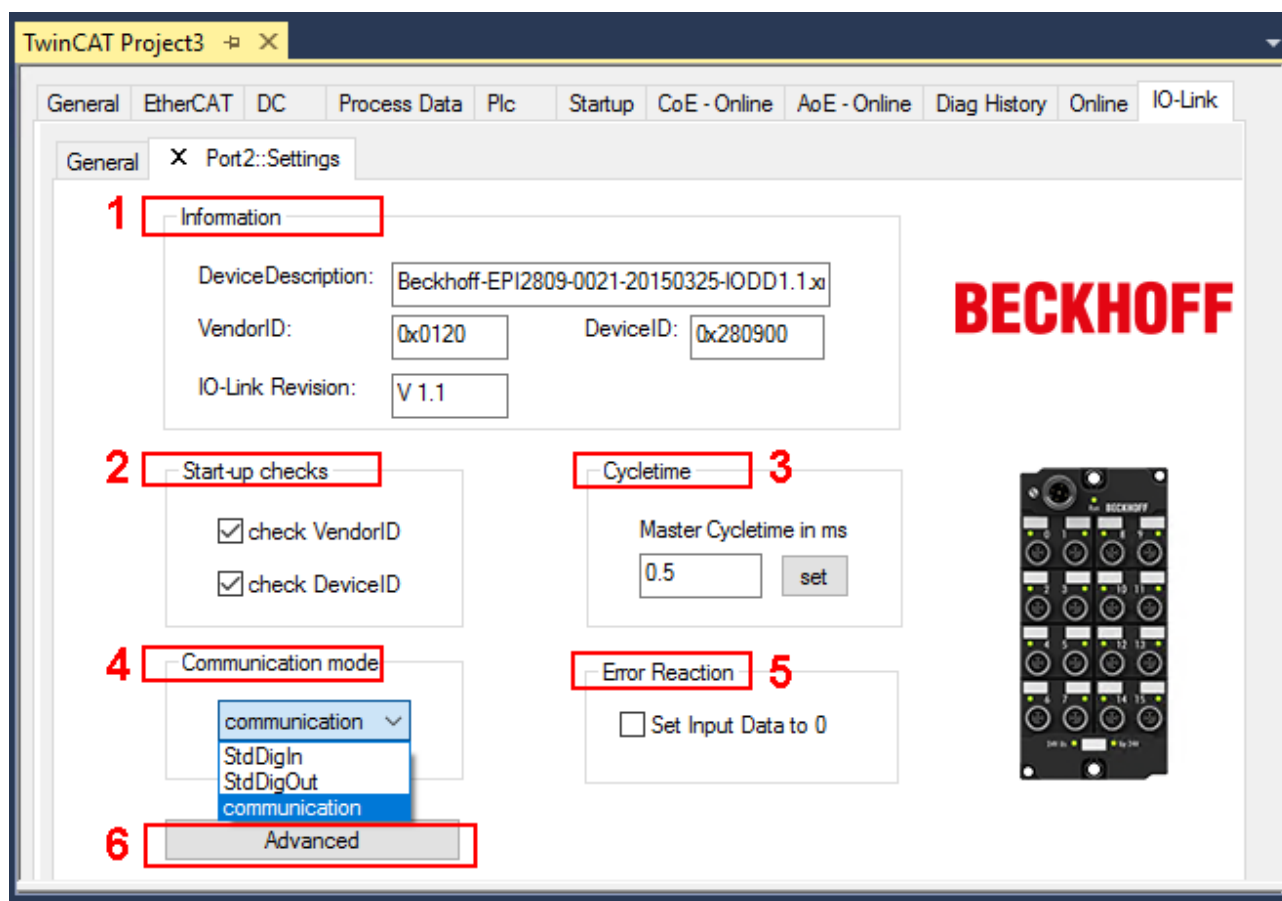


Fig. 43: Settings of the IO-Link devices

1. Information

This field is for information only; the IODD that was read in is displayed under Device Description. Furthermore, the VendorID, DeviceID and the IO-Link revision (V1.0 or V1.1) of the IO-Link devices are displayed. If the device is an IO-Link device V1.1, then the [parameter server \[► 24\]](#) function is supported.

The following settings can be made in the settings for the IO-Link devices (see figure above):

2. Start-up checks

This parameter can be used to specify that the Vendor ID and Device ID should be checked when the IO-Link device starts up.

⇒ This avoids errors when exchanging IO-Link devices.

3. CycleTime

Specifies the cycle time for the IO-Link master

4. Communication mode

Selection of the mode in which the IO-Link port is to be operated.

⇒ "Communication": Default mode for IO-Link devices

⇒ "StdDigIn / StdDigOut": Mode for non-IO-Link devices, automatically selected if the port is configured as a [digital input or output \[► 62\]](#).

5. Error Reaction

If the "Set Input Data to 0" field is activated:

⇒ input data are set to 0 in case of error

⇒ Status display: "Error"

6. Button "Advanced"

7. Data Storage

Pay attention to the sensor version:

⇒ V1.0 -> data storage is not supported

⇒ V1.1 -> data are stored in the parameter server (preset)

8. Process Data Format

Adaptation of the process data format

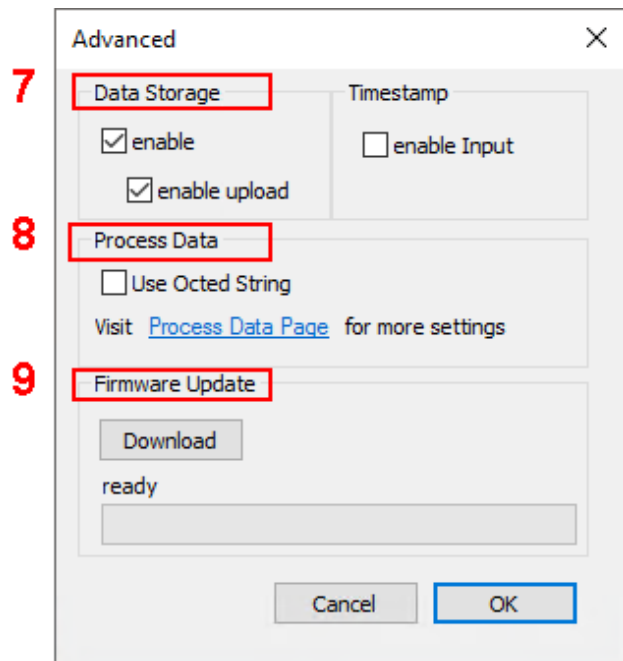
If Field "Use Octet String" is selected

⇒ complex data types (process data) are created as octet strings.

Advantage: simple further processing in the PLC

9. Firmware Update of the Beckhoff IO-Link devices

For a firmware update use the "Download" button. Observe the description in the documentation of EPIxxx boxes in chapter [Firmware Update des IO-Link Devices](#).



6.4 EPIxxxx, ERIxxxx - Setting of the IO-Link device parameters

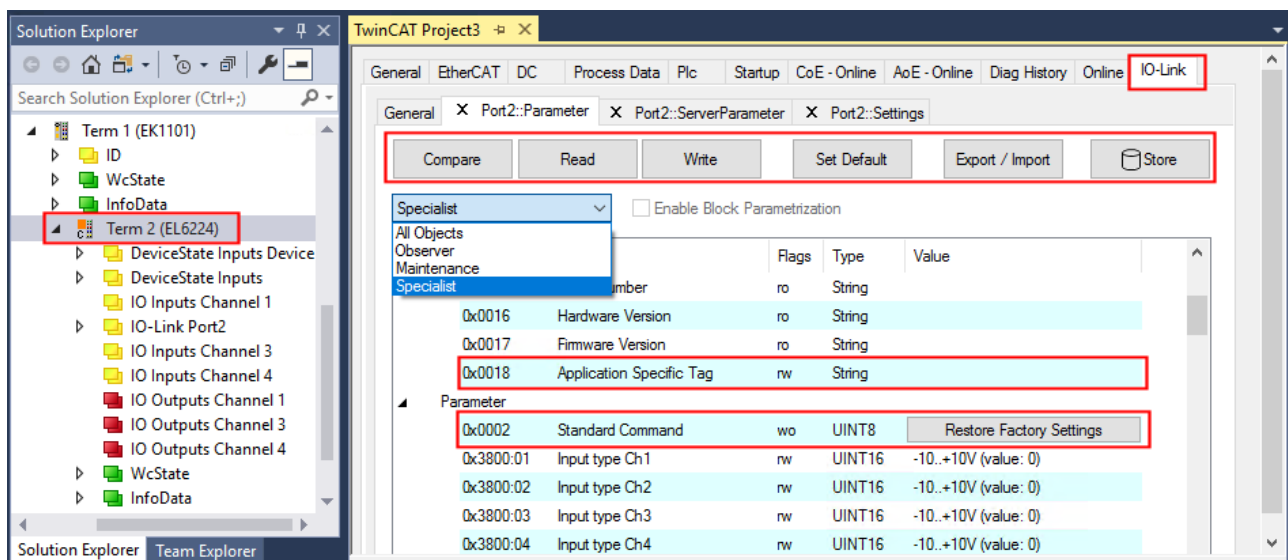
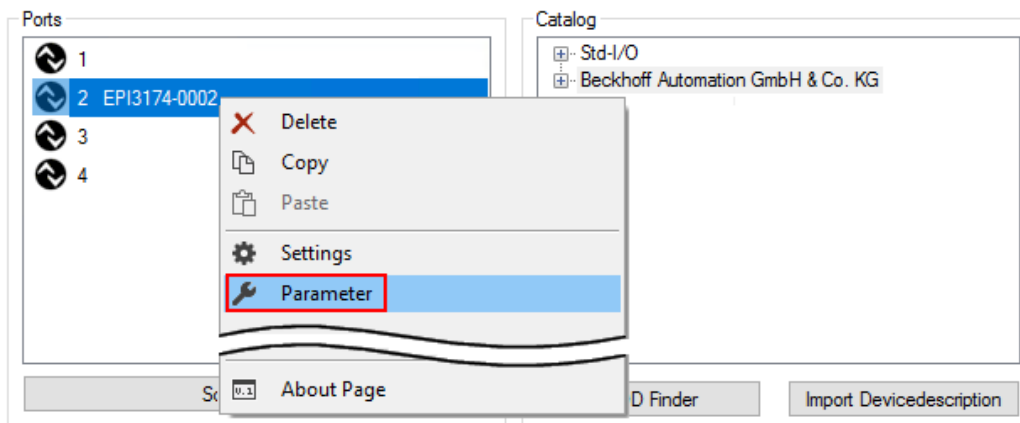
This chapter explains how to read out and set the IO-Link device parameters.

The number and type of the objects shown on the “Parameters” tab vary according to the type of sensor. The default settings as stored in the IODD can initially be seen.

To open the “Parameter” tab

1. Click the IO-Link master in the TwinCAT tree structure.
2. Click the “IO-Link” tab.
3. Select the port to which the IO-Link device is connected,
4. Double-click or by right-click to the port and select “Parameter”.

⇒ The “Parameter” tab is opened.



The device parameters are listed in the tab. The buttons [Compare](#) [► 73], [Read](#), [Write](#) [► 75], [Set Default](#) [► 76], [Export/Import](#) [► 77] and [Store](#) [► 78] are located at the top of the tab. The “Read”, “Write” and “Store” buttons are used to read out the parameters stored in the IO-Link device, load them and store them in the parameter server of the master.

Different user roles can be selected from the drop-down menu. The default user role is “Specialist”. The parameters are displayed in different representations and scopes.

Restarting the IO link device or restoring of the application parameters is possible via the parameter [Standard Command](#) [► 81].

Application specific information can be specified in parameter (0x0018) [Application Specific Tag](#) [► 82].

“Compare” button

1. Press the “Compare” button.
 - ⇒ the parameter data of the configuration are compared with the parameter sets in the sensor.
 - ⇒ The result is displayed in the “Parameter” tab see following figures.

Conformity of configuration and sensor data

The match is confirmed by a green tick in front of the index. Matching values are displayed in the “Value” field (see index 0x0018 “Application Specific Tag”).

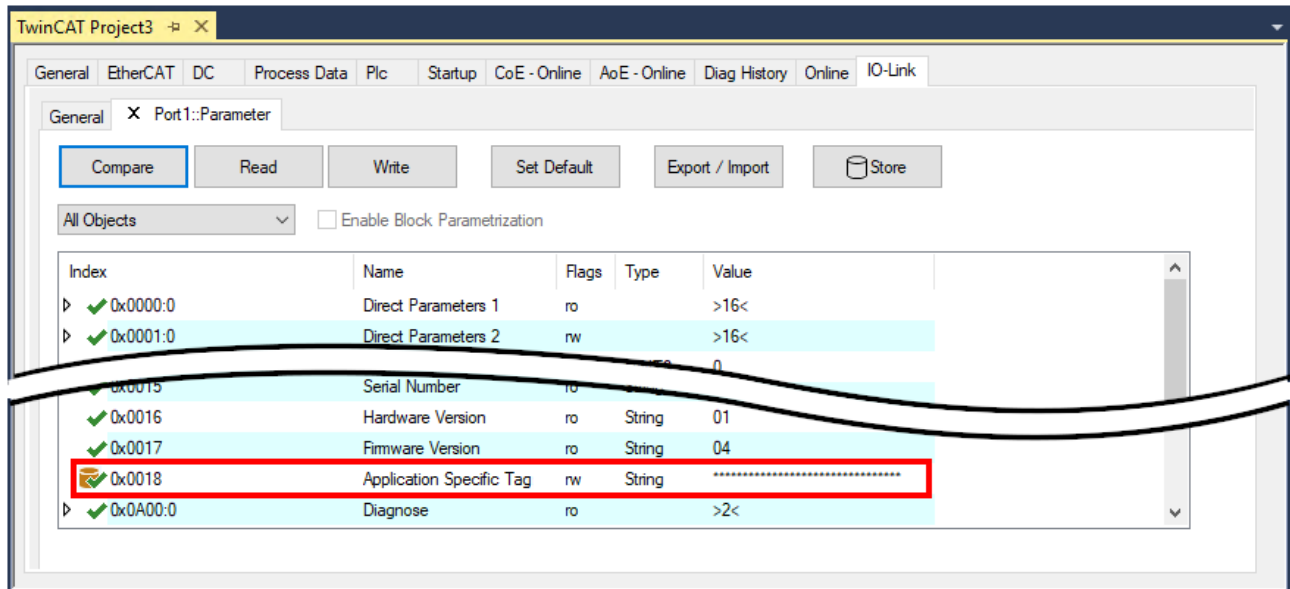


Fig. 44: Display of matching data in the “Parameter” tab

Deviations between configuration and sensor data

Deviations are indicated by a pen-symbol in front of the index. If there are different values in the “Value” field, the value “Compare” is displayed (see Index 0x0018 “Application Specific Tag”).

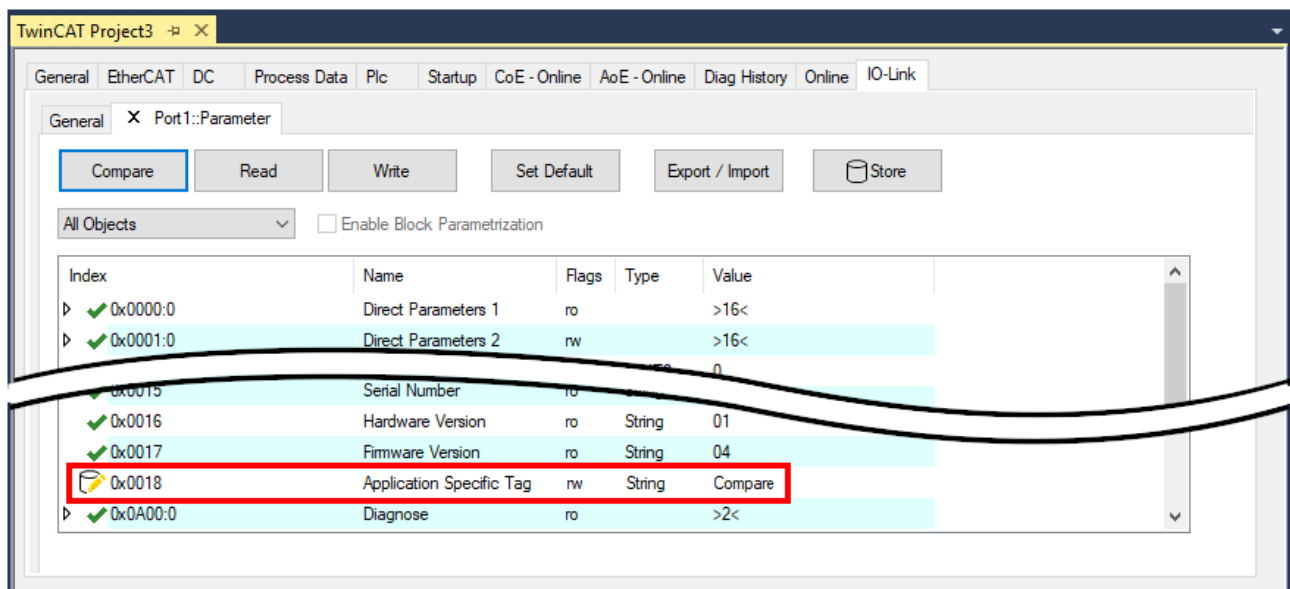


Fig. 45: Display of deviating data in the “Parameter” tab

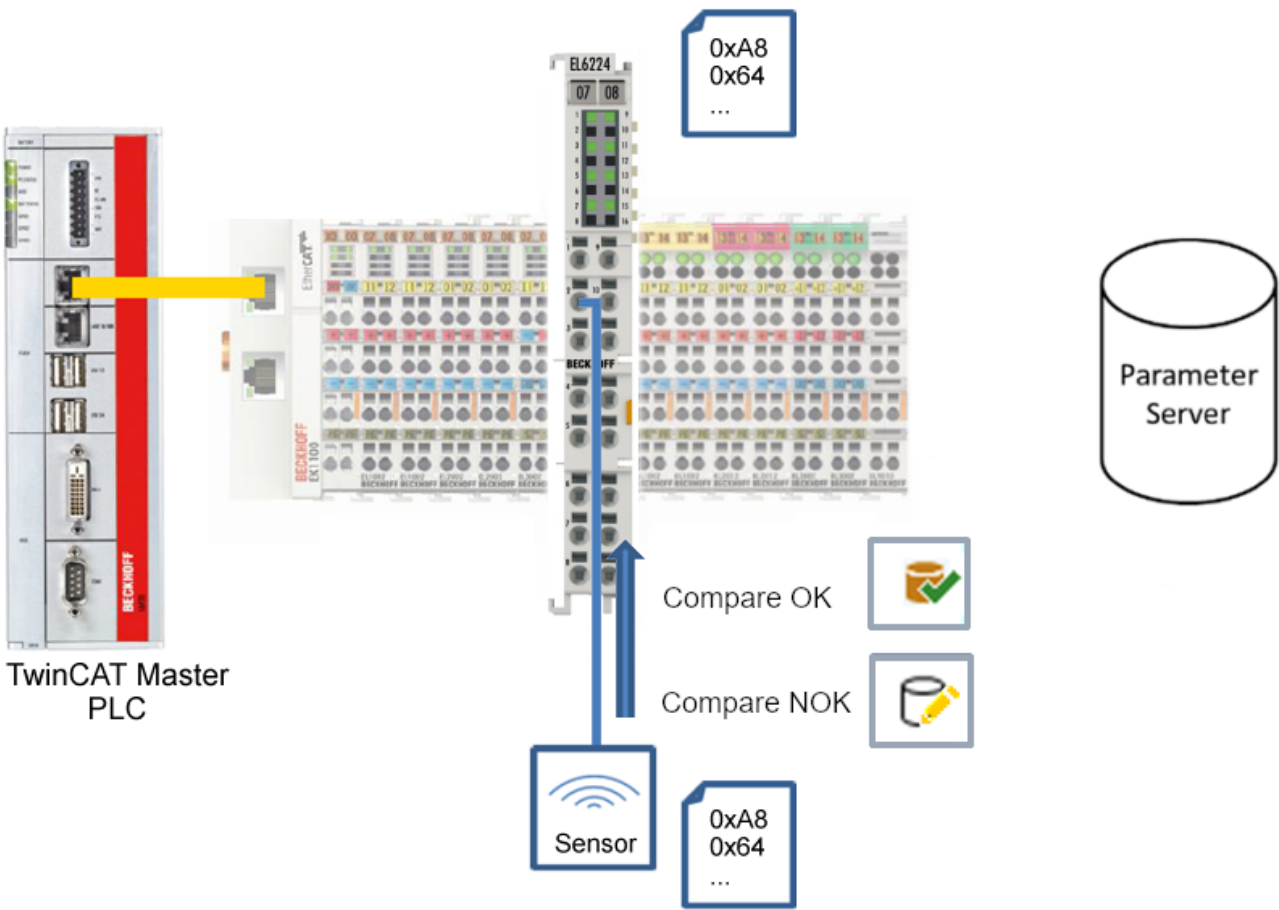


Fig. 46: Compare configuration and sensor data

“Read” button

The default values from the IODD file are always preset

1. Press the “Read” button
 - ⇒ The current parameter values of the sensor are read. The successful reading of the data is confirmed with a green tick in front of the index.

“Write” button

The default values from the IODD file are always preset

1. Enter the desired value under “Value”.
2. Press the Enter key.
 - ⇒ The values are accepted.
3. Press the “Write” button.
 - ⇒ The data is written to the device (offline configuration is possible). The successful writing process is confirmed via a storing symbol in front of the index.

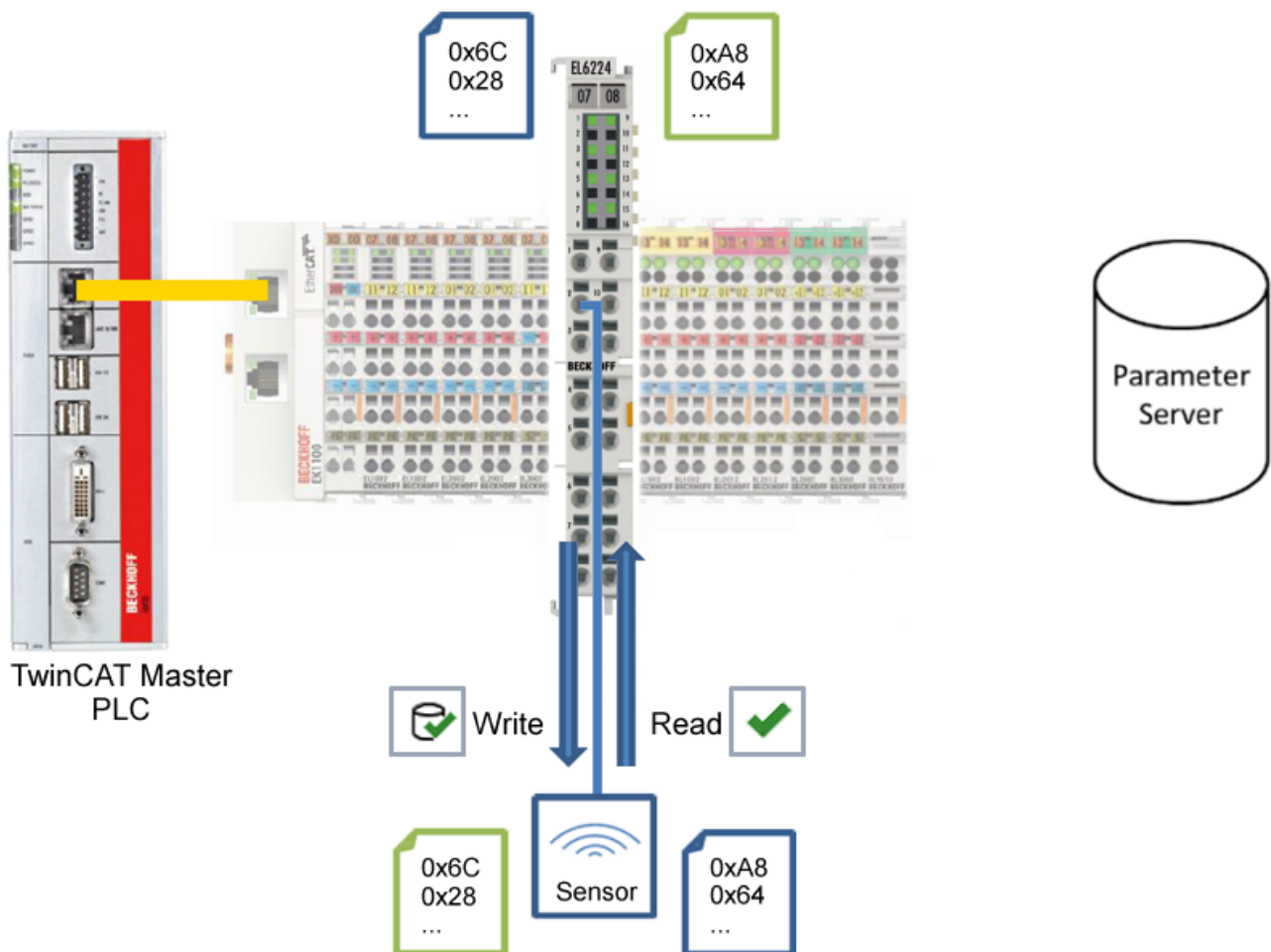


Fig. 47: Write parameter data to the sensor, read parameter data from the sensor.

“Set Default” button

1. Press the “Set Default” button.

⇒ All parameter values are set to the default settings.

i Write default-values to the sensor

Note that the default-values must also be written to the device via the “Write” button.

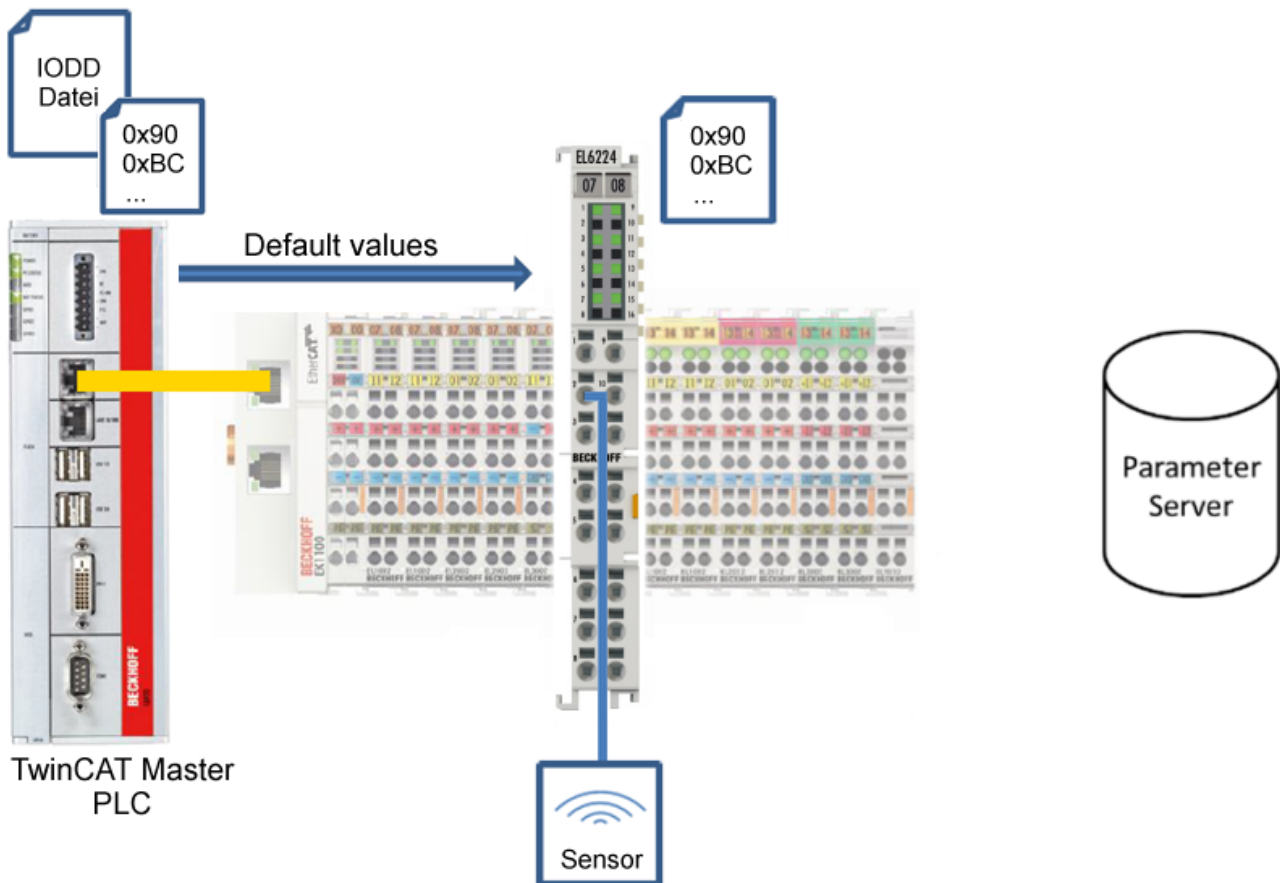


Fig. 48: Reset parameter values to default

“Export / Import” button

The set parameter values can be exported as a .vbs file and restored later via Import.

1. Press the “Export / Import” button (see the diagram below (1)).
 - ⇒ The Import / Export dialog is opened
2. Specify the path under which you want to export or import the .vbs file, see fig. (2) below and confirm with the “Open” button, see fig. (4) below.
3. In addition, the export options “Attach Store Command” and “Enable Block Parameterization” can be selected as shown in fig. (3) below.
 - ⇒ “Attach Store Command”: The parameters are loaded into the parameter server after the script has written all values.
 - ⇒ “Enable Block Parameterization”: Block parameterization is enabled. For some sensors, writing is only possible when block parameterization is enabled.
4. Press the “Export” or “Import” button
 - ⇒ The parameters are adopted from the imported file. The change of parameters is marked with a pencil symbol.
5. Write the new parameter values to the sensor via “Write” button.
 - ⇒ The data is written to the device (offline configuration is possible). The successful writing process is confirmed via a storing symbol in front of the index.

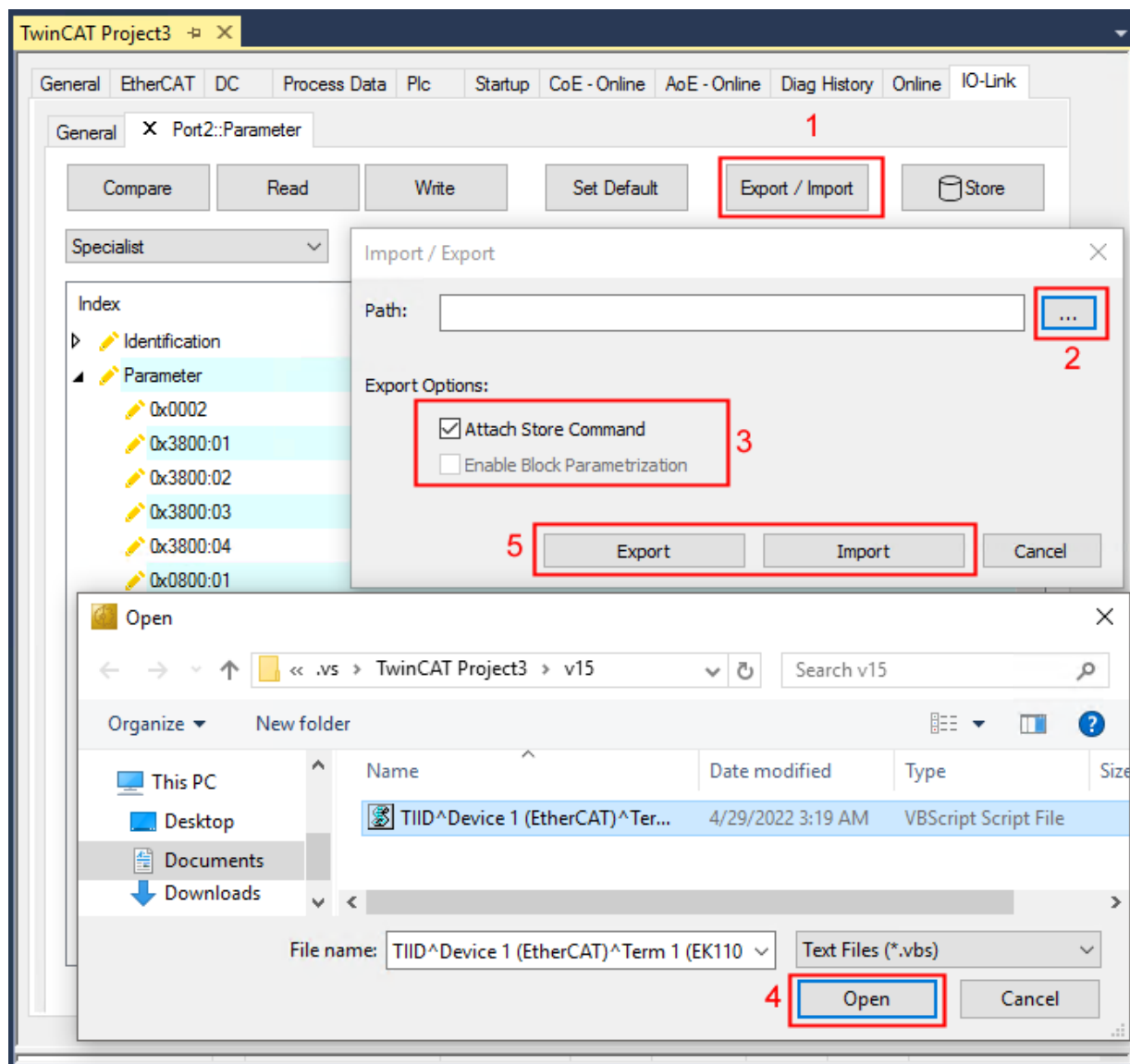


Fig. 49: Parameterization IO-Link device - Export / Import

“Store” button

1. Click “Store” (data storage):

- ⇒ The Beckhoff IO-Link master stores sensor-dependent-data, e. g. the following parameters (0x0018) “Application-Specific Tag”, (0x08n0) “Settings” and 0x3800 “Range Settings”.

The success of storing process is marked with the storing symbol.

- ⇒ If the IO-Link device is exchanged for a similar module, the device can be restored.

The stored values are displayed in the “ServerParameter” tab

1. Right-click on the device and select “Parameter Server” from the menu.

- ⇒ The stored values are displayed.

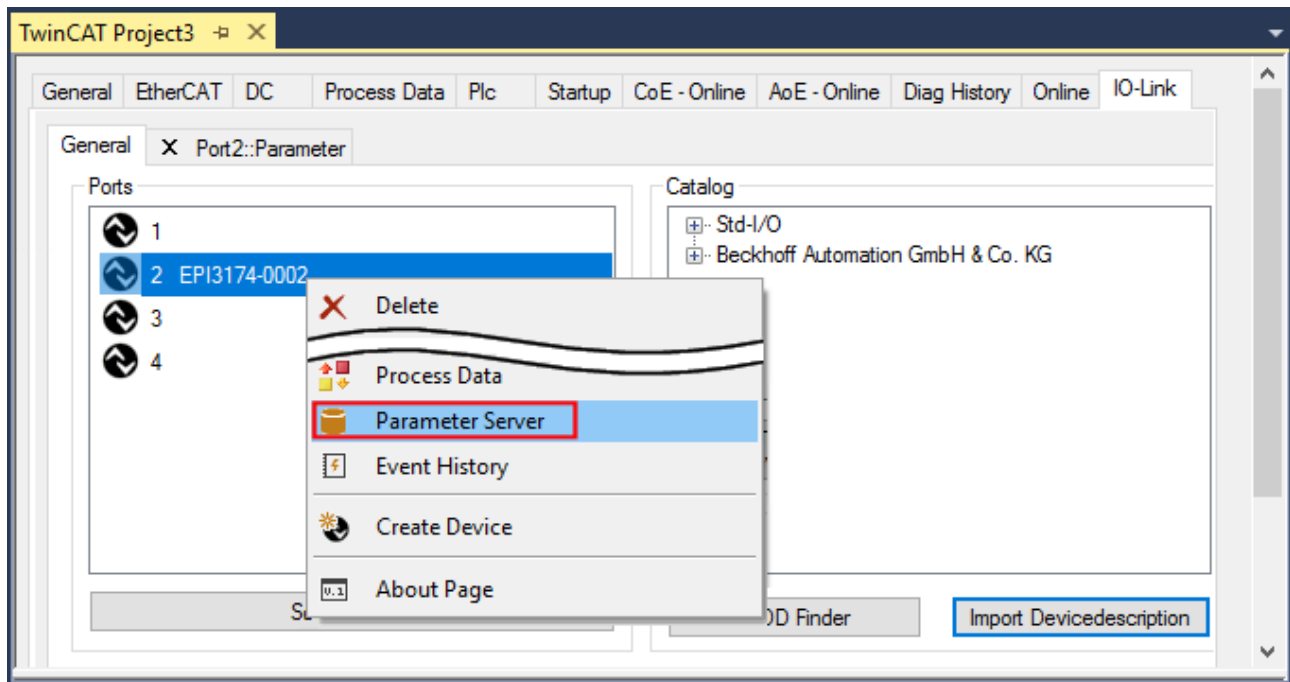


Fig. 50: Open the “ServerParameter” tab

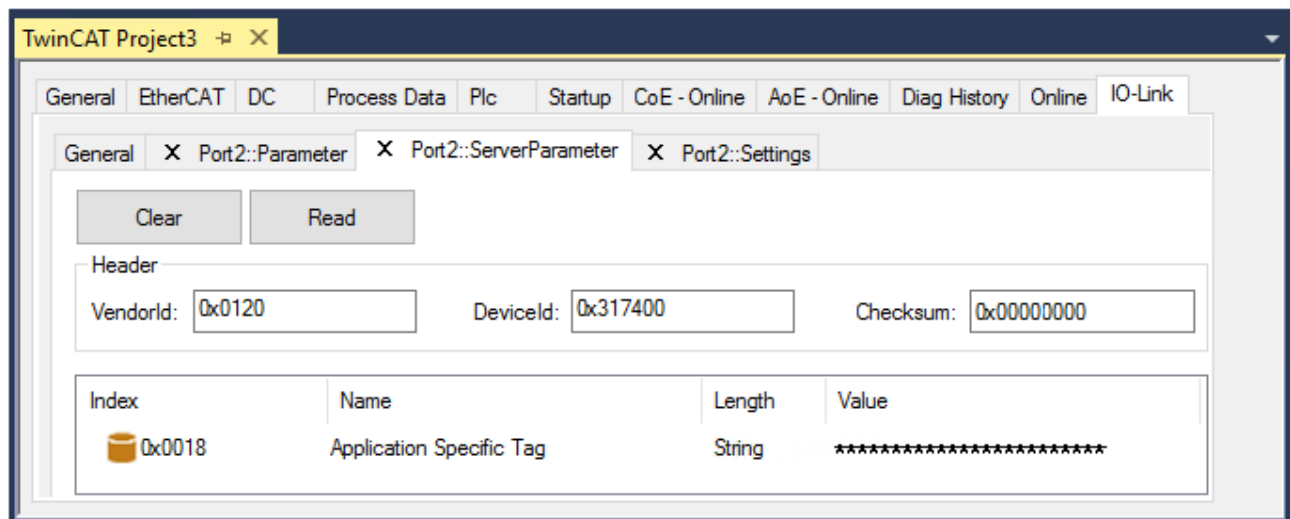


Fig. 51: „ServerParameter“ tab

Activate store button via PLC

As for CoE, the Indexgroup of an ADS command is specified as **0xF302** for the IO link data channel.

According to the IO-Link specification devices with ISDU support shall use index **0x0002** to receive the SystemCommand. The following list displays coding examples for system commands (ISDU), the complete table "Coding of SystemCommand (ISDU)" can be found in the [IO-Link specification](#).

Command (hex)	Command (dec)	Name of the command	Definition
....			
0x01	1	ParamUploadStart	Start Parameter Upload
0x02	2	ParamUploadEnd	Stop Parameter Upload
0x03	3	ParamDownloadStart	Start Parameter Download
0x04	4	ParamDownloadEnd	Stop Parameter Download
0x05	5	ParamDownloadStore	Finalize parameterization and start Data Storage
0x06	6	ParamBreak	Cancel all Param commands
....			

Use an ADS Write function block for activating the store-function via the plc. The following figure shows a sample code for activation of the store-Button (command 0x05 "ParamDownloadStore")

```

Case_Write:
  AdsWrite_EL6224( WRITE := FALSE );
  AdsWrite_EL6224.IDXGRP  := EL6224_Ch_iGrp;
  AdsWrite_EL6224.IDXOFFS := EL6224_Ch_iOffWri;
  AdsWrite_EL6224.LEN     := SIZEOF(EL6224_bywrite);
  AdsWrite_EL6224.SRCADDR := ADR(EL6224_bywrite);
  AdsWrite_EL6224( Write := TRUE );
  eSwitch1 := Case_WriBu;

EL6224_AoePortCh : UINT := 16#1001;
EL6224_Ch_iGrp   : UDINT := 16#F302;
EL6224_Ch_iOffManu : UDINT := 16#00100000;
EL6224_Ch_iOffPro  : UDINT := 16#00140000;
EL6224_Ch_iOffWri  : UDINT := 16#00020000;
EL6224_sManu       : STRING;
EL6224_sPro        : STRING;
EL6224_bywrite     : BYTE := 16#5;

```

Fig. 52: Sample code for activation of the store-function via the plc

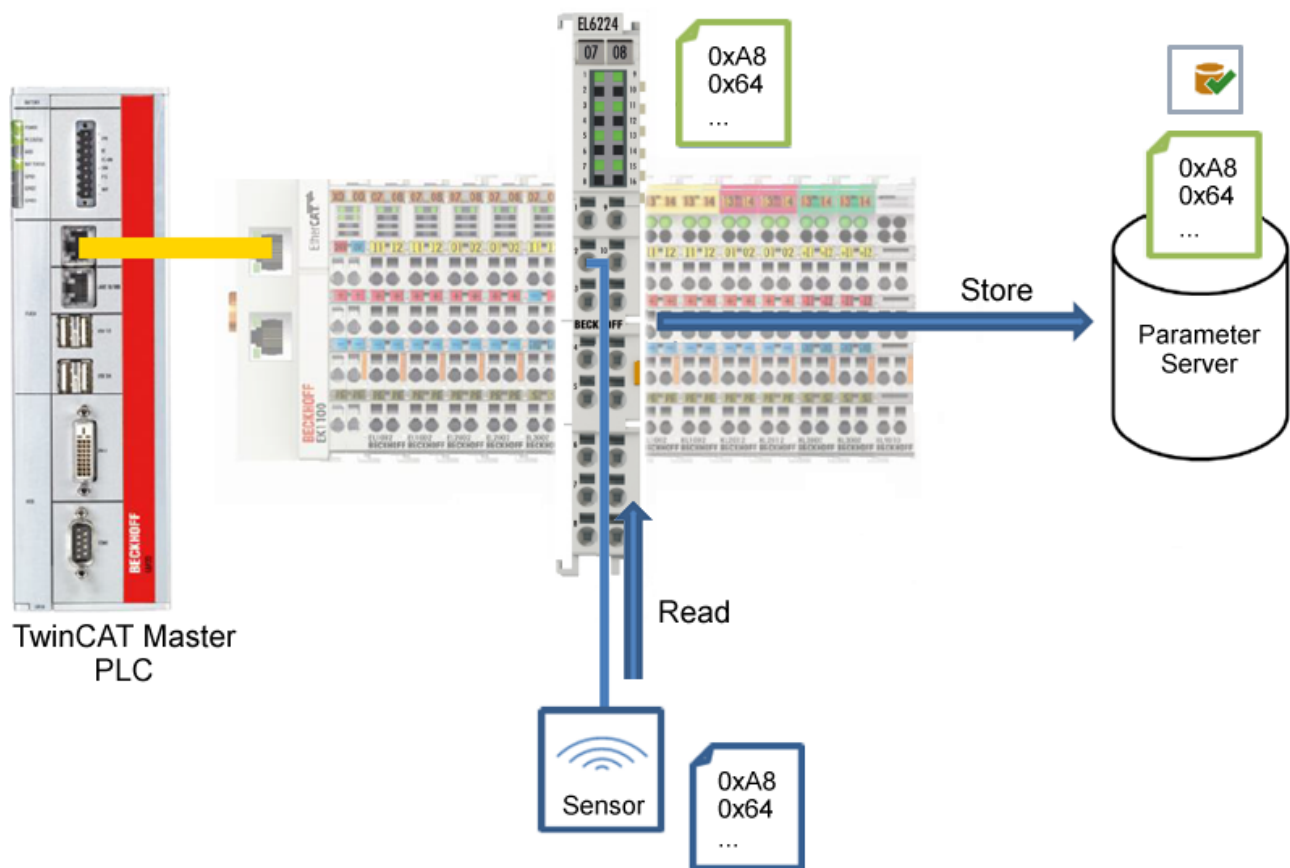


Fig. 53: Store parameters

Standard Command (Index 0x0002)

The IO-Link master writes various IO-Link-specific commands to the “Standard Command” during startup. Some of these commands are available in the TwinCAT interface (see figure below).

1. Click “Standard Command” in the parameter list of the “All Objects” user role, then double-click “Standard Command” in the right-hand field.
 2. Select the desired value from the list of different options and
 - “Device Reset”: Restarts the IO-Link device.
 - “Application Reset”: No function.
 - “Restore Factory Settings”: Restoring the application parameters, i.e. the Settings parameter (0x0800).
 3. Use the “Write” button (as described above [► 75]).
- ⇒ The data is written to the device (offline configuration is possible). The successful writing process is confirmed via a storing symbol in front of the index.

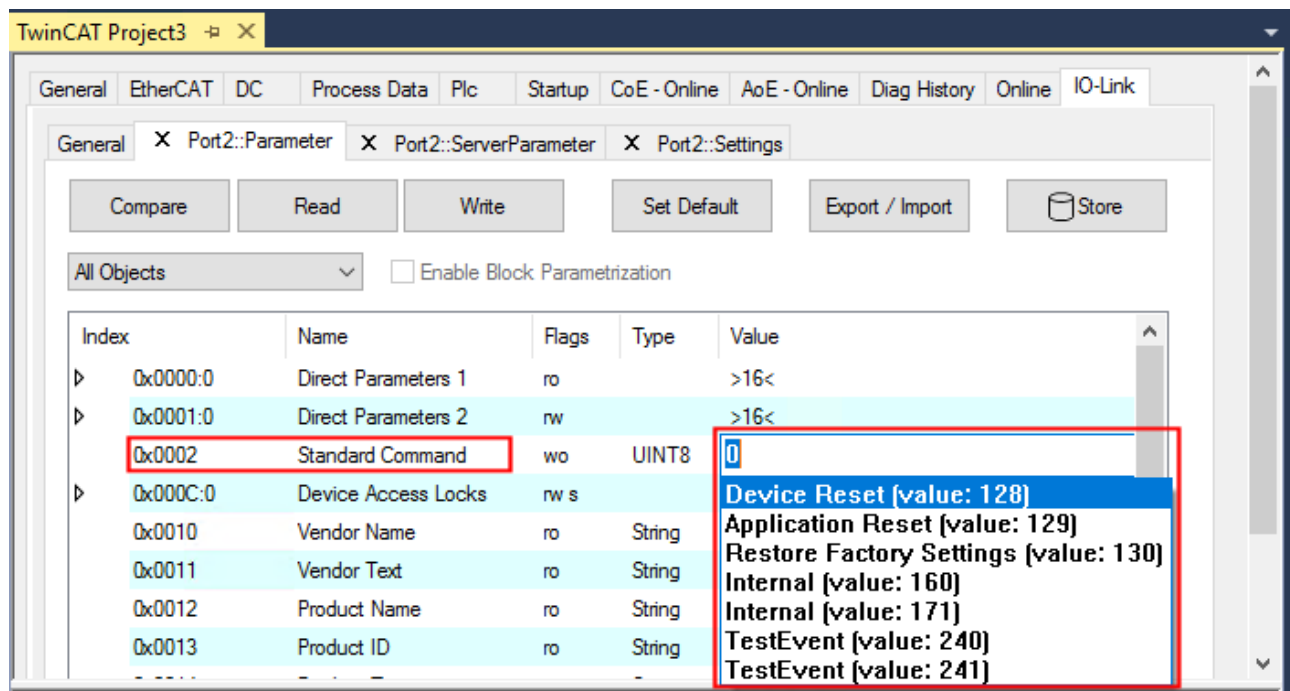


Fig. 54: Parameters IO-Link device “Standard Command”

“Application Specific Tag” (Index 0x0018)

Application-specific information can be entered and stored here.

1. Click “Application-Specific Tag” in the parameter list, then double-click “Application-Specific Tag” in the right-hand field.
2. Enter application-specific information and confirm with the Enter key.
3. Use the Write [► 75] button and the Store [► 78] button, if required (as described above).

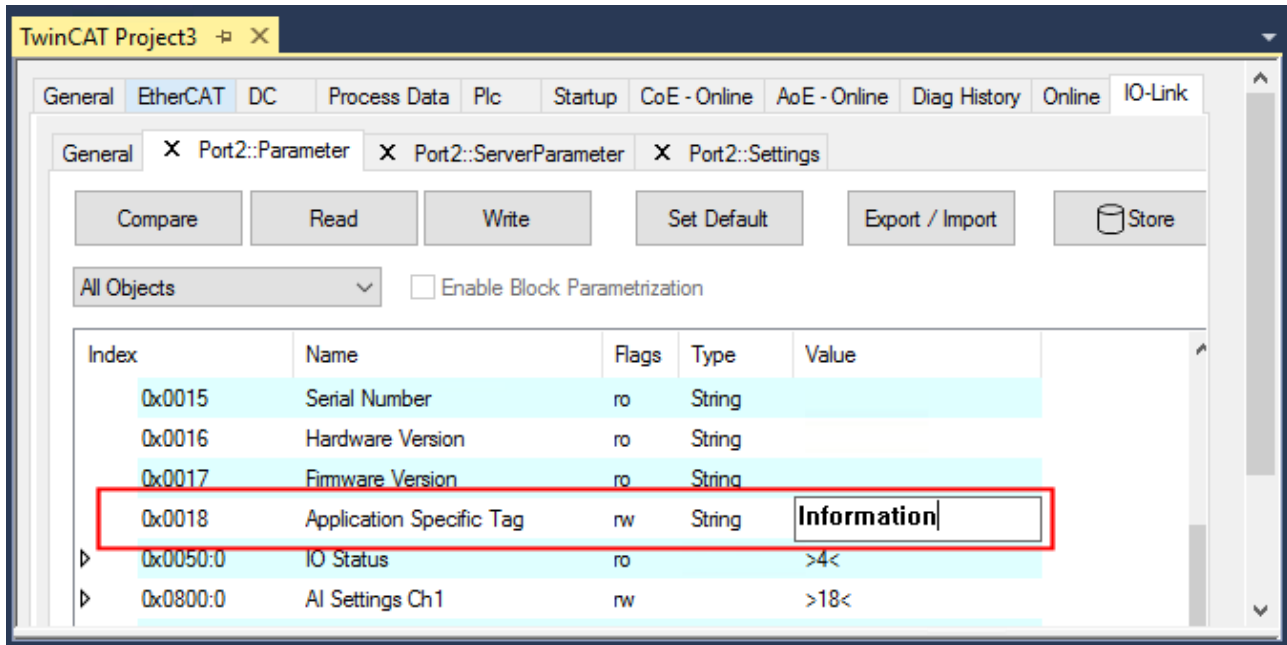


Fig. 55: Parameters IO-Link device: “Application Specific Tag”

6.5 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](#) [[▶ 163](#)] (double-click on the respective object) or via the [Process Data tab](#) [[▶ 160](#)] (allocation of PDOs). Please note the following general [CoE notes](#) [[▶ 31](#)] 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](#) [[▶ 200](#)]" for resetting changes

Introduction

The CoE overview contains objects for different intended applications:

- Objects required for parameterization during [commissioning](#) [[▶ 83](#)]
- Objects for indicating [internal settings](#) [[▶ 85](#)] (may be fixed)
- [Profile-specific objects](#) [[▶ 89](#)] that represent the general status displays and statuses of the inputs and outputs.

The following section first describes the objects required for normal operation, followed by a complete overview of missing objects.

6.5.1 Objects for commissioning

Index 1011 Restore default parameters

Index (hex)	Name	Meaning	Data type	Flags	Default
1011:0	Restore default parameters [▶ 200]	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})

Index 80n0 IO Settings Ch. 1 - 4 (for $0 \leq n \leq 3$)

Index (hex)	Name	Meaning	Data type	Flags	Default
80n0:0	IO Settings	Max. subindex	UINT8	RO	0x28 (40 _{dec})
80n0:04	Device ID	The device ID is used for validating the IO link device.	UINT32	RW	0x00000000 (0 _{dec})
80n0:05	VendorID	The vendor ID is used for validating the manufacturer of the IO link device.	UINT32	RW	0x00000000 (0 _{dec})
80n0:20	IO-Link revision	ID of the specification version based on which the IO link device communicates. Bit 0-3: MinorRev Bit 4-7: MajorRev	UINT8	RW	0x00 (0 _{dec})
80n0:21	FrameCapability	The Frame Capability indicates certain functionalities of the IO link device (e.g. ISDU supported). Bit 0: ISDU Bit 1: Type1 Bit 7: PHY1	UINT8	RW	0x00 (0 _{dec})
80n0:22	Min cycle time	The cycle time refers to the communication between the IO link master and the IO link device. This value is transferred in the IO link format for Min Cycle Time. Bit 6 und 7: Time Base Bit 0 to 5: Multiplier (see Table 1)	UINT8	RW	0x00 (0 _{dec})

Table 1

Time Base	Time base meaning	Calculation	Min. Cycle Time
00 _{bin}	0.1 ms	Multiplier x Time Base	0.0- 6.3 ms
01 _{bin}	0.4 ms	6.4 ms + Multiplier x Time Base	6.4- 31.6 ms
10 _{bin}	1.6 ms	32.0 ms + Multiplier x Time Base	32.0- 132.8 ms
11 _{bin}	6.4 ms	134.4 ms + Multiplier x Time Base	134.4- 537.6 ms

Index (hex)	Name	Meaning	Data type	Flags	Default
80n0:23	Offset time	Reserved	UINT8	RW	0x00 (0 _{dec})
80n0:24	Process data in length	These parameters are transferred in the IO link format for "Process data in length". Bit 7: BYTE (indicates whether the value in LENGTH interpreted as bit length [bit not set] or as byte length + 1 [bit set]) Bit 6: SIO (indicates whether the device supports the standard IO mode [bit set]) Bit 0 to 4: LENGTH (length of the process data)	UINT8	RW	0x00 (0 _{dec})

Index (hex)	Name	Meaning	Data type	Flags	Default
80n0:25	Process data out length	These parameters are transferred in the IO link format for "Process data out length". Bit 7: BYTE (indicates whether the value in LENGTH interpreted as bit length [bit not set] or as byte length + 1 [bit set]) Bit 6: SIO (indicates whether the device supports the standard IO mode [bit set]) Bit 0 to 4: LENGTH (length of the process data)	UINT8	RW	0x00 (0 _{dec})
80n0:26	Compatible ID	reserved	UINT16	RW	0x0000 (0 _{dec})
80n0:27	Reserved	reserved	UINT16	RW	0x0000 (0 _{dec})
80n0:28	Master Control	Controls the IO-Link master port and defines the various operating modes of the IO-Link master. Bits 0..3 0: IO link port inactive 1: IO link port as digital input port 2: IO link port as digital output port 3: IO link port in communication via the IO link protocol 4: IO link port in communication via the IO link protocol. IO link state is ComStop (none cyclic communication, data are exchanged on demand). Bits 4..15 2: Data storage active 4: Data storage upload inactive	UINT16	RW	0x0000 (0 _{dec})

6.5.2 Standard objects (0x1000-0x1FFF)

The standard objects have the same meaning for all EtherCAT slaves.

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	0x184C1389 (407638921 _{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	EL6224

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	01

Index 1018 Identity

Index (hex)	Name	Meaning	Data type	Flags	Default
1018:0	Identity	Information for identifying the slave	UINT8	RO	0x04 (4 _{dec})
1018:01	Vendor ID	Vendor ID of the EtherCAT slave	UINT32	RO	0x00000002 (2 _{dec})
1018:02	Product code	Product code of the EtherCAT slave	UINT32	RO	0x18503052 (407908434 _{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	0x00100000 (1048576 _{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	0x01 (1 _{dec})
10F0:01	Checksum	Checksum across all backup entries of the EtherCAT slave	UINT32	RO	0x00000000 (0 _{dec})

Index 1600 IO RxPDOPDO-Map Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1600:0	IO RxPDOPDO-Map Ch.1	PDO Mapping RxPDO 1	UINT8	RW	0x01 (1 _{dec})
1600:01	SubIndex 001	1. PDO Mapping entry (8 bits align)	UINT32	RW	0x0000:00, 8

Index 1601 IO RxPDOPDO-Map Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1601:0	IO RxPDOPDO-Map Ch.2	PDO Mapping RxPDO 2	UINT8	RW	0x01 (1 _{dec})
1601:01	SubIndex 001	1. PDO Mapping entry (8 bits align)	UINT32	RW	0x0000:00, 8

Index 1602 IO RxPDOPDO-Map Ch.3

Index (hex)	Name	Meaning	Data type	Flags	Default
1602:0	IO RxPDOPDO-Map Ch.3	PDO Mapping RxPDO 3	UINT8	RW	0x01 (1 _{dec})
1602:01	SubIndex 001	1. PDO Mapping entry (8 bits align)	UINT32	RW	0x0000:00, 8

Index 1603 IO RxPDOPDO-Map Ch.4

Index (hex)	Name	Meaning	Data type	Flags	Default
1603:0	IO RxPDOPDO-Map Ch.4	PDO Mapping RxPDO 4	UINT8	RW	0x01 (1 _{dec})
1603:01	SubIndex 001	1. PDO Mapping entry (8 bits align)	UINT32	RW	0x0000:00, 8

Index 1A00 IO TxPDOPDO-Map Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1A00:0	IO TxPDOPDO-Map Ch.1	PDO Mapping TxPDO 1	UINT8	RW	0x01 (1 _{dec})
1A00:01	SubIndex 001	1. PDO Mapping entry (8 bits align)	UINT32	RW	0x0000:00, 8

Index 1A01 IO TxPDOPDO-Map Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1A01:0	IO TxPDOPDO-Map Ch.2	PDO Mapping TxPDO 2	UINT8	RW	0x01 (1 _{dec})
1A01:01	SubIndex 001	1. PDO Mapping entry (8 bits align)	UINT32	RW	0x0000:00, 8

Index 1A02 IO TxPDOPDO-Map Ch.3

Index (hex)	Name	Meaning	Data type	Flags	Default
1A02:0	IO TxPDOPDO-Map Ch.3	PDO Mapping TxPDO 3	UINT8	RW	0x01 (1 _{dec})
1A02:01	SubIndex 001	1. PDO Mapping entry (8 bits align)	UINT32	RW	0x0000:00, 8

Index 1A03 IO TxPDOPDO-Map Ch.4

Index (hex)	Name	Meaning	Data type	Flags	Default
1A03:0	IO TxPDOPDO-Map Ch.4	PDO Mapping TxPDO 4	UINT8	RW	0x01 (1 _{dec})
1A03:01	SubIndex 001	1. PDO Mapping entry (8 bits align)	UINT32	RW	0x0000:00, 8

Index 1A04 DeviceState TxPDO-Map Inputs

Index (hex)	Name	Meaning	Data type	Flags	Default
1A04:0	DeviceState TxPDO-Map Inputs	PDO Mapping TxPDO 5	UINT8	RW	0x04 (4 _{dec})
1A04:01	SubIndex 001	1. PDO Mapping entry (object 0xF100 (Diagnosis Status data), entry 0x01 (State Ch1))	UINT32	RW	0xF100:01, 8
1A04:02	SubIndex 002	2. PDO Mapping entry (object 0xF100 (Diagnosis Status data), entry 0x02 (State Ch2))	UINT32	RW	0xF100:02, 8
1A04:03	SubIndex 003	3. PDO Mapping entry (object 0xF100 (Diagnosis Status data), entry 0x03 (State Ch3))	UINT32	RW	0xF100:03, 8
1A04:04	SubIndex 004	4. PDO Mapping entry (object 0xF100 (Diagnosis Status data), entry 0x04 (State Ch4))	UINT32	RW	0xF100:04, 8

Index 1A05 DeviceState TxPDO-Map Inputs Device

Index (hex)	Name	Meaning	Data type	Flags	Default
1A05:0	DeviceState TxPDO-Map Inputs Device	PDO Mapping TxPDO 6	UINT8	RW	0x05 (5 _{dec})
1A05:01	SubIndex 001	1. PDO Mapping entry (8 bits align)	UINT32	RW	0x0000:00, 8
1A05:02	SubIndex 00	2. PDO Mapping entry (4 bits align)	UINT32	RW	0x0000:00, 4
1A05:03	SubIndex 00	3. PDO Mapping entry (object 0xF101 (DeviceState status data), entry 0x0D (Device diag))	UINT32	RW	0xF101:0D, 1
1A05:04	SubIndex 00	4. PDO Mapping entry (2 bits align)	UINT32	RW	0x0000:00, 2
1A05:05	SubIndex 00	5. PDO Mapping entry (object 0xF101 (DeviceState status data), entry 0x0D (Device state))	UINT32	RW	0xF101:10, 1

Index 1C00 Sync manager type

Index (hex)	Name	Meaning	Data type	Flags	Default
1C00:0	Sync manager type	Using the sync managers	UINT8	RO	0x04 (4 _{dec})
1C00:01	SubIndex 001	Sync-Manager Type Channel 1: Mailbox Write	UINT8	RO	0x01 (1 _{dec})
1C00:02	SubIndex 002	Sync-Manager Type Channel 2: Mailbox Read	UINT8	RO	0x02 (2 _{dec})
1C00:03	SubIndex 003	Sync-Manager Type Channel 3: Process Data Write (Outputs)	UINT8	RO	0x03 (3 _{dec})
1C00:04	SubIndex 004	Sync-Manager Type Channel 4: Process Data Read (Inputs)	UINT8	RO	0x04 (4 _{dec})

Index 1C12 RxPDO assign

Index (hex)	Name	Meaning	Data type	Flags	Default
1C12:0	RxPDO assign	PDO Assign Outputs	UINT8	RW	0x04 (4 _{dec})
1C32:01	SubIndex 001	1. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x1600 (5632 _{dec})
1C12:02	SubIndex 002	2. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x1601 (5633 _{dec})
1C12:03	SubIndex 003	3. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x1602 (5634 _{dec})
1C12:04	SubIndex 004	4. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x1603 (5635 _{dec})

Index 1C13 TxPDO assign

Index (hex)	Name	Meaning	Data type	Flags	Default
1C13:0	TxPDO assign	PDO Assign Inputs	UINT8	RW	0x05 (5 _{dec})
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	0x1A02 (6658 _{dec})
1C13:04	SubIndex 004	4. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A03 (6659 _{dec})
1C13:05	SubIndex 005	5. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A04 (6660 _{dec})

Index 1C32 SM output parameter

Index (hex)	Name	Meaning	Data type	Flags	Default
1C32:0	SM output parameter	Synchronization parameters for the outputs	UINT8	RO	0x20 (32 _{dec})
1C32:01	Sync mode	Current synchronization mode: 0: Free Run 1: Synchronous with SM 2 event 2: DC-Mode - Synchronous with SYNC0 Event 3: DC-Mode - Synchronous with SYNC1 event	UINT16	RW	0x0000 (0 _{dec})
1C32:02	Cycle time	Cycle time (in ns): - Free Run: Cycle time of the local timer - Synchronous with SM 2 event: Master cycle time - DC mode: SYNC0/SYNC1 Cycle Time	UINT32	RW	0x000186A0 (100000 _{dec})
1C32:03	Shift time	Time between SYNC0 event and output of the outputs (in ns, DC mode only)	UINT32	RO	0x00000000 (0 _{dec})
1C32:04	Sync modes supported	Supported synchronization modes: - Bit 0 = 1: free run is supported - Bit 1 = 1: Synchron with SM 2 event is supported - Bit 2-3 = 01: DC mode is supported - Bit 4-5 = 10: Output shift with SYNC1 event (only DC mode) - Bit 14 = 1: dynamic times (measurement through writing of 0x1C32:08 [► 88])	UINT16	RO	0xC007 (49159 _{dec})
1C32:05	Minimum cycle time	Minimum cycle time (in ns)	UINT32	RO	0x000186A0 (100000 _{dec})
1C32:06	Calc and copy time	Minimum time between SYNC0 and SYNC1 event (in ns, DC mode only)	UINT32	RO	0x00000000 (0 _{dec})
1C32:08	Command	0: Measurement of the local cycle time is stopped 1: Measurement of the local cycle time is started The entries 0x1C32:03 [► 88], 0x1C32:05 [► 88], 0x1C32:06 [► 88], 0x1C32:09 [► 88], 0x1C33:03 [► 89], 0x1C33:06 [► 88], 0x1C33:09 [► 89] are updated with the maximum measured values. For a subsequent measurement the measured values are reset	UINT16	RW	0x0000 (0 _{dec})
1C32:09	Delay time	Time between SYNC1 event and output of the outputs (in ns, DC mode only)	UINT32	RO	0x00000000 (0 _{dec})
1C32:0B	SM event missed counter	Number of missed SM events in OPERATIONAL (DC mode only)	UINT16	RO	0x0000 (0 _{dec})
1C32:0C	Cycle exceeded counter	Number of occasions the cycle time was exceeded in OPERATIONAL (cycle was not completed in time or the next cycle began too early)	UINT16	RO	0x0000 (0 _{dec})
1C32:0D	Shift too short counter	Number of 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	0x20 (32 _{dec})
1C33:01	Sync mode	Current synchronization mode: 0: Free Run 1: Synchron with SM 3 Event (no outputs available) 2: DC - Synchron with SYNC0 Event 3: DC - Synchron with SYNC1 Event 34: Synchron with SM 2 Event (outputs available)	UINT16	RW	0x0000 (0 _{dec})
1C33:02	Cycle time	as 0x1C32:02 [► 88]	UINT32	RW	0x000186A0 (100000 _{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: Synchronous with SM 2 Event is supported (outputs available) - Bit 1: Synchronous 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 [► 88] or 0x1C33:08 [► 89])	UINT16	RO	0xC007 (49159 _{dec})
1C33:05	Minimum cycle time	as 0x1C32:05 [► 88]	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})
1C33:08	Command	as 0x1C32:08 [► 88]	UINT16	RW	0x0000 (0 _{dec})
1C33:09	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	as 0x1C32:11 [► 88]	UINT16	RO	0x0000 (0 _{dec})
1C33:0C	Cycle exceeded counter	as 0x1C32:12 [► 88]	UINT16	RO	0x0000 (0 _{dec})
1C33:0D	Shift too short counter	as 0x1C32:13 [► 88]	UINT16	RO	0x0000 (0 _{dec})
1C33:20	Sync error	as 0x1C32:32 [► 88]	BOOLEAN	RO	0x00 (0 _{dec})

6.5.3 Profile-specific objects (0x6000-0xFFFF)

Index 60n0 IO Inputs Ch. 1 - 4 (for 0 ≤ n ≤ 3)

Index (hex)	Name	Meaning	Data type	Flags	Default
60n0:0	IO Inputs Ch.1 - 4	Max. subindex	UINT8	RO	0x00 (0 _{dec})
60n0:01	Subindex 001	IO-Link input process data	-	RO	-
60n0:10	Subindex 016	IO-Link input process data	-	RO	-

Index 70n0 IO Outputs Ch. 1 - 4 (for 0 ≤ n ≤ 3)

Index (hex)	Name	Meaning	Data type	Flags	Default
70n0:0	IO Outputs Ch.1 - 4	Max. subindex	UINT8	RO	0x00 (0 _{dec})
70n0:01	Subindex 001	IO-Link output process data	-	RO	-
70n0:10	Subindex 016	IO-Link output process data	-	RO	-

Index 90n0 IO Info data Ch. 1 - 4 (for $0 \leq n \leq 3$)

Index (hex)	Name	Meaning	Data type	Flags	Default
90n0:0	IO Info data	Max. subindex	UINT8	RO	0x27 (39 _{dec})
90n0:04	Device ID	The device ID is used for validating the IO link device.	UINT32	RO	0x00000000 (0 _{dec})
90n0:05	VendorID	The vendor ID is used for validating the manufacturer of the IO link device.	UINT32	RO	0x00000000 (0 _{dec})
90n0:07	IO-Link revision	ID of the specification version based on which the IO link device communicates. Bit 0-3: MinorRev Bit 4-7: MajorRev	UINT8	RO	0x00 (0 _{dec})
90n0:20	FrameCapability	The Frame Capability indicates certain functionalities of the IO link device (e.g. ISDU supported). Bit 0: ISDU Bit 1: Type1 Bit 7: PHY1	UINT8	RO	0x00 (0 _{dec})
90n0:21	Min cycle time	The cycle time refers to the communication between the IO link master and the IO link device. This value is transferred in the IO link format for Min Cycle Time. Bit 6 und 7: Time Base Bit 0 to 5: Multiplier (see Table 2)	UINT8	RO	0x00 (0 _{dec})
90n0:22	Offset time	reserved	UINT8	RO	0x00 (0 _{dec})
90n0:23	Process data in length	These parameters are transferred in the IO link format for "Process data in length". Bit 7: BYTE (indicates whether the value in LENGTH interpreted as bit length [bit not set] or as byte length + 1 [bit set]) Bit 6: SIO (indicates whether the device supports the standard IO mode [bit set]) Bit 0 to 4: LENGTH (length of the process data)	UINT8	RO	0x00 (0 _{dec})
90n0:24	Process data out length	These parameters are transferred in the IO link format for "Process data out length". Bit 7: BYTE (indicates whether the value in LENGTH interpreted as bit length [bit not set] or as byte length + 1 [bit set]) Bit 6: SIO (indicates whether the device supports the standard IO mode [bit set]) Bit 0 to 4: LENGTH (length of the process data)	UINT8	RO	0x00 (0 _{dec})
90n0:26	Reserved	reserved	UINT16	RO	0x0000 (0 _{dec})
90n0:27	Reserved2	reserved	UINT16	RO	0x0000 (0 _{dec})

Table 2

Time Base	Time base meaning	Calculation	Min. Cycle Time
00b	0.100 ms	Multiplier x Time Base	0.000- 6.300 ms
01b	0.400 ms	6.400 ms + Multiplier x Time Base	6.400- 31.600 ms
10b	1.600 ms	32.000 ms + Multiplier x Time Base	32.000- 132.800 ms
11b	6.400 ms	134.400 ms + Multiplier x Time Base	134.400 - 537.600 ms

Index A0n0 IO Diag data Ch. 1 - 4 (for $0 \leq n \leq 3$)

Index (hex)	Name	Meaning	Data type	Flags	Default
A0n0:0	IO Diag data Ch.1 - 4	Max. subindex	UINT8	RO	0x02 (2 _{dec})
A0n0:01	IO-Link state	The value of the IO link state corresponds to a state from the IO link master state machine 0x00: MASTER_STATE_INACTIV 0x01: MASTER_STATE_DIGIN 0x02: MASTER_STATE_DIGOUT 0x03: MASTER_STATE_COMESTABLISH 0x04: MASTER_STATE_INITMASTER 0x05: MASTER_STATE_INITSLAVE 0x07: MASTER_STATE_PREOPERATE 0x08: MASTER_STATE_OPERATE 0x09: MASTER_STATE_STOP	UINT8	RO	0x00 (0 _{dec})
A0n0:02	Lost Frames	This parameter counts the number of lost IO link telegrams. This value is deleted whenever IO link starts up, otherwise it is incremented continuously.	UINT8	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	0x02 (2 _{dec})
F000:01	Module index distance	Index distance of the objects of the individual channels	UINT16	RO	0x0010 (16 _{dec})
F000:02	Maximum number of modules	Number of channels	UINT16	RO	0x0004 (4 _{dec})

Index F008 Code word

Index (hex)	Name	Meaning	Data type	Flags	Default
F008:0	Code word	reserved	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	0x04 (4 _{dec})
F010:01	SubIndex 001	-	UINT32	RW	0x0000184C (6220 _{dec})
F010:02	SubIndex 002	-	UINT32	RW	0x0000184C (6220 _{dec})
F010:03	SubIndex 003	-	UINT32	RW	0x0000184C (6220 _{dec})
F010:04	SubIndex 004	-	UINT32	RW	0x0000184C (6220 _{dec})

Index (hex)	Name	Meaning		Data type	Flags	Default
F100:0	Diagnosis Status data	Max. subindex		UINT8	RO	0x04 (4 _{dec})
F100:01	State Ch1	Status byte Ch. 1	See table “Meaning Status byte Ch. 1 - Ch. 4”	UINT8	RO	0x00 (0 _{dec})
F100:02	State Ch2	Status byte Ch. 2		UINT8	RO	0x00 (0 _{dec})
F100:03	State Ch3	Status byte Ch. 3		UINT8	RO	0x00 (0 _{dec})
F100:04	State Ch4	Status byte Ch. 4		UINT8	RO	0x00 (0 _{dec})

The status bytes are divided into two nibbles.

Meaning Status byte Ch. 1 - Ch. 4					
Low nibble:					
0x_0 = Port disabled					
0x_1 = Port in std dig in					
0x_2 = Port in std dig out					
0x_3 = Port in communication OP					
0x_4 = Port in communication COMSTOP / dig in Bit (only in std. IO Mode)					
0x_5 = not defined					
0x_6 = not defined					
0x_7 = not defined					
0x_8 = Process Data Invalid Bit					
Combinations are possible and are displayed as addition of the values (s. note)					
Higher nibble:					
0x1_ = Watchdog detected					
0x2_ = internal Error					
0x3_ = invalid Device ID					
0x4_ = invalid Vendor ID					
0x5_ = invalid IO-Link Version					
0x6_ = invalid Frame Capability					
0x7_ = invalid Cycle Time					
0x8_ = invalid PD in length					
0x9_ = invalid PD out length					
0xA_ = no Device detected					
0xB_ = error PreOP/Data storage					
Combinations are possible and are displayed as addition of the values (s. note)					



Addition of the values in case of simultaneously occurring diagnostic messages

If messages occur simultaneously, the value is displayed as a sum in the Status byte of the relevant channel.

- Often for example 0x03 "Port in communication OP" and 0x08 "Process Data Invalid Bit" occur simultaneously:

$$0x03 + 0x08 = 0x0B (11_{dec})$$

⇒ The value 0x0B (11_{dec}) is displayed in the Status byte.

Index F101 DeviceState status data

Index (hex)	Name	Meaning	Data type	Flags	Default
F101:0	DeviceState status data	Max. Subindex	UINT8	RO	0x10(16 _{dec})
F101:0D	Device diag	TRUE: A new diagnosis message available in the DiagHistory	BOOL	RO	FALSE
F101:10	Device state	TRUE collective message, at least 1 device with error	BOOL	RO	FALSE

Index F820 ADS Server Settings

Index (hex)	Name	Meaning	Data type	Flags	Default
F820:0	ADS Server Settings	Max. Subindex	UINT8	RW	0x02 (2 _{dez})
F820:01	Net ID	DiagHistory messages can be sent to this NetID and	UINT16	RW	0x0000 (0 _{dec})
F820:02	Port	Port via Emergency	UINT16	RW	0x0000 (0 _{dec})

Index F900 Info data

Index (hex)	Name	Meaning	Data type	Flags	Default
F900:0	Info data	Max. subindex	UINT8	RO	0x09 (9 _{dec})
F900:01	IO-Link version	-	UINT8	RO	0x10 (16 _{dec})

6.5.4 Objects TwinSAFE Single Channel (EL6224-0090)

Index 1690 TSC RxPDO-Map Master Message

Index (hex)	Name	Meaning	Data type	Flags	Default
1690:0	TSC RxPDO-Map Master Message	PDO Mapping RxPDO	UINT8	RO	0x04 (4 _{dec})
1690:01	SubIndex 001	1. PDO Mapping entry (object 0x700F (TSC Master Frame Elements), entry 0x01 (TSC__Master Cmd))	UINT32	RO	0x700F:01, 8
1690:02	SubIndex 002	2. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8
1690:03	SubIndex 003	3. PDO Mapping entry (object 0x700F (TSC Master Frame Elements), entry 0x03 (TSC__Master CRC_0))	UINT32	RO	0x700F:03, 16
1690:04	SubIndex 004	4. PDO Mapping entry (object 0x700F (TSC Master Frame Elements), entry 0x02 (TSC__Master ConnID))	UINT32	RO	0x700F:02, 16

Index 1A90 TSC TxPDO-Map Slave Message

Index (hex)	Name	Meaning	Data type	Flags	Default
1A90:0	TSC TxPDO-Map Slave Message	PDO Mapping TxPDO	UINT8	RW	0x04 (4 _{dez})
1A90:01	SubIndex 001	1. PDO Mapping entry (object 0x600F (TSC Slave Frame Elements), entry 0x01 (TSC__Slave Cmd))	USINT8	RW	0x600F:01, 8
1A90:02	SubIndex 002	2. PDO Mapping entry (object 0x6004 (unsigned Word Inputs Ch. 1), entry 0x01 (Offset Byte 0))	INT16	RW	0x6004:01, 16
1A90:03	SubIndex 003	3. PDO Mapping entry (object 0x600F (TSC Slave Frame Elements), entry 0x03 (TSC__Slave CRC_0))	UINT16	RW	0x600F:03, 16
...	...	...	...	...	...

Index 600F TSC Slave Frame Elements

Index (hex)	Name	Meaning	Data type	Flags	Default
600F:0	TSC Slave Frame Elements	Max. subindex	UINT8	RO	0x0A (10 _{dec})
600F:01	TSC__Slave Cmd	reserved	UINT8	RO	0x00 (0 _{dec})
600F:02	TSC__Slave ConnID	reserved	UINT16	RO	0x0000 (0 _{dec})
600F:03	TSC__Slave CRC_0	reserved	UINT16	RO	0x0000 (0 _{dec})
600F:04	TSC__Slave CRC_1	reserved	UINT16	RO	0x0000 (0 _{dec})
600F:05	TSC__Slave CRC_2	reserved	UINT16	RO	0x0000 (0 _{dec})
600F:06	TSC__Slave CRC_3	reserved	UINT16	RO	0x0000 (0 _{dec})
600F:07	TSC__Slave CRC_4	reserved	UINT16	RO	0x0000 (0 _{dec})
600F:08	TSC__Slave CRC_5	reserved	UINT16	RO	0x0000 (0 _{dec})
600F:09	TSC__Slave CRC_6	reserved	UINT16	RO	0x0000 (0 _{dec})
600F:0A	TSC__Slave CRC_7	reserved	UINT16	RO	0x0000 (0 _{dec})

Index 60a1 DWord Inputs Ch. a+1 (a = 0, 1, 2, 3)

Index (hex)	Name	Meaning	Data type	Flags	Default
60a1:0	DWord Inputs Ch. a+1	Max. subindex	UINT8	RO	0x1D (29 _{dec})
60a1:01	Offset Byte 0	Offset Byte 0	()	RO	()
60a1:02	Offset Byte 1	Offset Byte 1	()	RO	()
...	...	...	...	...	...
60a1:1D	Offset Byte 28	Offset Byte 28	()	RO	()

Index 60b2 unsigned DWord Inputs Ch. b+1 (b = 0, 1, 2, 3)

Index (hex)	Name	Meaning	Data type	Flags	Default
60b2:0	Unsigned DWord Inputs Ch. b+1	Max. subindex	UINT8	RO	0x1D (29 _{dec})
60b2:01	Offset Byte 0	Offset Byte 0	()	RO	()
60b2:02	Offset Byte 1	Offset Byte 1	()	RO	()
...	...	...	...	...	...
60b2:1D	Offset Byte 28	Offset Byte 28	()	RO	()

Index 60c3 Word Inputs Ch. c+1 (c = 0, 1, 2, 3)

Index (hex)	Name	Meaning	Data type	Flags	Default
60c3:0	Word Inputs Ch. c+1	Max. subindex	UINT8	RO	0x1F (31 _{dec})
60c3:01	Offset Byte 0	Offset Byte 0	()	RO	()
60c3:02	Offset Byte 1	Offset Byte 1	()	RO	()
...	...	...	...	...	...
60c3:1F	Offset Byte 30	Offset Byte 30	()	RO	()

Index 60d4 unsigned Word Inputs Ch. d+1 (d = 0, 1, 2, 3)

Index (hex)	Name	Meaning	Data type	Flags	Default
60d4:0	Unsigned Word Inputs Ch. d+1	Max. subindex	UINT8	RO	0x1F (31 _{dec})
60d4:01	Offset Byte 0	Offset Byte 0	()	RO	()
60d4:02	Offset Byte 1	Offset Byte 1	()	RO	()
...	...	...	...	...	...
60d4:1F	Offset Byte 30	Offset Byte 30	()	RO	()

Index 61e0 Bool Inputs Ch. e+1 (e = 0, 1, 2, 3)

Index (hex)	Name	Meaning	Data type	Flags	Default
61e0:0	Bool Inputs Ch. e+1	Max. subindex	UINT8	RO	0x40 (64 _{dec})
61e0:01	Offset Byte 0 Bit 0	Offset Byte 0 Bit 0	()	RO	()
61e0:02	Offset Byte 0 Bit 1	Offset Byte 0 Bit 1	()	RO	()
...	...	...	...	...	...
61e0:08	Offset Byte 0 Bit 7	Offset Byte 0 Bit 7	()	RO	()
61e0:09	Offset Byte 1 Bit 0	Offset Byte 1 Bit 0	()	RO	()
61e0:0A	Offset Byte 1 Bit 1	Offset Byte 1 Bit 1	()	RO	()
...	...	...	...	...	...
61e0:10	Offset Byte 1 Bit 7	Offset Byte 1 Bit 7	()	RO	()
...	...	...	...	...	...
...	...	...	...	...	...
61e0:39	Offset Byte 7 Bit 0	Offset Byte 7 Bit 0	()	RO	()
61e0:3A	Offset Byte 7 Bit 1	Offset Byte 7 Bit 1	()	RO	()
...	...	...	...	...	...
61e0:40	Offset Byte 7 Bit 7	Offset Byte 7 Bit 7	()	RO	()

Index 700F TSC Master Frame Elements

Index (hex)	Name	Meaning	Data type	Flags	Default
700F:0	TSC Master Frame Elements	Max. subindex	UINT8	RO	0x03 (3 _{dec})
700F:01	TSC__Master Cmd	reserved	UINT8	RO	0x00 (0 _{dec})
700F:02	TSC__Master ConnID	reserved	UINT16	RO	0x0000 (0 _{dec})
700F:03	TSC__Master CRC_0	reserved	UINT16	RO	0x0000 (0 _{dec})

Index 800F TSC Settings

Index (hex)	Name	Meaning	Data type	Flags	Default
800F:0	TSC Settings	Max. subindex	UINT8	RO	0x02 (2 _{dez})
800F:01	Address	TwinSAFE SC Adresse	UINT16	RW	0x0000 (0 _{dez})
800F:02	Connection Mode	Auswahl der TwinSAFE SC CRC	UINT32	RO	0x00000000 (0 _{dez})

7 Access to IO-Link data

7.1 IO-Link system communication

The Beckhoff EL6224 IO-Link master terminal is divided into two services.

- On the one hand it represents an IO-Link master in relation to the connected IO-Link devices,
- while on the other it is an EtherCAT slave in relation to the PLC TwinCAT master.

The system communication is illustrated in the following figure.

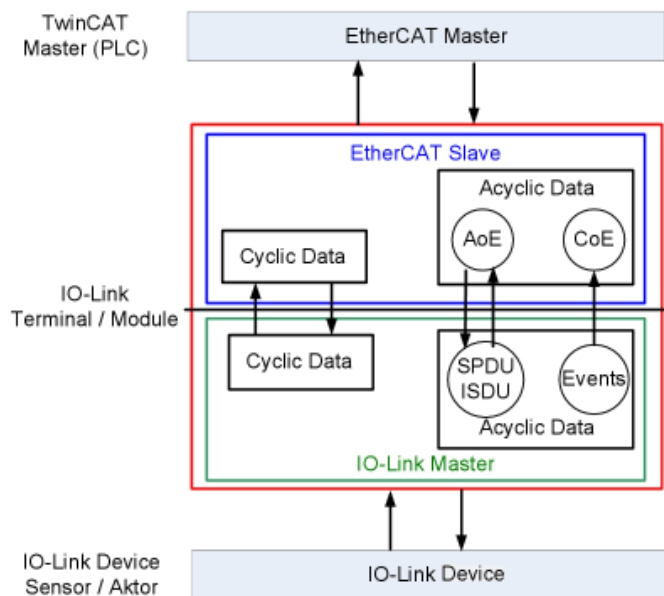


Fig. 56: Illustration of the system communication of an EtherCAT master

In principle cyclic and acyclic data are exchanged. The cyclic process data can be accessed via the [PDOs](#) [► 97], the acyclic data via [AoE](#) [► 98]. The events are additionally displayed under [Diag History](#) [► 101] in the System Manager.

- **Cyclic data:**
 - Process data
 - Value status
- **Acyclic data:**
 - Device data
 - Events

7.2 PDO Assignment

The scope of the process data offered varies depending on the configured IO-Link ports.

“DeviceState Inputs Device” and “DeviceState Inputs” are selected by default. Device-specific PDOs (0x1A0n “Port (n-1) Process Data”) are only displayed after configuring on the respective port and restarting the EtherCAT system or reloading the configuration in Config mode; see [Activating the configuration](#) [► 69].

Fig. 57: Illustration of the process data allocation, SM3 inputs using EL6224 as an example

SM3, PDO Assignment 0x1C13			
Index	Size (byte.bit)	Name	PDO content
0x1A05 (0x1A81*)	2.0	DeviceState Inputs Device	Index 0xF101:0D - Device Diag Index 0xF101:10 - Device State
0x1A04 (0x1A80*)	4.0	DeviceState Inputs	Index 0xF100:01 - State Ch1 Index 0xF100:02 - State Ch2 Index 0xF100:03 - State Ch3 Index 0xF100:04 - State Ch4
0x1A00	0.0 - 32.0	Port 1 Process Data	IO-Link device-specific / only active after configuration
0x1A01	0.0 - 32.0	Port 2 Process Data	IO-Link device-specific / only active after configuration
0x1A02	0.0 - 32.0	Port 3 Process Data	IO-Link device-specific / only active after configuration
0x1A03	0.0 - 32.0	Port 4 Process Data	IO-Link device-specific / only active after configuration

*) Due to the further development of the software, the objects 0x1A80/0x1A81 are used for newer product variants (e.g. EL6224-0090).



Process data representation

If data types are used that don't conform to IEC61131-3, they are represented as octed strings.

The status of the IO-Link ports 1-4 is indicated via index 0xF100:0n.
The indices 0xF101:xx provide general diagnostic data.

Index	Size (byte.bit)	Name	Meaning
0xF101:0D	0.1	Device Diag	Occurrence of events (on the device side) is reported via a status bit
0xF101:10	0.1	Device State	Interruption of communication with one of the devices is reported via a status bit
0xF100:01	1.0	State Ch.1	See table "Meaning Status byte Ch. 1 - Ch. 4"
0xF100:02	1.0	State Ch.2	
0xF100:03	1.0	State Ch.3	
0xF100:04	1.0	State Ch.4	

The status bytes are divided into two nibbles.

Meaning Status byte Ch. 1 - Ch. 4
Low nibble: 0x_0 = Port disabled 0x_1 = Port in std dig in 0x_2 = Port in std dig out 0x_3 = Port in communication OP 0x_4 = Port in communication COMSTOP / dig in Bit (only in std. IO Mode) 0x_5 = not defined 0x_6 = not defined 0x_7 = not defined 0x_8 = Process Data Invalid Bit Combinations are possible and are displayed as addition of the values (s. note)
Higher nibble: 0x1_ = Watchdog detected 0x2_ = internal Error 0x3_ = invalid Device ID 0x4_ = invalid Vendor ID 0x5_ = invalid IO-Link Version 0x6_ = invalid Frame Capability 0x7_ = invalid Cycle Time 0x8_ = invalid PD in length 0x9_ = invalid PD out length 0xA_ = no Device detected 0xB_ = error PreOP/Data storage Combinations are possible and are displayed as addition of the values (s. note)



Addition of the values in case of simultaneously occurring diagnostic messages

If messages occur simultaneously, the value is displayed as a sum in the Status byte of the relevant channel.

- Often for example 0x03 "Port in communication OP" and 0x08 "Process Data Invalid Bit" occur simultaneously:

$$0x03 + 0x08 = 0x0B \text{ (11}_{\text{dec}}\text{)}$$

⇒ The value 0x0B (11_{dec}) is displayed in the Status byte.

7.3 Accessing IO-Link parameters

The exchange of the acyclic data takes place via a specified index and subindex range that is device-specific and can be read about in the corresponding vendor documentation.

7.4 Parameter data exchange

An intelligent IO-Link sensor/actuator can support parameterization by ISDUs (Indexed Service Data Unit). The PLC must explicitly query or, when marked as such, send these acyclic service data.



ISDU access

TwinCAT supports access via ADS and via the CoE directory.

The respective parameter is addressed via the so-called ISDU index. The following ranges are available:

Designation	Index range
System	0x00..0x0F
Identification	0x10..0x1F
Diagnostic	0x20..0x2F
Communication	0x30..0x3F
Preferred Index	0x40..0xFE
Extended Index	0x0100..0x3FFF
	The range 0x4000 to 0xFFFF is reserved

The use and implementation of these ranges is the responsibility of the sensor/actuator manufacturer. For clarification, just a few of the possible indices are listed here. Please take a look at the relevant chapter "Object description and parameterization".

Index	Name
0010	Vendor Name
0011	Vendor Text
0012	Product Name
0013	Product ID
0015	Serial Number
0016	Hardware Revision
0017	Firmware Revision
...	...

IO-Link operating modes

The IO-Link ports on the IO-Link master can be operated in the following nine modes (see Object description and parameterization - IO-Link State, Index [0xA0n0](#) [[▶ 91](#)]:01):

- INACTIVE: Statemachine is inactive
- DIGINPUT: The port behaves like a digital input
- DIGOUTPUT: The port behaves like a digital output
- ESTABLISHCOMM: The IO-Link wakeup sequence is running
- INITMASTER: Readout the IO-Link device and check the communication parameters
- INITDEVICE: Initialization of the IO-Link device
- PREOPERATE: Parameter server is running
- OPERATE: The port is used for IO-Link communication
- STOP: Communication is stopped (COM-stop)

7.5 ADS

Communication relating to IO link demand data is initiated via an ADS command. An ADS address always consists of a NetID and PortNo. TwinCAT relays ADS commands to EL6224/EJ6224 via AoE (**A**DS **o**ver **E**therCAT), from where the command is relayed to the IO link master section and therefore the data channel.

AoE NetID

The EL6224/EJ6224 is assigned a dedicated AoE-NetID for communication with the IO link master section. This is allocated by the configuration tool (fig. *AoE-NetID allocation*).

NetID under “EL6224/EJ6224” -> „EtherCAT“-> „Advanced Settings“ -> „Mailbox“ -> “AoE”

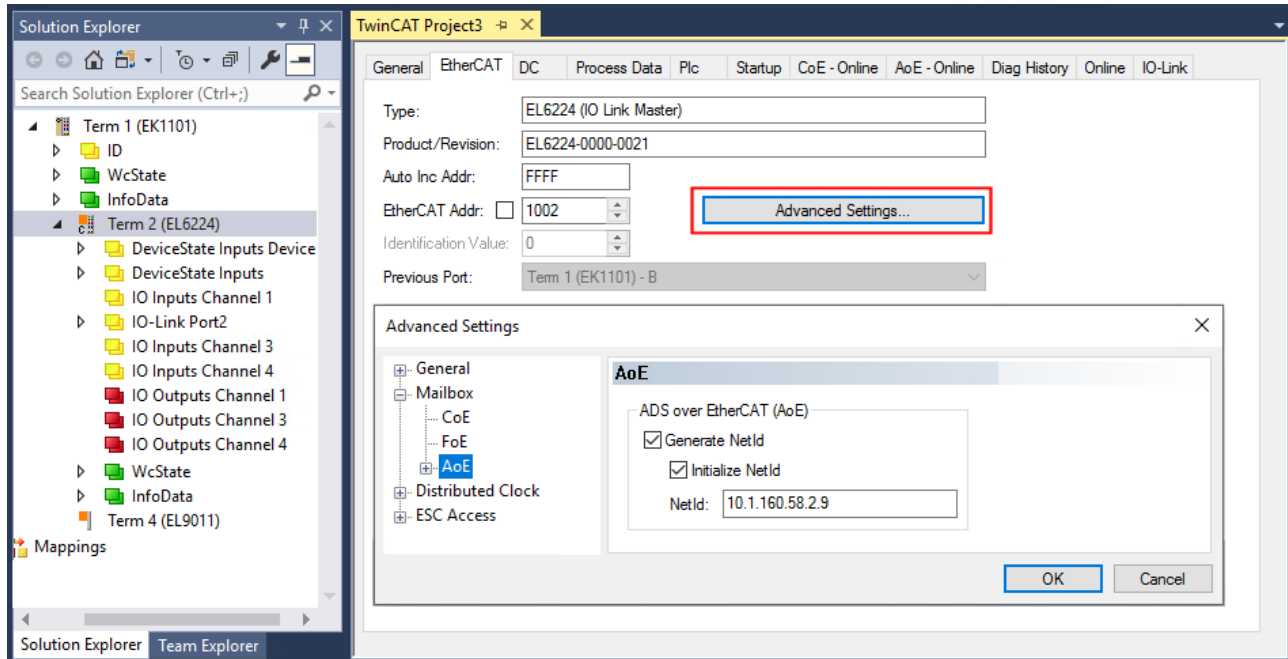


Fig. 58: AoE-NetID allocation using the example of EL6224

PortNo

The individual IO link ports for the master are allocated via the port number. The port numbers are allocated in ascending order from 0x1000. i.e. IO-Link Port1 corresponds to PortNo 0x1000 and IO-Link Portn corresponds to PortNo 0x1000 + n-1.

The following specification applies for the EL6224 (4-port IO link master):

- IO-Link Port1 corresponds to PortNo 0x1000
- IO-Link Port2 corresponds to PortNo 0x1001
- IO-Link Port3 corresponds to PortNo 0x1002
- IO-Link Port4 corresponds to PortNo 0x1003

ADS Indexgroup

As for CoE, the Indexgroup of an ADS command is specified as 0xF302 for the IO link data channel.

ADS Indexoffset

The IO link addressing with index and subindex is coded in the Indexoffset. The Indexoffset has a size of 4 bytes and is subdivided as follows: 2-byte index, 1-byte reserve, 1-byte subindex.

- Example: Indexoffset 0x12340056 corresponds to index 0x1234 and subindex 56

7.6 Access to events

Some of the IO-Link sensors forward events that occur to the master. These events may be items of information, warnings or error messages, e.g. short circuit or overheating.

The IO-Link master reports these events by setting the [Device Diag](#) [► 97] bit. Further information on the events can be read via the CoE directory or the DiagHistory tab.

General EtherCAT DC Process Data Startup CoE - Online Diag History Online IO-Link				
Update History		☒ Auto Update	Ack. Messages	Export Diag History
		☐ only new Messages	Advanced...	
Type	Flags	Timestamp	Message	
Warning	N	13.10.2014 10:11:18 433 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00A4 (0xFFFF8CB0) Unknown TextId	
Warning	N	13.10.2014 10:11:18 355 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00E4 (0xFFFF8CB0) Unknown TextId	
Error	N	13.10.2014 10:11:16 47 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00B4 (0xFFFF8CB4) Unknown TextId	
Error	N	13.10.2014 10:11:15 963 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00F4 (0xFFFF8CB4) Unknown TextId	
Error	N	13.10.2014 10:11:12 661 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00B4 (0xFFFF8CB5) Unknown TextId	
Error	N	13.10.2014 10:11:12 576 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00F4 (0xFFFF8CB5) Unknown TextId	
Warning	N	13.10.2014 10:11:07 500 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00E4 (0xFFFF8CA4) Unknown TextId	
Warning	N	13.10.2014 10:10:52 889 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00A4 (0xFFFF8CB1) Unknown TextId	
Warning	N	13.10.2014 10:10:52 811 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00E4 (0xFFFF8CB1) Unknown TextId	
Error	N	13.10.2014 10:10:51 758 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00B4 (0xFFFF8CB4) Unknown TextId	
Error	N	13.10.2014 10:10:51 673 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00F4 (0xFFFF8CB4) Unknown TextId	
Warning	N	13.10.2014 10:10:50 471 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00A4 (0xFFFF8CA4) Unknown TextId	
Warning	N	13.10.2014 10:10:50 393 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00E4 (0xFFFF8CA4) Unknown TextId	
Warning	N	13.10.2014 10:10:04 93 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00A4 (0xFFFF8CB0) Unknown TextId	
Warning	N	13.10.2014 10:10:04 9 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00E4 (0xFFFF8CB0) Unknown TextId	
Error	N	13.10.2014 10:10:01 194 ms	(0x0001) IO-Link Master Port 1: Eventqualifier = 0x00F4 (0xFFFF8CB5) Unknown TextId	

Fig. 59: DiagHistory tab

The events are arranged according to type (information, warning, error), flag, occurrence of the event (time stamp) and message (port number & event code).

The meaning of the individual messages can be taken from the vendor documentation. The IO-Link device can be directly allocated on the basis of the port number. The events occurring can be managed using the various buttons.

- **Update History:** if the "Auto Update" field is not selected, then the current events can be displayed via the "Update History" button
- **Auto Update:** if this field is selected, then the list of events occurring is automatically updated
- **Only new Messages:** if this field is selected, then only those messages that have not yet been confirmed are displayed.
- **Ack. Messages:** an event that occurs is reported via the [Device Diag](#) [► 97]. Confirming the message will reset the bit to 0.
- **Export Diag History:** the events that have occurred can be exported as a "txt" file and thus archived.
- **Advanced:** this field currently (as at 3rd quarter 2015) has no function.

7.7 PLC library: Tc3_IoLink

The PLC library "Tc3_IoLink" is used for communication with IoLink devices.

Function blocks are available for this purpose that support the "Common Profile" and "Smart Sensor Profile" and enable parameters to be read and written.

See software documentation in the Beckhoff Information System:

[TwinCAT 3 | PLC Library: Tc3_IoLink](#)

8 TwinSAFE SC

8.1 TwinSAFE SC

8.1.1 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](#).

8.1.2 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.

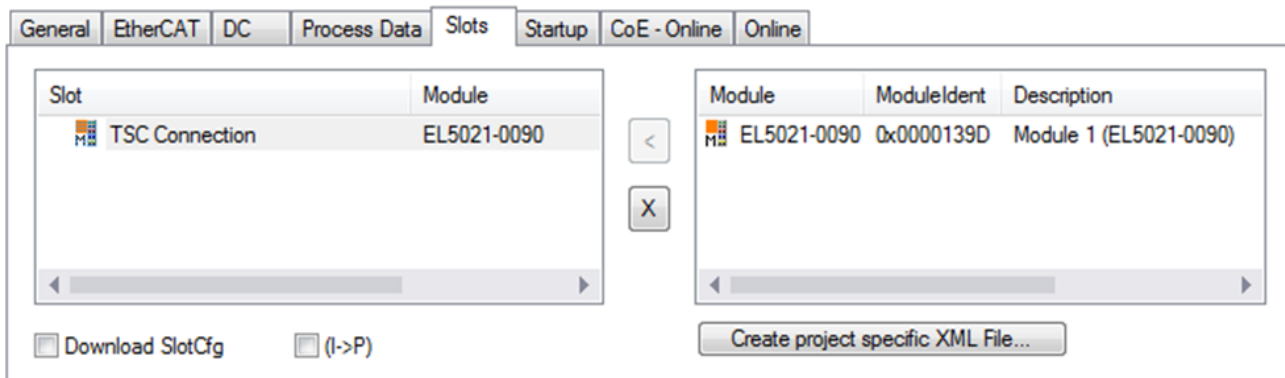


Fig. 60: Adding the TwinSAFE SC process data under the component, e.g. EL5021-0090

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

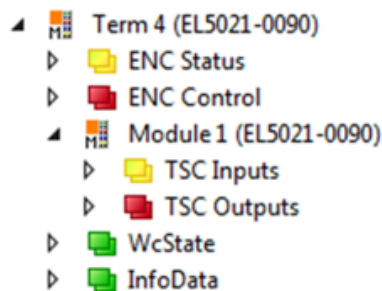


Fig. 61: TwinSAFE SC component process data, example EL5021-0090

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

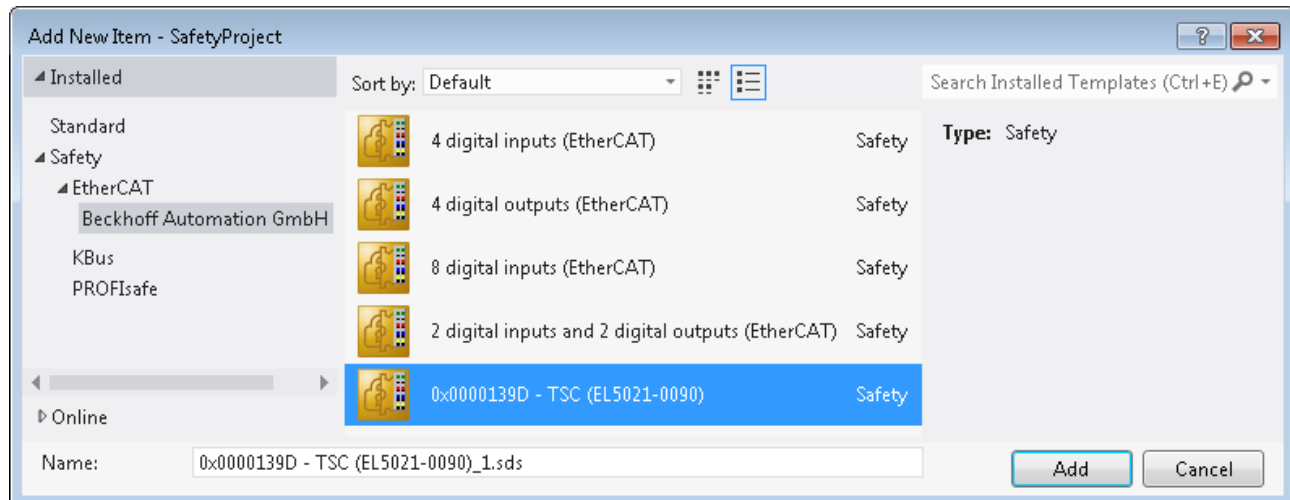


Fig. 62: Adding a TwinSAFE SC connection

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.

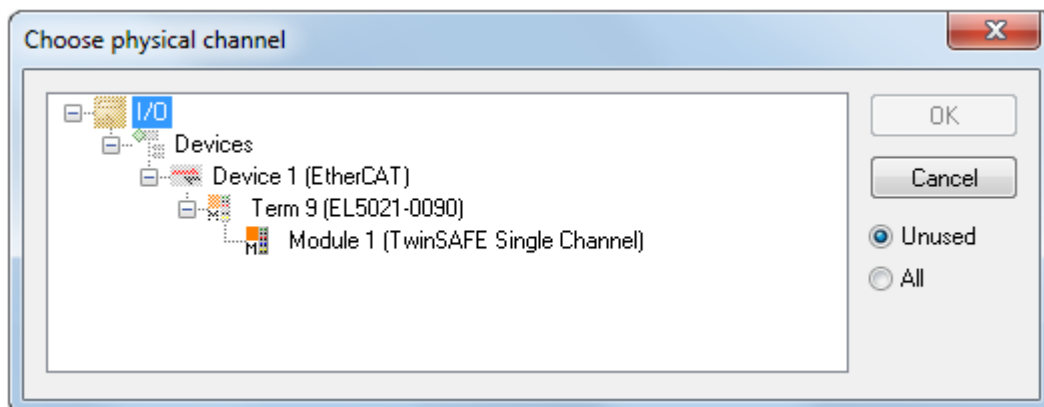


Fig. 63: Creating a link to TwinSAFE SC terminal

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

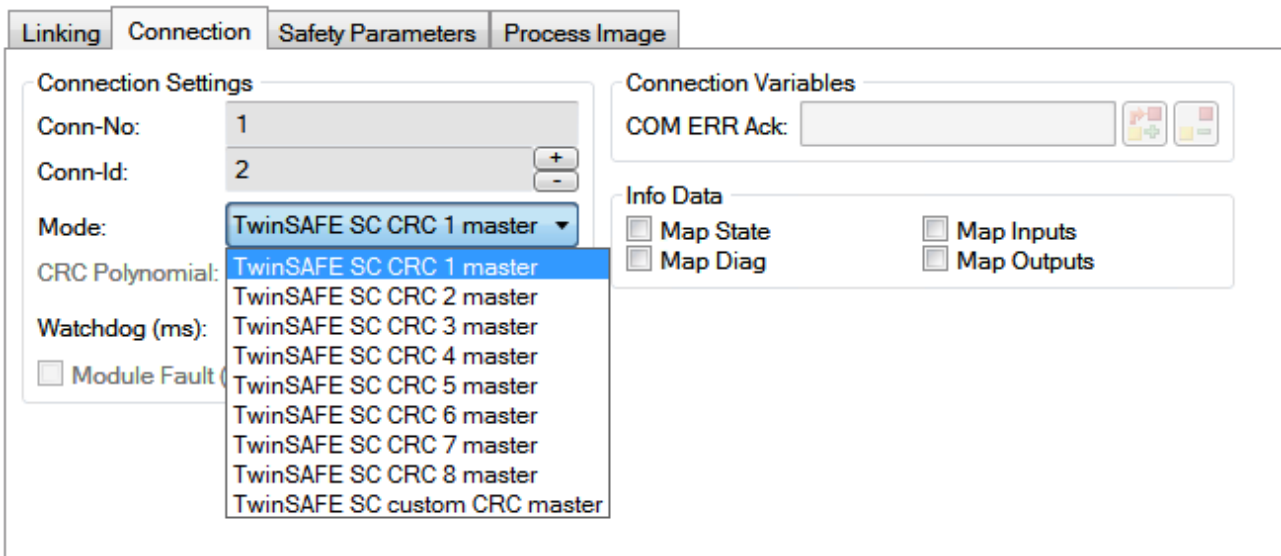


Fig. 64: Selecting a free CRC

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.

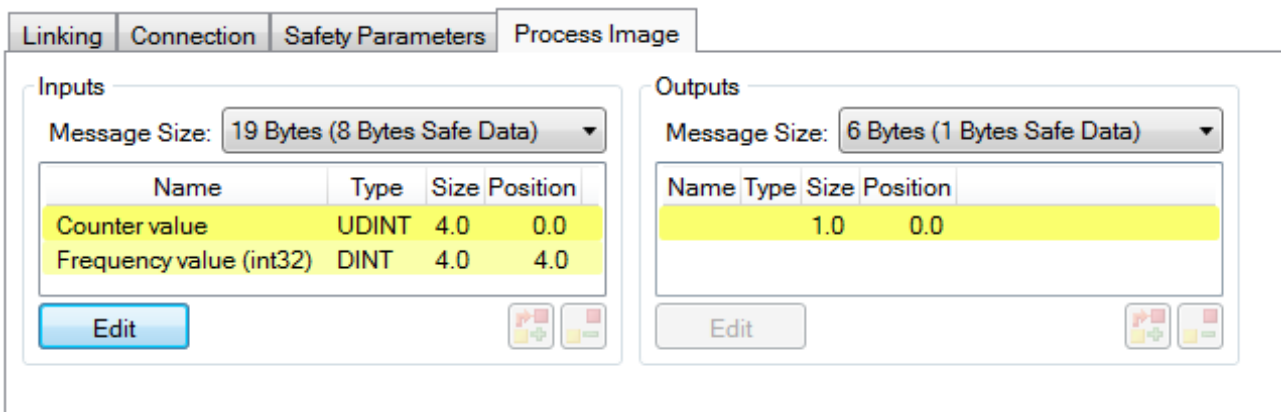


Fig. 65: Selecting the process data size and the process data

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)*.

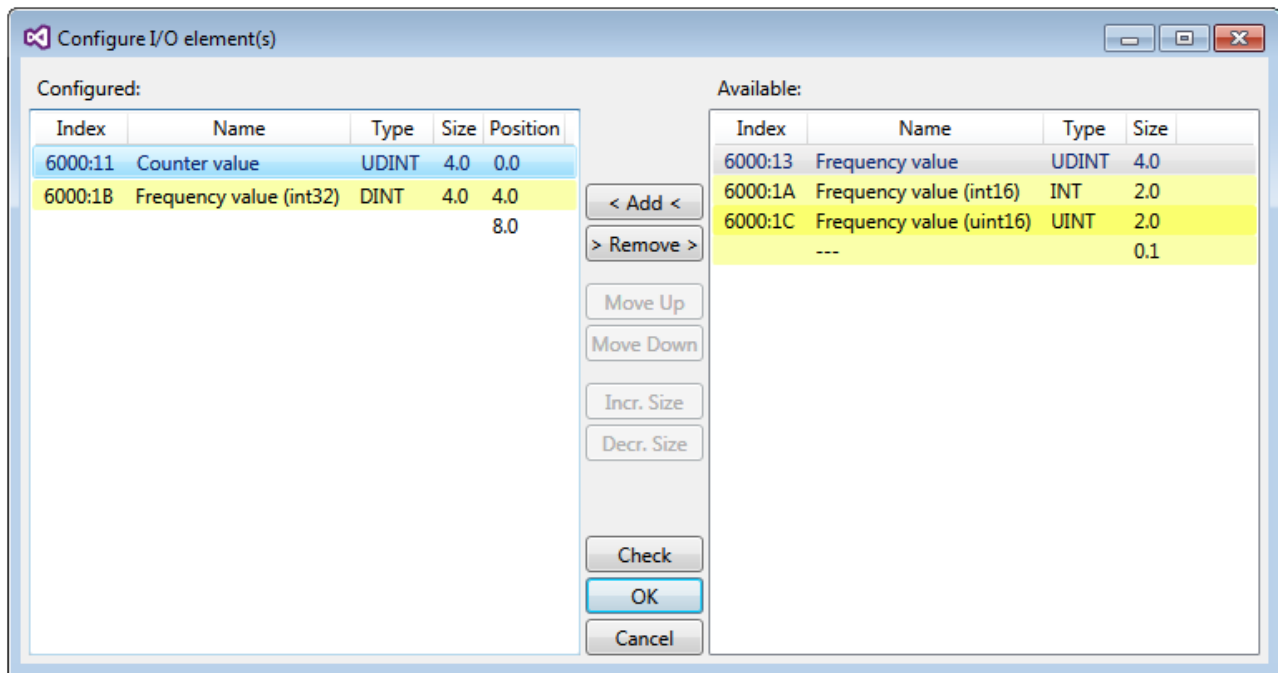


Fig. 66: Selection of the process data

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. 67: 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

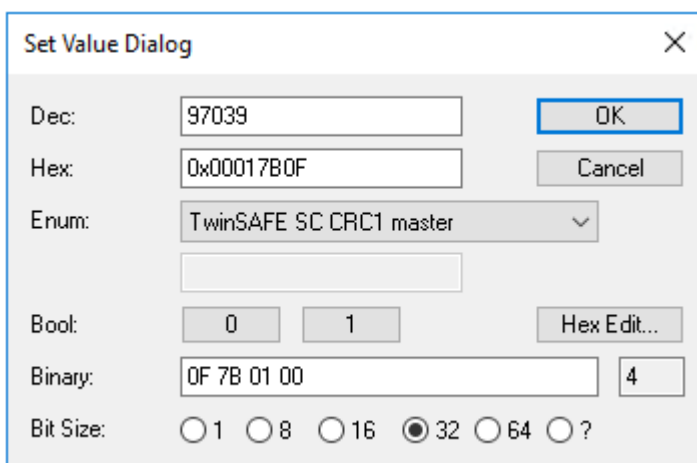


Fig. 68: Entering the safety address and the CRC



TwinSAFE SC connections

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

8.2 TwinSAFE SC process data EL6224-0090

The process data size and the process data themselves can be selected under the *Process Image* tab in TwinCAT. The process data depend on the connected IO-Link device and have to be adjusted accordingly. The corresponding information can be found in the sensor-specific data sheet. By default, a 16-bit unsigned integer is mapped with byte offset 0 from IO-Link channel 1.

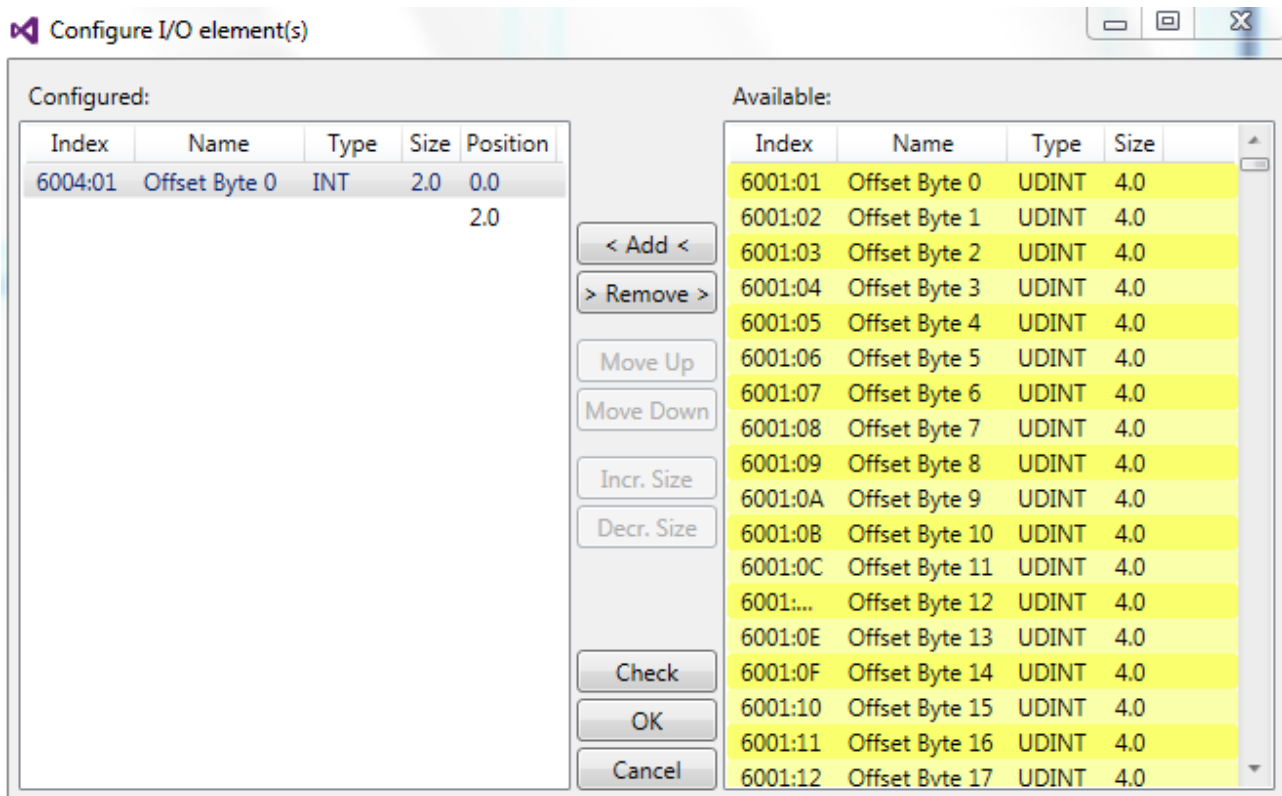


Fig. 69: EL6224-0090, process data image (default)

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

The following rule applies for process data mapping:

0x60ab:0c		
a = IO-Link port	b = data type integer	c = byte offset
0 = IO-Link port 1	0 = data type bit	01 = offset byte 0
1 = IO-Link port 2	1 = UDINT	02 = offset byte 1
2 = IO-Link port 3	2 = DINT	03 = offset byte 2
3 = IO-Link port 4	3 = UINT	
	4 = INT	0p = offset byte p-1

Examples for index mapping:

0x6004:01 => IO-Link port 1, data type INT, offset byte 0

0x6011:01 => IO-Link port 2, data type UDINT, offset byte 0



TwinSAFE SC objects

The TwinSAFE SC objects of the EL6224-0090 are listed in the chapter [Objects TwinSAFE Single Channel \(EL6224-0090\) \[► 93\]](#).

9 Configuration with TwinCAT

9.1 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:

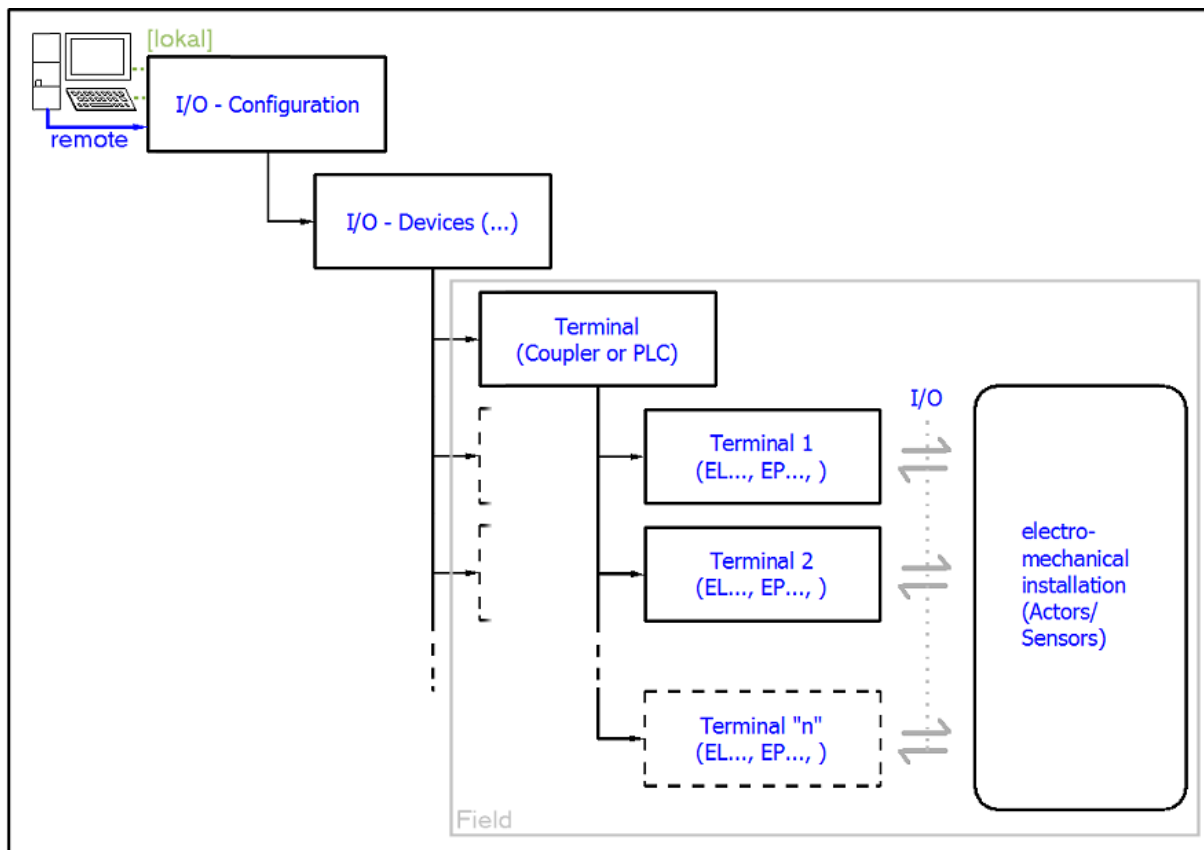


Fig. 70: Relationship between user side (commissioning) and installation

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)

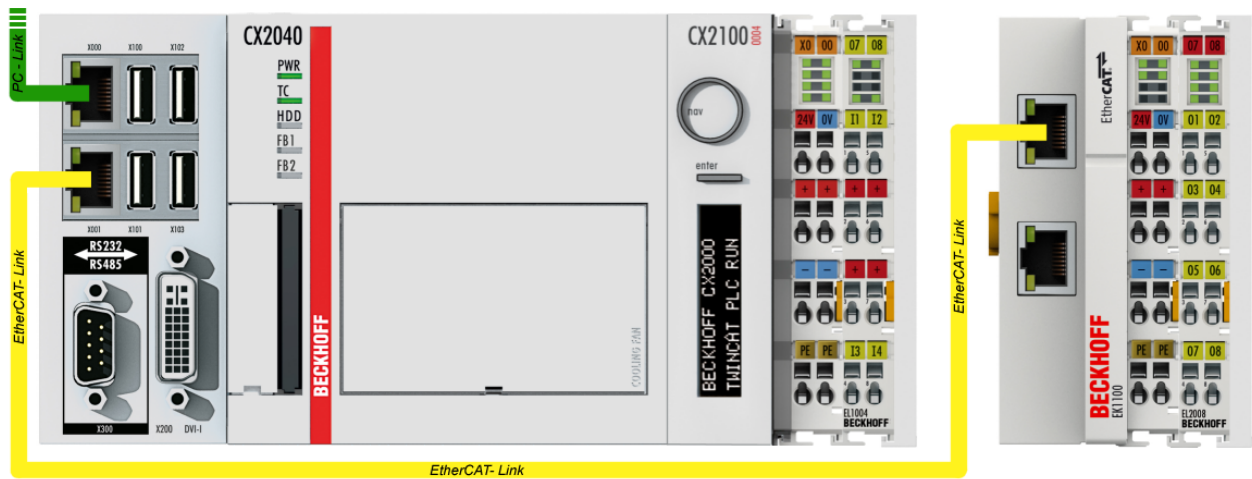


Fig. 71: Control configuration with Embedded PC, input (EL1004) and output (EL2008)

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.

9.1.1 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:

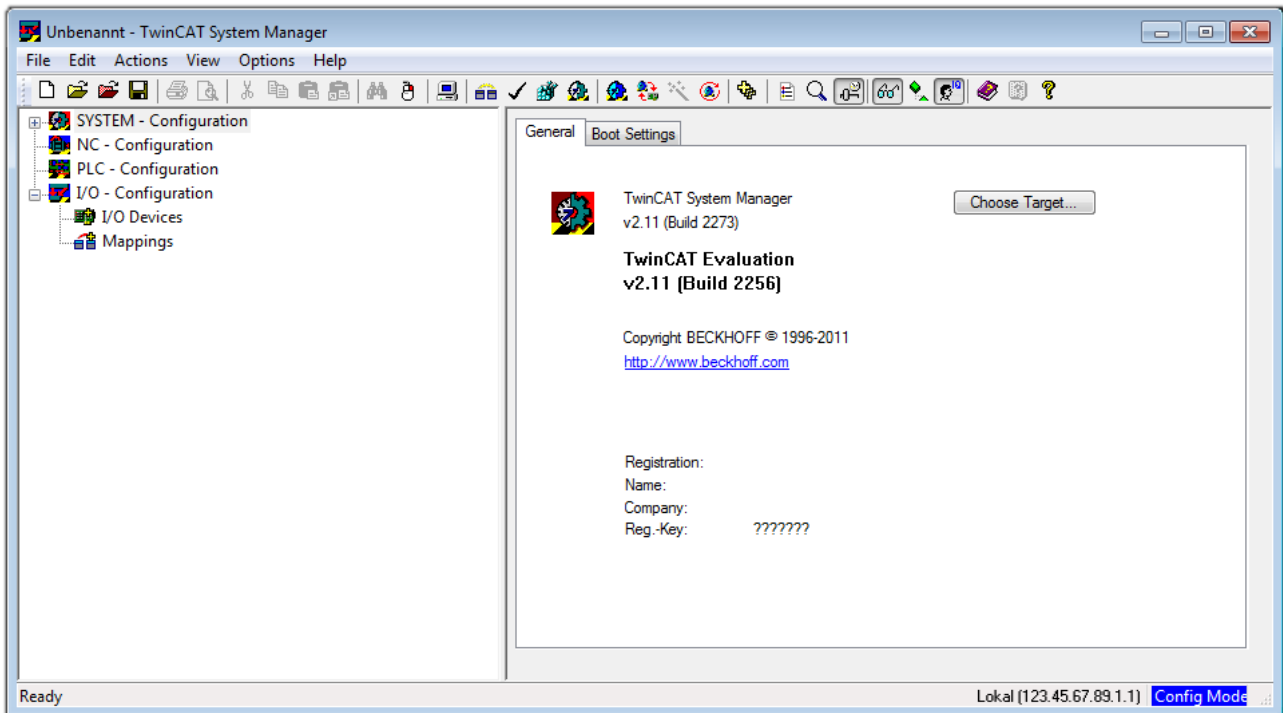


Fig. 72: Initial TwinCAT 2 user interface

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](#) [► 113]”.

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:

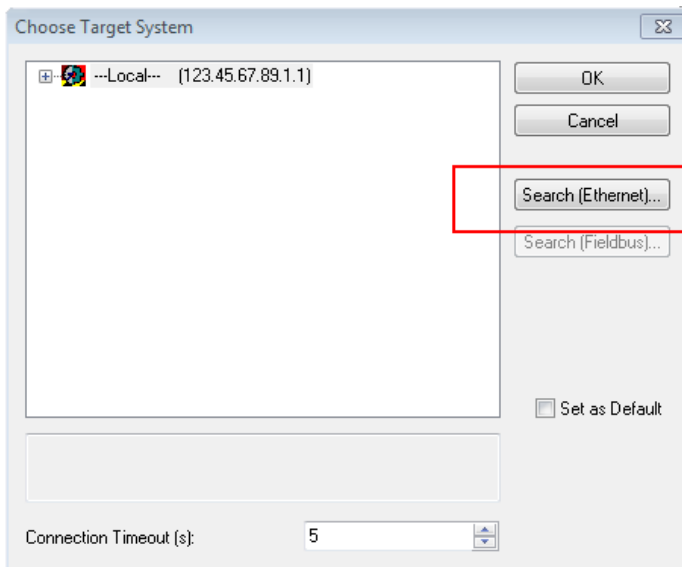


Fig. 73: Selection of the target system

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

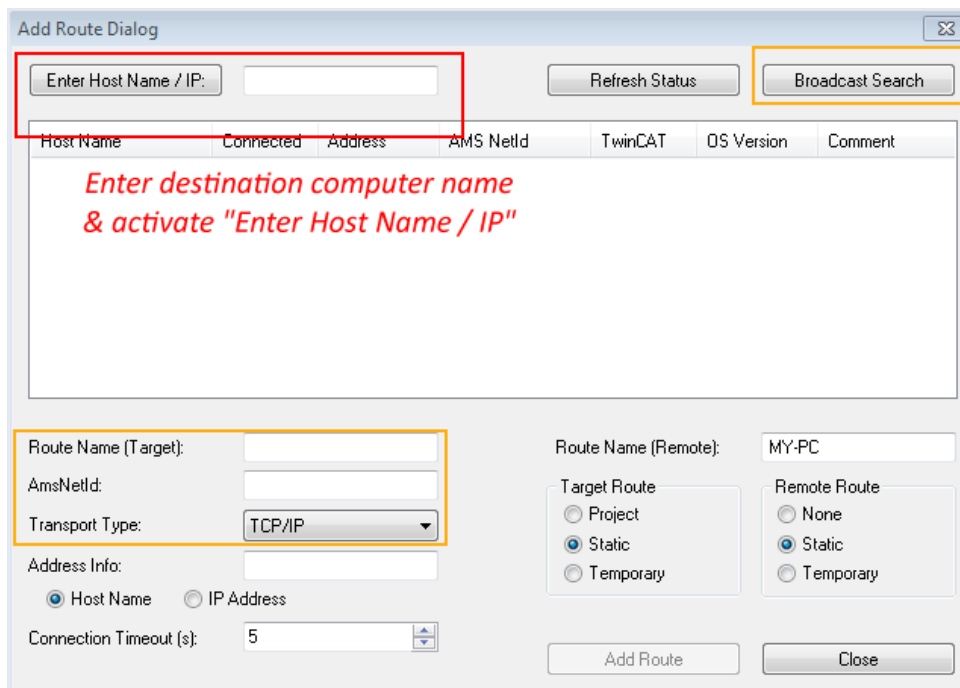
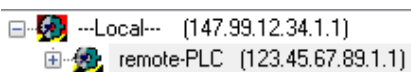


Fig. 74: specify the PLC for access by the TwinCAT System Manager: selection of the target system

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).

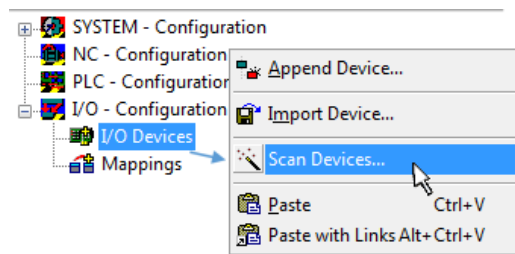


Fig. 75: Select “Scan Devices...”

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

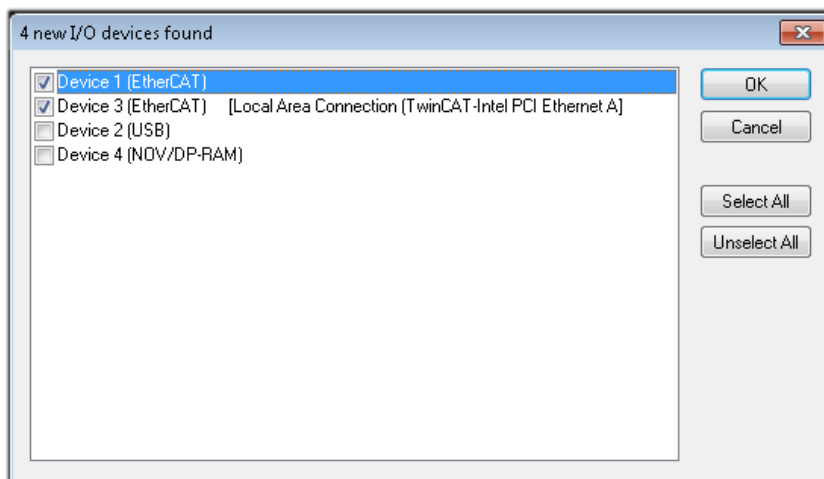


Fig. 76: Automatic detection of I/O devices: selection of the devices to be integrated

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](#) [► 109] described at the beginning of this section, the result is as follows:

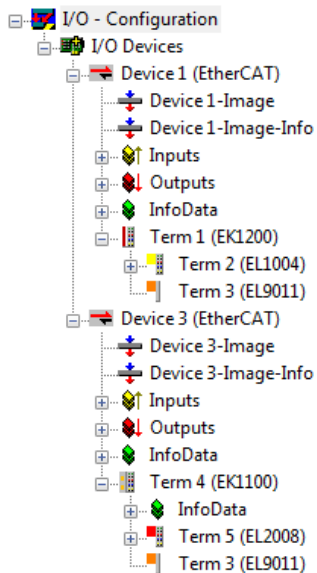


Fig. 77: Mapping of the configuration in the TwinCAT 2 System Manager

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:

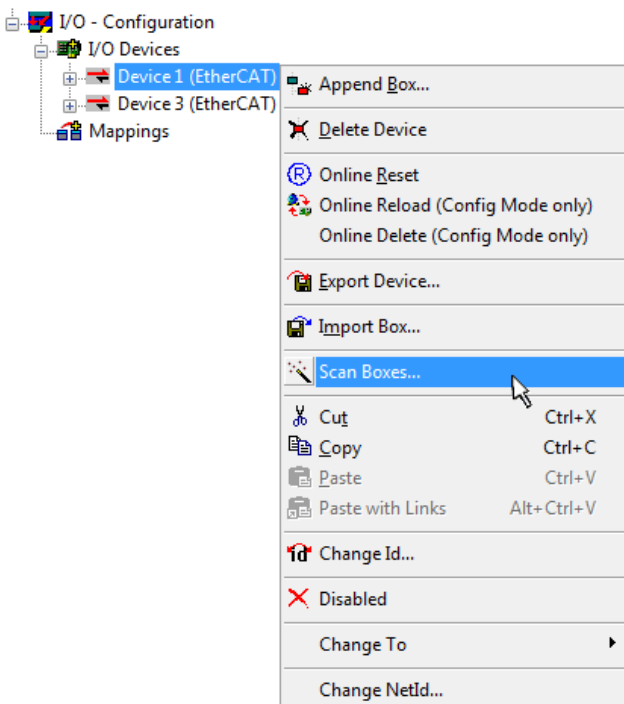


Fig. 78: Reading of individual terminals connected to a device

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:

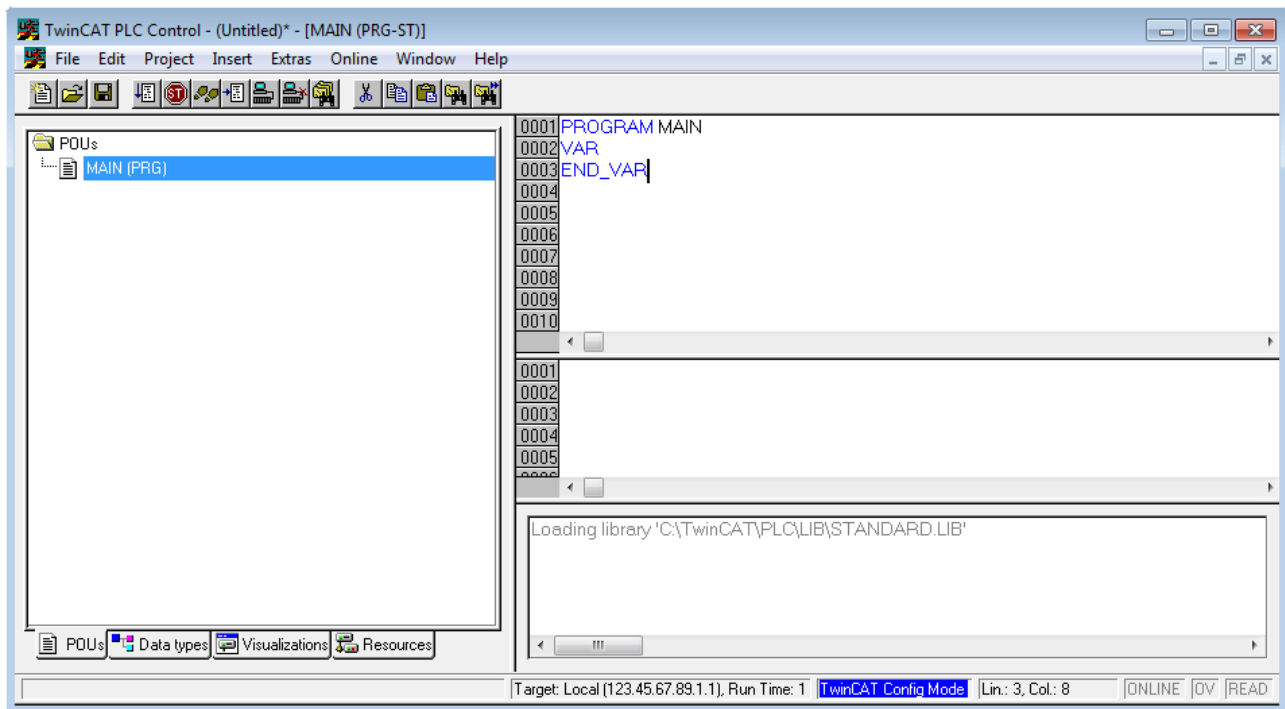


Fig. 79: TwinCAT PLC Control after startup

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

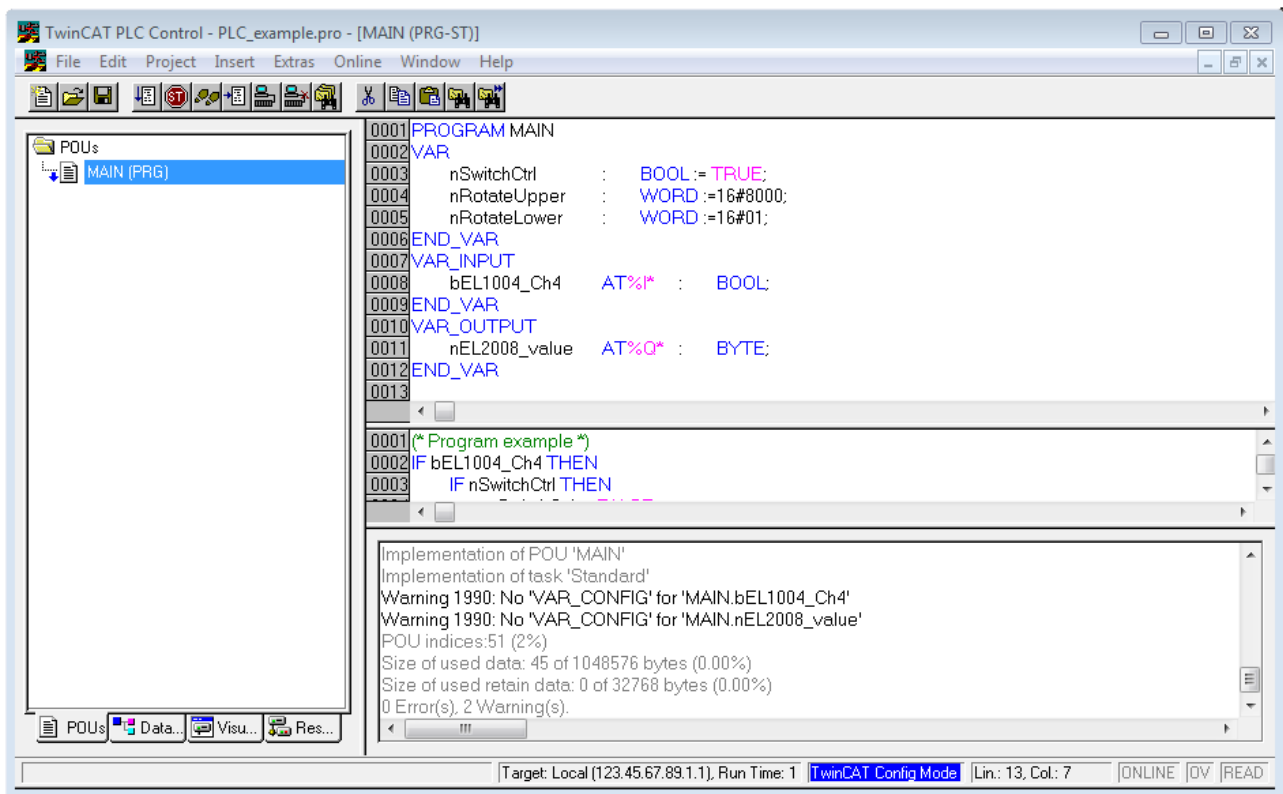


Fig. 80: Example program with variables after a compile process (without variable integration)

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...”:

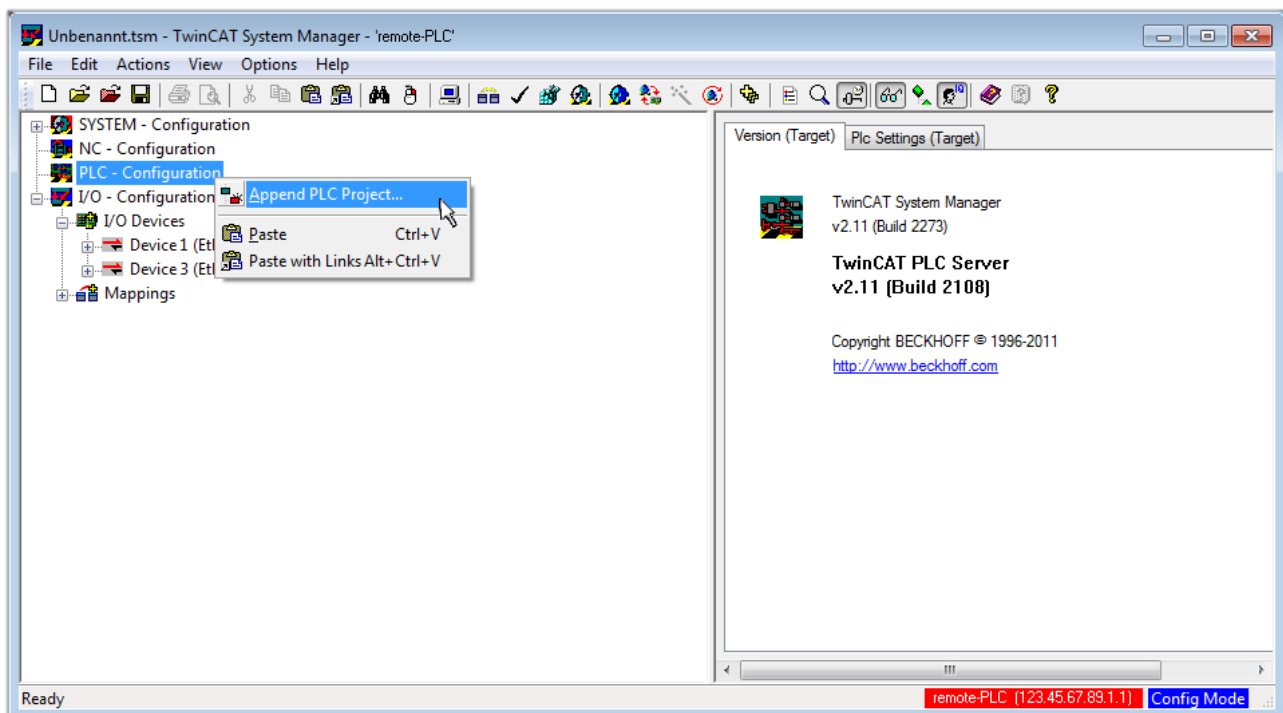


Fig. 81: Appending the TwinCAT PLC Control project

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:

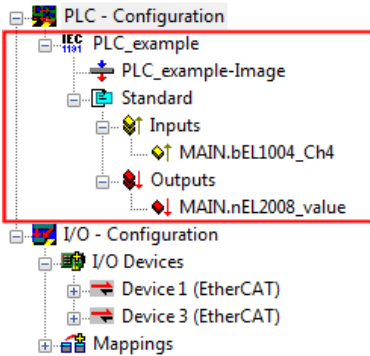


Fig. 82: PLC project integrated in the PLC configuration of the System Manager

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”:

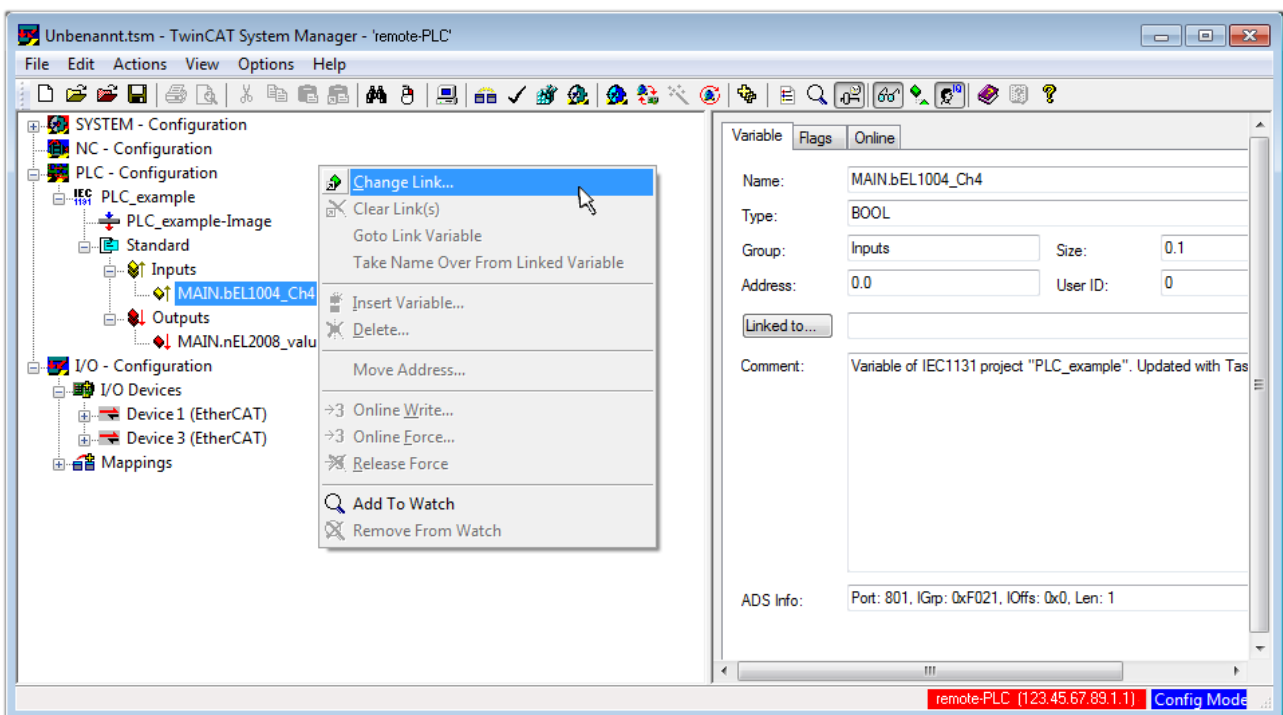


Fig. 83: Creating the links between PLC variables and process objects

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

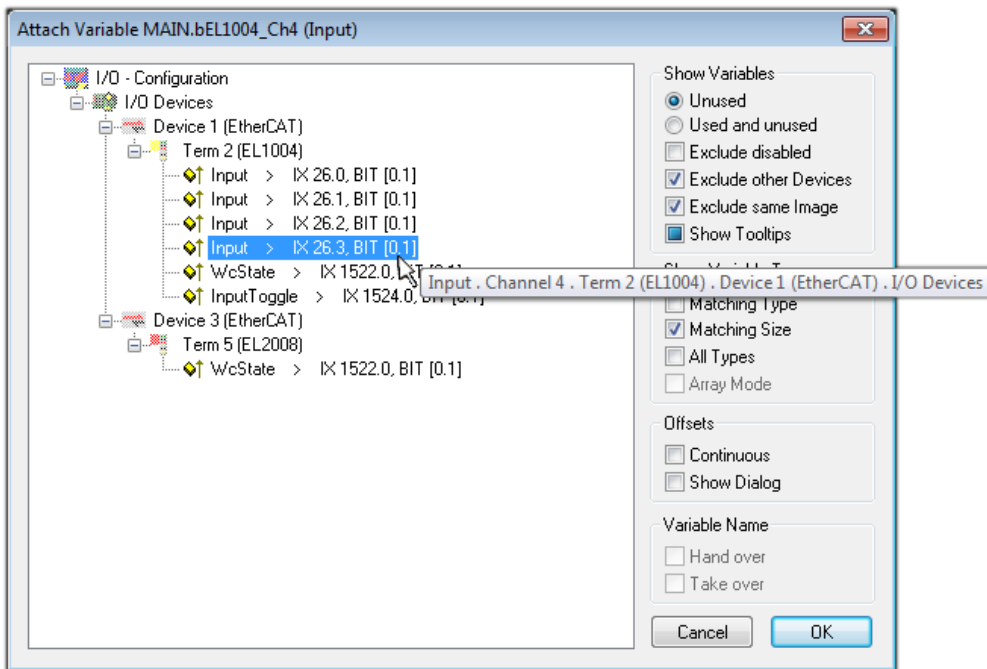


Fig. 84: Selecting BOOL-type PDO

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:

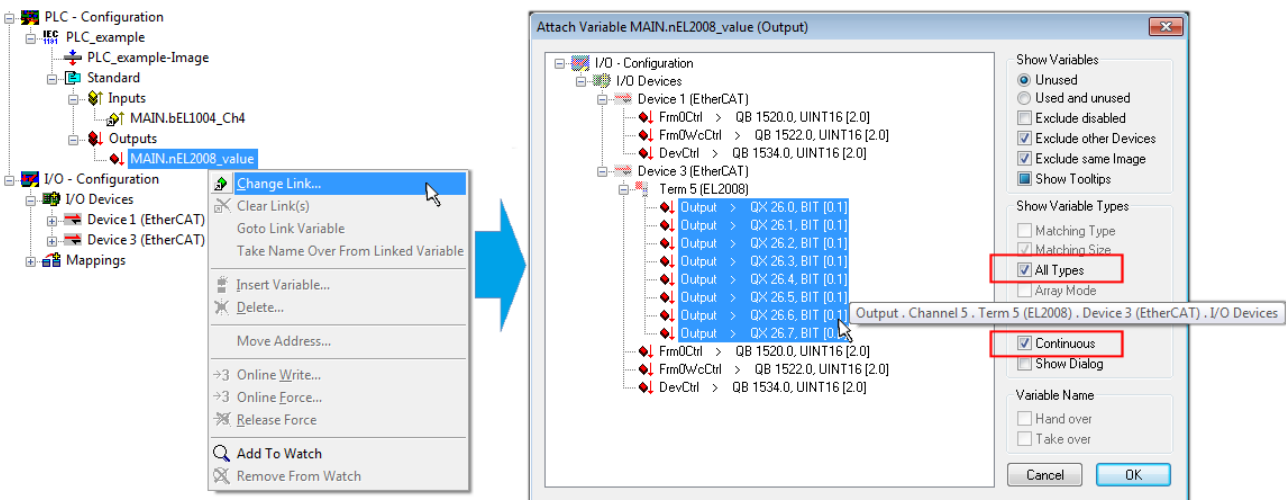


Fig. 85: Selecting several PDOs simultaneously: activate “Continuous” and “All types”

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:

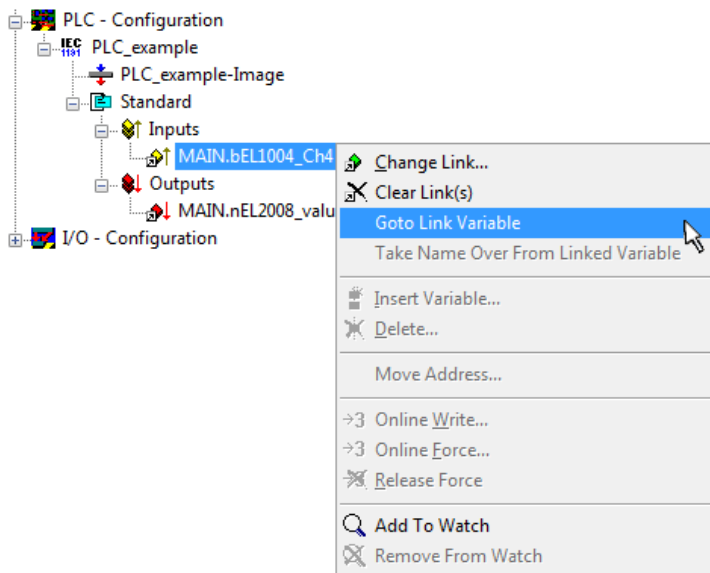
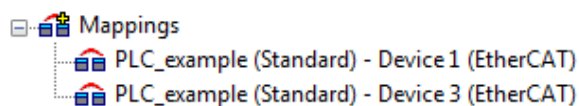


Fig. 86: Application of a “Goto Link Variable”, using “MAIN.bEL1004_Ch4” as an example

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 **RTime 0%** 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...”:

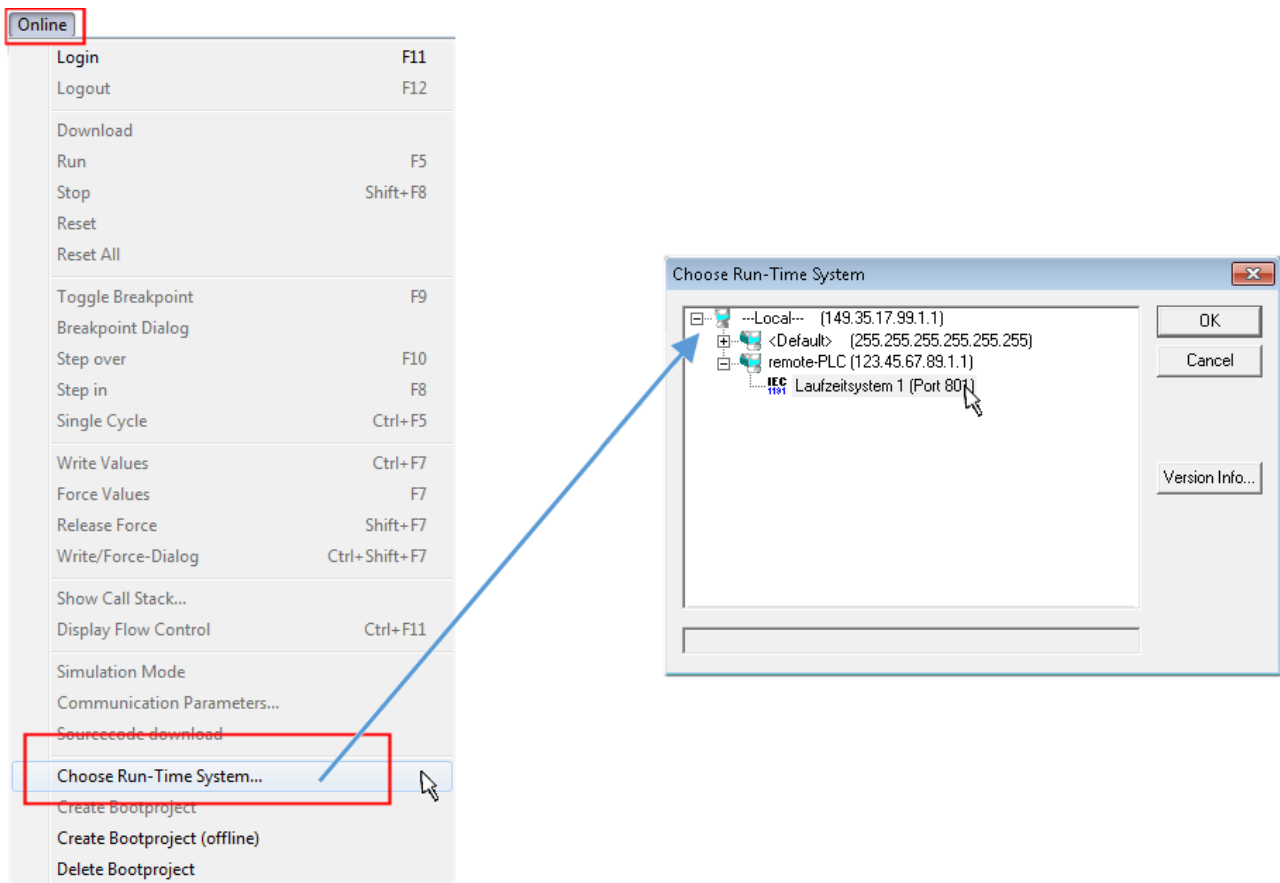


Fig. 87: Choose target system (remote)

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:

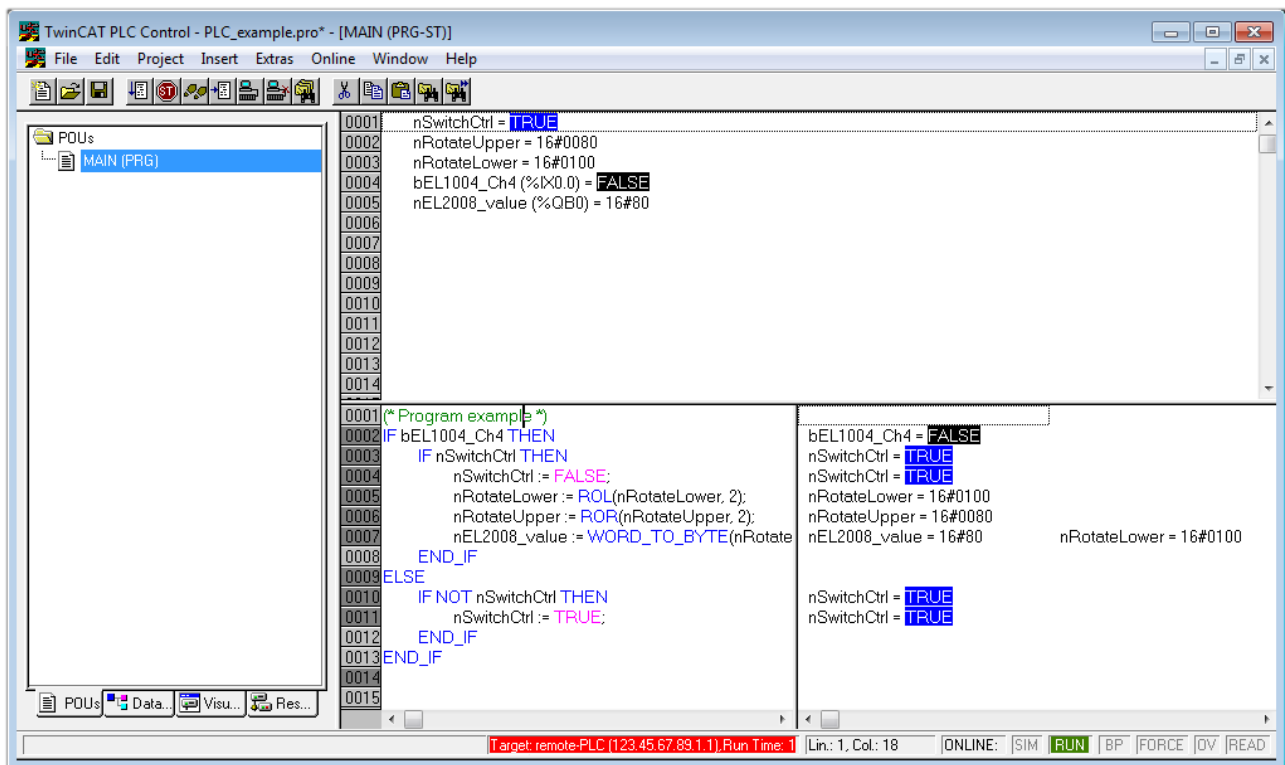


Fig. 88: PLC Control logged in, ready for program startup

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

9.1.2 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:

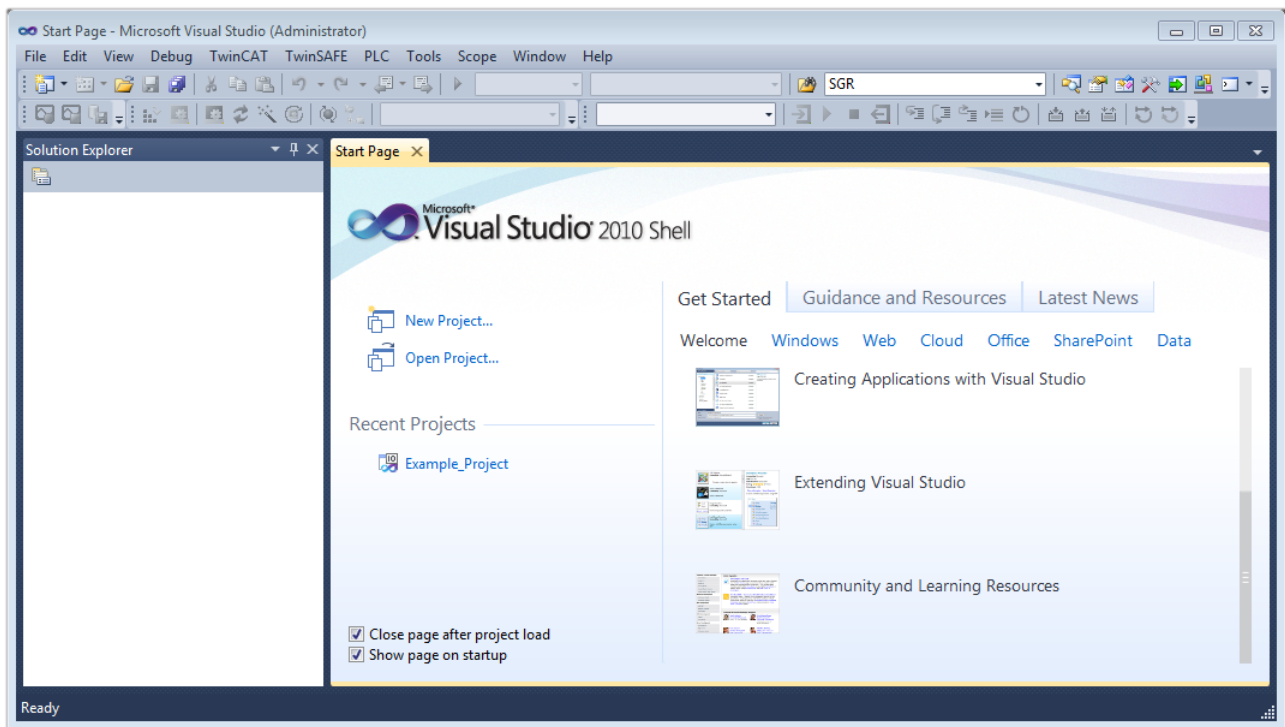


Fig. 89: Initial TwinCAT 3 user interface

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):

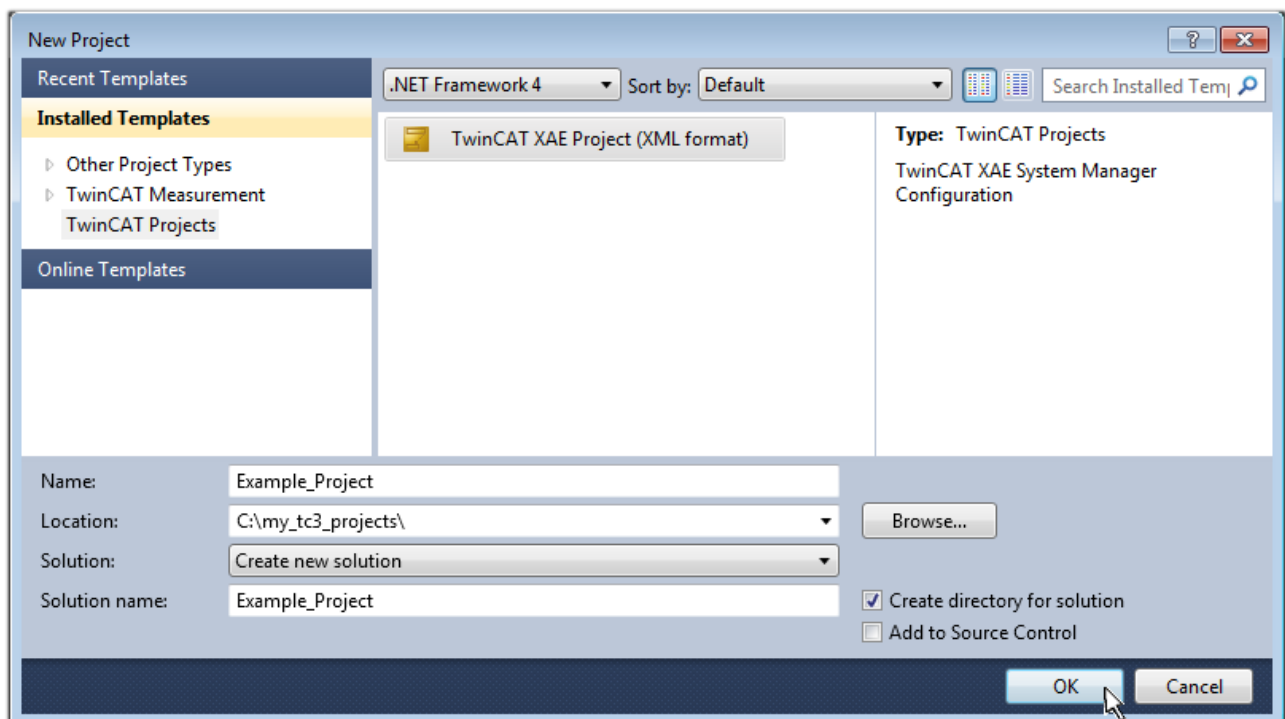


Fig. 90: Create new TwinCAT 3 project

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

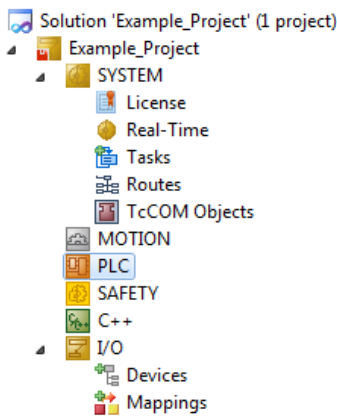
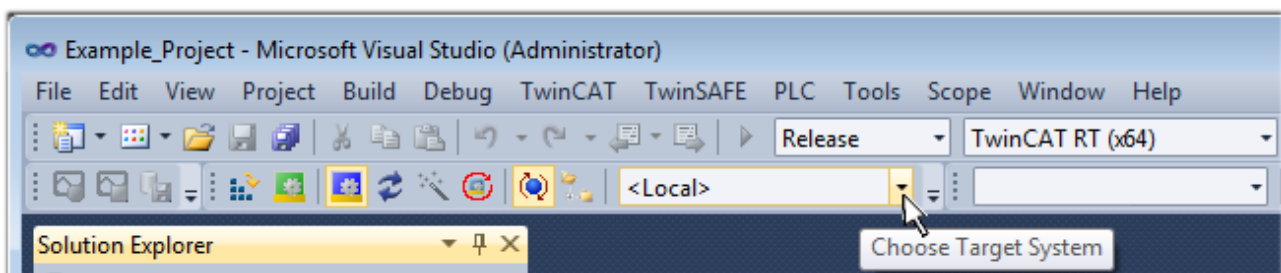


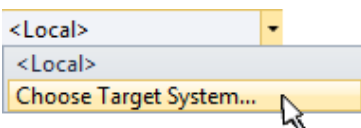
Fig. 91: New TwinCAT 3 project in the project folder explorer

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 |> 124|](#)”.

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:

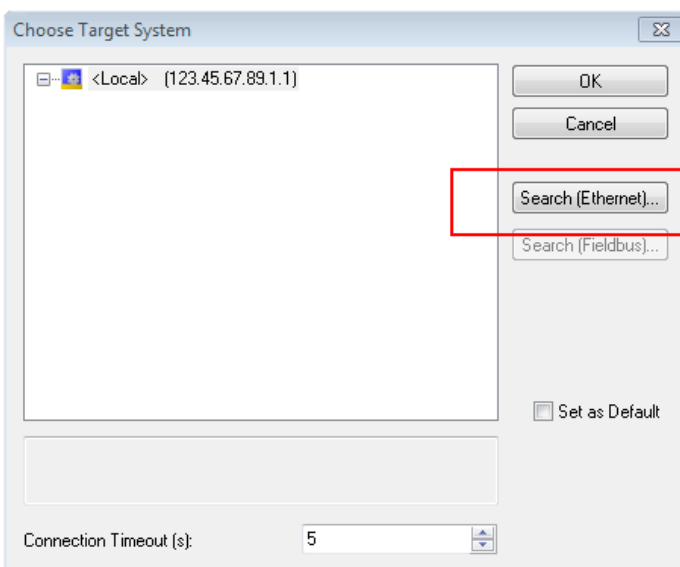


Fig. 92: Selection dialog: Choose the target system

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

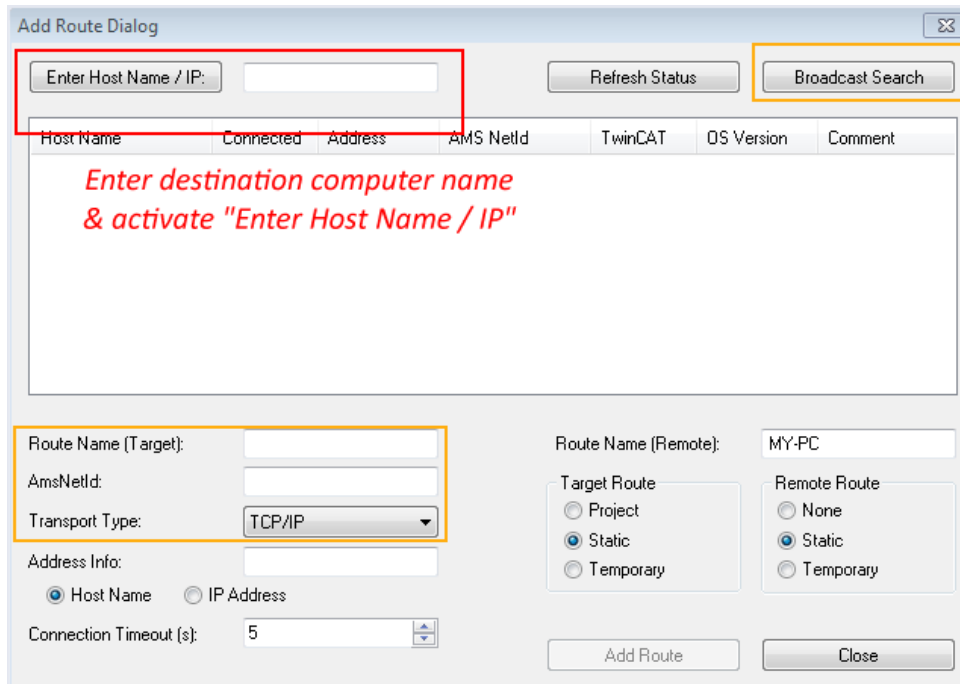
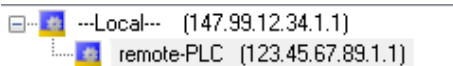


Fig. 93: specify the PLC for access by the TwinCAT System Manager: selection of the target system

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)”.

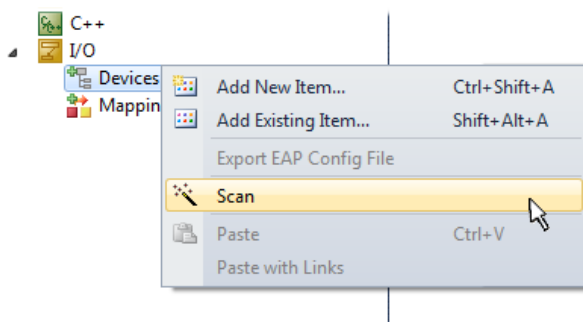


Fig. 94: Select “Scan”

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

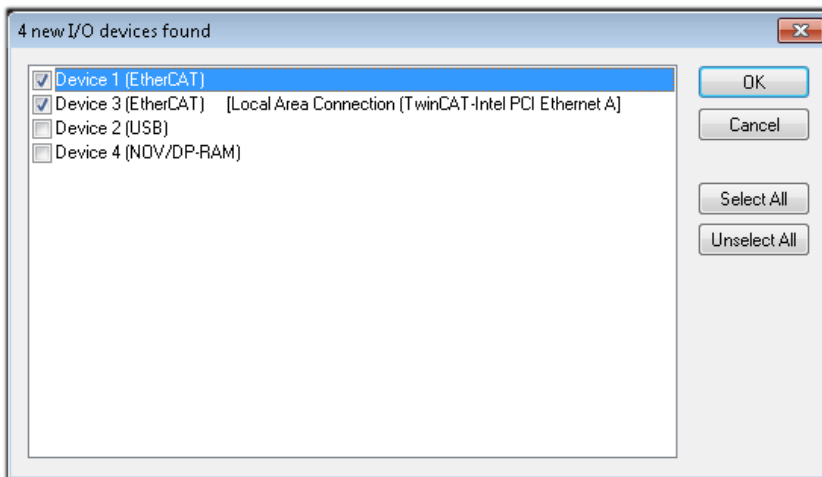


Fig. 95: Automatic detection of I/O devices: selection of the devices to be integrated

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 \[► 109\]](#) described at the beginning of this section, the result is as follows:

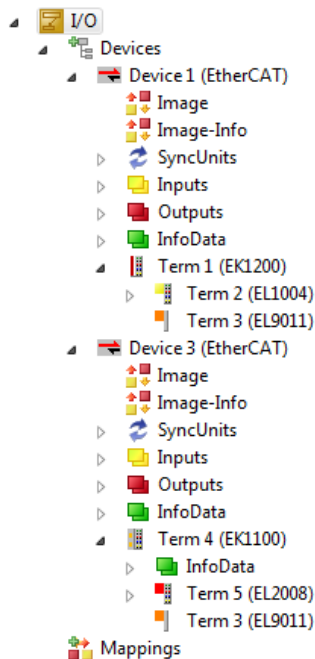


Fig. 96: Mapping of the configuration in VS shell of the TwinCAT 3 environment

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:

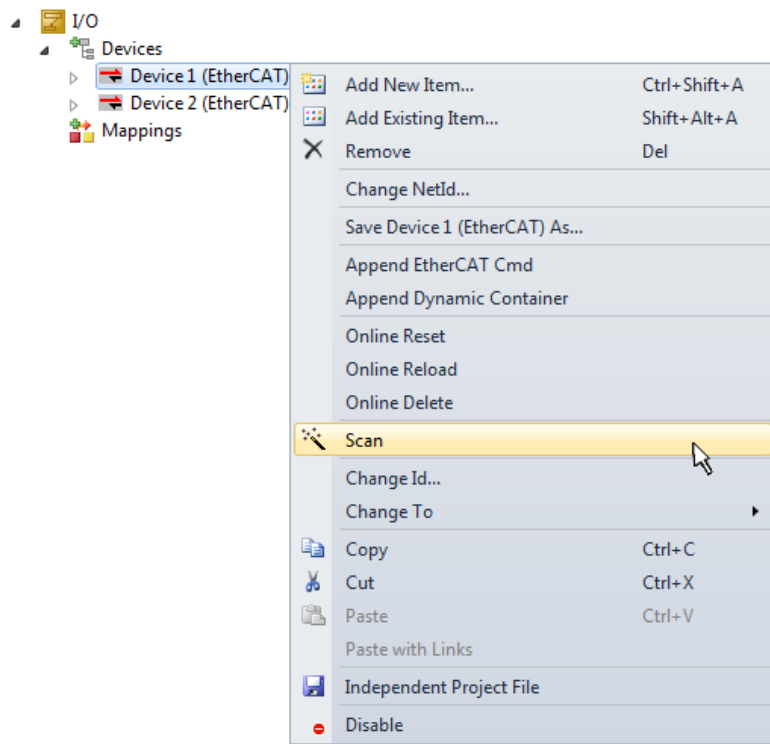


Fig. 97: Reading of individual terminals connected to a device

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....":

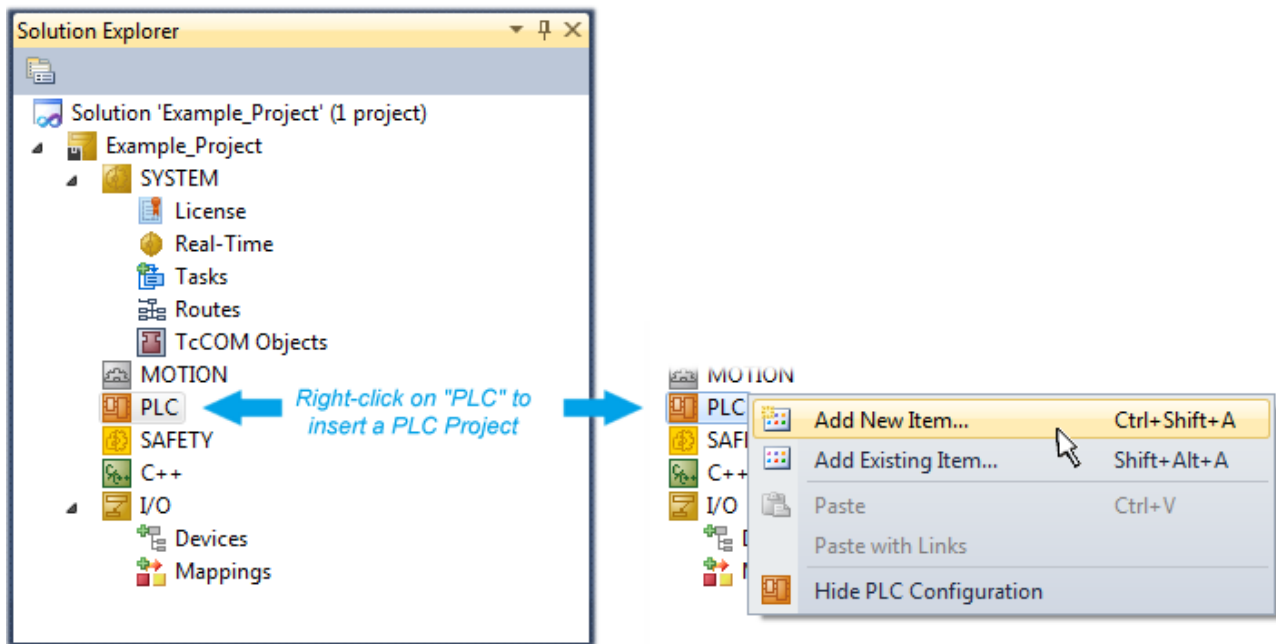


Fig. 98: Adding the programming environment in "PLC"

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

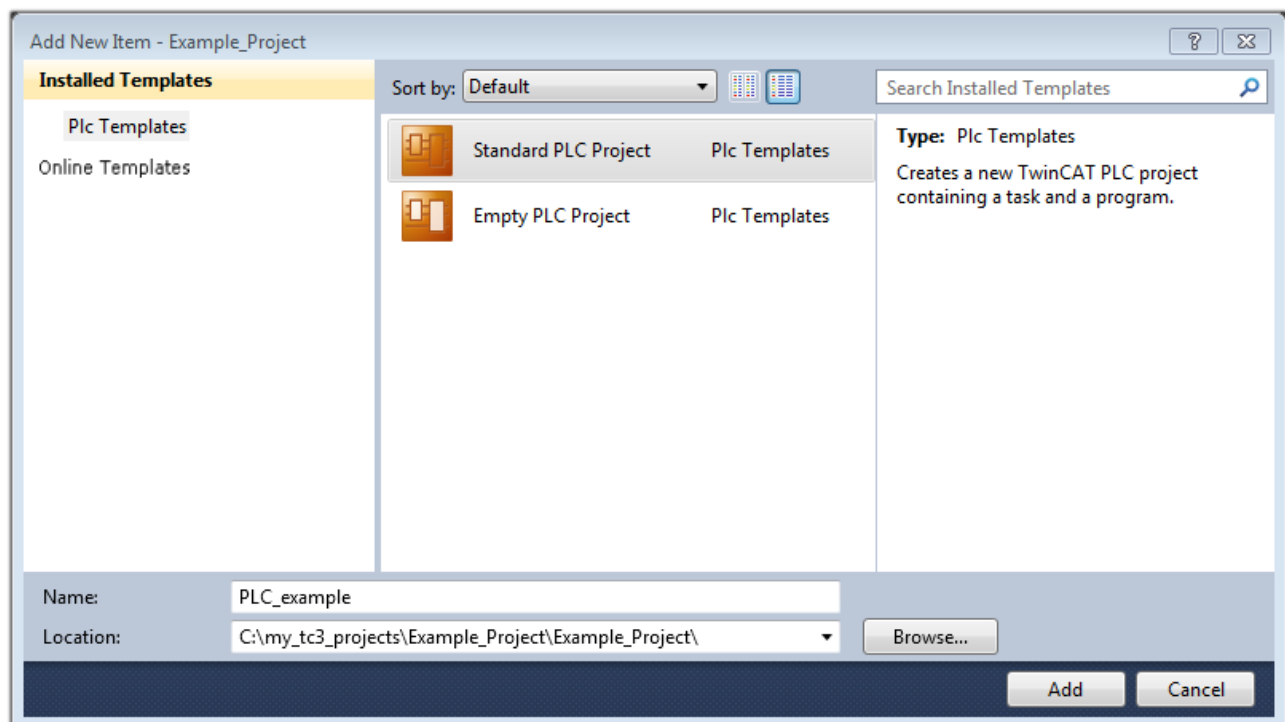


Fig. 99: Specifying the name and directory for the PLC programming environment

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:

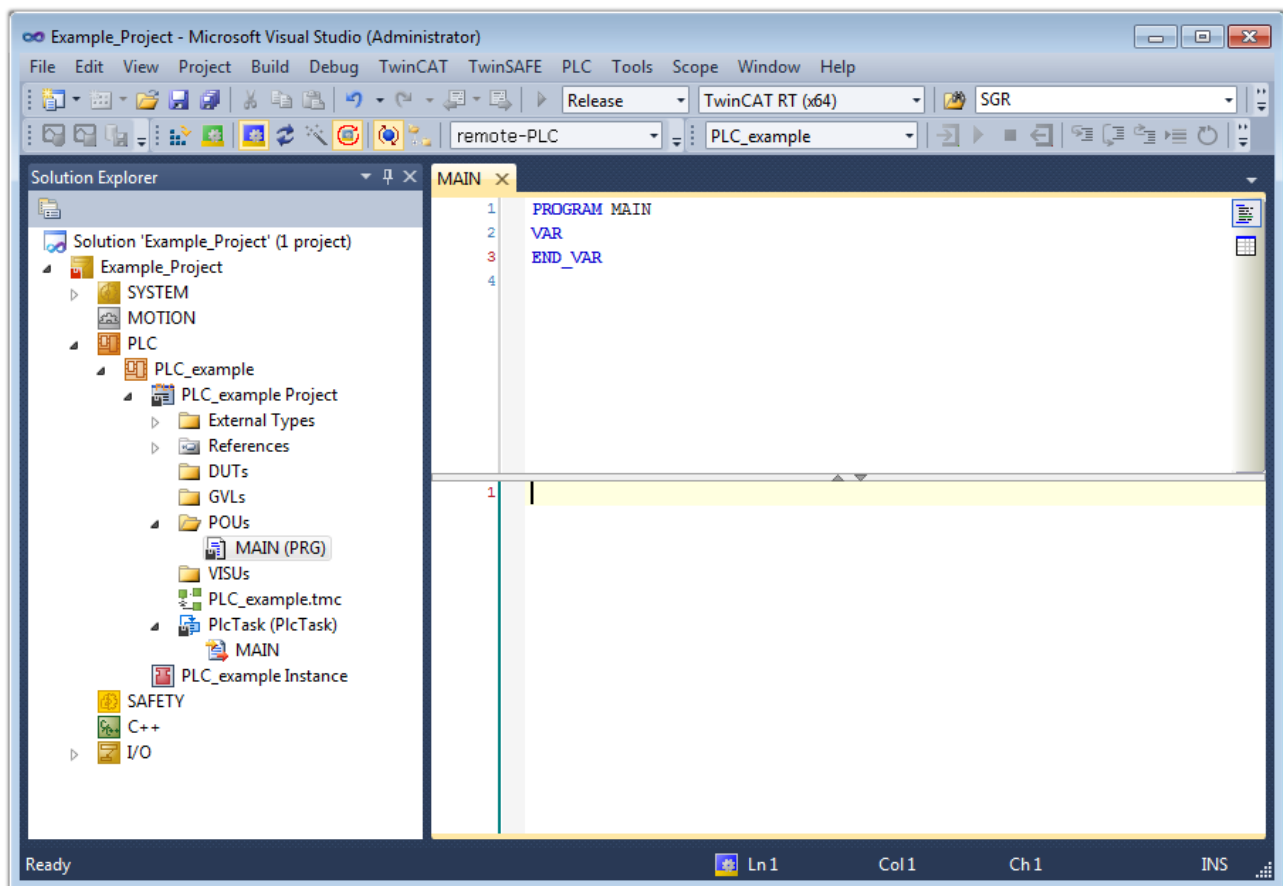


Fig. 100: Initial “Main” program for the standard PLC project

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

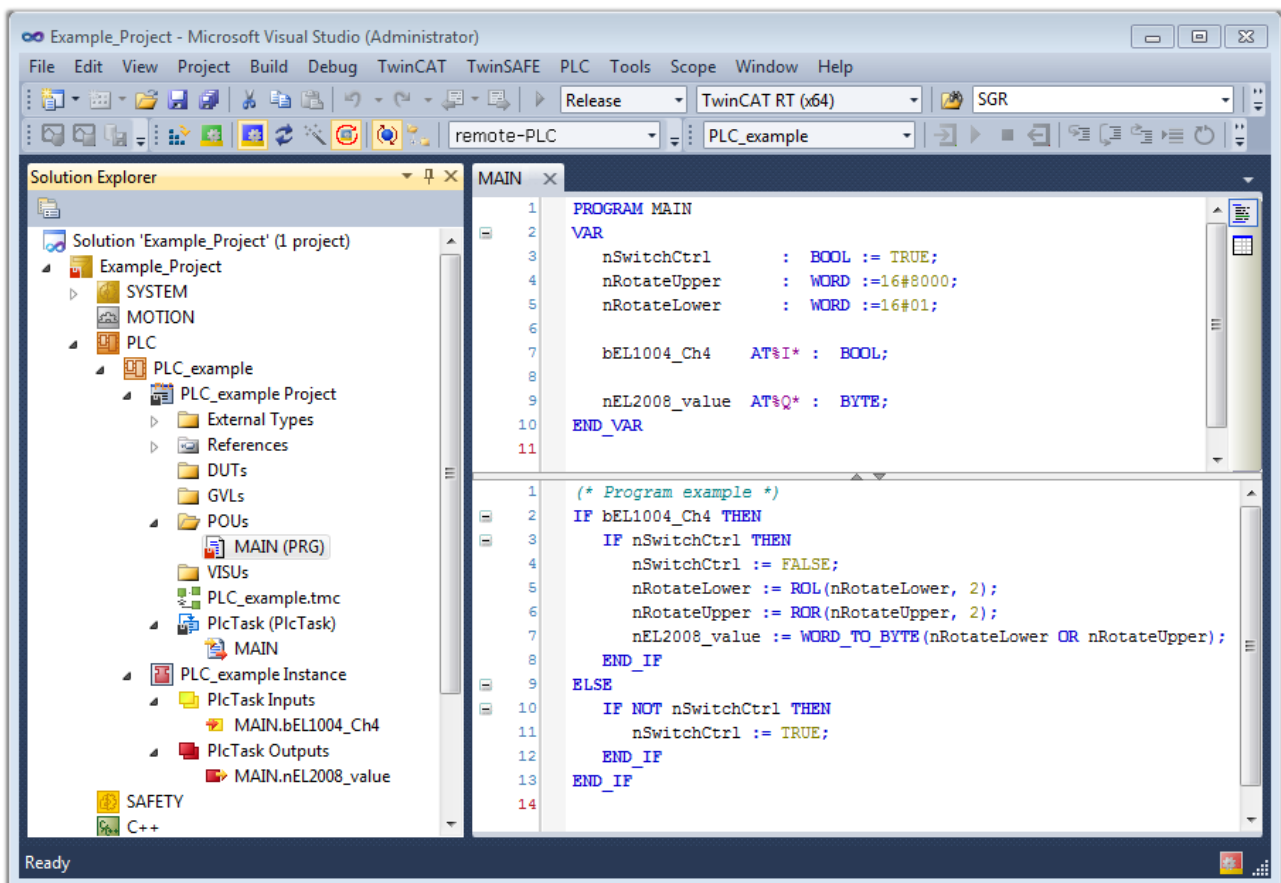


Fig. 101: Example program with variables after a compile process (without variable integration)

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

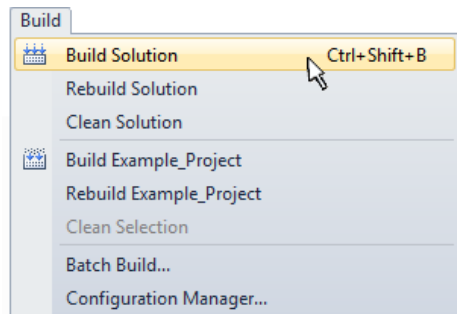
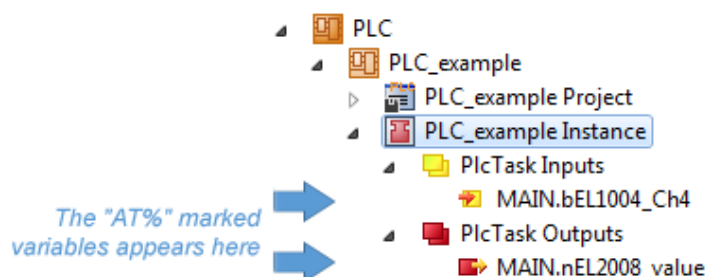


Fig. 102: Start program compilation

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:

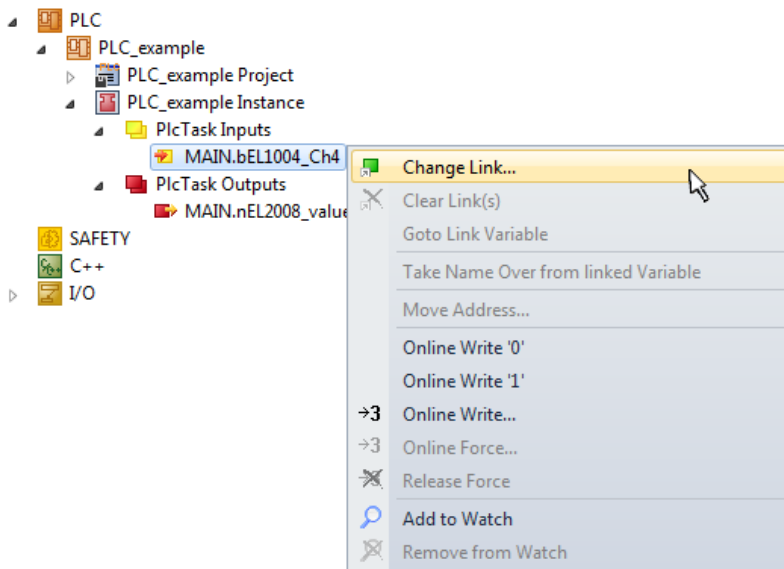


Fig. 103: Creating the links between PLC variables and process objects

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

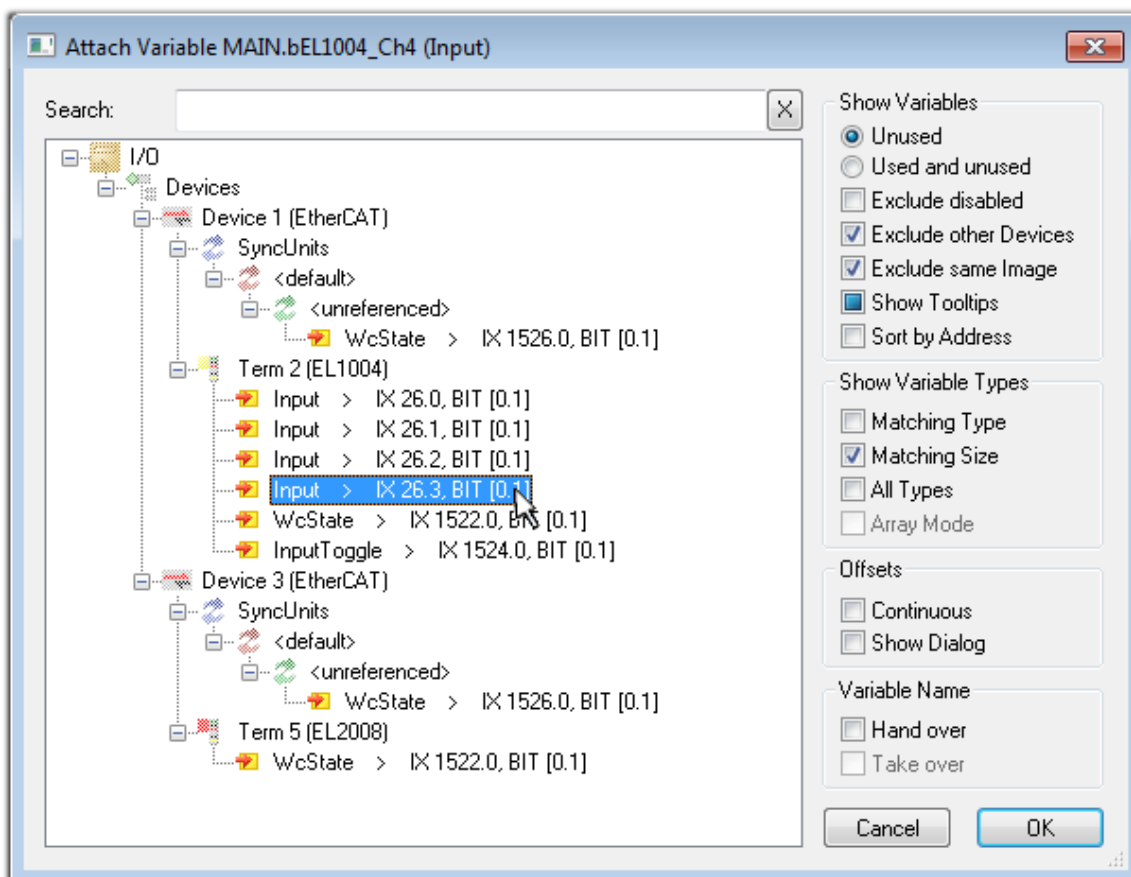


Fig. 104: Selecting BOOL-type PDO

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:

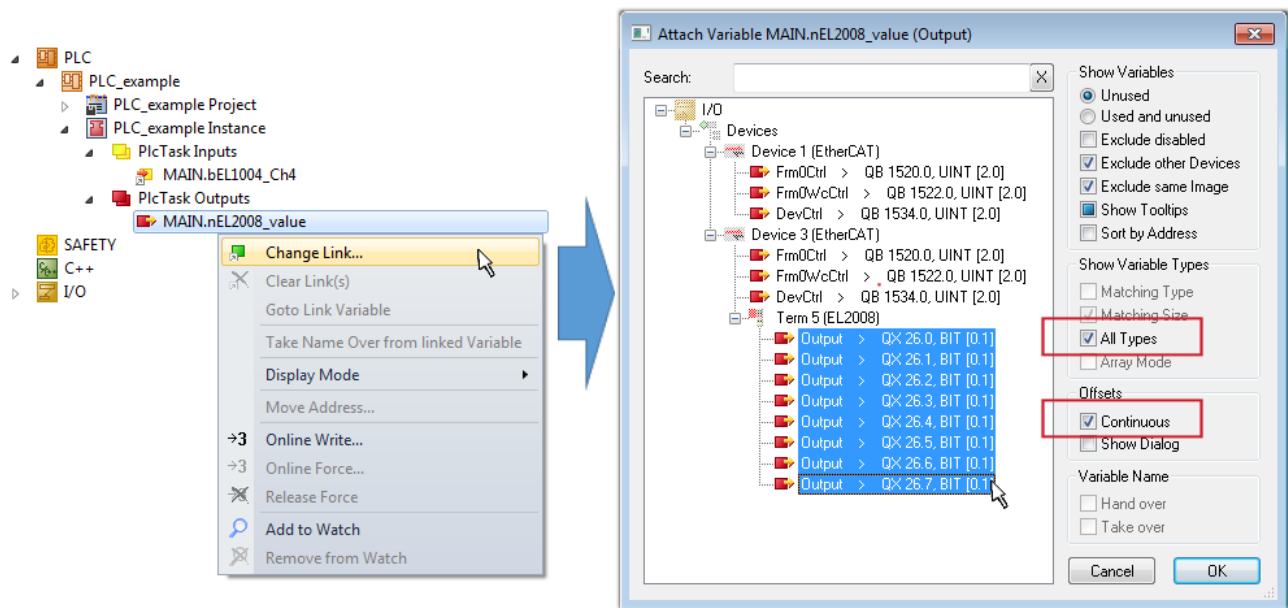


Fig. 105: Selecting several PDOs simultaneously: activate “Continuous” and “All types”

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:

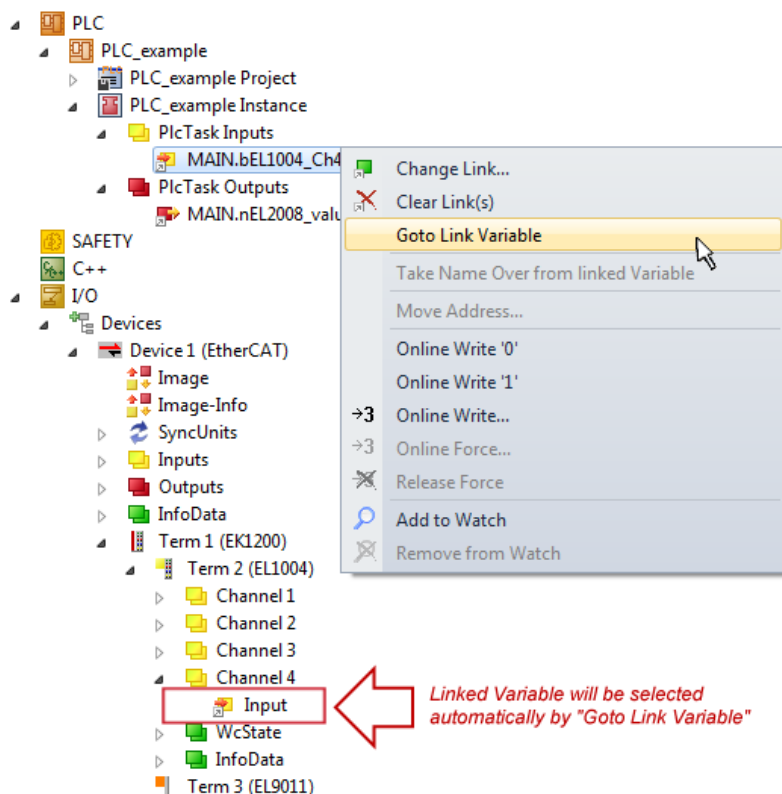


Fig. 106: Application of a “Goto Link Variable”, using “MAIN.bEL1004_Ch4” as an example

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

1 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.

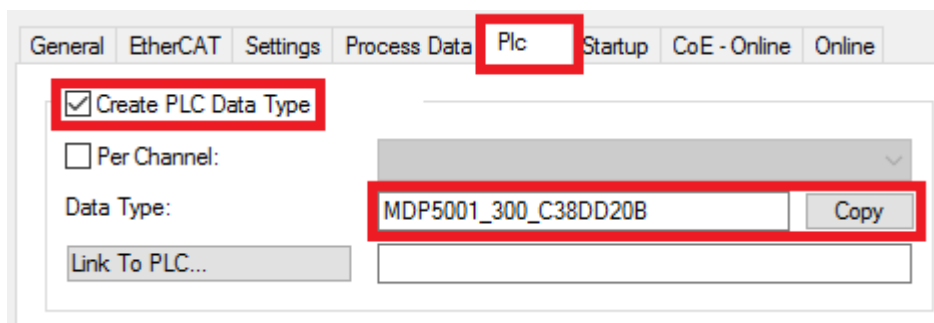


Fig. 107: Creating a PLC data type

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

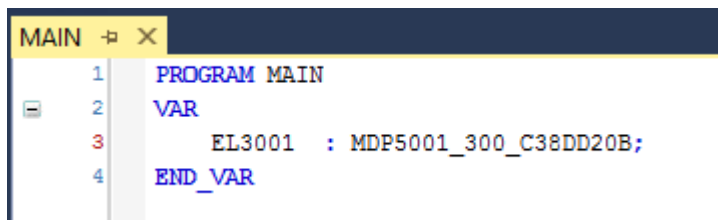


Fig. 108: Instance_of_struct

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.

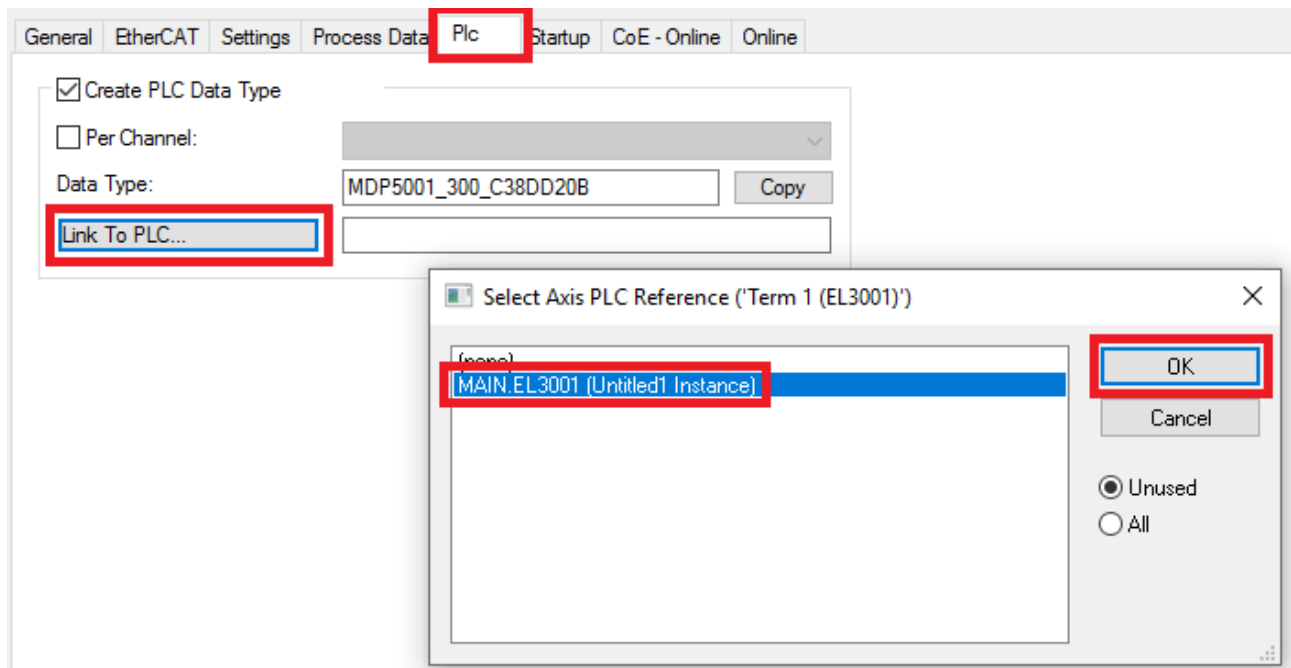


Fig. 109: Linking the structure

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

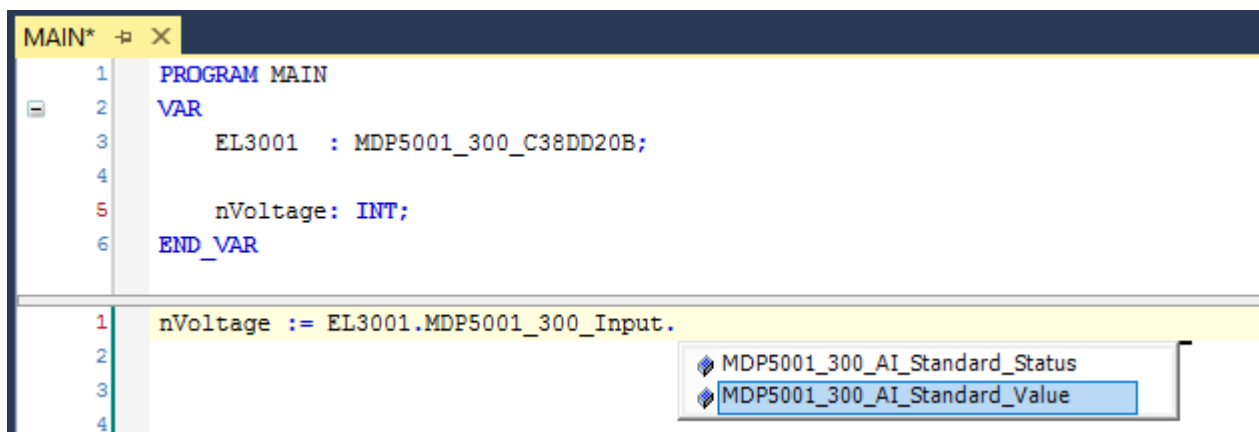
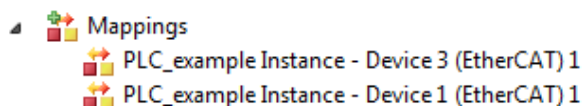


Fig. 110: Reading a variable from the structure of the process data

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:

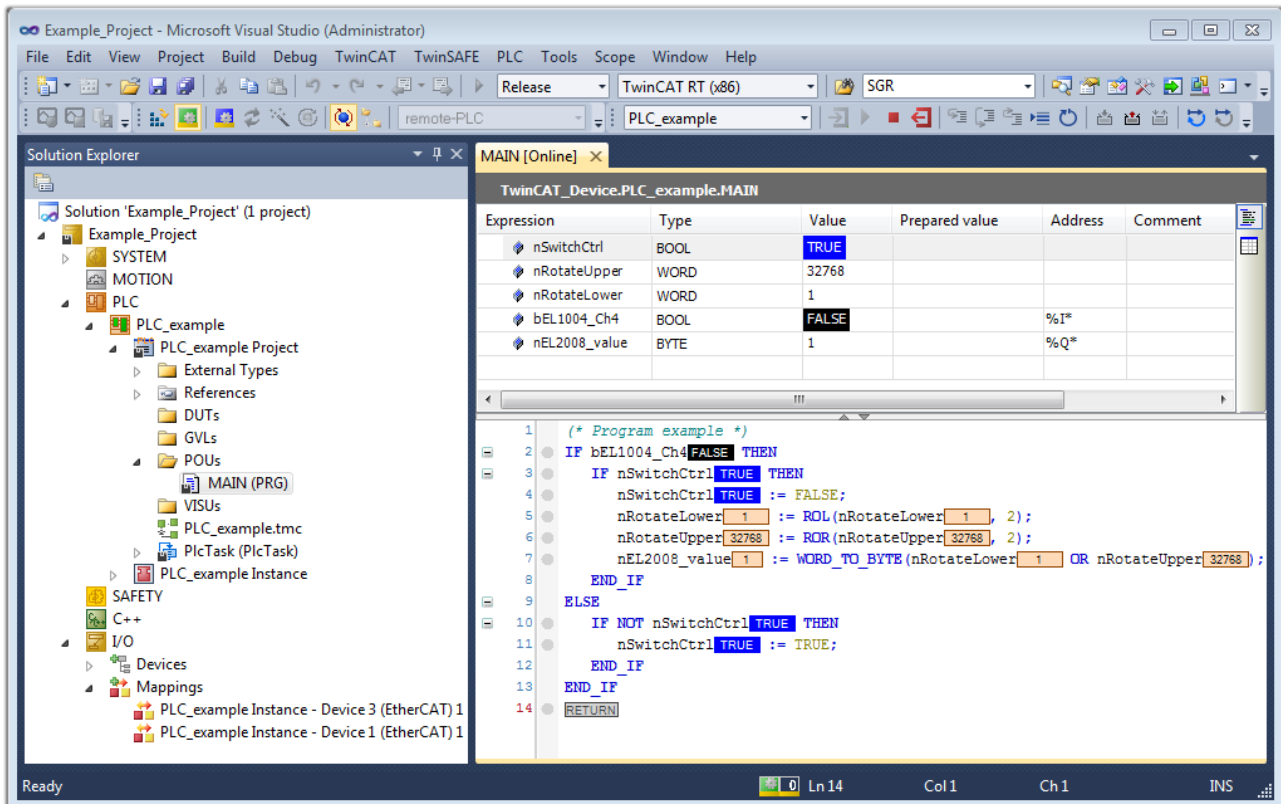


Fig. 111: TwinCAT 3 development environment (VS shell): logged-in, after program startup

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).

9.2 TwinCAT Development Environment

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

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

Details:

- **TwinCAT 2:**
 - Connects I/O devices to tasks in a variable-oriented manner
 - Connects tasks to tasks in a variable-oriented manner
 - Supports units at the bit level
 - Supports synchronous or asynchronous relationships
 - Exchange of consistent data areas and process images

- Datalink on NT - Programs by open Microsoft Standards (OLE, OCX, ActiveX, DCOM+, etc.)
- Integration of IEC 61131-3-Software-SPS, Software- NC and Software-CNC within Windows NT/ 2000/XP/Vista, Windows 7, NT/XP Embedded, CE
- Interconnection to all common fieldbusses
- More...

Additional features:

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

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

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

9.2.1 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.

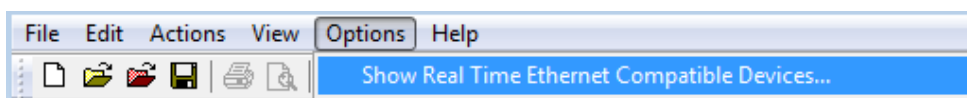


Fig. 112: System Manager "Options" (TwinCAT 2)

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

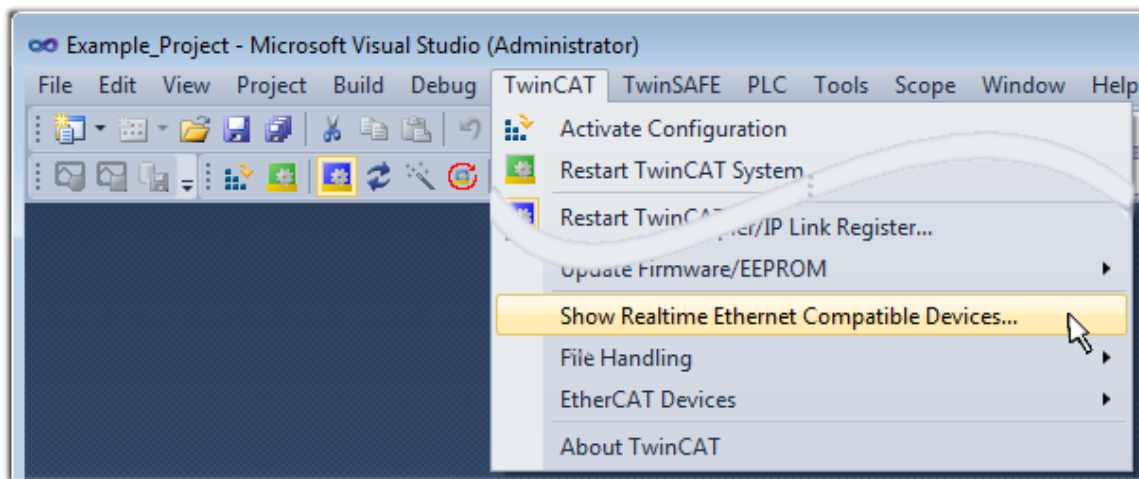


Fig. 113: Call up under VS Shell (TwinCAT 3)

B: Via TcRteInstall.exe in the TwinCAT directory

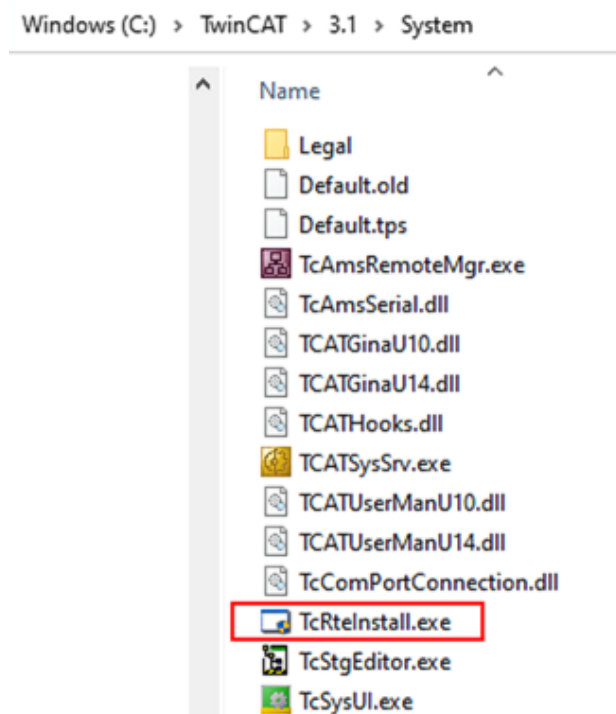


Fig. 114: TcRteInstall in the TwinCAT directory

In both cases, the following dialog appears:

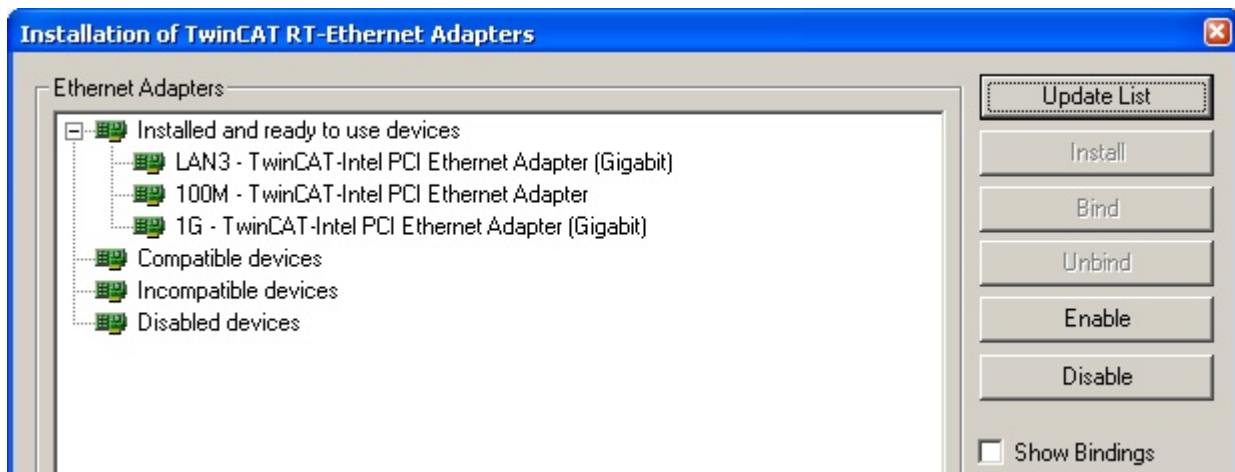


Fig. 115: Overview of network interfaces

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

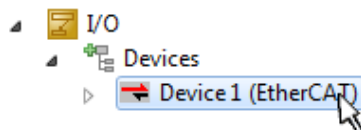
A Windows warning regarding the unsigned driver can be ignored.

Alternatively an EtherCAT-device can be inserted first of all as described in chapter [Offline configuration creation](#), section “Creating the EtherCAT device” [► 146] in order to view the compatible ethernet ports via its EtherCAT properties (tab “Adapter”, button “Compatible Devices...”):



Fig. 116: 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)

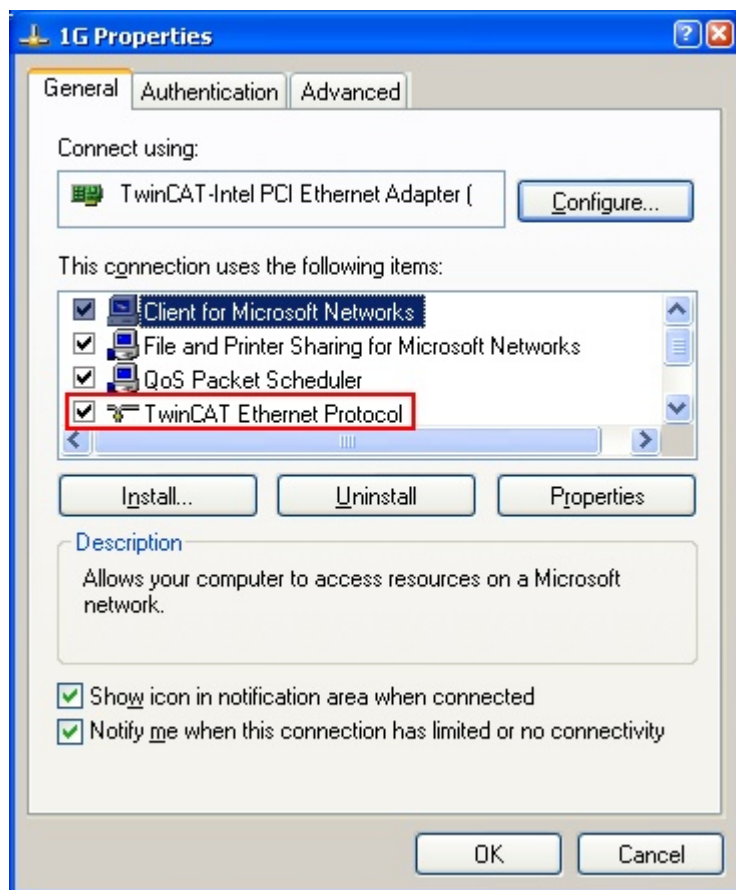


Fig. 117: Windows properties of the network interface

A correct setting of the driver could be:

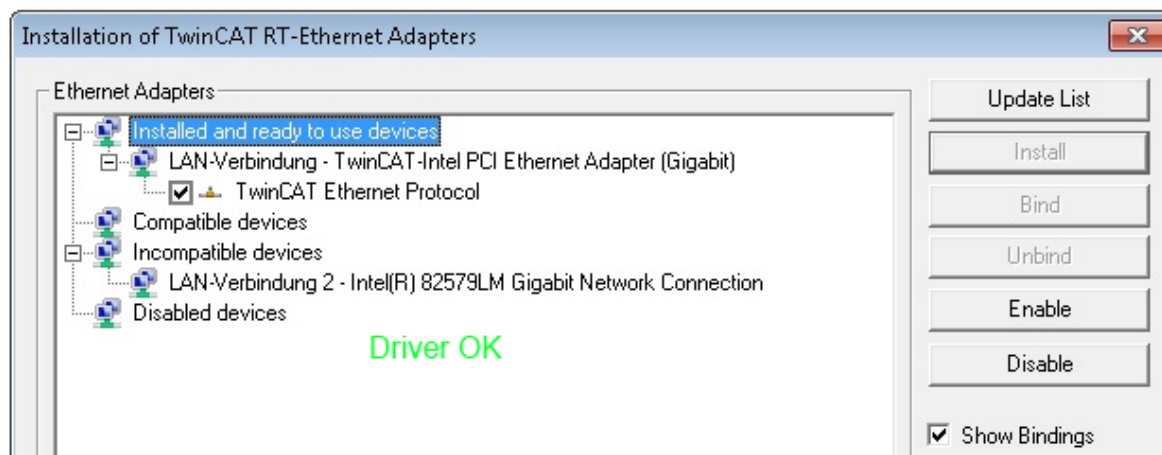


Fig. 118: Exemplary correct driver setting for the Ethernet port

Other possible settings have to be avoided:

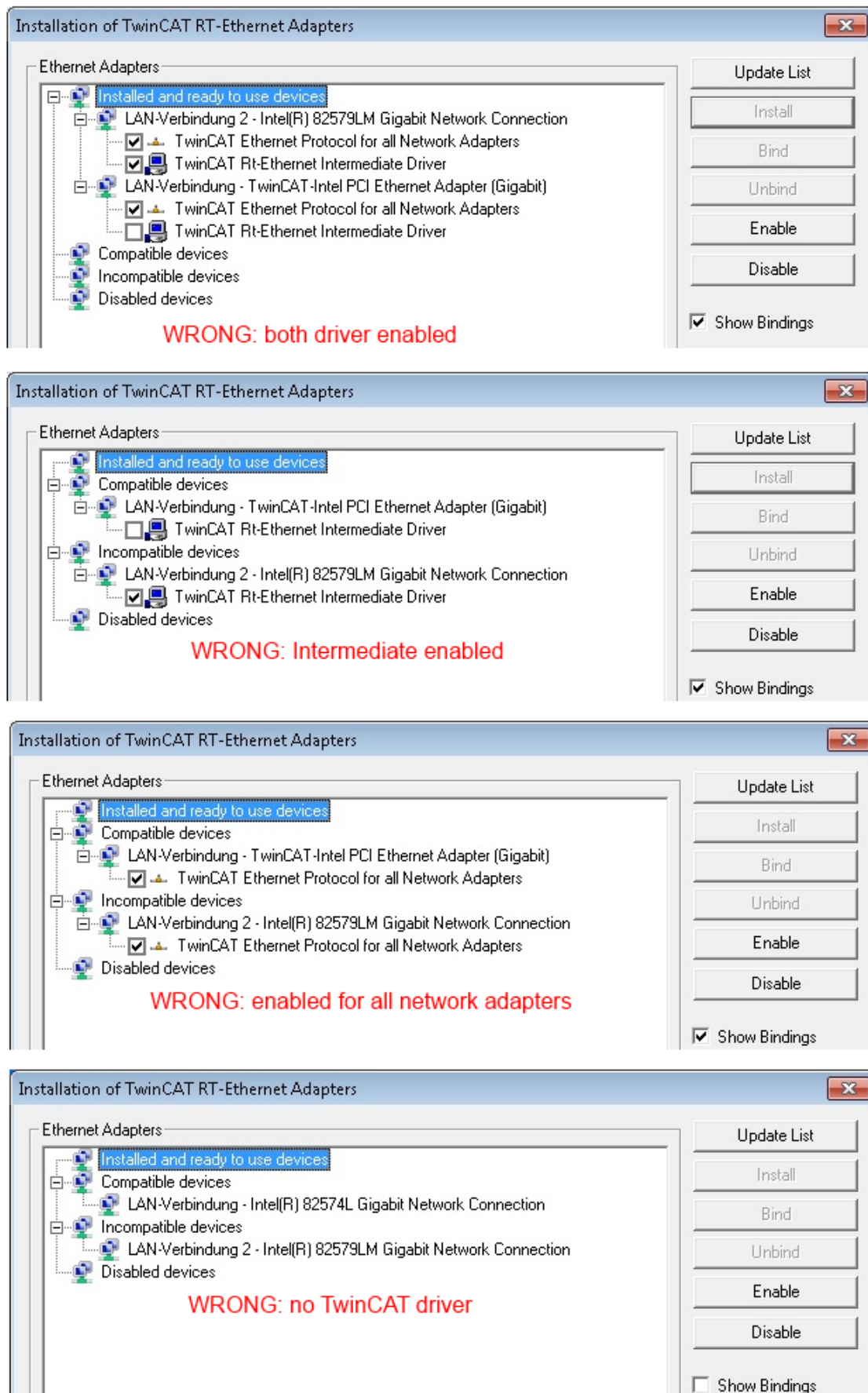


Fig. 119: Incorrect driver settings for the Ethernet port

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.

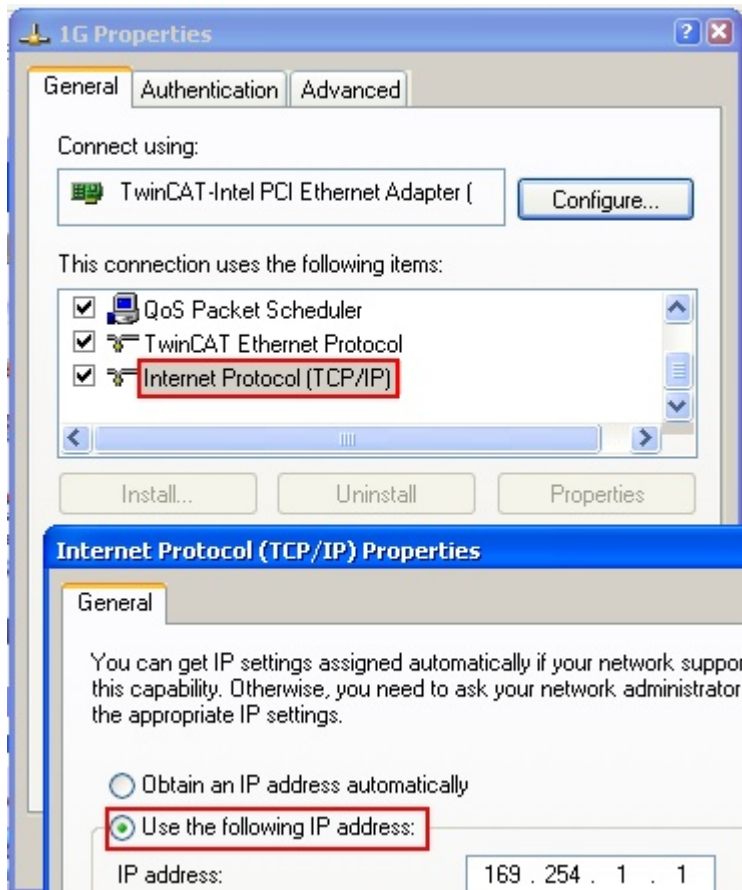


Fig. 120: TCP/IP setting for the Ethernet port

9.2.2 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](#) [► 145] 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”

Name
(EL2521-0025-1018)
Revision

Fig. 121: 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](#) [► 12].

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.

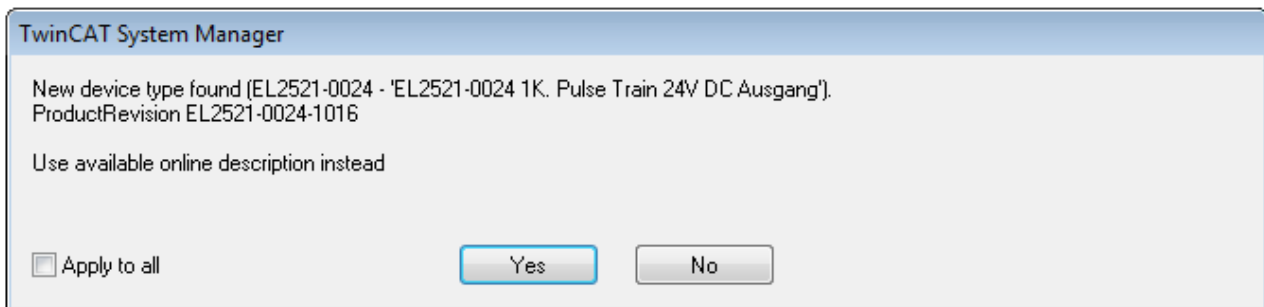


Fig. 122: OnlineDescription information window (TwinCAT 2)

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

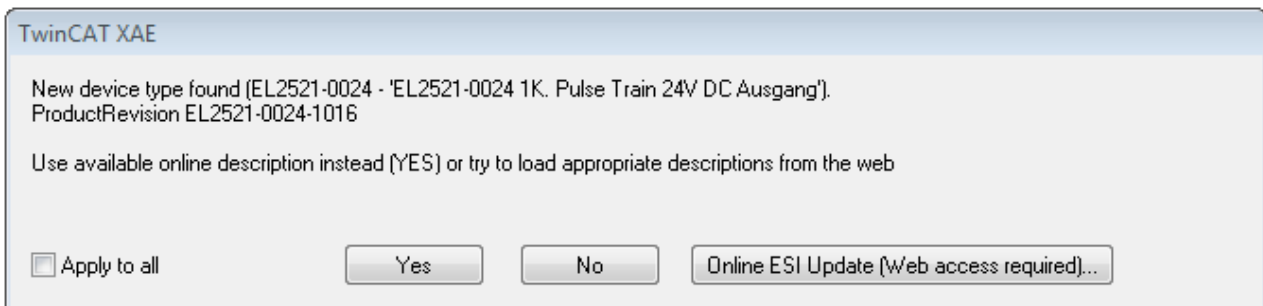


Fig. 123: Information window OnlineDescription (TwinCAT 3)

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 [► 146]”.

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.

OnlineDescriptionCache000000002.xml

Fig. 124: File OnlineDescription.xml created by the System Manager

If a slave is 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*).

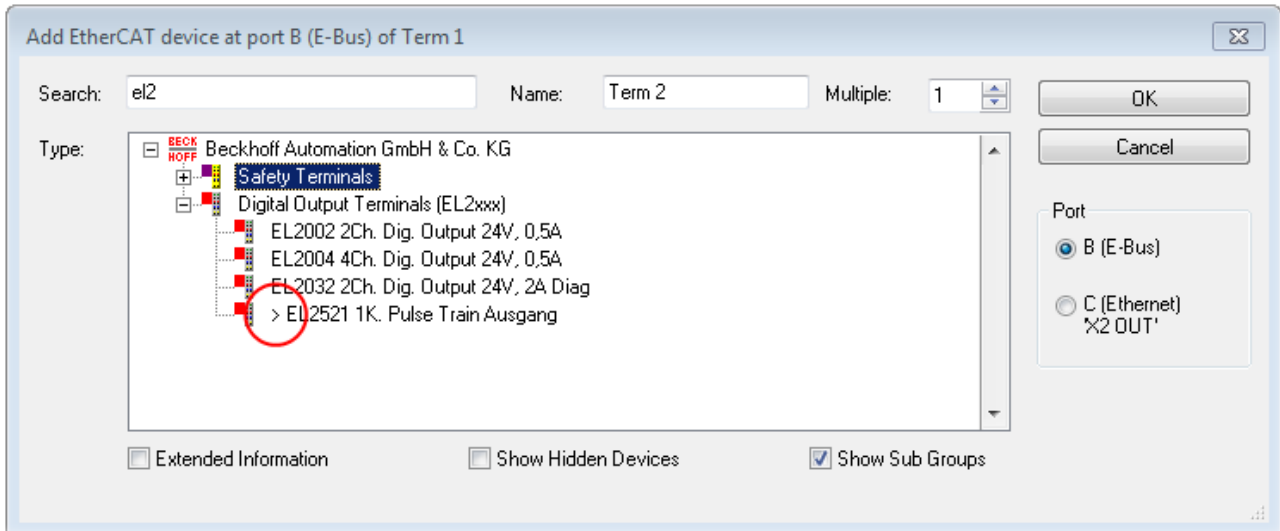


Fig. 125: Indication of an online recorded ESI of EL2521 as an example

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.

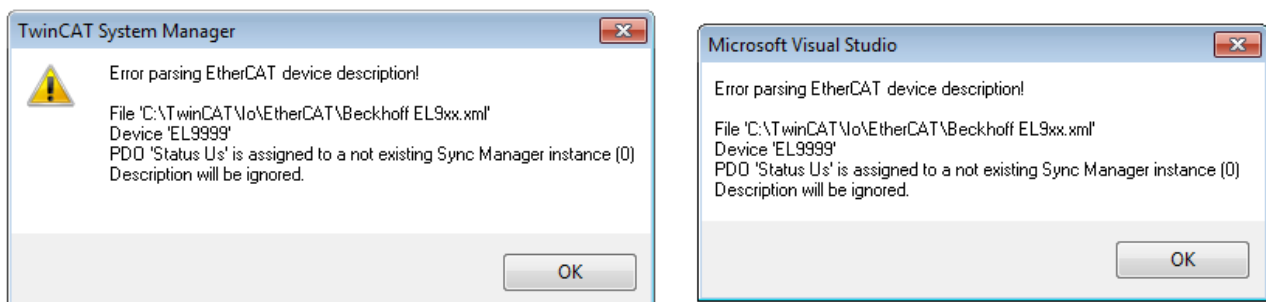


Fig. 126: Information window for faulty ESI file (left: TwinCAT 2; right: TwinCAT 3)

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

9.2.3 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:

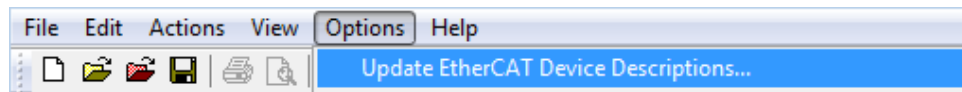


Fig. 127: Using the ESI Updater (>= TwinCAT 2.11)

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

Selection under TwinCAT 3:

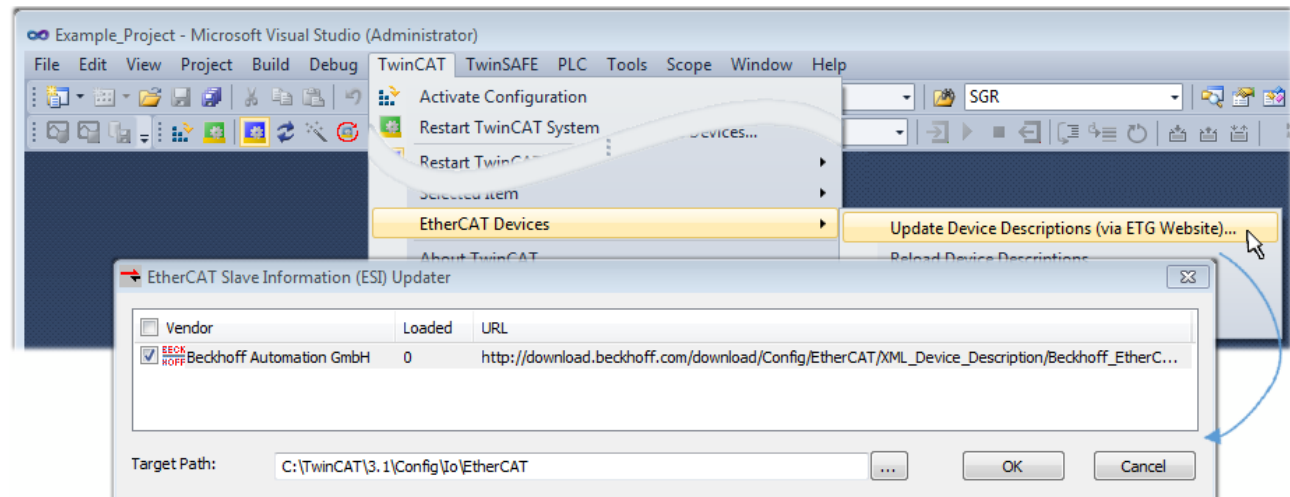


Fig. 128: Using the ESI Updater (TwinCAT 3)

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)...".

9.2.4 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" [► 141].

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 [► 151] (Ethernet port at the IPC)
- detecting the connected EtherCAT devices [► 152]. This step can be carried out independent of the preceding step
- troubleshooting [► 155]

The scan with existing configuration [► 156] can also be carried out for comparison.

9.2.5 OFFLINE configuration creation

Creating the EtherCAT device

Create an EtherCAT device in an empty System Manager window.

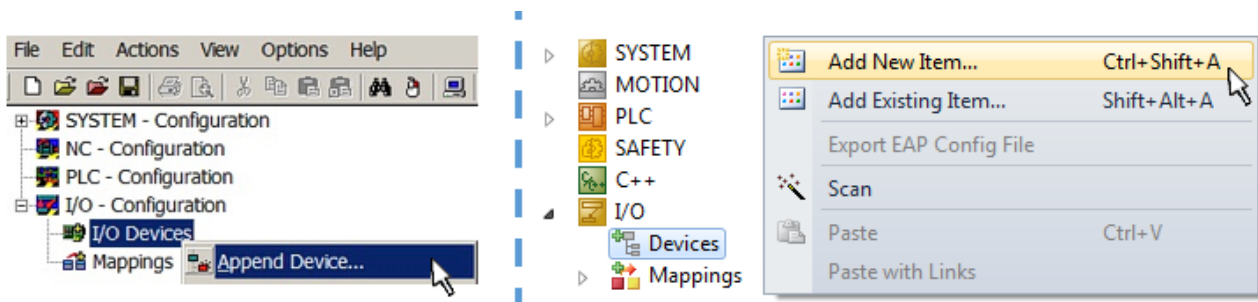


Fig. 129: Append EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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”.

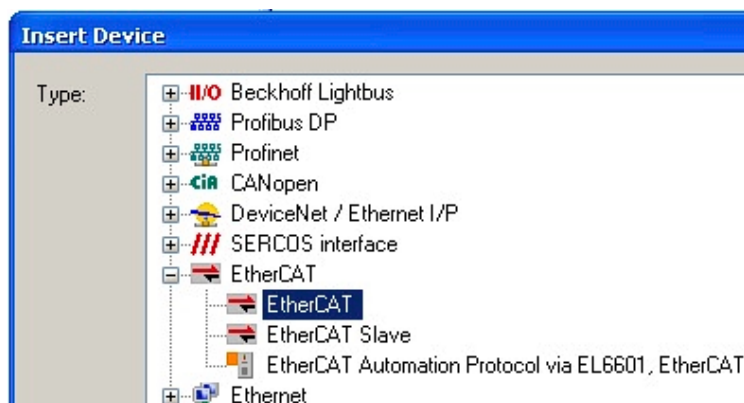


Fig. 130: Selecting the EtherCAT connection (TwinCAT 2.11, TwinCAT 3)

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

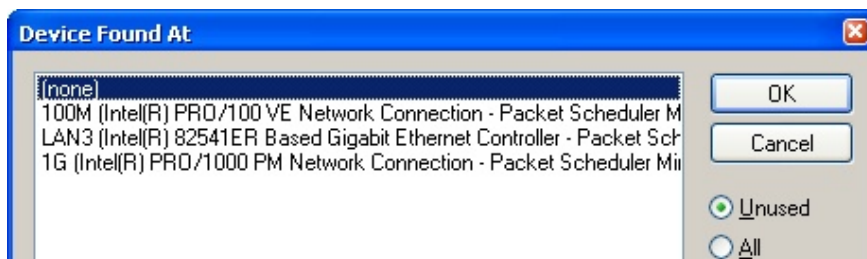


Fig. 131: Selecting the Ethernet port

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)”.

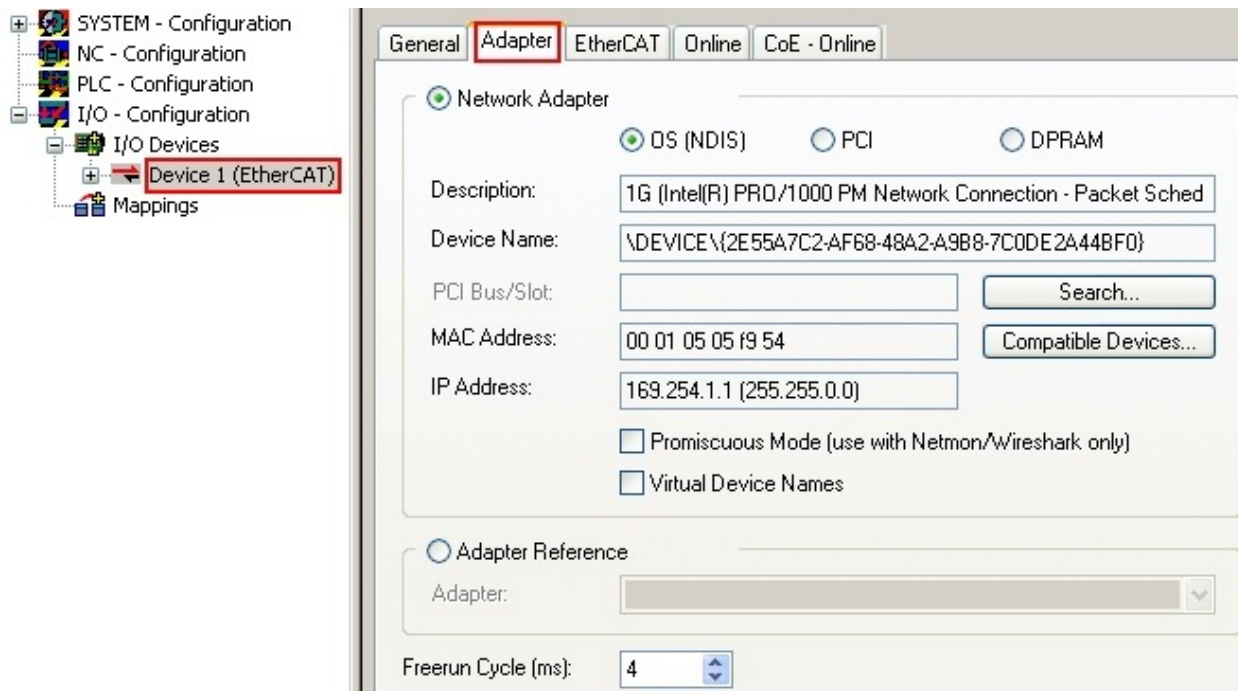
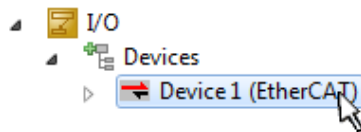


Fig. 132: EtherCAT device properties (TwinCAT 2)

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

i 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 \[▶ 135\]](#).

Defining EtherCAT slaves

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

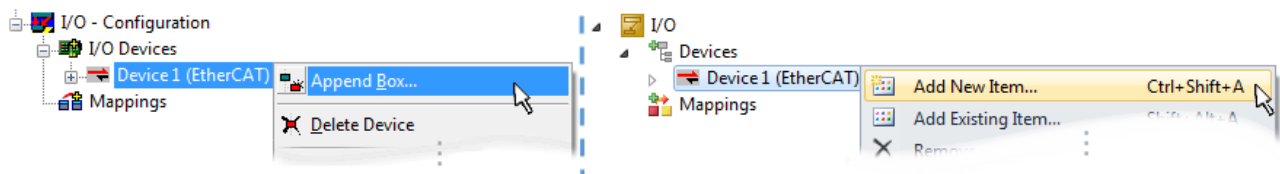


Fig. 133: Appending EtherCAT devices (left: TwinCAT 2; right: TwinCAT 3)

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).

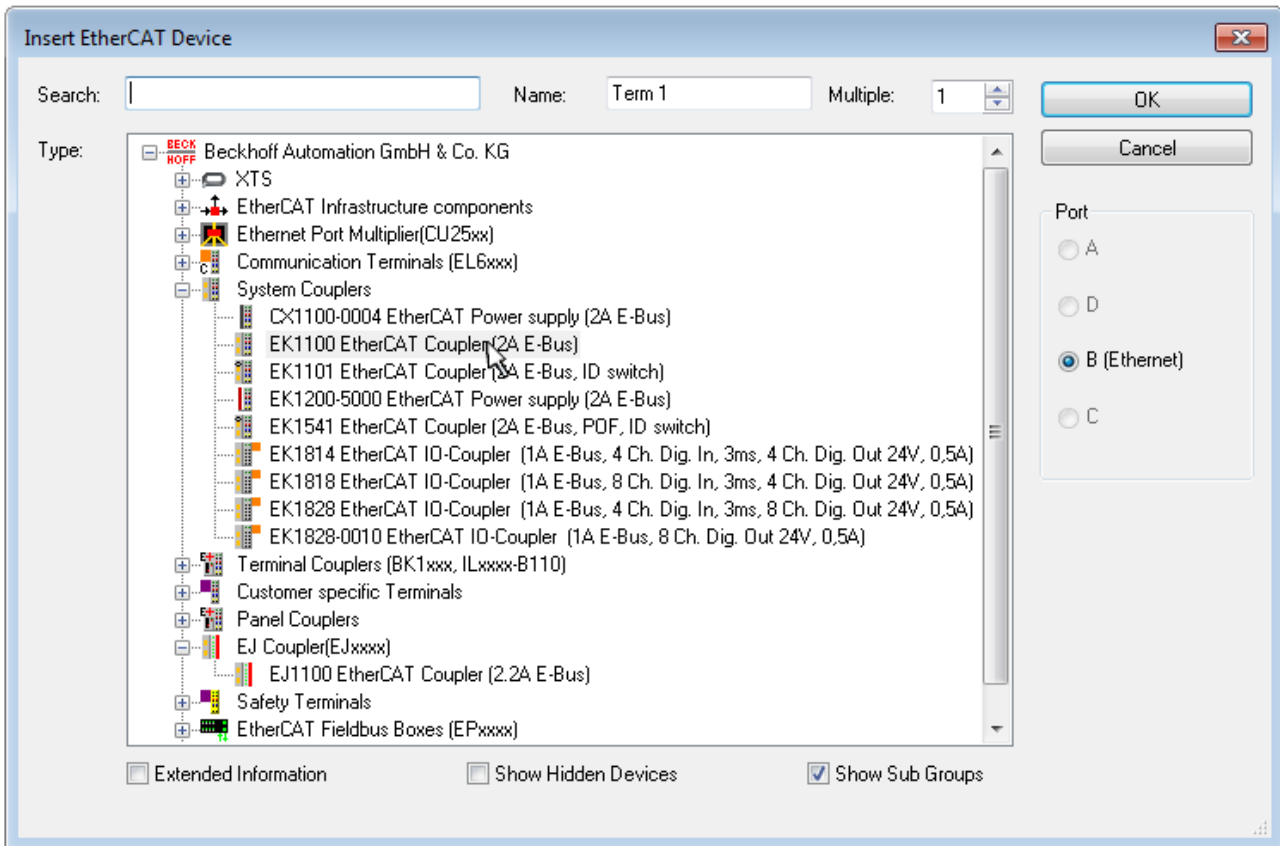


Fig. 134: Selection dialog for new EtherCAT device

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”.

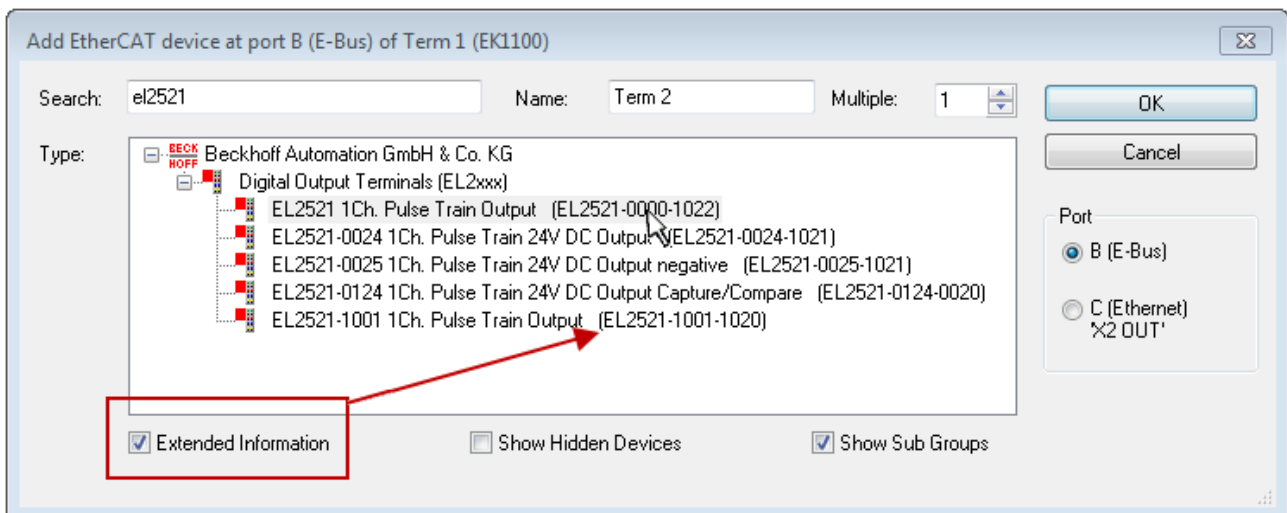


Fig. 135: Display of device revision

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”.

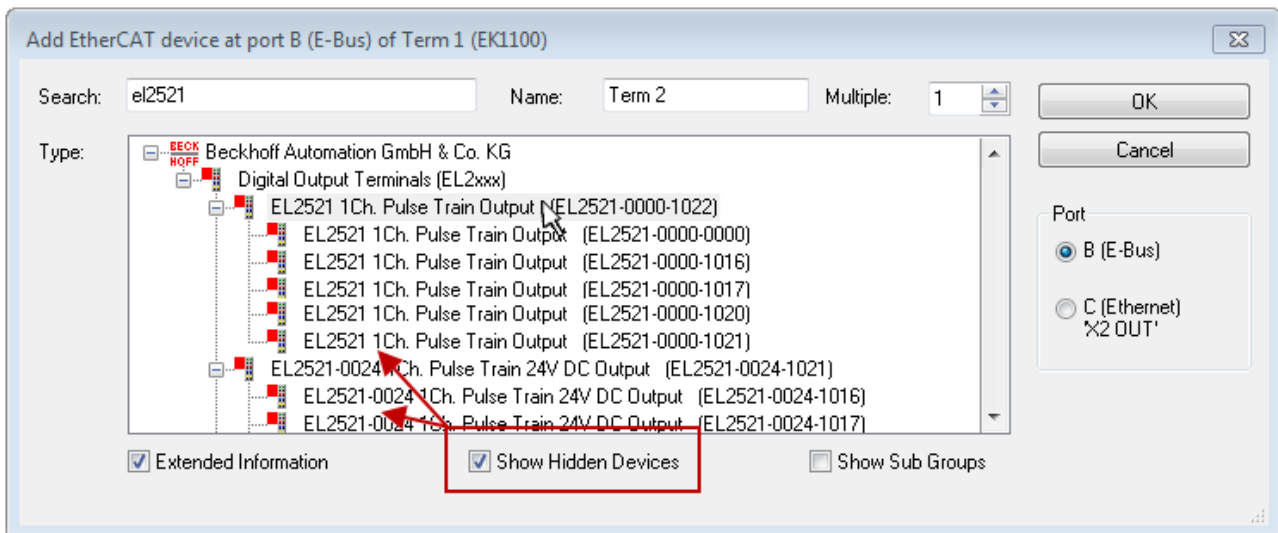


Fig. 136: Display of previous revisions

● Device selection based on revision, compatibility

i 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 $\geq$ 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.

Name
(EL2521-0025-1018)
Revision

Fig. 137: 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, ...

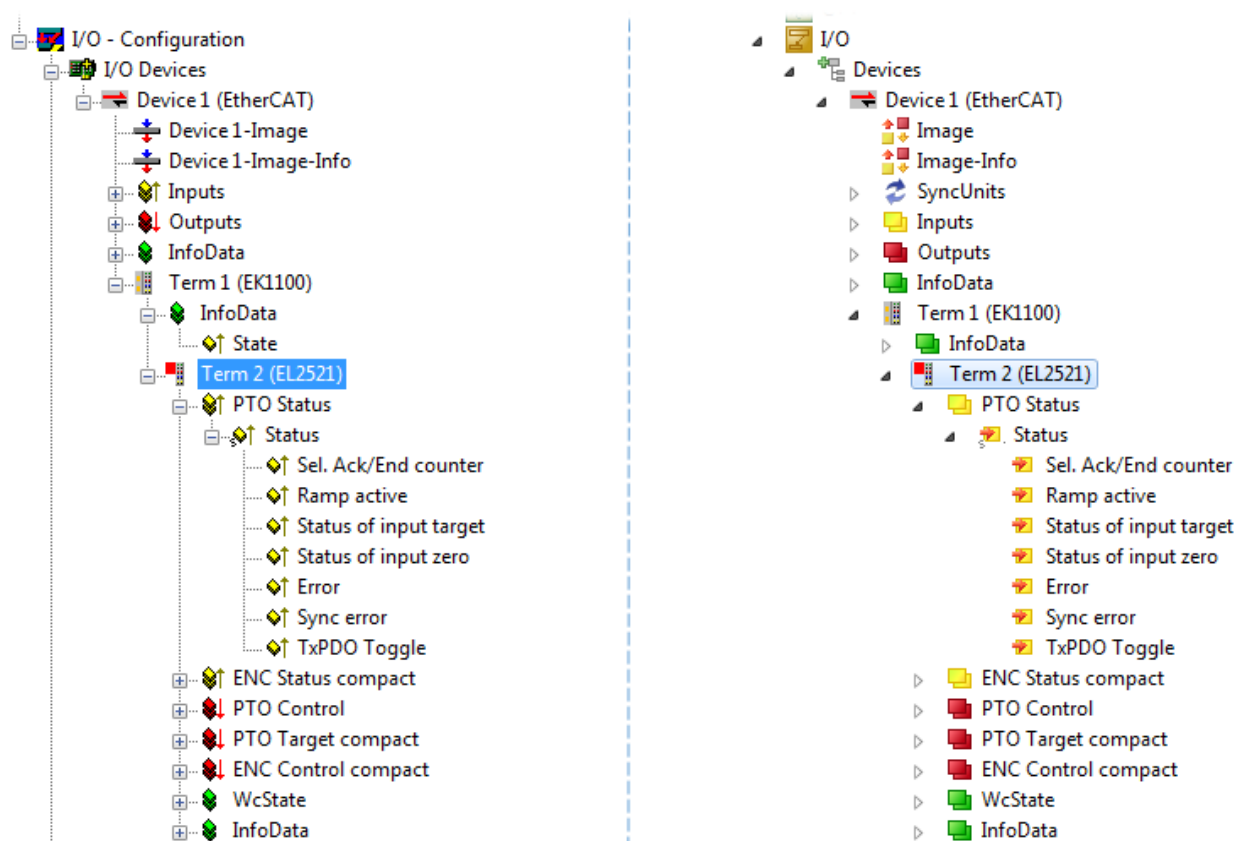


Fig. 138: EtherCAT terminal in the TwinCAT tree (left: TwinCAT 2; right: TwinCAT 3)

9.2.6 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:  .
- 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)”

1 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.

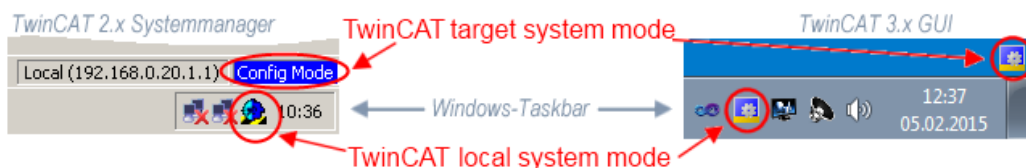


Fig. 139: Differentiation local/target system (left: TwinCAT 2; right: TwinCAT 3)

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

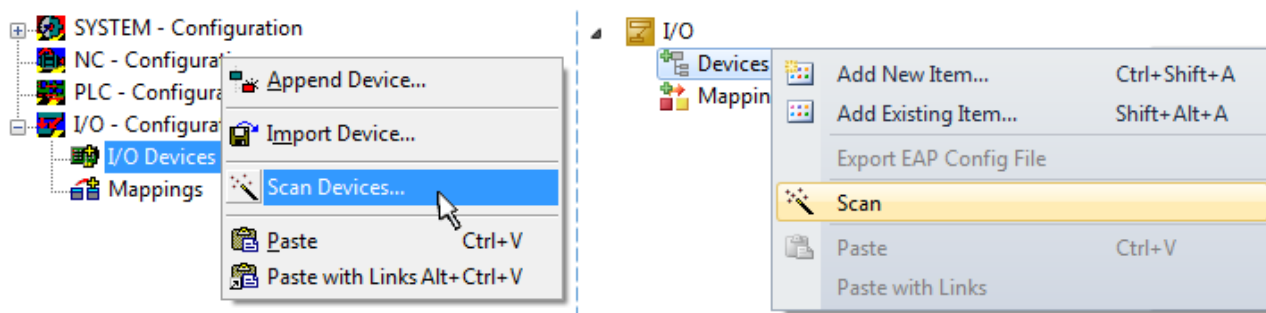


Fig. 140: Scan Devices (left: TwinCAT 2; right: TwinCAT 3)

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

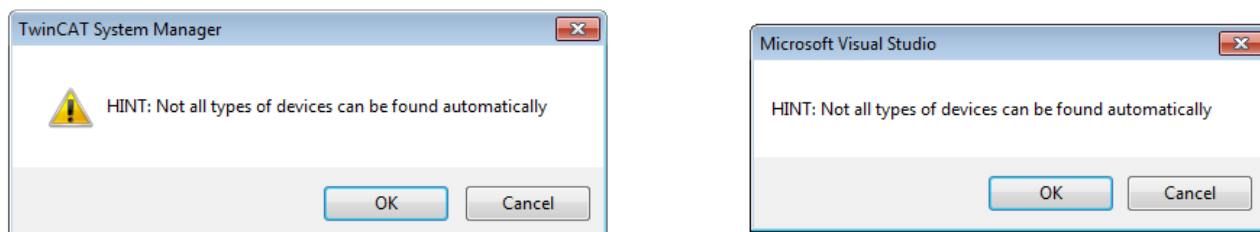


Fig. 141: Note for automatic device scan (left: TwinCAT 2; right: TwinCAT 3)

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”.

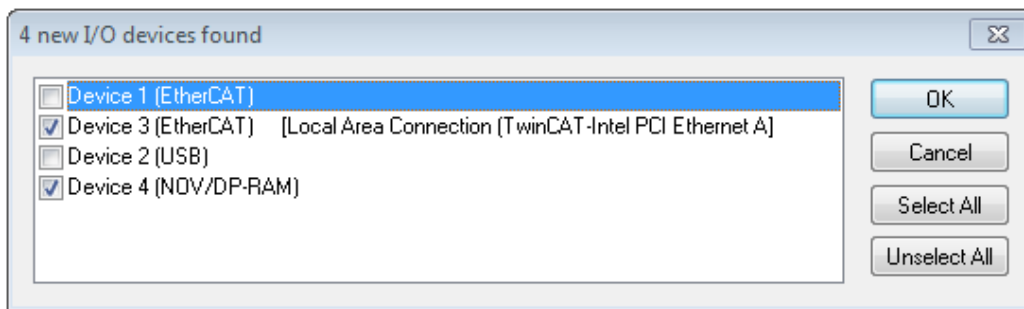


Fig. 142: Detected Ethernet devices

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

i

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](#) [► 135].

Detecting/Scanning the EtherCAT devices

● Online scan functionality

i

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.

Name
(EL2521-0025-1018)
Revision

Fig. 143: 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](#) [► 156] 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:

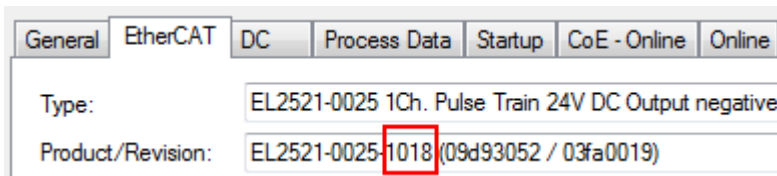


Fig. 144: Installing EtherCAT terminal with revision -1018

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 [► 156] 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:

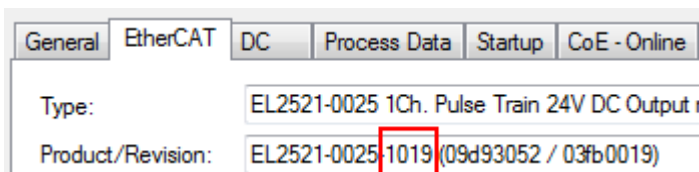


Fig. 145: Detection of EtherCAT terminal with revision -1019

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

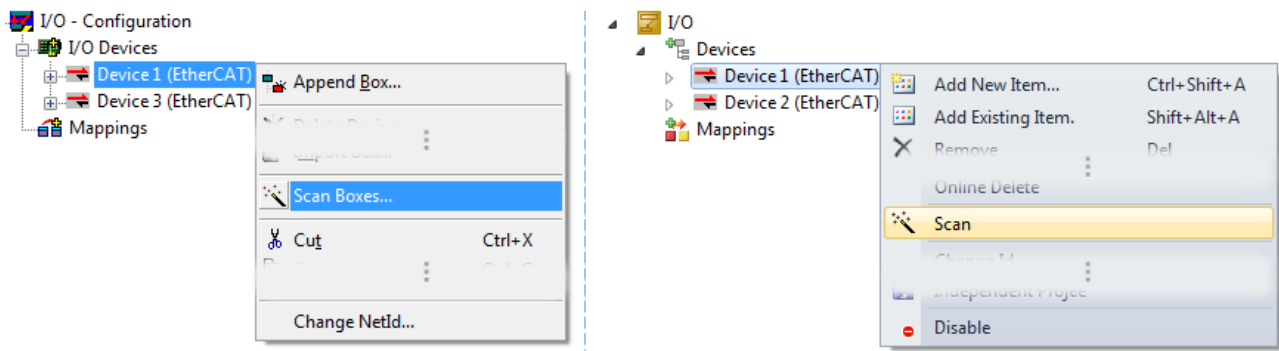


Fig. 147: Manual scanning for devices on a specified EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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. 148: Scan progress example by TwinCAT 2

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



Fig. 149: 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. 150: Displaying of "Free Run" and "Config Mode" toggling right below in the status bar



Fig. 151: 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*.

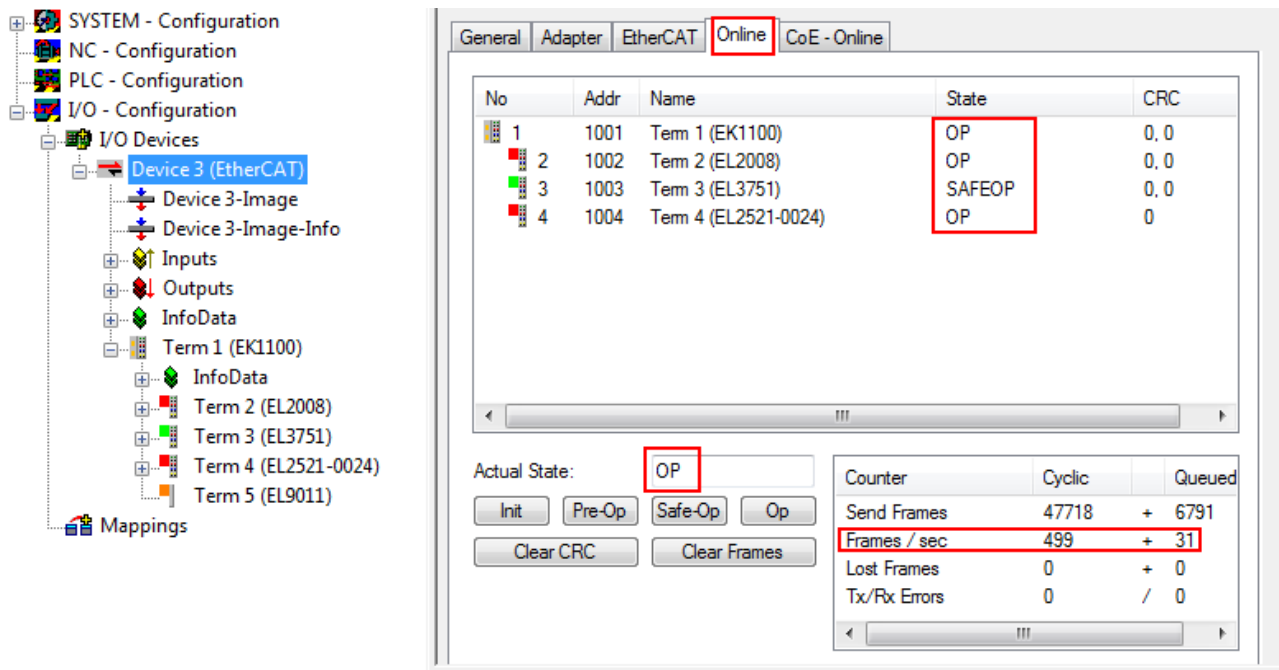


Fig. 152: Online display example

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](#) [► 146].

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.

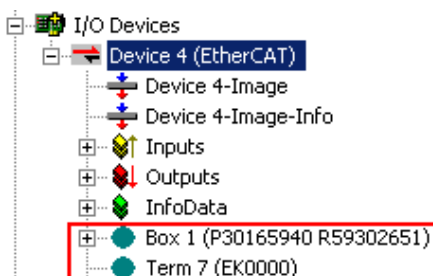


Fig. 153: Faulty identification

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. 154: 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.

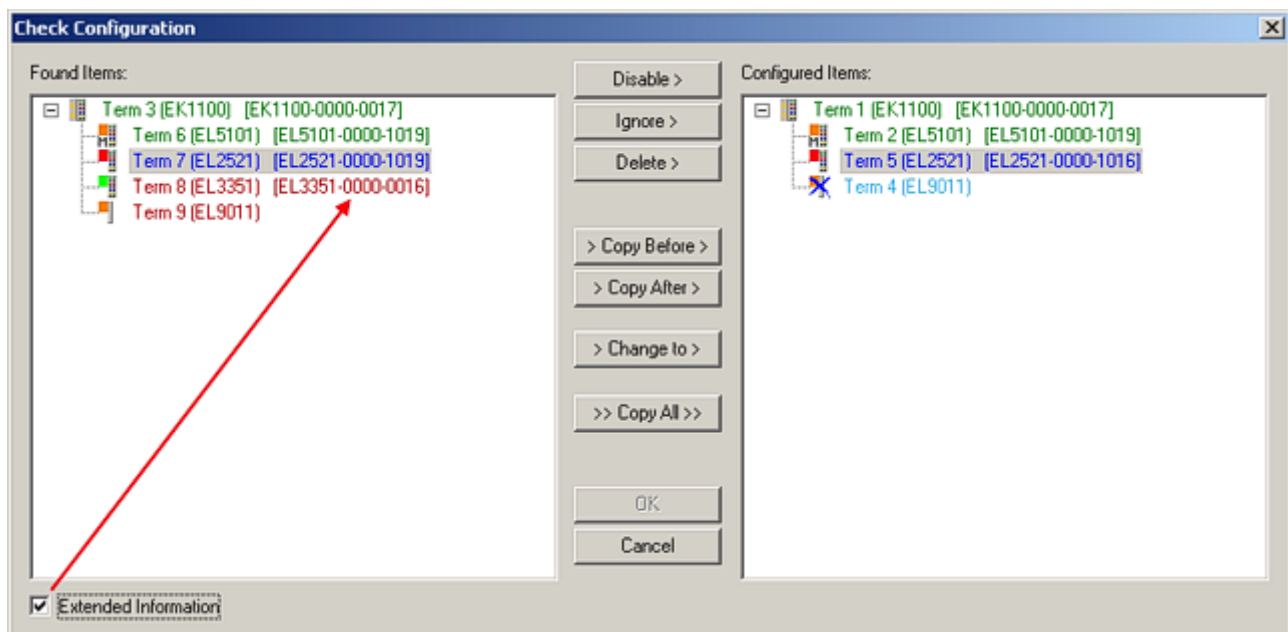


Fig. 155: Correction dialog

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.

i 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 $\geq$ 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.

Name
(EL2521-0025-1018)
Revision

Fig. 156: 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, ...

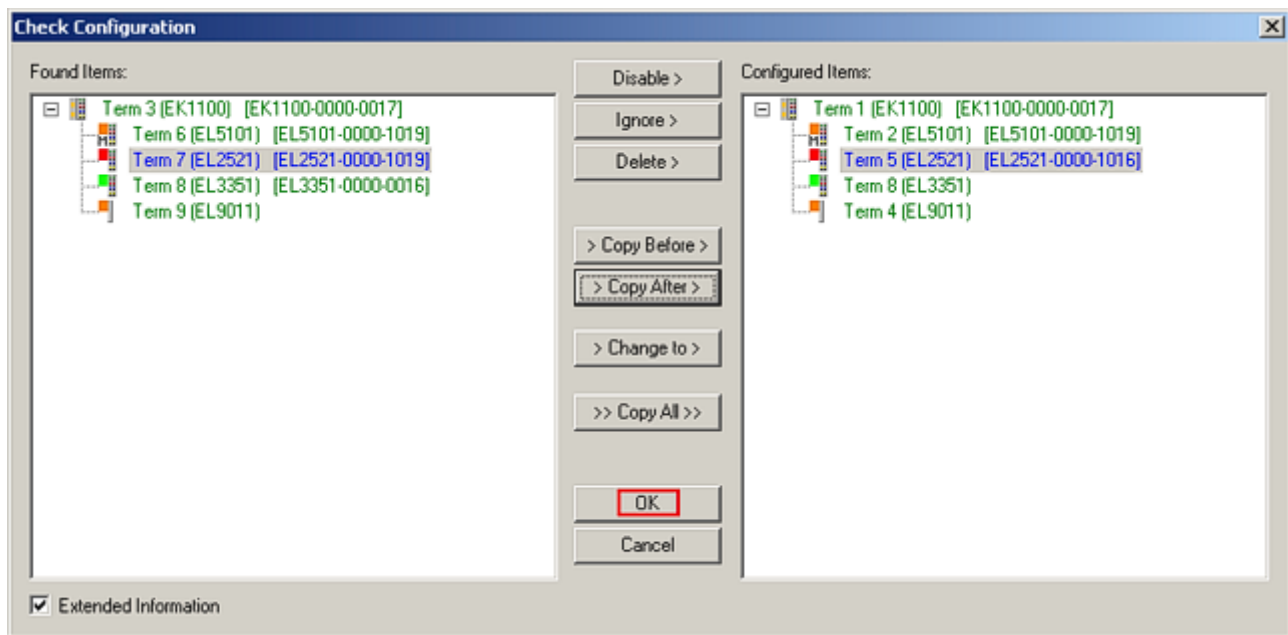


Fig. 157: Correction dialog with modifications

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.

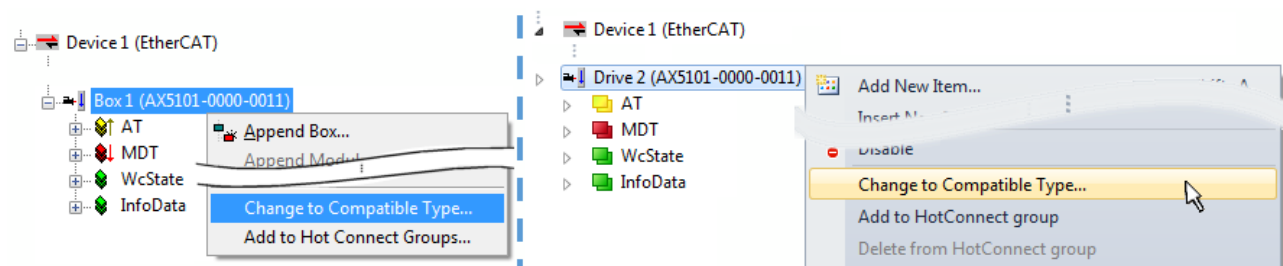


Fig. 158: Dialog “Change to Compatible Type...” (left: TwinCAT 2; right: TwinCAT 3)

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, Entry.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

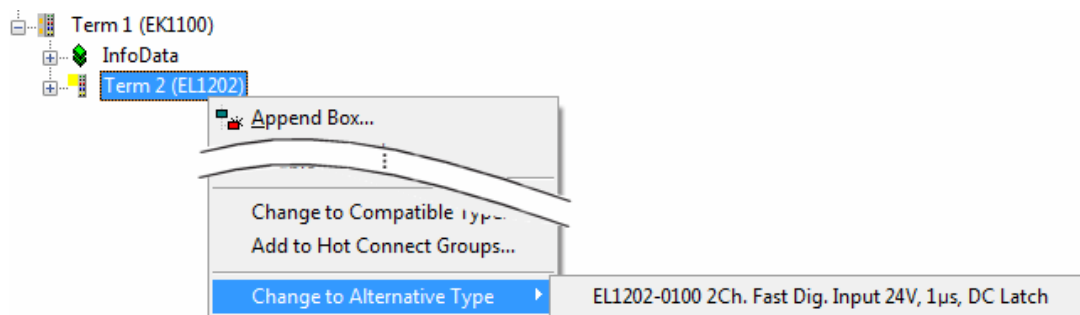


Fig. 159: TwinCAT 2 Dialog Change to Alternative Type

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).

9.2.7 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).

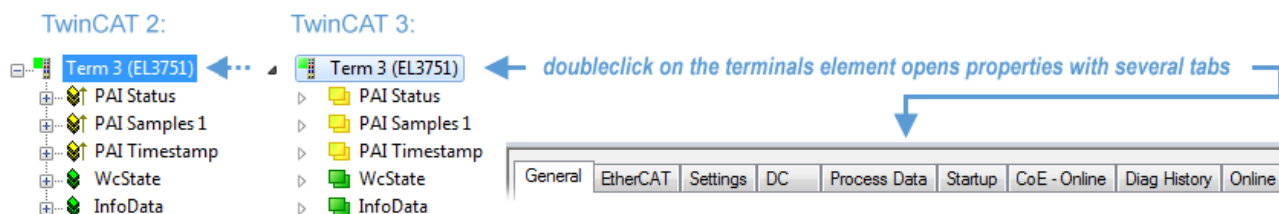


Fig. 160: Branch element as terminal EL3751

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

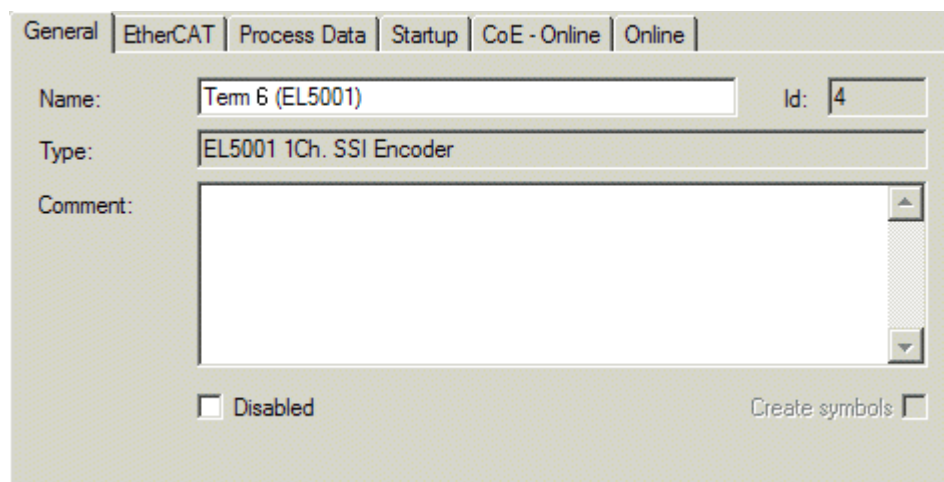


Fig. 161: “General” tab

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. 162: “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 (**Process Data Objects**, 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.

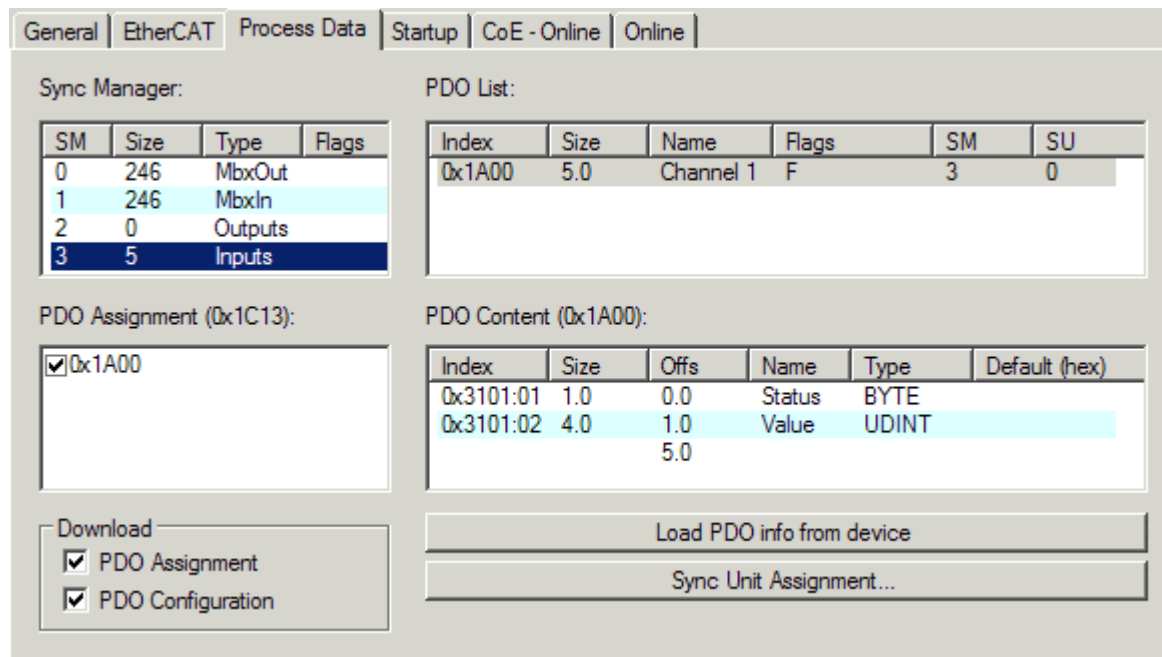


Fig. 163: "Process Data" tab

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").

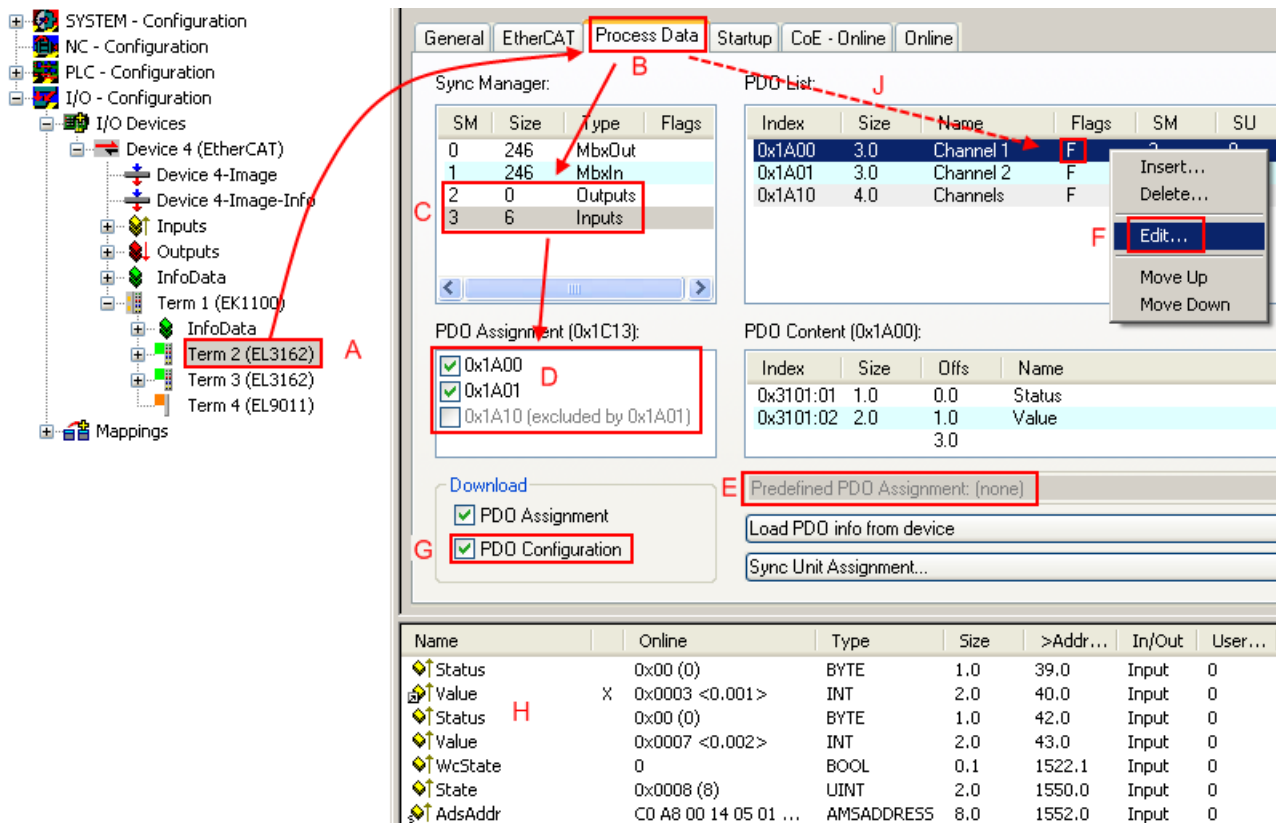


Fig. 164: Configuring the process data

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 [► 167] 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.

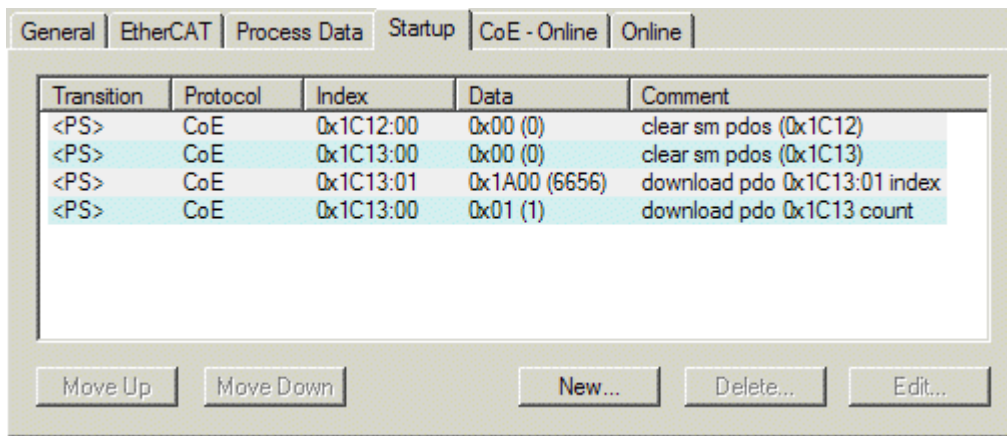


Fig. 165: "Startup" tab

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.

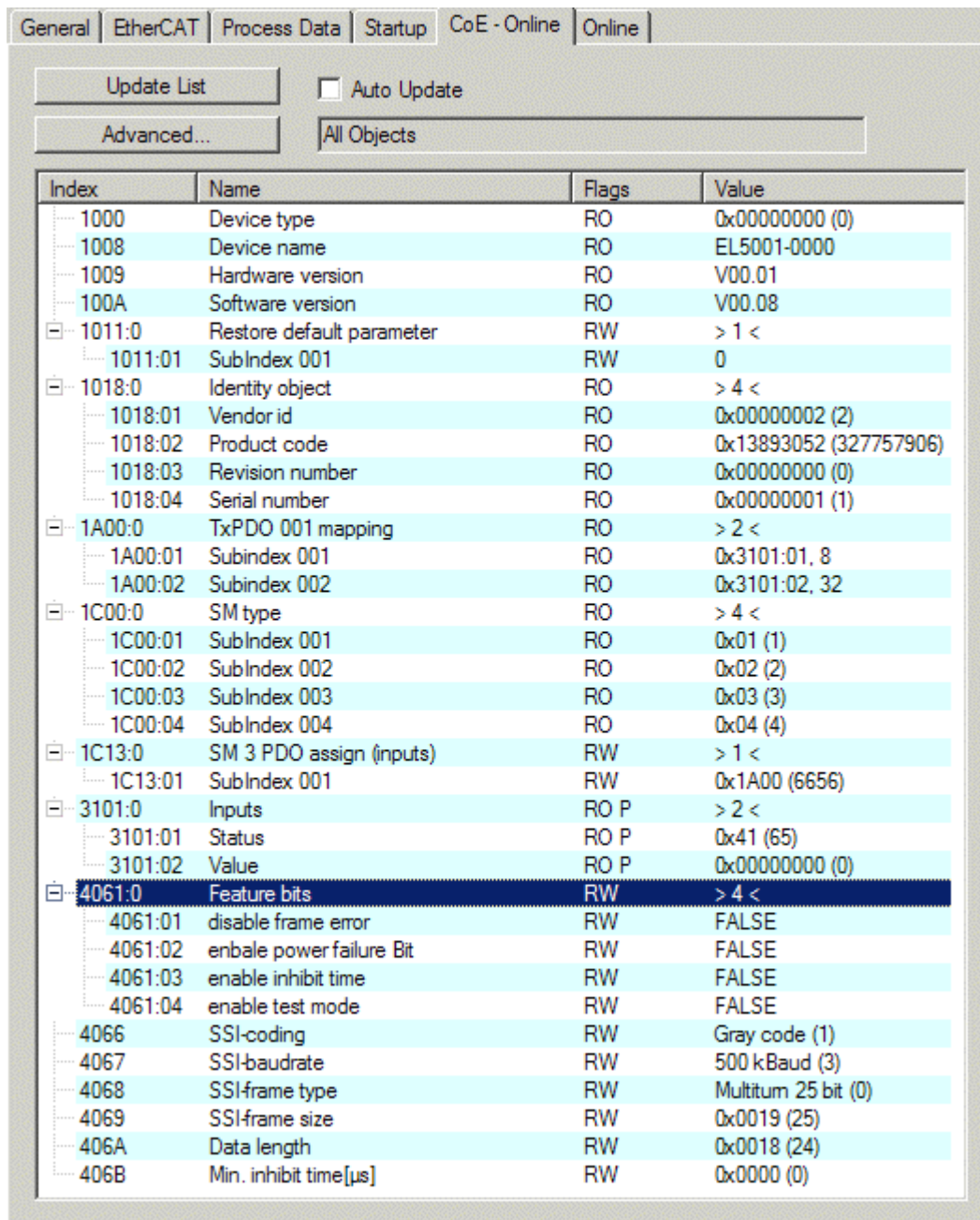


Fig. 166: "CoE - Online" tab

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.

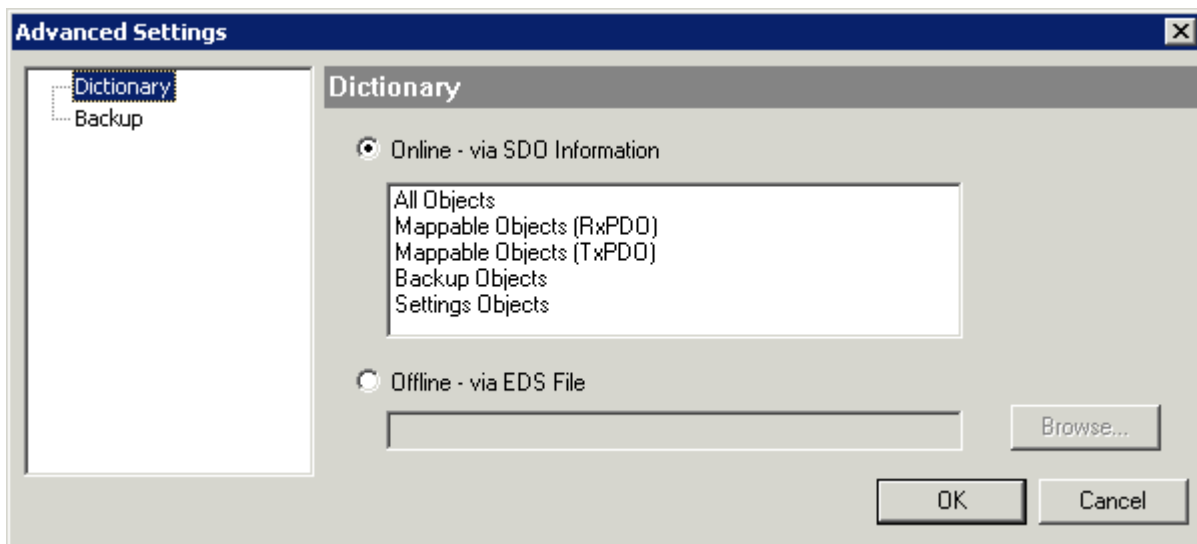


Fig. 167: Dialog “Advanced settings”

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

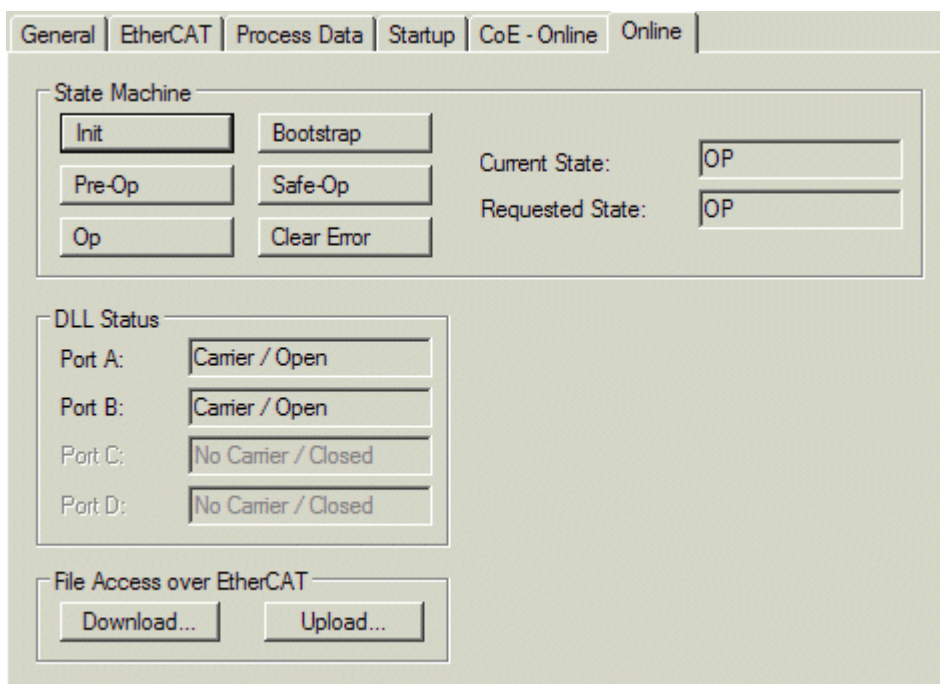


Fig. 168: “Online” tab

State Machine

Init	This button attempts to set the EtherCAT device to the <i>Init</i> state.
Pre-Op	This button attempts to set the EtherCAT device to the <i>pre-operational</i> state.
Op	This button attempts to set the EtherCAT device to the <i>operational</i> state.
Bootstrap	This button attempts to set the EtherCAT device to the <i>Bootstrap</i> state.
Safe-Op	This button attempts to set the EtherCAT device to the <i>safe-operational</i> 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 <i>Clear Error</i> 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)

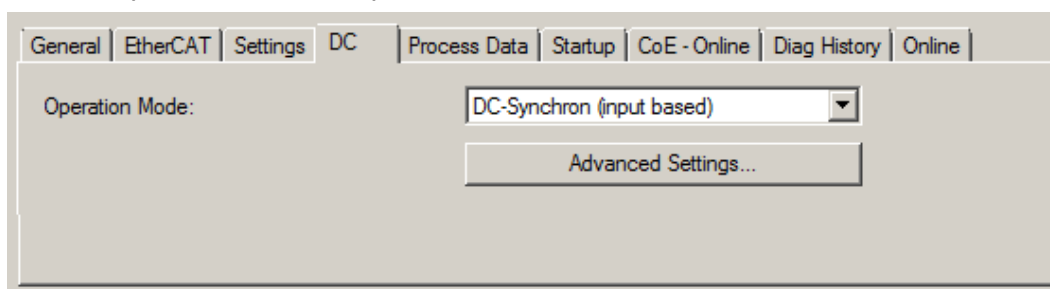


Fig. 169: “DC” tab (Distributed Clocks)

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

9.2.7.1 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.

i 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 \[► 165\]](#)),
 - 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 <i>PDO Assignment</i> 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 \[► 162\]](#) 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.

9.2.8 Import/Export of EtherCAT devices with SCI and XTI

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

9.2.8.1 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 screenshot shows the TwinCAT Project34 interface. The Solution Explorer on the left shows the project structure, with 'Term 2 (EL3702)' selected. The main window displays the 'Process Data' tab, which includes a 'Sync Manager' table, a 'PDO List' table, a 'PDO Assignment (0x1C12)' list, and a 'PDO Content (0x1B00)' table. The 'Online' tab at the bottom shows a table of data points.

Sync Manager:

SM	Size	Type	Flags
0	6	Inputs	
1	6	Inputs	
2	4	Inputs	

PDO List:

Index	Size	Name
0x1B00	2.0	Ch1 CycleCount
0x1A00	2.0	Ch1 Sample 0
0x1A01	2.0	Ch1 Sample 1
0x1A02	2.0	Ch1 Sample 2
0x1A03	2.0	Ch1 Sample 3
0x1A04	2.0	Ch1 Sample 4
0x1A05	2.0	Ch1 Sample 5

PDO Assignment (0x1C12):

☐ 0x1AE0
☐ 0x1AE1
☐ 0x1AE2
☒ 0x1B10

PDO Content (0x1B00):

Index	Size	Offs	Name
0x6800:01	2.0	0.0	Ch1 CycleCount
		2.0	

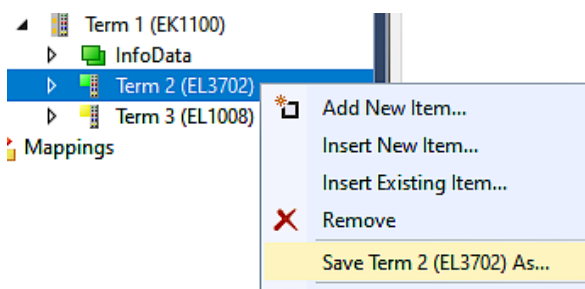
Online:

Name	Online	Type	Size	>Addr...
Ch1 CycleCount		UINT	2.0	58.0
Ch1 Value		INT	2.0	60.0
Ch1 Value		INT	2.0	62.0
Ch2 CycleCount		UINT	2.0	64.0
Ch2 Value		INT	2.0	66.0
Ch2 Value		INT	2.0	68.0
StartTimeNextLa...		UDINT	4.0	70.0
WcState		BIT	0.1	1522.2

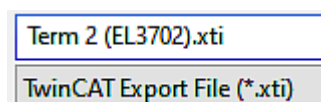
The two methods for exporting and importing the modified terminal referred to above are demonstrated below.

9.2.8.2 Procedure within TwinCAT with xti files

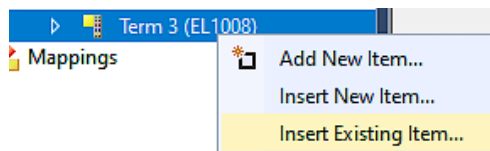
Each IO device can be exported/saved individually:



The xti file can be stored:



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



9.2.8.3 Procedure within and outside TwinCAT with sci file

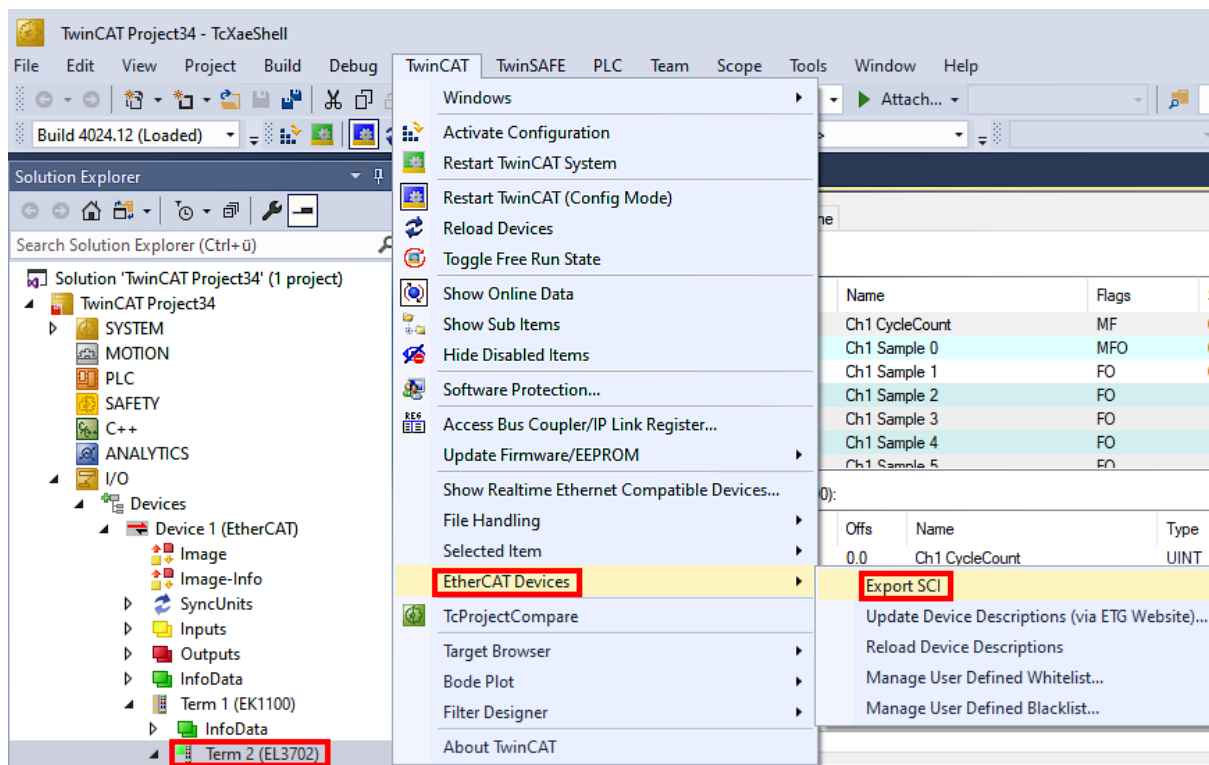
Note regarding availability (2021/01)

The SCI method is available from TwinCAT 3.1 build 4024.14.

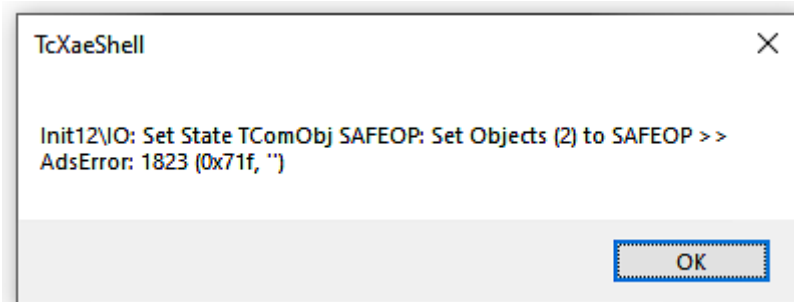
The Slave Configuration Information (SCI) describes a specific complete configuration for an EtherCAT slave (terminal, box, drive...) based on the setting options of the device description file (ESI, EtherCAT Slave Information). That is, it includes PDO, CoE, synchronization.

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.	

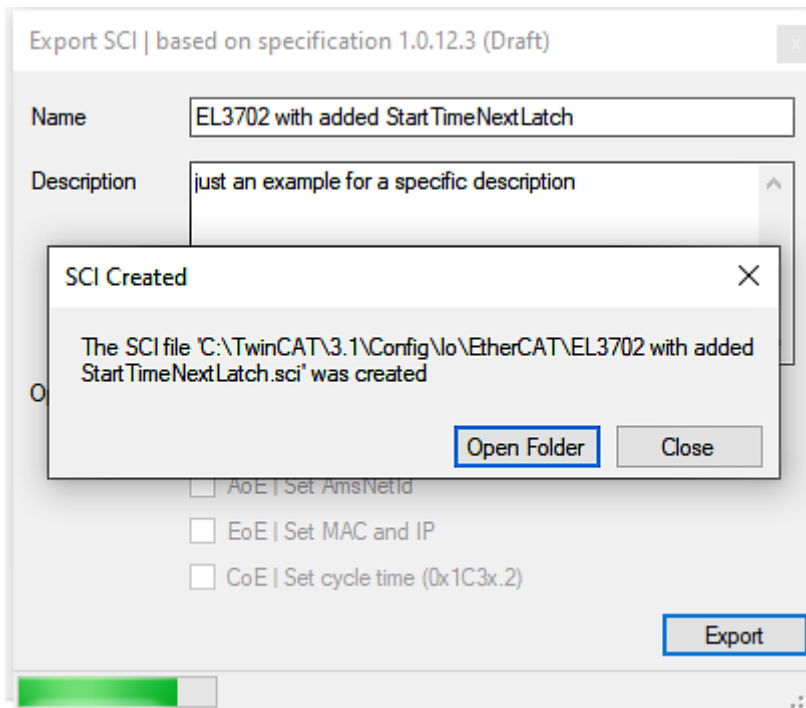
- A list view is available for multiple selections (*Export multiple SCI files*):

- 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:

Dateiname:	EL3702 with added StartTimeNextLatch.sci
Dateityp:	SCI file (*.sci)

- The export takes place:

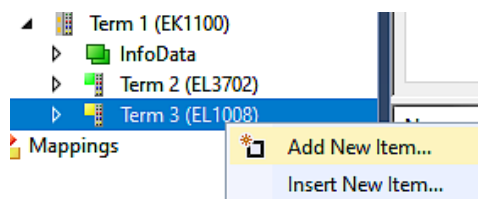


Import

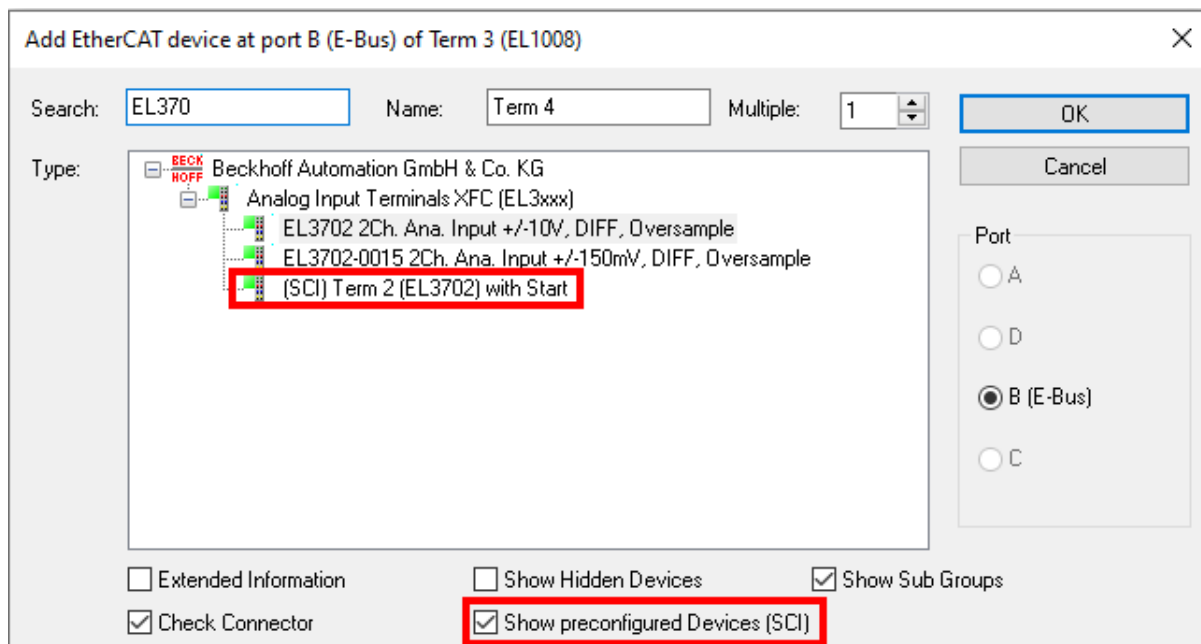
- An sci description can be inserted manually into the TwinCAT configuration like any normal Beckhoff device description.
- The sci file must be located in the TwinCAT ESI path, usually under:
C:\TwinCAT\3.1\Config\Io\EtherCAT

	EL3702 with added StartTimeNextLatch.sci	11.01.2021 13:29	SCI-Datei	6 KB
---	--	------------------	-----------	------

- Open the selection dialog:

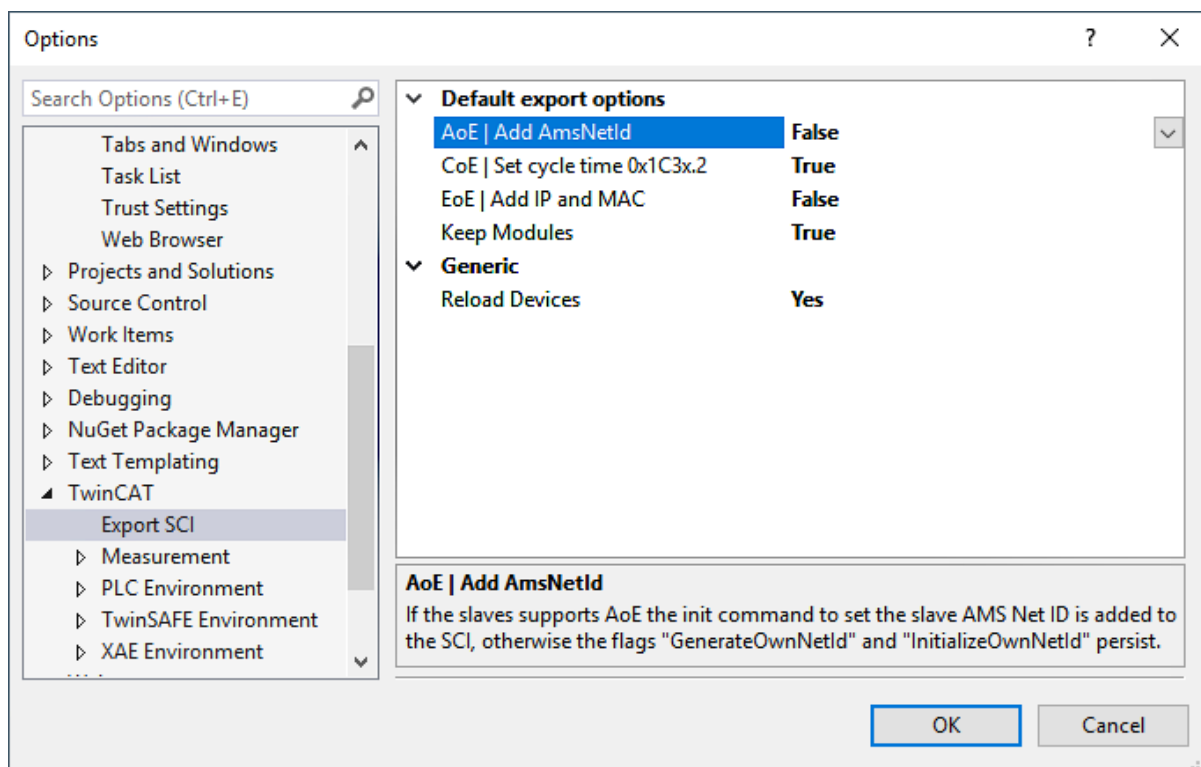


- Display SCI devices and select and insert the desired device:



Additional Notes

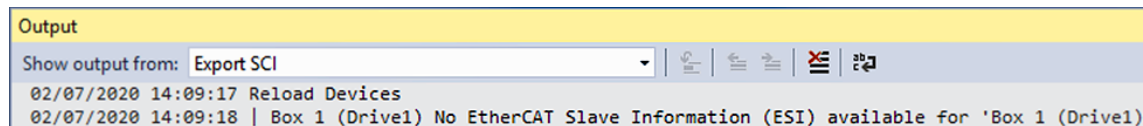
- Settings for the SCI function can be made via the general Options dialog (Tools → Options → TwinCAT → Export SCI):



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:



9.3 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.

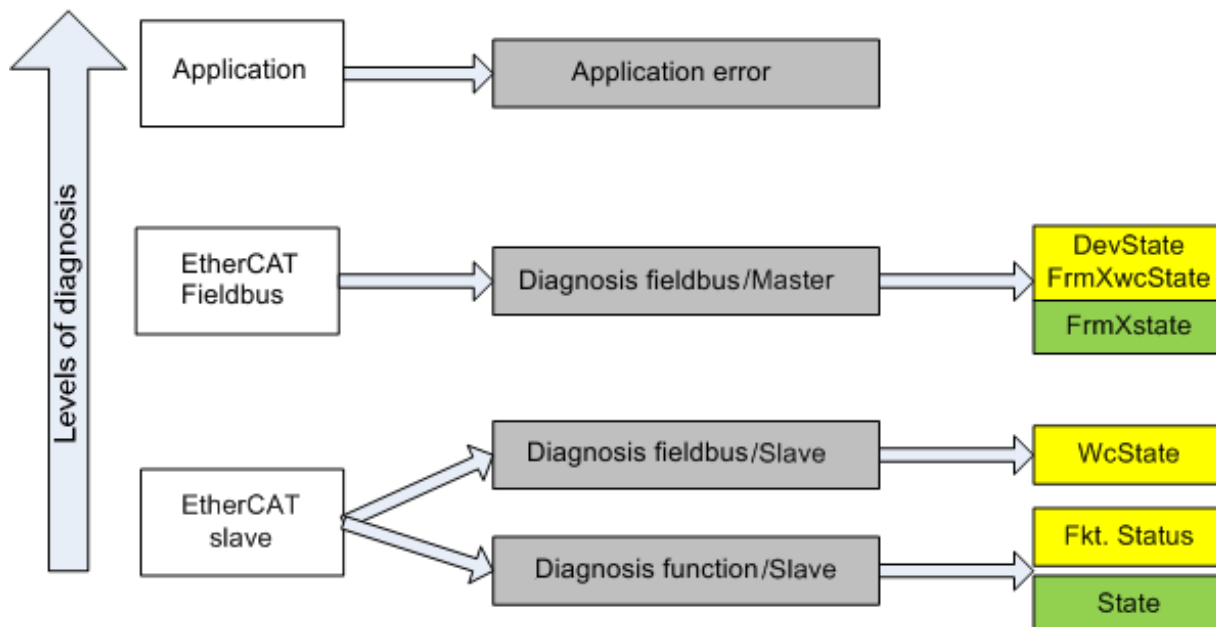


Fig. 170: Selection of the diagnostic information of an EtherCAT Slave

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.

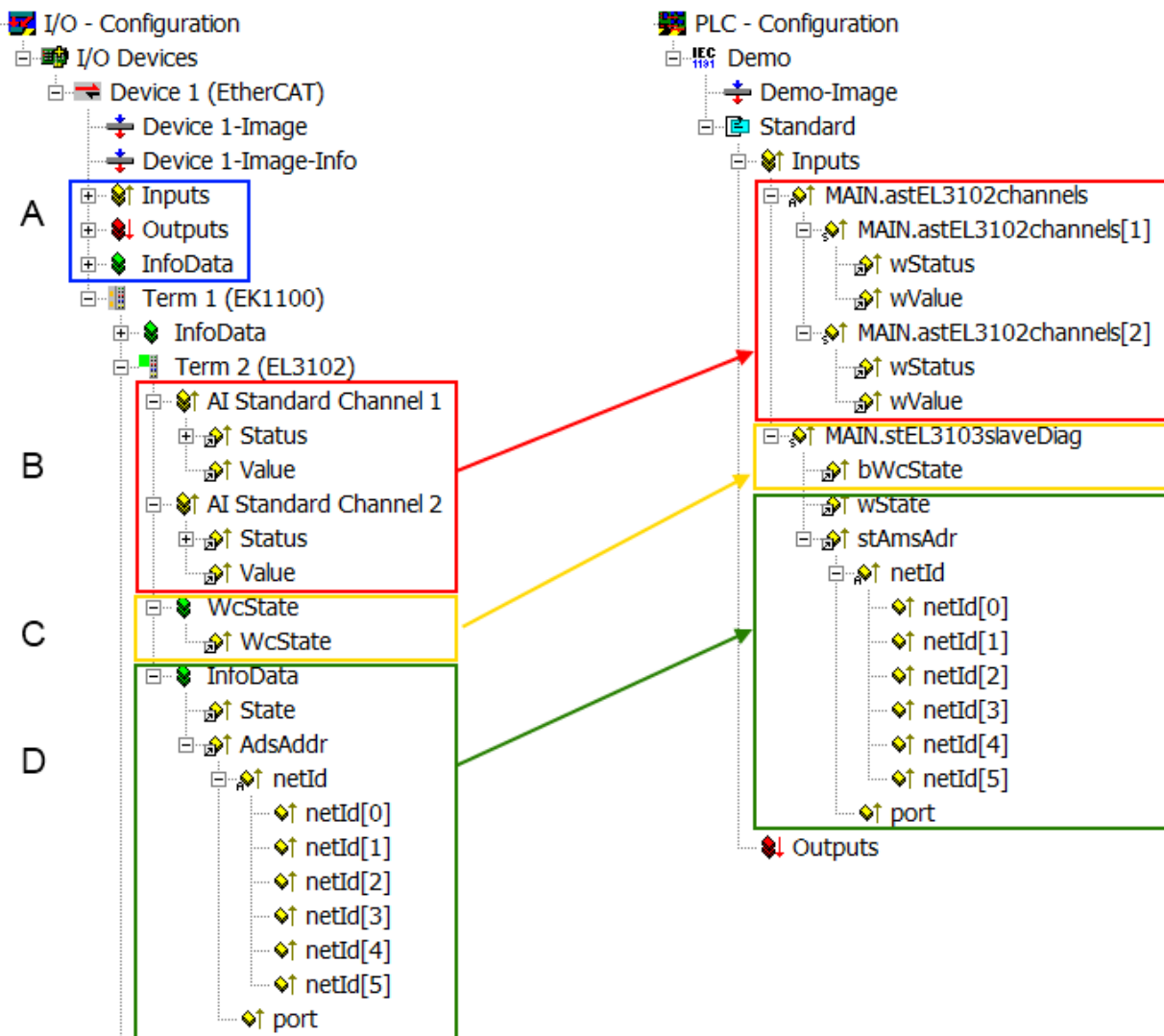


Fig. 171: Basic EtherCAT Slave Diagnosis in the PLC

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 <i>TcEtherCAT.lib</i> • 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. <i>AdsAddr</i> 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 <i>port</i> (= 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*:

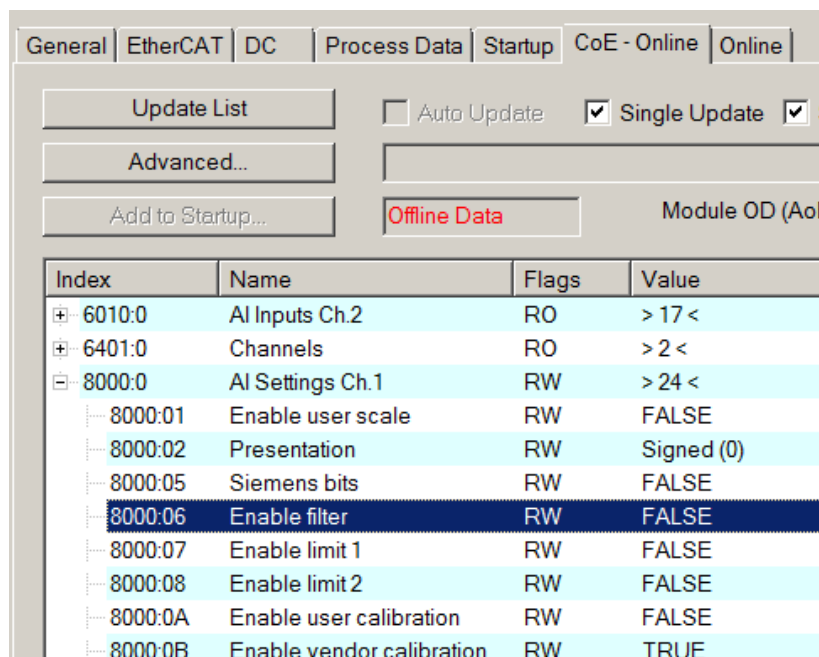


Fig. 172: EL3102, CoE directory

i 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.

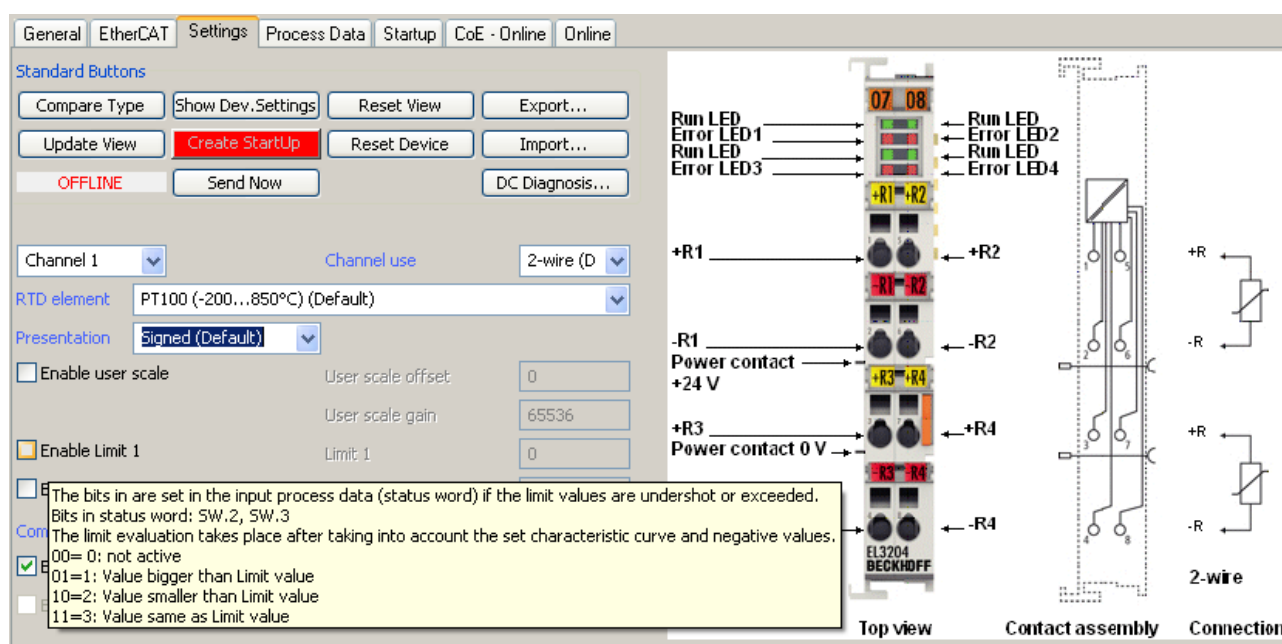


Fig. 173: Example of commissioning aid for a EL3204

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 \[► 29\]](#)" 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.

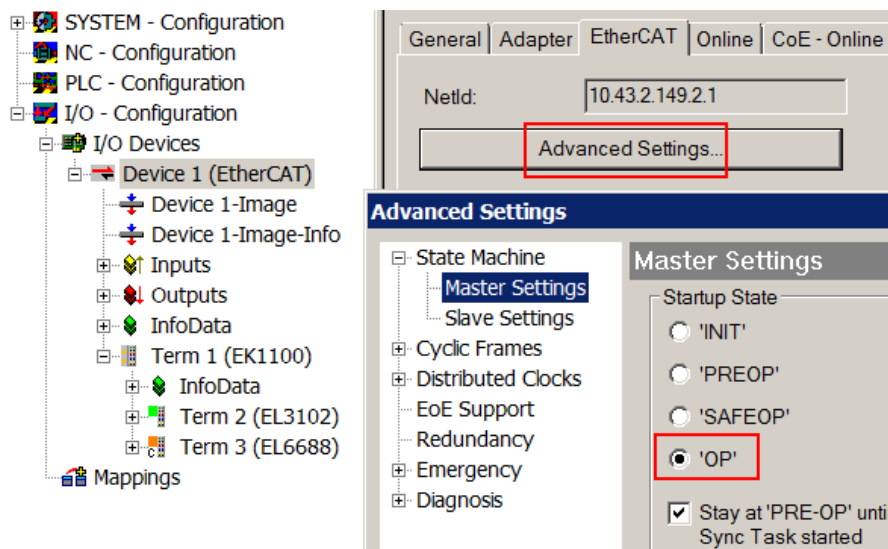


Fig. 174: Default behaviour of the System Manager

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

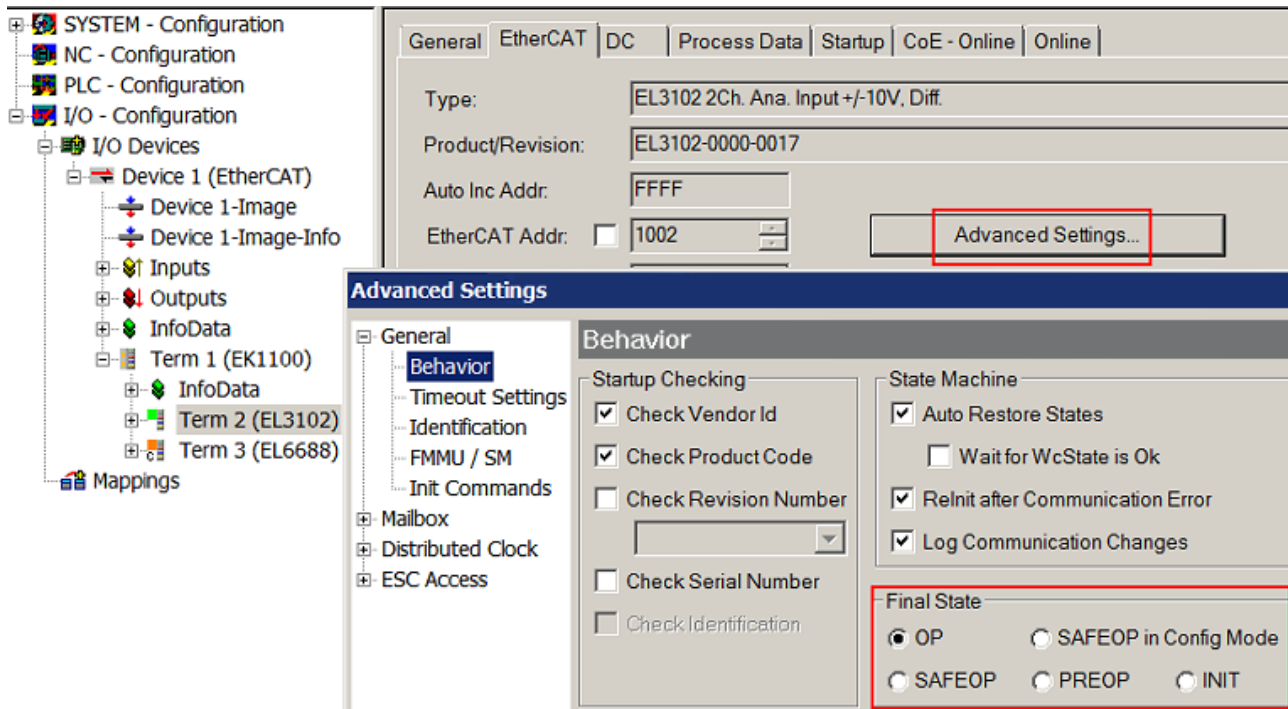


Fig. 175: Default target state in the Slave

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.

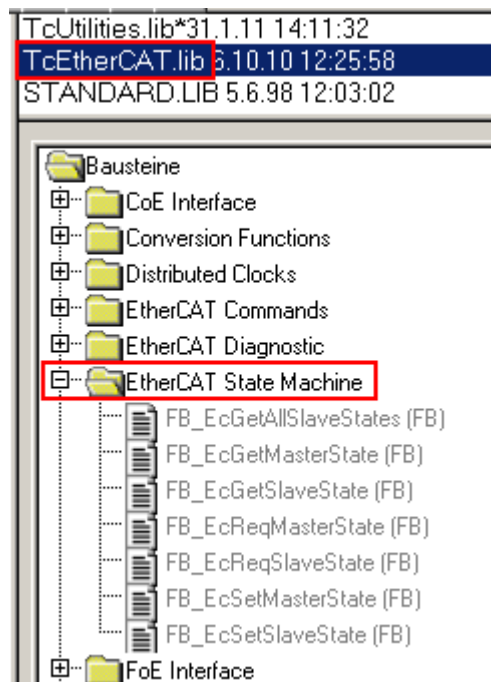


Fig. 176: PLC function blocks

Note regarding E-Bus current

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

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

General Adapter EtherCAT Online CoE - Online						
NetId:		10.43.2.149.2.1		Advanced Settings...		
Number	Box Name	Address	Type	In Size	Out S...	E-Bus (..
1	Term 1 (EK1100)	1001	EK1100			
2	Term 2 (EL3102)	1002	EL3102	8.0		1830
3	Term 4 (EL2004)	1003	EL2004		0.4	1730
4	Term 5 (EL2004)	1004	EL2004		0.4	1630
5	Term 6 (EL7031)	1005	EL7031	8.0	8.0	1510
6	Term 7 (EL2808)	1006	EL2808		1.0	1400
7	Term 8 (EL3602)	1007	EL3602	12.0		1210
8	Term 9 (EL3602)	1008	EL3602	12.0		1020
9	Term 10 (EL3602)	1009	EL3602	12.0		830
10	Term 11 (EL3602)	1010	EL3602	12.0		640
11	Term 12 (EL3602)	1011	EL3602	12.0		450
12	Term 13 (EL3602)	1012	EL3602	12.0		260
13	Term 14 (EL3602)	1013	EL3602	12.0		70
14	Term 3 (EL6688)	1014	EL6688	22.0		-240 !

Fig. 177: 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:

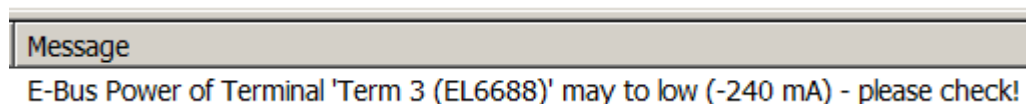


Fig. 178: Warning message for exceeding E-Bus current

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

10 Error handling and diagnostics

10.1 ADS error codes and further error diagnostics

Error codes are generated in the event of an error during ADS access to an IO-Link device.

The possible error codes are listed in tables C.1 and C.2.

Example of an AdsReturnCode

AdsReturnCode 0x**80110700**

- **80**: Device Application Error (IO-Link Spec),
- **11**: Index not Available (IO-Link Spec),
- **0700**: General ADS Error

ErrorTypes (IO-Link Spec)

Incident	Error Code	Additional Code	Name	Definition
Device application error – no details	0x80	0x00	APP_DEV	This ErrorType shall be used if the requested service has been refused by the Device application and no detailed
Index not available	0x80	0x11	IDX_NOTAVAIL	This ErrorType shall be used whenever a read or write access occurs to a not existing Index.
Subindex not available	0x80	0x12	SUBIDX_NOTAVAIL	This ErrorType shall be used whenever a read or write access occurs to a not existing Subindex.
Service temporarily not available	0x80	0x20	SERV_NOTAVAIL	This ErrorType shall be used if a parameter is not accessible for a read or write service due to the current state of the Device application.
Service temporarily not available – local control	0x80	0x21	SERV_NOTAVAIL_LOCTRL	This ErrorType shall be used if a parameter is not accessible for a read or write service due to an ongoing local operation at the Device (for example operation or parameterization via an on-board Device control panel).
Service temporarily not available – Device control	0x80	0x22	SERV_NOTAVAIL_DEVCTRL	This ErrorType shall be used if a read or write service is not accessible due to a remote triggered state of the device application (for example parameterization during a remote triggered teach-in operation or calibration).
Access denied	0x80	0x23	IDX_NOT_WRITEABLE	This ErrorType shall be used if a write service tries to access a read-only parameter.
Parameter value out of range	0x80	0x30	PAR_VALOUTOFRNG	This ErrorType shall be used for a write service to a parameter outside its permitted range of values.
Parameter value above limit	0x80	0x31	PAR_VALGTLM	This ErrorType shall be used for a write service to a parameter above its specified value range.
Parameter value below limit	0x80	0x32	PAR_VALLTLM	This ErrorType shall be used for a write service to a parameter below its specified value range.
Parameter length overrun	0x80	0x33	VAL_LENVERRUN	This ErrorType shall be used when the content of a write service to a parameter is greater than the parameter specified length. This ErrorType shall also be used, if a data object is too large to be processed by the Device application (for example ISDU buffer restriction).
Parameter length underrun	0x80	0x34	VAL_LENUNDRUN	This ErrorType shall be used when the content of a write service to a parameter is less than the parameter specified length (for example write access of an Unsigned16 value to an Unsigned32 parameter).
Function not available	0x80	0x35	FUNC_NOTAVAIL	This ErrorType shall be used for a write service with a command value not supported by the Device application (for example a SystemCommand with a value not implemented).
Function temporarily unavailable	0x80	0x36	FUNC_UNAVAILTEMP	This ErrorType shall be used for a write service with a command value calling a Device function not available due to the current state of the Device application (for example a SystemCommand).
Invalid parameter set	0x80	0x40	PAR_SETINVALID	This ErrorType shall be used if values sent via single parameter transfer are not consistent with other actual parameter settings (for example overlapping set points for a binary data setting)
Inconsistent parameter set	0x80	0x41	PAR_SETINCONSIST	This ErrorType shall be used at the termination of a block parameter transfer with ParamDownloadEnd or ParamDownloadStore if the plausibility check shows inconsistencies
Application not ready	0x80	0x82	APP_DEVNOTRDY	This ErrorType shall be used if a read or write service is refused due to a temporarily unavailable application (for example peripheral controllers during startup).
Vendor specific	0x81	0x00	UNSPECIFIC	This ErrorType will be propagated directly to higher level processing elements as an error (no warning) by the Master.
Vendor specific	0x81	0x01 to 0xFF	VENDOR_SPECIFIC	

Table C.1 ErrorTypes, IO-Link Spec

Derived ErrorTypes (IO-Link Spec)

Incident	Error Code	Additional Code	Name	Definition
Master – Communication error	0x10	0x00	COM_ERR	The Master generates a negative service response with this ErrorType if a communication error occurred during a read or write service, for example the SDCI connection is interrupted.
Master – ISDU timeout	0x11	0x00	I-SERVICE_TIMEOUT	The Master generates a negative service response with this ErrorType, if a Read or Write service is pending longer than the specified I-Service timeout in the Master.
Device Event – ISDU error (DL, Error, single shot, 0x5600)	0x11	0x00	I-SERVICE_TIMEOUT	If the Master received an Event with the EventQualifier and the EventCode 0x5600, a negative service response indicating a service timeout is generated and returned to the requester (Master – ISDU timeout).
Device Event – ISDU illegal service primitive (AL, Error, single shot, 0x5800)	0x11	0x00	I-SERVICE_TIMEOUT	If the Master received an Event with the EventQualifier and the EventCode 0x5800, a negative service response indicating a service timeout is generated and returned to the requester (Master – ISDU timeout).
Master – ISDU checksum error	0x56	0x00	M_ISDU_CHECKSUM	The Master generates a negative service response with this ErrorType, if its data link layer detects an ISDU checksum error.
Master – ISDU illegal service primitive	0x57	0x00	M_ISDU_ILLEGAL	The Master generates a negative service response with this ErrorType, if its data link layer detects an ISDU illegal service primitive.
Device Event – ISDU buffer overflow (DL, Error, single shot, 0x5200)	0x80	0x33	VAL_LEN0VRRUN	If the Master received an Event with the EventQualifier and the EventCode 0x5200, a negative service response indicating a parameter length overrun is generated and returned to the requester (see parameter length overrun) Events from legacy Devices shall be redirected in compatibility mode to this derived ErrorType

*Table C.2 Derived ErrorTypes, IO-Link Spec***Further error diagnosis options****Device State Inputs Device (0x1A05)**

It is indicated in the PDO "Device Diag [► 97]" (0xF101:0D) that at least one event has occurred in the "Diag History".

"Device State [► 97]" is the standard status bit for EtherCAT slaves and shows, for example, that communication with one of the slaves has been interrupted.

Device State Inputs (0x1A04)

The status of the IO-Link devices is displayed at the respective port (see Comment field [► 97] in the System Manager).

Nominal/actual comparison of the parameter objects

The indices 0x90n0 (Info data) can be referred to for validation of the configuration indices 0x80n0 of the connected IO-Link device.

In case of error these objects can be used to compare the configuration with the actual state.

Lost Frame Counter

The Lost Frame counter in object 0xA0n0:02 [► 91] is for the diagnosis of the transmission quality. TwinCAT provides the possibility here to diagnose problems, e. g. with the wiring, EMC or power supply.

11 Appendix

11.1 EtherCAT AL Status Codes

For detailed information please refer to the [EtherCAT system description](#).

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. ELxxx-xxx_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:

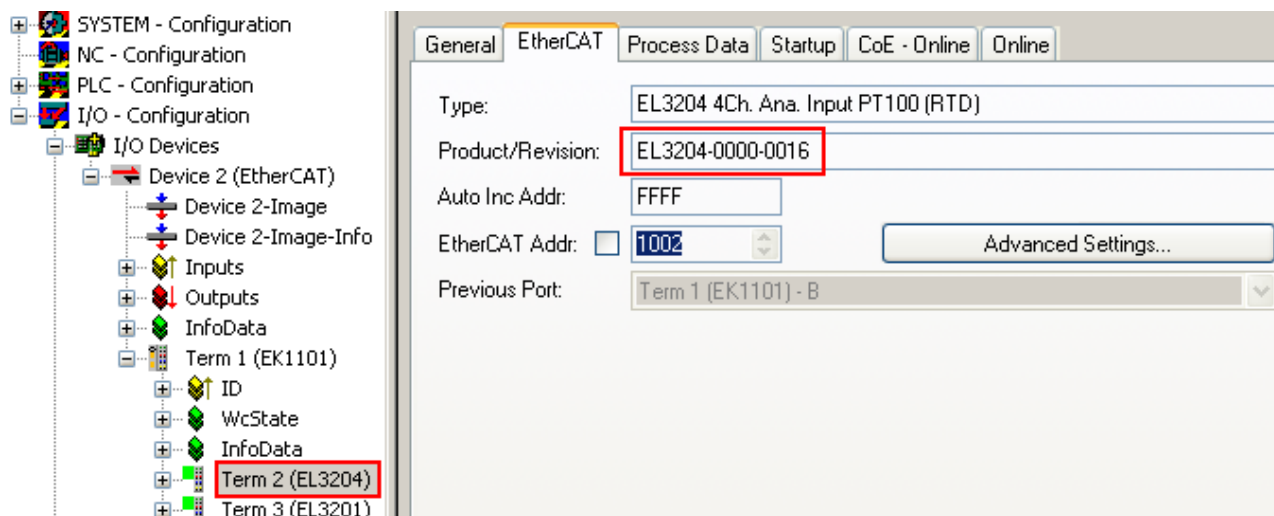


Fig. 179: Device identifier consisting of name EL3204-0000 and revision -0016

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](#).

i 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:

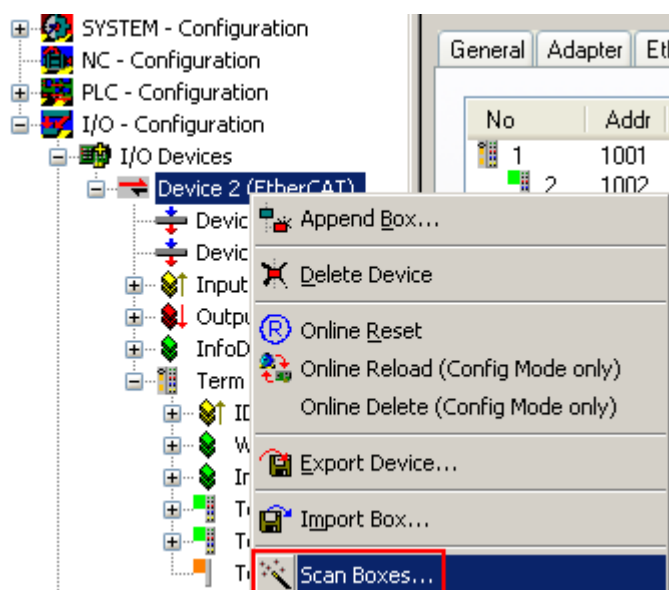


Fig. 180: Scan the subordinate field by right-clicking on the EtherCAT device

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



Fig. 181: Configuration is identical

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

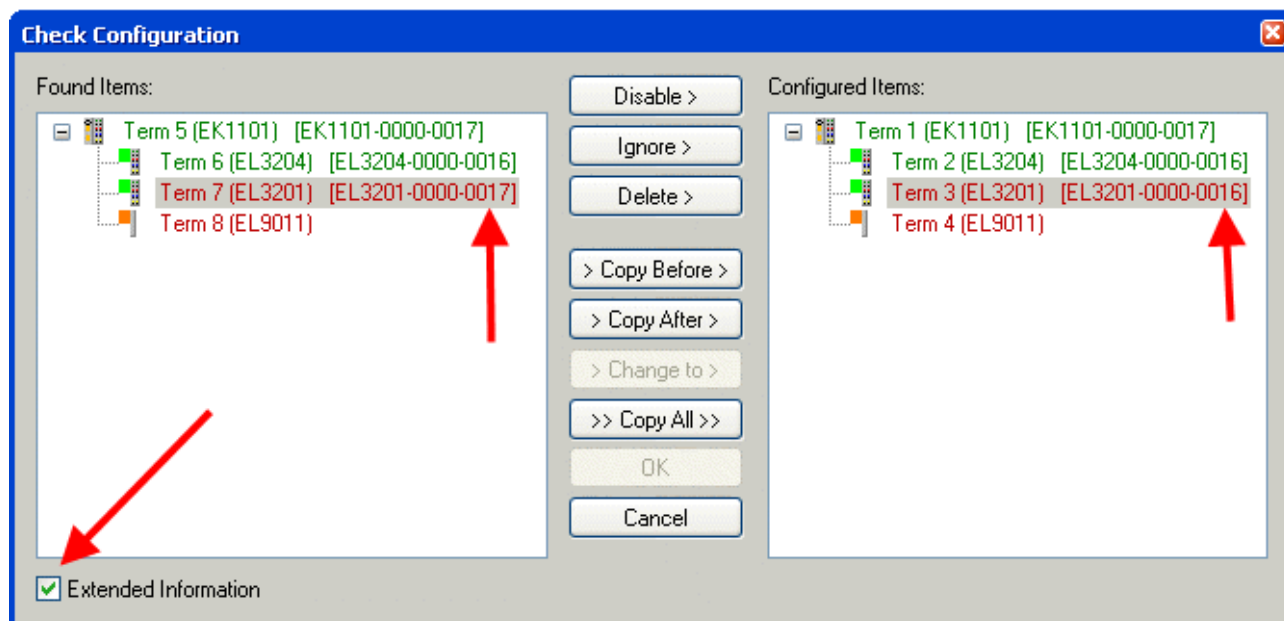


Fig. 182: Change dialog

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*

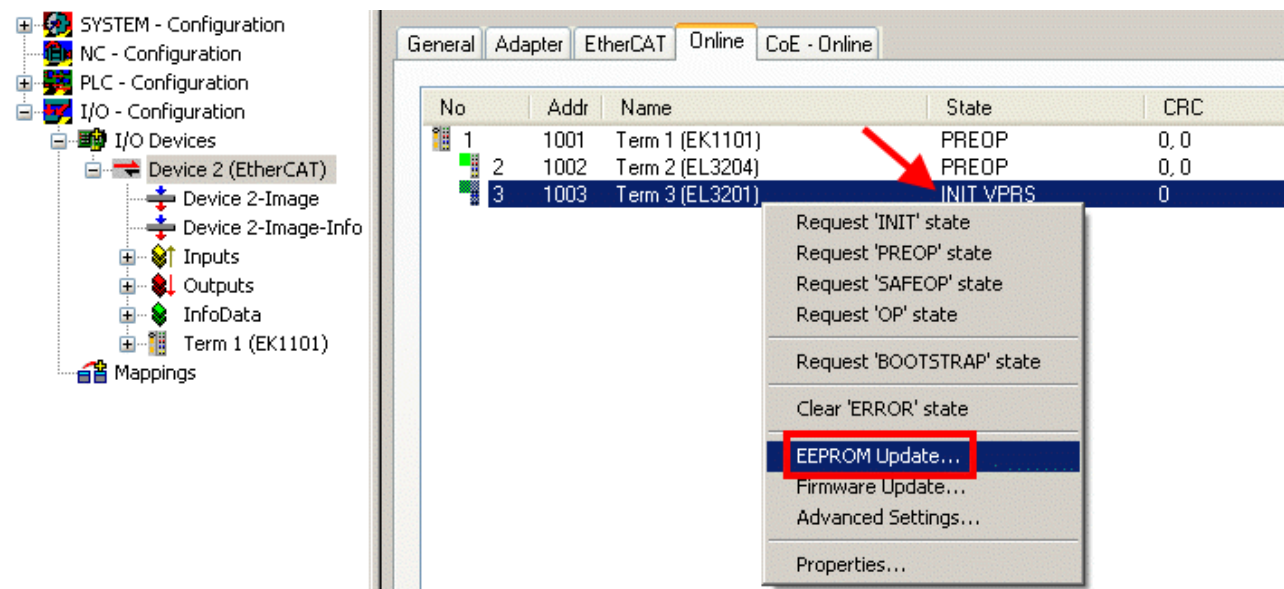


Fig. 183: EEPROM Update

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.

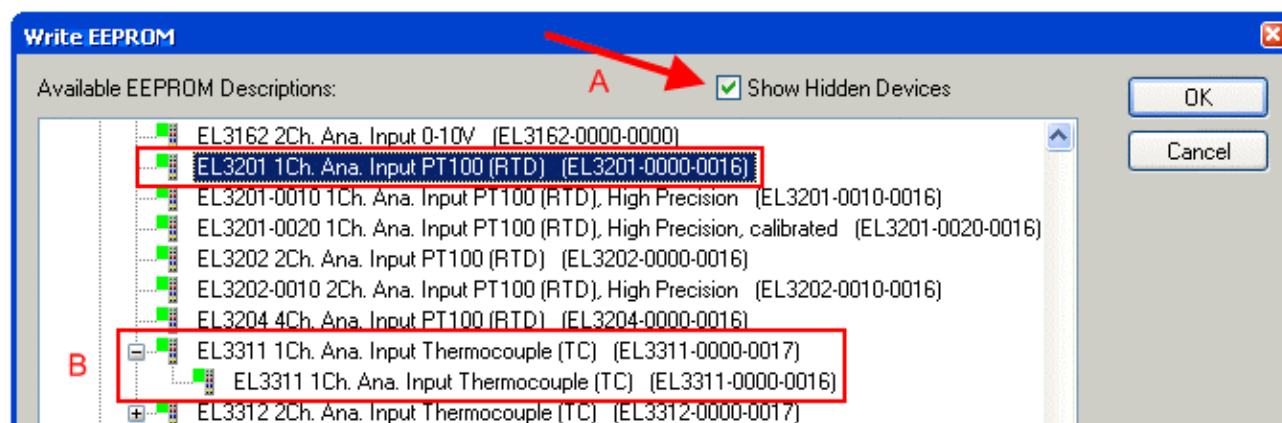


Fig. 184: Selecting the new ESI

A progress bar in the System Manager shows the progress. Data are first written, then verified.

i 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).

i 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.

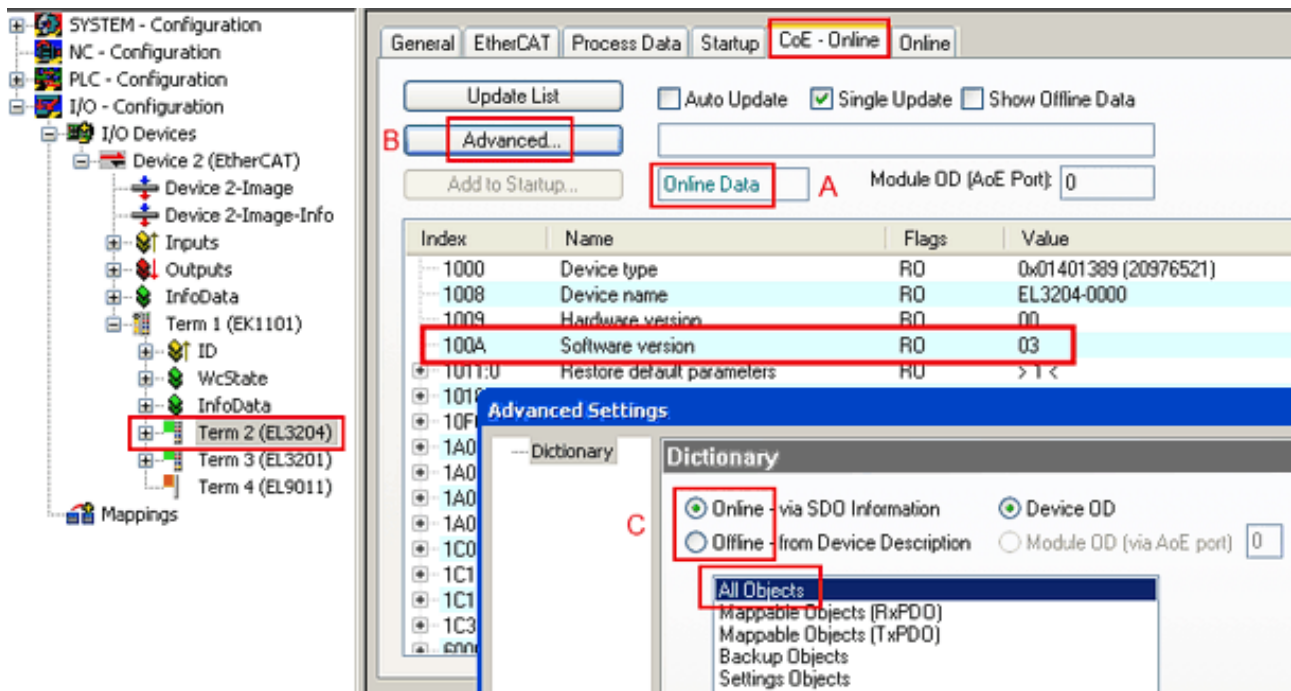


Fig. 185: Display of EL3204 firmware version

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 *All Objects*.

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*.

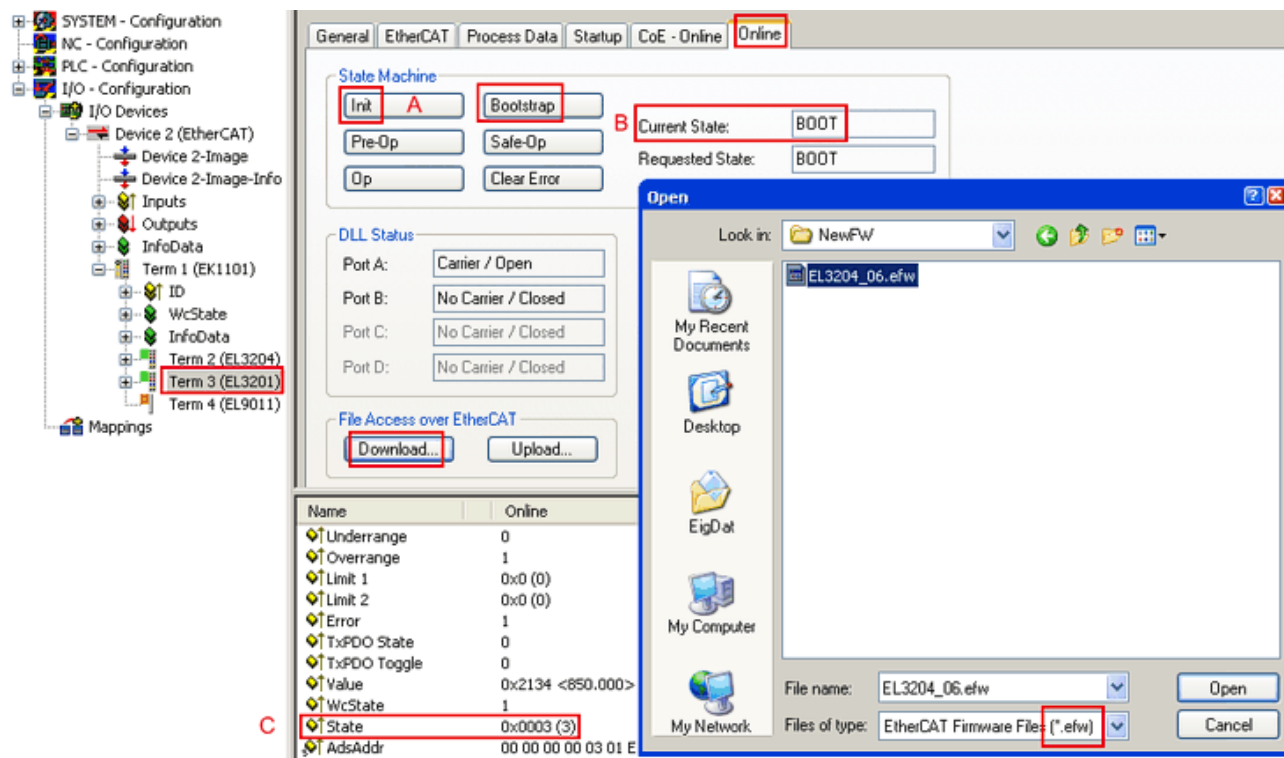
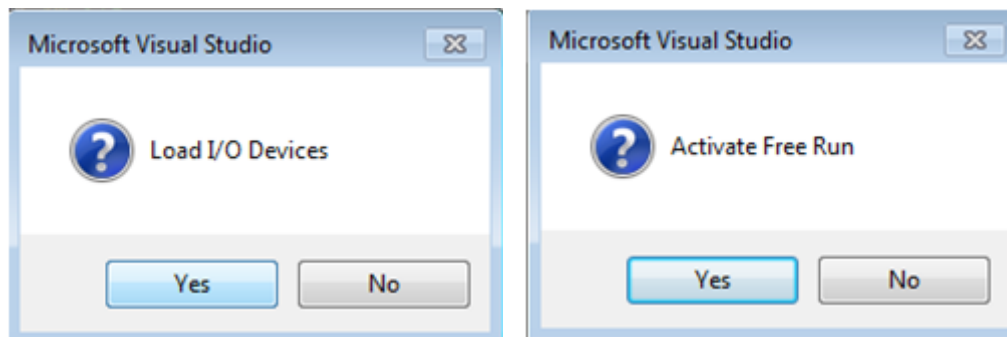


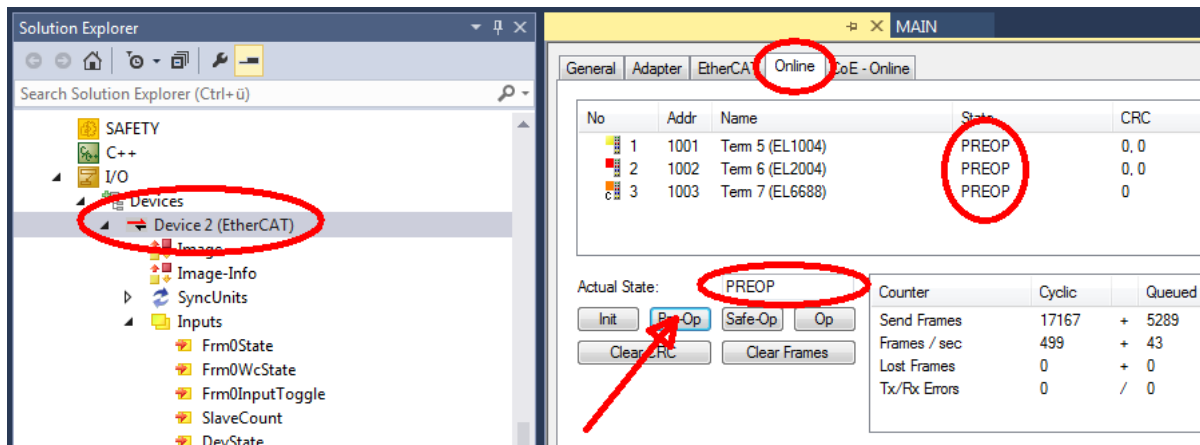
Fig. 186: Firmware Update

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.

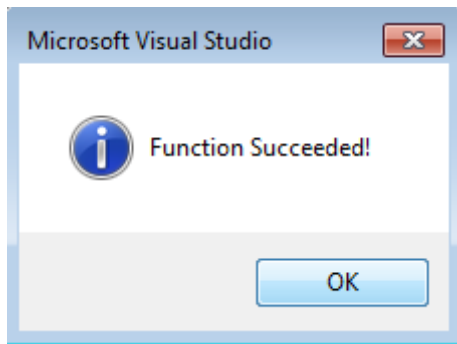


- Switch EtherCAT Master to PreOP



- 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.

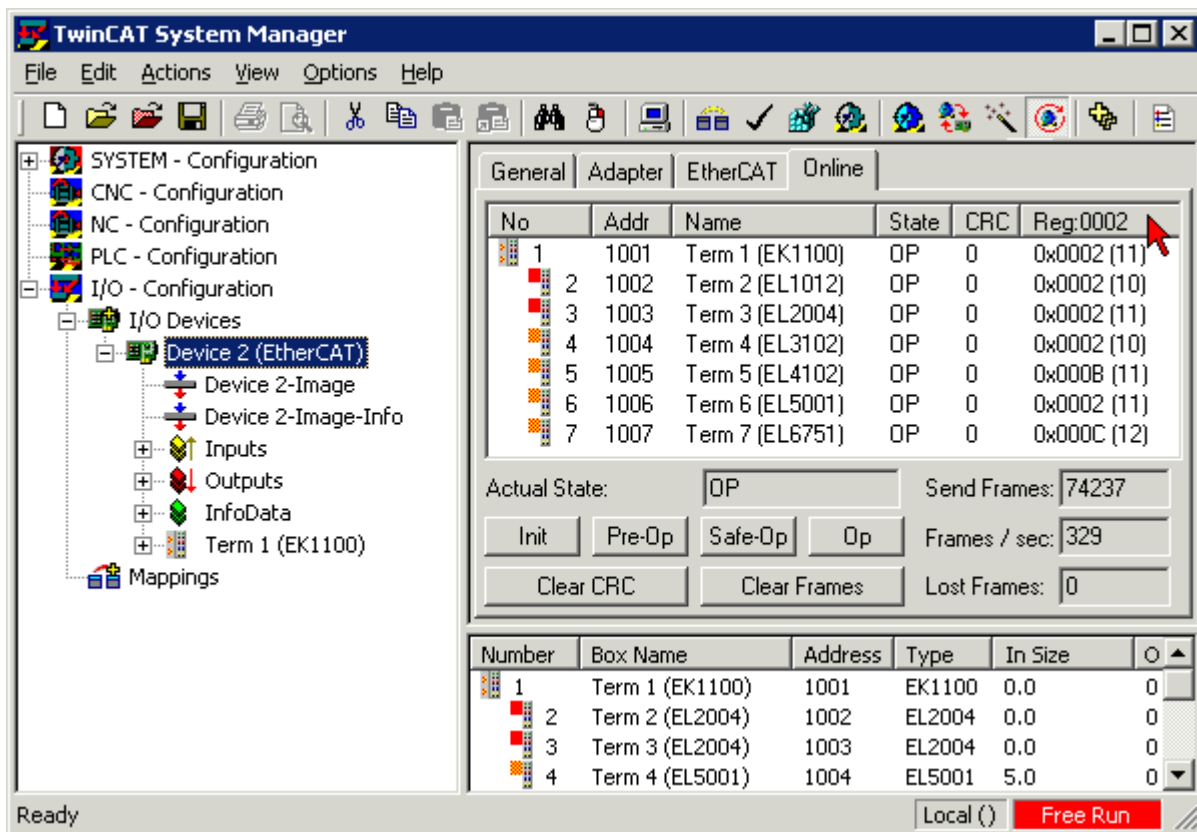
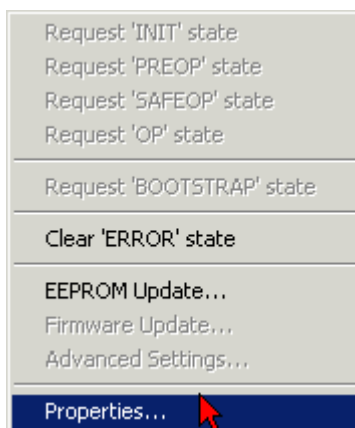


Fig. 187: FPGA firmware version definition

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

Fig. 188: 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.

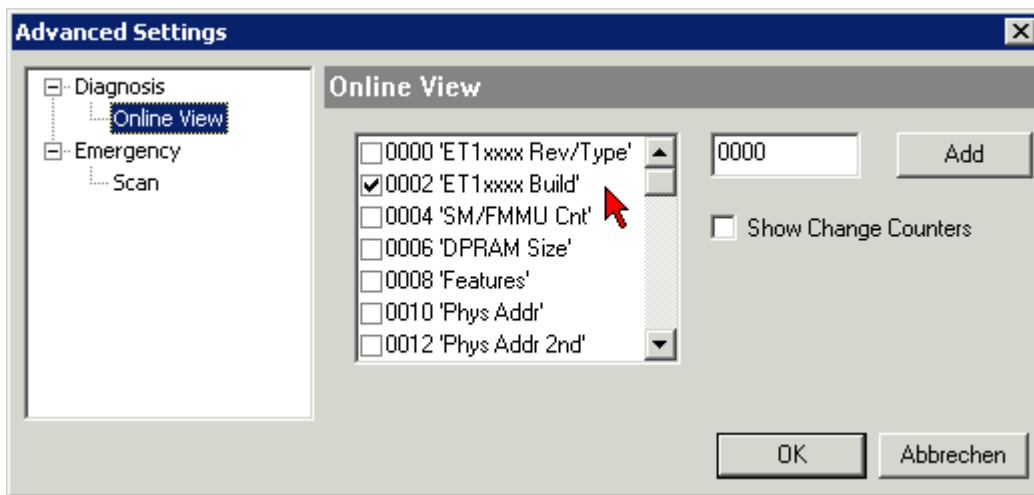


Fig. 189: Dialog *Advanced Settings*

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.

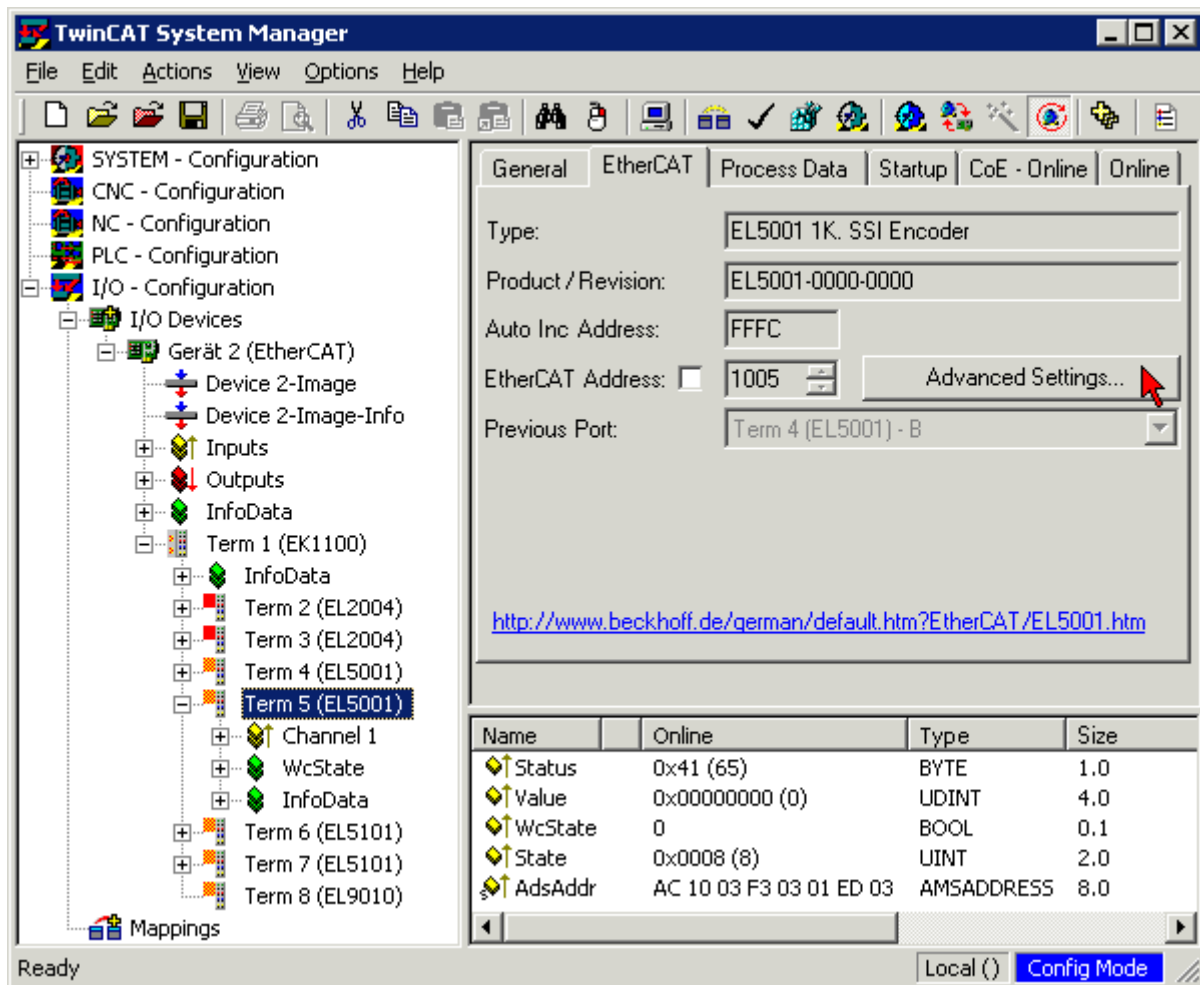
Older firmware versions can only be updated by the manufacturer!

Updating an EtherCAT device

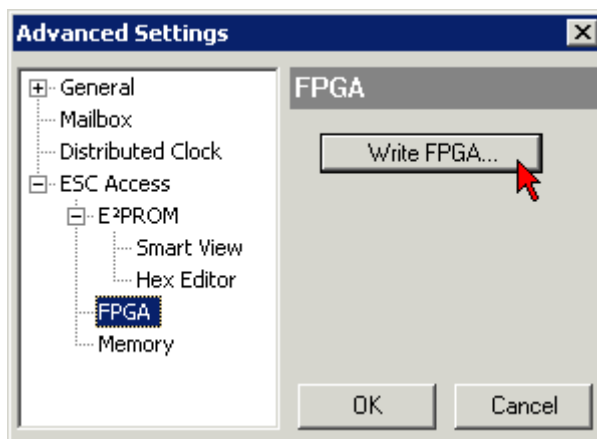
The following sequence order have to be met if no other specifications are given (e.g. by the Beckhoff support):

- 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.

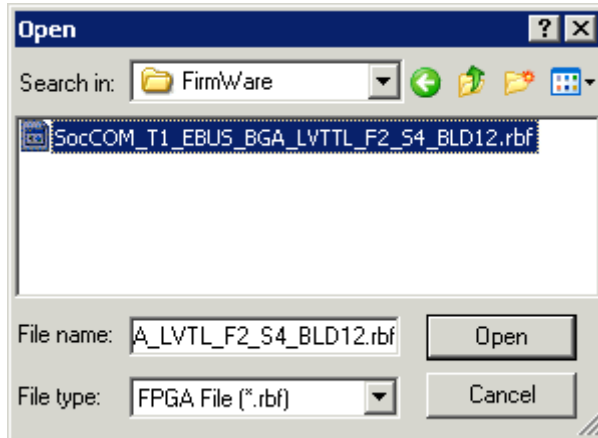
- In the TwinCAT System Manager select the terminal for which the FPGA firmware is to be updated (in the example: Terminal 5: EL5001) and click the *Advanced Settings* button in the *EtherCAT* tab:



- 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.

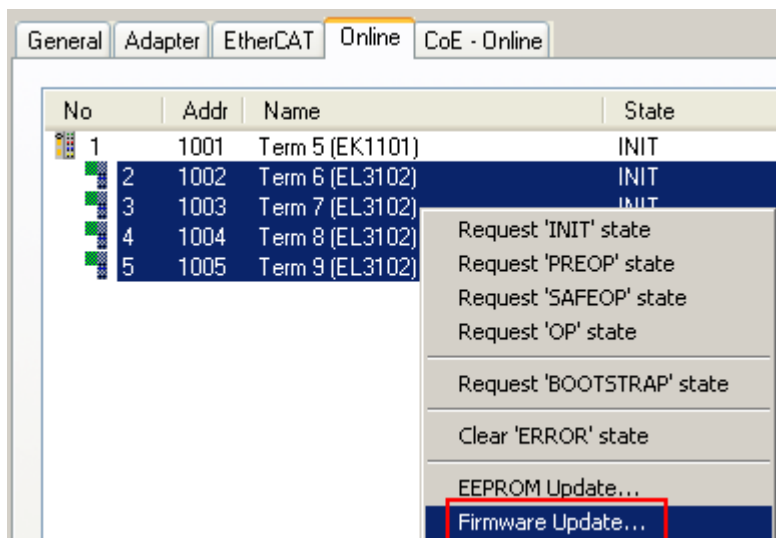


Fig. 190: Multiple selection and firmware update

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

11.3 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 \[► 187\]](#).

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!

EL6224			
Hardware (HW)	Firmware (FW)	Revision no.	Release date
00 - 19	01	EL6224-0000-0016	2008/05
	02	EL6224-0000-0017	2009/07
	03		2010/06
	04		2011/04
	05		2011/06
	06	EL6224-0000-0018	2012/08
		EL6224-0000-0019	2012/09
	07	EL6224-0000-0020	2013/04
	08		2013/06
	09		2014/03
	10	EL6224-0000-0020	2014/10
	11		2015/04
	12		2018/01
20 - 23*	13	EL6224-0000-0021	2019/06
	14		2020/07
	15		2021/12
	16*		2022/03

EL6224-0090			
Hardware (HW)	Firmware (FW)	Revision no.	Release date
16 - 20	01	EL6224-0090-0016	2017/03
	02		2018/04
20 - 23*	03	EL6224-0090-0017	2020/01
	04*		2022/03

*) 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.

11.4 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*).

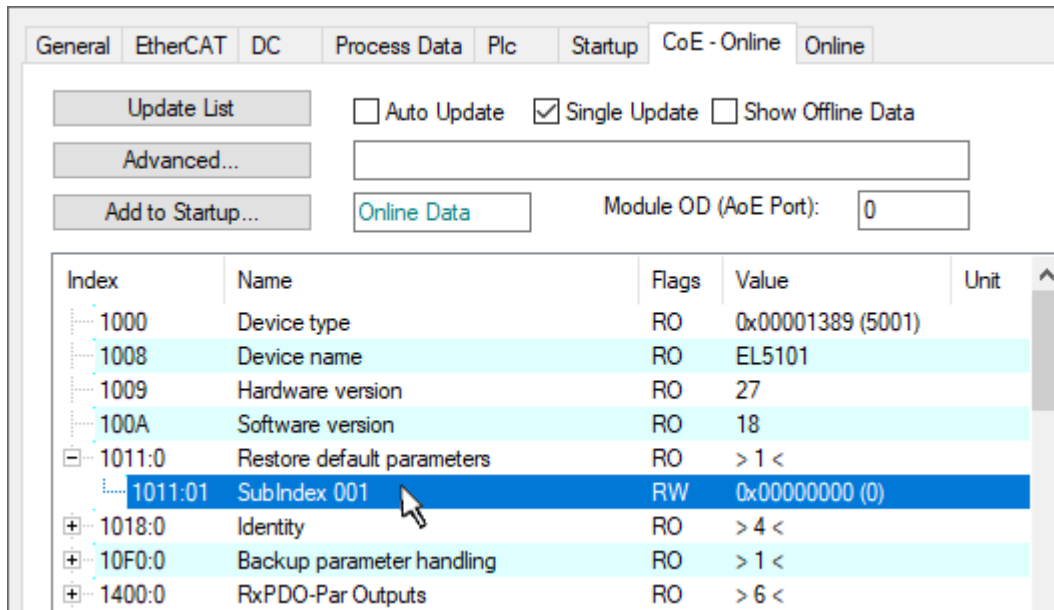


Fig. 191: Selecting the *Restore default parameters* PDO

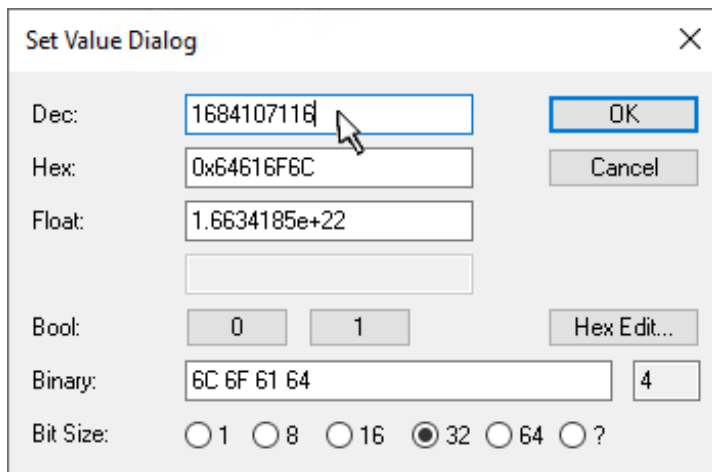


Fig. 192: Entering a restore value in the Set Value dialog

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.

11.5 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on her internet pages: www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Support

The Beckhoff Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline: +49 5246 963 157
e-mail: support@beckhoff.com
web: www.beckhoff.com/support

Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline: +49 5246 963 460
e-mail: service@beckhoff.com
web: www.beckhoff.com/service

Headquarters Germany

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Germany

Phone: +49 5246 963 0
e-mail: info@beckhoff.com
web: www.beckhoff.com

Trademark statements

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

Third-party trademark statements

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.

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

More Information:
www.beckhoff.com/EL6224

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

