



Documentation

EL6224-00x0

IO-Link Terminal

Version: 2.7
Date: 2019-03-15

BECKHOFF

Table of contents

1 Foreword	7
1.1 IO-Link Terminals Product Overview	7
1.2 Notes on the documentation	7
1.3 Safety instructions	9
1.4 Documentation issue status	10
1.5 Version identification of EtherCAT devices	11
2 Product overview	15
2.1 Introduction	15
2.2 Technical data	17
2.3 Start	17
3 Basics communication	18
3.1 EtherCAT basics	18
3.2 EtherCAT cabling – wire-bound	18
3.3 General notes for setting the watchdog	19
3.4 EtherCAT State Machine	21
3.5 CoE Interface	23
3.6 Distributed Clock	28
4 Mounting and wiring	29
4.1 Instructions for ESD protection	29
4.2 Notes on TwinSAFE SC	29
4.2.1 Safety instructions	29
4.2.2 Environmental conditions	29
4.2.3 Transport / storage	30
4.2.4 Control cabinet / terminal box	30
4.3 Installation on mounting rails	30
4.4 Installation instructions for enhanced mechanical load capacity	34
4.5 Connection	34
4.5.1 Connection system	34
4.5.2 Wiring	37
4.5.3 Shielding	38
4.6 Positioning of passive Terminals	39
4.7 Installation positions	39
4.8 ATEX - Special conditions (extended temperature range)	42
4.9 ATEX Documentation	43
4.10 EL6224 - LEDs and connection (HD housing)	44
5 Commissioning	45
5.1 TwinCAT Quick Start	45
5.1.1 TwinCAT 2	48
5.1.2 TwinCAT 3	58
5.2 TwinCAT Development Environment	70
5.2.1 Installation of the TwinCAT real-time driver	70
5.2.2 Notes regarding ESI device description	76
5.2.3 TwinCAT ESI Updater	80

5.2.4	Distinction between Online and Offline	80
5.2.5	OFFLINE configuration creation	81
5.2.6	ONLINE configuration creation	86
5.2.7	EtherCAT subscriber configuration	94
5.3	General Notes - EtherCAT Slave Application	103
5.4	IO-Link basics	111
5.4.1	IO-Link system layout	111
5.4.2	Establishment of IO Link communication	112
5.4.3	Device description IODD	113
5.4.4	Parameter server	113
5.4.5	Data transfer rate	113
5.5	Configuration of the IO link master	114
5.5.1	Integrating IO link devices	115
5.5.2	Removal of the IO-Link devices	117
5.5.3	Configuration of the IO-Link ports as digital inputs/outputs	117
5.6	Parameterization of the IO-Link devices	118
5.7	Access to IO-Link data	121
5.7.1	IO-Link system communication	121
5.7.2	PDO Assignment	122
5.7.3	Accessing IO-Link parameters	123
5.7.4	Parameter data exchange	124
5.7.5	ADS	125
5.7.6	Access to events	126
5.8	TwinSAFE SC	126
5.8.1	TwinSAFE SC - operating principle	126
5.8.2	TwinSAFE SC - configuration	127
5.9	TwinSAFE SC process data EL6224-0090	131
5.10	Object description and parameterization	132
5.10.1	Objects for commissioning	132
5.10.2	Standard objects (0x1000-0x1FFF)	134
5.10.3	Profile-specific objects (0x6000-0xFFFF)	139
5.10.4	Objects TwinSAFE Single Channel (EL6224-0090)	142
6	Error handling and diagnostics	145
6.1	ADS error codes and further error diagnostics	145
7	Appendix	148
7.1	EtherCAT AL Status Codes	148
7.2	UL notice	148
7.3	Firmware Update EL/ES/EM/ELM/EPxxxx	149
7.3.1	Device description ESI file/XML	150
7.3.2	Firmware explanation	153
7.3.3	Updating controller firmware *.efw	154
7.3.4	FPGA firmware *.rbf	155
7.3.5	Simultaneous updating of several EtherCAT devices	159
7.4	Firmware compatibility	160
7.5	Restoring the delivery state	160

7.6	Support and Service	161
-----	---------------------------	-----

1 Foreword

1.1 IO-Link Terminals Product Overview

EL6224 [▶ 15]	IO-Link Terminal
EL6224-0090 [▶ 16]	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.

It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

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®, TwinCAT®, EtherCAT®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC® 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.

Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents: EP1590927, EP1789857, DE102004044764, DE102007017835 with corresponding applications or registrations in various other countries.

The TwinCAT Technology is covered, including but not limited to the following patent applications and patents: EP0851348, US6167425 with corresponding applications or registrations in various other countries.



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.

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.

Description of instructions

In this documentation the following instructions are used.
These instructions must be read carefully and followed without fail!

DANGER

Serious risk of injury!

Failure to follow this safety instruction directly endangers the life and health of persons.

WARNING

Risk of injury!

Failure to follow this safety instruction endangers the life and health of persons.

CAUTION

Personal injuries!

Failure to follow this safety instruction can lead to injuries to persons.

NOTE

Damage to environment/equipment or data loss

Failure to follow this instruction can lead to environmental damage, equipment damage or data loss.



Tip or pointer

This symbol indicates information that contributes to better understanding.

1.4 Documentation issue status

Version	Comment
2.7	<ul style="list-style-type: none"> • Update chapter "Technical data" • Update chapter "TwinSAFE SC" • Update chapter "Connection system" -> "Connection" • Chapter "ATEX - Special conditions (standard temperature range)" removed Chapter "ATEX - Special conditions (extended temperature range)" added • Correction in chapter "Objects TwinSAFE Single Channel (EL6224-0090)" • Update structure • Update revision status
2.6	<ul style="list-style-type: none"> • Addenda EL6224-0090 • Update structure
2.5	<ul style="list-style-type: none"> • Update chapter "Commissioning" • Update structure
2.4	<ul style="list-style-type: none"> • Update chapter "Technical data" • Addenda chapter "Instructions for ESD protection" • Addenda chapter "ATEX - Special conditions (standard temperature range)" and notice "ATEX documentation" • Update revision status
2.3	<ul style="list-style-type: none"> • Correction of Technical data
2.2	<ul style="list-style-type: none"> • Update chapter "Object description" • Update chapter "Technical data" • Update structure
2.1	<ul style="list-style-type: none"> • Update chapter "Access to IO-Link data" • Update revision status • Update structure
2.0	<ul style="list-style-type: none"> • Migration • Update structure
1.6	<ul style="list-style-type: none"> • Addendum chapter "Configuration of the IO-Link master" • Addendum chapter "Parameterization of the IO-Link devices " • Addendum chapter "Access to IO-Link data" • Update chapter "IO-Link principles" • Update chapter "Object description" • Update chapter "Technical data" • Update chapter "ADS error codes" • Update structure • Update revision status
1.5	<ul style="list-style-type: none"> • Update chapter "IO-Link principles" • Update chapter "Object description" • Update chapter "Basic function principles" • Update chapter "Technical data" • Update chapter "ADS error codes" • Update structure
1.4	<ul style="list-style-type: none"> • Update chapter "Technical data"
1.3	<ul style="list-style-type: none"> • Addendum to firmware compatibility, new structure
1.2	<ul style="list-style-type: none"> • Addendum to new housing
1.1	<ul style="list-style-type: none"> • Addenda & corrections
1.0	<ul style="list-style-type: none"> • First public issue
0.2	<ul style="list-style-type: none"> • Corrections
0.1	<ul style="list-style-type: none"> • Provisional documentation for EL6224

1.5 Version identification of EtherCAT devices

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. "EL5021 EL terminal, standard IP20 IO device with batch number and revision ID (since 2014/01)".
- The type, version and revision are read as decimal numbers, even if they are technically saved in hexadecimal.

Identification number

Beckhoff EtherCAT devices from the different lines have different kinds of identification numbers:

Production lot/batch number/serial number/date code/D number

The serial number 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

Ser. no.: 12063A02: 12 - production week 12 06 - production year 2006 3A - firmware version 3A 02 - hardware version 02

Exceptions can occur in the **IP67 area**, where the following syntax can be used (see respective device documentation):

Syntax: D ww yy x y z u

D - prefix designation

ww - calendar week

yy - year

x - firmware version of the bus PCB

y - hardware version of the bus PCB

z - firmware version of the I/O PCB

u - hardware version of the I/O PCB

Example: D.22081501 calendar week 22 of the year 2008 firmware version of bus PCB: 1 hardware version of bus PCB: 5 firmware version of I/O PCB: 0 (no firmware necessary for this PCB) hardware version of I/O PCB: 1

Unique serial number/ID, ID number

In addition, in some series each individual module has its own unique serial number.

See also the further documentation in the area

- IP67: [EtherCAT Box](#)
- Safety: [TwinSafe](#)
- Terminals with factory calibration certificate and other measuring terminals

Examples of markings



Fig. 1: EL5021 EL terminal, standard IP20 IO device with serial/ batch number and revision ID (since 2014/01)



Fig. 2: EK1100 EtherCAT coupler, standard IP20 IO device with serial/ batch number



Fig. 3: CU2016 switch with serial/ batch number

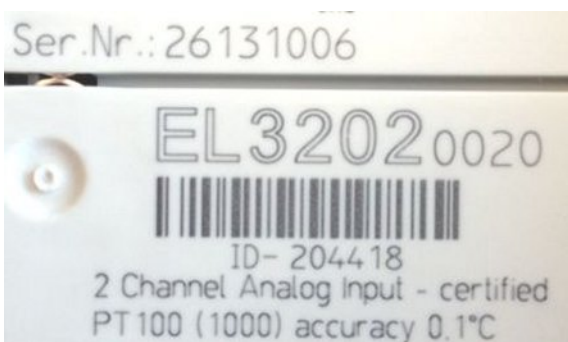


Fig. 4: EL3202-0020 with serial/ batch number 26131006 and unique ID-number 204418



Fig. 5: EP1258-0001 IP67 EtherCAT Box with batch number/ date code 22090101 and unique serial number 158102

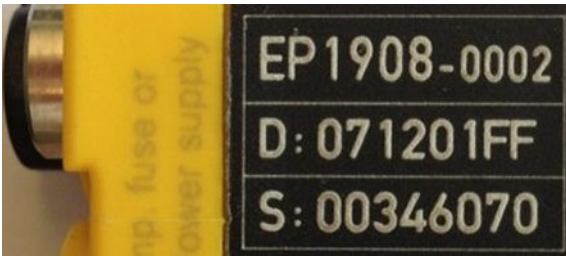


Fig. 6: EP1908-0002 IP67 EtherCAT Safety Box with batch number/ date code 071201FF and unique serial number 00346070



Fig. 7: EL2904 IP20 safety terminal with batch number/ date code 50110302 and unique serial number 00331701



Fig. 8: ELM3604-0002 terminal with unique ID number (QR code) 100001051 and serial/ batch number 44160201

2 Product overview

2.1 Introduction

EL6224 – IO-Link Terminal

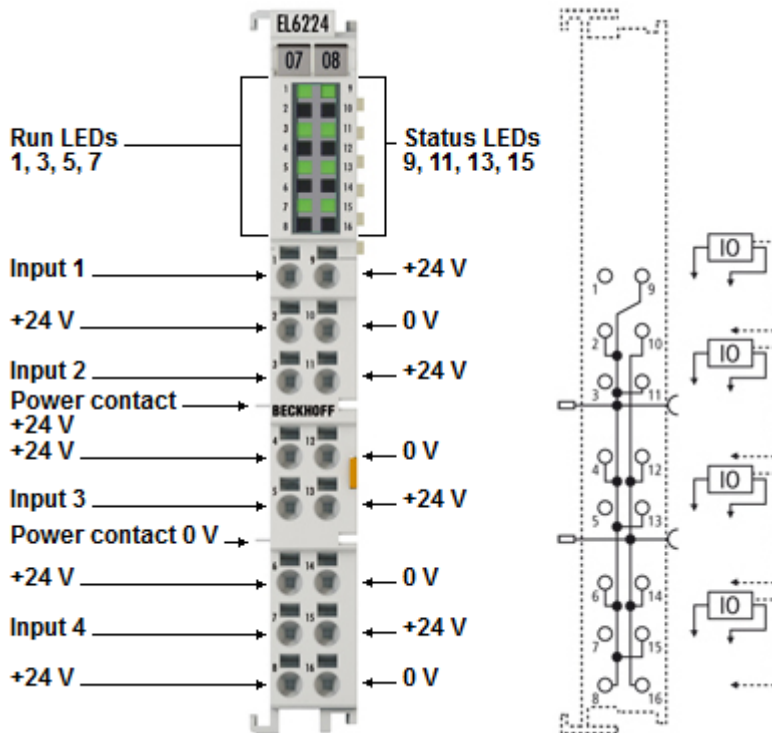


Fig. 9: EL6224

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.

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.

The direct plug-in technique enables toolless construction.

EL6224-0090 – IO-Link Terminal, TwinSAFE Single Channel

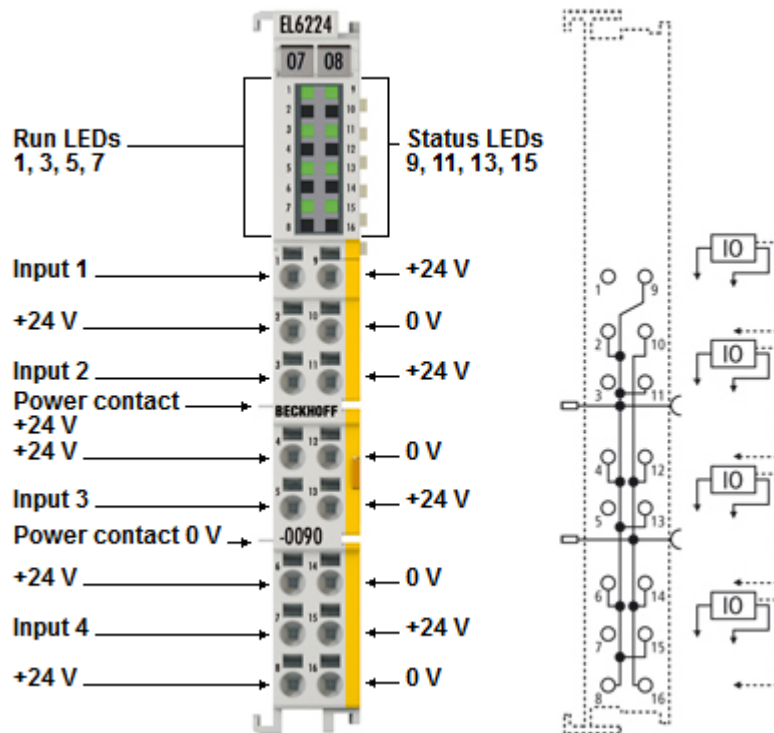


Fig. 10: EL6224-0090

The EL6224-0090 IO-Link terminal enables connection of up to four IO-Link devices, e.g. actuators, sensors or combinations of both. The parameterization of the IO-Link devices with service data can be done from TwinCAT via ADS or very conveniently via the integrated IO-Link configuration tool. IO-Link devices can be operated in 3-wire connection mode.

With the aid of the [TwinSAFE SC technology \[► 126\]](#) (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 \[► 111\]](#)
- [Basic functional principles EL6224 \[► 114\]](#)
- [Object description and parameterization \[► 132\]](#)
- [ADS Error Codes \[► 145\]](#)

2.2 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	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 [▶ 30]	on 35 mm mounting rail conforms to EN 60715	
Vibration/shock resistance	conforms to EN 60068-2-6/EN 60068-2-27, see also Installation instructions [▶ 34] for terminals with increased mechanical load capacity	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	
Approval	CE cULus [▶ 148] ATEX [▶ 42]	

2.3 Start

For commissioning:

- mount the EL6224 as described in the chapter [Mounting and wiring \[\[▶ 29\]\(#\)\]](#)
- configure the EL6224 in TwinCAT as described in the chapter [Commissioning \[\[▶ 45\]\(#\)\]](#).

3 Basics communication

3.1 EtherCAT basics

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

3.2 EtherCAT cabling – wire-bound

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

Cables and connectors

For connecting EtherCAT devices only Ethernet connections (cables + plugs) that meet the requirements of at least category 5 (Cat5) according to EN 50173 or ISO/IEC 11801 should be used. EtherCAT uses 4 wires for signal transfer.

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

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

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

● Recommended cables

i Suitable cables for the connection of EtherCAT devices can be found on the [Beckhoff website!](#)

E-Bus supply

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

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

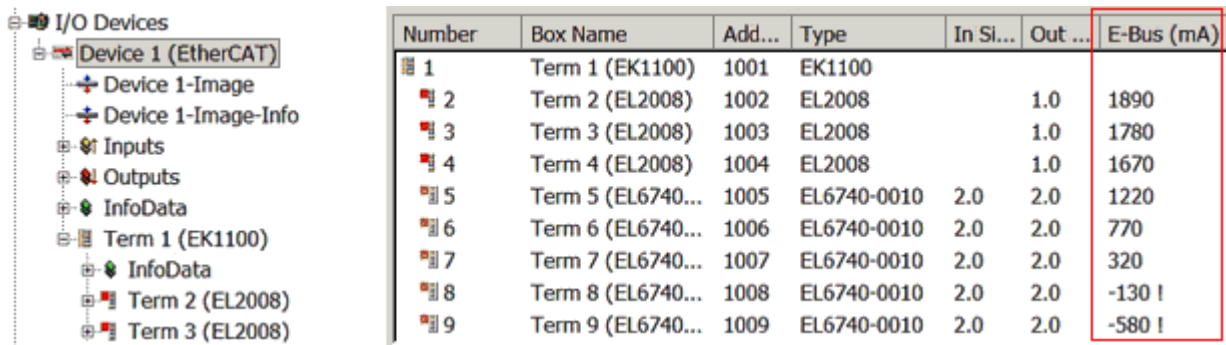


Fig. 11: System manager current calculation

NOTE

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

3.3 General notes for setting the watchdog

ELxxxx terminals are equipped with a safety feature (watchdog) that switches off the outputs after a specifiable time e.g. in the event of an interruption of the process data traffic, depending on the device and settings, e.g. in OFF state.

The EtherCAT slave controller (ESC) in the EL2xxx terminals features 2 watchdogs:

- SM watchdog (default: 100 ms)
- PDI watchdog (default: 100 ms)

SM watchdog (SyncManager Watchdog)

The SyncManager watchdog is reset after each successful EtherCAT process data communication with the terminal. If no EtherCAT process data communication takes place with the terminal for longer than the set and activated SM watchdog time, e.g. in the event of a line interruption, the watchdog is triggered and the outputs are set to FALSE. The OP state of the terminal is unaffected. The watchdog is only reset after a successful EtherCAT process data access. Set the monitoring time as described below.

The SyncManager watchdog monitors correct and timely process data communication with the ESC from the EtherCAT side.

PDI watchdog (Process Data Watchdog)

If no PDI communication with the EtherCAT slave controller (ESC) takes place for longer than the set and activated PDI watchdog time, this watchdog is triggered.

PDI (Process Data Interface) is the internal interface between the ESC and local processors in the EtherCAT slave, for example. The PDI watchdog can be used to monitor this communication for failure.

The PDI watchdog monitors correct and timely process data communication with the ESC from the application side.

The settings of the SM- and PDI-watchdog must be done for each slave separately in the TwinCAT System Manager.

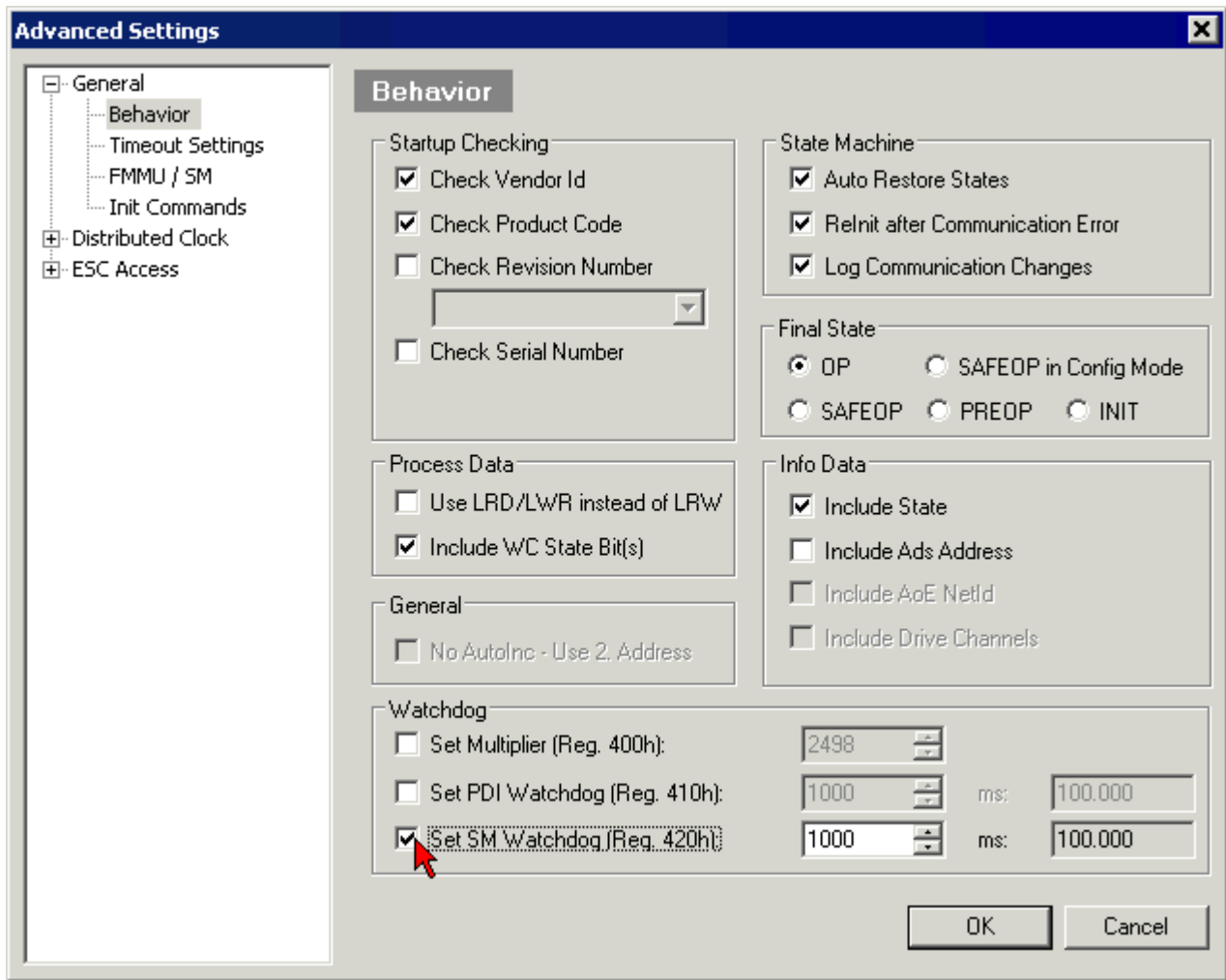


Fig. 12: EtherCAT tab -> Advanced Settings -> Behavior -> Watchdog

Notes:

- the multiplier is valid for both watchdogs.
- each watchdog has its own timer setting, the outcome of this in summary with the multiplier is a resulting time.
- Important: the multiplier/timer setting is only loaded into the slave at the start up, if the checkbox is activated.
If the checkbox is not activated, nothing is downloaded and the ESC settings remain unchanged.

Multiplier

Multiplier

Both watchdogs receive their pulses from the local terminal cycle, divided by the watchdog multiplier:

$$1/25 \text{ MHz} * (\text{watchdog multiplier} + 2) = 100 \text{ } \mu\text{s} \text{ (for default setting of 2498 for the multiplier)}$$

The standard setting of 1000 for the SM watchdog corresponds to a release time of 100 ms.

The value in multiplier + 2 corresponds to the number of basic 40 ns ticks representing a watchdog tick. The multiplier can be modified in order to adjust the watchdog time over a larger range.

Example "Set SM watchdog"

This checkbox enables manual setting of the watchdog times. If the outputs are set and the EtherCAT communication is interrupted, the SM watchdog is triggered after the set time and the outputs are erased. This setting can be used for adapting a terminal to a slower EtherCAT master or long cycle times. The default SM watchdog setting is 100 ms. The setting range is 0..65535. Together with a multiplier with a range of 1..65535 this covers a watchdog period between 0..~170 seconds.

Calculation

Multiplier = 2498 → watchdog base time = $1 / 25 \text{ MHz} * (2498 + 2) = 0.0001 \text{ seconds} = 100 \mu\text{s}$
SM watchdog = 10000 → $10000 * 100 \mu\text{s} = 1 \text{ second watchdog monitoring time}$

⚠ CAUTION

Undefined state possible!

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

⚠ CAUTION

Damage of devices and undefined state possible!

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

3.4 EtherCAT State Machine

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

A distinction is made between the following states:

- Init
- Pre-Operational
- Safe-Operational and
- Operational
- Boot

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

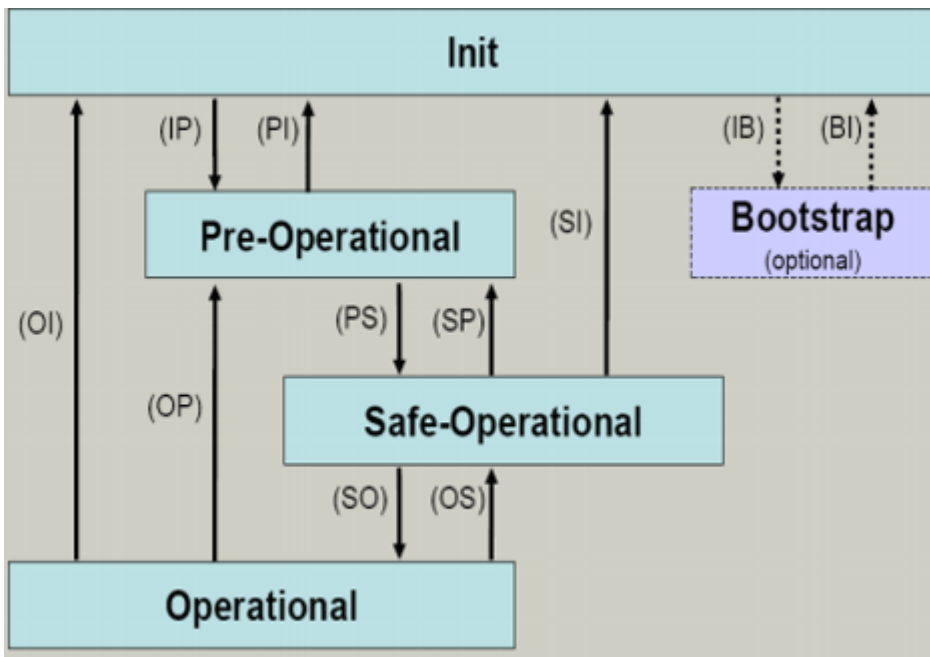


Fig. 13: States of the EtherCAT State Machine

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 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 DP-RAM areas of the EtherCAT slave controller (ECSC).

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` [▶ 19] monitoring sets the outputs of the 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 watchdog monitoring in the module, the outputs can be switched or set also in the `SAFEOP` state.

Operational (Op)

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

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

Boot

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

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

3.5 CoE Interface

General description

The CoE interface (CANopen over EtherCAT) 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 read 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 parameter 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 2 levels via hexadecimal numbering: (main)index, followed by subindex. The value ranges are

- Index: 0x0000 ...0xFFFF (0...65535_{dez})
- SubIndex: 0x00...0xFF (0...255_{dez})

A parameter localized in this way is normally written as 0x8010:07, with preceding "x" 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: In some EtherCAT devices the channel parameters are stored here (as an alternative to the 0x8000 range).
- 0x6000: Input PDOs ("input" from the perspective of the EtherCAT master)
- 0x7000: Output PDOs ("output" from the perspective of the EtherCAT master)

● Availability

I 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:

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 RxDPO-Par Ch.1	RO	> 6 <
1401:0	PwM RxDPO-Par Ch.2	RO	> 6 <
1402:0	PwM RxDPO-Par h.1 Ch.1	RO	> 6 <
1403:0	PwM RxDPO-Par h.1 Ch.2	RO	> 6 <
1600:0	PwM RxDPO-Map Ch.1	RO	> 1 <

Fig. 14: "CoE Online" tab

The figure above shows the CoE objects available in device "EL2502", ranging from 0x1000 to 0x1600. The subindices for 0x1018 are expanded.

Data management and function "NoCoeStorage"

Some parameters, particularly the setting parameters of the slave, are configurable and writeable. This can be done in write or read mode

- via the System Manager (Fig. "CoE Online" tab) by clicking
This is useful for commissioning of the system/slaves. Click on the row of the index to be parameterised and enter a value in the "SetValue" dialog.
- from the control system/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 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.
- 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.

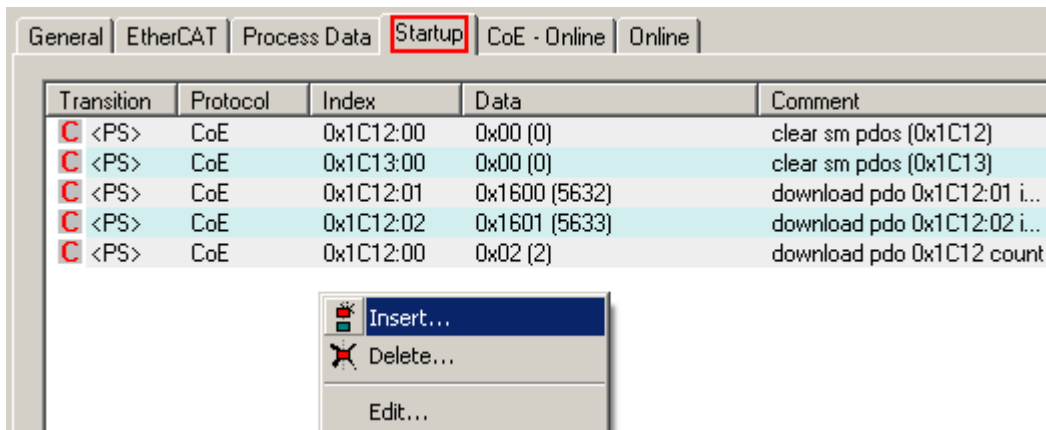


Fig. 15: Startup list in the TwinCAT System Manager

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

Online/offline list

While working with the TwinCAT System Manager, a distinction has to be made whether the EtherCAT device is "available", i.e. switched on and linked via EtherCAT and therefore **online**, or whether a configuration is created **offline** without connected slaves.

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** is shown in red.

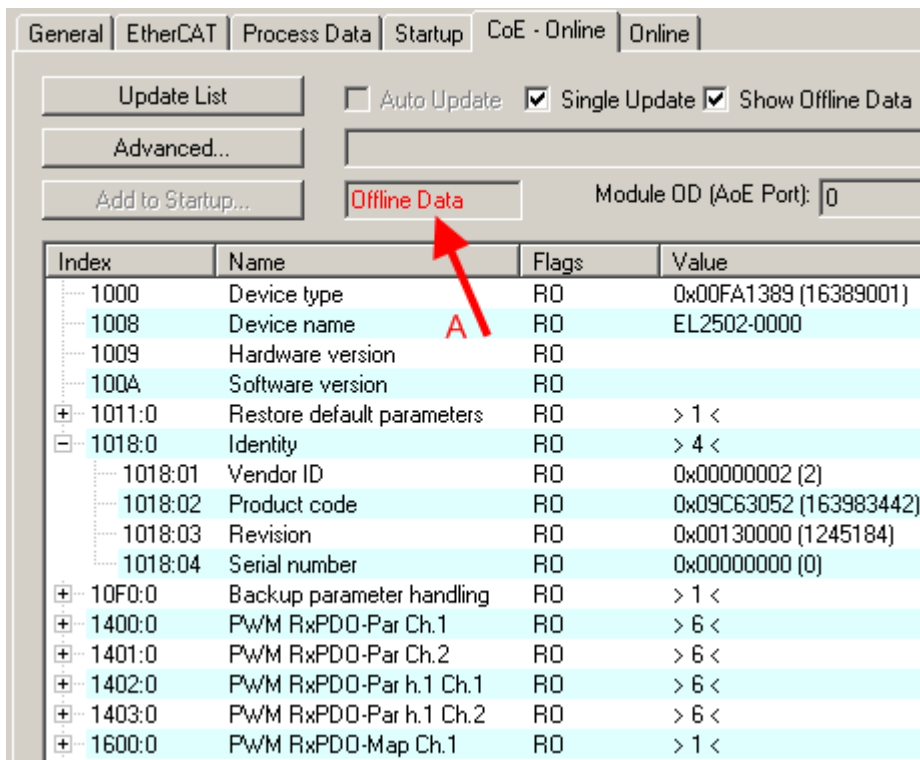


Fig. 16: *Offline list*

- 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 version of the equipment according to the electronic information is displayed
 - **Online** is shown in green.

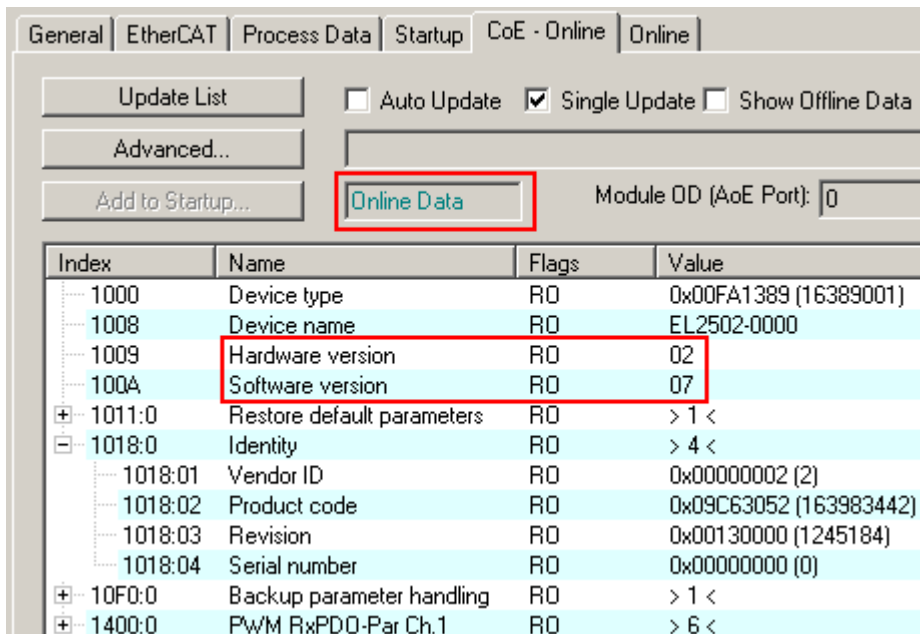


Fig. 17: *Online list*

Channel-based order

The CoE list is available in EtherCAT devices that usually feature several functionally equivalent channels. For example, a 4-channel analog 0..10 V input terminal also has 4 logical channels and therefore 4 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 $16_{\text{dec}}/10_{\text{hex}}$ 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.

3.6 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](#).

4 Mounting and wiring

4.1 Instructions for ESD protection

NOTE

Destruction of the devices by electrostatic discharge possible!

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

- Please ensure you are electrostatically discharged and avoid touching the contacts of the device directly.
- Avoid contact with highly insulating materials (synthetic fibers, plastic film etc.).
- Surroundings (working place, packaging and personnel) should be grounded probably, when handling with the devices.
- Each assembly must be terminated at the right hand end with an [EL9011](#) or [EL9012](#) bus end cap, to ensure the protection class and ESD protection.

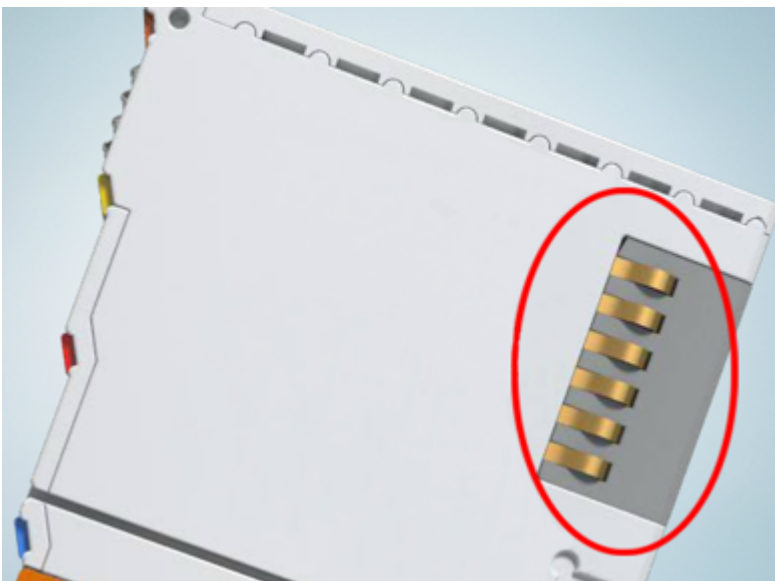


Fig. 18: Spring contacts of the Beckhoff I/O components

4.2 Notes on TwinSAFE SC

4.2.1 Safety instructions

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

4.2.2 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 environments
- in an environment that leads to unacceptable soiling of the TwinSAFE component

NOTE**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.

4.2.3 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).

4.2.4 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.

4.3 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!

Assembly

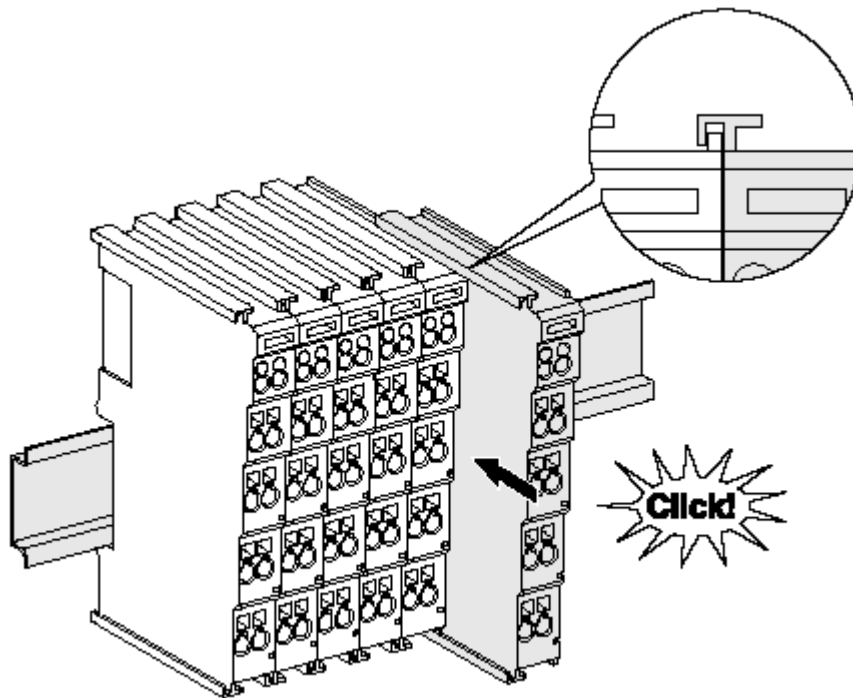


Fig. 19: Attaching on mounting rail

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.

i Fixing of mounting rails

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

Disassembly

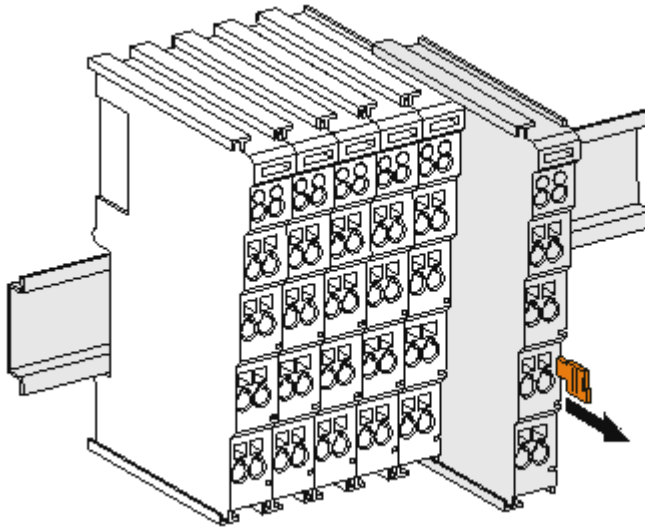


Fig. 20: Disassembling of terminal

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

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

Connections within a bus terminal block

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

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

i Power Contacts

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

PE power contact

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

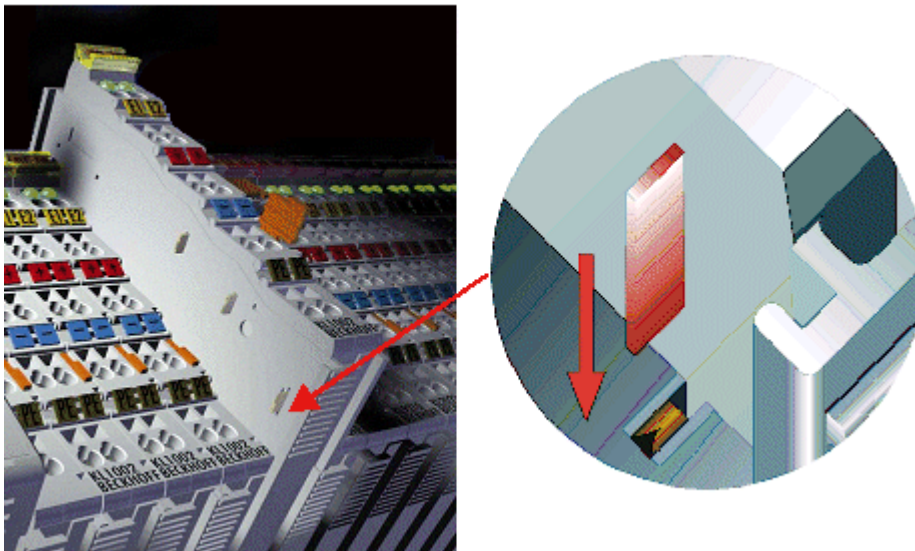


Fig. 21: Power contact on left side

NOTE

Possible damage of the device

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

⚠ WARNING

Risk of electric shock!

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

4.4 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

For terminals with enhanced mechanical load capacity, the following additional installation instructions 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 with) or 32 terminals (24 mm mounting with)
- 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.

4.5 Connection

4.5.1 Connection system

⚠ WARNING

Risk of electric shock and damage of device!

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

Overview

The Bus Terminal system offers different connection options for optimum adaptation to the respective application:

- The terminals of ELxxxx and KLxxxx series with standard wiring include electronics and connection level in a single enclosure.

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

Standard wiring (ELxxxx / KLxxxx)



Fig. 22: Standard wiring

The terminals of ELxxxx and KLxxxx series have been tried and tested for years. They feature integrated screwless spring force technology for fast and simple assembly.

Pluggable wiring (ESxxxx / KSxxxx)



Fig. 23: 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. 24: *High Density Terminals*

The Bus Terminals from these series with 16 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**

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

Ultrasonically "bonded" (ultrasonically welded) conductors

● **Ultrasonically "bonded" conductors**

i It is also possible to connect the Standard and High Density Terminals with ultrasonically "bonded" (ultrasonically welded) conductors. In this case, please note the tables concerning the wire-size width below!

4.5.2 Wiring

⚠ WARNING

Risk of electric shock and damage of device!

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

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

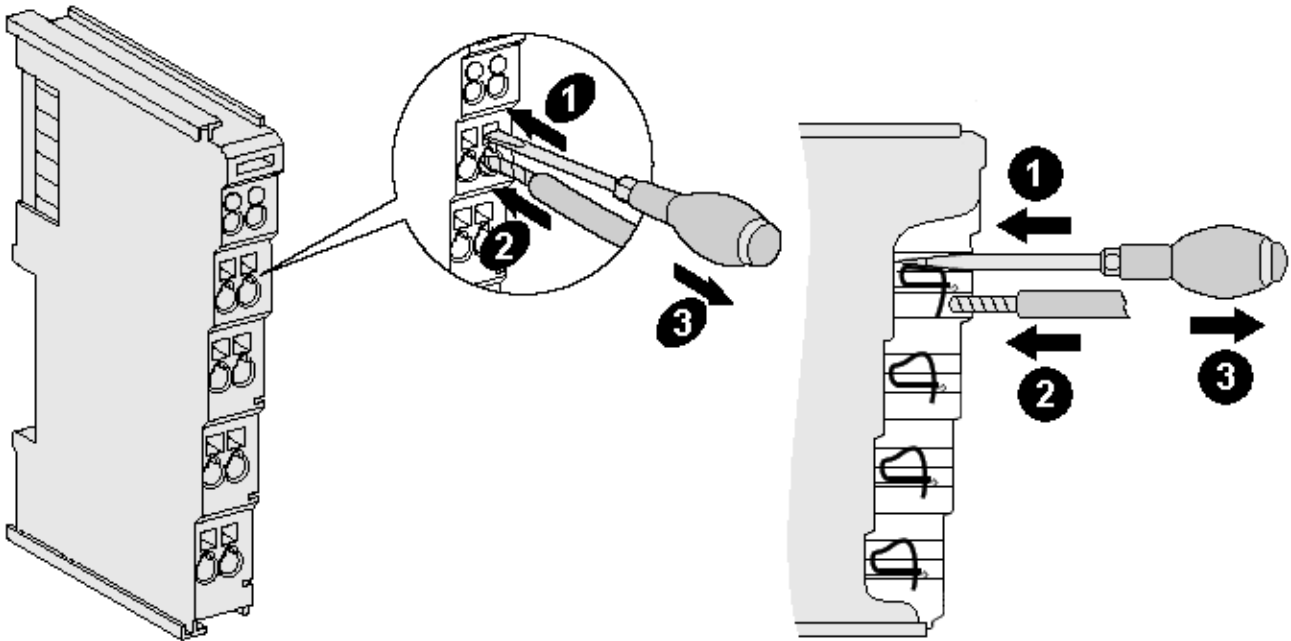


Fig. 25: Connecting a cable on a terminal point

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:

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. The terminal point closes automatically when the pressure is released, holding the wire securely and permanently.

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 [▶ 36]) with 16 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 "bonded" conductors)	only 1.5 mm ²
Wire stripping length	8 ... 9 mm

4.5.3 Shielding



Shielding

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

4.6 Positioning of passive Terminals

i **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 2 passive terminals!

Examples for positioning of passive terminals (highlighted)

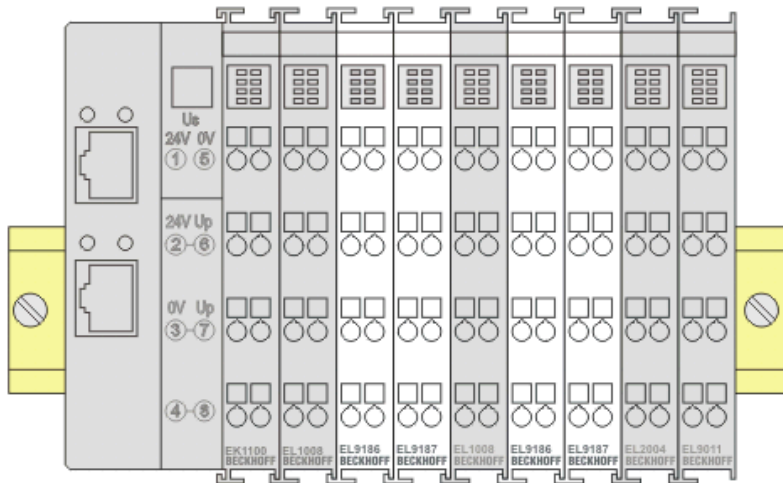


Fig. 26: Correct positioning

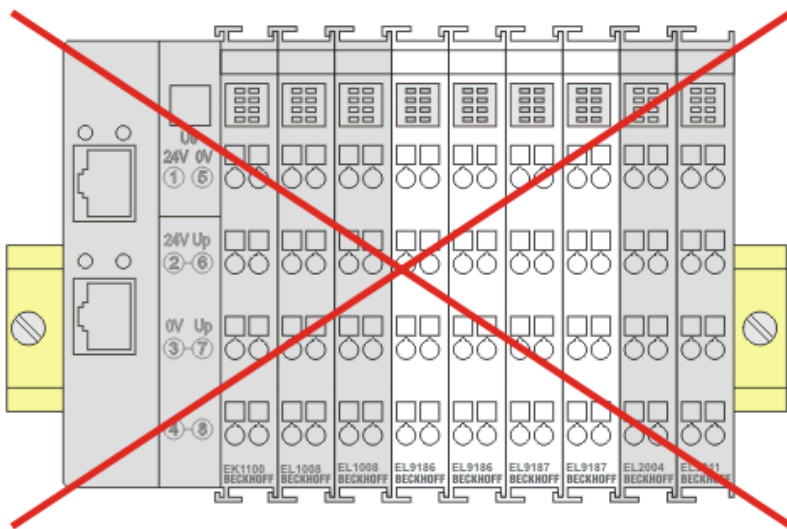


Fig. 27: Incorrect positioning

4.7 Installation positions

NOTE

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.

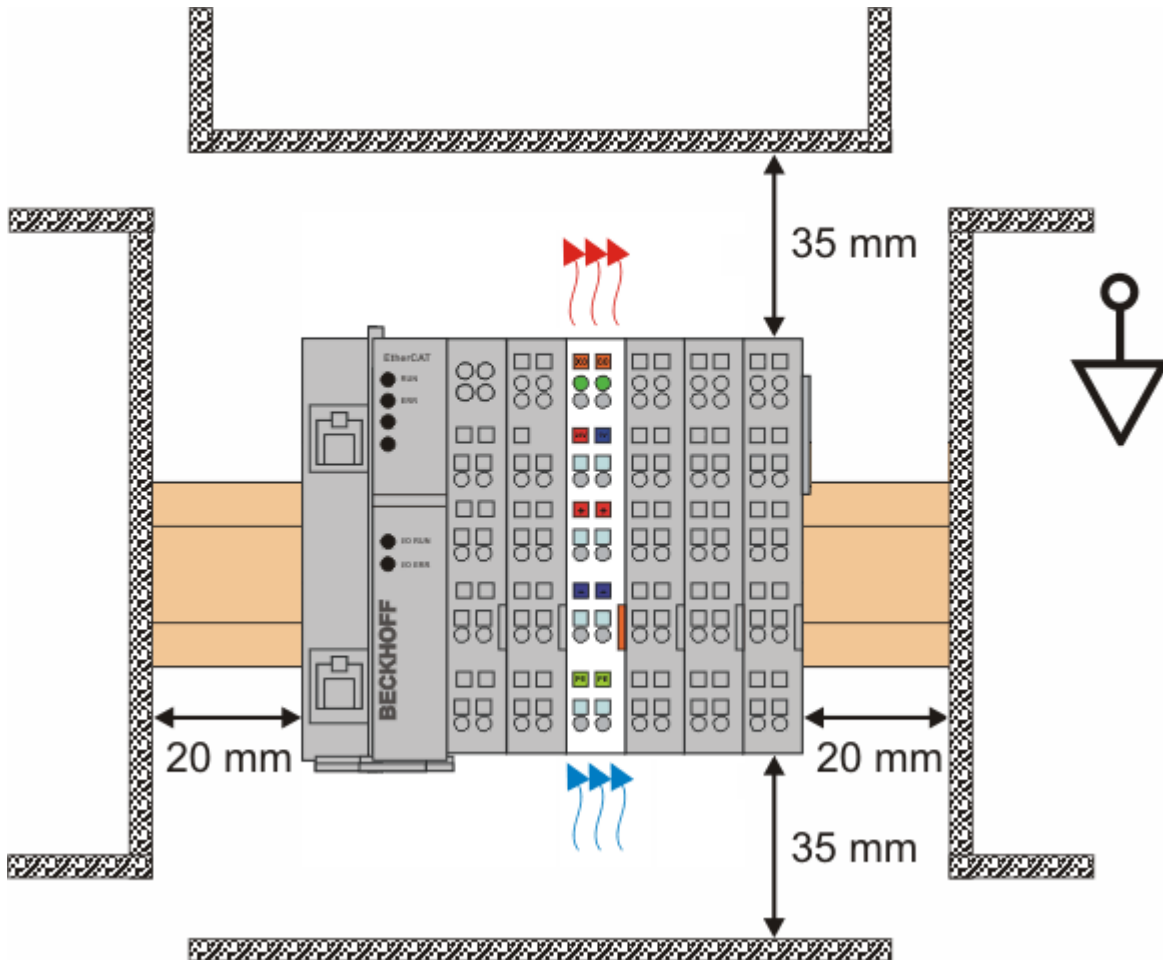


Fig. 28: Recommended distances for standard installation position

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.

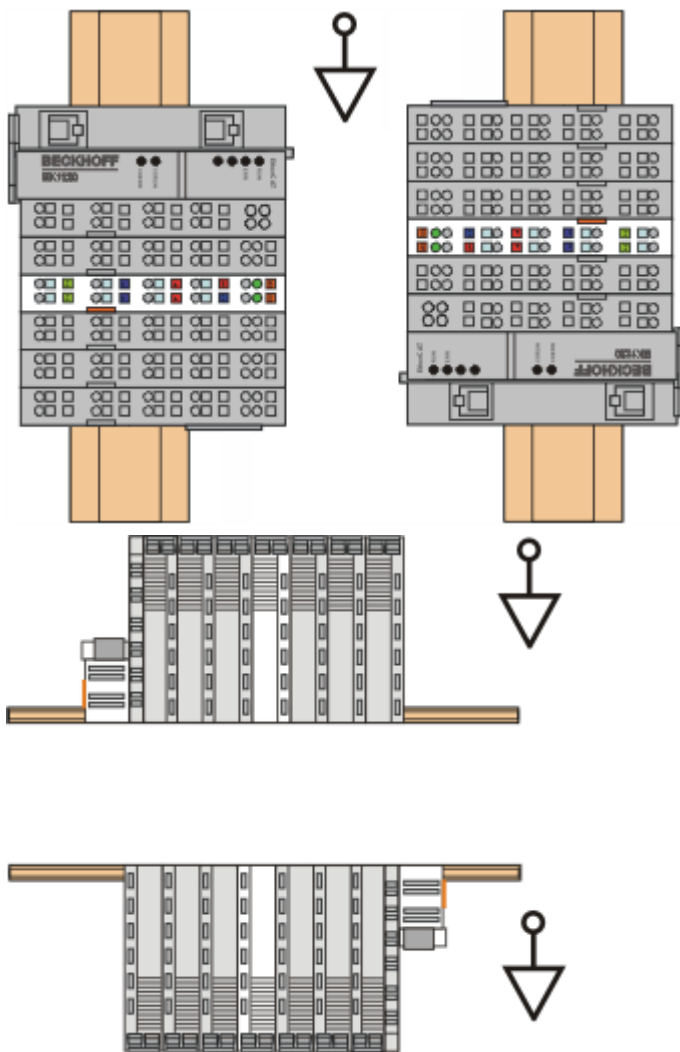


Fig. 29: Other installation positions

4.8 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 (directive 94/9/EU)!

- The certified components are to be installed in a suitable housing that guarantees a protection class of at least IP54 in accordance with EN 60529! 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 60079-0:2012+A11:2013
- EN 60079-15:2010

Marking

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



II 3G KEMA 10ATEX0075 X Ex nA IIC T4 Gc Ta: -25 ... 60°C

or



II 3G KEMA 10ATEX0075 X Ex nC IIC T4 Gc Ta: -25 ... 60°C

4.9 ATEX Documentation



Notes about operation of the Beckhoff terminal systems in potentially explosive areas (ATEX)

Pay also attention to the continuative documentation

Notes about operation of the Beckhoff terminal systems in potentially explosive areas (ATEX)

that is available in the download area of the Beckhoff homepage <http://www.beckhoff.com!>

4.10 EL6224 - LEDs and connection (HD housing)

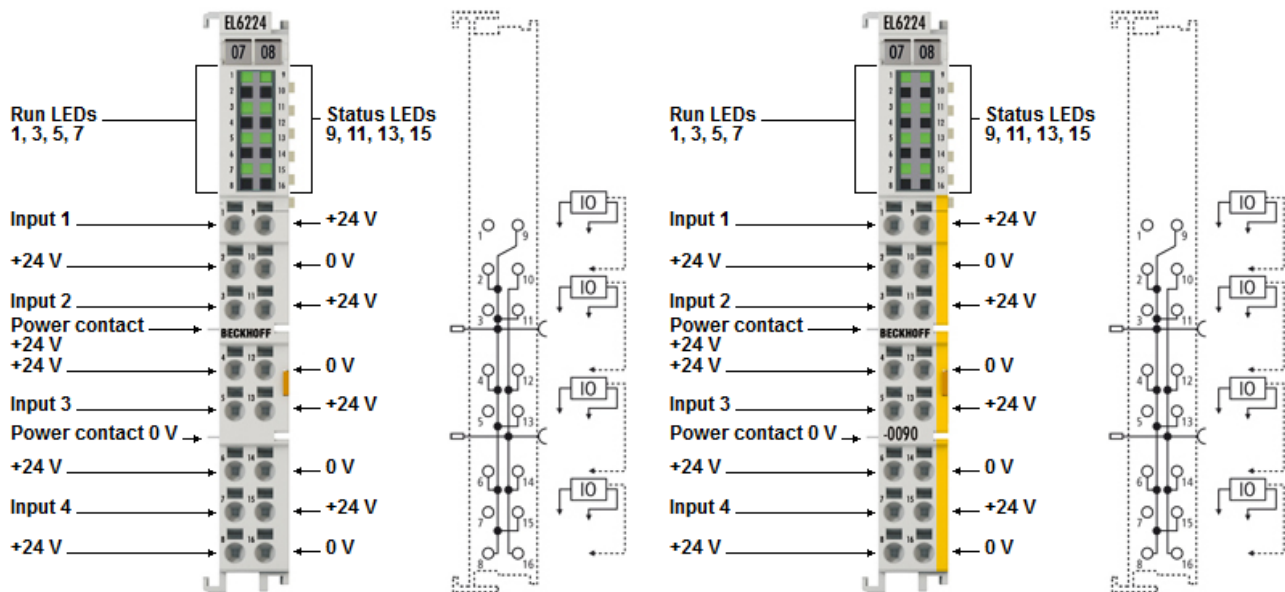


Fig. 30: EL6224_EL6224-0090 - LEDs and connection

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

LEDs			
LED	Color	Meaning	
RUN	green	These LEDs indicate the terminal's operating state:	
		off	State of the EtherCAT State Machine [►_21]: INIT = initialization of the terminal or BOOTSTRAP = function for <u>firmware updates</u> [►_149] 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 [►_94] 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 Commissioning

5.1 TwinCAT Quick Start

TwinCAT is a development environment for real-time control including 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 Systemmanual:**
Fieldbus Components → EtherCAT Terminals → EtherCAT System Documentation → Setup in the TwinCAT System Manager
- **TwinCAT 2** → TwinCAT System Manager → I/O - Configuration
- In particular, TwinCAT driver installation:
Fieldbus components → Fieldbus Cards and Switches → FC900x – PCI Cards for Ethernet → Installation

Devices contain the 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 offline mode can be found under <http://infosys.beckhoff.com>:
TwinCAT 2 → TwinCAT System Manager → IO - Configuration → Adding 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 from user PC to the individual control elements:

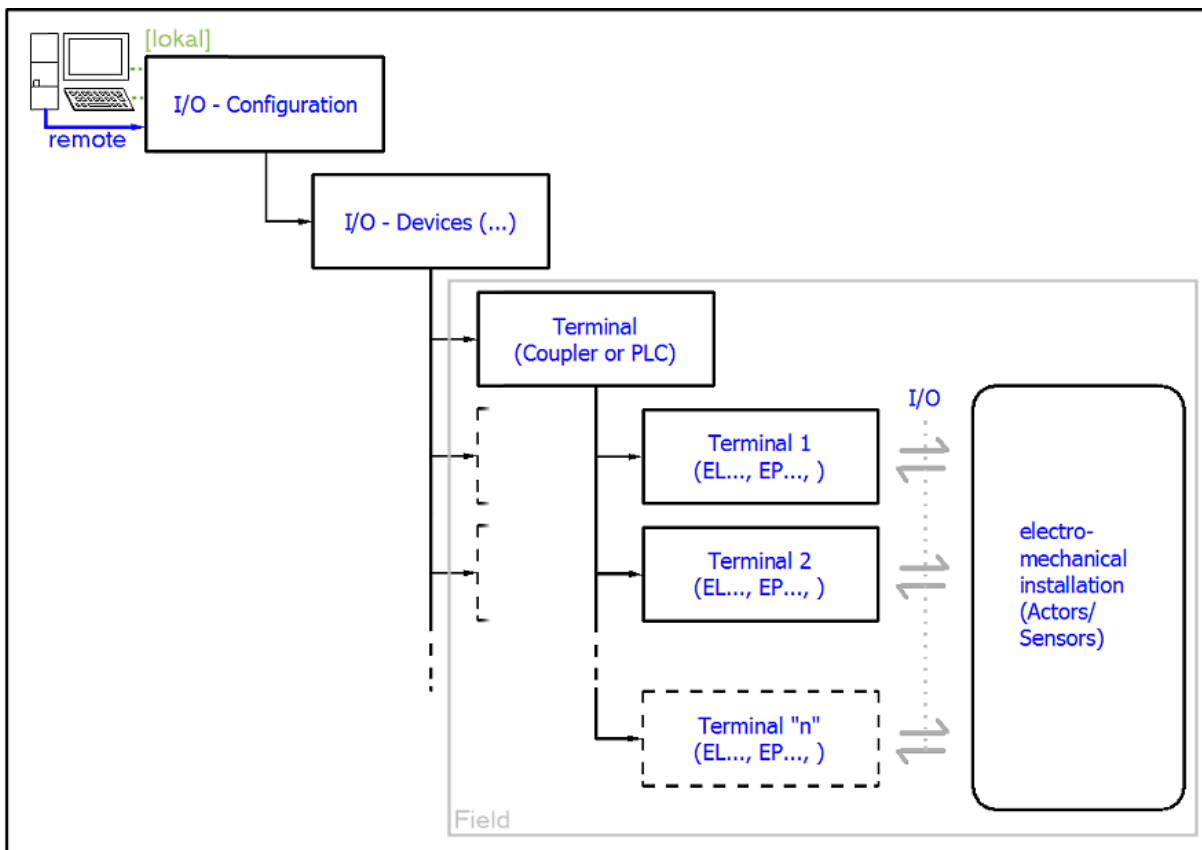


Fig. 31: Relationship between user side (commissioning) and installation

The user inserting of certain components (I/O device, terminal, box...) is the same in TwinCAT 2 and TwinCAT 3. The descriptions below relate to the online procedure.

Sample configuration (actual configuration)

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

- Control system (PLC) **CX2040** including **CX2100-0004** power supply unit
- Connected to the CX2040 on the right (E-bus):
EL1004 (4-channel analog input terminal -10...+10 V)
- 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)

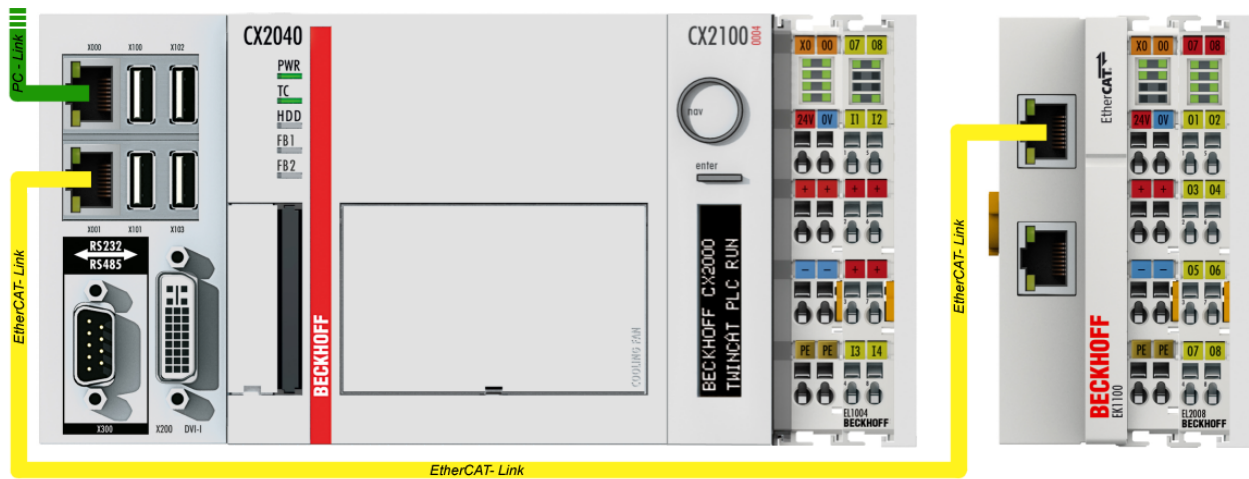


Fig. 32: Control configuration with Embedded PC, input (EL1004) and output (EL2008)

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

5.1.1 TwinCAT 2

Startup

TwinCAT 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:

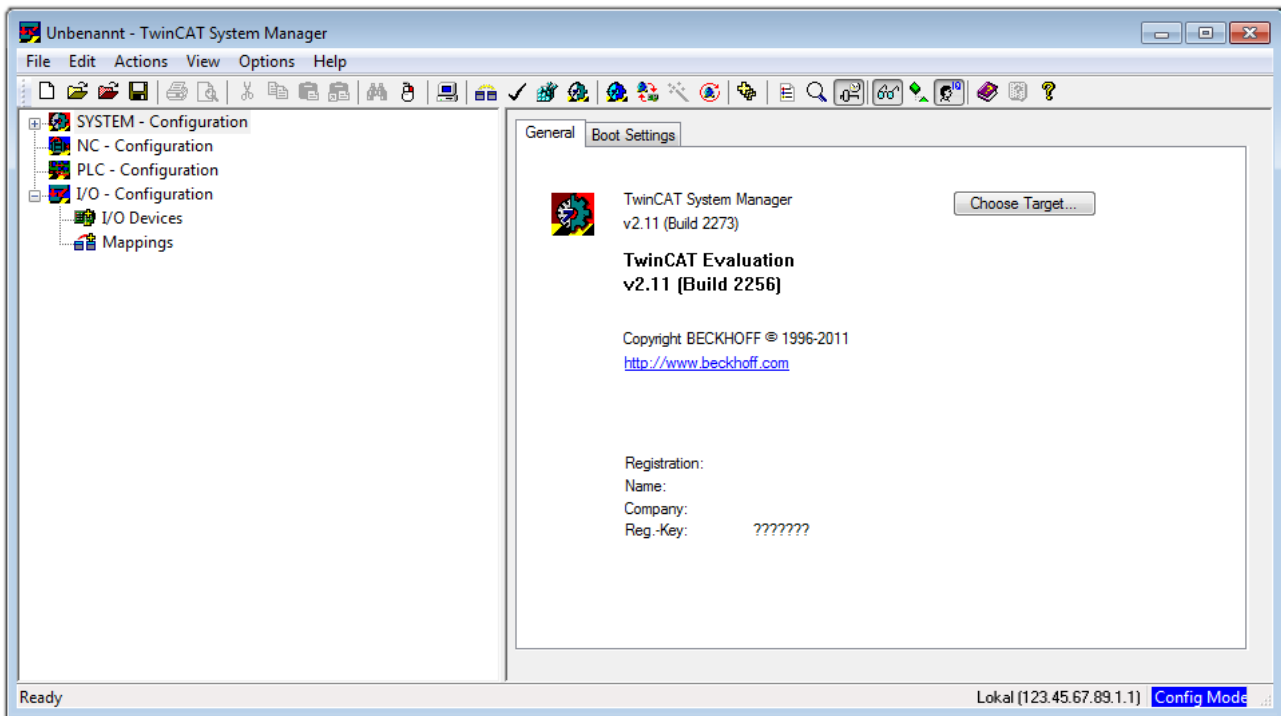



Fig. 33: Initial TwinCAT 2 user interface

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 thereby the next step is "Insert Device [▶ 50]".

If the intention is to address the TwinCAT runtime environment installed on a PLC as development environment remotely from another system, the target system must be made known first. In the menu under

"Actions" → "Choose Target System...", via the symbol "  " or the "F8" key, open the following window:

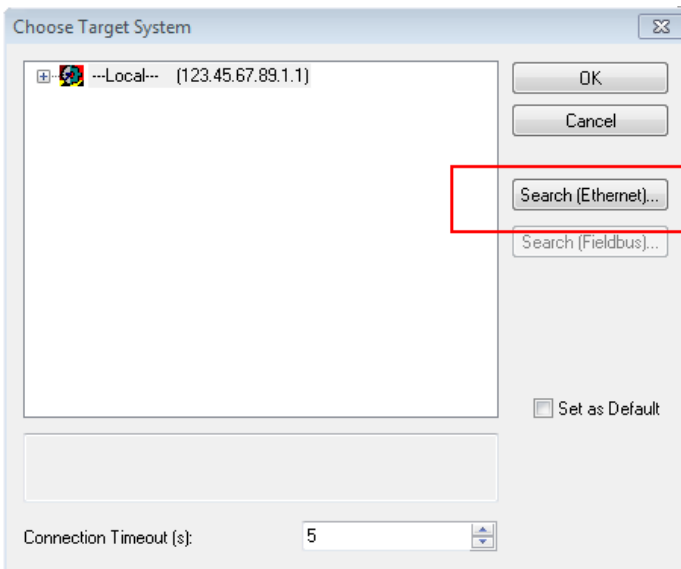


Fig. 34: Selection of the target system

Use "Search (Ethernet)..." to enter the target system. Thus a next 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.

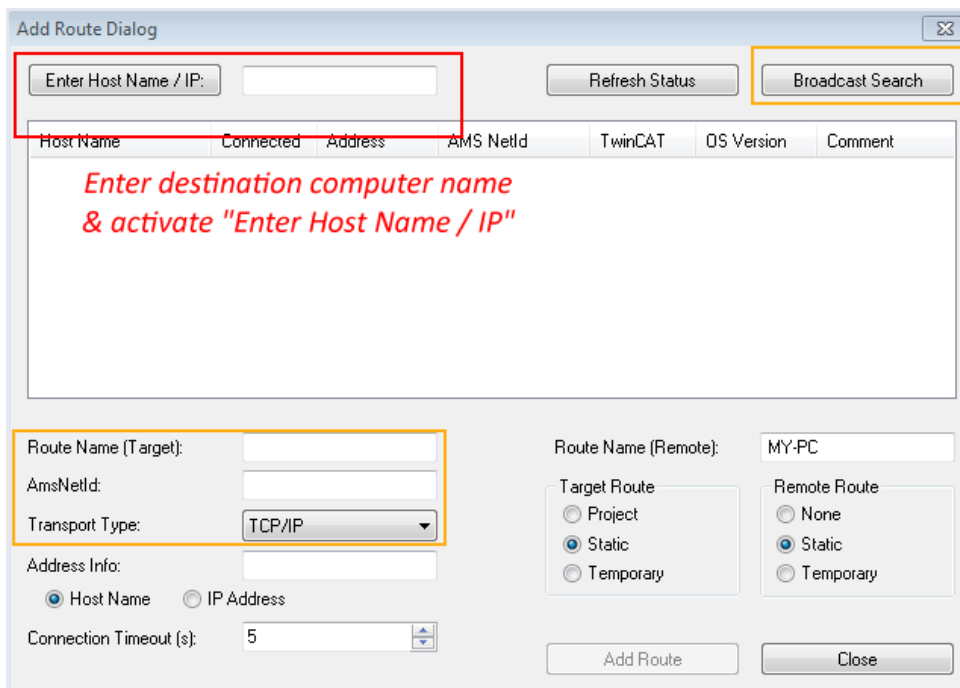
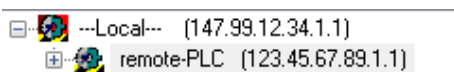


Fig. 35: Specify the PLC for access by the TwinCAT System Manager: selection of the target system



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



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 menu "Actions" → "Set/Reset TwinCAT to Config Mode..." (Shift + F4).

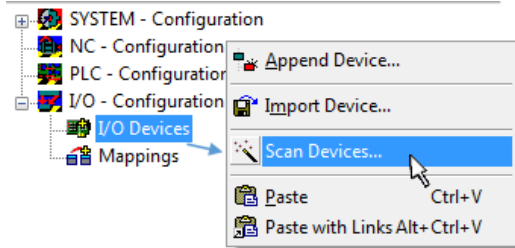


Fig. 36: Select "Scan Devices..."

Confirm the warning message, which follows, and select "EtherCAT" in the dialog:

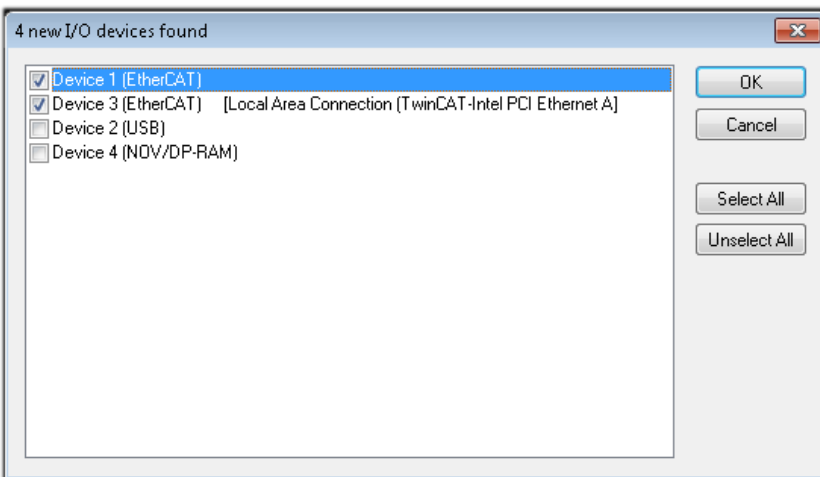


Fig. 37: Automatic detection of I/O devices: selection the devices to be integrated

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 [sample configuration](#) [▶ 46] described at the beginning of this section, the result is as follows:

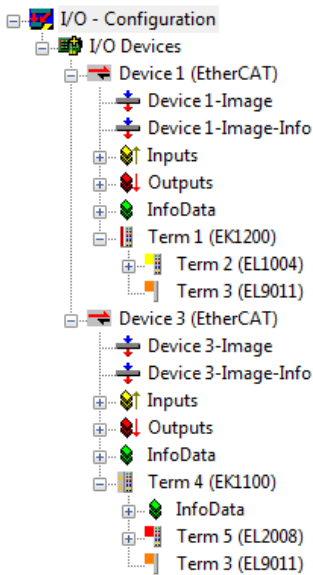


Fig. 38: Mapping of the configuration in the TwinCAT 2 System Manager

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

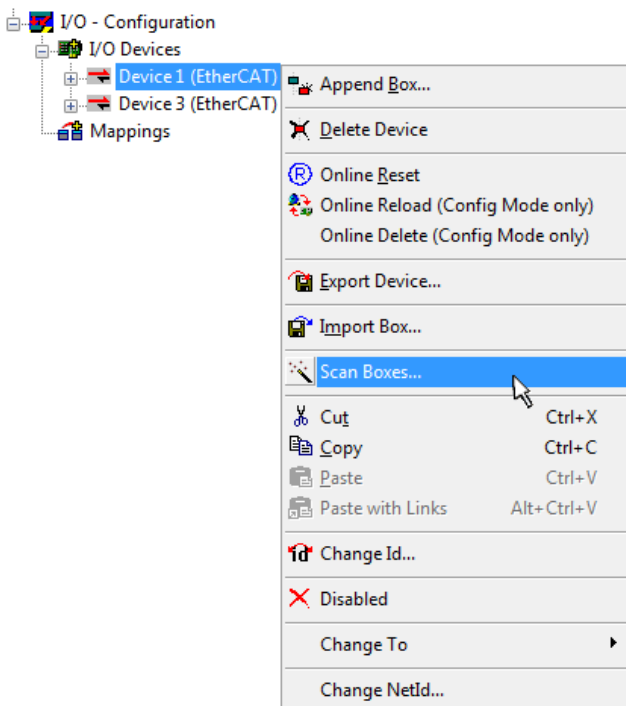


Fig. 39: Reading of individual terminals connected to a device

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 the creation of 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 to Structured Text (ST).

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

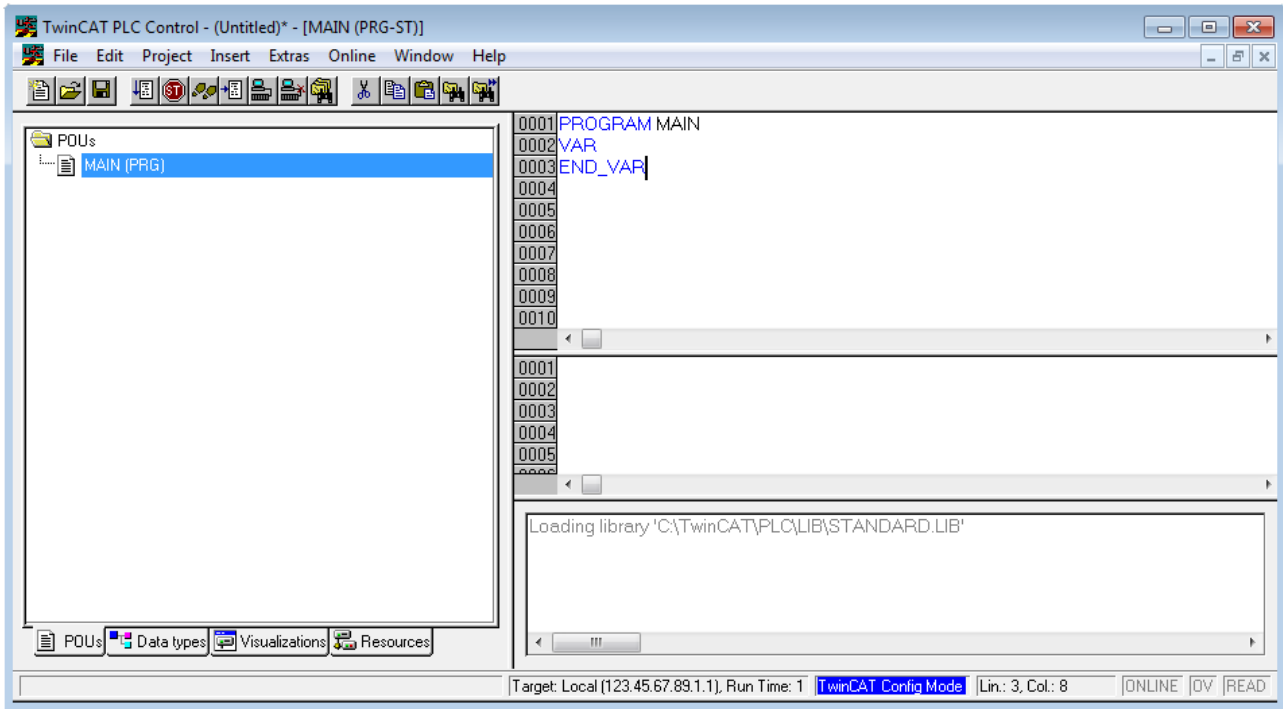


Fig. 40: TwinCAT PLC Control after startup

Sample variables and a sample program have been created and stored under the name "PLC_example.pro":

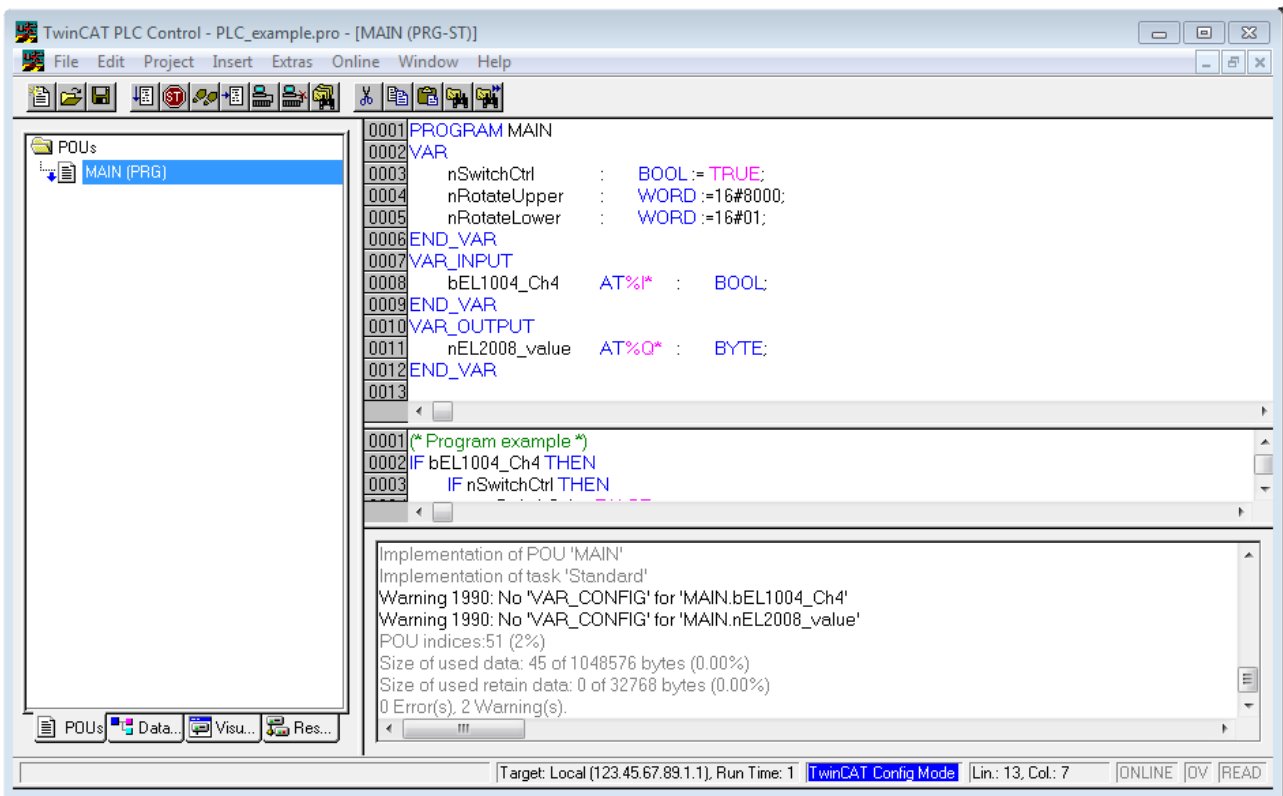


Fig. 41: Sample program with variables after a compile process (without variable integration)

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** via the context menu of the PLC configuration; right-click and select "Append PLC Project...":

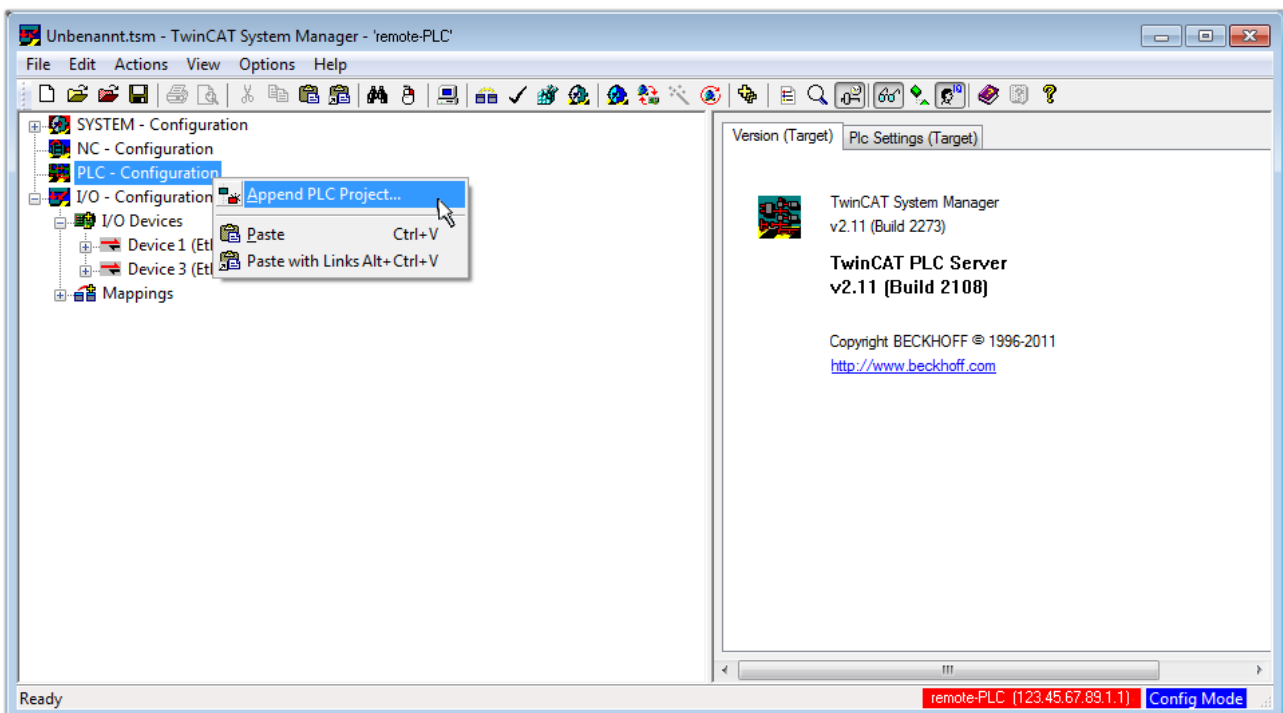


Fig. 42: Appending the TwinCAT PLC Control project

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:

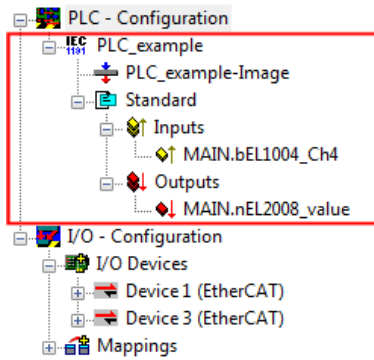


Fig. 43: PLC project integrated in the PLC configuration of the System Manager

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":

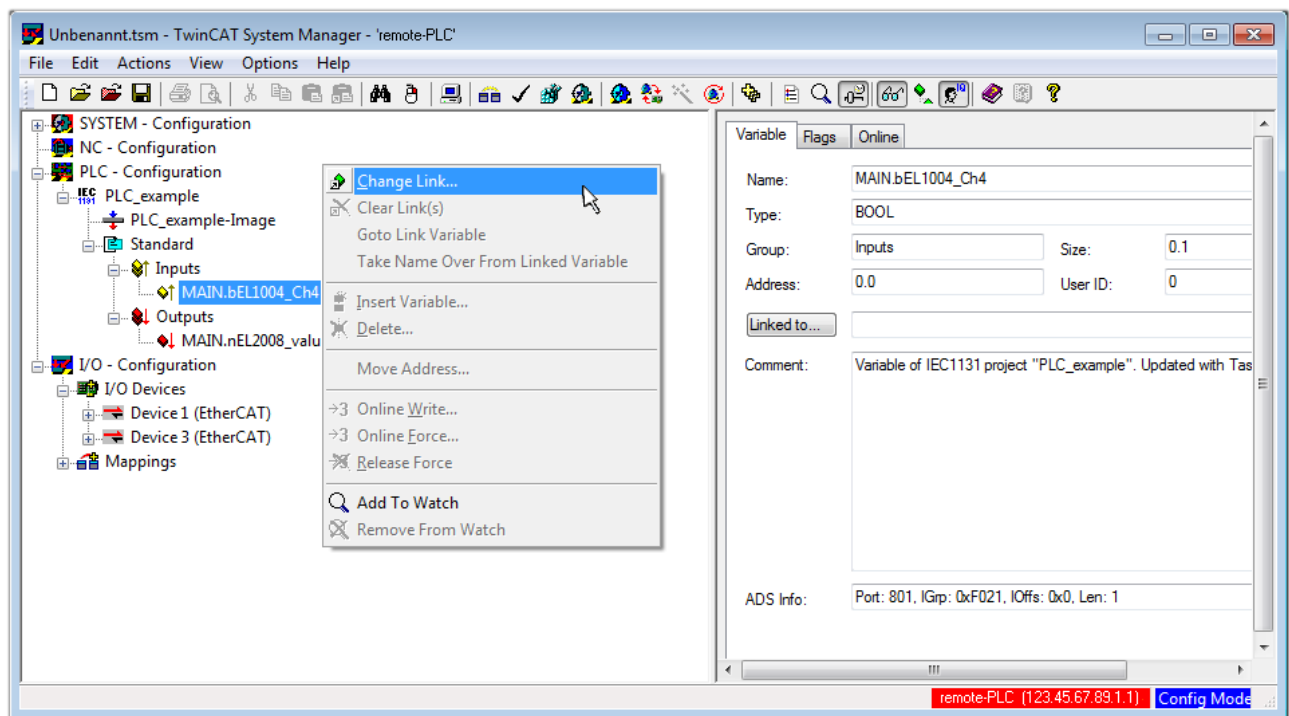


Fig. 44: Creating the links between PLC variables and process objects

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

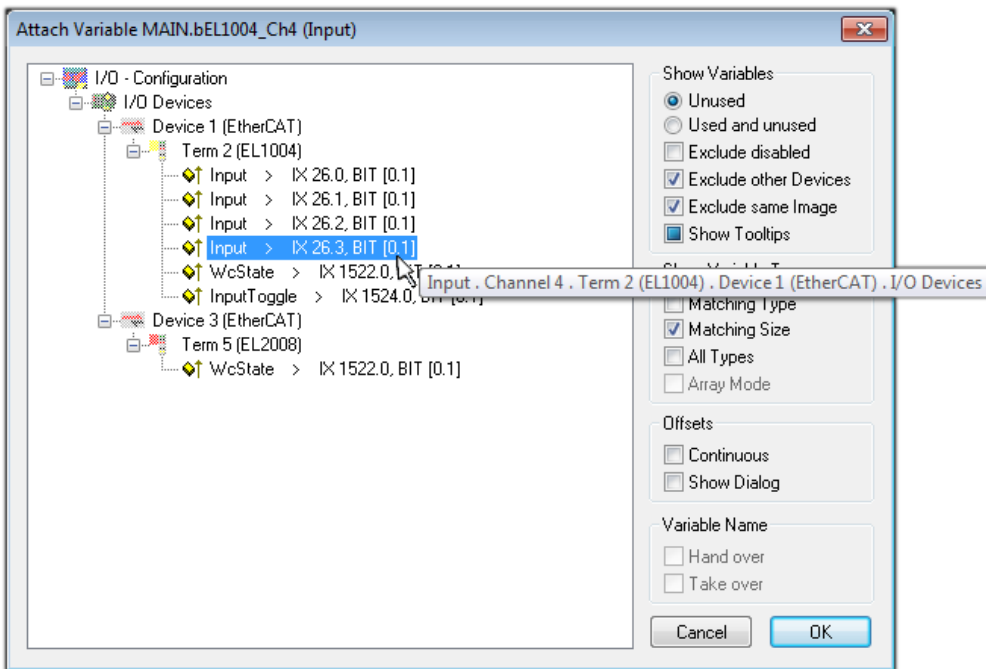


Fig. 45: Selecting PDO of type BOOL

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

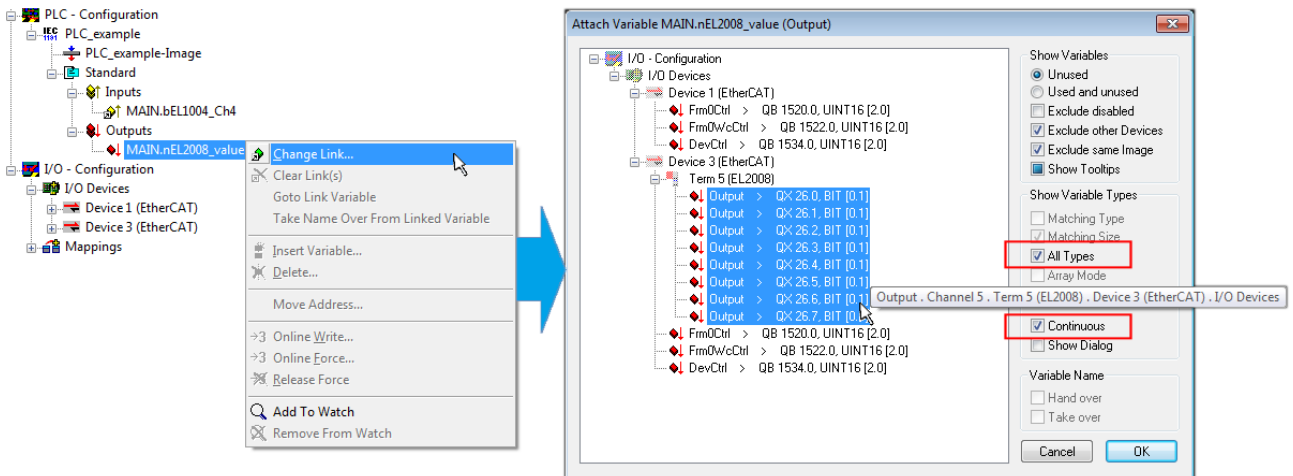



Fig. 46: Selecting several PDOs simultaneously: activate "Continuous" and "All types"

Note that the "Continuous" checkbox was also activated. This is designed to allocate the bits contained in the byte of the variable "nEL2008_value" sequentially to all eight selected output bits of the EL2008 terminal. In this way it is 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 () at the yellow or red object of the variable indicates that a link exists. The links can also be checked by selecting a "Goto Link Variable" from the context menu of a variable. The object opposite, in this case the PDO, is automatically selected:

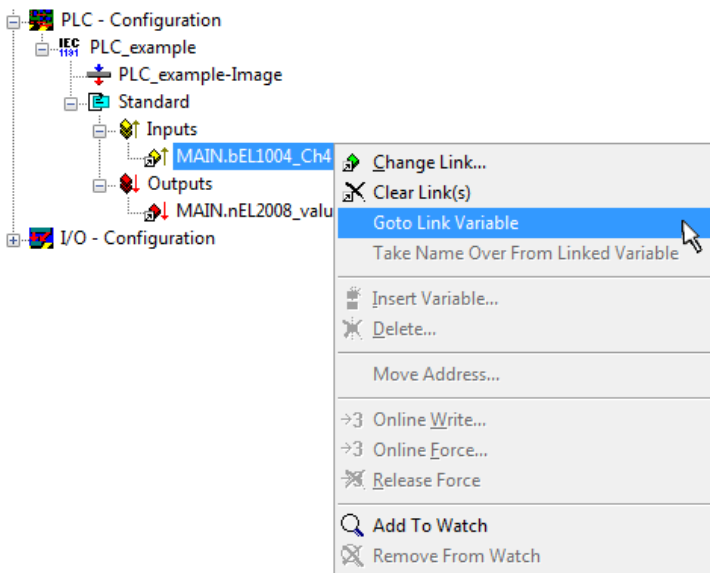

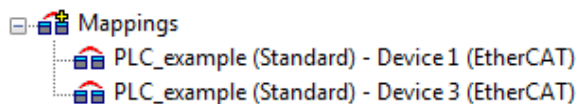


Fig. 47: Application of a "Goto Link" variable, using "MAIN.bEL1004_Ch4" as a sample

The process of assigning variables to the PDO is completed via the menu selection "Actions" → "Generate

Mappings", key Ctrl+M or by clicking on the symbol  in the menu.


This can be visualized in the configuration:




The process of creating links can also take place in the opposite direction, i.e. starting with individual PDOs to variable. However, in this example it would then 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 possible to allocate this a set of bit-standardised variables (type "BOOL"). Here, too, a "Goto Link Variable" from the context menu of a PDO 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 are 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 Ethernet via "Online" → "Choose Run-Time System...":

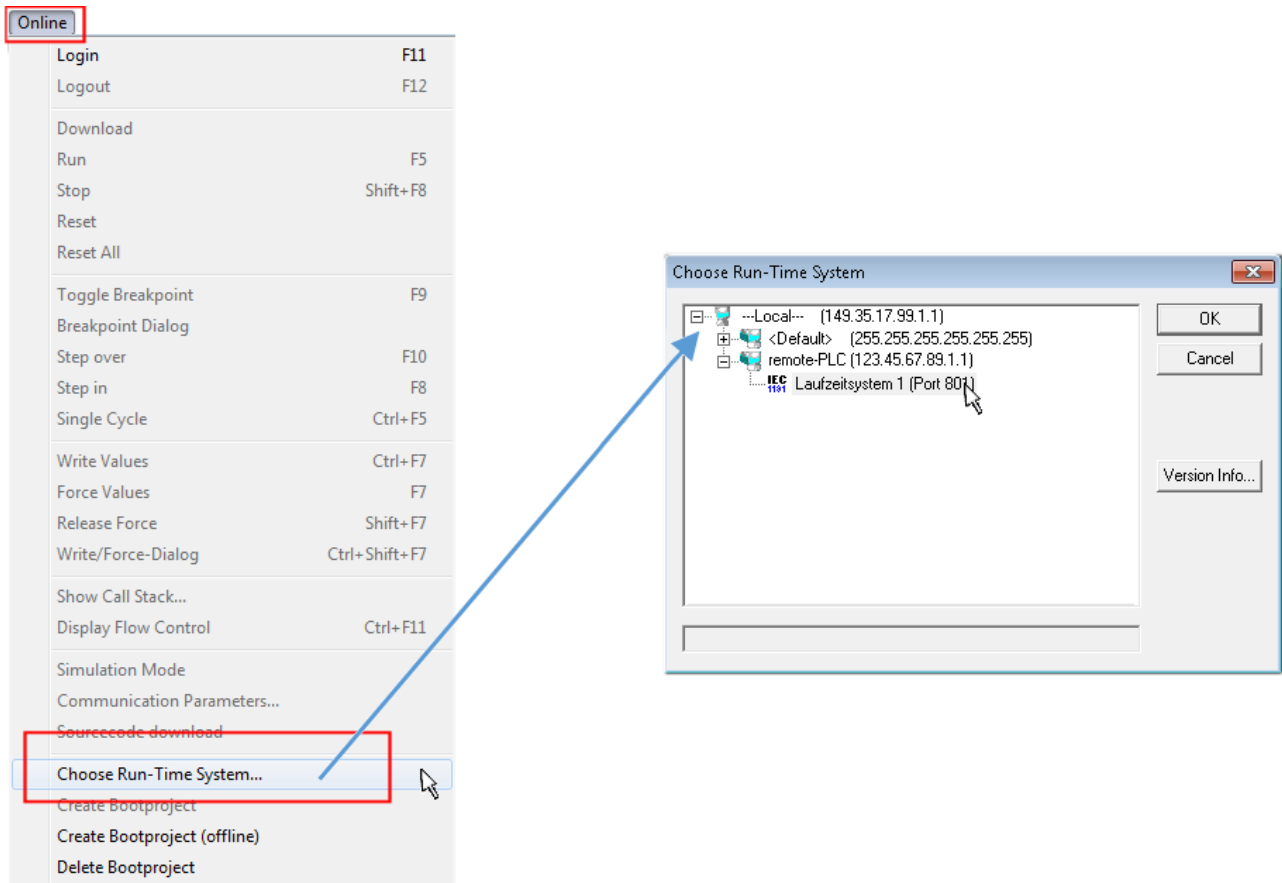

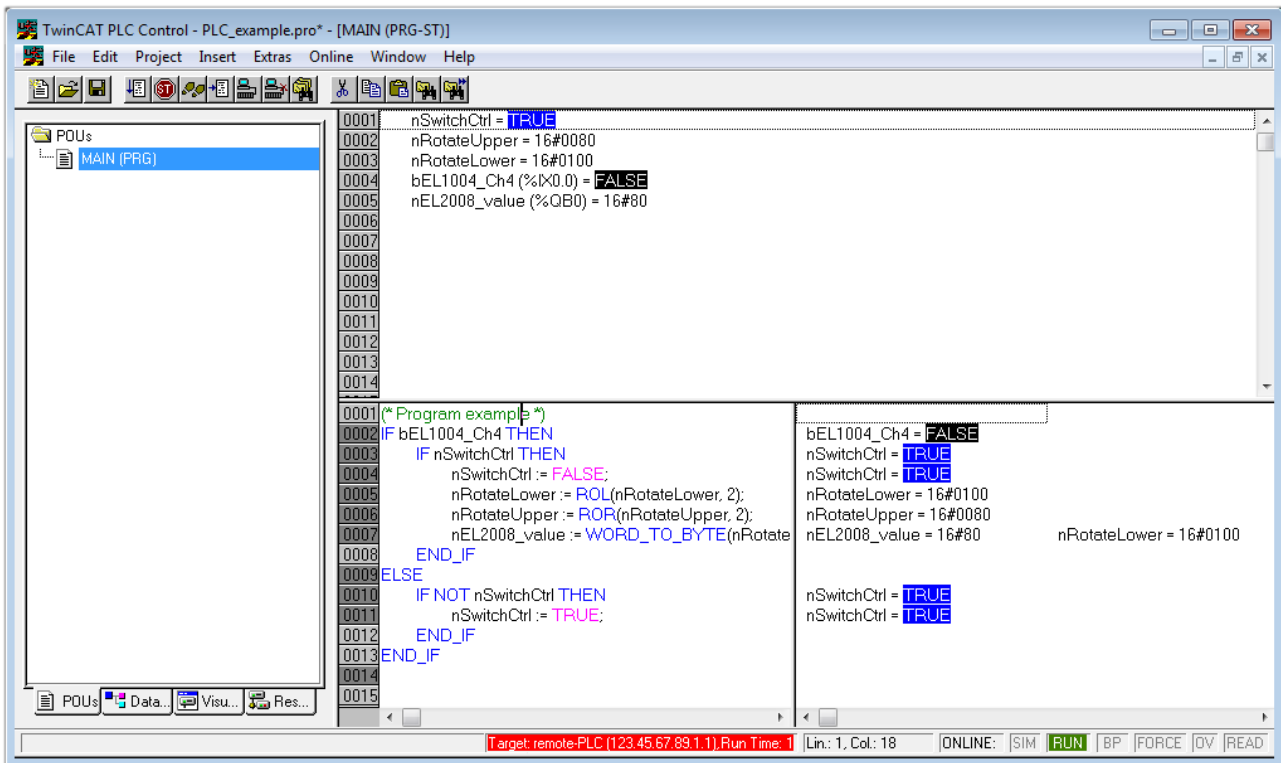


Fig. 48: Choose target system (remote)

In this sample "Runtime system 1 (port 801)" is selected and confirmed. Link the PLC with the real-time

system via 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 acknowledged with "Yes". The runtime environment is ready for the program start:



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

5.1.2 TwinCAT 3


Startup

TwinCAT makes the development environment areas available together with Microsoft Visual Studio: after startup, the project folder explorer appears on the left in the general window area (cf. "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. 50: 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):

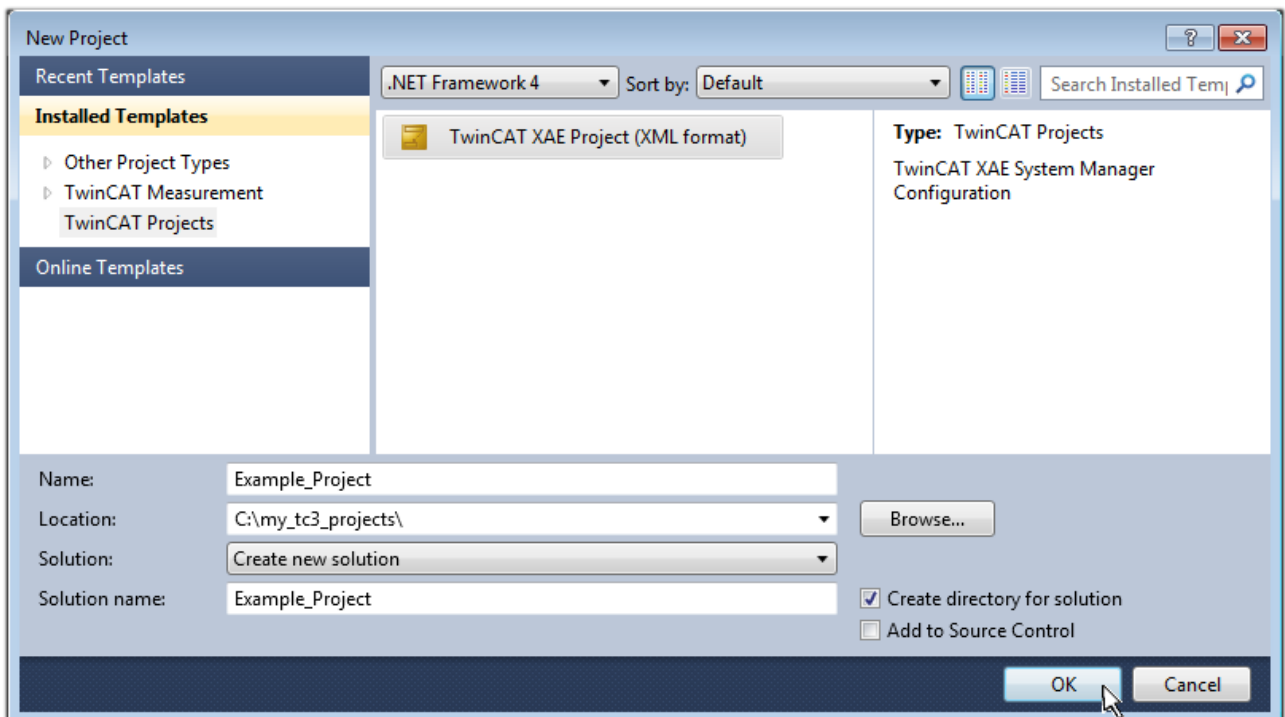


Fig. 51: Create new TwinCAT project

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

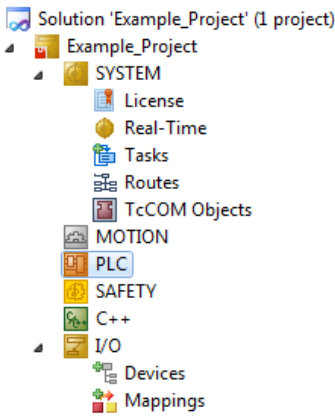
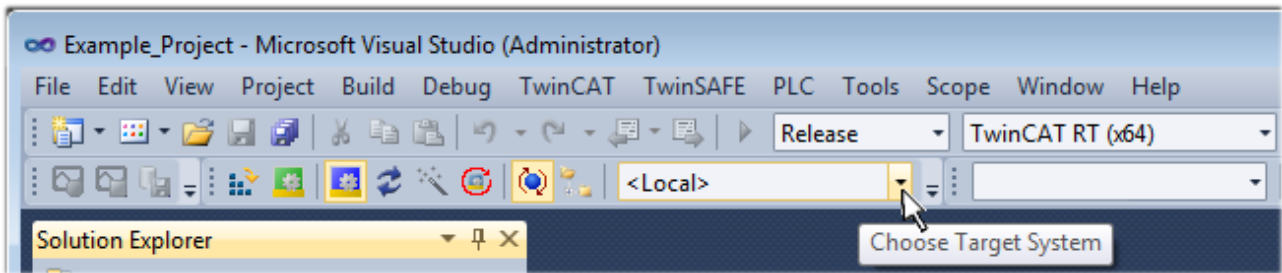


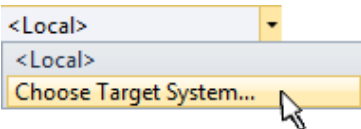
Fig. 52: New TwinCAT3 project in the project folder explorer

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, TwinCAT can be used in local mode and thereby the next step is "Insert Device [▶ 61]".

If the intention is to address the TwinCAT runtime environment installed on a PLC as development environment remotely from another system, 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:

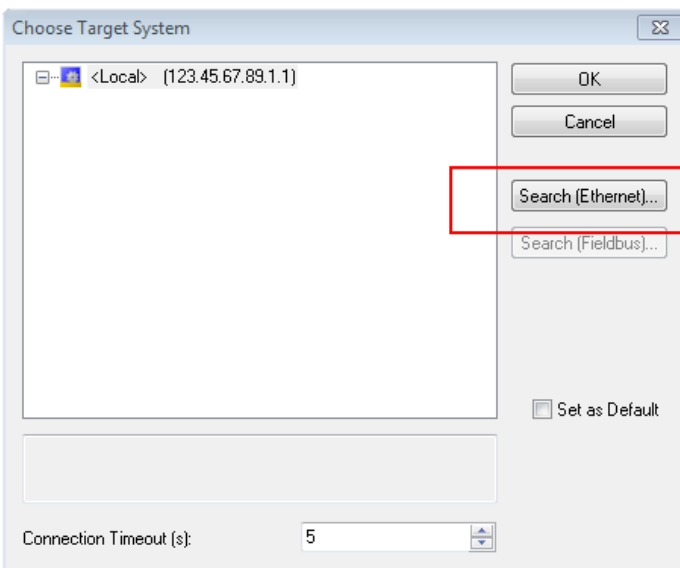


Fig. 53: Selection dialog: Choose the target system

Use "Search (Ethernet)..." to enter the target system. Thus a next 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.

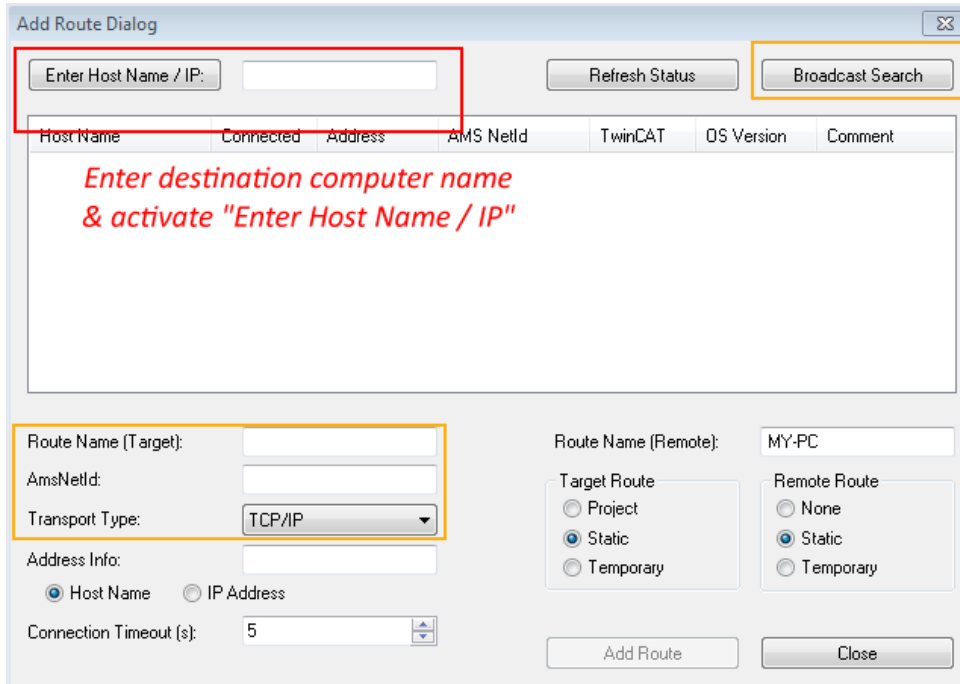
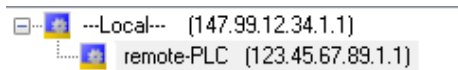


Fig. 54: Specify the PLC for access by the TwinCAT System Manager: selection of the target system


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




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

Adding devices

In the project folder explorer of the Visual Studio shell user interface on the left, select "Devices" within

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)".

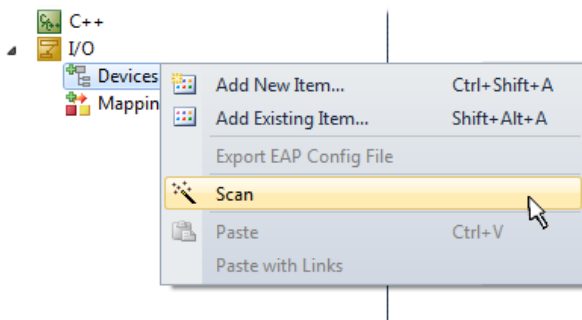


Fig. 55: Select "Scan"

Confirm the warning message, which follows, and select "EtherCAT" in the dialog:

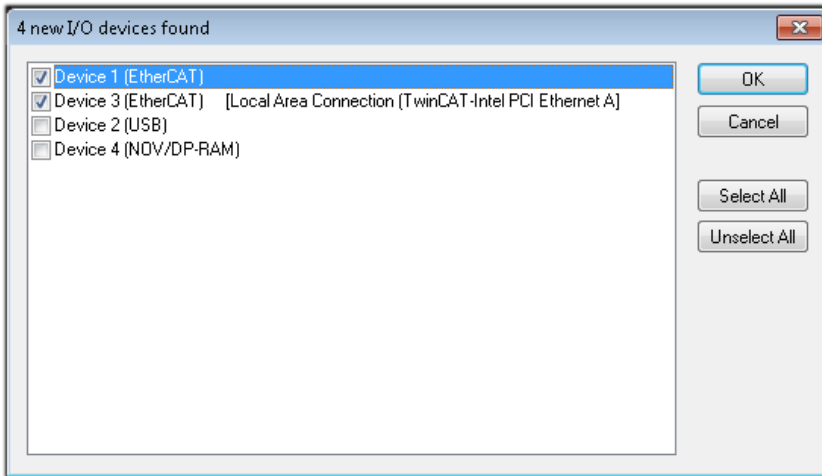


Fig. 56: Automatic detection of I/O devices: selection the devices to be integrated

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 [sample configuration \[▶ 46\]](#) described at the beginning of this section, the result is as follows:

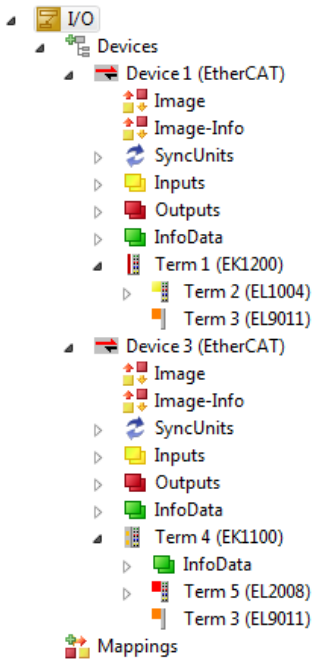


Fig. 57: Mapping of the configuration in VS shell of the TwinCAT3 environment

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

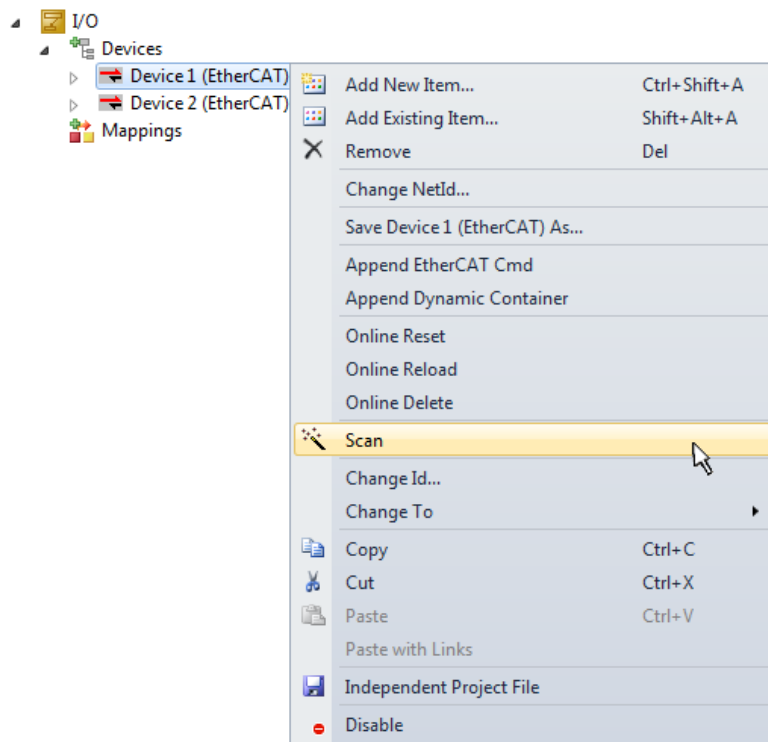


Fig. 58: Reading of individual terminals connected to a device

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

Programming the PLC

TwinCAT PLC Control is the development environment for the creation of 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 to Structured Text (ST).

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

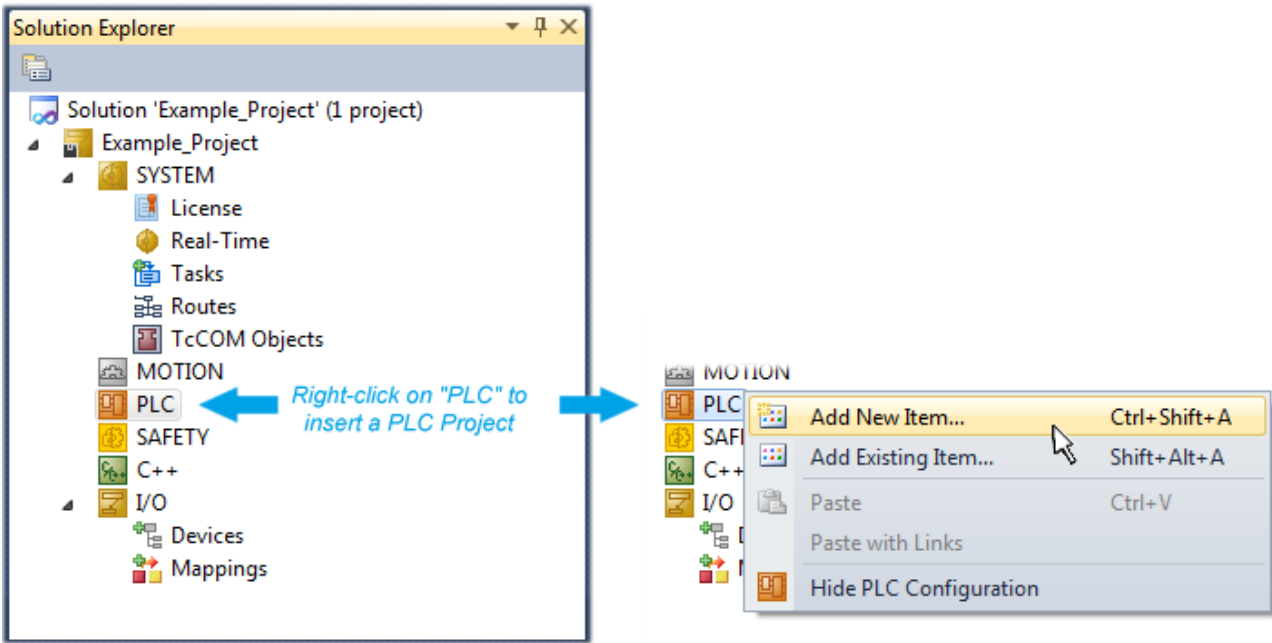


Fig. 59: Adding the programming environment in "PLC"

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

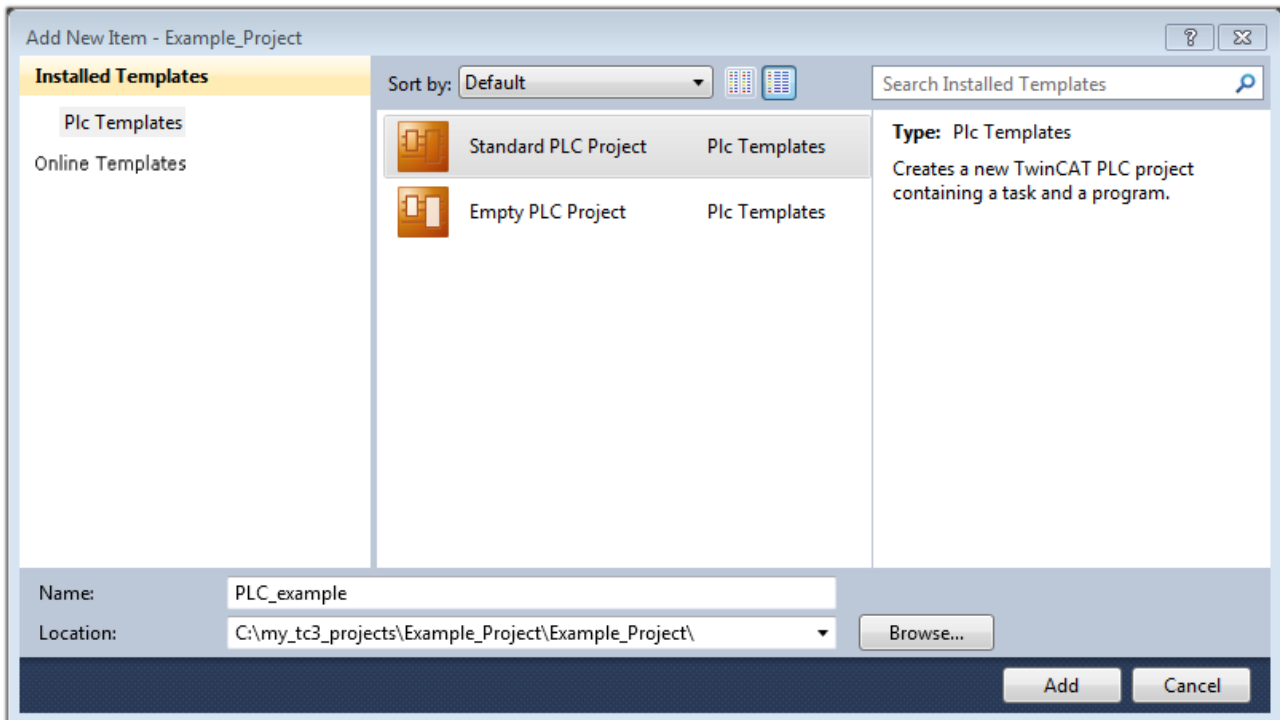


Fig. 60: Specifying the name and directory for the PLC programming environment

The "Main" program, which already exists by 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:

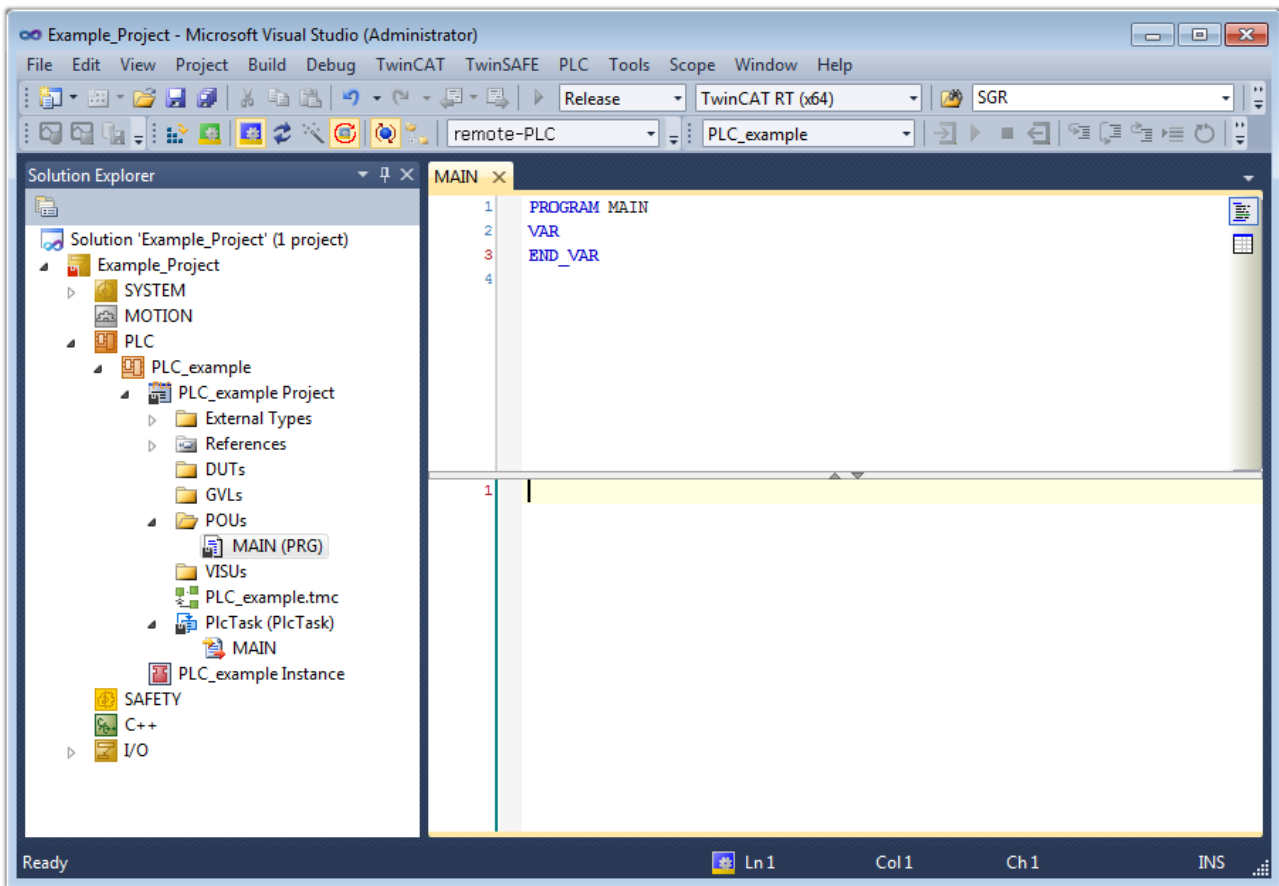


Fig. 61: Initial "Main" program of the standard PLC project

To continue, sample variables and a sample program have now been created:

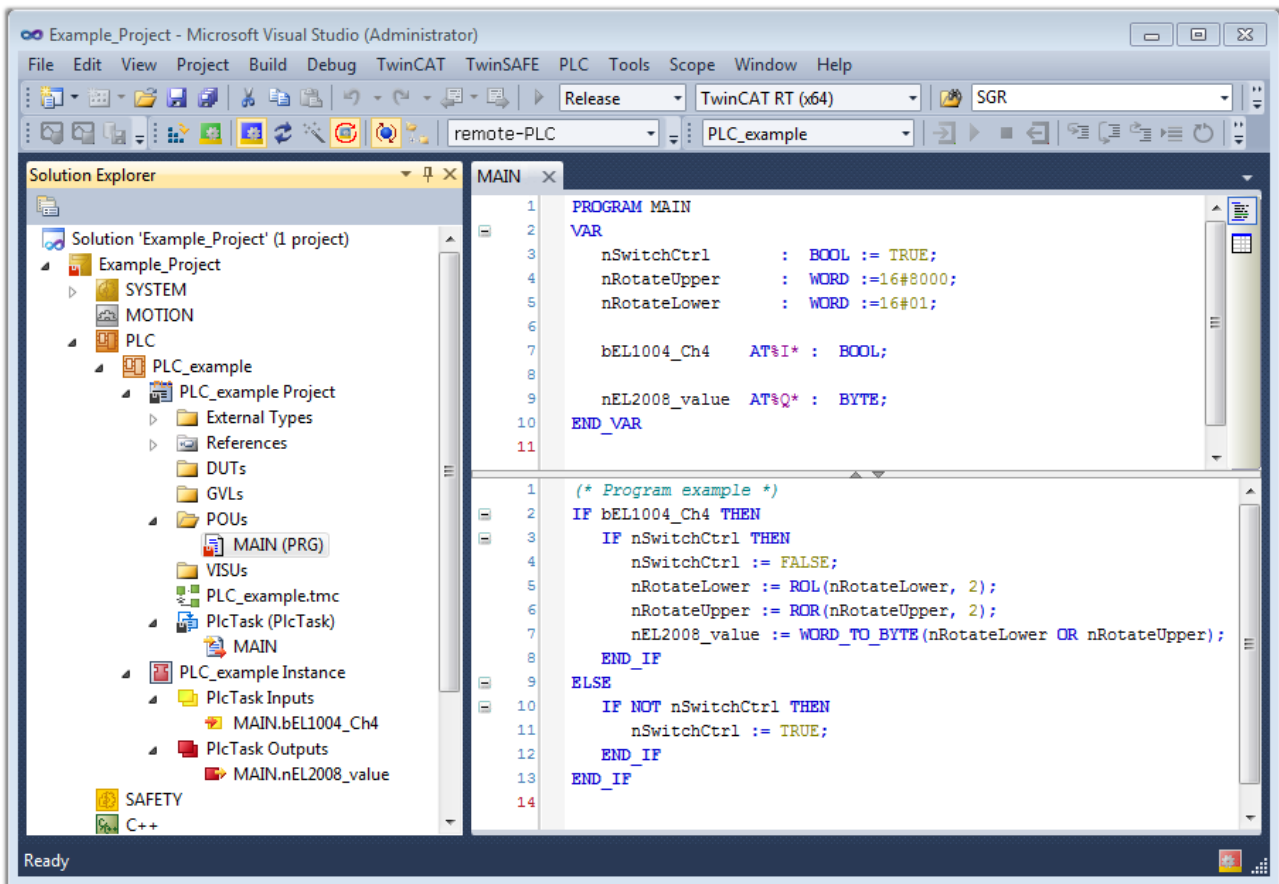


Fig. 62: Sample program with variables after a compile process (without variable integration)

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

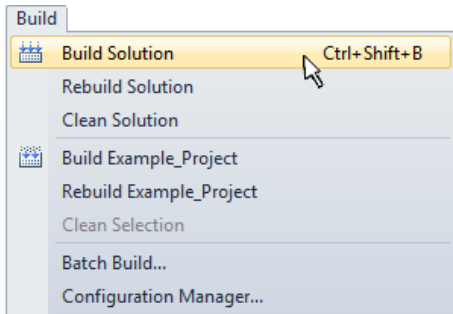
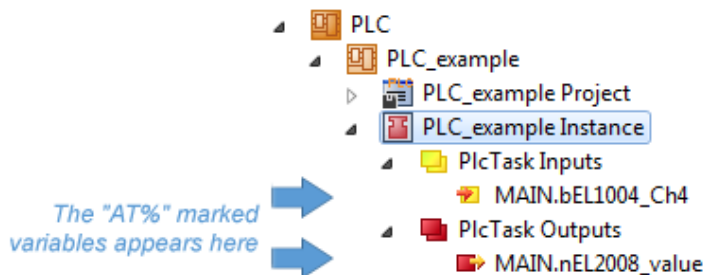


Fig. 63: Start program compilation

The following variables, identified in the ST/ PLC program with "AT%", are then available in 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 for selecting a suitable process object (PDO) for linking:

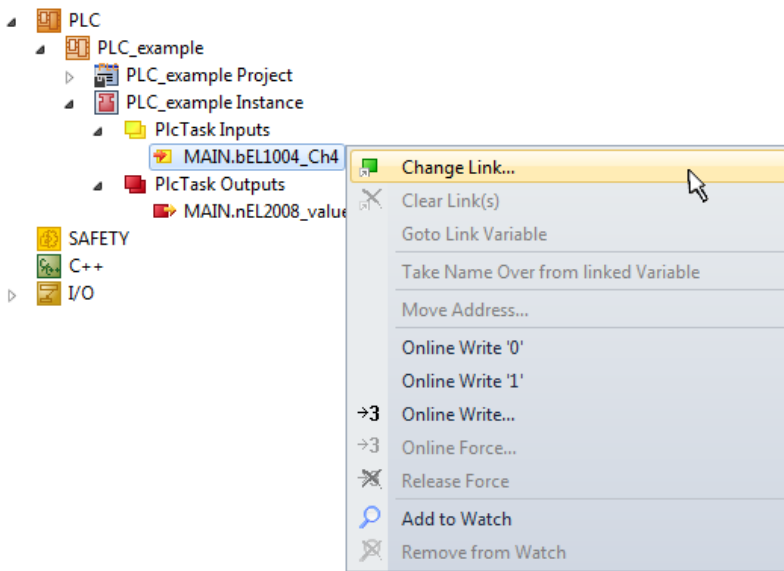


Fig. 64: Creating the links between PLC variables and process objects

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

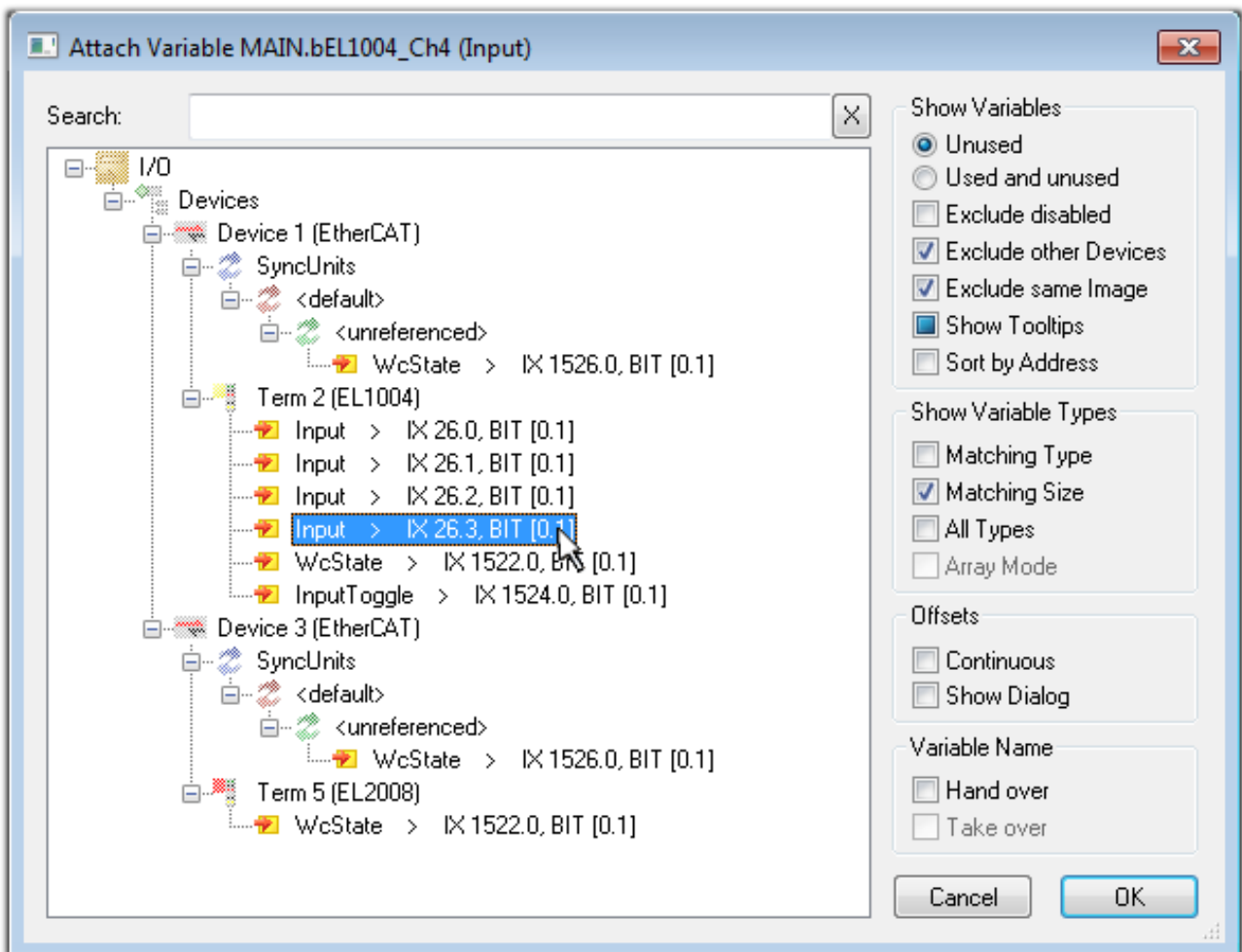


Fig. 65: Selecting PDO of type BOOL

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

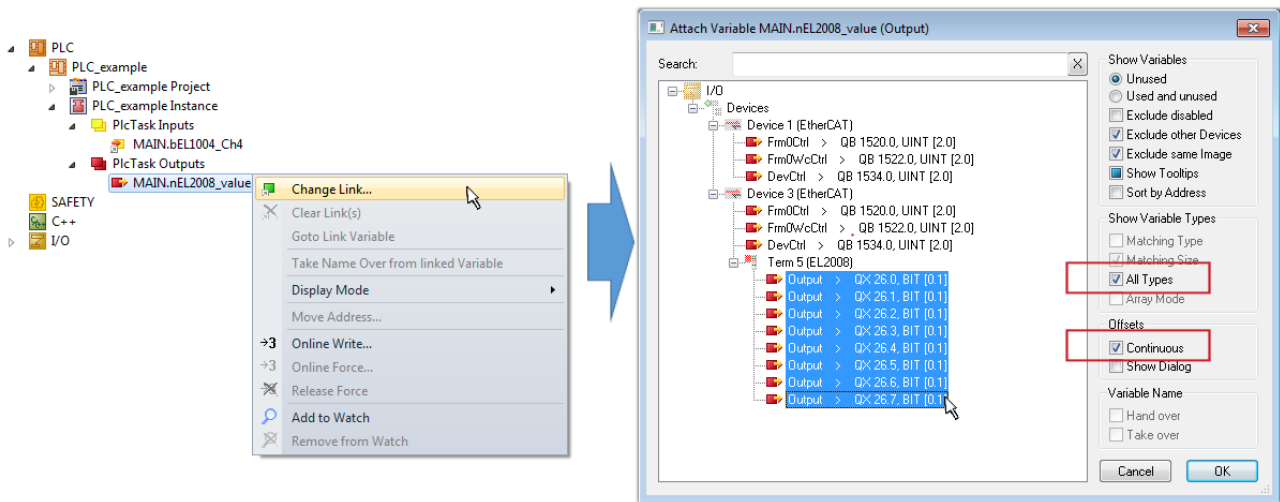



Fig. 66: Selecting several PDOs simultaneously: activate "Continuous" and "All types"

Note that the "Continuous" checkbox was also activated. This is designed to allocate the bits contained in the byte of the variable "nEL2008_value" sequentially to all eight selected output bits of the EL2008 terminal. In this way it is 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 () at the yellow or red object of the variable indicates that a link exists. The links can also be checked by selecting a "Goto Link Variable" from the context menu of a variable. The object opposite, in this case the PDO, is automatically selected:

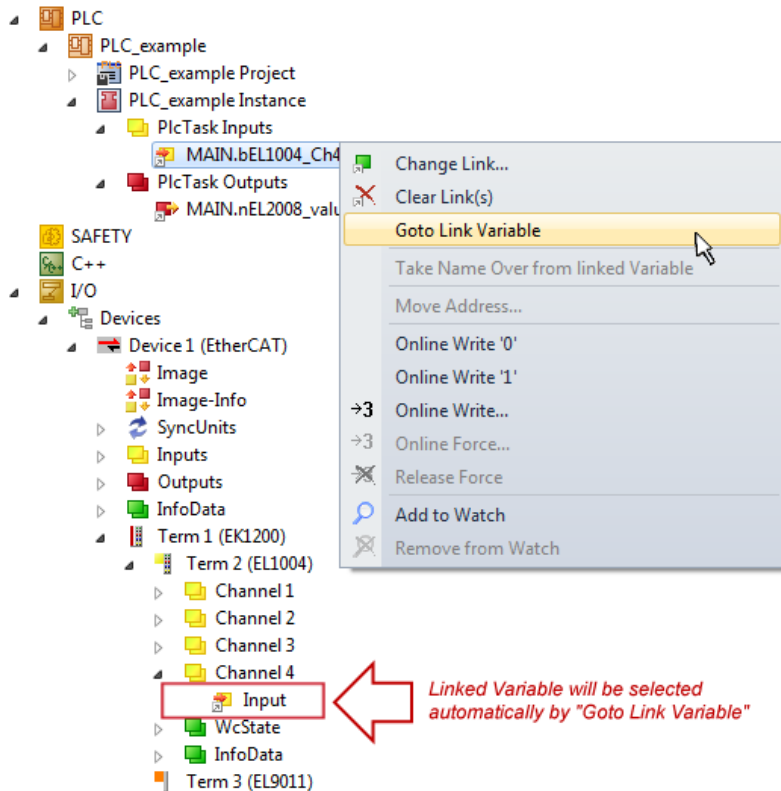

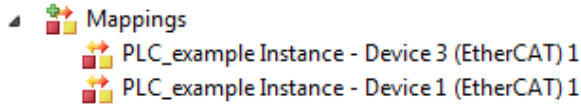



Fig. 67: Application of a "Goto Link" variable, using "MAIN.bEL1004_Ch4" as a sample

The process of creating links can also take place in the opposite direction, i.e. starting with individual PDOs to variable. However, in this example it would then 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 possible to allocate this a set of bit-standardised variables (type "BOOL"). Here, too, a "Goto Link Variable" from the context menu of a PDO 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 with  or via the menu under "TwinCAT" in order to transfer settings of the development environment to the runtime system. Confirm the messages "Old configurations are 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

program start by click on symbol , the "F5" key or via "PLC" in the menu selecting "Start". The started programming environment shows the runtime values of individual variables:

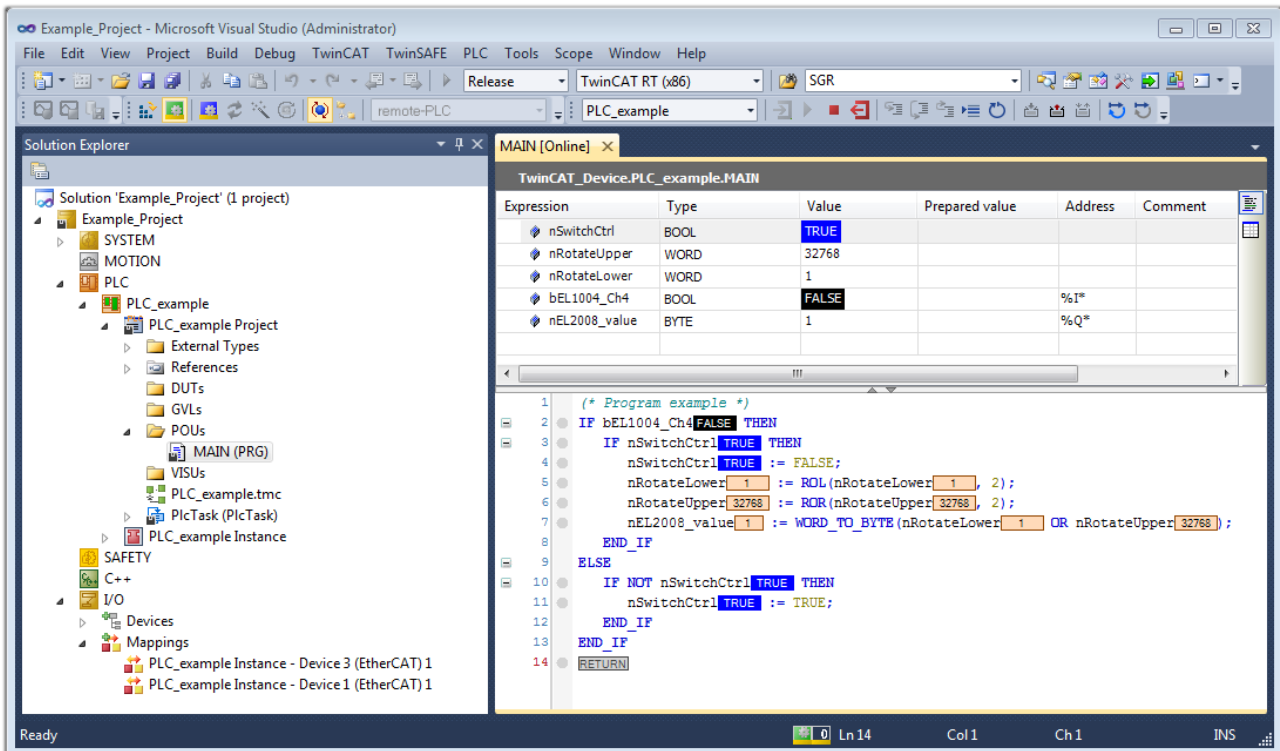


Fig. 68: TwinCAT development environment (VS shell): logged-in, after program startup

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

5.2 TwinCAT Development Environment

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

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

Details:

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

Additional features:

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

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

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

5.2.1 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. One option is described here.

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

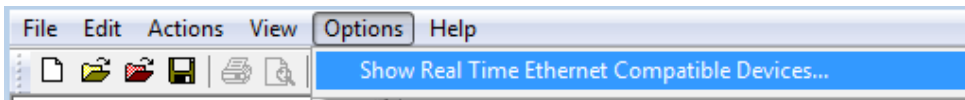


Fig. 69: System Manager “Options” (TwinCAT 2)

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

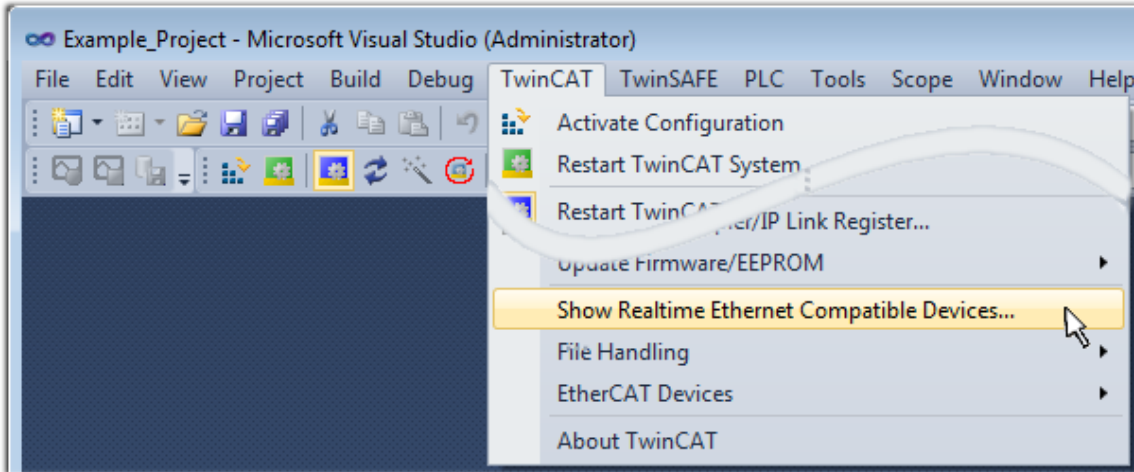


Fig. 70: Call up under VS Shell (TwinCAT 3)

The following dialog appears:

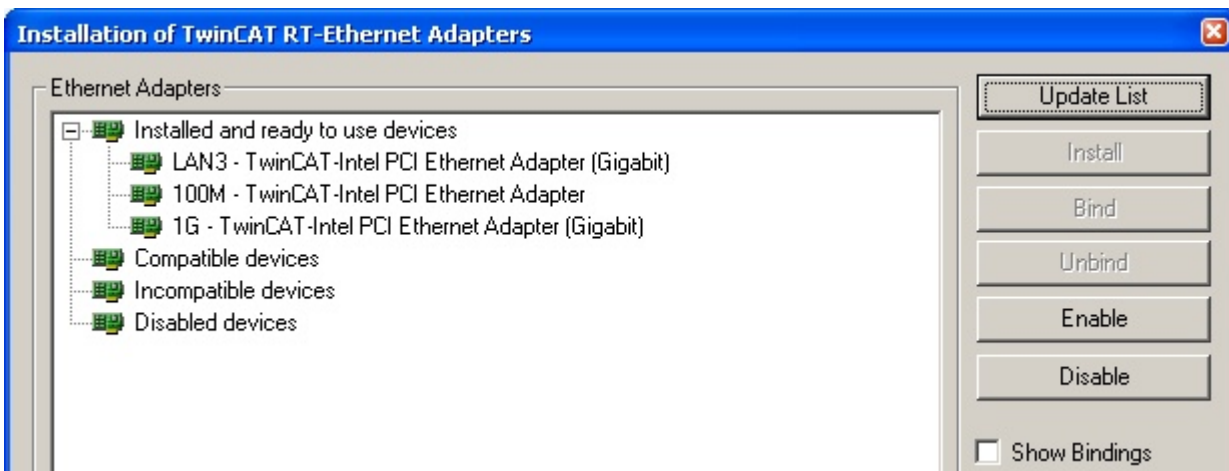


Fig. 71: Overview of network interfaces

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

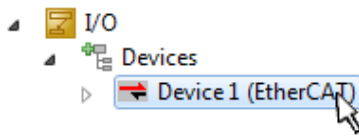
A Windows warning regarding the unsigned driver can be ignored.

Alternatively an EtherCAT-device can be inserted first of all as described in chapter [Offline configuration creation, section “Creating the EtherCAT device” \[▶ 81\]](#) in order to view the compatible ethernet ports via its EtherCAT properties (tab „Adapter“, button „Compatible Devices...“):



Fig. 72: 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)

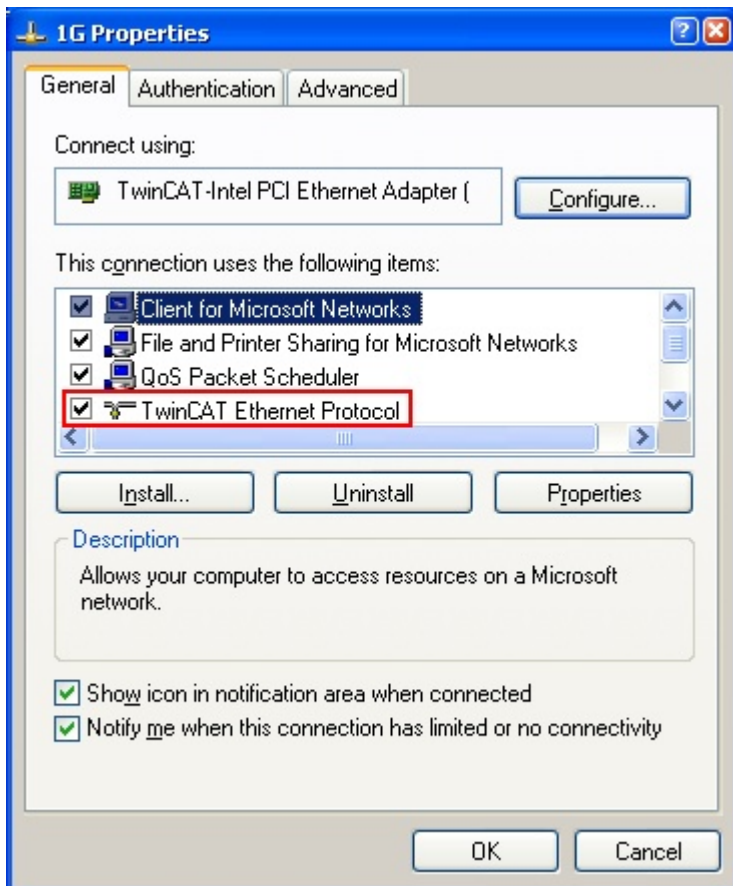


Fig. 73: Windows properties of the network interface

A correct setting of the driver could be:

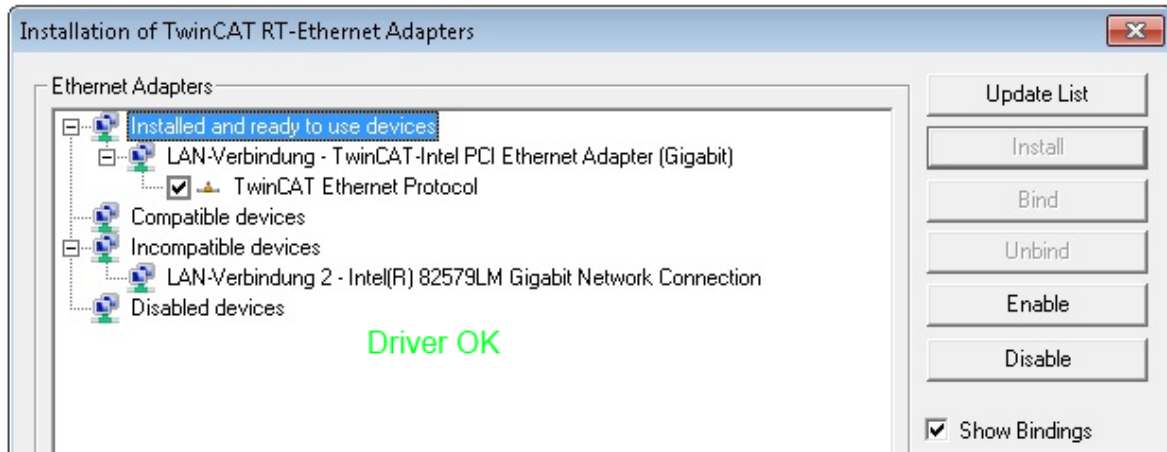


Fig. 74: Exemplary correct driver setting for the Ethernet port

Other possible settings have to be avoided:

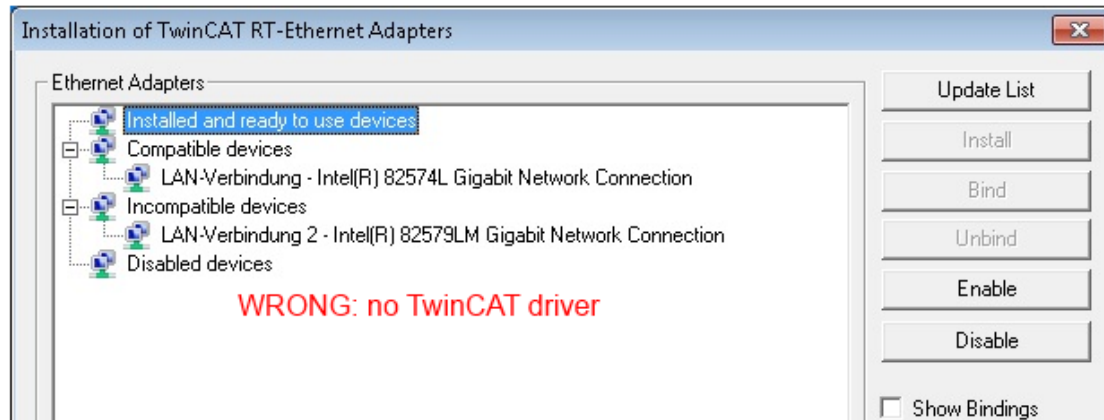
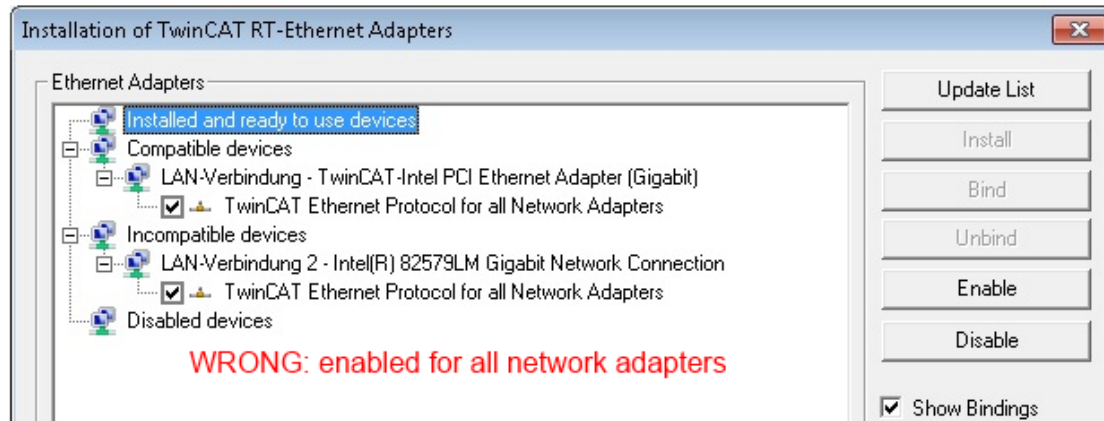
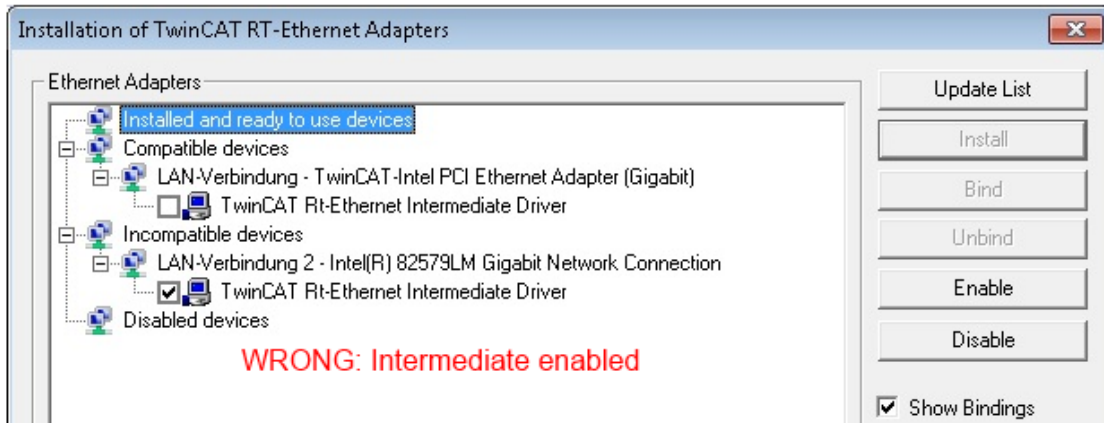
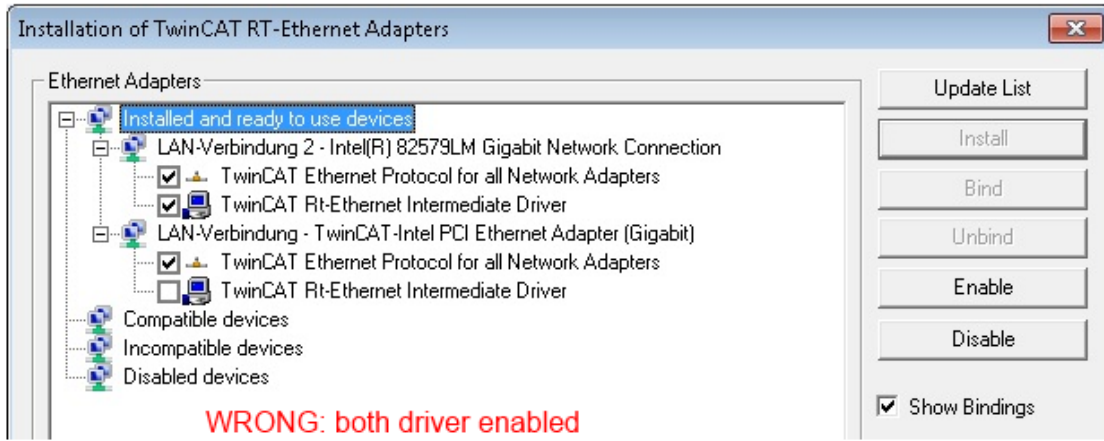


Fig. 75: Incorrect driver settings for the Ethernet port

IP address of the port used

i 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.

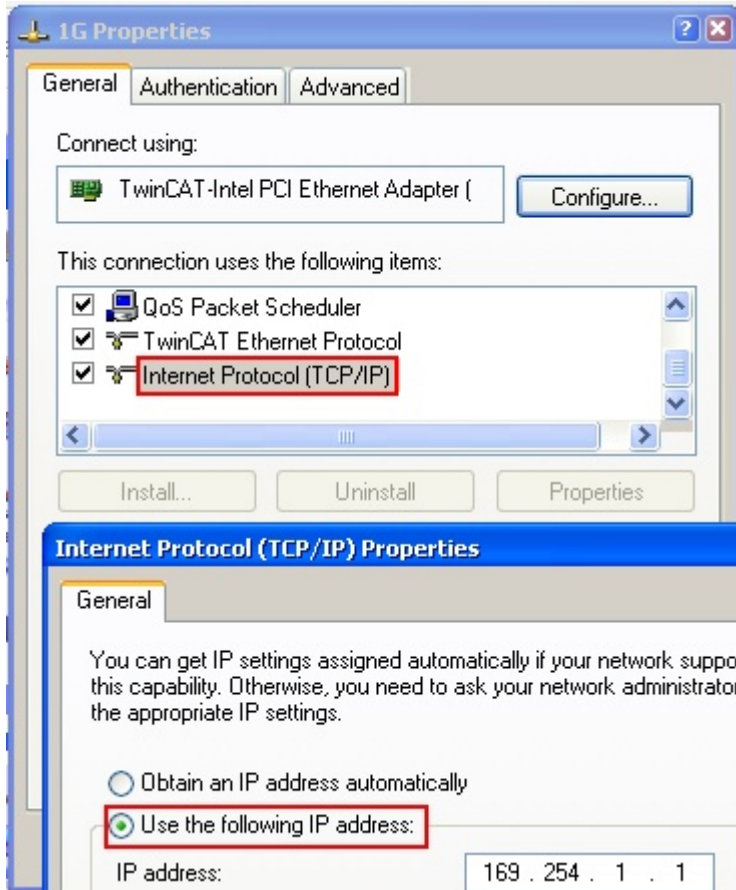


Fig. 76: TCP/IP setting for the Ethernet port

5.2.2 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](#) [► 80] is available for this purpose.



ESI

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

Device differentiation

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

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

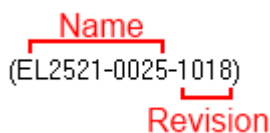


Fig. 77: Identifier structure

The order identifier consisting of name + type (here: EL2521-0010) describes the device function. The revision indicates the technical progress and is managed by Beckhoff. In principle, a device with a higher revision can replace a device with a lower revision, unless specified otherwise, e.g. in the documentation. Each revision has its own ESI description. See [further notes](#) [► 11].

Online description

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

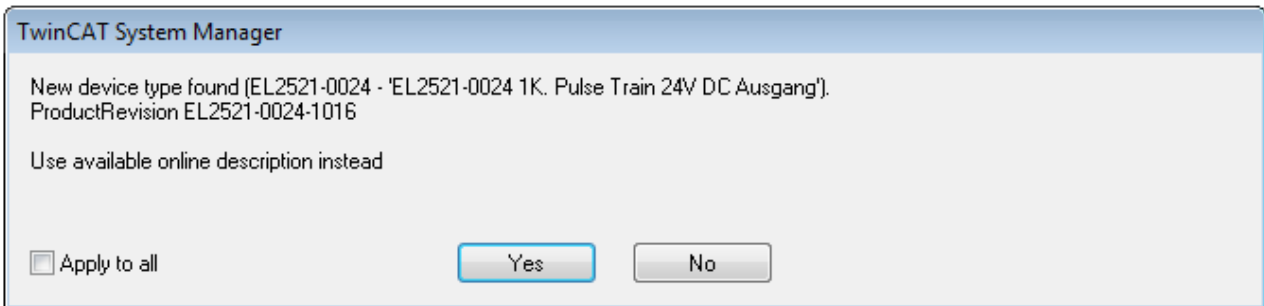


Fig. 78: *OnlineDescription information window (TwinCAT 2)*

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

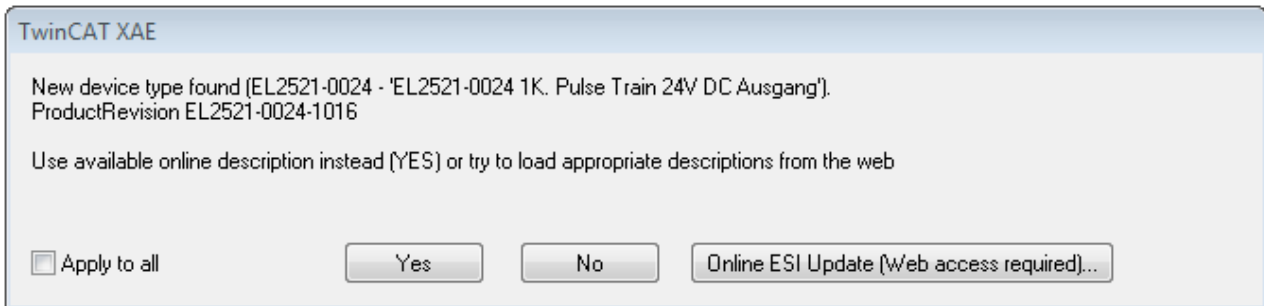


Fig. 79: *Information window OnlineDescription (TwinCAT 3)*

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.

NOTE

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’ [► 81].

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

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

OnlineDescriptionCache00000002.xml

Fig. 80: File *OnlineDescription.xml* created by the System Manager

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

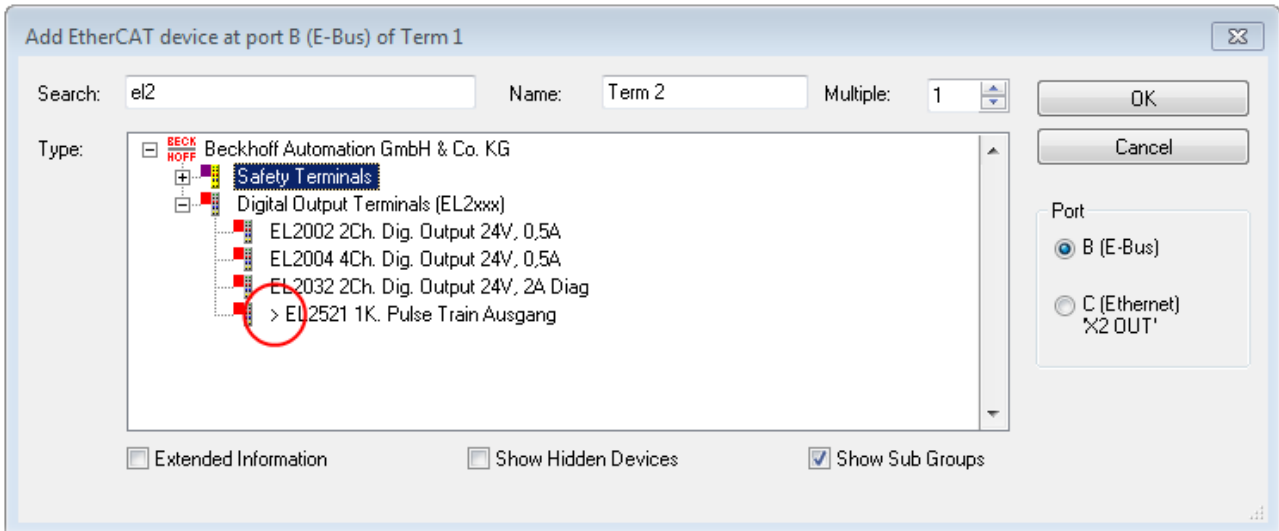


Fig. 81: Indication of an online recorded ESI of EL2521 as an example

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

i 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.

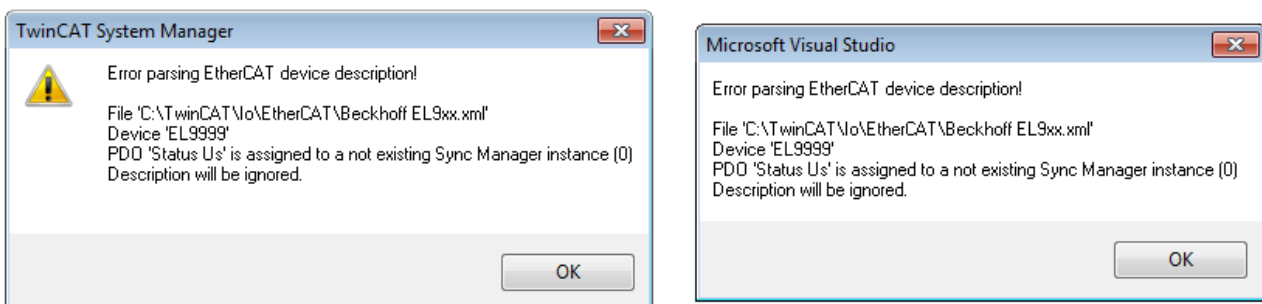


Fig. 82: Information window for faulty ESI file (left: TwinCAT 2; right: TwinCAT 3)

Reasons may include:

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

5.2.3 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:

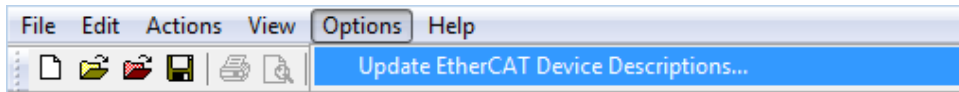


Fig. 83: Using the ESI Updater (\geq TwinCAT 2.11)

The call up takes place under:

“Options” → “Update EtherCAT Device Descriptions”

Selection under TwinCAT 3:

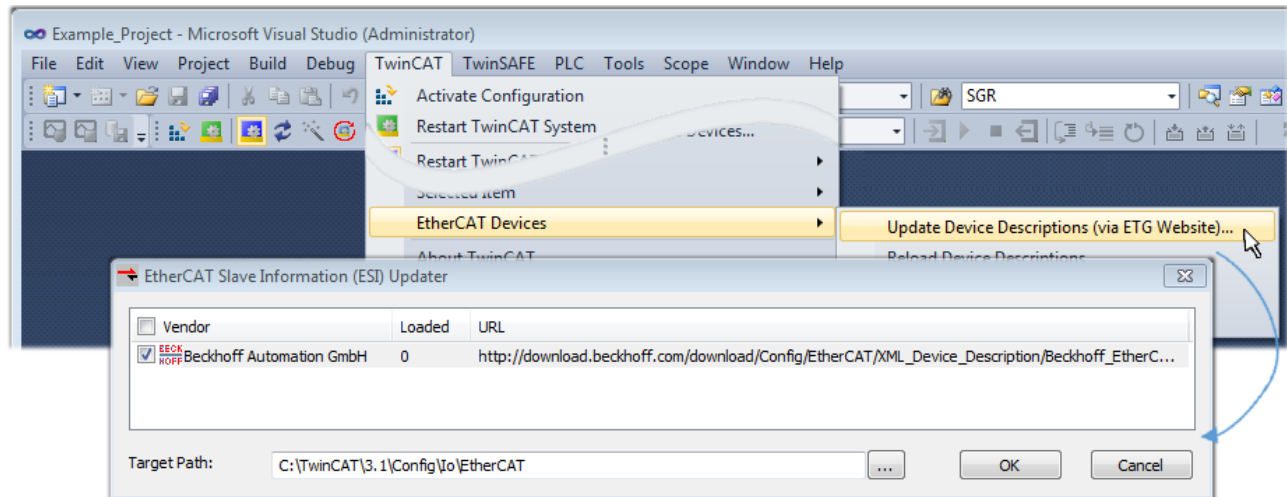


Fig. 84: Using the ESI Updater (TwinCAT 3)

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)...”.

5.2.4 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” \[▶ 76\]](#).

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 [▶ 86] (Ethernet port at the IPC)
- detecting the connected EtherCAT devices [▶ 87]. This step can be carried out independent of the preceding step
- troubleshooting [▶ 90]

The scan with existing configuration [▶ 91] can also be carried out for comparison.

5.2.5 OFFLINE configuration creation

Creating the EtherCAT device

Create an EtherCAT device in an empty System Manager window.

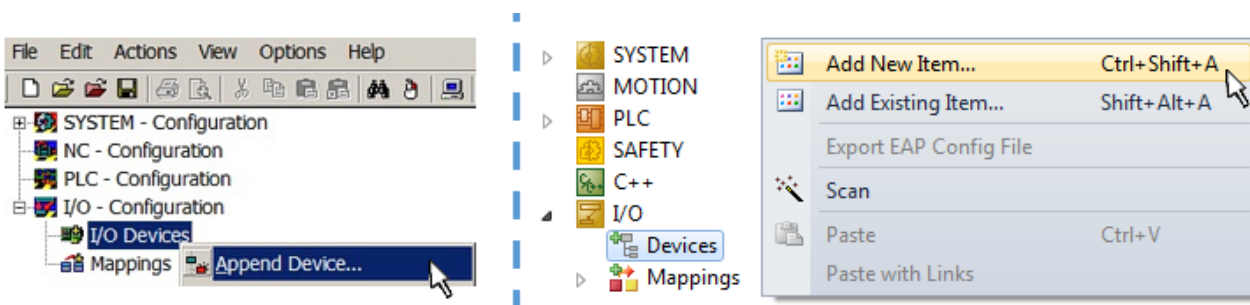


Fig. 85: Append EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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".

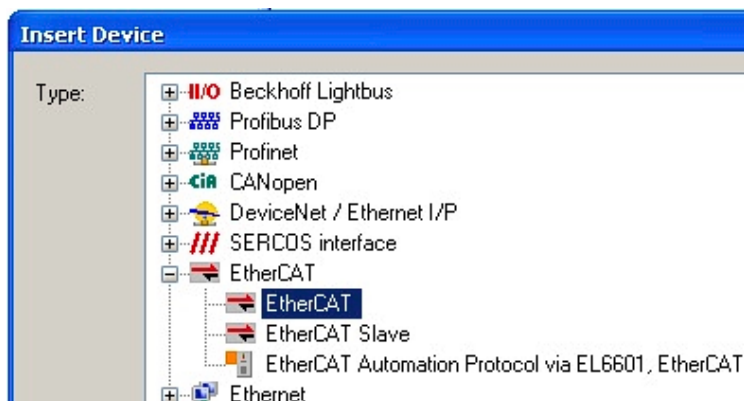


Fig. 86: Selecting the EtherCAT connection (TwinCAT 2.11, TwinCAT 3)

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

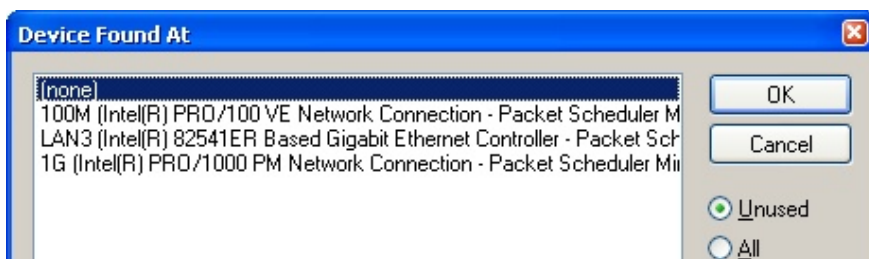


Fig. 87: Selecting the Ethernet port

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)”.

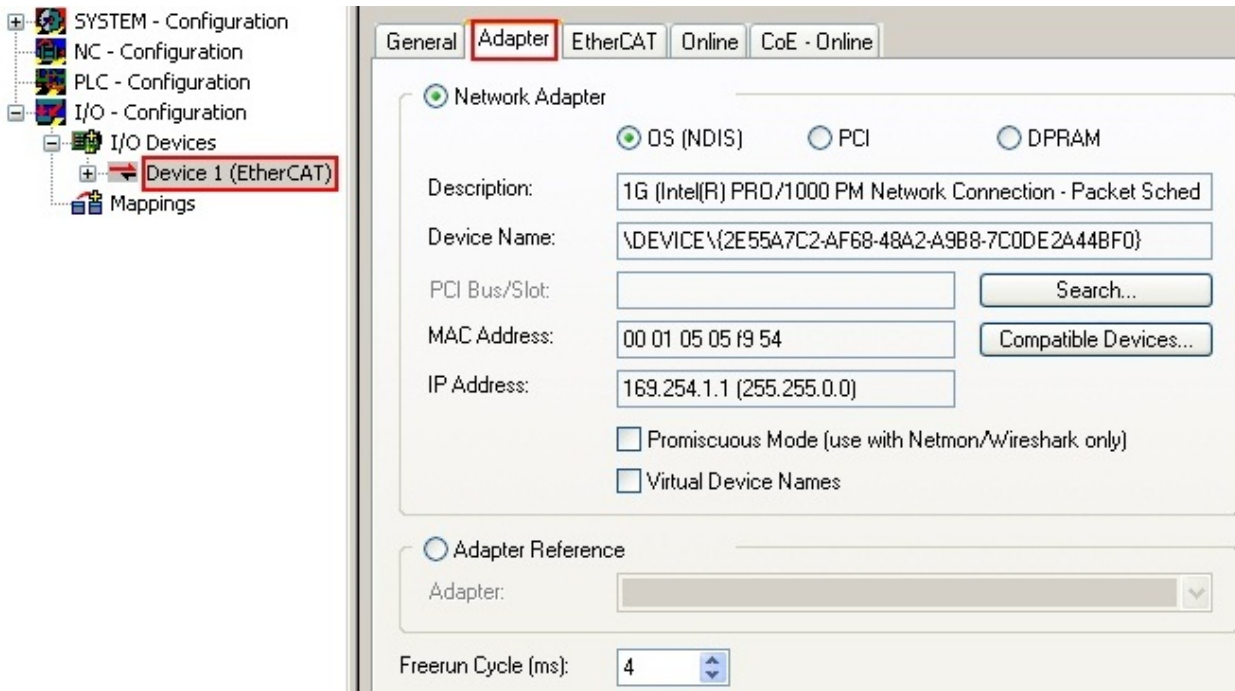
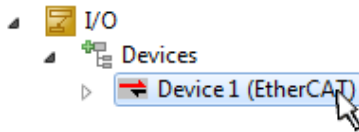


Fig. 88: EtherCAT device properties (TwinCAT 2)

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



i **Selecting the Ethernet port**

Ethernet ports can only be selected for EtherCAT devices for which the TwinCAT real-time driver is installed. This has to be done separately for each port. Please refer to the respective [installation page \[▶ 70\]](#).

Defining EtherCAT slaves

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

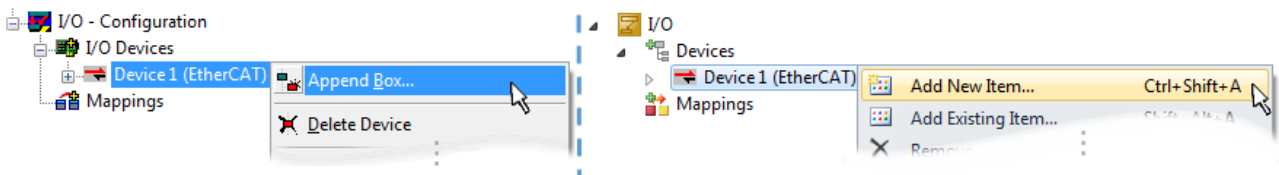


Fig. 89: Appending EtherCAT devices (left: TwinCAT 2; right: TwinCAT 3)

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: EK couplers, EP boxes, devices with RJ45/M8/M12 connector

- “E-Bus”: LVDS “terminal bus”, “EJ-module”: EL/ES terminals, various modular modules

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

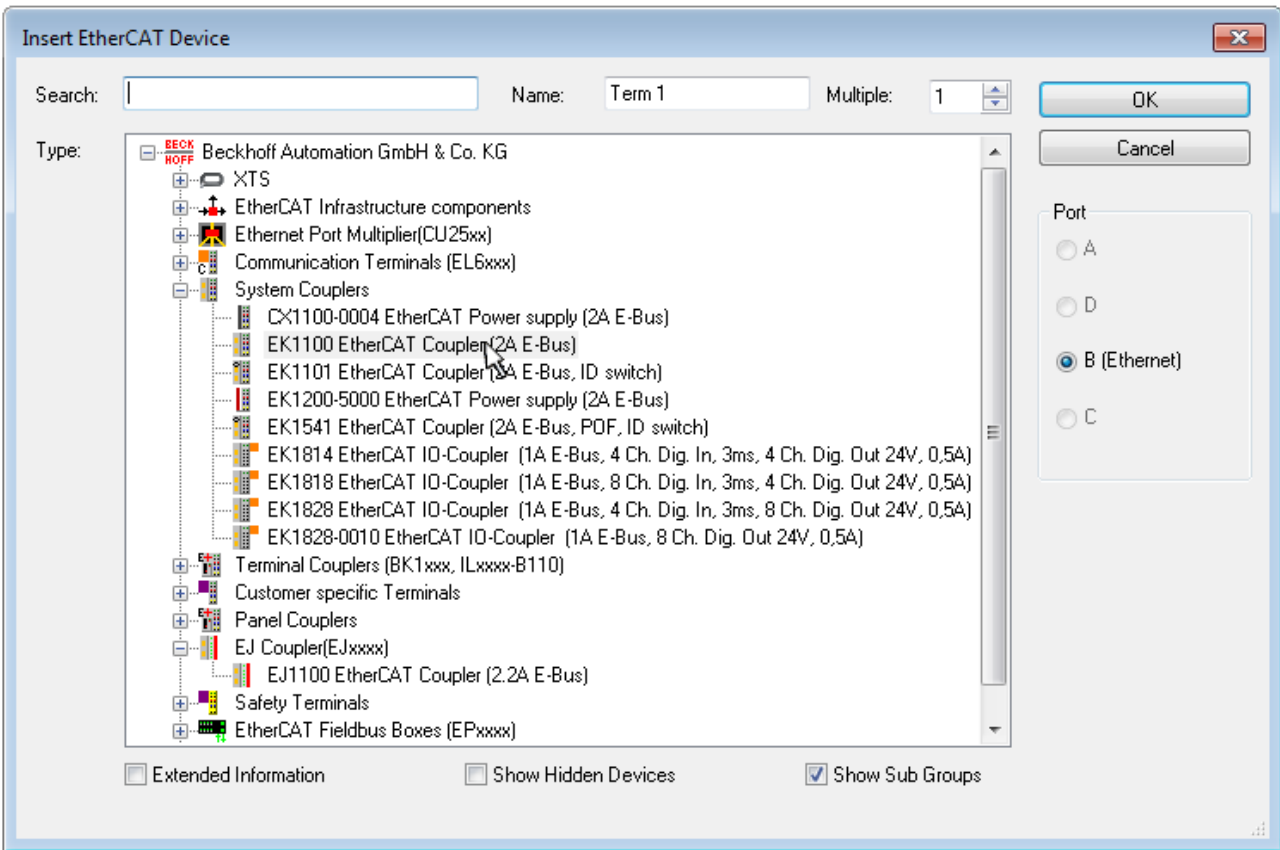


Fig. 90: Selection dialog for new EtherCAT device

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”.

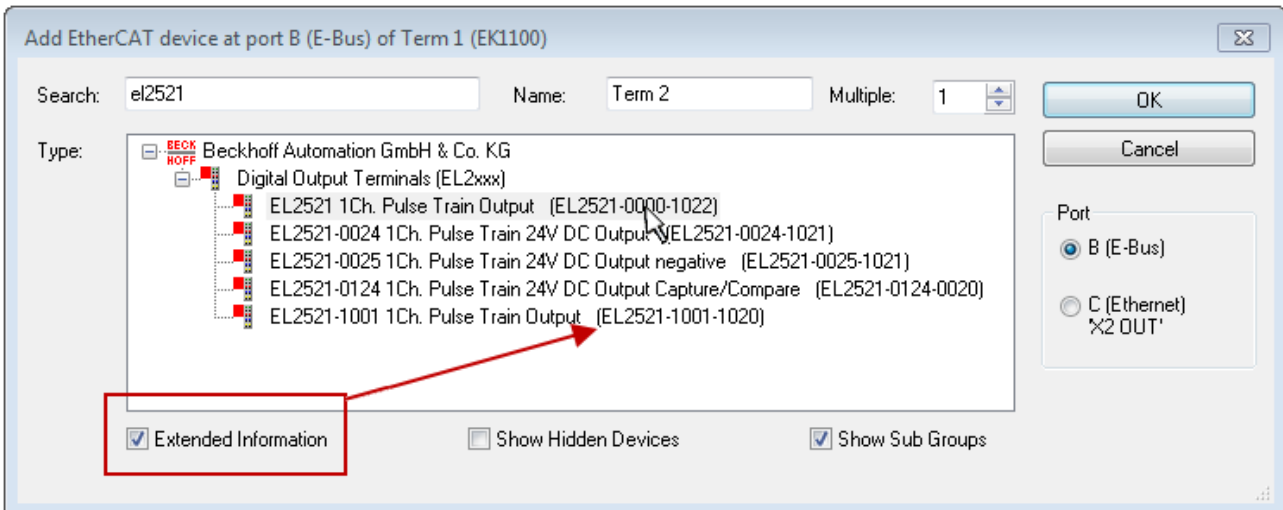


Fig. 91: Display of device revision

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”.

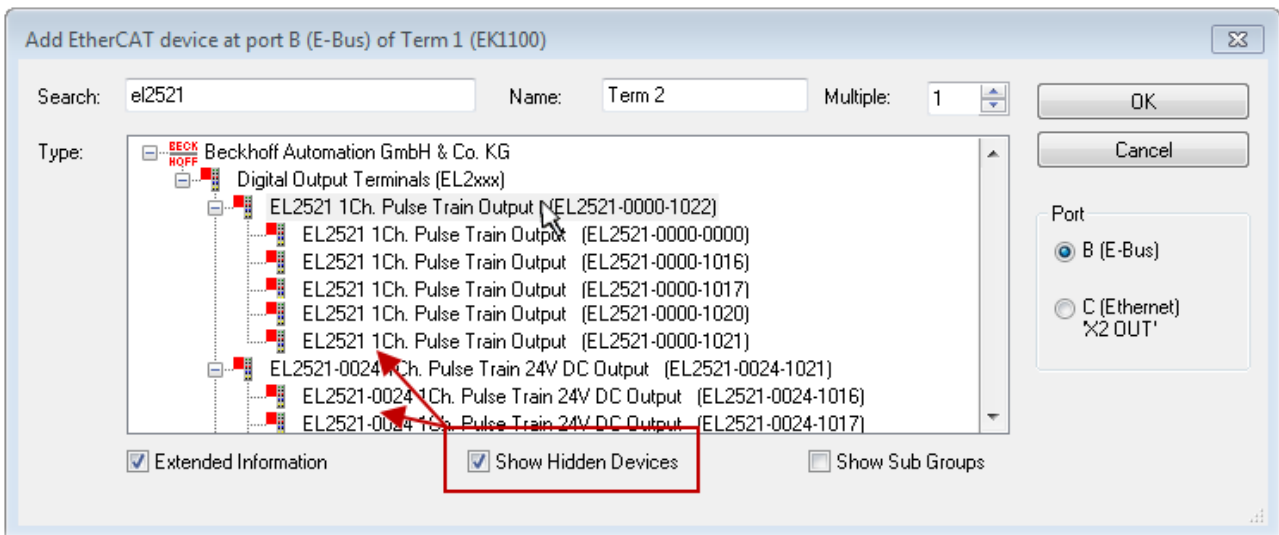


Fig. 92: *Display of previous revisions*

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 >= device revision in the configuration

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

Example:

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

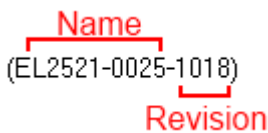


Fig. 93: *Name/revision of the terminal*



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 parameterised as follows: linking with the task, CoE/DC settings, plug-in definition, startup settings, ...



5.2.6 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)”

● Online scanning in Config mode

i 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.

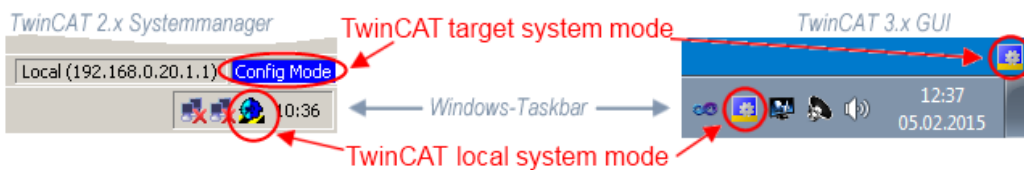


Fig. 95: Differentiation local/target system (left: TwinCAT 2; right: TwinCAT 3)

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

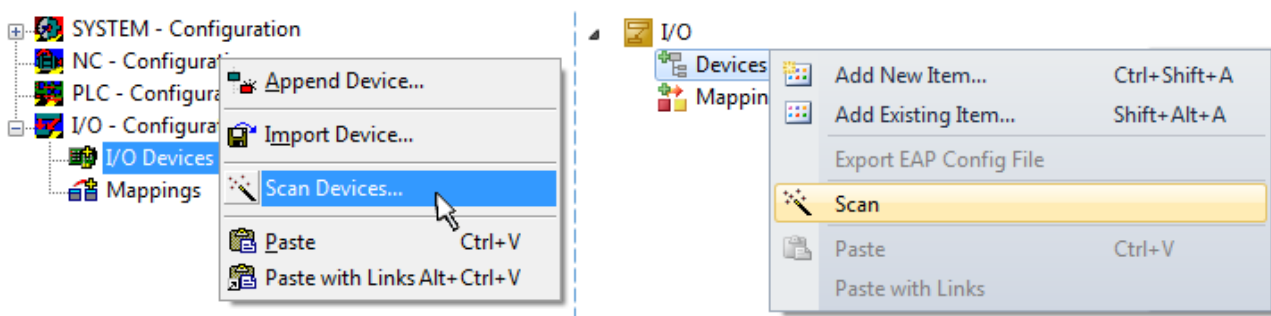


Fig. 96: Scan Devices (left: TwinCAT 2; right: TwinCAT 3)

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

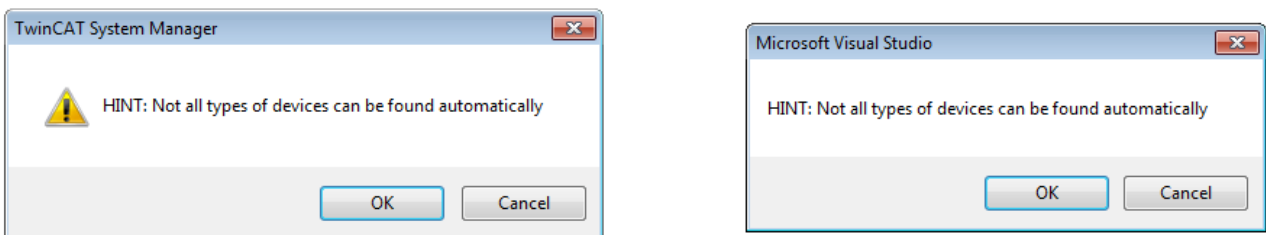


Fig. 97: Note for automatic device scan (left: TwinCAT 2; right: TwinCAT 3)

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” .

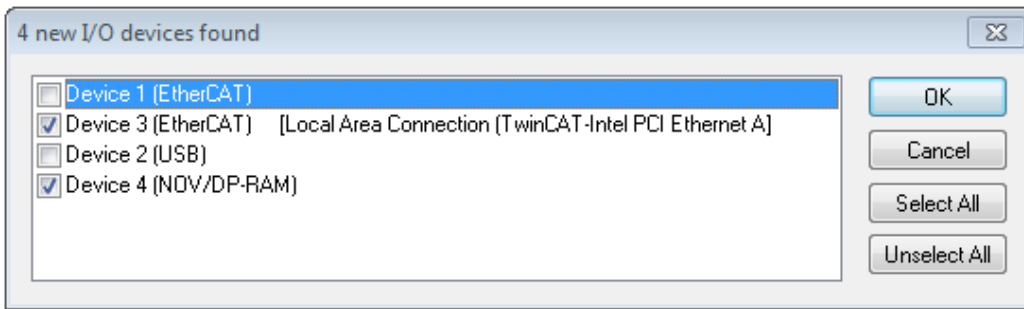


Fig. 98: Detected Ethernet devices

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

● **Selecting the Ethernet port**



Ethernet ports can only be selected for EtherCAT devices for which the TwinCAT real-time driver is installed. This has to be done separately for each port. Please refer to the respective [installation page](#) |▶ 70|.

Detecting/Scanning the EtherCAT devices

● **Online scan functionality**



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

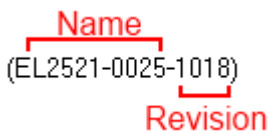


Fig. 99: Example default state

NOTE

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](#) |▶ 91| 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:

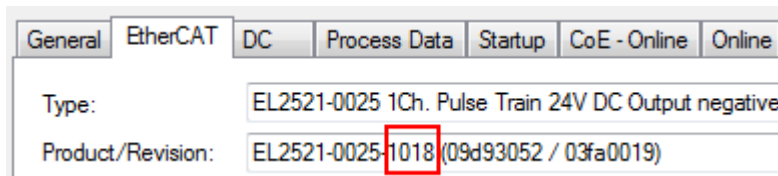


Fig. 100: Installing EtherCAT terminal with revision -1018

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 [▶ 91](#) 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:

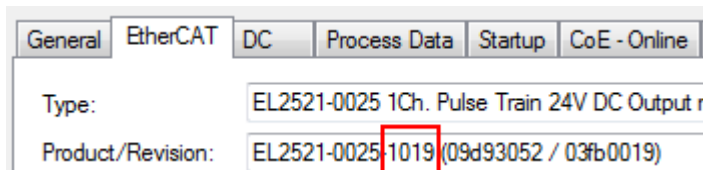


Fig. 101: Detection of EtherCAT terminal with revision -1019

This is usually not noticed by the commissioning engineers. TwinCAT cannot signal anything either, since virtually a new configuration is 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. 102: Scan query after automatic creation of an EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

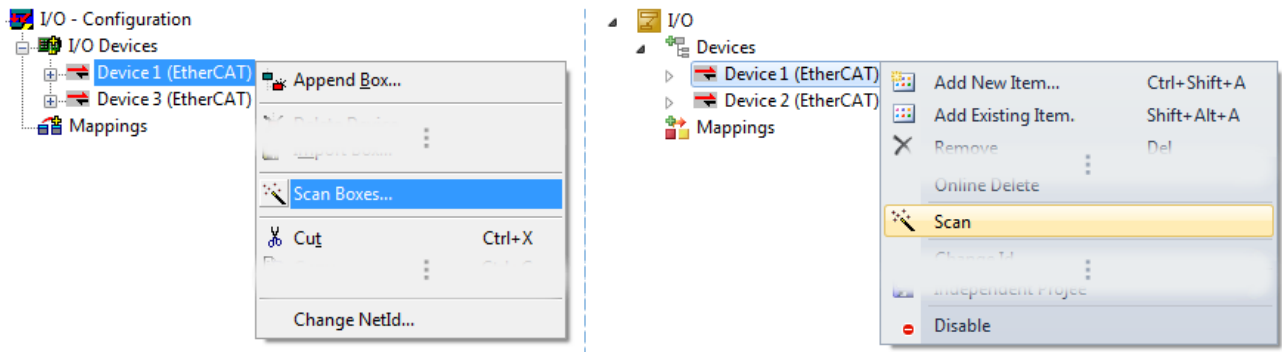


Fig. 103: Manual triggering of a device scan on a specified EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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. 104: Scan progress exemplary by TwinCAT 2

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



Fig. 105: 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. 106: Displaying of “Free Run” and “Config Mode” toggling right below in the status bar



Fig. 107: 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”.

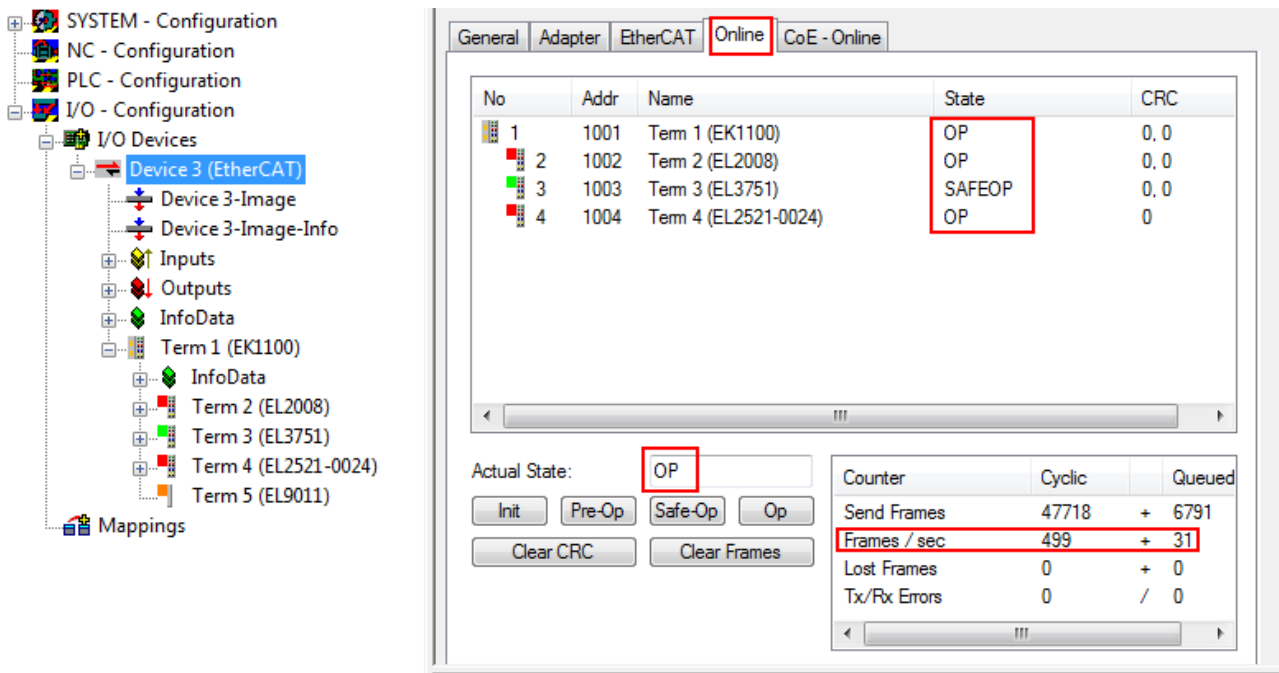


Fig. 108: Online display example

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 \[► 81\]](#).

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.

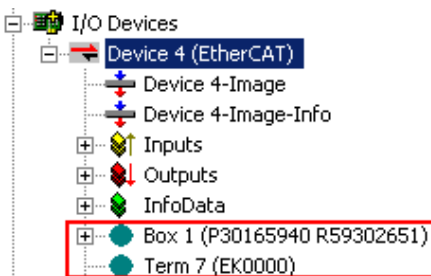


Fig. 109: Faulty identification

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

NOTE

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. 110: 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.

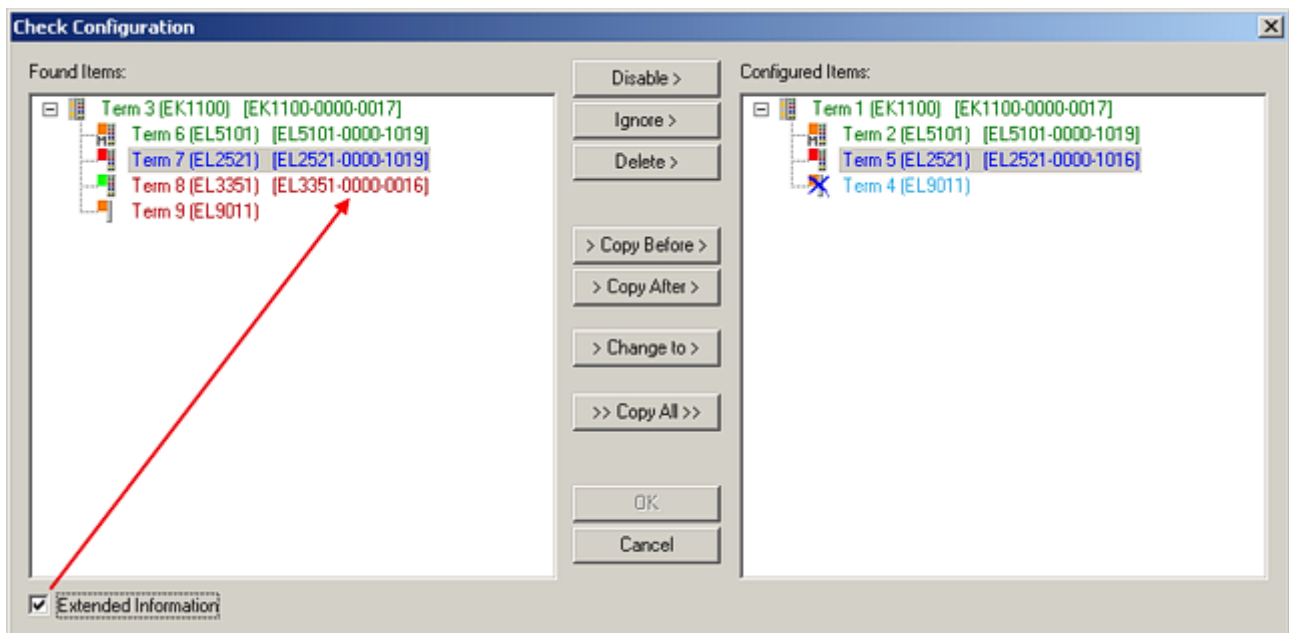


Fig. 111: Correction dialog

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

Colour	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	<ul style="list-style-type: none"> 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. <p>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.</p> <p>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.</p>

● 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. 112: 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 parameterised as follows: linking with the task, CoE/DC settings, plug-in definition, startup settings, ...

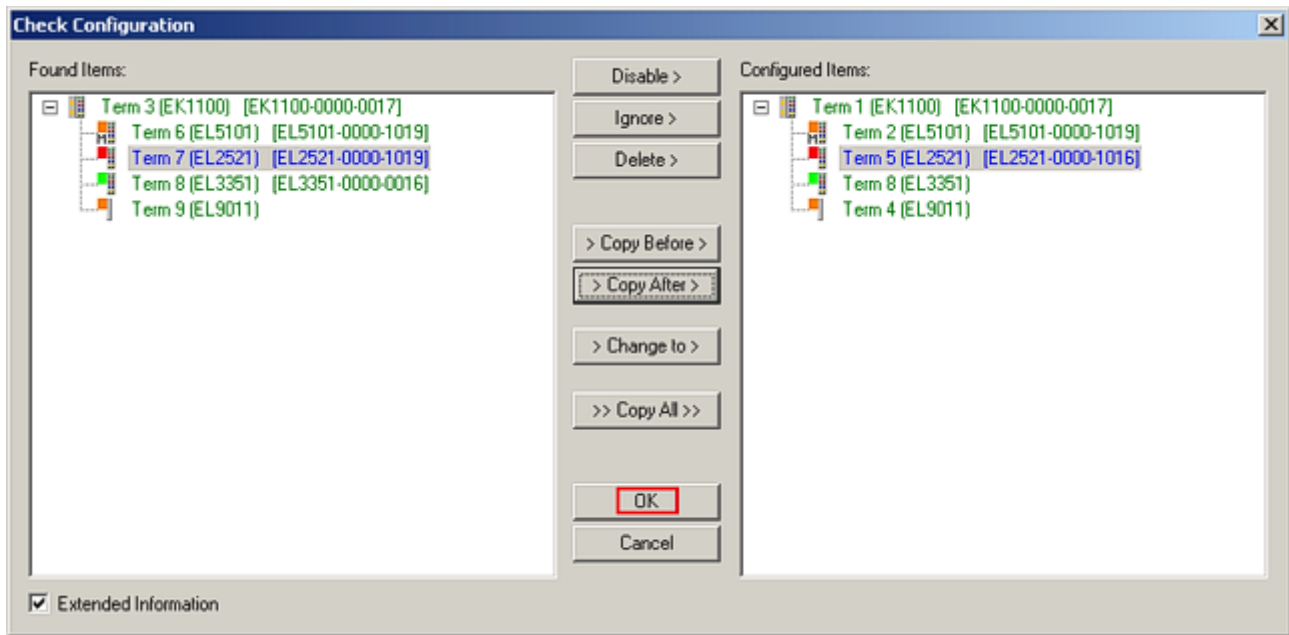


Fig. 113: Correction dialog with modifications

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.

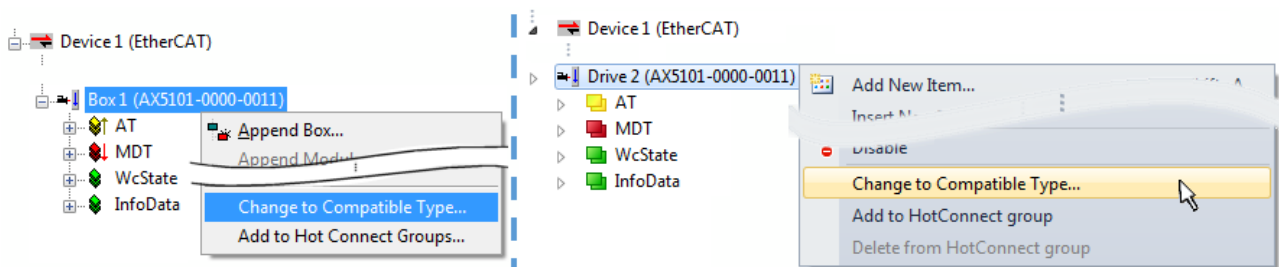


Fig. 114: Dialog “Change to Compatible Type...” (left: TwinCAT 2; right: TwinCAT 3)

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*

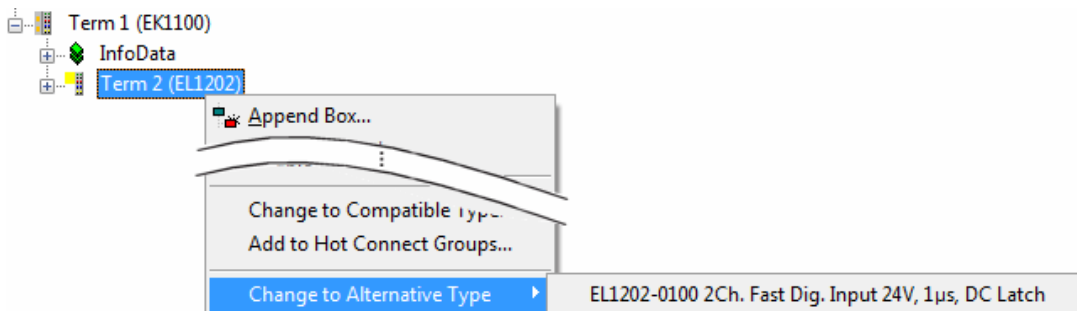


Fig. 115: TwinCAT 2 Dialog Change to Alternative Type

If called, the System Manager searches in the procured device ESI (in this example: EL 1202-0000) for details of compatible devices contained there. The configuration is changed and the ESI-EEPROM is overwritten at the same time – therefore this process is possible only in the online state (ConfigMode).

5.2.7 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).

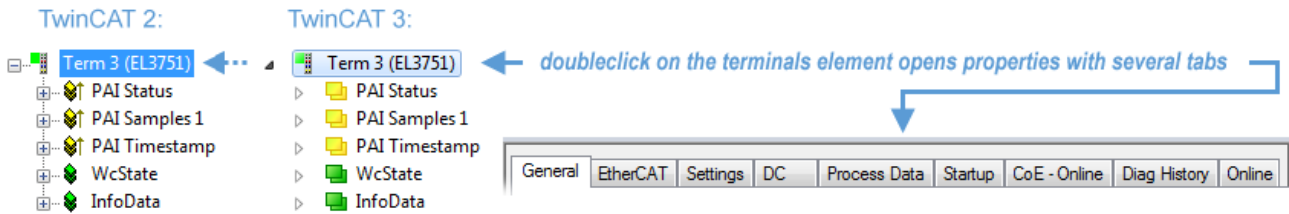


Fig. 116: Branch element as terminal EL3751

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

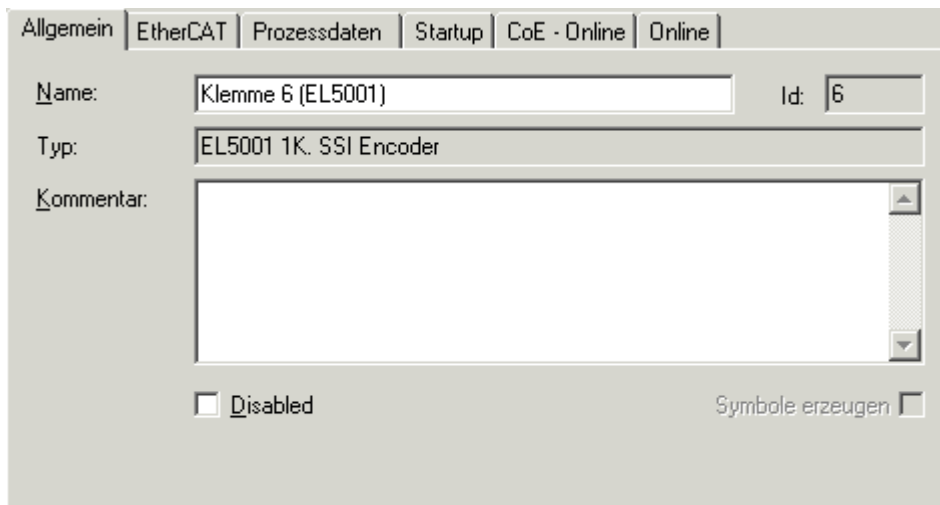


Fig. 117: "General" tab

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

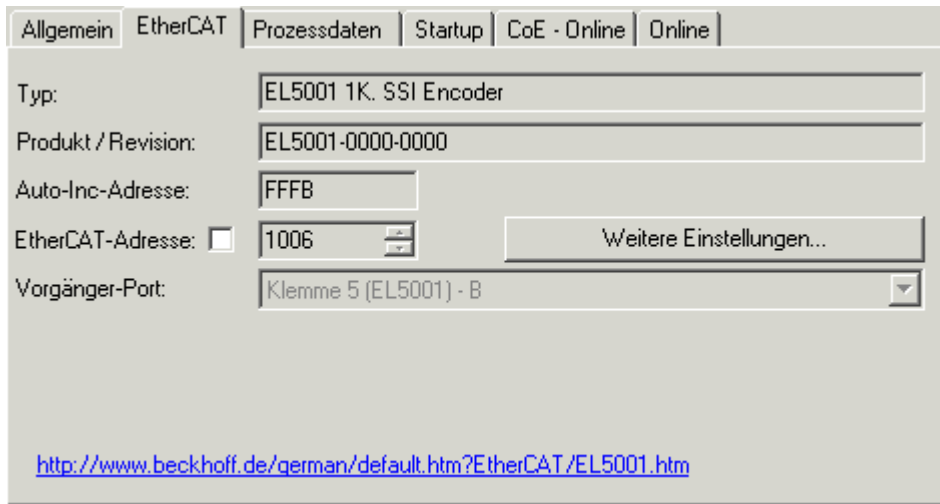


Fig. 118: „EtherCAT“ tab

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.

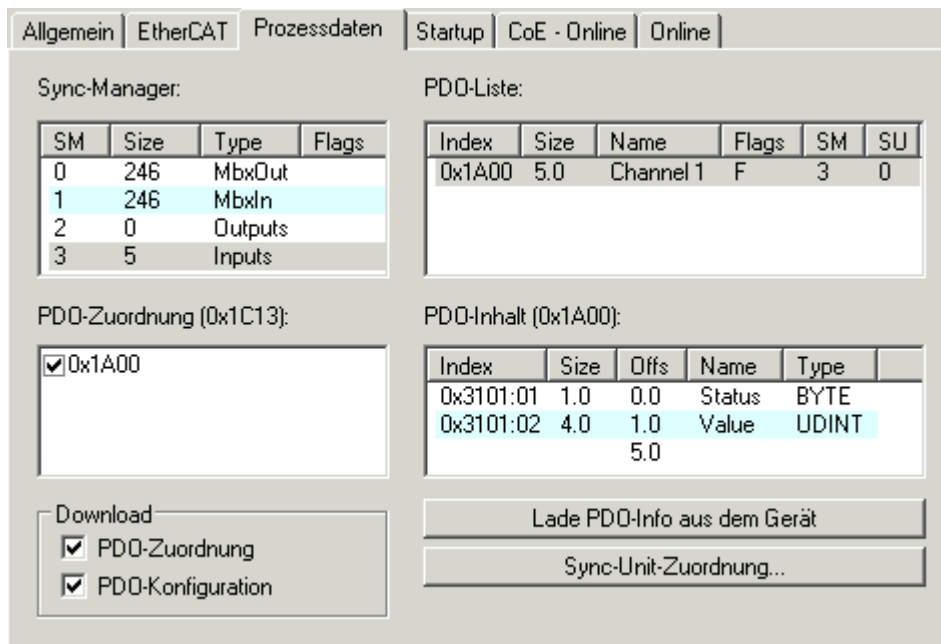


Fig. 119: "Process Data" tab

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").

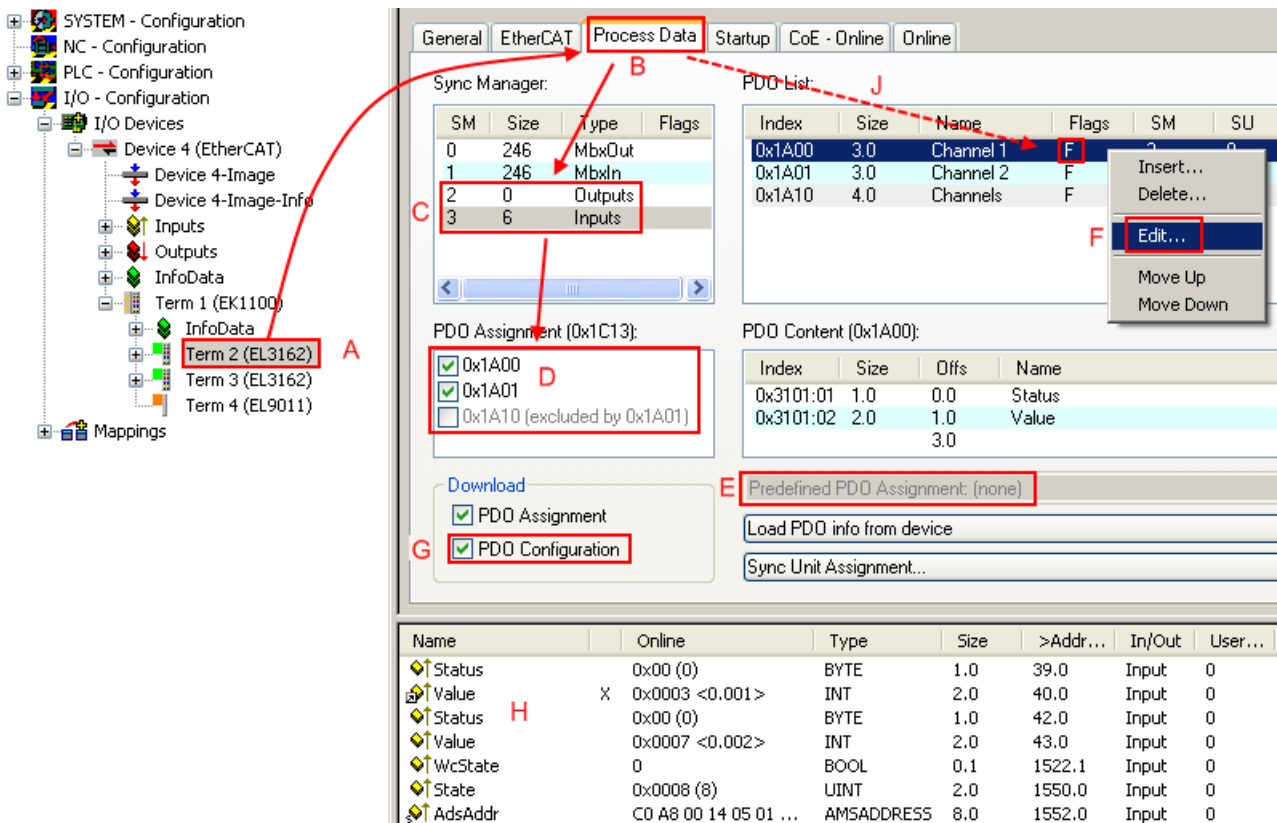


Fig. 120: Configuring the process data

i 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” log-ger message: This error message (“invalid SM IN cfg” or “invalid SM OUT cfg”) also indicates the reason for the failed start.

A detailed description [▶ 102] 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.

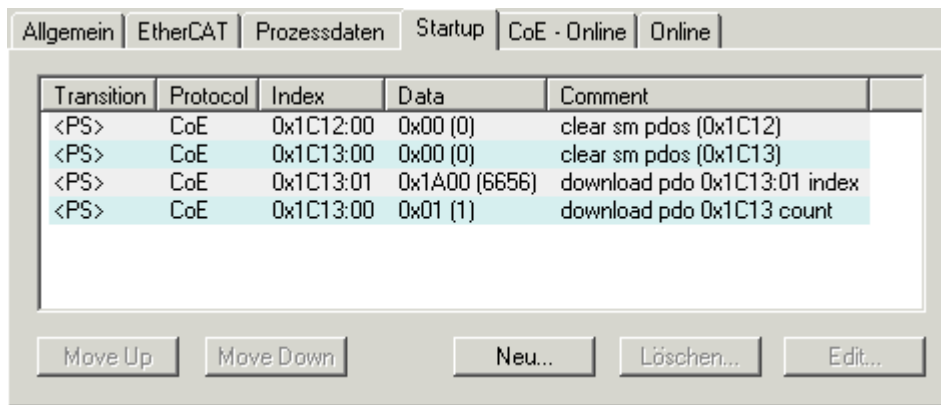


Fig. 121: „Startup“ tab

Column	Description
Transition	Transition to which the request is sent. This can either be <ul style="list-style-type: none"> 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.

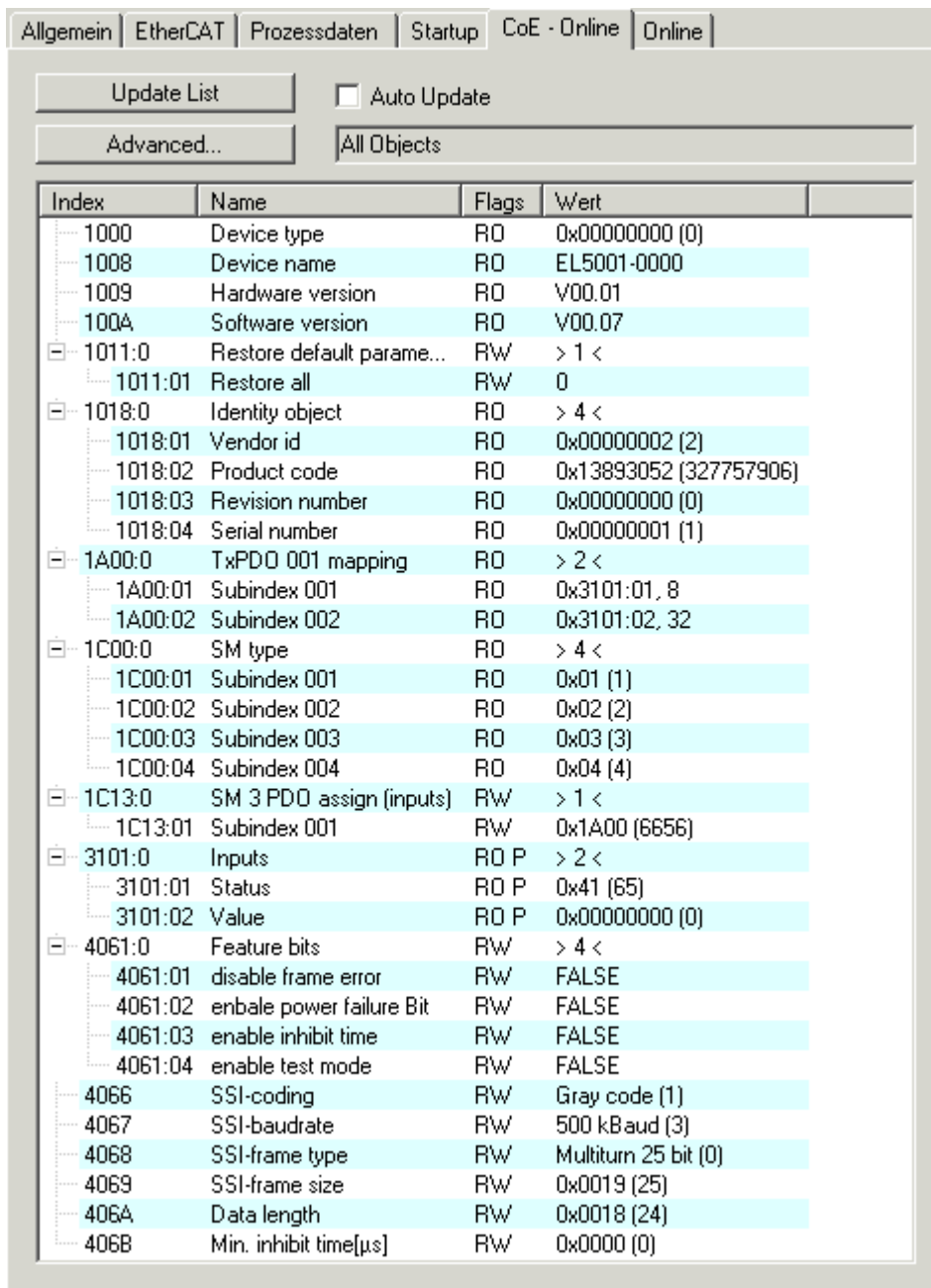


Fig. 122: "CoE – Online" tab

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.

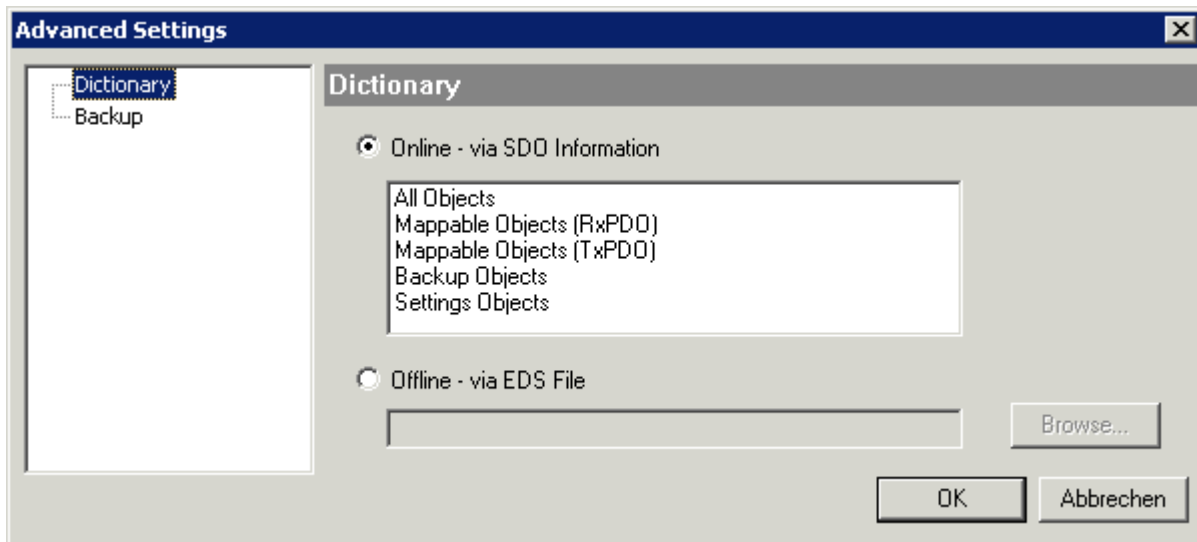


Fig. 123: Dialog "Advanced settings"

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

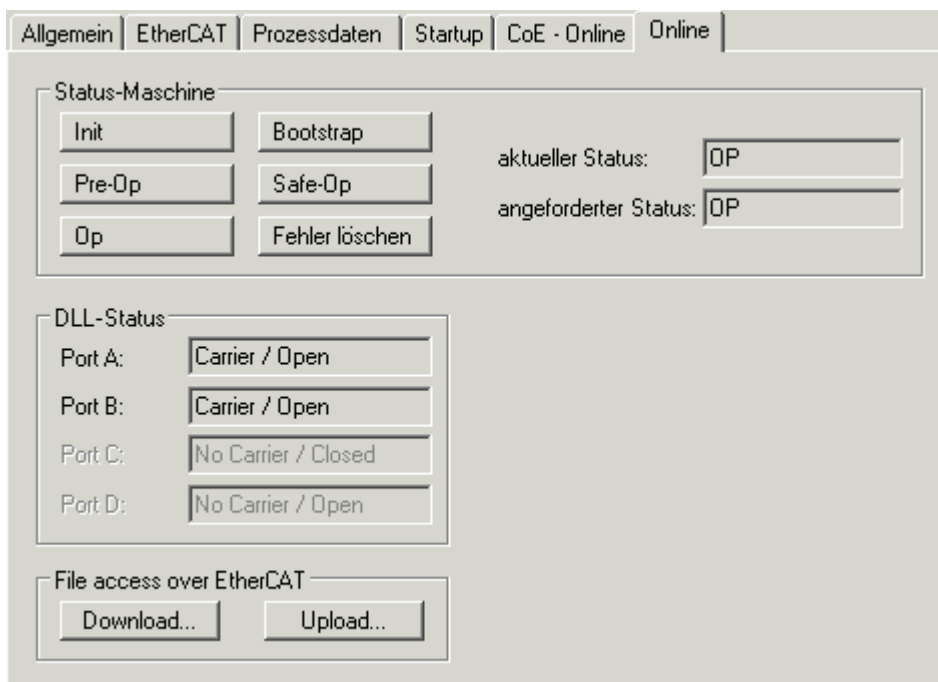


Fig. 124: „Online“ tab

State Machine

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

DLL Status

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

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

File Access over EtherCAT

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

"DC" tab (Distributed Clocks)

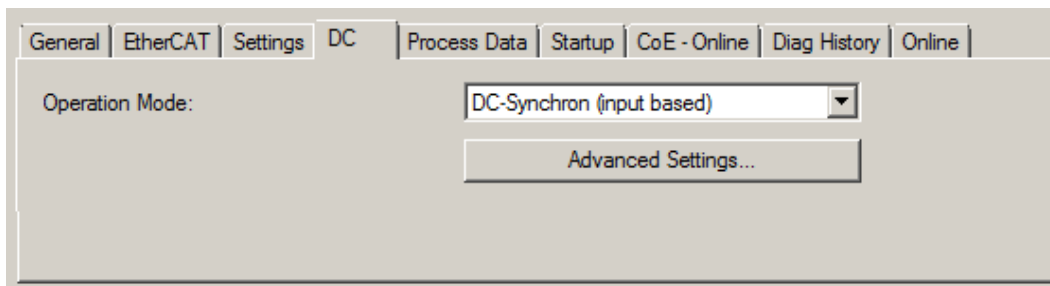


Fig. 125: "DC" tab (Distributed Clocks)

- 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 are specified on <http://infosys.beckhoff.com>:

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

5.2.7.1 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 \[▶ 100\]](#)),
 - 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 \[► 97\]](#) tab.

PDO Configuration

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

5.3 General Notes - EtherCAT Slave Application

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.

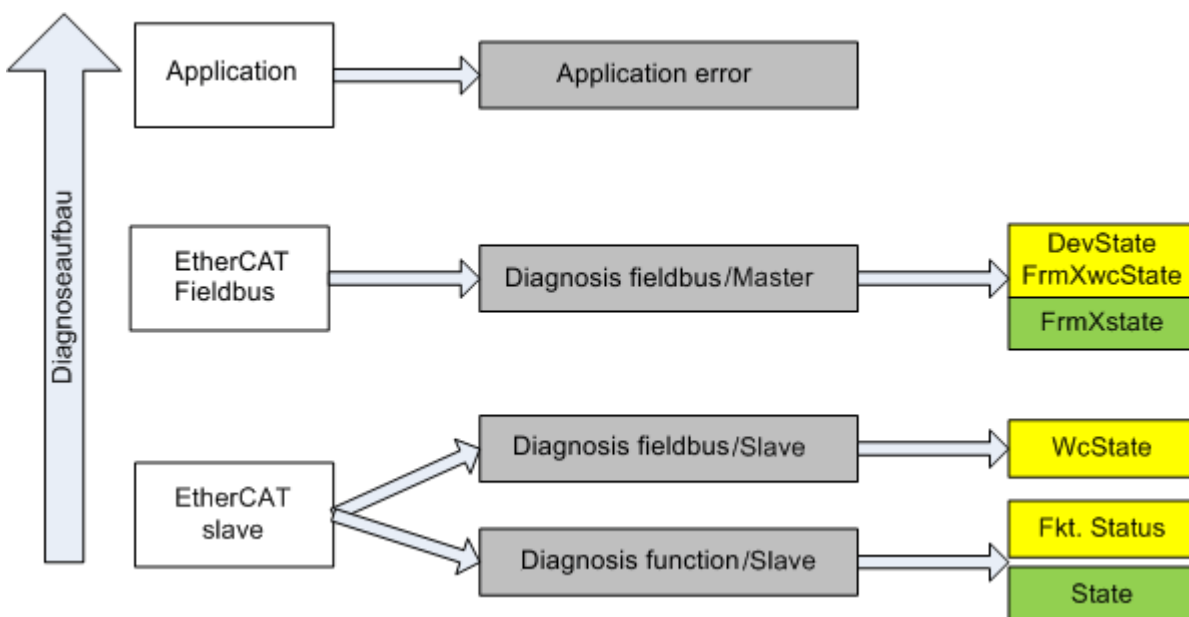


Fig. 126: Selection of the diagnostic information of an EtherCAT Slave

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.

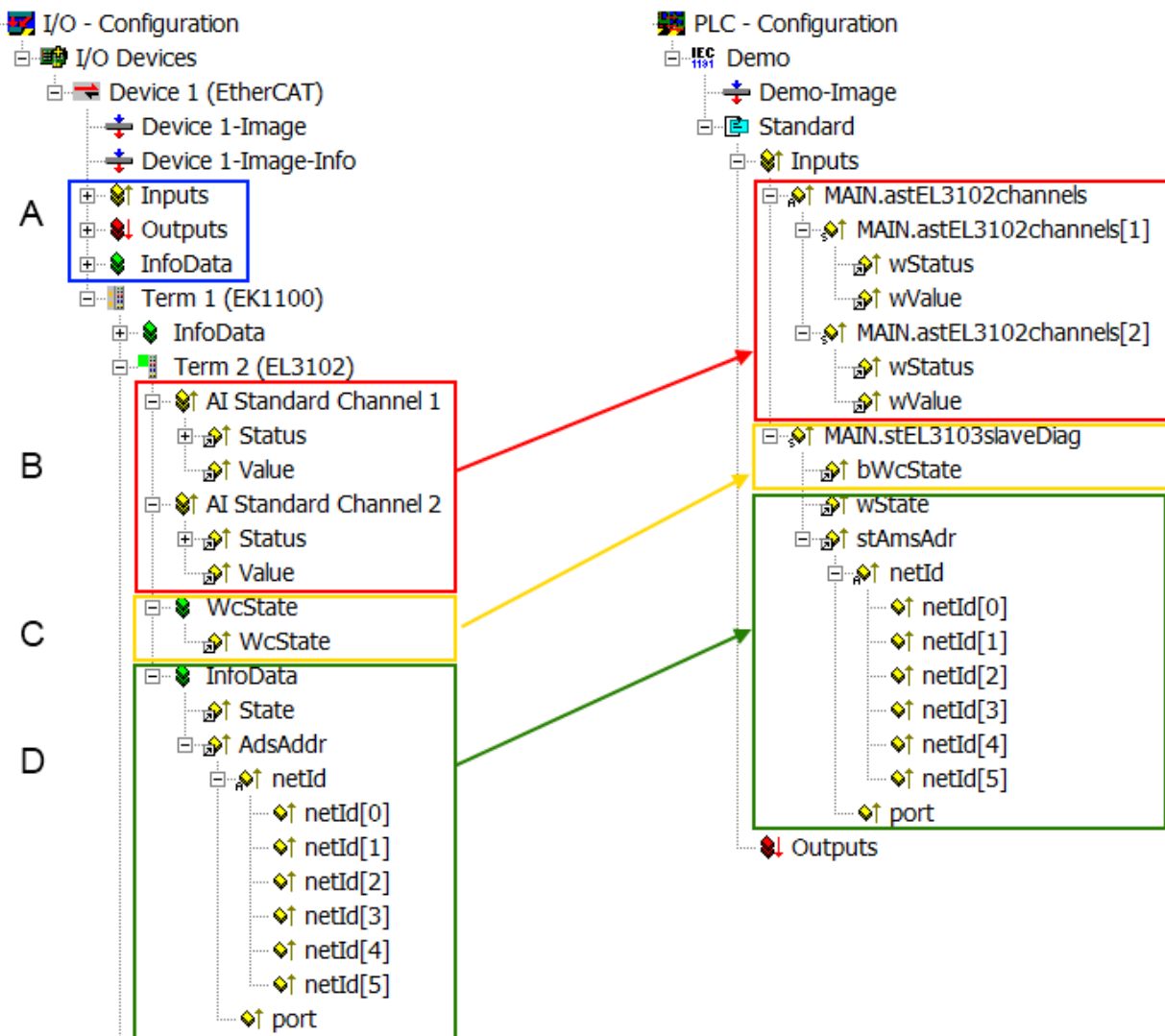


Fig. 127: Basic EtherCAT Slave Diagnosis in the PLC

The following aspects are covered here:

Code	Function	Implementation	Application/evaluation
A	The EtherCAT Master's diagnostic information updated acyclically (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: <ul style="list-style-type: none"> • 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 <ul style="list-style-type: none"> • 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 <ul style="list-style-type: none"> • 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.

NOTE

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".

Index	Name	Flags	Value
6010:0	AI Inputs Ch.2	RO	> 17 <
6401:0	Channels	RO	> 2 <
8000:0	AI Settings Ch.1	RW	> 24 <
8000:01	Enable user scale	RW	FALSE
8000:02	Presentation	RW	Signed (0)
8000:05	Siemens bits	RW	FALSE
8000:06	Enable filter	RW	FALSE
8000:07	Enable limit 1	RW	FALSE
8000:08	Enable limit 2	RW	FALSE
8000:0A	Enable user calibration	RW	FALSE
8000:0B	Enable vendor calibration	RW	TRUE

Fig. 128: EL3102, CoE directory

● EtherCAT System Documentation



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

A few brief extracts:

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

Commissioning aid in the TwinCAT System Manager

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

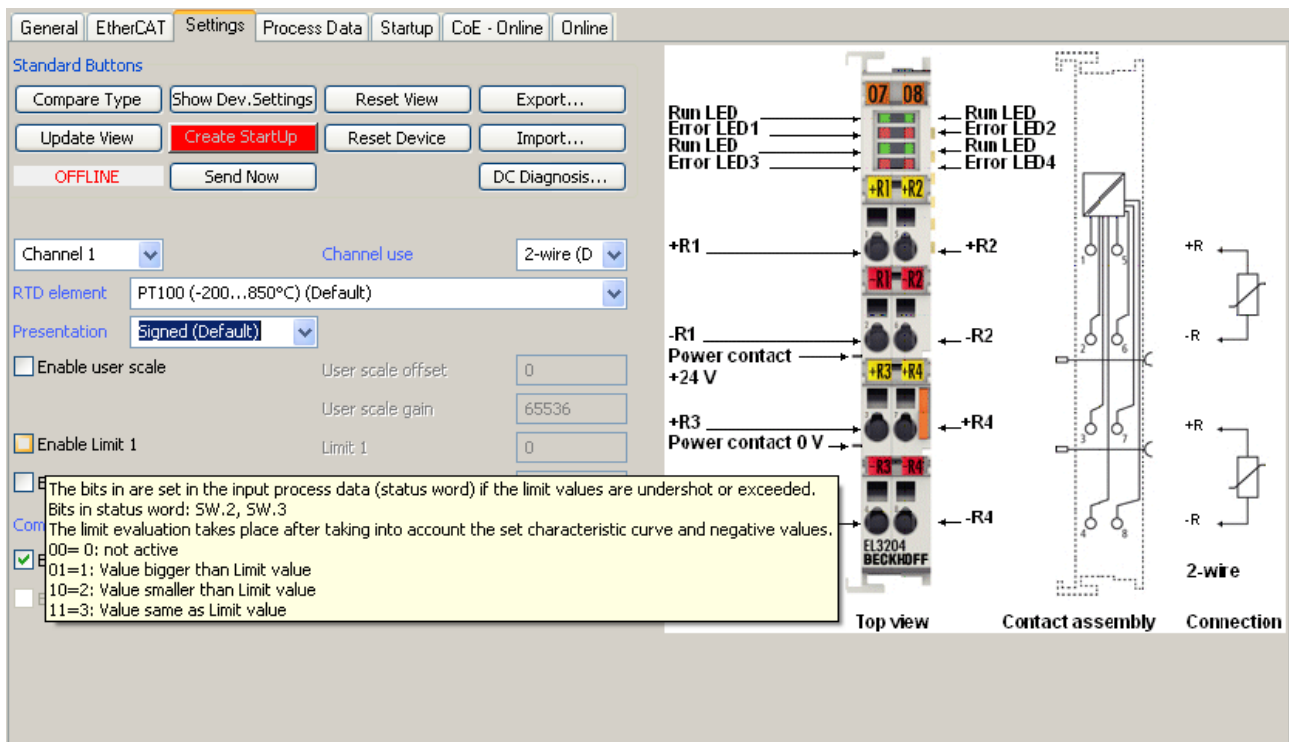


Fig. 129: Example of commissioning aid for a EL3204

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 \[► 21\]](#)" 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.

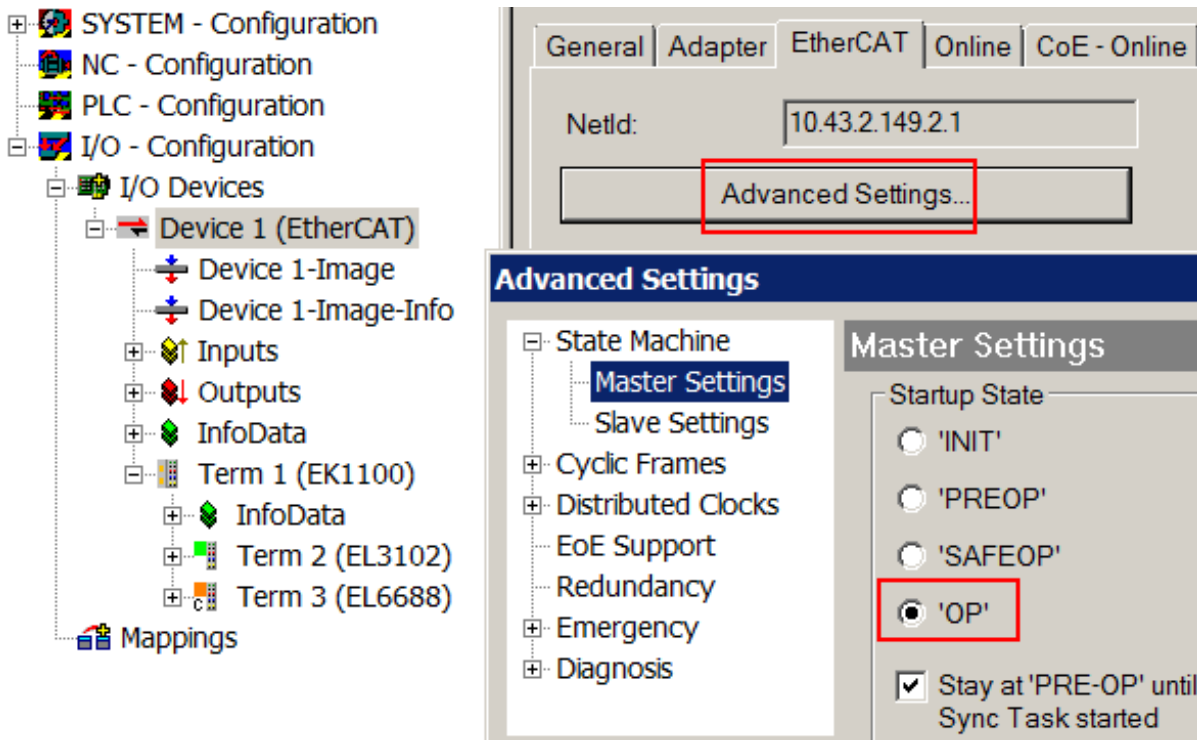


Fig. 130: Default behaviour of the System Manager

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

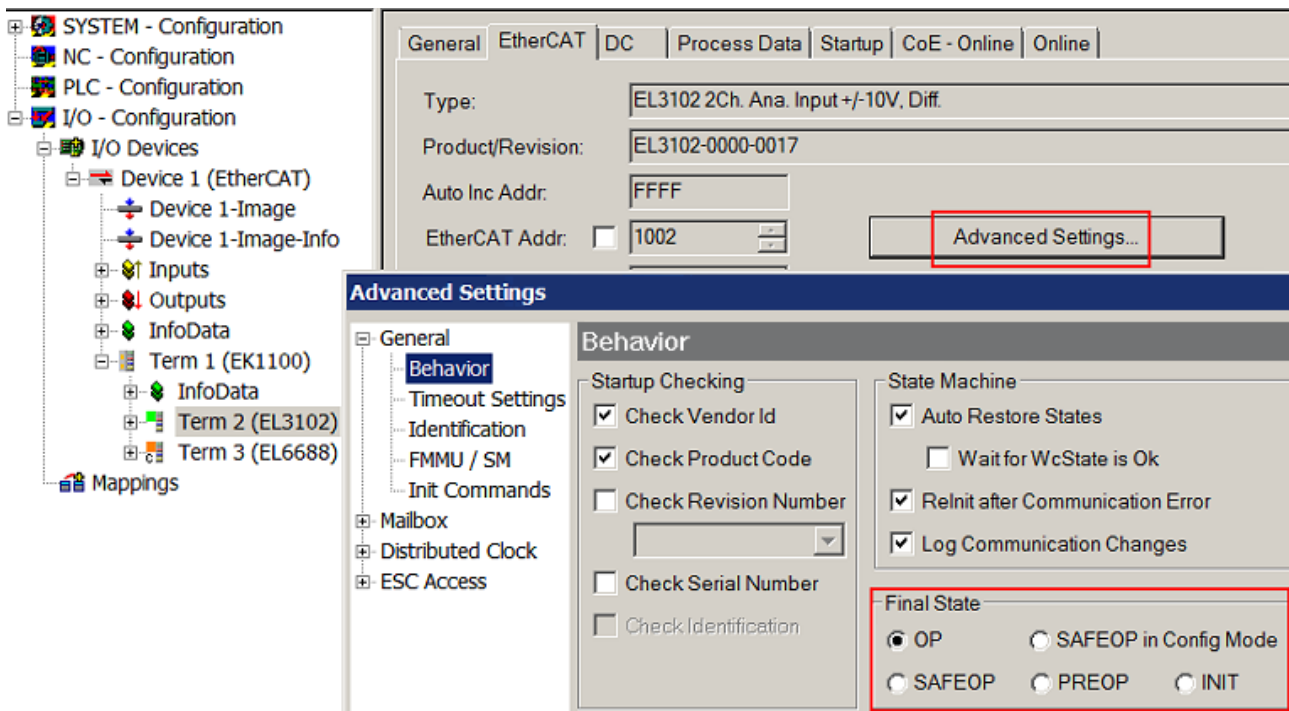


Fig. 131: Default target state in the Slave

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.

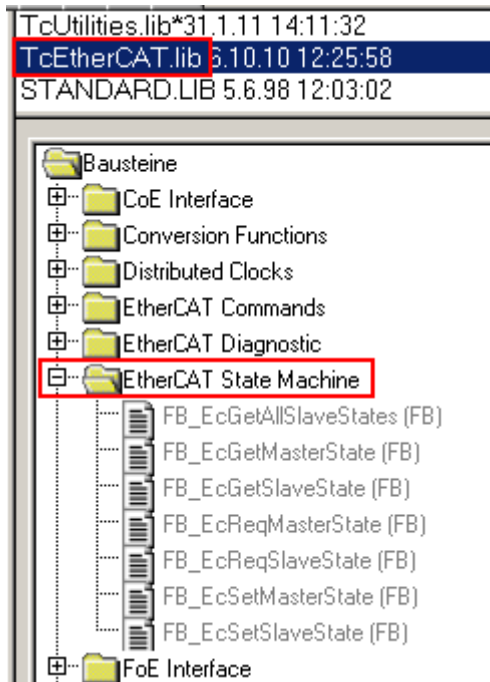


Fig. 132: PLC function blocks

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. 133: 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:

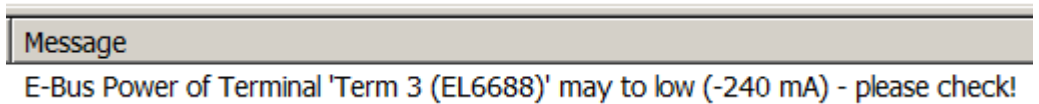


Fig. 134: Warning message for exceeding E-Bus current

NOTE

Caution! Malfunction possible!

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

5.4 IO-Link basics

IO-Link represents a communication system for the connection of intelligent sensors and actuators to an automation system in the IEC 61131-9 standard 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.](#)

● IO-Link specification

i The development of the EL6224 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.

5.4.1 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 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 Fig. *IO-Link point-to-point communication*).

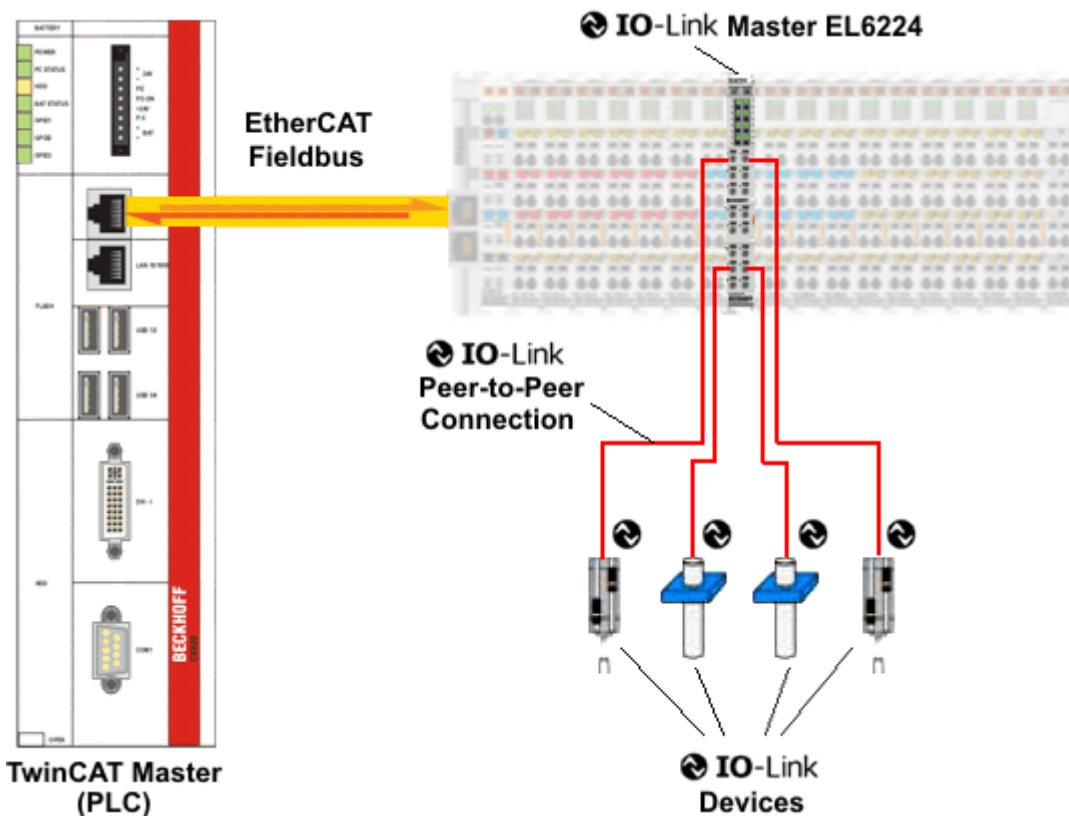


Fig. 135: IO-Link point-to-point communication

5.4.2 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 [▶ 116] the IO-Link port.

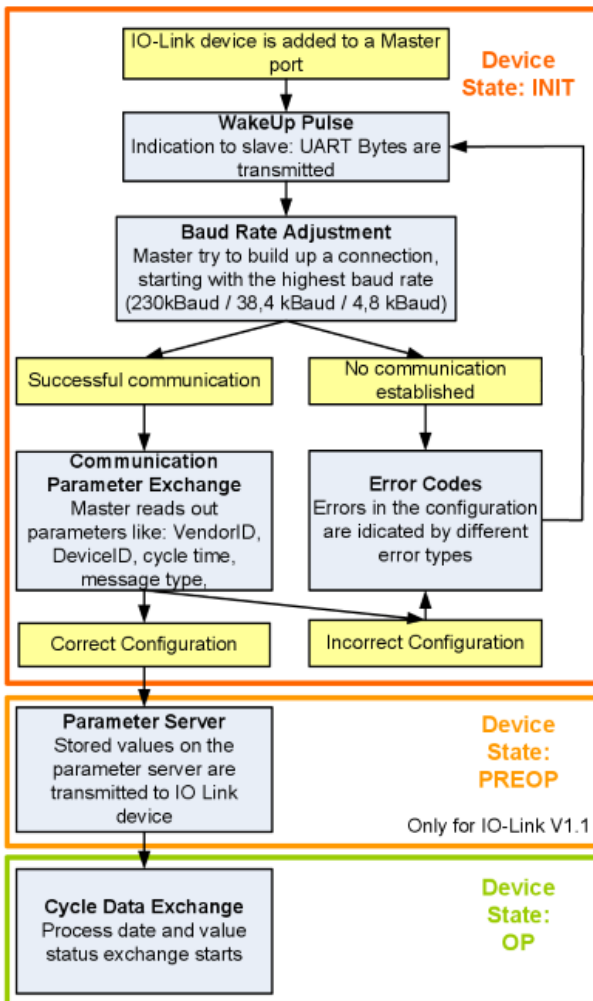


Fig. 136: Establishment of IO Link communication

- 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 slave that UART bytes are to be sent from now on.
From this point on, all data will be interpreted by the IO-Link slave as UART bytes.
- The master runs through all baud rates [▶ 113], 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 slave, 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 [▶ 113] 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 slave.

5.4.3 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](#) [[▶ 116](#)] with TwinCAT and adopted in the System Manager (cf. integration of the IO-Link devices).

5.4.4 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](#) [[▶ 118](#)]. The Beckhoff IO-Link master supports specification V1.1 from FW 07.

- 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 & 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 & 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 via the Master's special event. 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.
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.

5.4.5 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

5.5 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.

The System Manager extension for EL6224 is required for TwinCAT version 2.10, build 1325 to 1330.

Configuration of the IO link master requires a plug-in, which is usually supplied with the TwinCAT installation.

When adding the IO-Link master (see chapter [Commissioning \[▶ 70\]](#)) in the TwinCAT System Manager, an additional tab called "IO-Link" is created (Fig. "IO-Link Devices" tab). If the tab is not displayed, the associated System Manager extension is missing. The extension is available for download from the Beckhoff website.

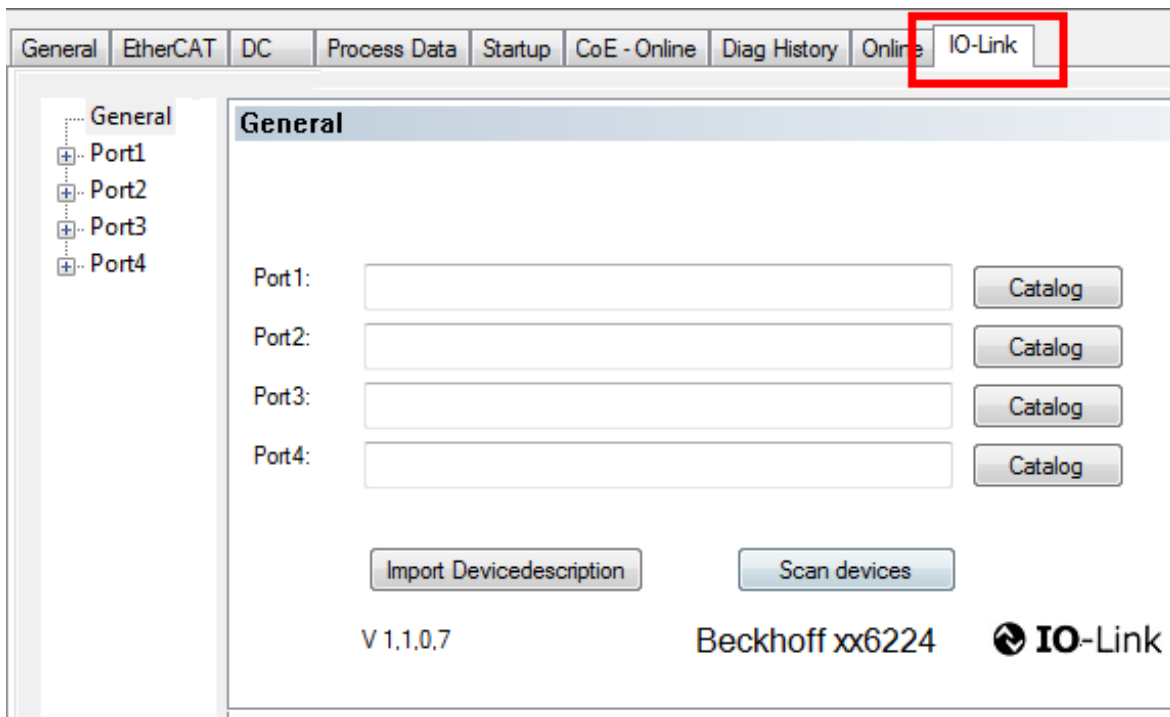


Fig. 137: "IO Link" tab

5.5.1 Integrating IO link devices

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

1. [Importing the IODD file \[▶ 115\]](#) (offline & online)
2. [Automatic scanning of the IO-Link ports \[▶ 116\]](#) (online)
3. [Manual insertion via catalog \[▶ 116\]](#) (offline & online)

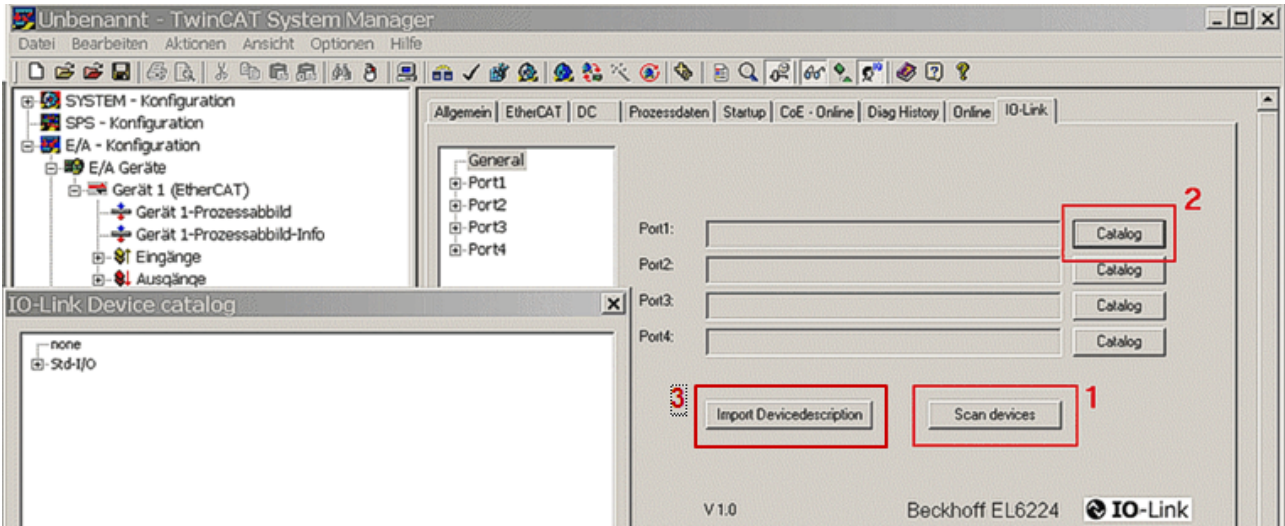


Fig. 138: "IO-Link" tab, integration of the devices

The [integration of the IODD \[▶ 115\]](#) 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.

If the IODD is not available, the IO-Link device should be integrated online by [scanning \[▶ 116\]](#). Manual integration of the IO-Link devices via the [catalog \[▶ 116\]](#) should only be done if the IODD from the vendor and the IO-Link are not available when the project is created.

5.5.1.1 1. Importing the device description IODD ("Import Device description" button)

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:

- Press the "Import Devicedescription" button on the "IO-Link" tab
- Select and open the .xml file for the desired sensor
- 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

● No manual copying of the XML files

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

- Online configuration: once the IO-Link device has been connected, it is automatically detected and created with the corresponding parameters by pressing the "Scan devices" button. 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 IO-Link device catalog is called with the "Catalog" button; the device descriptions that have already been imported are listed here, arranged in a tree structure according to the vendor. The selection of the IO-Link device must be confirmed with "OK".
- Restart the EtherCAT system or reload the configuration in Config mode (F4).
- 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).

5.5.1.2 2. Automatic scanning of the IO link ports

When automatically scanning the IO-Link port, the steps "WakeUp Pulse", "Setting the baud rate", "Reading the communication parameters" and, if applicable, "Parameter server" and "Cyclic data exchange" are performed; cf. "[Establishment of the IO-Link communication](#)" [► 112]. The corresponding IO-Link device must be connected to the IO-Link port for this.

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

5.5.1.3 3. Manual Insertion via catalog ("Catalog" button)

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.

The configuration can then be carried out manually. By saving the project, the settings for the individual ports are saved. The devices created are not saved in the *IO-Link device catalog*.

The procedure to manual insert the IO-Link devices via the catalog is as follows:

- Press the "Catalog" button on the "IO-Link" tab, Fig. "*IO-Link, integration of the devices (1); IO-Link Device Catalog*" is displayed.
- If the corresponding IODD of the IO-Link device is already available, the corresponding device can be selected here, sorted by vendor.
- If no IODD is available, the device can be manually added. These data are not saved in the catalog and must be manually entered for each port.
- The basic communication parameters can be created in the catalog dialogue via the "Create Device" menu. The mandatory fields here are: Vendor ID, Device ID and process data length, see Fig. *Creating IO-Link devices*. The values VendorID and DeviceID can be entered both in hexadecimal notation (input format: 0xnxxx) and as decimal numbers (nnnn).
- If the IO-Link device version is 1.1, then the parameter server is activated by the selection of the field.
- Restart the EtherCAT system or reload the configuration in Config mode (F4)

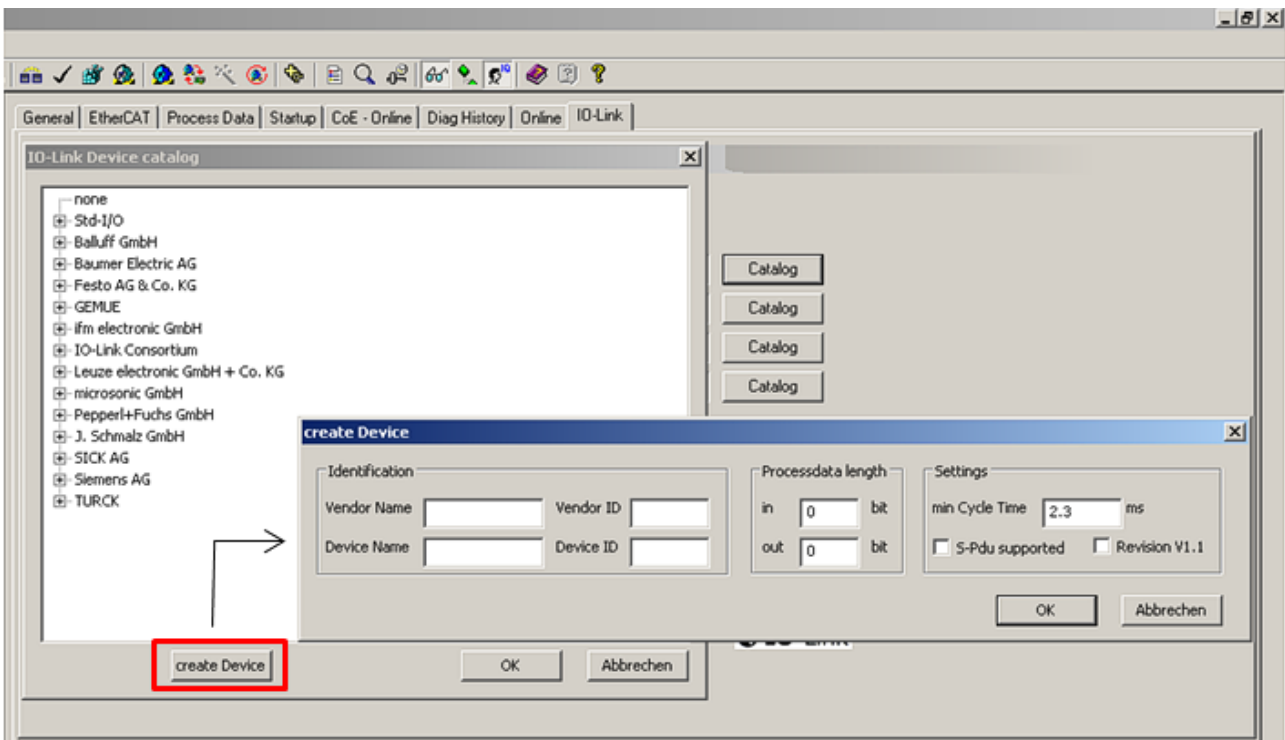


Fig. 139: Creating IO link devices



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.

5.5.2 Removal of the IO-Link devices

If an IO-Link device that is already configured is to be removed, this can be done via the selection "none" in the *IO-Link Device Catalog*. After restarting the EtherCAT system or reloading the configuration in Config mode (F4), the already created process data are removed.

5.5.3 Configuration of the IO-Link ports as digital inputs/outputs

If the IO-Link port is to be used as a simple digital input or output, the selection "Std-I/O" -> "dig in" or "dig out" can be made in the *IO-Link Device Catalog*. After restarting the EtherCAT system or reloading the configuration in Config mode (F4), the corresponding process data are created.

5.6 Parameterization of the IO-Link devices

The IO-Link interface offers the particular advantage that there is now an option to parameterize simple sensors and actuators via the IO-Link interface using the PLC. The device can thus be adapted to the special customer application.

The device description (IODD) is required for the configuration or parameterization of the IO-Link devices, because it contains the possible parameters and setting values for the devices. The correct integration of the IODD is explained in the chapter [Configuration of the IO-Link master](#) [▶ 115].

The "[Settings](#) [▶ 118]" and "[Parameters](#) [▶ 119]" tabs are available after importing the IODD and integrating the IO-Link devices.

IO Link device settings

The basic settings for the devices can be found for each port under "General" -> "Port X" -> "Settings".

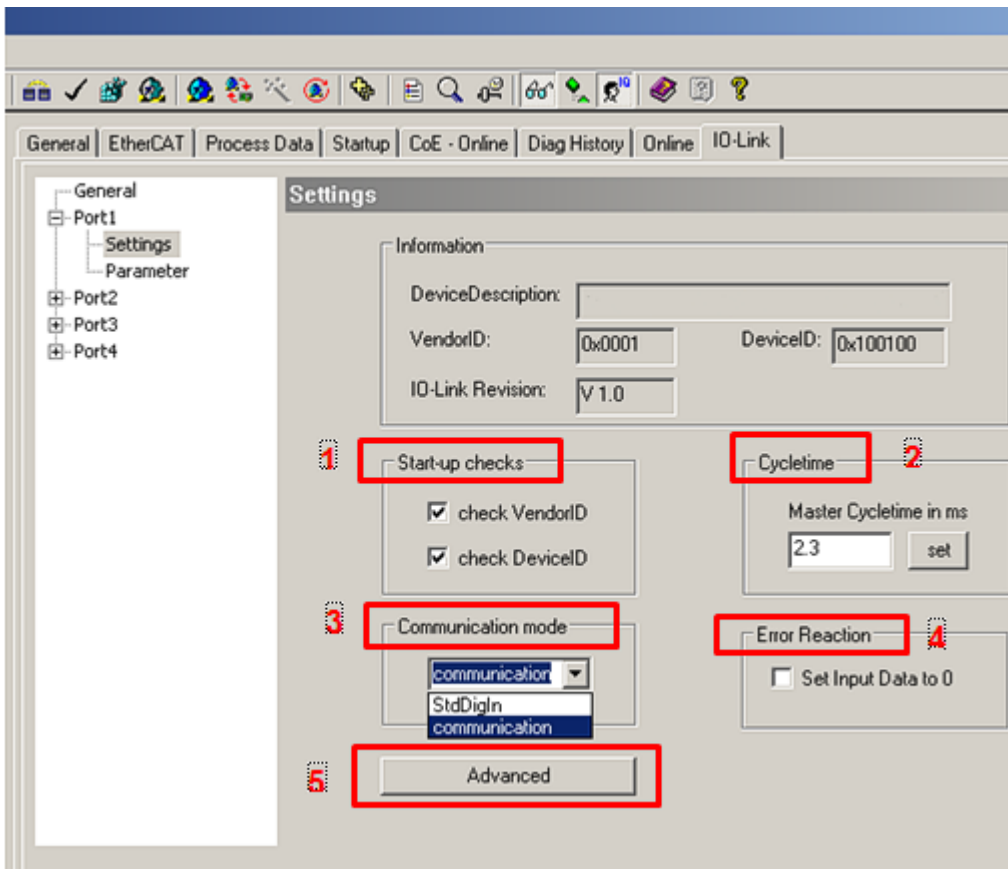


Fig. 140: IO link device settings

The following settings can be made in the IO-Link device settings (Fig. *IO-Link device settings*):

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 (V 1.0 or V 1.1) of the IO-Link devices are displayed. If the device is an IO-Link device V1.1, then the [parameter server](#) [▶ 113] function is supported.

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 [► 117]
- Baud rate is set automatically on the basis of the WakeUp telegram

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: "Fault"

6. Button "Advanced"

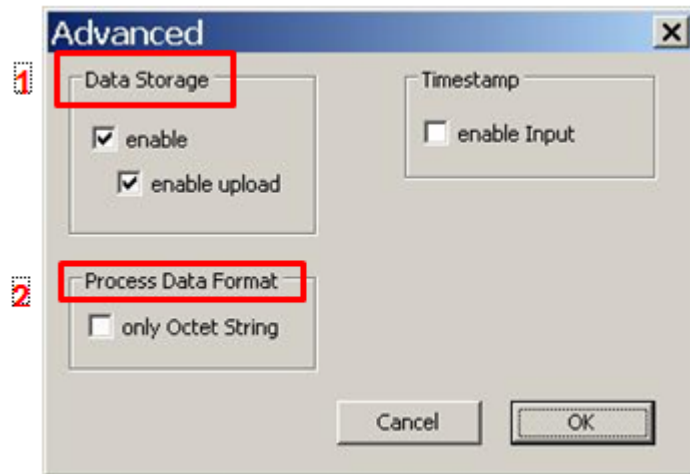


Fig. 141: Advanced Settings

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)

Process Data Format

- Adaptation of the process data format
- "only Octet String" field selected
 -> complex data types (process data) are created as octet strings
 -> Advantage: simple further processing in the PLC

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 IO-Link device can initially be seen. The parameters stored in the IO-Link device are read, loaded and stored in the parameter server of the master using the "read", "write" and "store" buttons respectively.

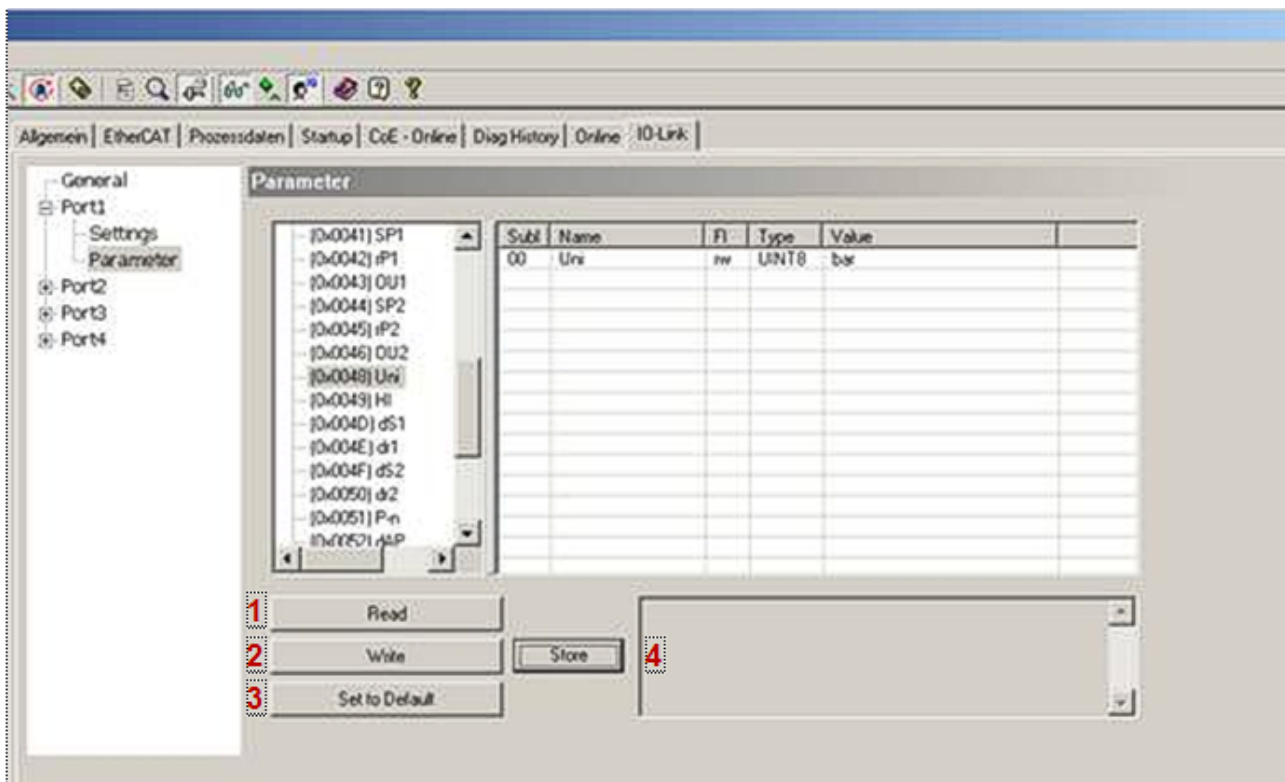


Fig. 142: Reading/writing parameters

Read / Write: Reading/writing of parameters

1) Reading parameters: (Read-Button)

- The default values from the IODD file are always preset.
- The parameter values stored in the IO-Link device are only read when the "Read" button is pressed. However, only the selected object is read.

2) Writing parameters: (Write-Button)

- The default values from the IODD file are always preset.
- Entry / selection of the desired value under "Value".
- The values are loaded into the IO-Link device using the "Write" button. Only the selected object is loaded into the IO-Link device. Each change must be confirmed with the "Write" button.

3) Set to Default-Button

- The parameter values are set to the default settings by selecting the desired object and pressing the "Set to Default" button.

4) Store-Button

The parameter server function is supported with IO-Link V1.1. This is particularly useful for the simple and safe exchange of IO-Link devices. The parameter server is not supported if an IO-Link device is parameterized to V1.0 – the "Store" button is greyed out.

- On pressing the "Store" button, all settings/parameters stored in the sensor are loaded into the parameter server.
- When a sensor is exchanged, the settings are transferred from the parameter server directly to the new sensor.
- For V1.1 sensors: data are explicitly loaded into the parameter server and saved.
- This function is not supported in the case of V1.0 sensors.

5.7 Access to IO-Link data

5.7.1 IO-Link system communication

The Beckhoff EL6224 IO-Link master terminal is divided into two services. On the one hand the terminal 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 Fig. *Illustration of the system communication of an EtherCAT master.*

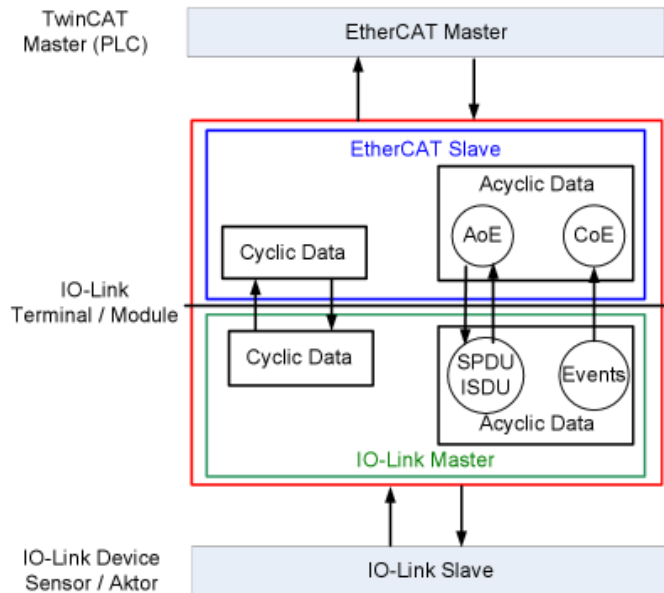


Fig. 143: Illustration of the system communication of an EtherCAT master

In principle cyclic and acyclic data are exchanged. The cyclic process data can be accessed via the PDOs [[▶ 122](#)], the acyclic data via AoE [[▶ 123](#)]. The events are additionally displayed under Diag_History [[▶ 126](#)] in the System Manager.

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

5.7.2 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 (F4); see [Configuration of the IO-Link master](#) [▶ 114].

The screenshot shows the configuration interface for the EL6224 process data allocation. The 'Process Data' tab is selected, and the 'Sync Manager' and 'PDO List' sections are visible. The 'PDO Assignment (0x1C13)' section shows checkboxes for 0x1A05 and 0x1A04. The 'PDO Content (0x1A05)' table shows details for index 0xF101:0D.

SM	Size	Type	Flags
0	256	MbxOut	
1	256	MbxIn	
2	0	Outputs	
3	6	Inputs	

Index	Size	Name	Flags	SM	SU
0x1A05	2.0	DeviceState Inputs Device	F	3	0
0x1A04	4.0	DeviceState Inputs	F	3	0

Index	Size	Offs	Name	Type	De
---	1.0	0.0	---		
---	0.4	1.0	---		
0xF101:0D	0.1	1.4	Device Diag	BOOL	
---	0.2	1.5	---		

Fig. 144: Illustration of the EL6224 process data allocation, SM3 inputs

SM3, PDO Assignment 0x1C13			
Index	Size (byte.bit)	Name	PDO content
0x1A05	2.0	DeviceState Inputs Device	Index 0xF101:0D - Device Diag Index 0xF101:10 - Device State
0x1A04	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

● Process data representation

i 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 slave side) is reported via a status bit
0xF101:10	0.1	Device State	Interruption of communication with one of the slaves is reported via a status bit
0xF100:01	1.0	State Ch.1	0x_0 = Port disabled
0xF100:02	1.0	State Ch.2	0x_1 = Port in std dig in
0xF100:03	1.0	State Ch.3	0x_2 = Port in std dig out
0xF100:04	1.0	State Ch.4	0x_3 = Port in communication OP
			0x_4 = Port in communication COMSTOP / dig in Bit (only in std. IO Mode)
			0x_8 = Process Data Invalid Bit
			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 (Error in IO-Link State PreOP)

5.7.3 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.

5.7.4 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 \[▶ 141\]: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)

5.7.5 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 terminal EL6224 via AoE (ADS over EtherCAT), from where the command is relayed to the IO link master section and therefore the data channel.

AoE NetID

The EL6224 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* -> „EtherCAT“-> „Advanced Settings“ -> „Mailbox“ -> “AoE“

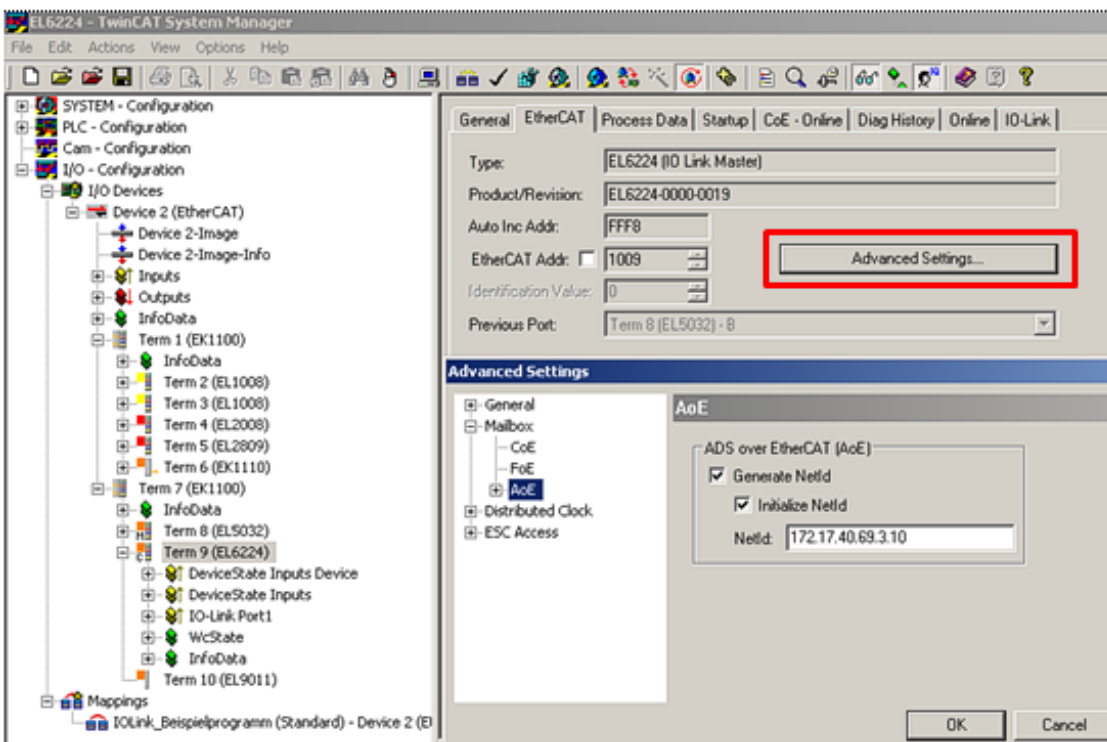


Fig. 145: AoE-NetID allocation

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

5.7.6 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 \[▶ 122\]](#) bit. Further information on the events can be read via the CoE directory or the DiagHistory tab.

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. 146: 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 \[▶ 122\]](#). 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.

5.8 TwinSAFE SC

5.8.1 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](#).

5.8.2 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.

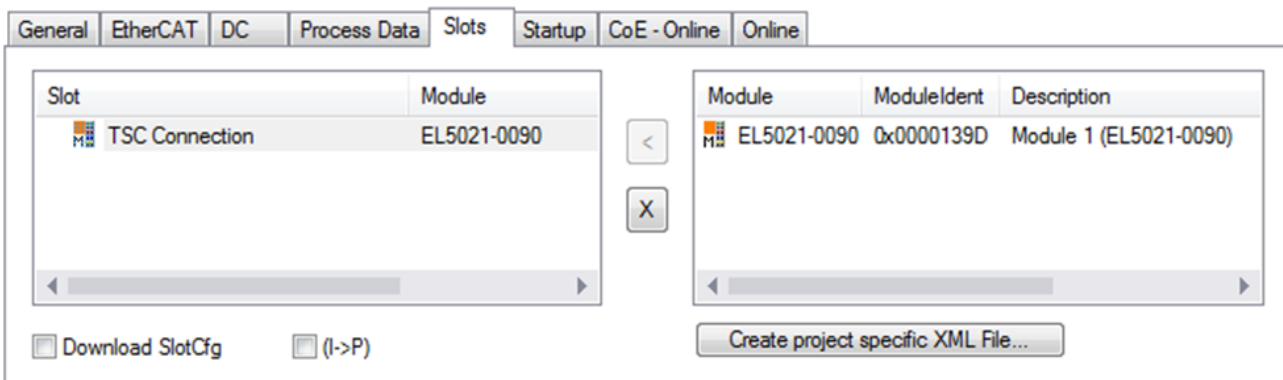


Fig. 147: Adding the TwinSAFE SC process data under the component, e.g. EL5021-0090

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

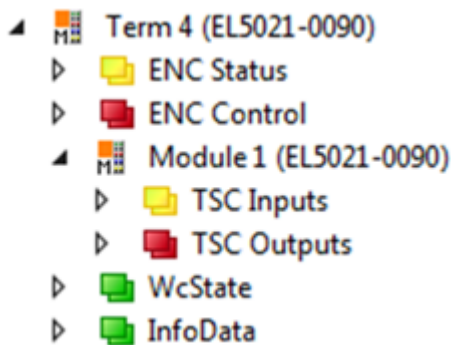


Fig. 148: TwinSAFE SC component process data, example EL5021-0090

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

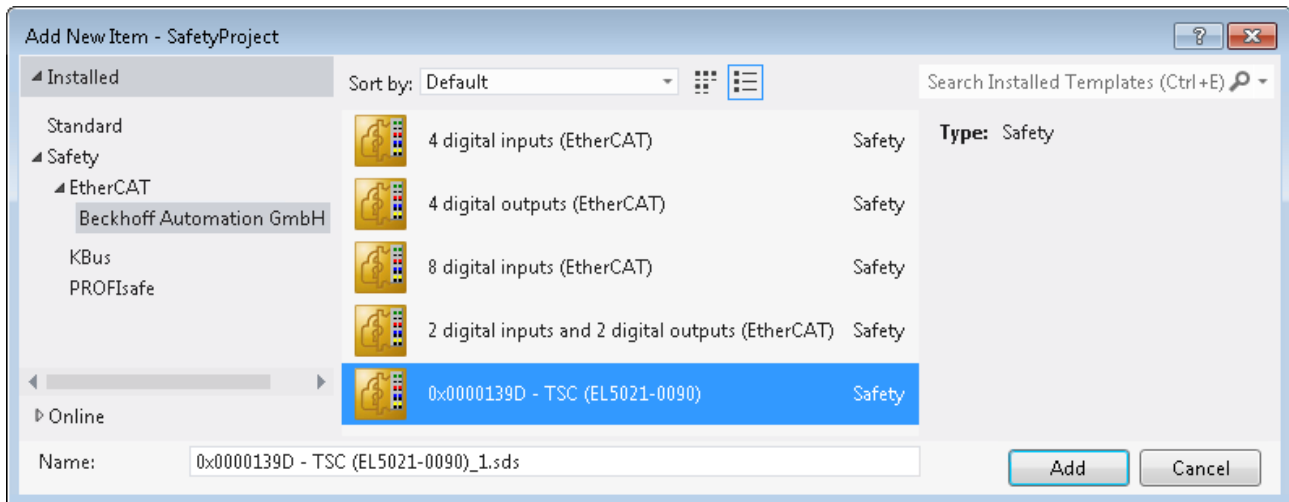



Fig. 149: Adding a TwinSAFE SC connection

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.

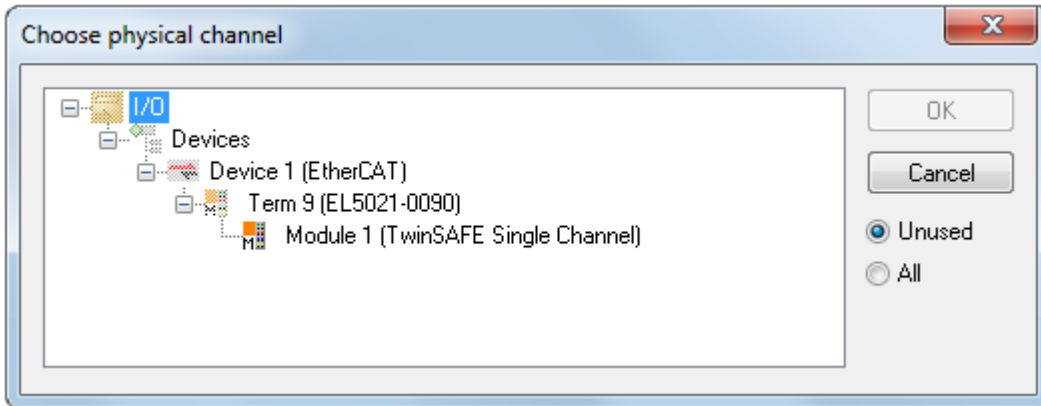


Fig. 150: Creating a link to TwinSAFE SC terminal

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

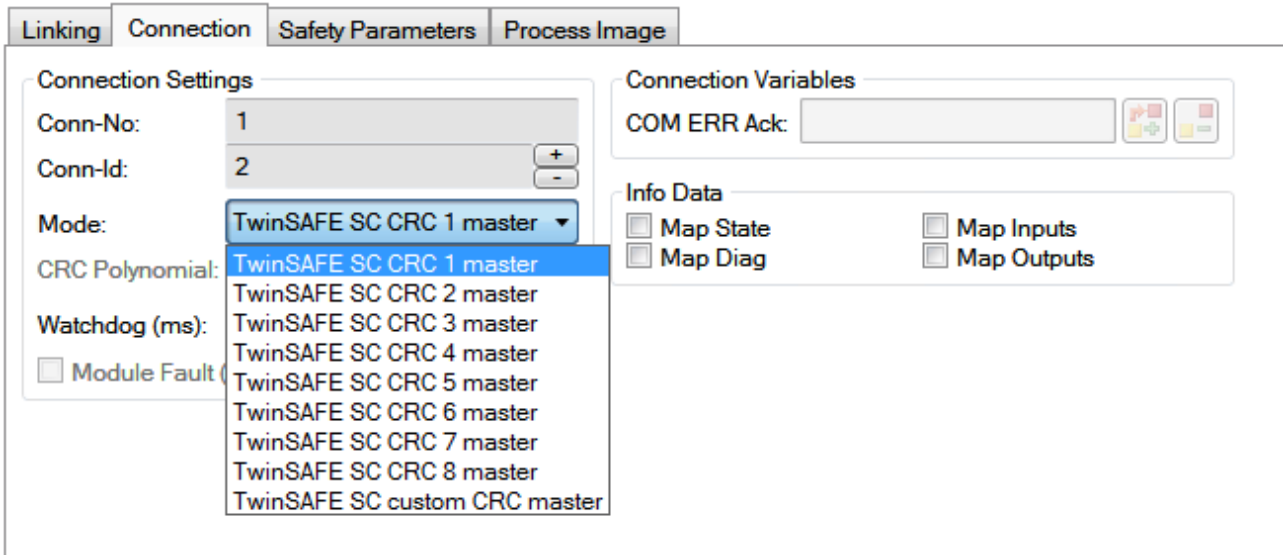


Fig. 151: Selecting a free CRC

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.

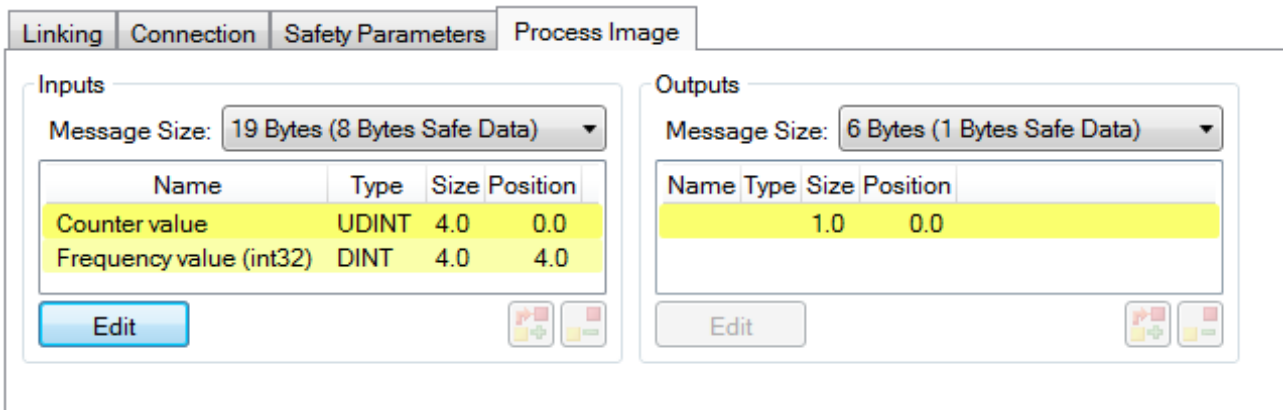


Fig. 152: Selecting the process data size and the process data

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)*.

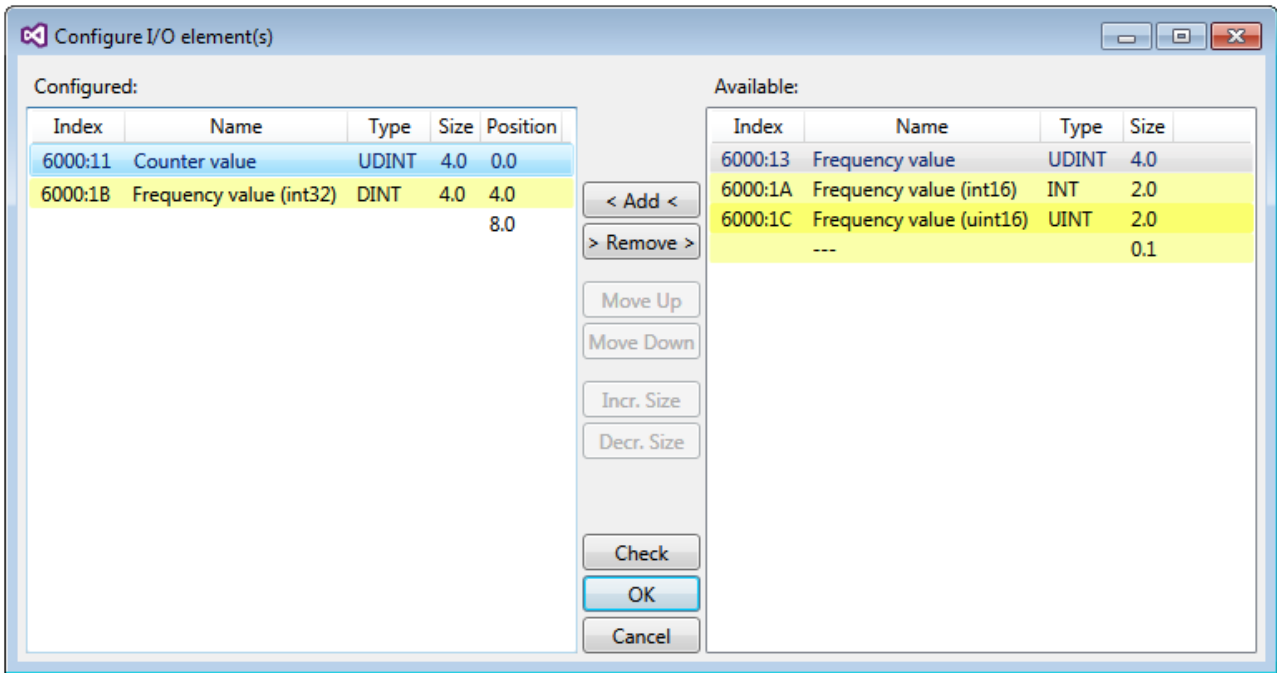


Fig. 153: Selection of the process data

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. 154: 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

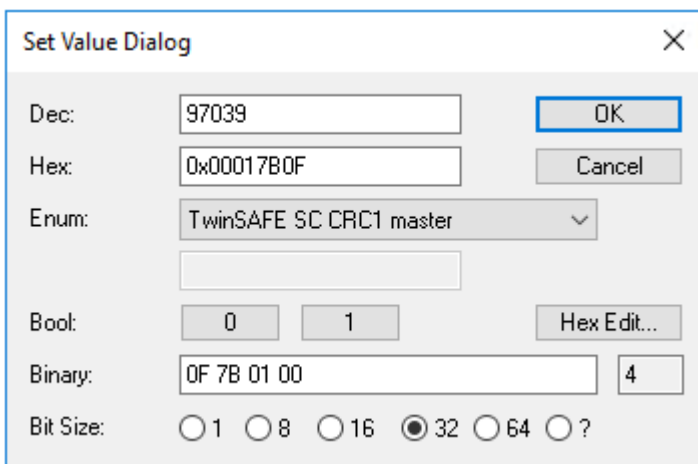


Fig. 155: Entering the safety address and the CRC

i TwinSAFE SC connections

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

5.9 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 slave 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.

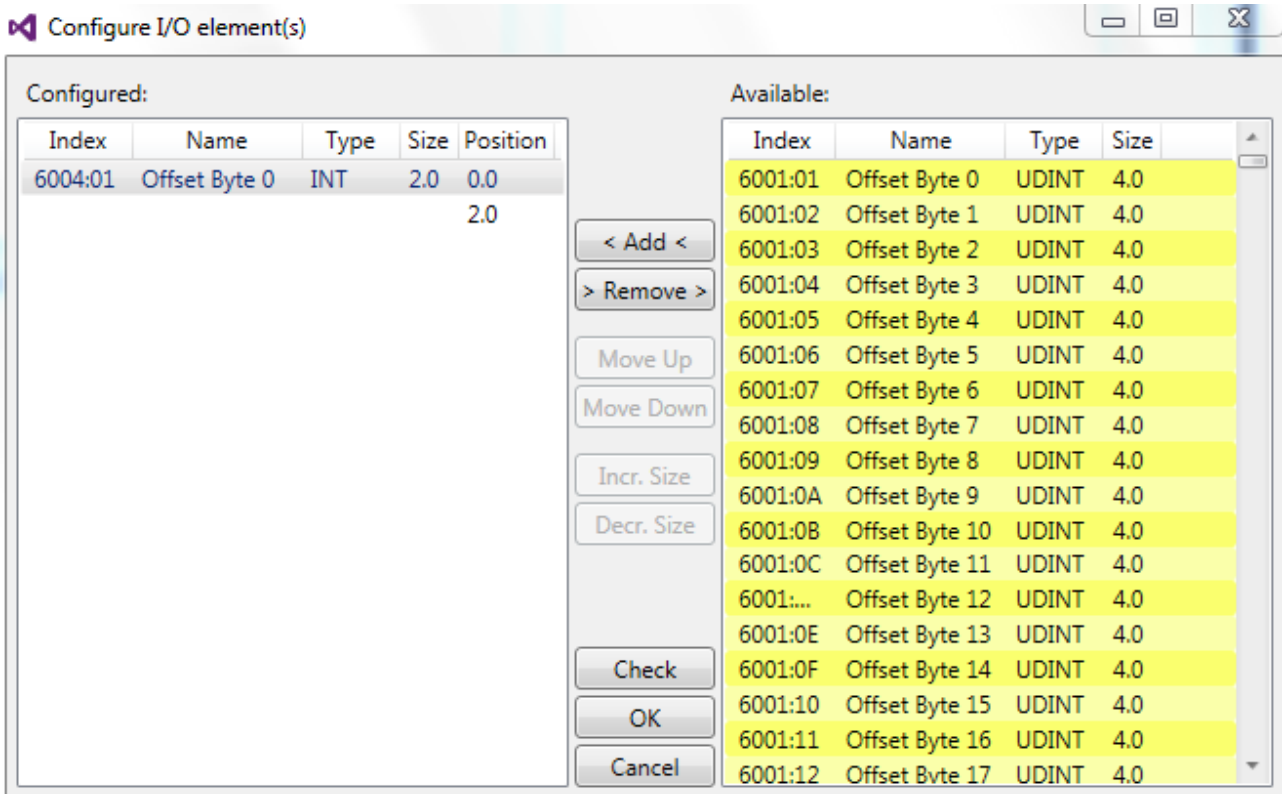


Fig. 156: EL6224-0090, process data image (default)

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 \(EL6224-0090\)](#) [[▶ 142](#)].

5.10 Object description and parameterization

● EtherCAT XML Device Description



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.

● Parameterization via the CoE list (CAN over EtherCAT)



The EtherCAT device is parameterized via the [CoE-Online tab](#) [[▶ 98](#)] (double-click on the respective object) or via the [Process Data tab](#) [[▶ 95](#)] (allocation of PDOs). Please note the following general [CoE notes](#) [[▶ 23](#)] 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” for resetting changes

Introduction

The CoE overview contains objects for different intended applications:

- Objects required for parameterization during [commissioning](#) [[▶ 132](#)]
- Objects for indicating [internal settings](#) [[▶ 134](#)] (may be fixed)
- [Profile-specific objects](#) [[▶ 139](#)] 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.

5.10.1 Objects for commissioning

Index 1011 Restore default parameters

Index (hex)	Name	Meaning	Data type	Flags	Default
1011:0	Restore default parameters [▶ 160]	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})

5.10.2 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 _{dez})
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: <ul style="list-style-type: none"> 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): <ul style="list-style-type: none"> 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: <ul style="list-style-type: none"> 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 [▶ 138]) 	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	<ul style="list-style-type: none"> 0: Measurement of the local cycle time is stopped 1: Measurement of the local cycle time is started <p>The entries 0x1C32:03 [▶ 138], 0x1C32:05 [▶ 138], 0x1C32:06 [▶ 138], 0x1C32:09 [▶ 138], 0x1C33:03 [▶ 139], 0x1C33:06 [▶ 138], 0x1C33:09 [▶ 139] are updated with the maximum measured values. For a subsequent measurement the measured values are reset</p>	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: <ul style="list-style-type: none"> • 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 [138]	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: <ul style="list-style-type: none"> • 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 [138] or 0x1C33:08 [139]) 	UINT16	RO	0xC007 (49159 _{dec})
1C33:05	Minimum cycle time	as 0x1C32:05 [138]	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 [138]	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 [138]	UINT16	RO	0x0000 (0 _{dec})
1C33:0C	Cycle exceeded counter	as 0x1C32:12 [138]	UINT16	RO	0x0000 (0 _{dec})
1C33:0D	Shift too short counter	as 0x1C32:13 [138]	UINT16	RO	0x0000 (0 _{dec})
1C33:20	Sync error	as 0x1C32:32 [138]	BOOLEAN	RO	0x00 (0 _{dec})

5.10.3 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 ≤ n ≤ 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 F100 Diagnosis Status data

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 0x_0 = Port disabled 0x_1 = Port in std dig in	UINT8	RO	0x00 (0 _{dec})
F100:02	State Ch2	Status byte Ch. 2 0x_2 = Port in std dig out 0x_3 = Port in communication OP	UINT8	RO	0x00 (0 _{dec})
F100:03	State Ch3	Status byte Ch. 3 0x_4 = Port in communication COM-STOP / dig in Bit (only in std. IO Mode)	UINT8	RO	0x00 (0 _{dec})
F100:04	State Ch4	Status byte Ch. 4 0x_8 = Process Data Invalid Bit 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	UINT8	RO	0x00 (0 _{dec})

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 slave 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 _{dec})
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})

5.10.4 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 _{dec})
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 60c2 Word Inputs Ch. c+1 (c = 0, 1, 2, 3)

Index (hex)	Name	Meaning	Data type	Flags	Default
60c2:0	Word Inputs Ch. c+1	Max. subindex	UINT8	RO	0x1D (29 _{dec})
60c2:01	Offset Byte 0	Offset Byte 0	()	RO	()
60c2:02	Offset Byte 1	Offset Byte 1	()	RO	()
...
60c2:1D	Offset Byte 28	Offset Byte 28	()	RO	()

Index 60d2 unsigned Word Inputs Ch. d+1 (d = 0, 1, 2, 3)

Index (hex)	Name	Meaning	Data type	Flags	Default
60d2:0	Unsigned Word Inputs Ch. d+1	Max. subindex	UINT8	RO	0x1D (29 _{dec})
60d2:01	Offset Byte 0	Offset Byte 0	()	RO	()
60d2:02	Offset Byte 1	Offset Byte 1	()	RO	()
...
60d2:1D	Offset Byte 28	Offset Byte 28	()	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})

6 Error handling and diagnostics

6.1 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_VALGTLIM	This ErrorType shall be used for a write service to a parameter above its specified value range.
Parameter value below limit	0x80	0x32	PAR_VALLTLIM	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_LENVERRUN	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 [▶ 122]" (0xF101:0D) that at least one event has occurred in the "Diag History".

"Device State [▶ 122]" 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 [▶ 122] in the System Manager).

Nominal/actual comparison of the parameter objects

The indices 0x90x0 (diagnosis) can be referred to for validation of the configuration indices 0x80x0 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 0xA0x0:02 [▶ 141] 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.

7 Appendix

7.1 EtherCAT AL Status Codes

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

7.2 UL notice

	<p>Application Beckhoff EtherCAT modules are intended for use with Beckhoff's UL Listed EtherCAT System only.</p>
	<p>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).</p>
	<p>For devices with Ethernet connectors Not for connection to telecommunication circuits.</p>

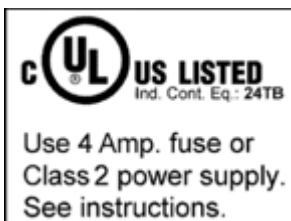
Basic principles

Two UL certificates are met in the Beckhoff EtherCAT product range, depending upon the components:

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



2. UL certification according to UL508 with limited power consumption. The current consumed by the device is limited to a max. possible current consumption of 4 A. Devices with this kind of certification are marked by this sign:



Almost all current EtherCAT products (as at 2010/05) are UL certified without restrictions.

Application

If terminals certified *with restrictions* are used, then the current consumption at 24 V_{DC} must be limited accordingly by means of supply

- from an isolated source protected by a fuse of max. 4 A (according to UL248) or
- from a voltage supply complying with *NEC class 2*.
A voltage source complying with *NEC class 2* may not be connected in series or parallel with another *NEC class 2* compliant voltage supply!

These requirements apply to the supply of all EtherCAT bus couplers, power adaptor terminals, Bus Terminals and their power contacts.

7.3 Firmware Update EL/ES/EM/ELM/EPxxxx

This section describes the device update for Beckhoff EtherCAT slaves from the EL/ES, ELM, EM, EK and EP series. A firmware update should only be carried out after consultation with Beckhoff support.

Storage locations

An EtherCAT slave stores operating data in up to 3 locations:

- 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.
- In addition, each EtherCAT slave has a memory chip, a so-called **ESI-EEPROM**, for storing its own device description (ESI: EtherCAT Slave Information). On power-up this description is loaded and the EtherCAT communication is set up accordingly. The device description is available from the download area of the Beckhoff website at (<https://www.beckhoff.de>). All ESI files are accessible there as zip files.

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 3 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

NOTE

Risk of damage to the device!

Note the following when downloading new device files

- Firmware downloads to an EtherCAT device must not be interrupted
- Flawless EtherCAT communication must be ensured. CRC errors or LostFrames must be avoided.
- 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.

7.3.1 Device description ESI file/XML

NOTE

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:

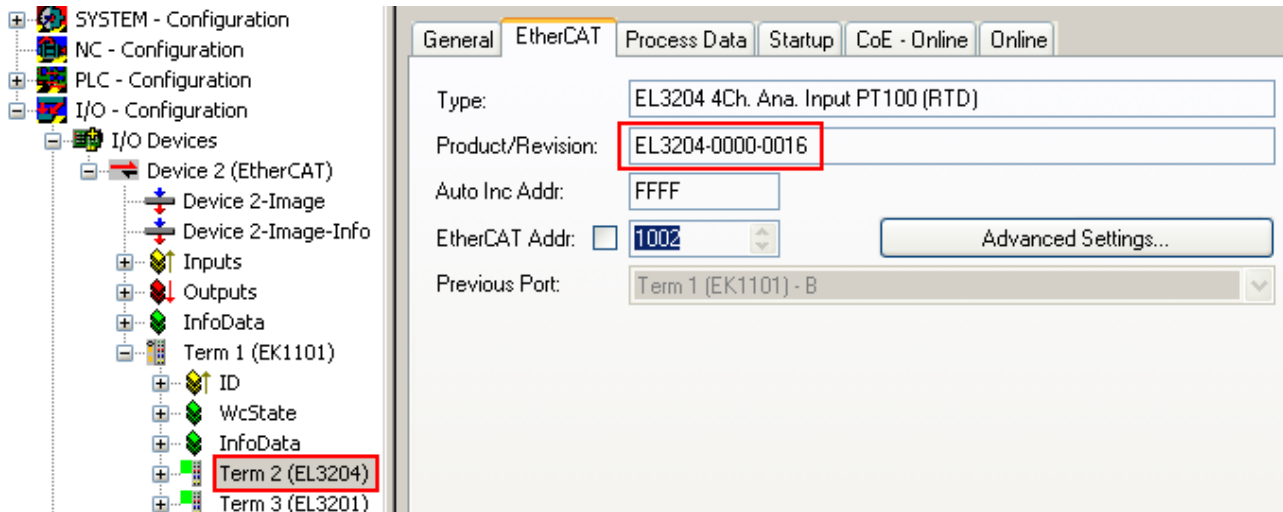


Fig. 157: Device identifier consisting of name EL3204-0000 and revision -0016

The configured identifier must be compatible with the actual device description used as hardware, i.e. the description which the slave has loaded on start-up (in this case EL3204). Normally the configured revision must be the same or lower than that actually present in the terminal network.

For further information on this, please refer to the [EtherCAT system documentation](#).

● Update of XML/ESI description

i 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:

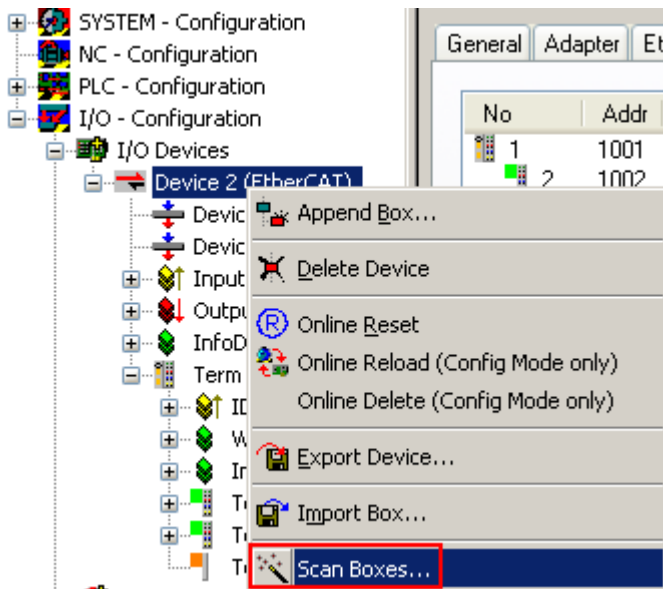


Fig. 158: Scan the subordinate field by right-clicking on the EtherCAT device

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



Fig. 159: Configuration is identical

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

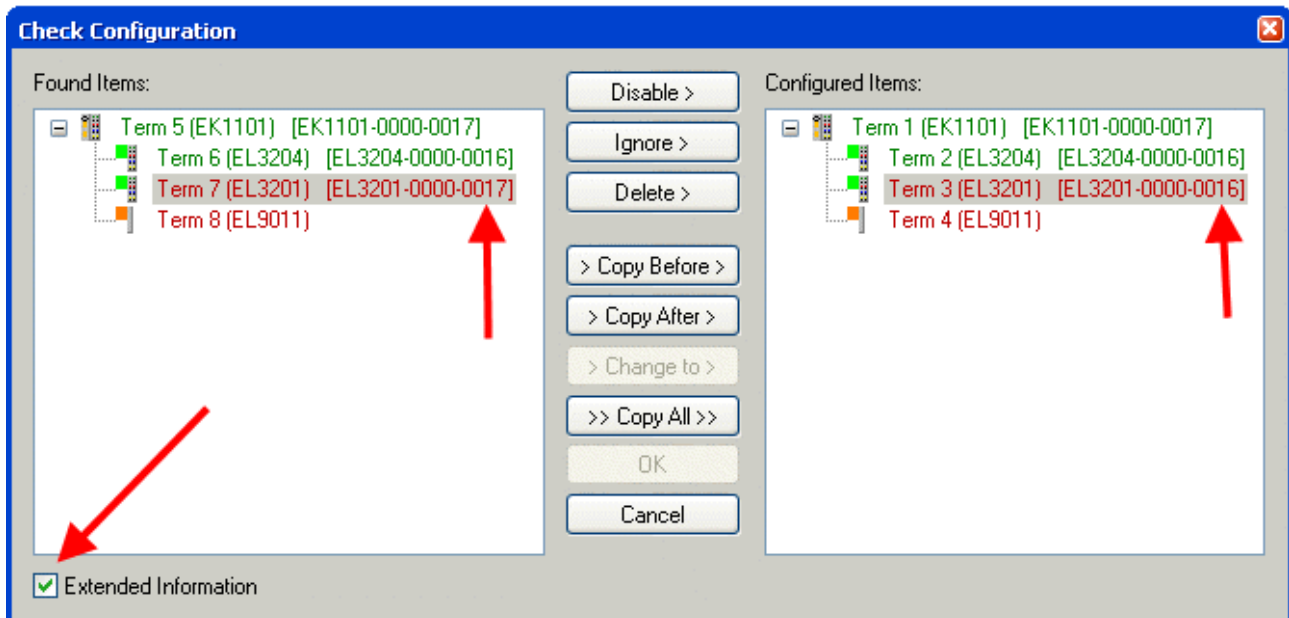


Fig. 160: Change dialog

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*

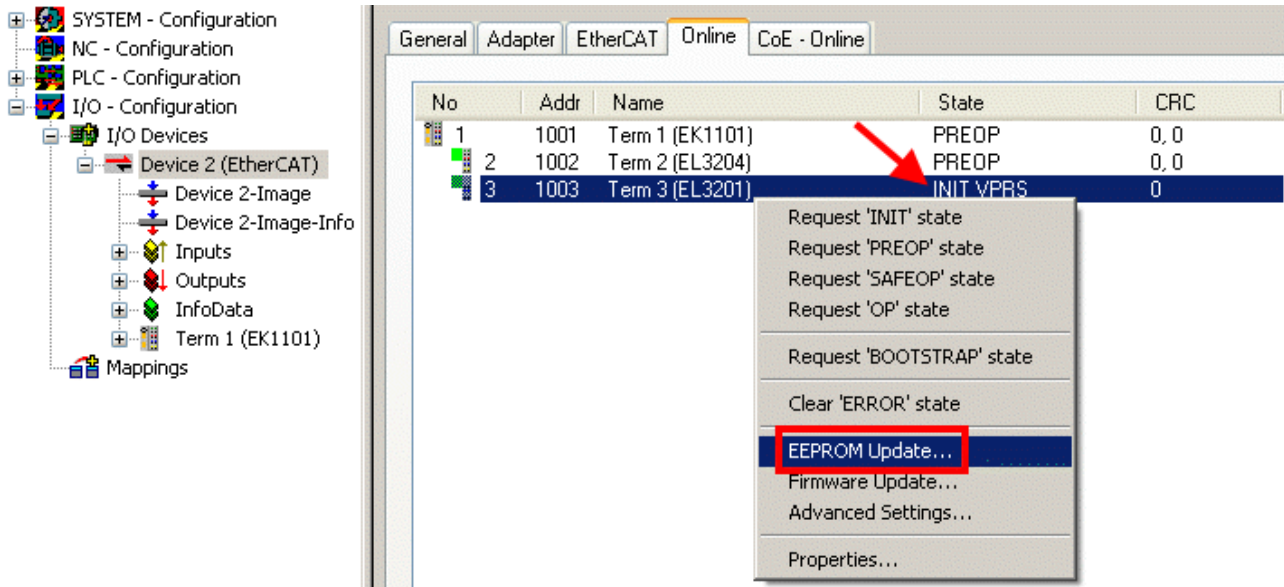


Fig. 161: EEPROM Update

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.

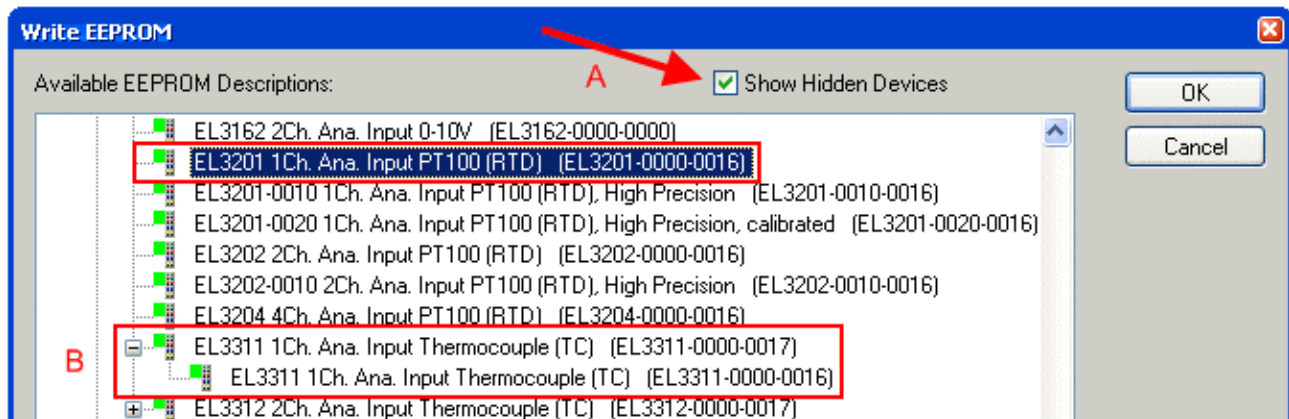


Fig. 162: Selecting the new ESI

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.

7.3.2 Firmware explanation

Determining the firmware version

Determining the version on laser inscription

Beckhoff EtherCAT slaves feature serial numbers applied by laser. The serial number has the following structure: **KK YY FF HH**

- KK - week of production (CW, calendar week)
- YY - year of production
- FF - firmware version
- HH - hardware version

Example with ser. no.: 12 10 03 02:

- 12 - week of production 12
- 10 - year of production 2010
- 03 - firmware version 03
- 02 - hardware version 02

Determining the version via the System Manager

The TwinCAT System Manager shows the version of the controller firmware if the master can access the slave online. Click on the E-Bus Terminal whose controller firmware you want to check (in the example terminal 2 (EL3204)) and select the tab *CoE Online* (CAN over EtherCAT).

● CoE Online and Offline CoE

i

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.

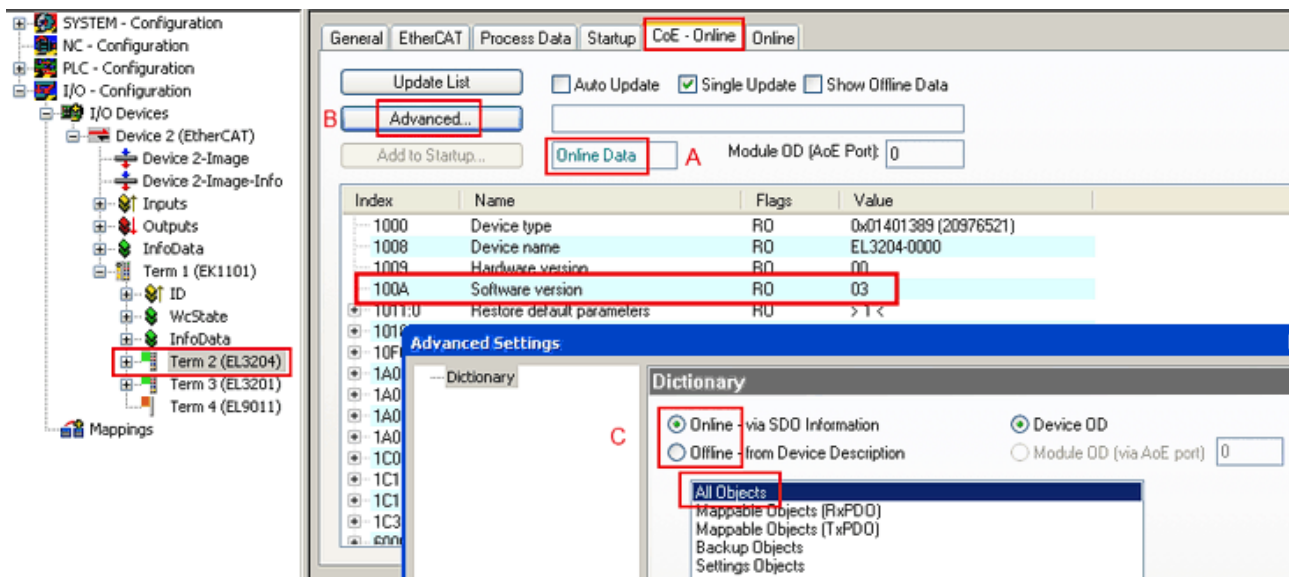


Fig. 163: Display of EL3204 firmware version

In (A) TwinCAT 2.11 shows that the Online CoE directory is currently displayed. If this is not the case, the Online directory can be loaded via the *Online* option in Advanced Settings (B) and double-clicking on *AllObjects*.

7.3.3 Updating controller firmware *.efw

i 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*.

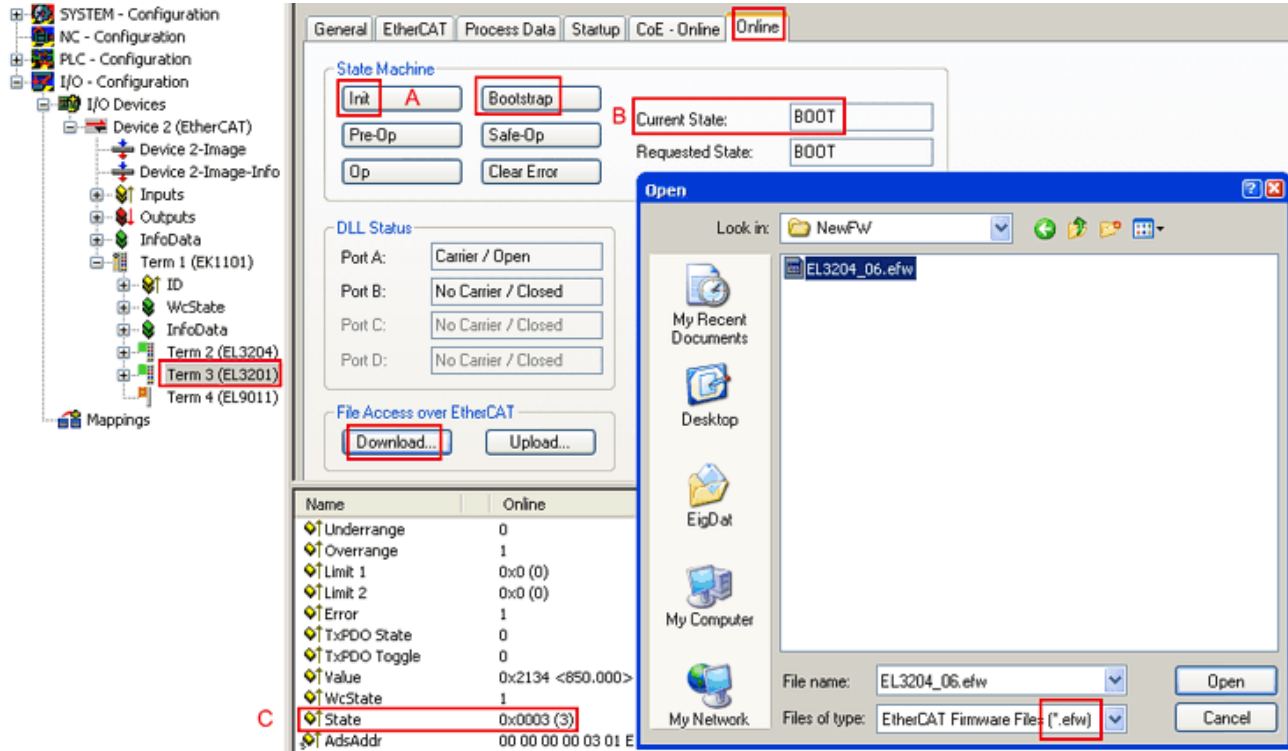
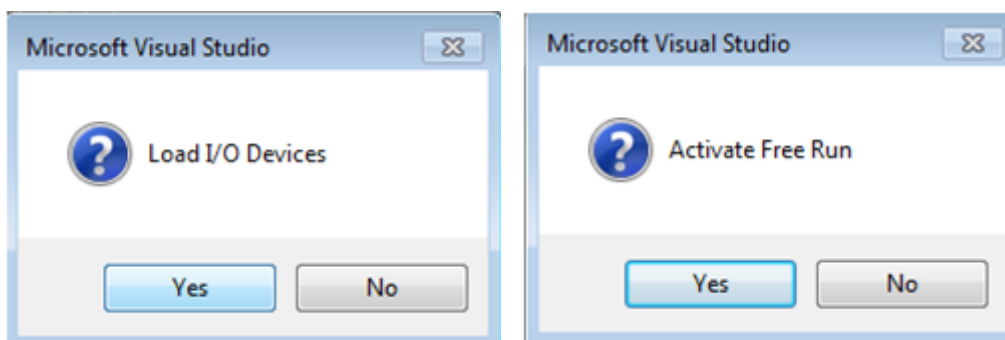


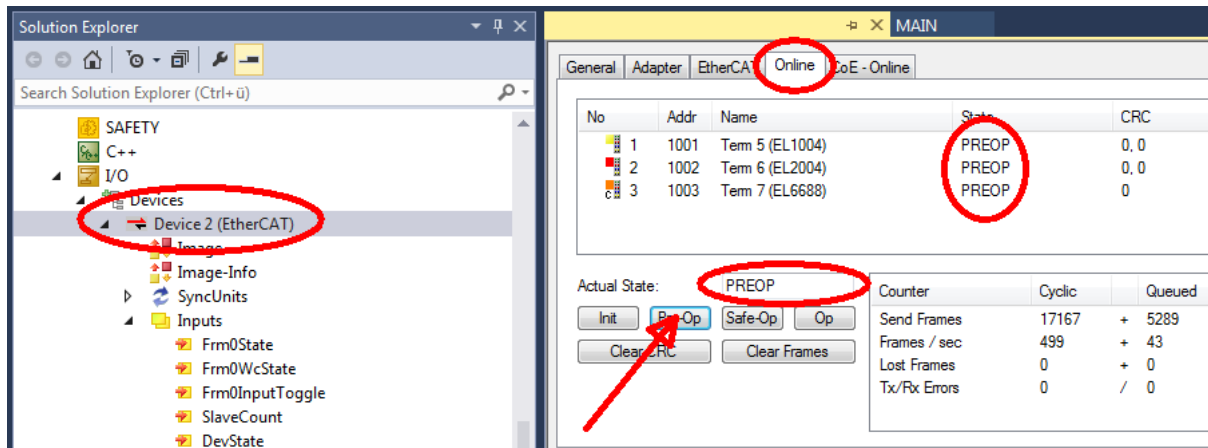
Fig. 164: Firmware Update

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

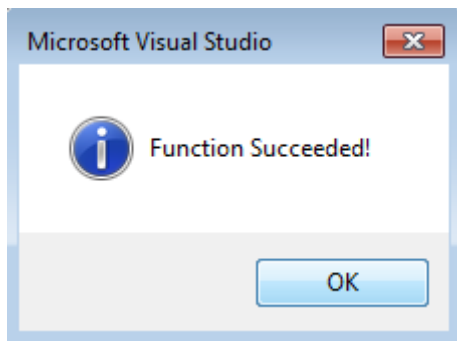
- Switch TwinCAT system to ConfigMode/FreeRun with cycle time ≥ 1 ms (default in ConfigMode is 4 ms). A FW-Update during real time operation is not recommended.



- 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 pass word 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.

7.3.4 FPGA firmware *.rbf

If an FPGA chip deals with the EtherCAT communication an update may be accomplished via an *.rbf file.

- Controller firmware for processing I/O signals
- FPGA firmware for EtherCAT communication (only for terminals with FPGA)

The firmware version number included in the terminal serial number contains both firmware components. If one of these firmware components is modified this version number is updated.

Determining the version via the 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.

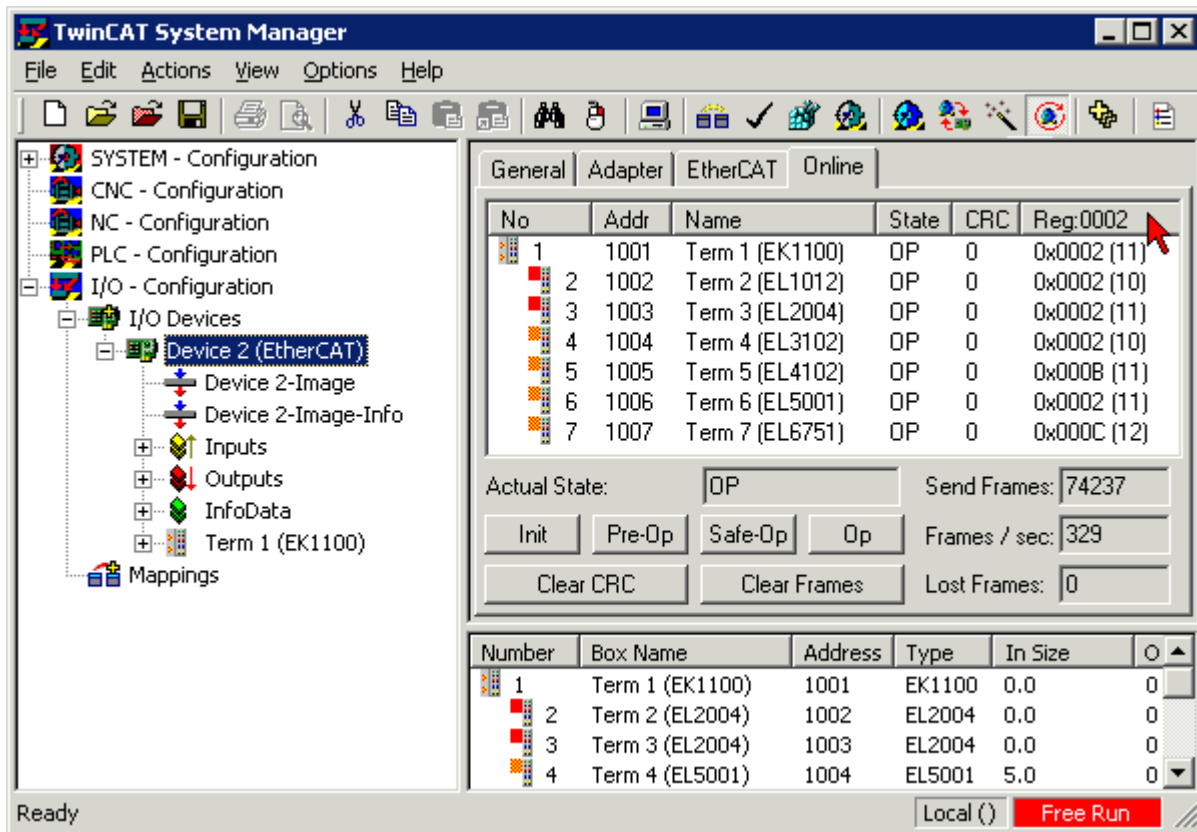


Fig. 165: FPGA firmware version definition

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

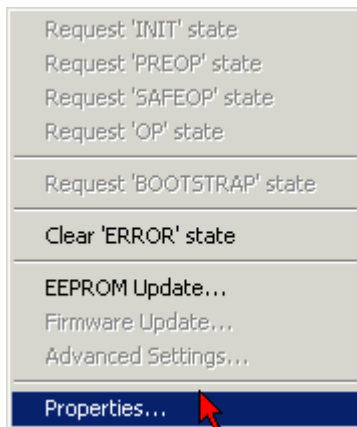


Fig. 166: Context menu *Properties*

The *Advanced Settings* dialog appears where the columns to be displayed can be selected. Under *Diagnosis/Online View* select the *'0002 ETxxx Build'* check box in order to activate the FPGA firmware version display.

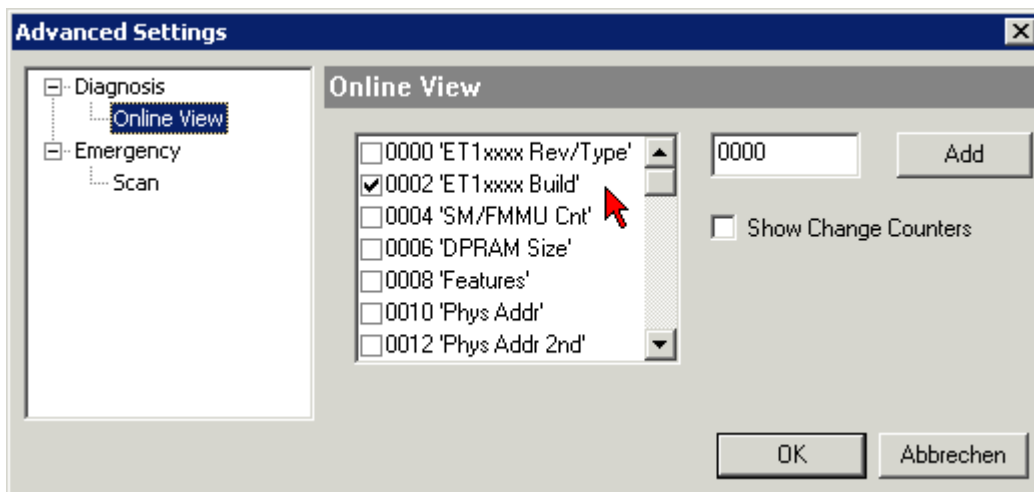


Fig. 167: Dialog *Advanced Settings*

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.

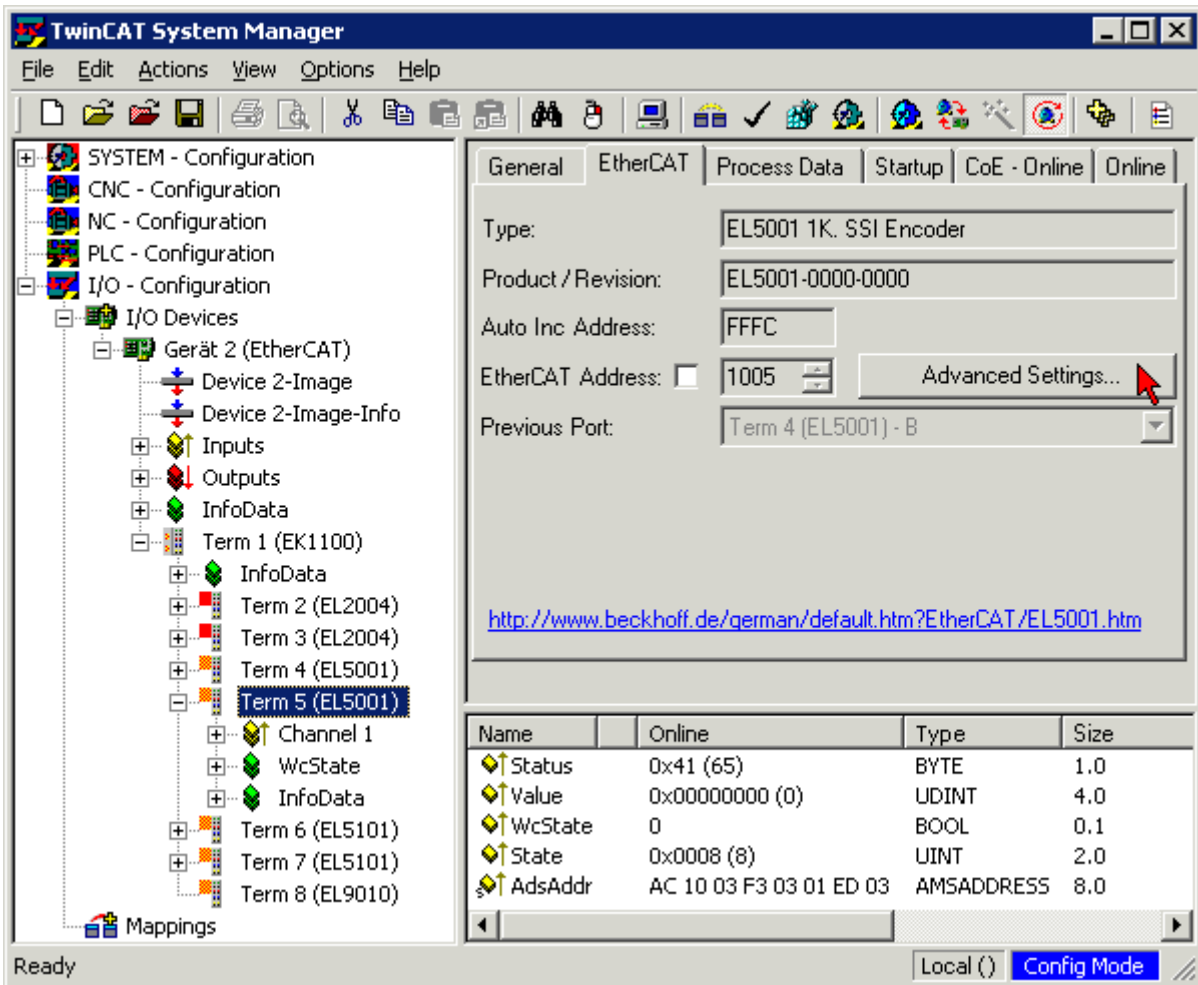
Older firmware versions can only be updated by the manufacturer!

Updating an EtherCAT device

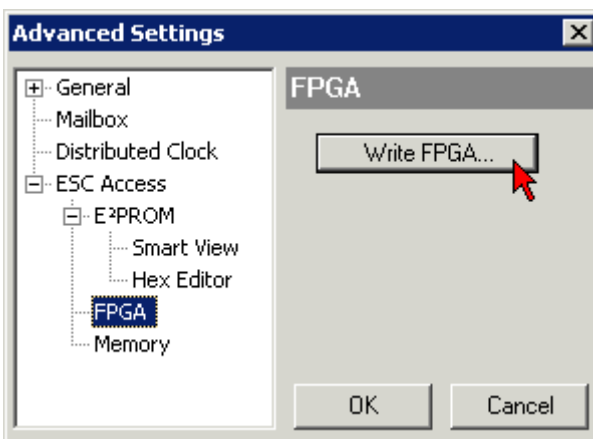
The following sequence order have to be met if no other specifications are given (e.g. by the Beckhoff support):

- 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.

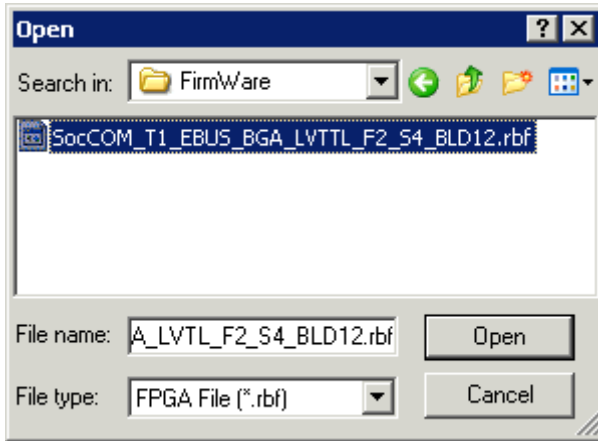
- In the TwinCAT System Manager select the terminal for which the FPGA firmware is to be updated (in the example: Terminal 5: EL5001) and click the *Advanced Settings* button in the *EtherCAT* tab:



- 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

NOTE

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!

7.3.5 Simultaneous updating of several EtherCAT devices

The firmware and ESI descriptions of several devices can be updated simultaneously, provided the devices have the same firmware file/ESI.

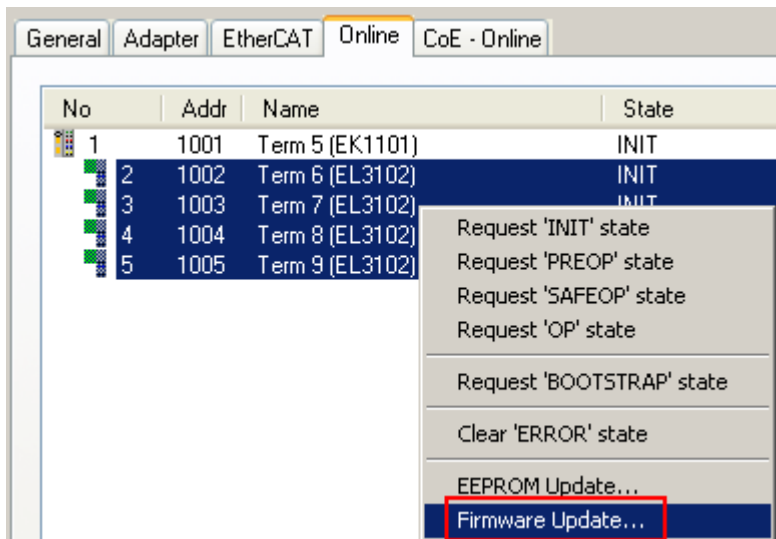


Fig. 168: Multiple selection and firmware update

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

7.4 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.

NOTE

Risk of damage to the device!

Pay attention to the instructions for firmware updates on the [separate page \[▶ 149\]](#). 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 - 05	01	EL6224-0000-0016	2008/05
	02	EL6224-0000-0017	2009/07
	03		2010/06
02 - 17*	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

EL6224-0090			
Hardware (HW)	Firmware (FW)	Revision no.	Release date
16*	01*	EL6224-0090-0016	2017/03

*) 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.

7.5 Restoring the delivery state

To restore the delivery state for backup objects in ELxxxx terminals, the CoE object Restore default parameters, *SubIndex 001* can be selected in the TwinCAT System Manager (Config mode) (see Fig. *Selecting the Restore default parameters PDO*)

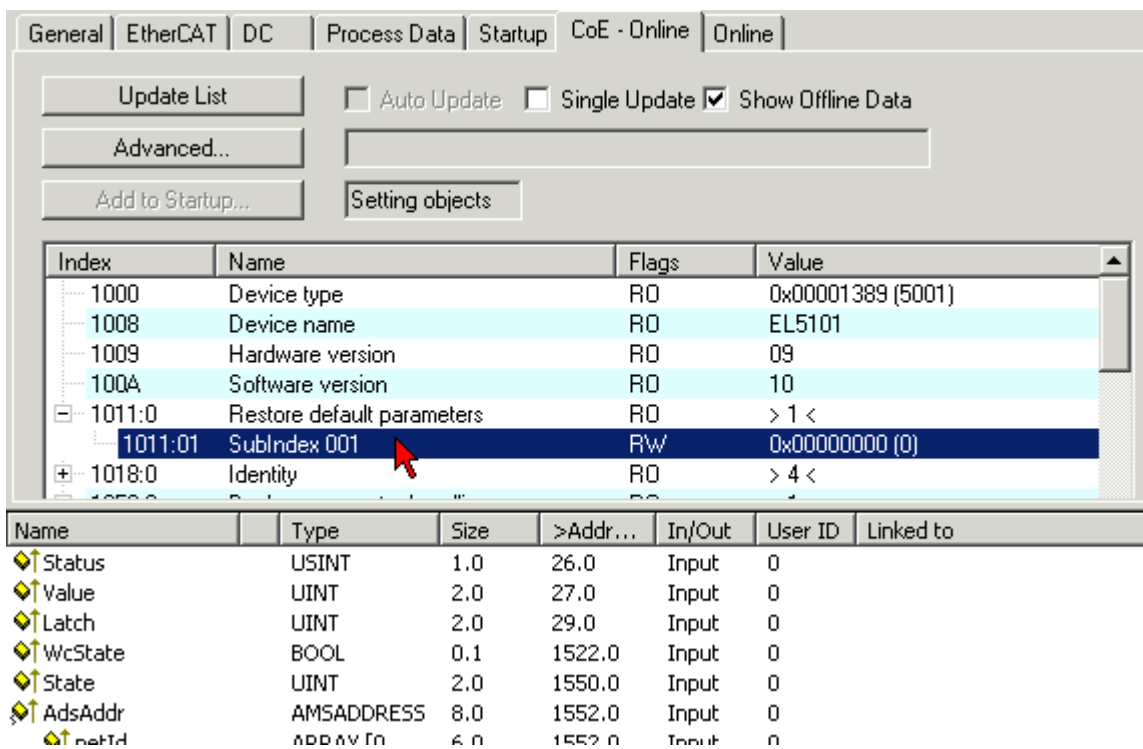


Fig. 169: Selecting the "Restore default parameters" PDO

Double-click on SubIndex 001 to enter the Set Value dialog. Enter the value **1684107116** in field *Dec* or the value **0x64616F6C** in field *Hex* and confirm with *OK* (Fig. *Entering a restore value in the Set Value dialog*). All backup objects are reset to the delivery state.

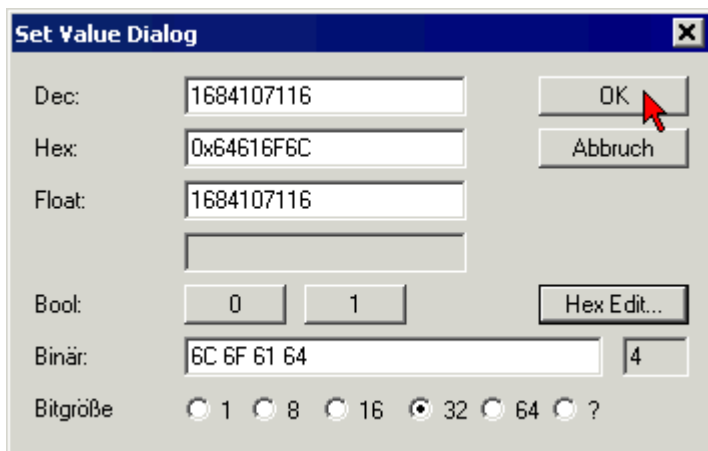


Fig. 170: Entering a restore value in the Set Value dialog

i Alternative restore value

In some older terminals the backup objects can be switched with an alternative restore value: Decimal value: 1819238756, Hexadecimal value: 0x6C6F6164An incorrect entry for the restore value has no effect.

7.6 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:

<http://www.beckhoff.com>

You will also find further documentation for Beckhoff components there.

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

Phone:	+49(0)5246/963-0
Fax:	+49(0)5246/963-198
e-mail:	info@beckhoff.com

Beckhoff Support

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(0)5246/963-157
Fax:	+49(0)5246/963-9157
e-mail:	support@beckhoff.com

Beckhoff 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(0)5246/963-460
Fax:	+49(0)5246/963-479
e-mail:	service@beckhoff.com

List of illustration

Fig. 1	EL5021 EL terminal, standard IP20 IO device with serial/ batch number and revision ID (since 2014/01).....	12
Fig. 2	EK1100 EtherCAT coupler, standard IP20 IO device with serial/ batch number.....	13
Fig. 3	CU2016 switch with serial/ batch number.....	13
Fig. 4	EL3202-0020 with serial/ batch number 26131006 and unique ID-number 204418	13
Fig. 5	EP1258-00001 IP67 EtherCAT Box with batch number/ date code 22090101 and unique serial number 158102.....	14
Fig. 6	EP1908-0002 IP67 EtherCAT Safety Box with batch number/ date code 071201FF and unique serial number 00346070	14
Fig. 7	EL2904 IP20 safety terminal with batch number/ date code 50110302 and unique serial number 00331701.....	14
Fig. 8	ELM3604-0002 terminal with unique ID number (QR code) 100001051 and serial/ batch number 44160201.....	14
Fig. 9	EL6224	15
Fig. 10	EL6224-0090	16
Fig. 11	System manager current calculation	19
Fig. 12	EtherCAT tab -> Advanced Settings -> Behavior -> Watchdog	20
Fig. 13	States of the EtherCAT State Machine.....	22
Fig. 14	"CoE Online " tab	24
Fig. 15	Startup list in the TwinCAT System Manager	25
Fig. 16	Offline list.....	26
Fig. 17	Online list	26
Fig. 18	Spring contacts of the Beckhoff I/O components.....	29
Fig. 19	Attaching on mounting rail	31
Fig. 20	Disassembling of terminal.....	32
Fig. 21	Power contact on left side.....	33
Fig. 22	Standard wiring.....	35
Fig. 23	Pluggable wiring	35
Fig. 24	High Density Terminals.....	36
Fig. 25	Connecting a cable on a terminal point	37
Fig. 26	Correct positioning	39
Fig. 27	Incorrect positioning.....	39
Fig. 28	Recommended distances for standard installation position	40
Fig. 29	Other installation positions	41
Fig. 30	EL6224_EL6224-0090 - LEDs and connection	44
Fig. 31	Relationship between user side (commissioning) and installation.....	46
Fig. 32	Control configuration with Embedded PC, input (EL1004) and output (EL2008)	47
Fig. 33	Initial TwinCAT 2 user interface.....	48
Fig. 34	Selection of the target system	49
Fig. 35	Specify the PLC for access by the TwinCAT System Manager: selection of the target system ..	49
Fig. 36	Select "Scan Devices..."	50
Fig. 37	Automatic detection of I/O devices: selection the devices to be integrated.....	50
Fig. 38	Mapping of the configuration in the TwinCAT 2 System Manager.....	51
Fig. 39	Reading of individual terminals connected to a device.....	51
Fig. 40	TwinCAT PLC Control after startup	52
Fig. 41	Sample program with variables after a compile process (without variable integration).....	53

Fig. 42	Appending the TwinCAT PLC Control project	53
Fig. 43	PLC project integrated in the PLC configuration of the System Manager	54
Fig. 44	Creating the links between PLC variables and process objects	54
Fig. 45	Selecting PDO of type BOOL	55
Fig. 46	Selecting several PDOs simultaneously: activate "Continuous" and "All types"	55
Fig. 47	Application of a "Goto Link" variable, using "MAIN.bEL1004_Ch4" as a sample	56
Fig. 48	Choose target system (remote)	57
Fig. 49	PLC Control logged in, ready for program startup	58
Fig. 50	Initial TwinCAT 3 user interface	59
Fig. 51	Create new TwinCAT project	59
Fig. 52	New TwinCAT3 project in the project folder explorer	60
Fig. 53	Selection dialog: Choose the target system	60
Fig. 54	Specify the PLC for access by the TwinCAT System Manager: selection of the target system ..	61
Fig. 55	Select "Scan"	61
Fig. 56	Automatic detection of I/O devices: selection the devices to be integrated	62
Fig. 57	Mapping of the configuration in VS shell of the TwinCAT3 environment	62
Fig. 58	Reading of individual terminals connected to a device	63
Fig. 59	Adding the programming environment in "PLC"	64
Fig. 60	Specifying the name and directory for the PLC programming environment	64
Fig. 61	Initial "Main" program of the standard PLC project	65
Fig. 62	Sample program with variables after a compile process (without variable integration)	66
Fig. 63	Start program compilation	66
Fig. 64	Creating the links between PLC variables and process objects	67
Fig. 65	Selecting PDO of type BOOL	67
Fig. 66	Selecting several PDOs simultaneously: activate "Continuous" and "All types"	68
Fig. 67	Application of a "Goto Link" variable, using "MAIN.bEL1004_Ch4" as a sample	68
Fig. 68	TwinCAT development environment (VS shell): logged-in, after program startup	69
Fig. 69	System Manager "Options" (TwinCAT 2)	71
Fig. 70	Call up under VS Shell (TwinCAT 3)	71
Fig. 71	Overview of network interfaces	71
Fig. 72	EtherCAT device properties(TwinCAT 2): click on „Compatible Devices...“ of tab “Adapter”	72
Fig. 73	Windows properties of the network interface	72
Fig. 74	Exemplary correct driver setting for the Ethernet port	73
Fig. 75	Incorrect driver settings for the Ethernet port	74
Fig. 76	TCP/IP setting for the Ethernet port	75
Fig. 77	Identifier structure	76
Fig. 78	OnlineDescription information window (TwinCAT 2)	77
Fig. 79	Information window OnlineDescription (TwinCAT 3)	77
Fig. 80	File OnlineDescription.xml created by the System Manager	78
Fig. 81	Indication of an online recorded ESI of EL2521 as an example	78
Fig. 82	Information window for faulty ESI file (left: TwinCAT 2; right: TwinCAT 3)	78
Fig. 83	Using the ESI Updater (>= TwinCAT 2.11)	80
Fig. 84	Using the ESI Updater (TwinCAT 3)	80
Fig. 85	Append EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)	81
Fig. 86	Selecting the EtherCAT connection (TwinCAT 2.11, TwinCAT 3)	81
Fig. 87	Selecting the Ethernet port	81

Fig. 88	EtherCAT device properties (TwinCAT 2)	82
Fig. 89	Appending EtherCAT devices (left: TwinCAT 2; right: TwinCAT 3).....	82
Fig. 90	Selection dialog for new EtherCAT device	83
Fig. 91	Display of device revision	83
Fig. 92	Display of previous revisions	84
Fig. 93	Name/revision of the terminal.....	84
Fig. 94	EtherCAT terminal in the TwinCAT tree (left: TwinCAT 2; right: TwinCAT 3).....	85
Fig. 95	Differentiation local/target system (left: TwinCAT 2; right: TwinCAT 3).....	86
Fig. 96	Scan Devices (left: TwinCAT 2; right: TwinCAT 3).....	86
Fig. 97	Note for automatic device scan (left: TwinCAT 2; right: TwinCAT 3).....	86
Fig. 98	Detected Ethernet devices	87
Fig. 99	Example default state	87
Fig. 100	Installing EthetCAT terminal with revision -1018	88
Fig. 101	Detection of EtherCAT terminal with revision -1019.....	88
Fig. 102	Scan query after automatic creation of an EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)	88
Fig. 103	Manual triggering of a device scan on a specified EtherCAT device (left: TwinCAT 2; right: TwinCAT 3).....	89
Fig. 104	Scan progressexemplary by TwinCAT 2	89
Fig. 105	Config/FreeRun query (left: TwinCAT 2; right: TwinCAT 3).....	89
Fig. 106	Displaying of “Free Run” and “Config Mode” toggling right below in the status bar	89
Fig. 107	TwinCAT can also be switched to this state by using a button (left: TwinCAT 2; right: TwinCAT 3)	89
Fig. 108	Online display example	90
Fig. 109	Faulty identification	90
Fig. 110	Identical configuration (left: TwinCAT 2; right: TwinCAT 3).....	91
Fig. 111	Correction dialog	91
Fig. 112	Name/revision of the terminal.....	92
Fig. 113	Correction dialog with modifications	93
Fig. 114	Dialog “Change to Compatible Type...” (left: TwinCAT 2; right: TwinCAT 3).....	93
Fig. 115	TwinCAT 2 Dialog Change to Alternative Type	93
Fig. 116	Branch element as terminal EL3751.....	94
Fig. 117	“General” tab.....	94
Fig. 118	„EtherCAT“ tab.....	95
Fig. 119	“Process Data” tab.....	96
Fig. 120	Configuring the process data.....	97
Fig. 121	„Startup“ tab.....	98
Fig. 122	“CoE – Online” tab	99
Fig. 123	Dialog “Advanced settings”.....	100
Fig. 124	„Online“ tab	100
Fig. 125	"DC" tab (Distributed Clocks).....	101
Fig. 126	Selection of the diagnostic information of an EtherCAT Slave	103
Fig. 127	Basic EtherCAT Slave Diagnosis in the PLC.....	104
Fig. 128	EL3102, CoE directory	106
Fig. 129	Example of commissioning aid for a EL3204	107
Fig. 130	Default behaviour of the System Manager	108
Fig. 131	Default target state in the Slave	108

Fig. 132 PLC function blocks	109
Fig. 133 Illegally exceeding the E-Bus current	110
Fig. 134 Warning message for exceeding E-Bus current	110
Fig. 135 IO-Link point-to-point communication	111
Fig. 136 Establishment of IO Link communication.....	112
Fig. 137 "IO Link" tab.....	114
Fig. 138 "IO-Link" tab, integration of the devices.....	115
Fig. 139 Creating IO link devices.....	117
Fig. 140 IO link device settings.....	118
Fig. 141 Advanced Settings.....	119
Fig. 142 Reading/writing parameters.....	120
Fig. 143 Illustration of the system communication of an EtherCAT master	121
Fig. 144 Illustration of the EL6224 process data allocation, SM3 inputs	122
Fig. 145 AoE-NetID allocation	125
Fig. 146 DiagHistory tab	126
Fig. 147 Adding the TwinSAFE SC process data under the component, e.g. EL5021-0090	127
Fig. 148 TwinSAFE SC component process data, example EL5021-0090	127
Fig. 149 Adding a TwinSAFE SC connection	128
Fig. 150 Creating a link to TwinSAFE SC terminal	128
Fig. 151 Selecting a free CRC	129
Fig. 152 Selecting the process data size and the process data	129
Fig. 153 Selection of the process data	130
Fig. 154 CoE objects 0x8010:01 and 0x8010:02.....	130
Fig. 155 Entering the safety address and the CRC	130
Fig. 156 EL6224-0090, process data image (default).....	131
Fig. 157 Device identifier consisting of name EL3204-0000 and revision -0016	150
Fig. 158 Scan the subordinate field by right-clicking on the EtherCAT device	151
Fig. 159 Configuration is identical	151
Fig. 160 Change dialog	151
Fig. 161 EEPROM Update	152
Fig. 162 Selecting the new ESI.....	152
Fig. 163 Display of EL3204 firmware version	153
Fig. 164 Firmware Update	154
Fig. 165 FPGA firmware version definition	156
Fig. 166 Context menu Properties	156
Fig. 167 Dialog Advanced Settings	157
Fig. 168 Multiple selection and firmware update	159
Fig. 169 Selecting the "Restore default parameters" PDO	161
Fig. 170 Entering a restore value in the Set Value dialog.....	161