BECKHOFF New Automation Technology

Documentation | EN

EL2502-0010

2 Channel Pulse Width Output Terminal, 24 V DC with Timestamp

2025-05-16 | Version: 1.4

Table of contents

1 Foreword 5
1.1 Notes on the documentation 5
1.2 Guide through documentation 6
1.3 Safety instructions 7
1.4 Documentation issue status 8
1.5 Version identification of EtherCAT devices 9
1.5.1 General notes on marking 9
1.5.2 Version identification of EL terminals 9
1.5.3 Beckhoff Identification Code (BIC) 11
1.5.4 Electronic access to the BIC (eBIC) 13

2 Product overview 15
2.1 Introduction 15
2.2 Technical data 16
2.3 Technology 18
2.4 Start 18

3 Mounting and wiring 19
3.1 Instructions for ESD protection 19
3.2 Explosion protection 20
3.2.1 ATEX - Special conditions (standard temperature range) 20
3.2.2 IECEx - Special conditions 21
3.2.3 Continuative documentation for ATEX and IECEx 22
3.3 UL notice 23
3.4 Installation on mounting rails 24
3.5 Installation instructions for enhanced mechanical load capacity 27
3.6 Connection 28
3.6.1 Connection system 28
3.6.2 Wiring 30
3.6.3 Shielding 31
3.7 Note - power supply 32
3.8 Prescribed installation position 33
3.9 Positioning of passive Terminals 34
3.10 LEDs and connection 35
3.11 Disposal 36

4 Commissioning with TwinCAT 37
4.1 TwinCAT Quick Start 37
4.1.1 TwinCAT 2 40
4.1.2 TwinCAT 3 50
4.2 TwinCAT Development Environment 63
4.2.1 Installation of the TwinCAT real-time driver 64
4.2.2 Notes regarding ESI device description 70
4.2.3 TwinCAT ESI Updater 74
4.2.4 Distinction between Online and Offline 76
4.2.5 OFFLINE configuration creation 76

4.2.6 ONLINE configuration creation ... 82
4.2.7 EtherCAT subscriber configuration ... 90
4.2.8 Import/Export of EtherCAT devices with SCI and XTI ... 100
4.3 General Commissioning Instructions for an EtherCAT Slave ... 106
5 EL2502-0010 - Commissioning and operation ... 114
5.1 Basic operation modes and process data ... 114
5.1.1 Operation mode “PWM, DC” ... 116
5.1.2 Operation mode "PWM, Period" ... 122
5.1.3 PWM” operation mode ... 124
5.2 Safe state in the event of a bus error (watchdog) ... 126
5.3 Switch channel to high-resistance (Tristate) ... 126
5.4 Object description and parameterization ... 127
5.4.1 Restore object ... 127
5.4.2 0x80xx | Configuration data ... 128
5.4.3 0x60xx | Input data ... 129
5.4.4 0x70xx | Output data ... 130
5.4.5 0x80nE, 0xFxxx | Information and diagnostic data ... 131
5.4.6 0x10xx - 0x1C33 | Standard objects ... 131
6 Appendix ... 140
6.1 EtherCAT AL Status Codes ... 140
6.2 Firmware compatibility ... 141
6.3 Firmware Update EL/ES/EM/ELM/EP/EPP/ERPxxxx ... 142
6.3.1 Device description ESI file/XML ... 143
6.3.2 Firmware explanation ... 146
6.3.3 Updating controller firmware *.efw ... 147
6.3.4 FPGA firmware *.rbf ... 149
6.3.5 Simultaneous updating of several EtherCAT devices ... 153
6.4 Restoring the delivery state ... 154
6.5 Support and Service ... 156

1 Foreword

1.1 Notes on the documentation

Intended audience

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

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

Disclaimer

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

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

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

Trademarks

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

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

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

Copyright

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

Third-party brands

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

1.2 Guide through documentation

NOTICE

Further components of documentation

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

Title	Description
EtherCAT System Documentation (PDF)	• System overview • EtherCAT basics • Cable redundancy • Hot Connect • EtherCAT devices configuration
Explosion Protection for Terminal Systems (PDF)	Notes on the use of the Beckhoff terminal systems in hazardous areas according to ATEX and IECEx
Infrastructure for EtherCAT/Ethernet (PDF)	Technical recommendations and notes for design, implementation and testing
Software Declarations I/O (PDF)	Open source software declarations for Beckhoff I/O components

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

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

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

1.3 Safety instructions

Safety regulations

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

Exclusion of liability

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

Personnel qualification

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

Signal words

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

Personal injury warnings

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

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

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

Warning of damage to property or environment

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

Information on handling the product

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

1.4 Documentation issue status

Version	Comment
1.4	• Update chapter "Technical data" • Update chapter "Technology" • Update chapter "EL2502 - Commissioning and operation" • Update chapter "Watchdog" • Update chapter "Object description and parameterization" • Structural update
1.3	• Update chapter "Technical data" • Update chapter "Mounting and wiring" • Update revision status • Structural update
1.2	• Update chapter "Technical data" • Structural update
1.1	• Update chapter "Technical data" • Update chapter "Version identification of EtherCAT devices" • Structural update • Update notes • Update revision status
1.0	• First release
0.1	• Provisional documentation for EL2502-0010

1.5 Version identification of EtherCAT devices

1.5.1 General notes on marking

Designation

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

- family key
- type
- version
- revision

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

Notes

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

1.5.2 Version identification of EL terminals

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

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

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

1.5.3 Beckhoff Identification Code (BIC)

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

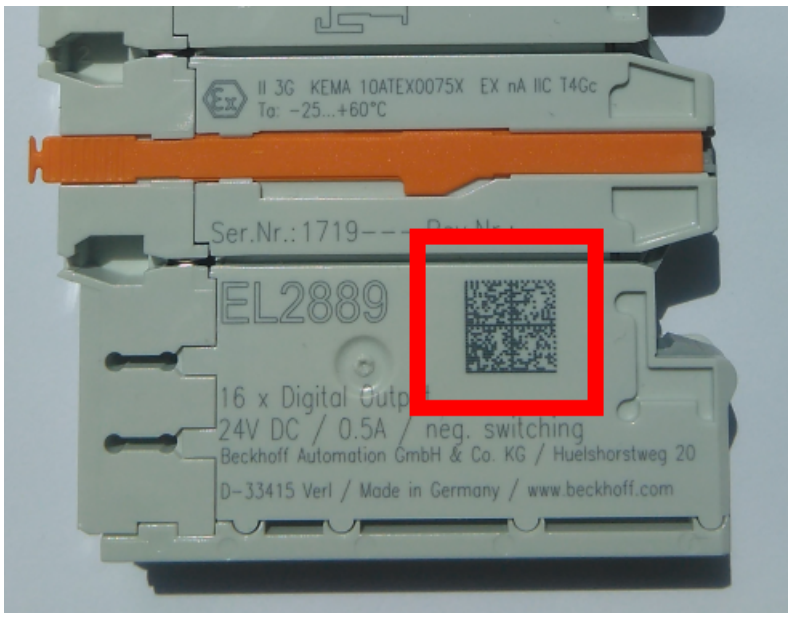


Fig. 2: BIC as data matrix code (DMC, code scheme ECC200)

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

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

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

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

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

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

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

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

Structure of the BIC

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

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

Accordingly as DMC:

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

BTN

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

NOTICE

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

1.5.4 Electronic access to the BIC (eBIC)

Electronic BIC (eBIC)

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

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

K-bus devices (IP20, IP67)

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

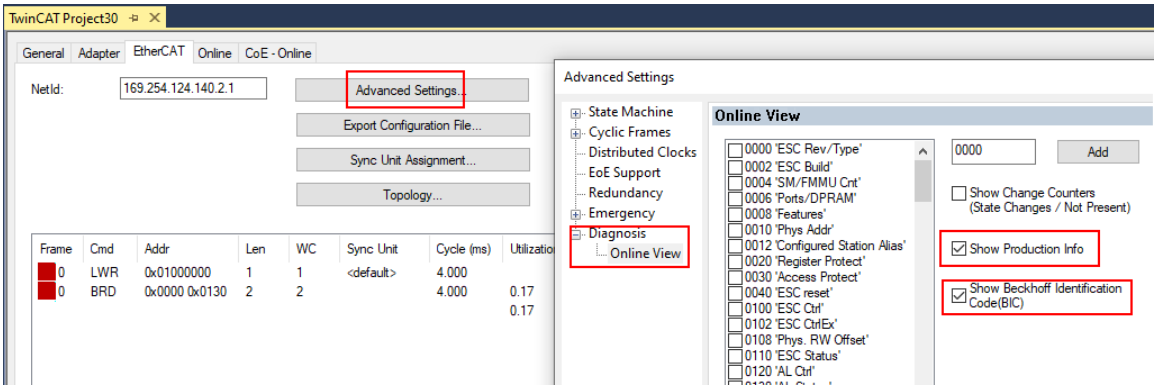
EtherCAT devices (IP20, IP67)

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

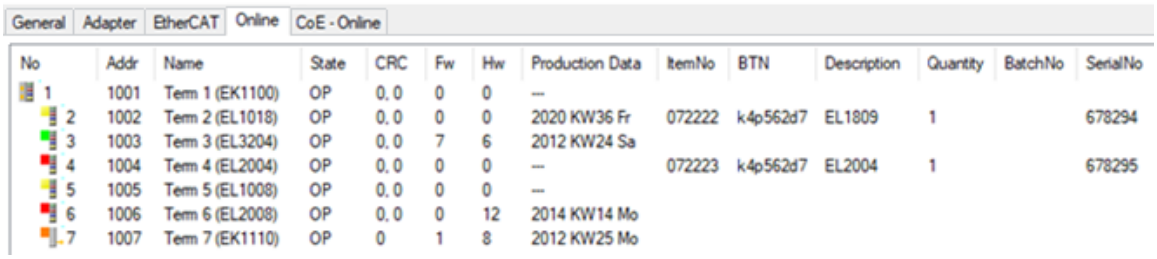
Beckhoff also stores the eBIC in the ESI-EEPROM. The eBIC was introduced into Beckhoff IO production (terminals, box modules) in 2020; as of 2023, implementation is largely complete.

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

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

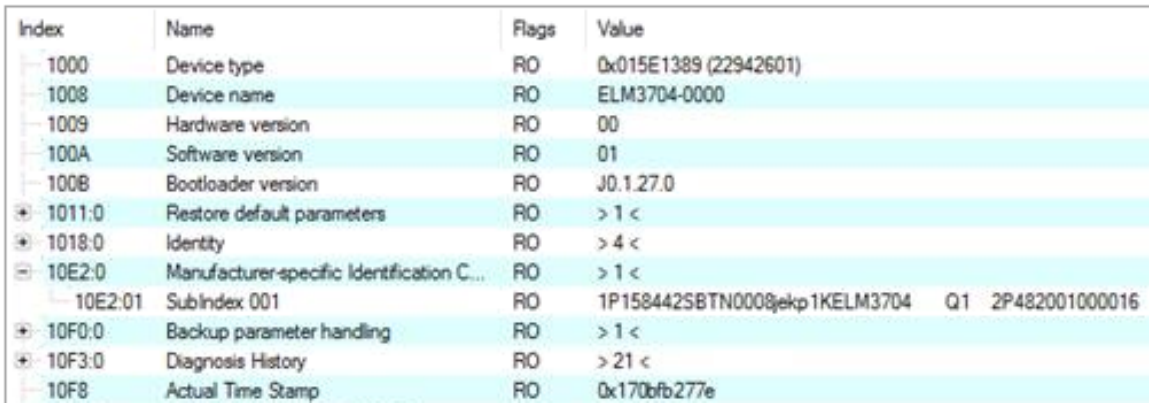


  - The BTN and its contents are then displayed:



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

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



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

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

PROFIBUS, PROFINET, and DeviceNet devices

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

2 Product overview

2.1 Introduction

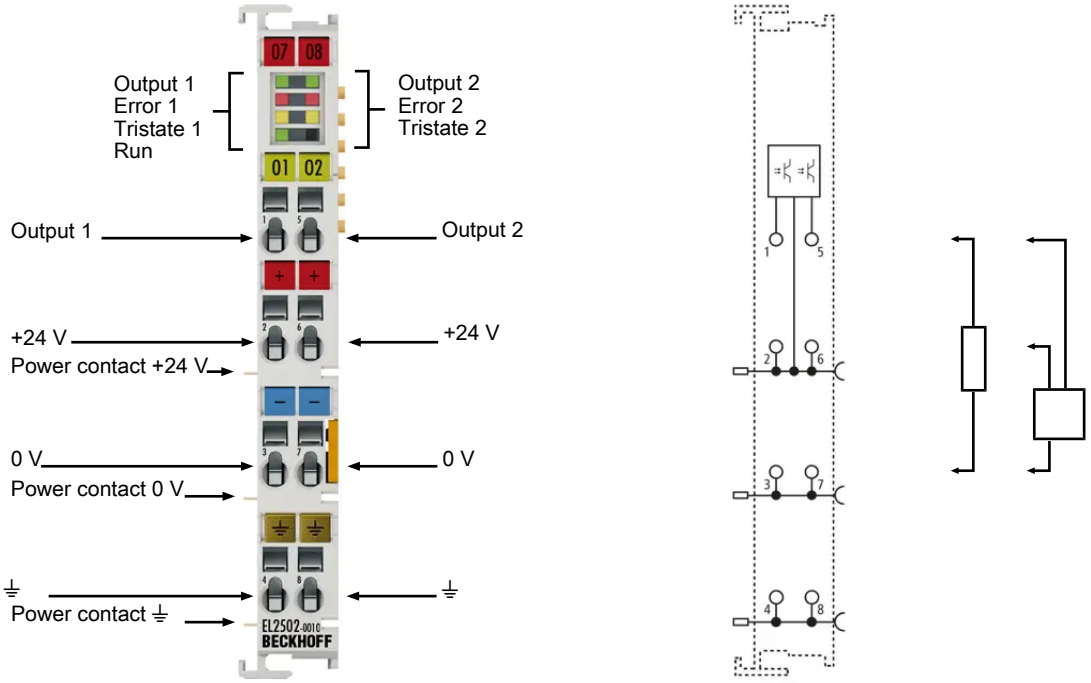


Fig. 4: EL2502-0010

2-channel pulse width output terminal, 24 V_{DC}, with timestamp

The EL2502-0010 output terminal modulates a binary signal in the pulse width and outputs it electrically isolated from the E-bus. The mark/space ratio is prescribed by a 16 bit value from the automation device.

The output stage is protected against overload, short-circuit and overtemperature.

The EtherCAT Terminal has two channels that indicate their signal state via light emitting diodes. The LEDs are clocked with the outputs and indicate the duty cycle by their brightness.
The outputs are switched on or off with high precision to match the transmitted timestamp. This technology enables output switching times to be specified precisely across the system. The distributed clocks are used for reference.

2.2 Technical data

PWM output	EL2502-0010
Number and type of outputs	2x PWM output, push-pull outputs
Nominal auxiliary voltage	24 V_{DC} (-15 %/+20 %)
Load type	ohmic, inductive, lamp load
Output delay due to 24 V power supply unit	< 1 µs typ.
Output current (per channel)	max. 0.5 A (short-circuit proof, 1 A driver module)
Short-circuit current	< 1.5 A typ.
Base frequency	20 Hz ... 500 kHz, the maximum frequency depends on the operation mode: • 300 kHz (basic operation modes: "PWM DC" and "PWM Period") • 500 kHz Default: 250 Hz (basic operation mode: "PWM")
Duty cycle	0 ... 100 % (pulse duration T_{ON} > 750 ns, pause duration T_{OFF} > 500 ns)
Resolution	9 ... 15 bits depending on the frequency in the selected basic operation mode ("PWM, DC", "PWM, Period", "PWM")
Reverse polarity protection	yes

Function and communication	EL2502-0010
Distributed clocks	yes
Accuracy of Distributed Clocks	<< 1 µs
Configuration	via TwinCAT System Manager
Bit width in process image	2 x 128-Bit-In/Output
Special features	• separate frequency can be set for each channel, • Switch-on and switch-off times can be set via timestamp

Supply and potentials	EL2502-0010
Current consumption via E-bus	100 mA typ.
Power supply for the electronics	via the E-bus
Current consumption of power contacts	typ. 30 mA
Electrical isolation	500 V (E-bus/field voltage)

General data	EL2502-0010
Weight	approx. 50 g
Dimensions (W x H x D)	approx. 15 mm x 100 mm x 70 mm
Installation [▶ 24]	on 35 mm mounting rail, conforms to EN 60715
Installation position	see note [▶ 33]!

Environmental conditions	EL2502-0010
Permissible ambient temperature range during operation	0 °C ... + 55 °C
Permissible ambient temperature range during storage	-25 °C ... + 85 °C
Permissible relative air humidity	95 %, no condensation

Standards and approvals	EL2502-0010
Enhanced mechanical load capacity	yes, see also Installation instructions for enhanced mechanical load capacity [▶ 27]
Vibration / shock resistance	conforms to EN 60068-2-6 / EN 60068-2-27
EMC immunity / emission	conforms to EN 61000-6-2 / EN 61000-6-4
Protection rating	IP20
Identification / approval*	CE, UKCA, EAC, CCC, cULus [▶ 23], ATEX [▶ 20], IECEx [▶ 21]

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

Ex markings

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

2.3 Technology

The EL2502-0010 output terminal modulates a binary 24 V signal in the pulse width, i.e. changes the duty cycle. The output frequency is configurable for each channel.

Thanks to the implemented distributed clocks functionality, the terminal is able to switch the PWM output on or off or change its state by means of the transmitted timestamp.

Using the "DC Immediately" function, you have the option of using

- "DC Immediately" = TRUE
  to abort the currently running PWM output after reaching the specified timestamp and clock out with newly specified parameters or
- "DC Immediately" = FALSE (factory setting)
  to adopt the new PWM output parameters only after completion of the current period.

The peripheral side of the electronics is electrically isolated from the internal E-bus, and therefore also from the fieldbus.

The process data resolution of 16 bits is converted to the hardware resolution of 10 bits inside the terminal and depends on the selected operation mode.

For a description of the operation, see chapter "EL2502-0010 Commissioning and operation [▶ 114]".

2.4 Start

For commissioning:

- mount the terminal as described in the chapter "Mounting and wiring [▶ 19]",
- configure the terminal in TwinCAT as described in the chapter “Commissioning with TwinCAT [▶ 37]”,
- parameterize the terminal as described in the chapter "EL2502-0010 - Commissioning and operation" [▶ 114].

3 Mounting and wiring

3.1 Instructions for ESD protection

NOTICE

Destruction of the devices by electrostatic discharge possible!

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

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

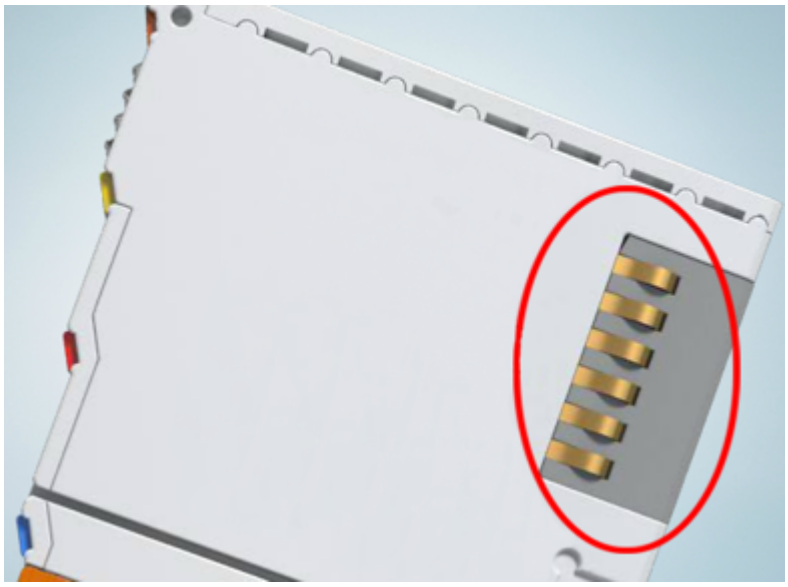


Fig. 5: Spring contacts of the Beckhoff I/O components

3.2 Explosion protection

3.2.1 ATEX - Special conditions (standard temperature range)

⚠ WARNING

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

- The certified components are to be installed in a suitable housing that guarantees a protection class of at least IP54 in accordance with EN 60079-7! The environmental conditions during use are thereby to be taken into account!
- If the temperatures during rated operation are higher than 70°C at the feed-in points of cables, lines or pipes, or higher than 80°C at the wire branching points, then cables must be selected whose temperature data correspond to the actual measured temperature values!
- Observe the permissible ambient temperature range of 0 to 55°C for the use of Beckhoff fieldbus components standard temperature range in potentially explosive areas!
- Measures must be taken to protect against the rated operating voltage being exceeded by more than 40% due to short-term interference voltages!
- The individual terminals may only be unplugged or removed from the Bus Terminal system if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- The connections of the certified components may only be connected or disconnected if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- The fuses of the KL92xx/EL92xx power feed terminals may only be exchanged if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- Address selectors and ID switches may only be adjusted if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!

Standards

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

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

Marking

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

II 3 G Ex ec IIC T4 Gc
KEMA 10 ATEX0075 X
DEKRA 22UKEX6024X
Ta: 0 ... + 55°C

3.2.2 IECEx - Special conditions

⚠ WARNING

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

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

Standards

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

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

Marking

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

IECEx DEK 16.0078 X

Ex ec IIC T4 Gc

3.2.3 Continuative documentation for ATEX and IECEx

NOTICE	
	Continuative documentation about explosion protection according to ATEX and IECEx Pay also attention to the continuative documentation **Ex. Protection for Terminal Systems** Notes on the use of the Beckhoff terminal systems in hazardous areas according to ATEX and IECEx, that is available for download within the download area of your product on the Beckhoff homepage www.beckhoff.com!

3.3 UL notice

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

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

⚠ CAUTION	
	For devices with Ethernet connectors Not for connection to telecommunication circuits.

Basic principles

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

3.4 Installation on mounting rails

⚠ WARNING

Risk of electric shock and damage of device!

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

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

Assembly

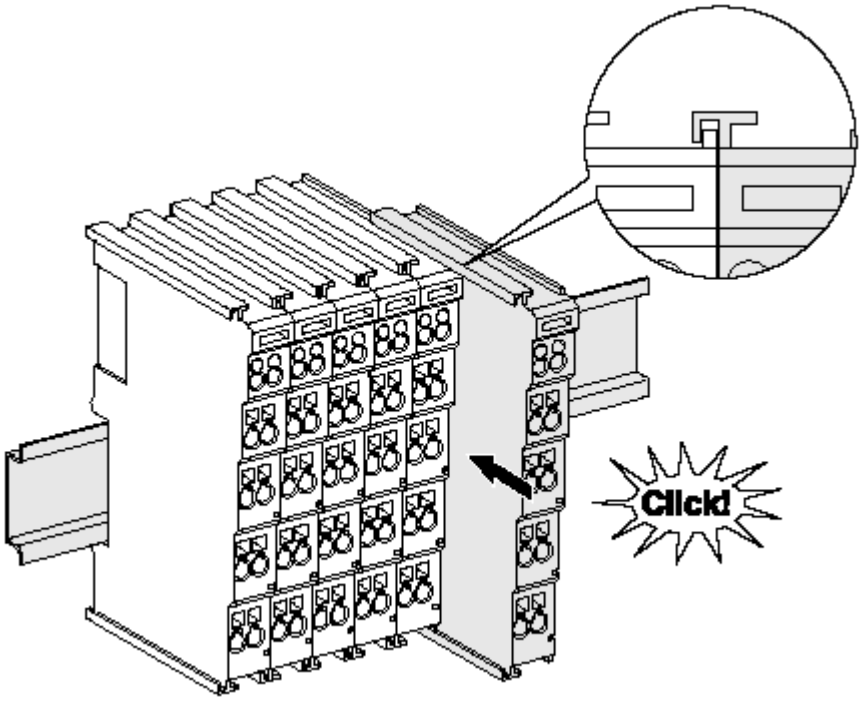


Fig. 6: Attaching on mounting rail

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

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

Fixing of mounting rails

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

NOTICE

Ground the mounting rail!

Ensure that the mounting rail is sufficiently grounded.

Connections within a bus terminal block

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

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

Power Contacts

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

Power contact ⏚

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

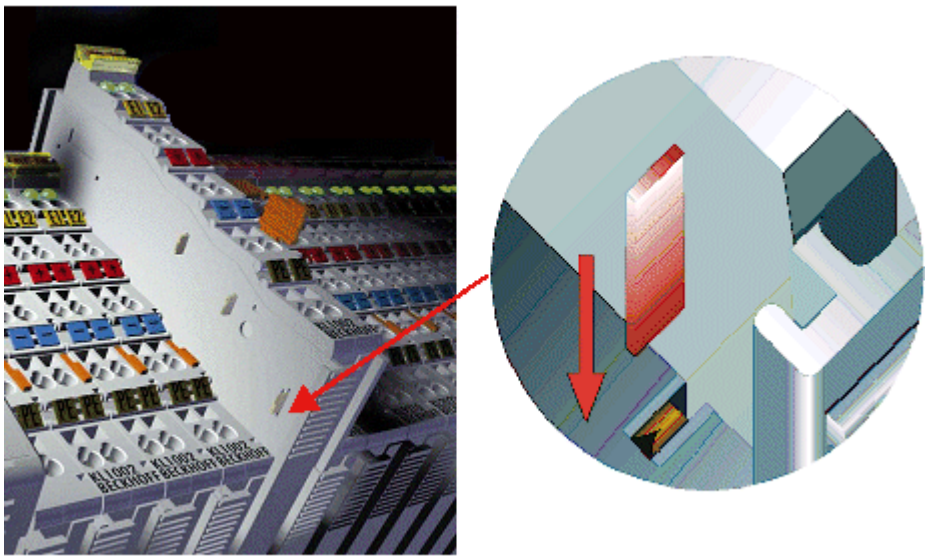


Fig. 7: Power contact on left side

⚠ WARNING

Risk of electric shock!

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

NOTICE

Possible damage of the device

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

Disassembly

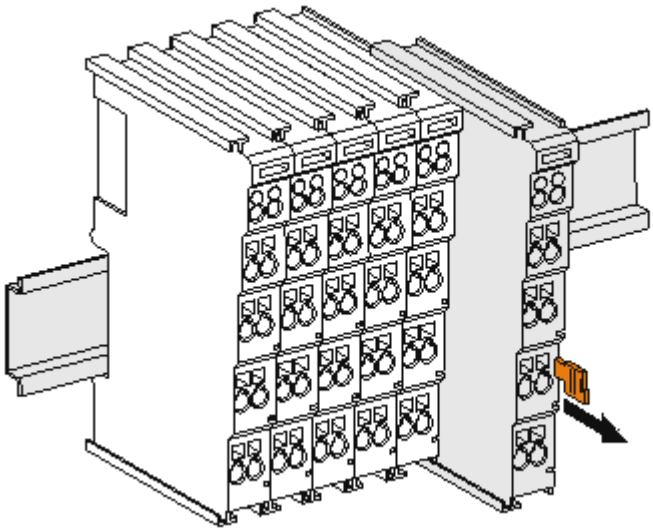


Fig. 8: Disassembling of terminal

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.

3.5 Installation instructions for enhanced mechanical load capacity

⚠ WARNING

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

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

Additional checks

The terminals have undergone the following additional tests:

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

Additional installation instructions and notes

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

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

3.6 Connection

3.6.1 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)

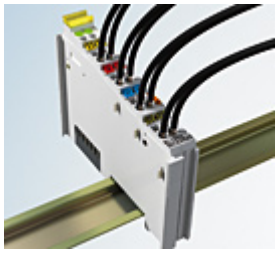


Fig. 9: Standard wiring

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

Pluggable wiring (ESxxxx / KSxxxx)

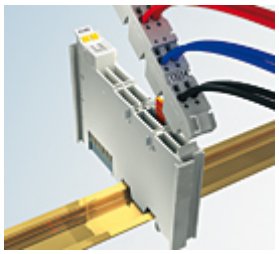


Fig. 10: Pluggable wiring

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

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

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

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

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

High Density Terminals (HD Terminals)

Fig. 11: High Density Terminals

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

Wiring HD Terminals

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

Ultrasonically compacted (ultrasonically welded) strands

Ultrasonically compacted (ultrasonically welded) strands

Ultrasonically compacted (ultrasonically welded) strands can also be connected to the standard and high-density terminals. In this case, please note the tables concerning the wire-size width [▶ 31]!

3.6.2 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

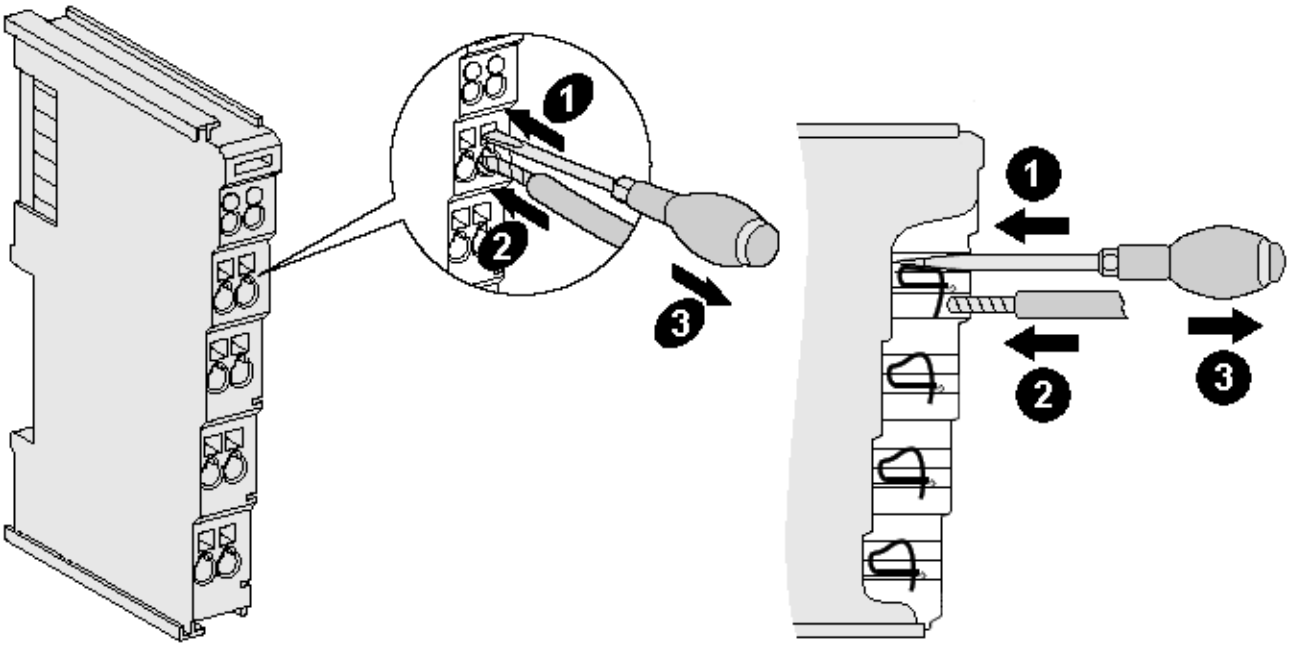


Fig. 12: Connecting a cable on a terminal point

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

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

See the following table for the suitable wire size width:

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

High Density Terminals (HD Terminals [▶ 29]) with 16/32 terminal points

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

Terminal housing	High Density Housing
Wire size width (single core wires)	0.08 ... 1.5 mm^2
Wire size width (fine-wire conductors)	0.25 ... 1.5 mm^2
Wire size width (conductors with a wire end sleeve)	0.14 ... 0.75 mm^2
Wire size width (ultrasonically compacted [ultrasonically welded] strands)	only 1.5 mm^2 (see notice [▶ 29])
Wire stripping length	8 ... 9 mm

3.6.3 Shielding

Shielding

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

3.7 Note - power supply

⚠ WARNING

Power supply from SELV / PELV power supply unit!

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

Notes:

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

3.8 Prescribed installation position

NOTICE

Constraints regarding installation position and operating temperature range

When installing the terminals ensure that an adequate spacing is maintained between other components above and below the terminal in order to guarantee adequate ventilation!

Prescribed installation position

The prescribed 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.

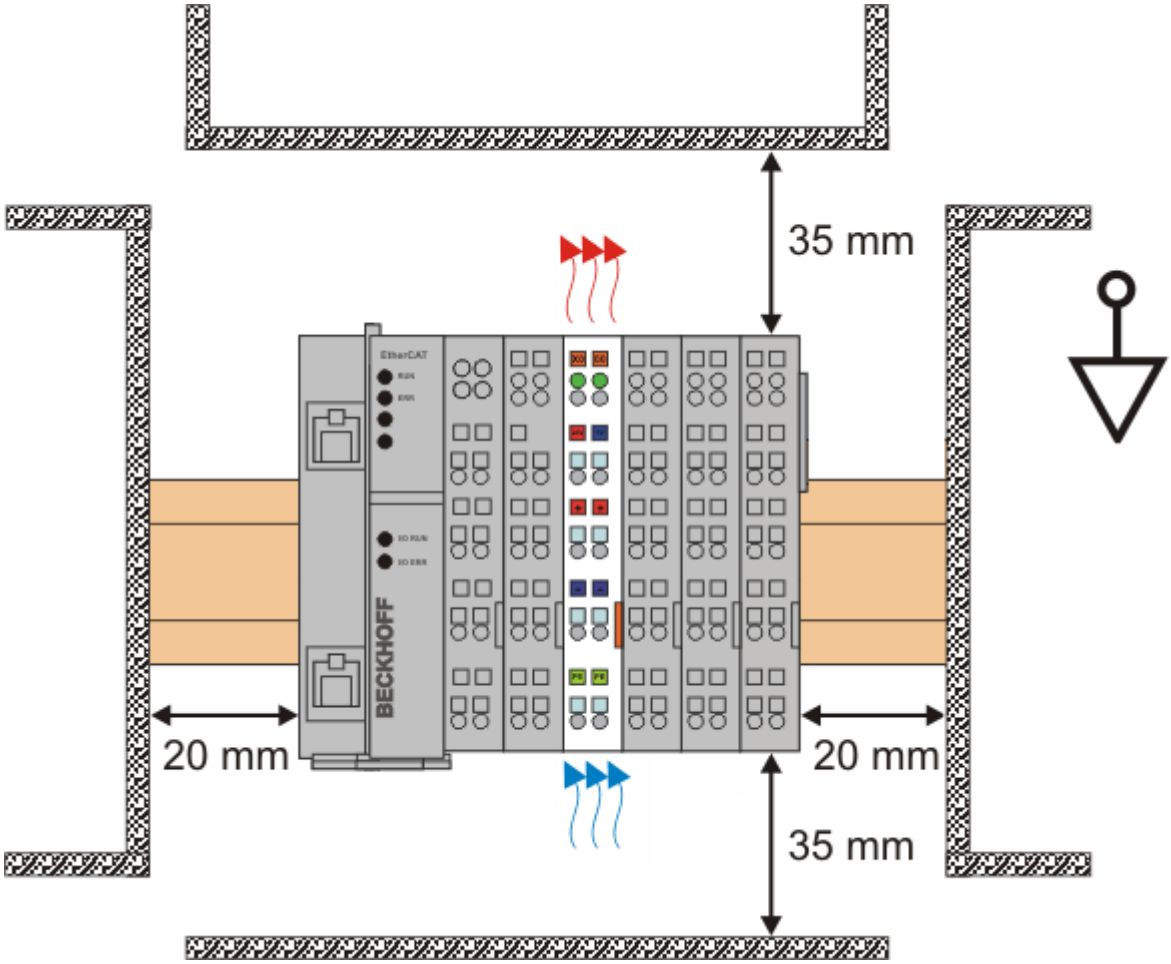


Fig. 13: Recommended minimum distances for standard installation position

Compliance with the distances shown in Fig. *Recommended distances for standard installation position* is strongly recommended.

3.9 Positioning of passive Terminals

Hint for positioning of passive terminals in the bus terminal block

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

Examples for positioning of passive terminals (highlighted)

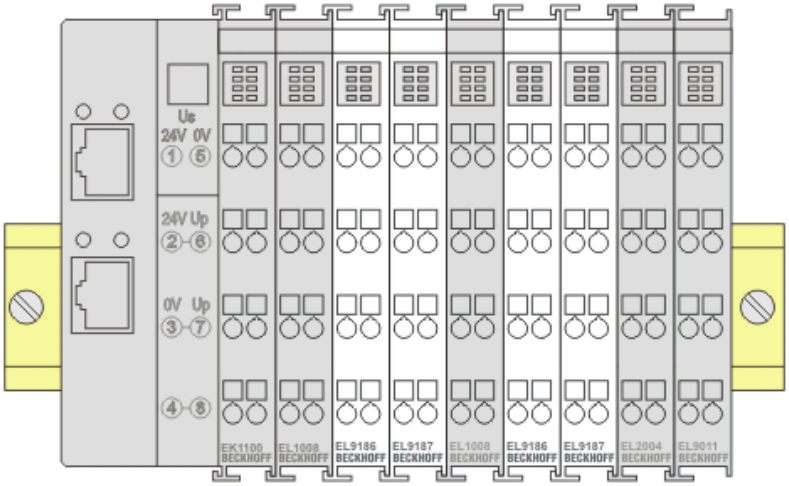


Fig. 14: Correct positioning

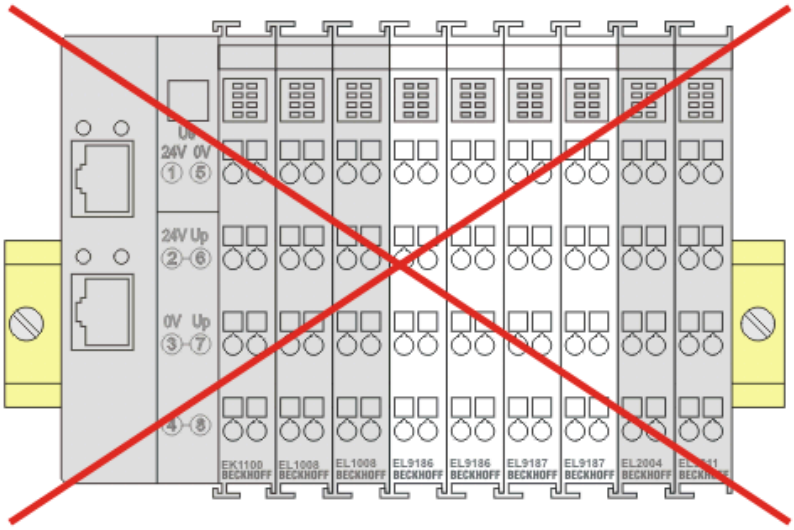


Fig. 15: Incorrect positioning

3.10 LEDs and connection

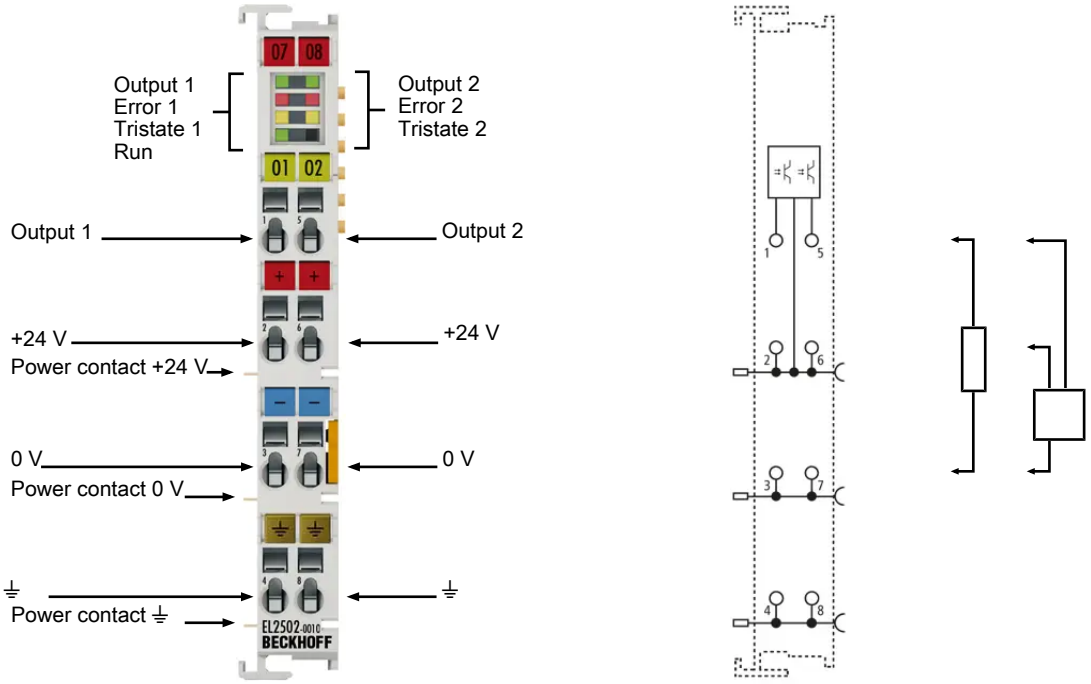


Fig. 16: EL2502-0010

Connection

Terminal point		Description
Name	No.	
Output 1	1	Output 1
+24 V	2	+24 V (internally connected to terminal point 6 and positive power contact)
0 V	3	0 V (internally connected to terminal point 7 and negative power contact)
⏚	4	⏚ connection
Output 2	5	Output 2
+24 V	6	+24 V (internally connected to terminal point 2 and positive power contact)
0 V	7	0 V (internally connected to terminal point 3 and negative power contact)
⏚	8	⏚ connection

LEDs

LED	Color	Meaning	
Output 1 Output 2	green	Signal at the corresponding output	
Tristate 1 Tristate 2	yellow	The corresponding output is in Tristate [▶ 126]	
Error 1 Error 2	red	There is an error on the corresponding channel	
Run	green	This LED indicates the terminal's operating state:	
		off	State of the EtherCAT State Machine: **INIT** = initialization of the terminal
		flashing uniformly	State of the EtherCAT State Machine: **PREOP** = function for mailbox communication and different default settings set
		flashing slowly	State of the EtherCAT State Machine: **SAFEOP** = verification of the Sync Manager [▶ 90] 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
		flashing rapidly	State of the EtherCAT State Machine: **BOOTSTRAP** = function for Firmware updates [▶ 142] of the terminal

3.11 Disposal

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

4 Commissioning with TwinCAT

4.1 TwinCAT Quick Start

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

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

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

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

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

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

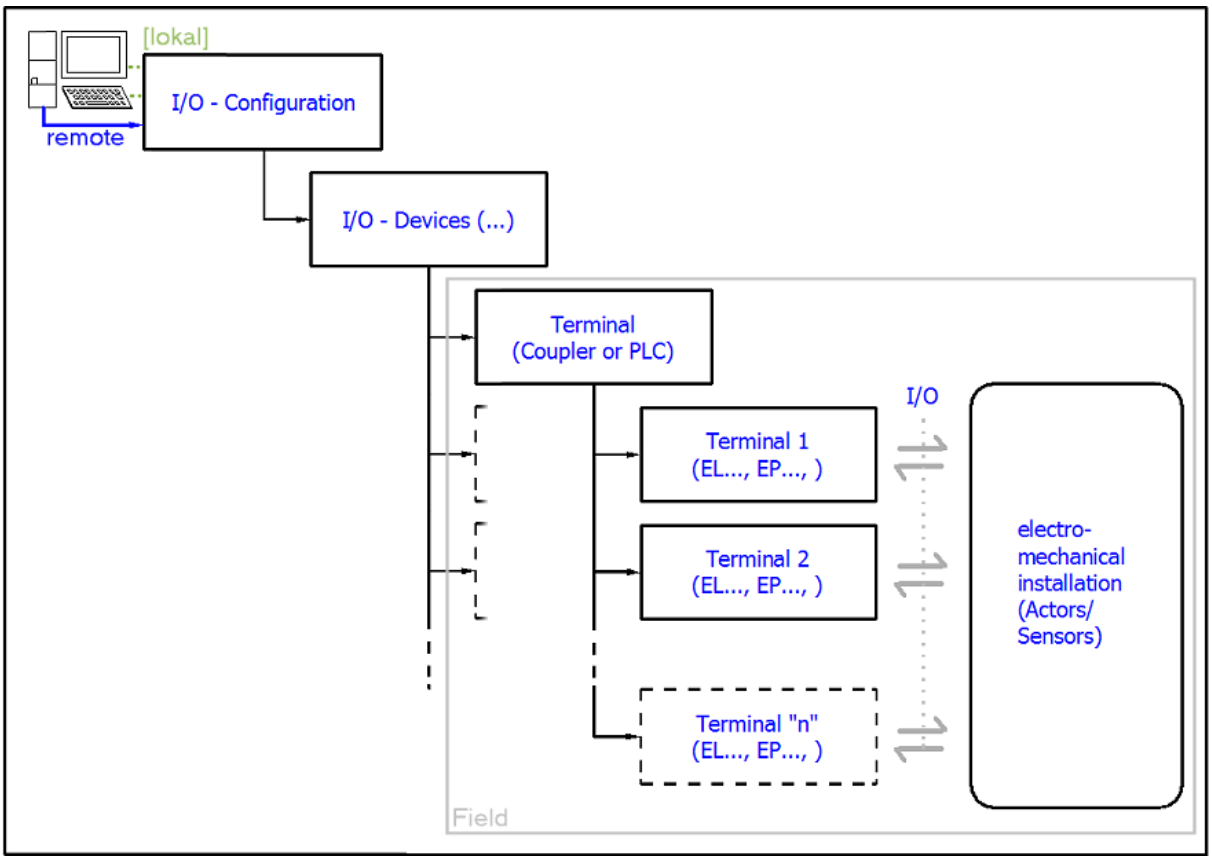


Fig. 17: Relationship between user side (commissioning) and installation

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

Example configuration (actual configuration)

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

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

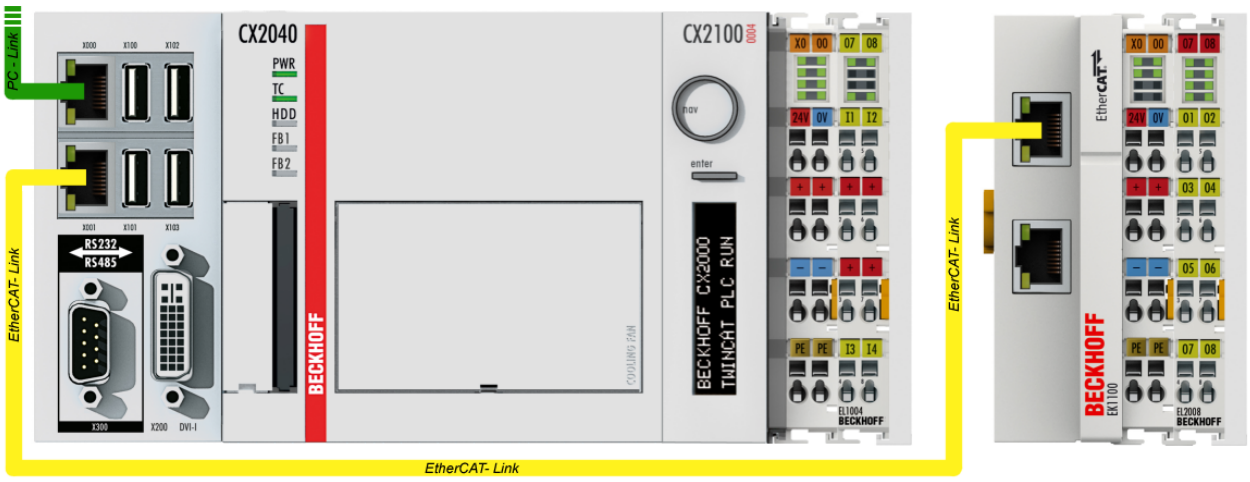


Fig. 18: Control configuration with Embedded PC, input (EL1004) and output (EL2008)

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.

4.1.1 TwinCAT 2

Startup

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

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

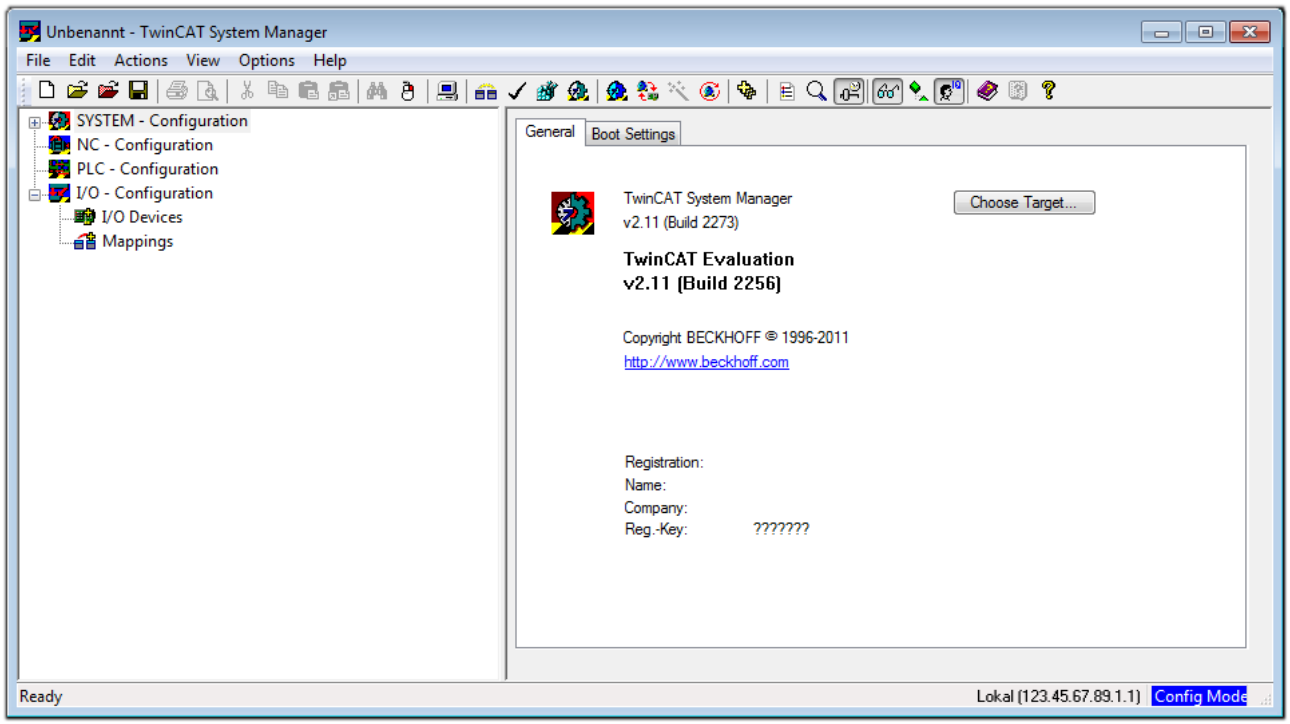


Fig. 19: Initial TwinCAT 2 user interface

Generally, TwinCAT can be used in local or remote mode. Once the TwinCAT system, including the user interface (standard) is installed on the respective PLC, TwinCAT can be used in local mode and thus the next step is “Insert Device [▶ 42]”.

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

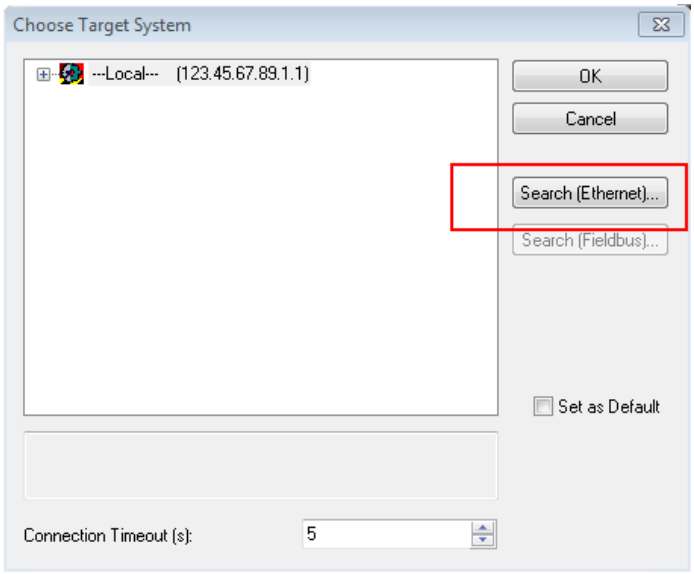


Fig. 20: Selection of the target system

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

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

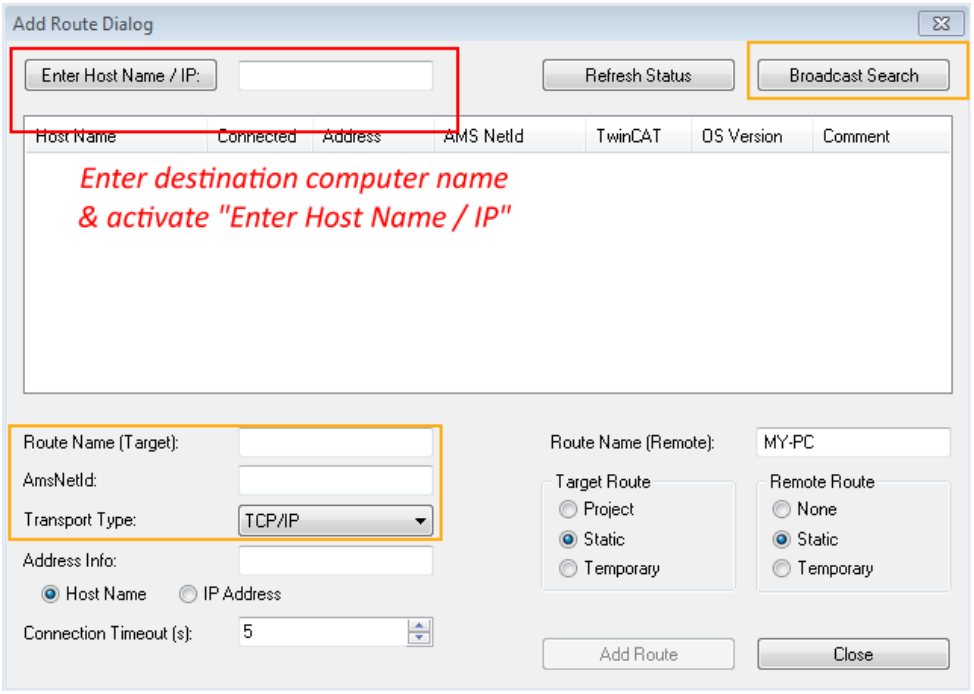
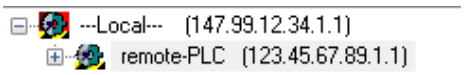


Fig. 21: specify the PLC for access by the TwinCAT System Manager: selection of the target system

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



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

Adding devices

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

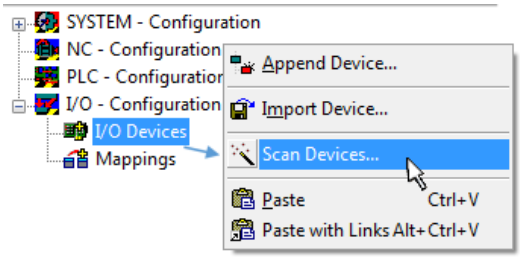


Fig. 22: Select “Scan Devices...”

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

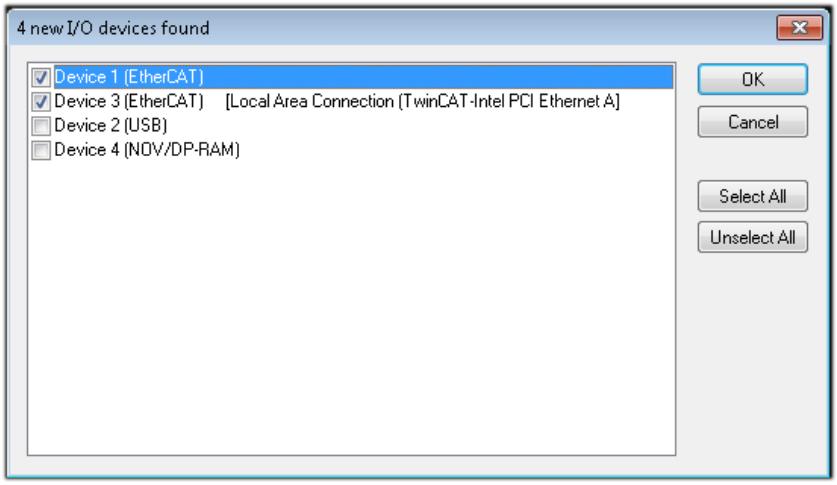


Fig. 23: Automatic detection of I/O devices: selection of the devices to be integrated

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

Based on the example configuration [▶ 38] described at the beginning of this section, the result is as follows:

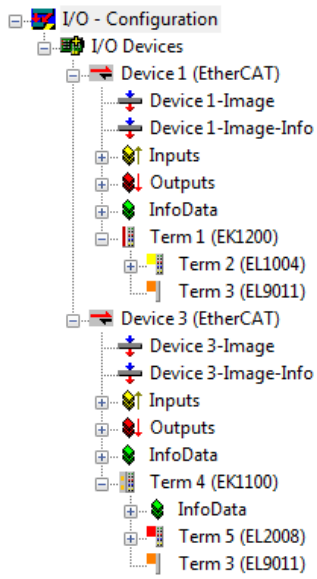


Fig. 24: Mapping of the configuration in the TwinCAT 2 System Manager

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

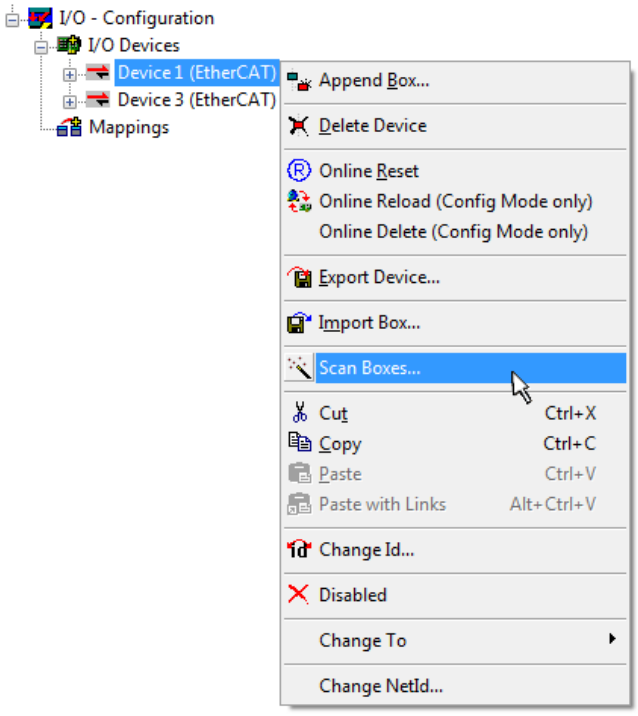


Fig. 25: Reading of individual terminals connected to a device

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

Programming and integrating the PLC

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

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

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

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

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

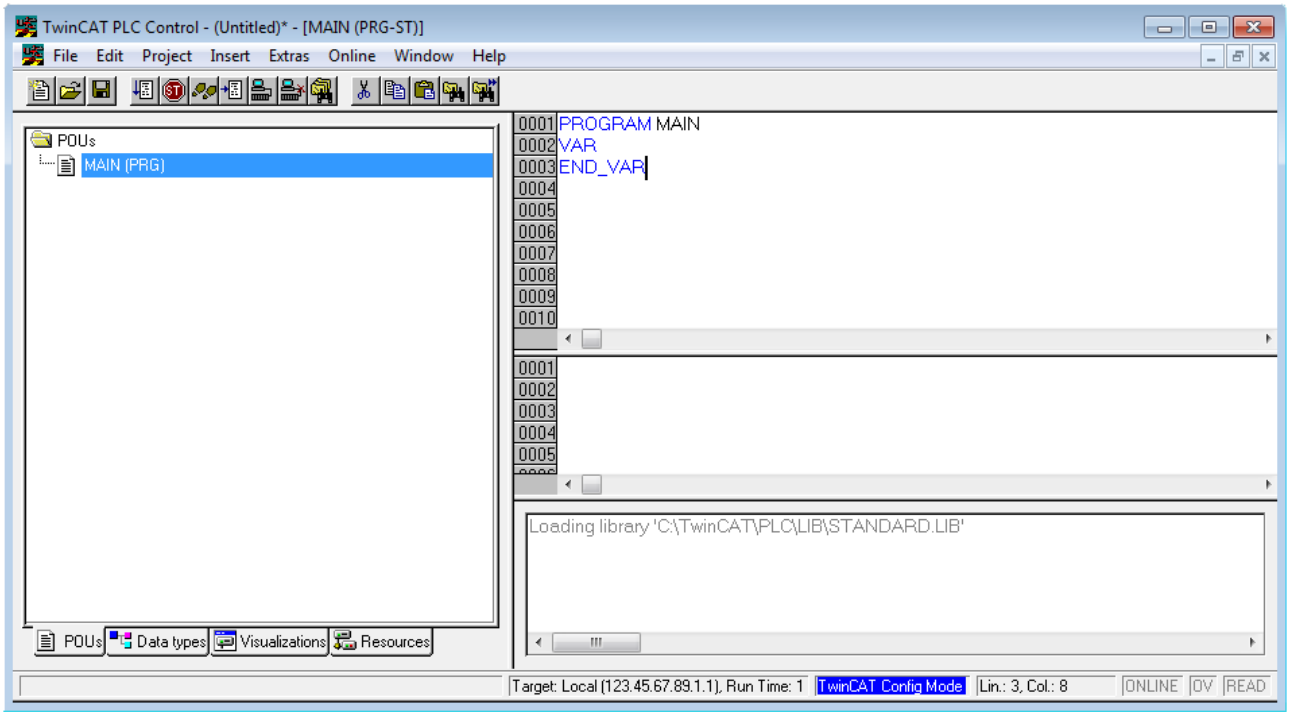


Fig. 26: TwinCAT PLC Control after startup

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

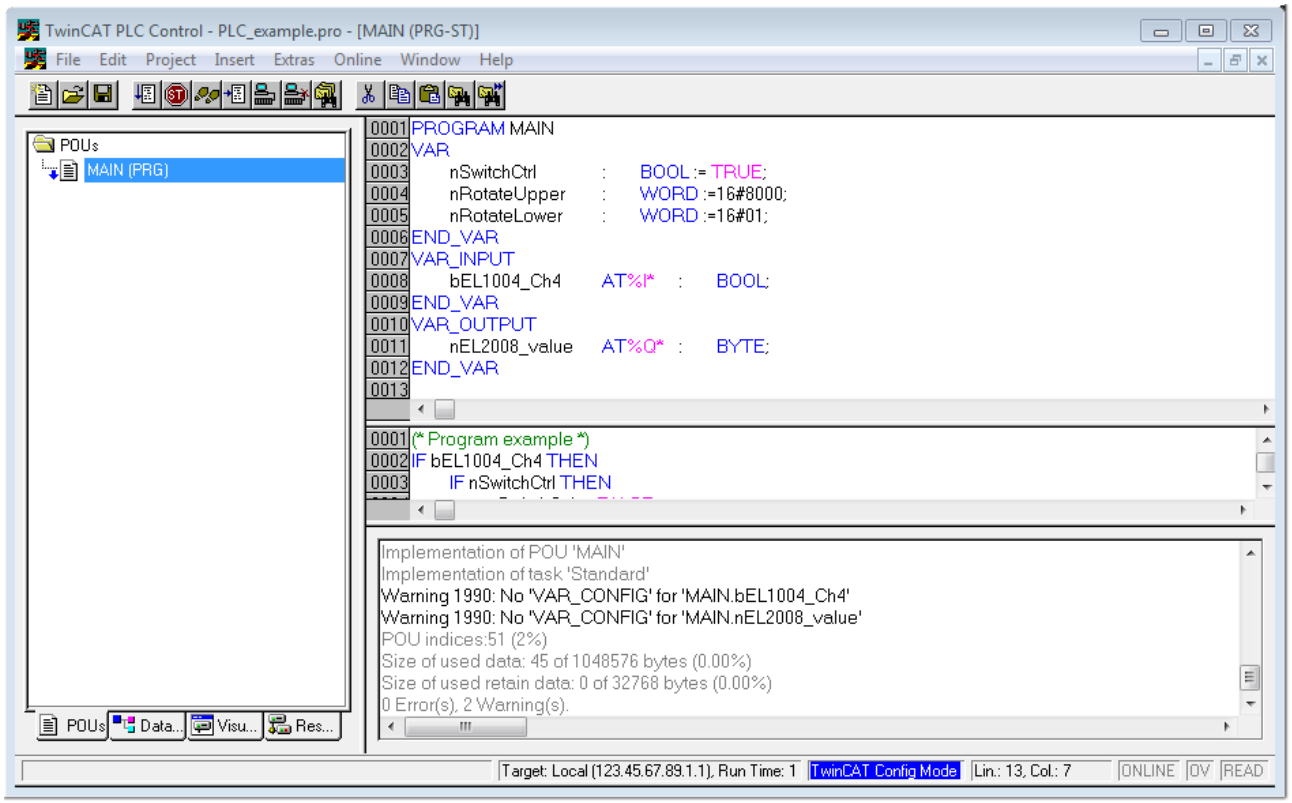


Fig. 27: Example program with variables after a compile process (without variable integration)

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

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

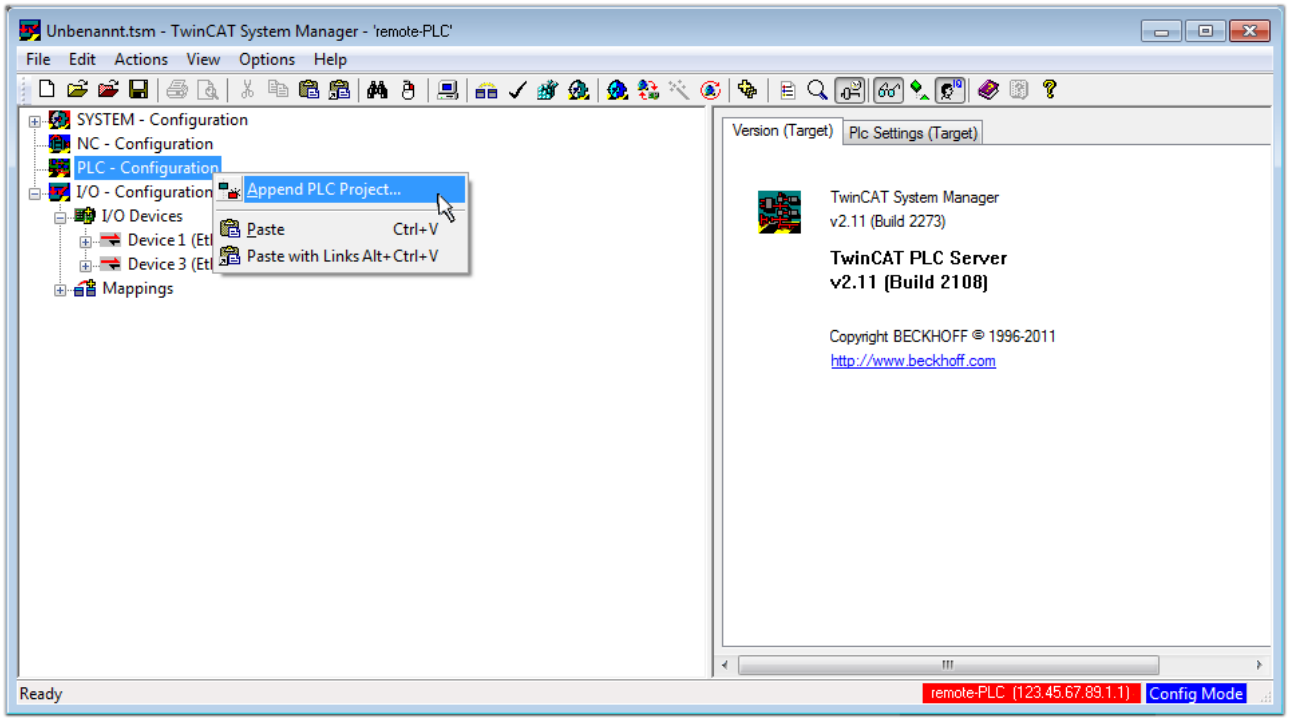


Fig. 28: Appending the TwinCAT PLC Control project

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:

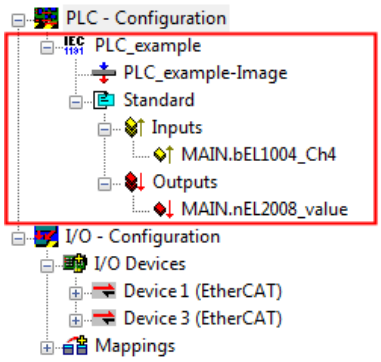


Fig. 29: PLC project integrated in the PLC configuration of the System Manager

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”:

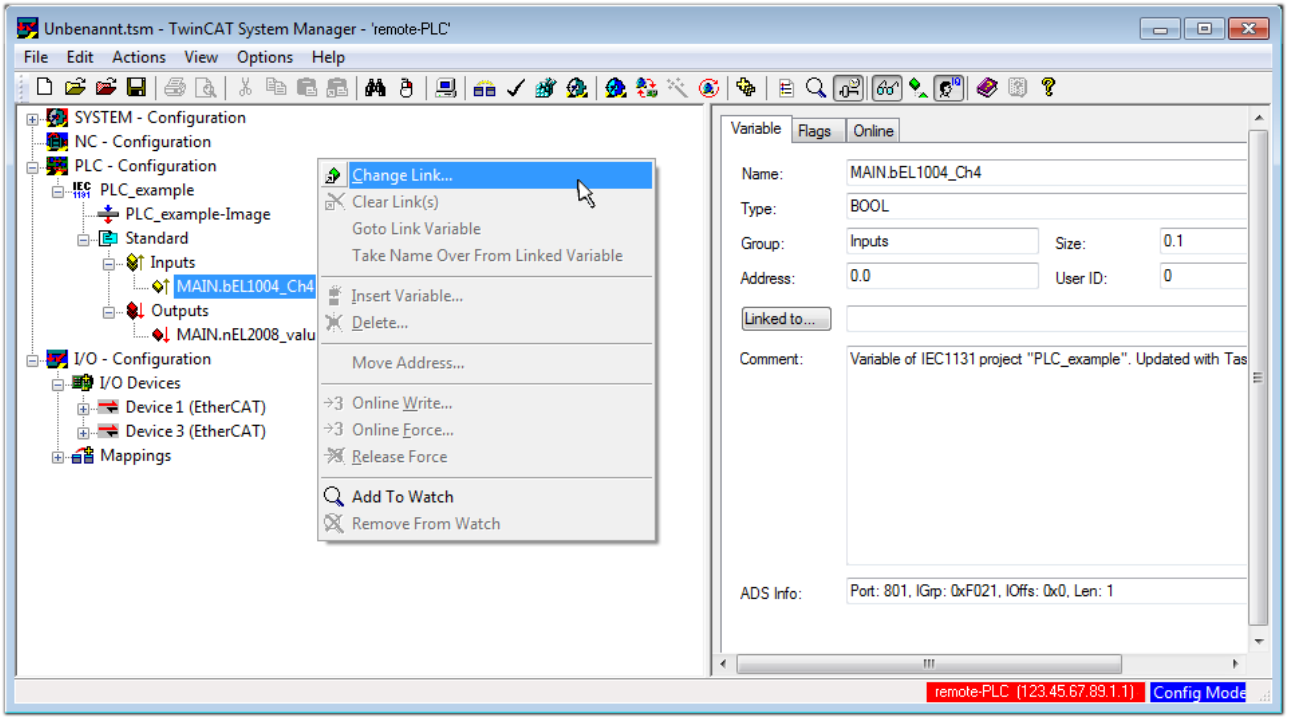


Fig. 30: Creating the links between PLC variables and process objects

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

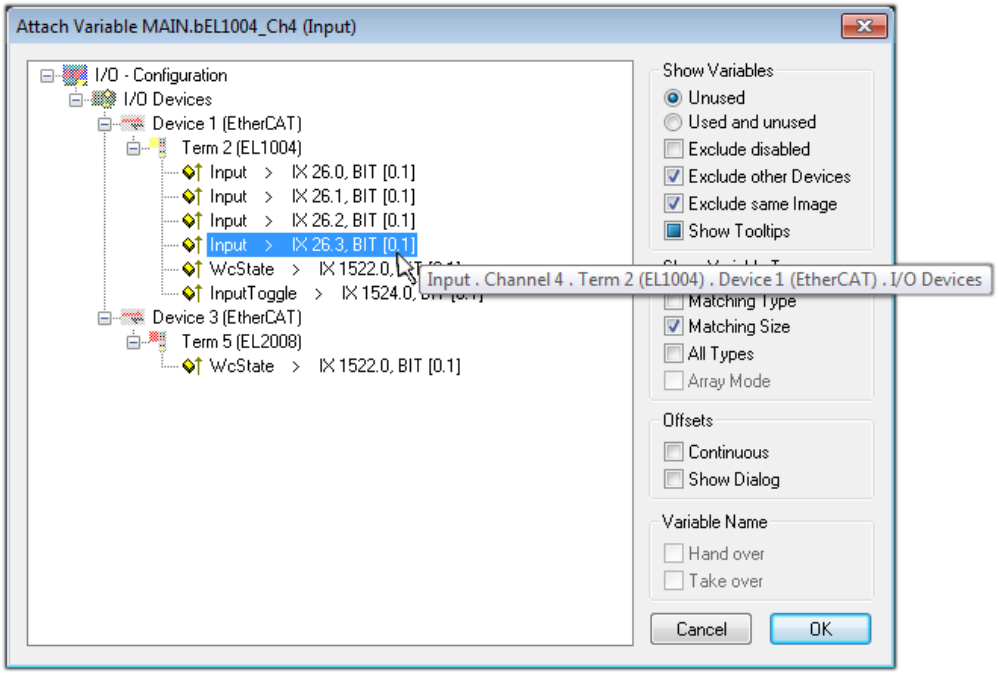


Fig. 31: Selecting BOOL-type PDO

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

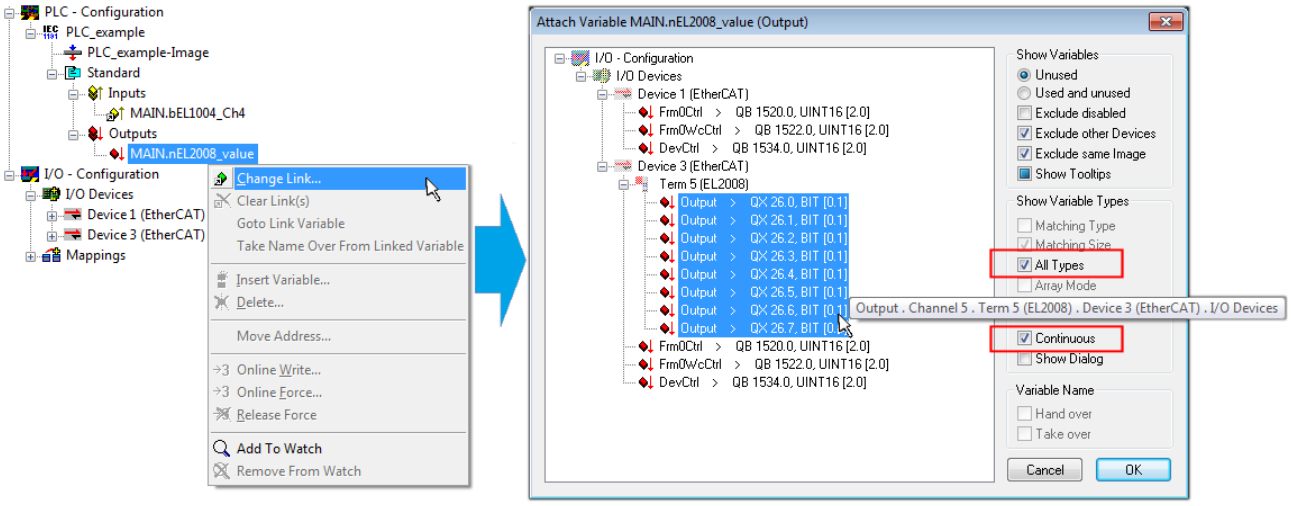


Fig. 32: Selecting several PDOs simultaneously: activate “Continuous” and “All types”

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

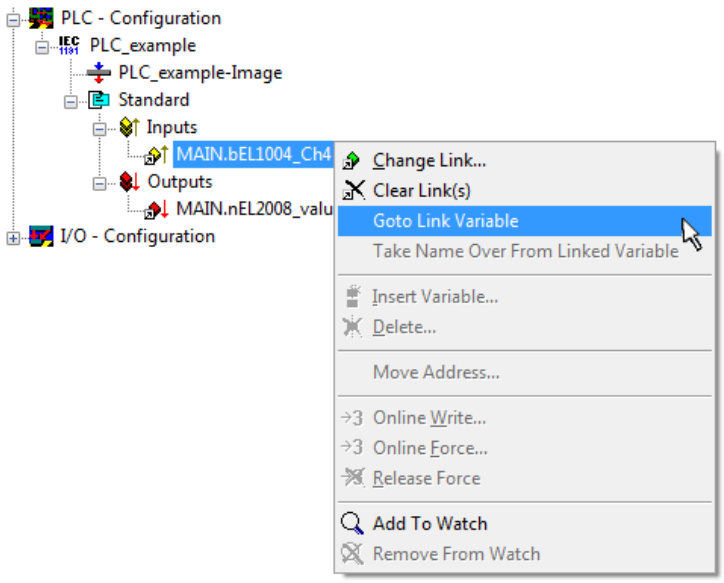
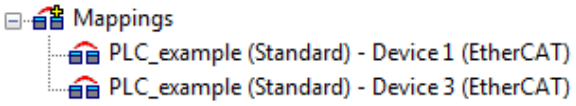


Fig. 33: Application of a “Goto Link Variable”, using “MAIN.bEL1004_Ch4” as an example

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

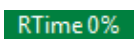
This can be visualized in the configuration:



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

Activation of the configuration

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

A few seconds later, the real-time status 

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

Starting the controller

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

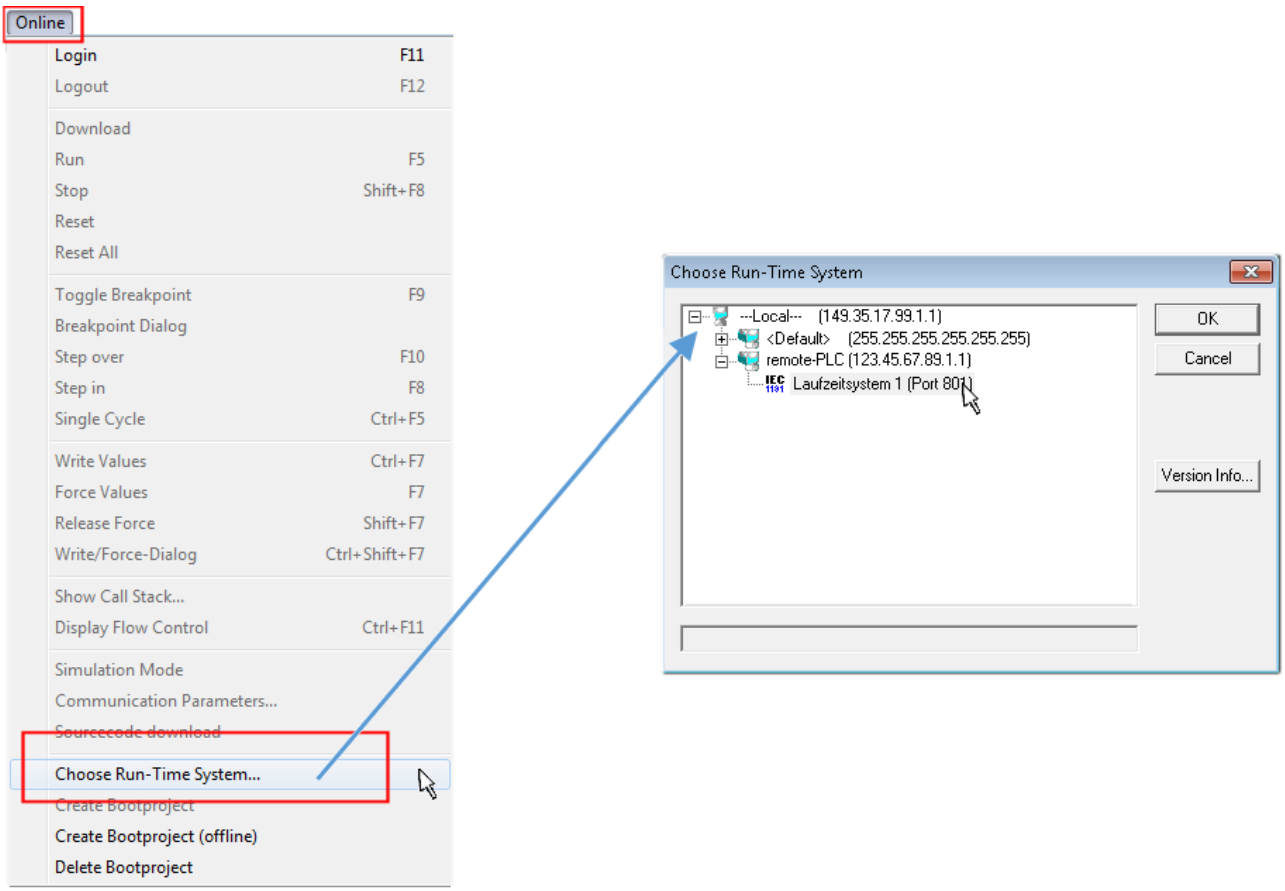


Fig. 34: Choose target system (remote)

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

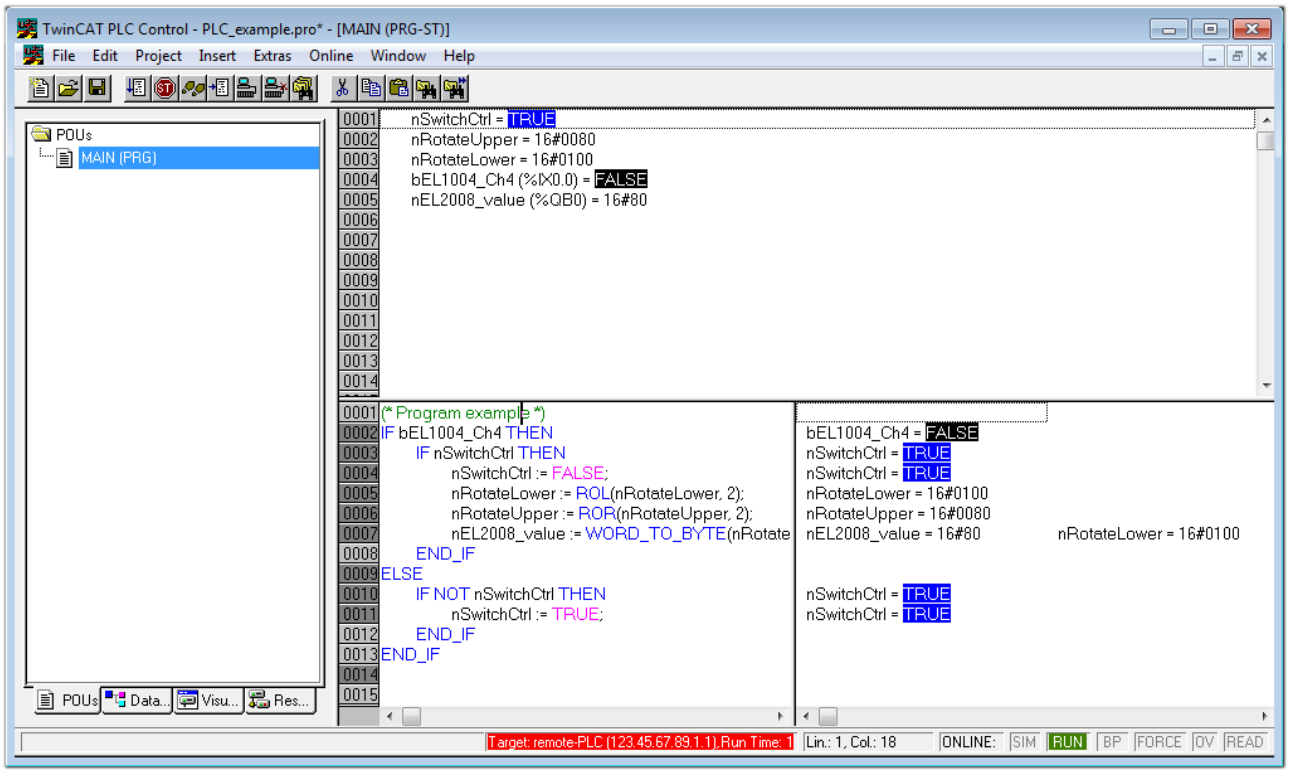


Fig. 35: PLC Control logged in, ready for program startup

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

4.1.2 TwinCAT 3

Startup

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

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

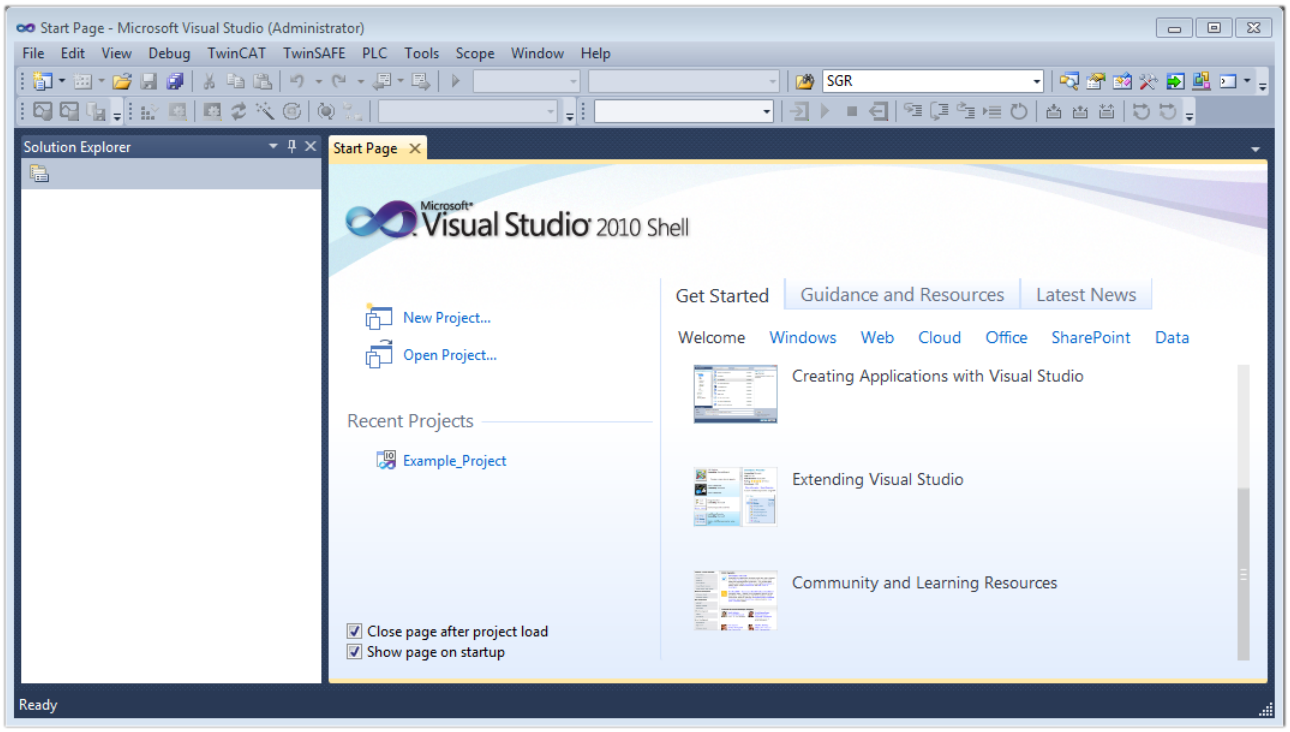


Fig. 36: Initial TwinCAT 3 user interface

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):

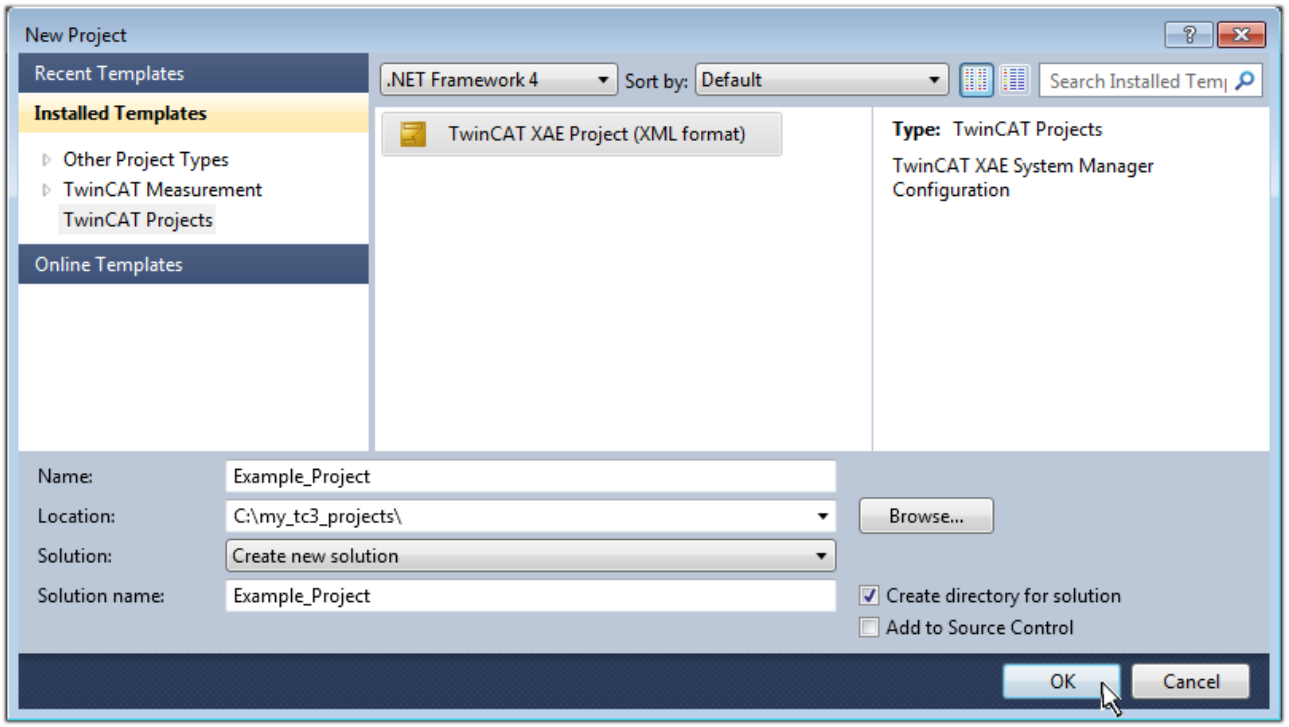


Fig. 37: Create new TwinCAT 3 project

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

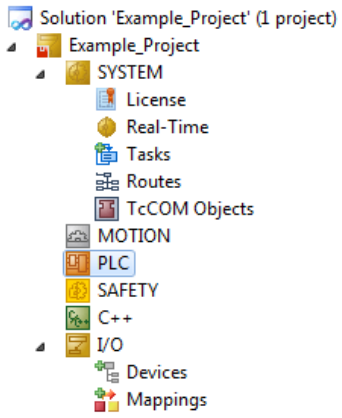
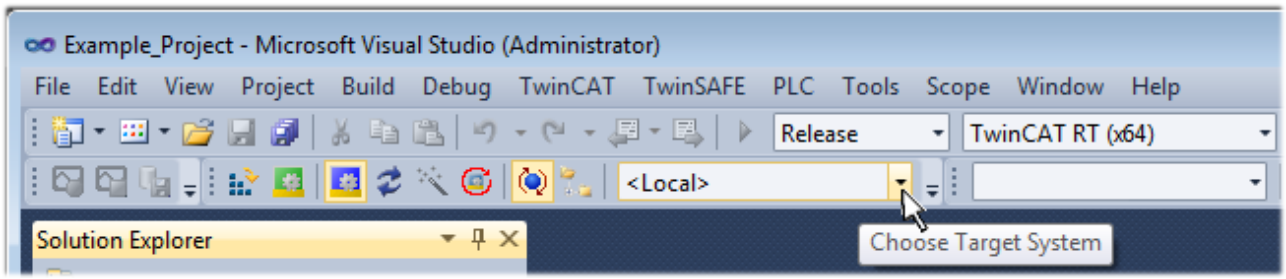


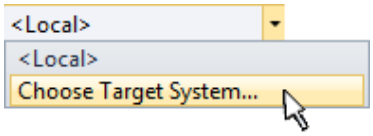
Fig. 38: New TwinCAT 3 project in the project folder explorer

Generally, TwinCAT can be used in local or remote mode. Once the TwinCAT system including the user interface (standard) is installed on the respective PLC (locally), TwinCAT can be used in local mode and the process can be continued with the next step, "Insert Device [▶ 53]".

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



expand the pull-down menu:



and open the following window:

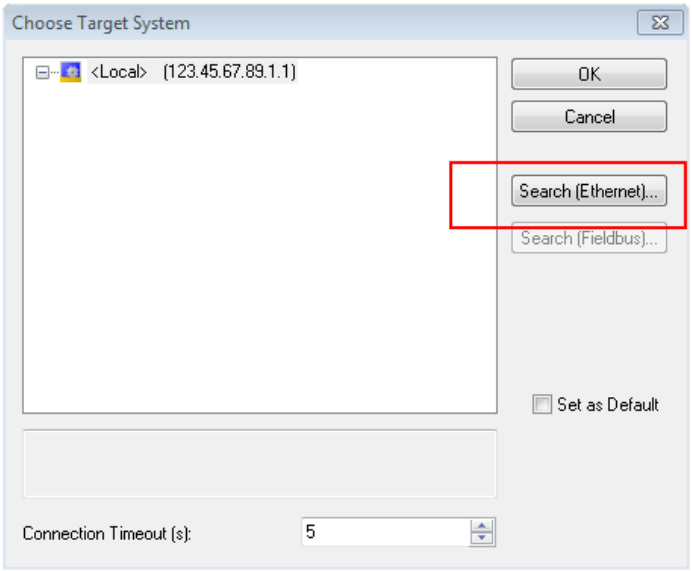


Fig. 39: Selection dialog: Choose the target system

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

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

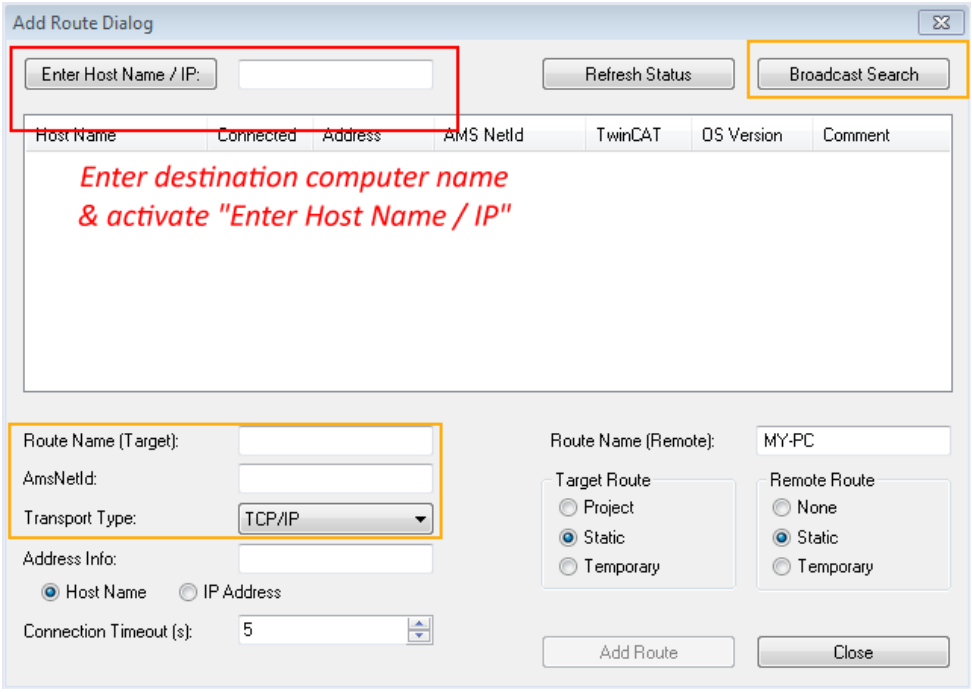
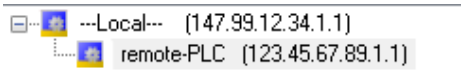


Fig. 40: specify the PLC for access by the TwinCAT System Manager: selection of the target system

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



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

Adding devices

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

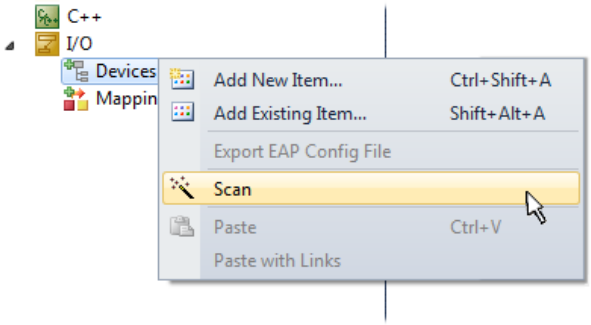


Fig. 41: Select “Scan”

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

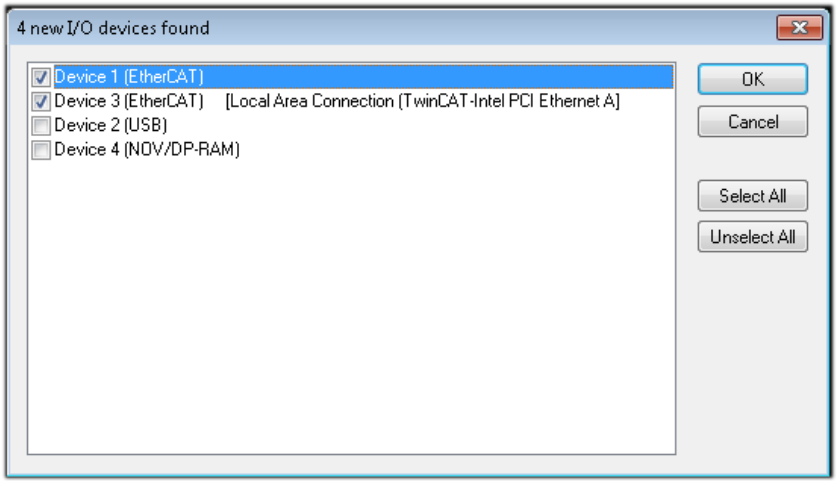


Fig. 42: Automatic detection of I/O devices: selection of the devices to be integrated

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

Based on the example configuration [▶ 38] described at the beginning of this section, the result is as follows:

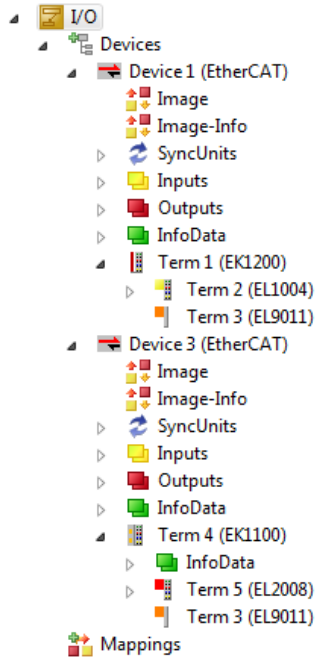


Fig. 43: Mapping of the configuration in VS shell of the TwinCAT 3 environment

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

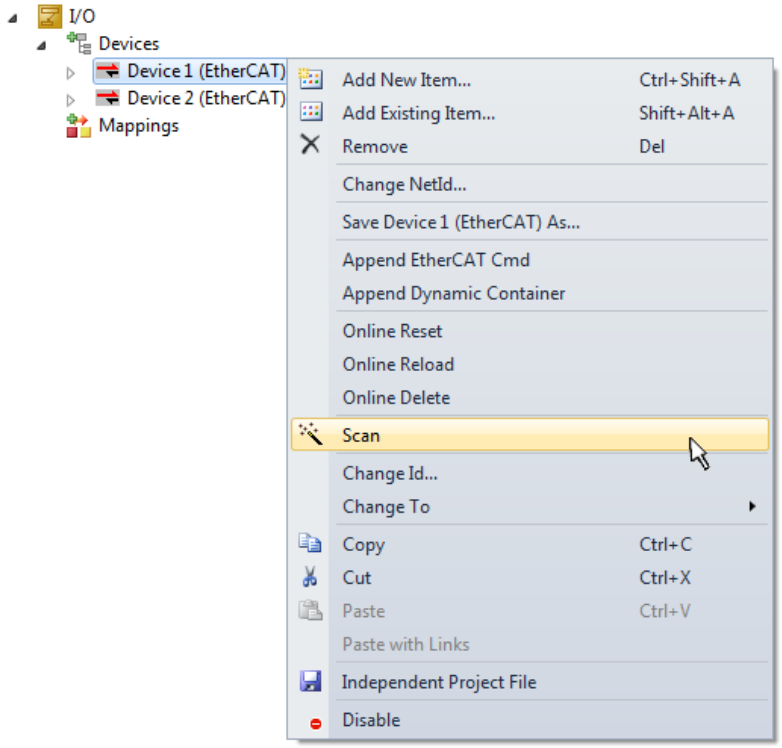


Fig. 44: Reading of individual terminals connected to a device

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

Programming the PLC

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

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

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

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

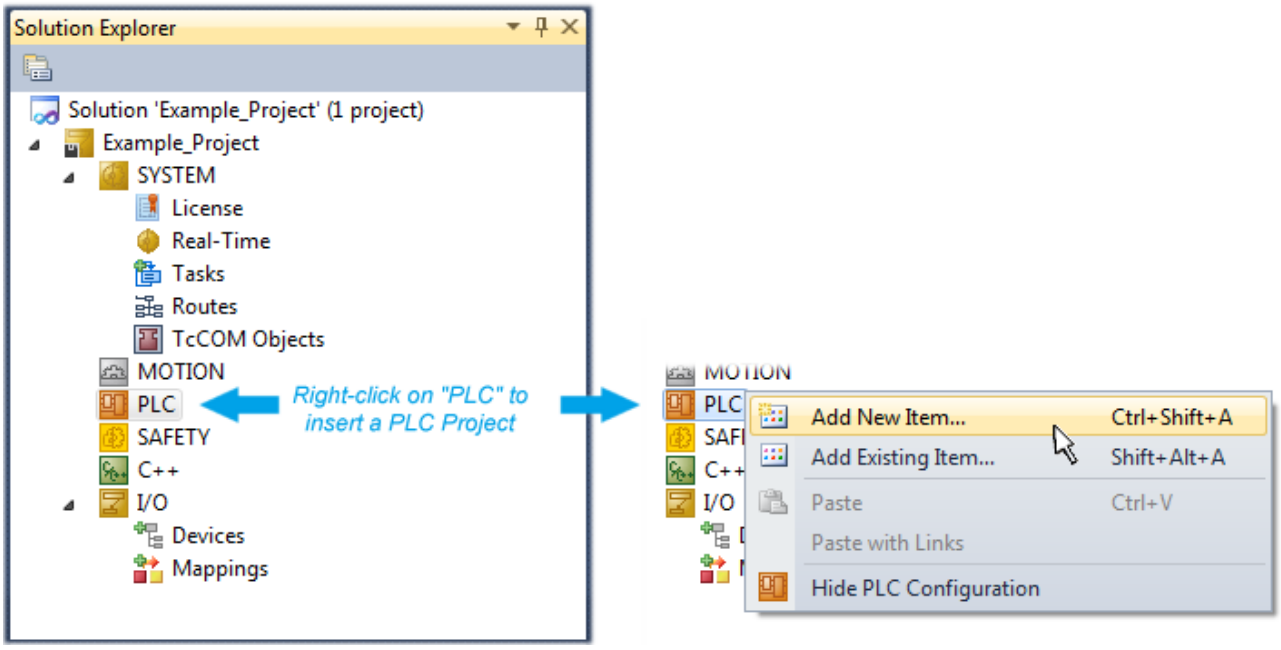


Fig. 45: Adding the programming environment in “PLC”

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

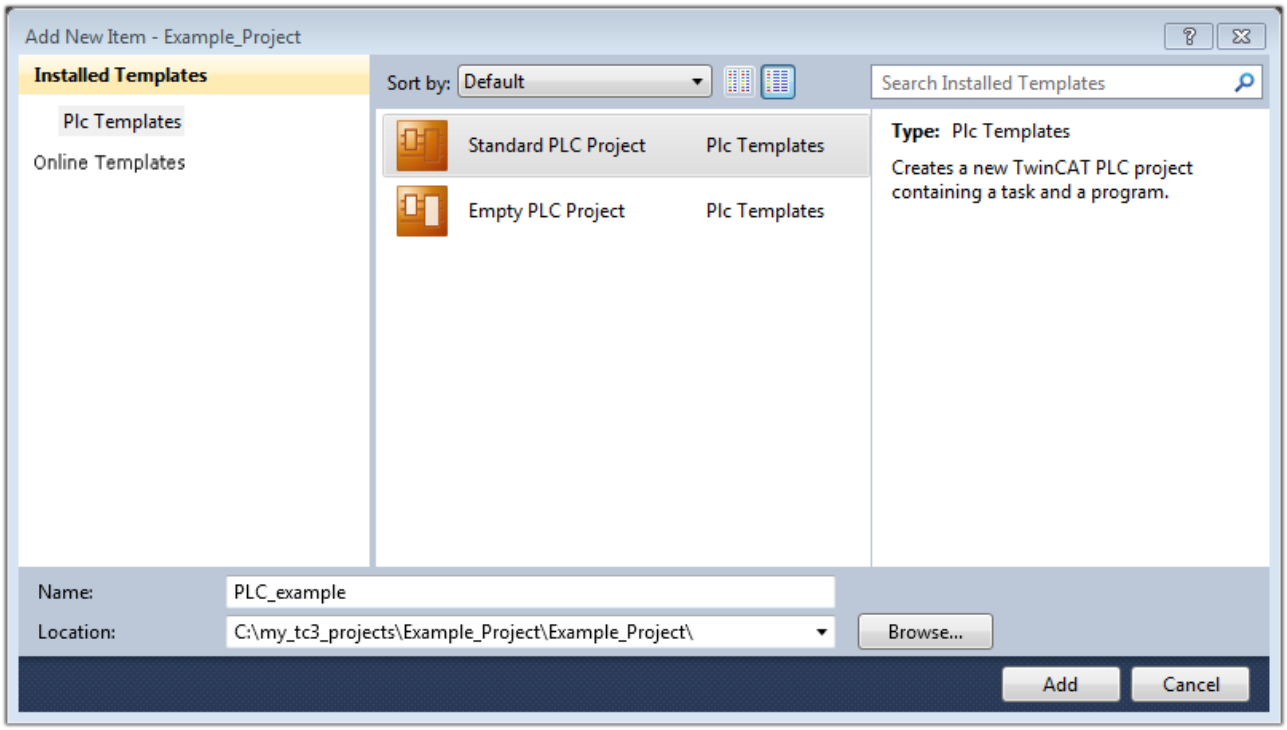


Fig. 46: Specifying the name and directory for the PLC programming environment

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

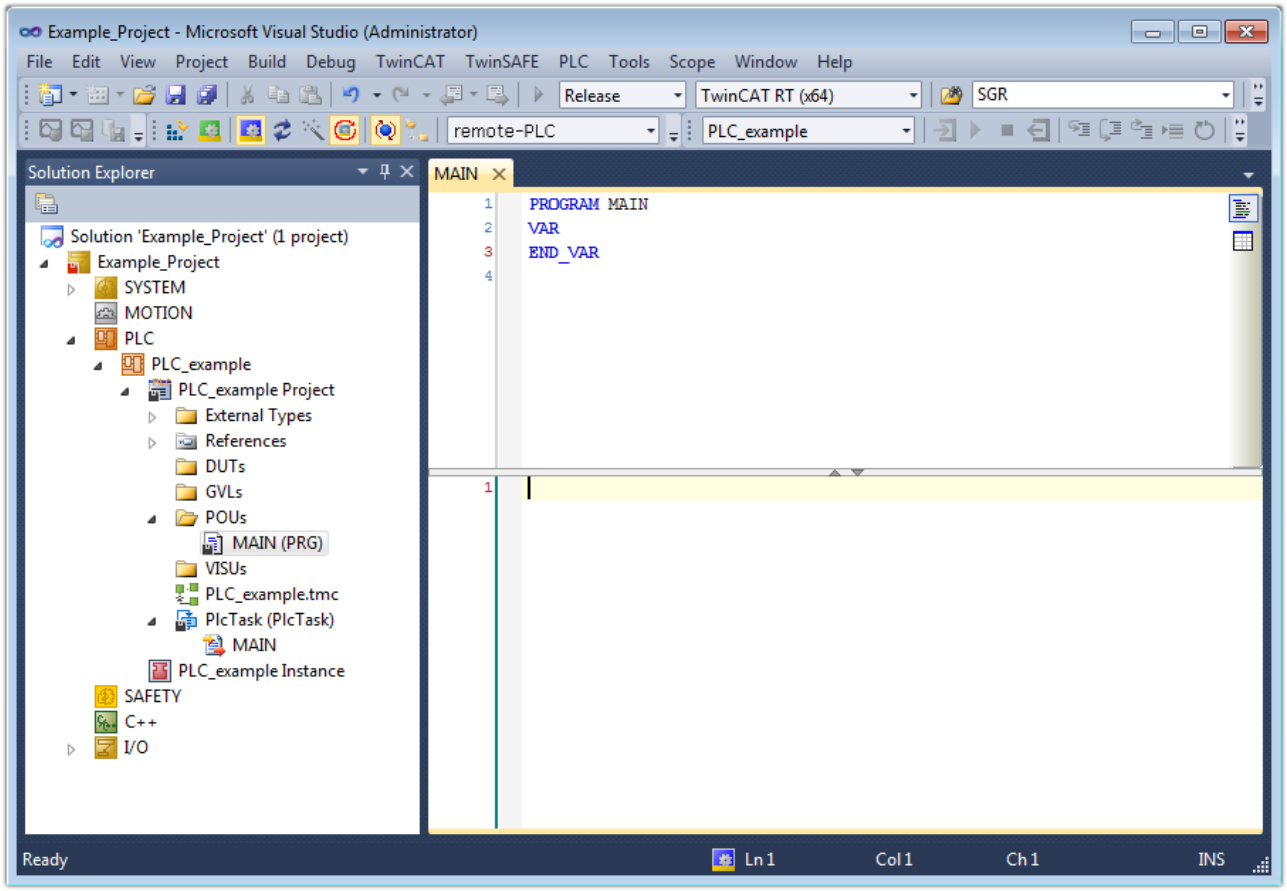


Fig. 47: Initial “Main” program for the standard PLC project

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

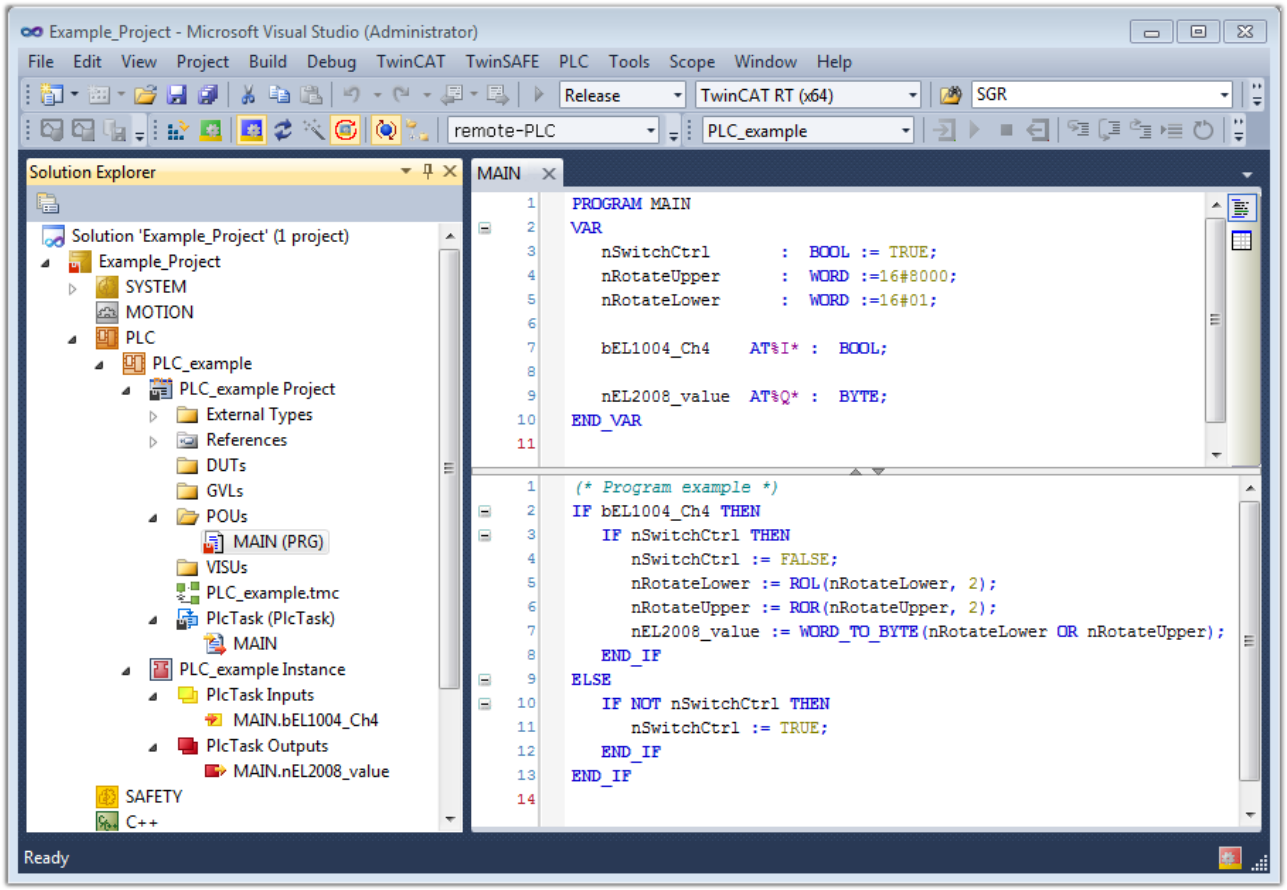


Fig. 48: Example program with variables after a compile process (without variable integration)

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

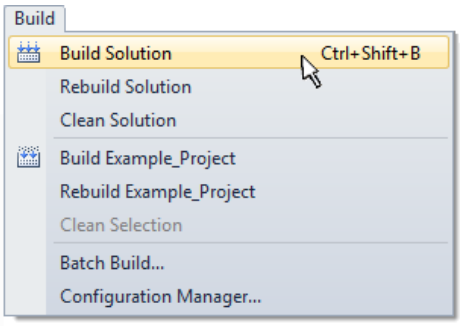
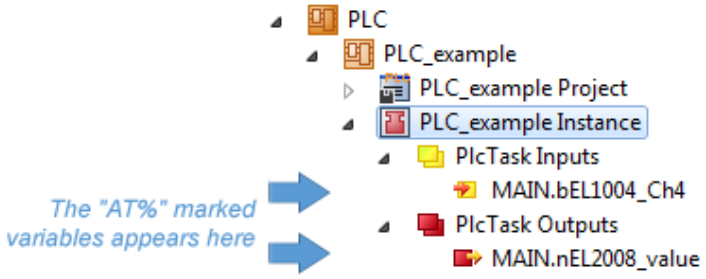


Fig. 49: Start program compilation

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



Assigning variables

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

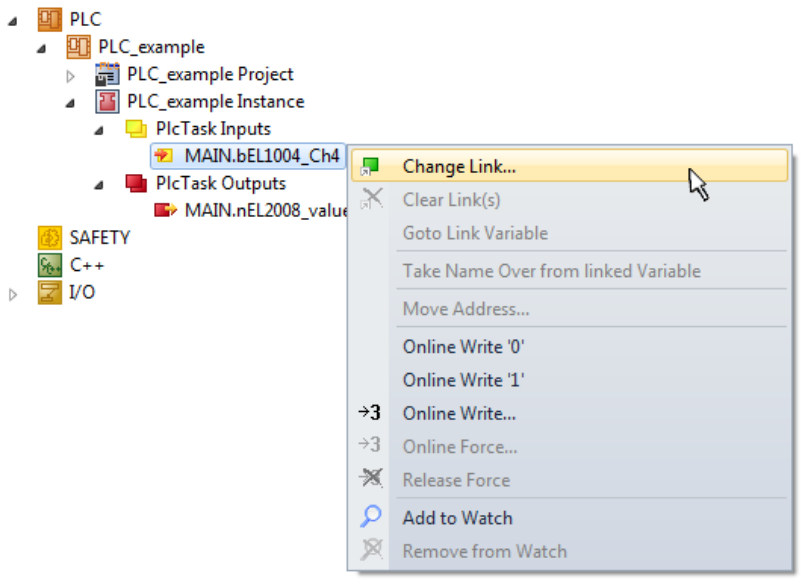


Fig. 50: Creating the links between PLC variables and process objects

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

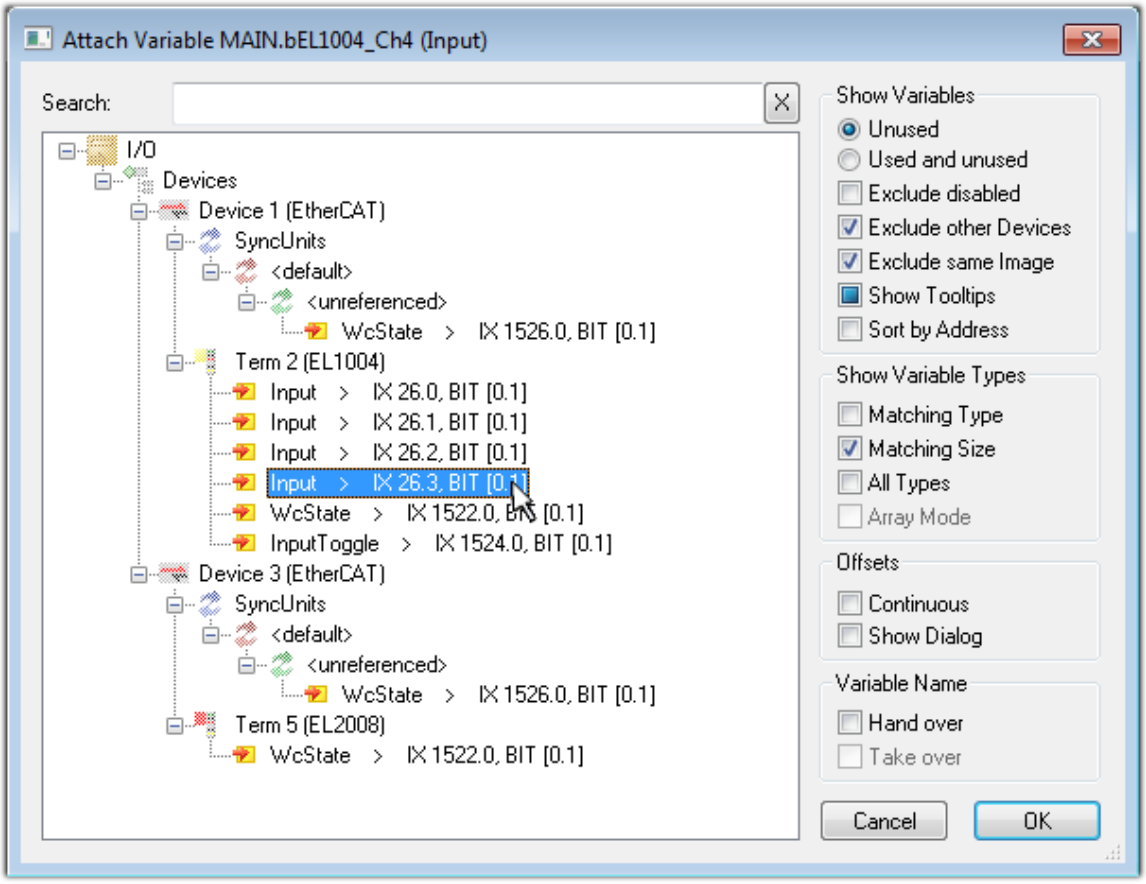


Fig. 51: Selecting BOOL-type PDO

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

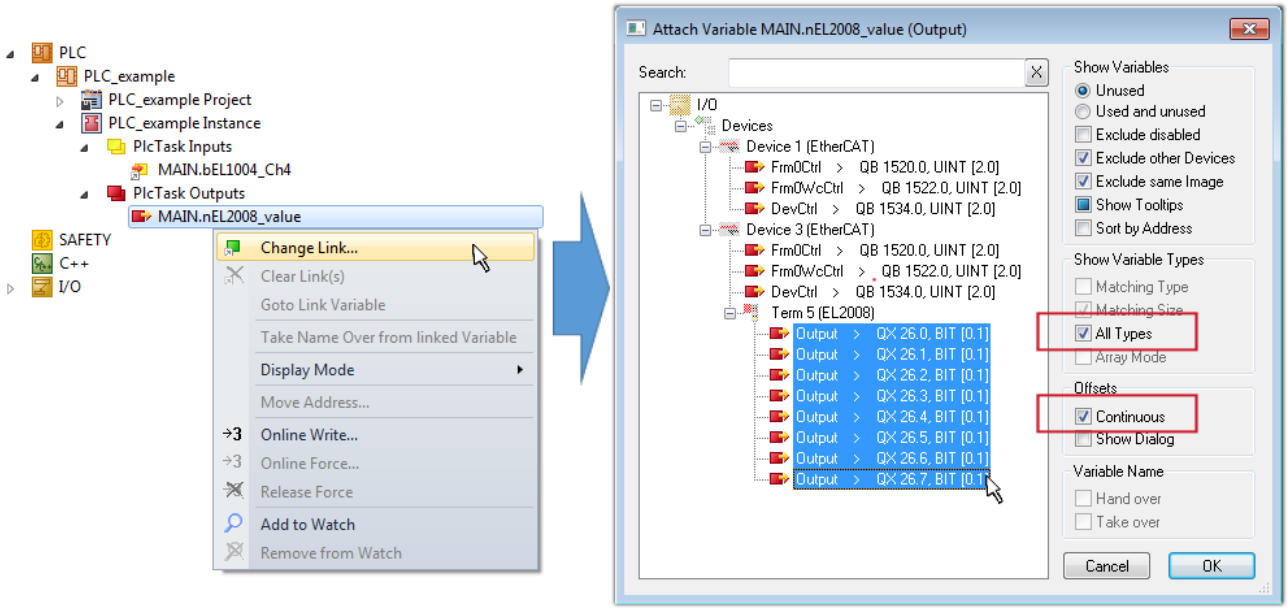


Fig. 52: Selecting several PDOs simultaneously: activate “Continuous” and “All types”

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

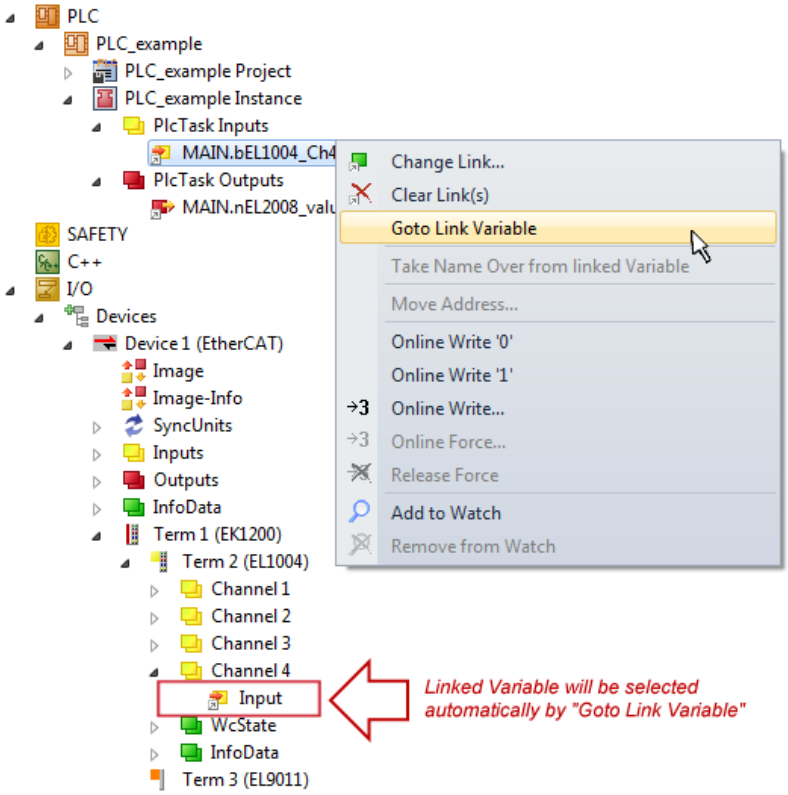


Fig. 53: Application of a “Goto Link Variable”, using “MAIN.bEL1004_Ch4” as an example

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

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

Note on type of variable assignment

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

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

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

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

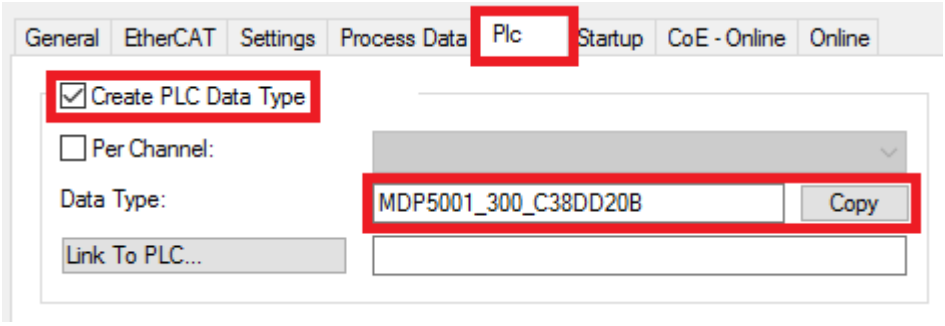


Fig. 54: Creating a PLC data type

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

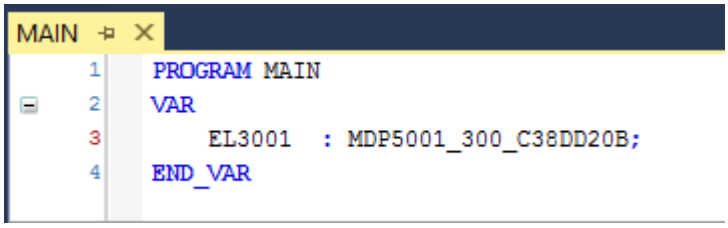


Fig. 55: Instance_of_struct

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

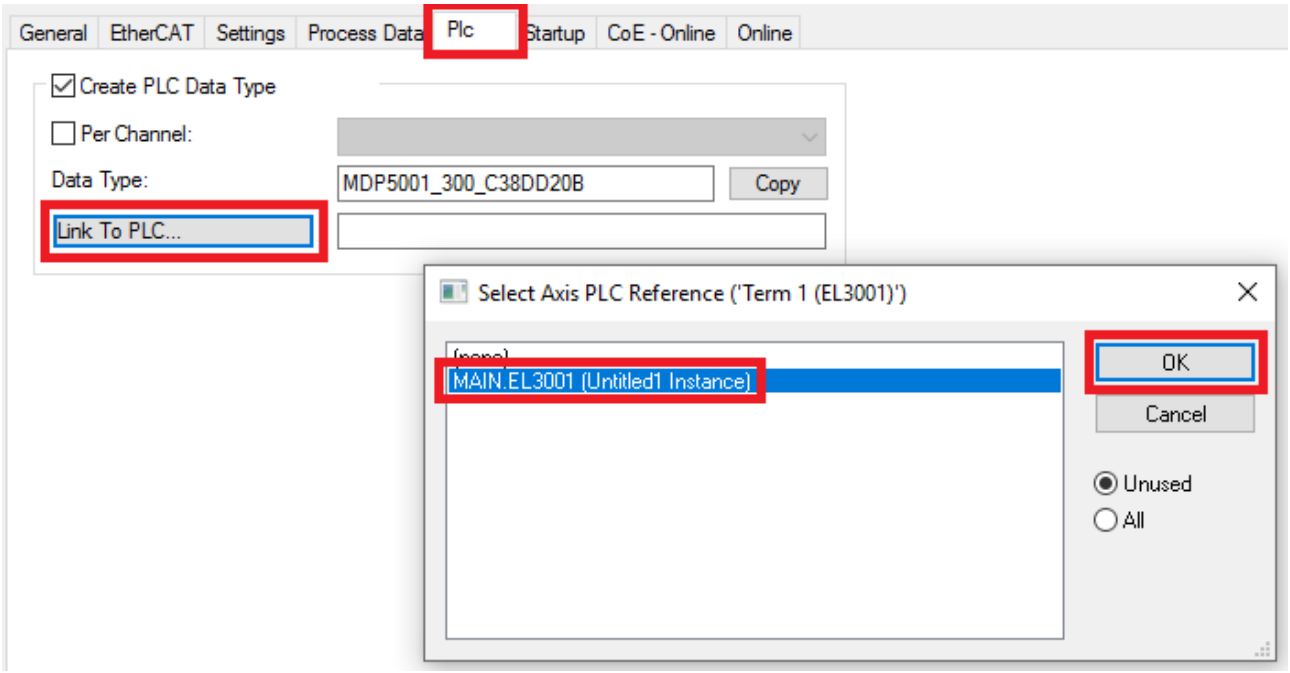


Fig. 56: Linking the structure

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

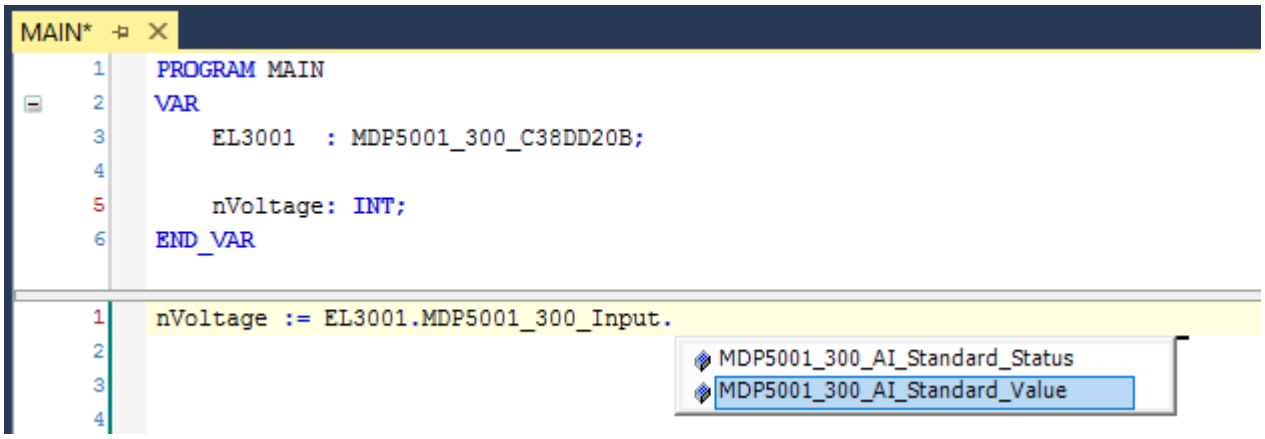
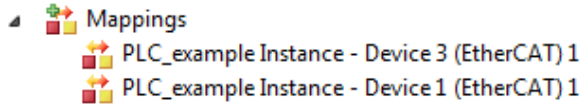


Fig. 57: Reading a variable from the structure of the process data

Activation of the configuration

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



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

Starting the controller

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

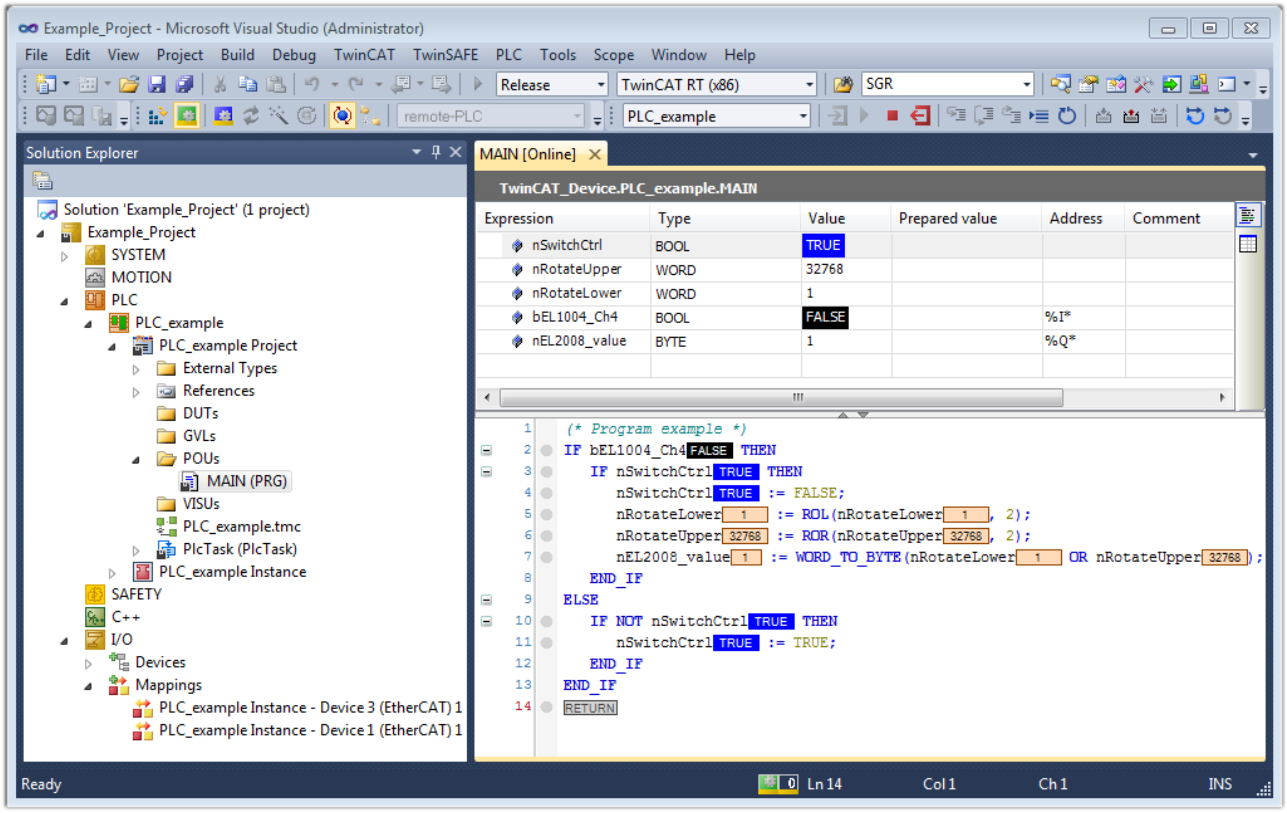


Fig. 58: TwinCAT 3 development environment (VS shell): logged-in, after program startup

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

4.2 TwinCAT Development Environment

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

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

Details:

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

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

Additional features:

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

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

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

4.2.1 Installation of the TwinCAT real-time driver

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

This can be done in several ways.

A: Via the TwinCAT Adapter dialog

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

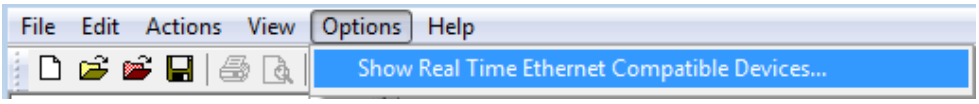


Fig. 59: System Manager “Options” (TwinCAT 2)

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

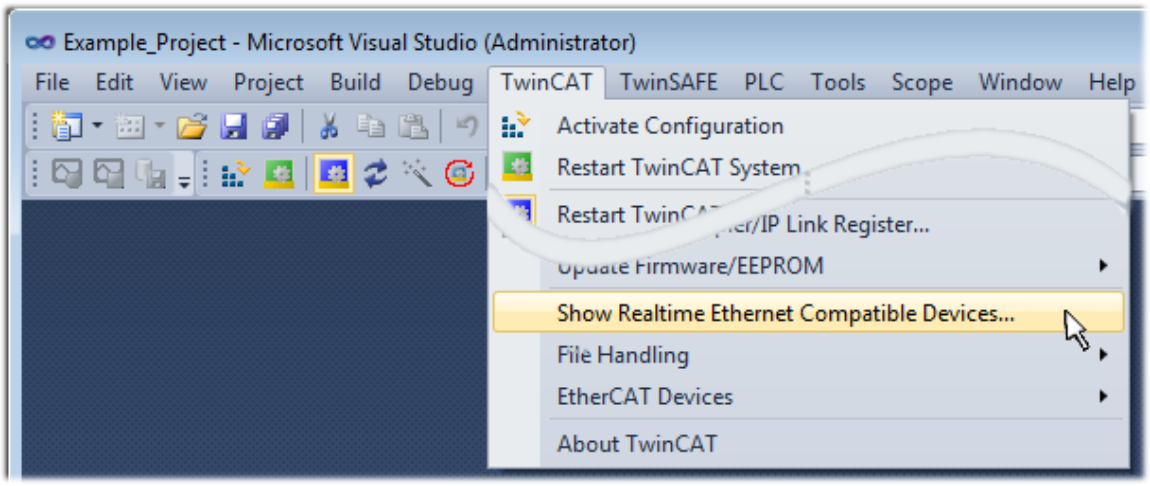


Fig. 60: Call up under VS Shell (TwinCAT 3)

B: Via TcRteInstall.exe in the TwinCAT directory

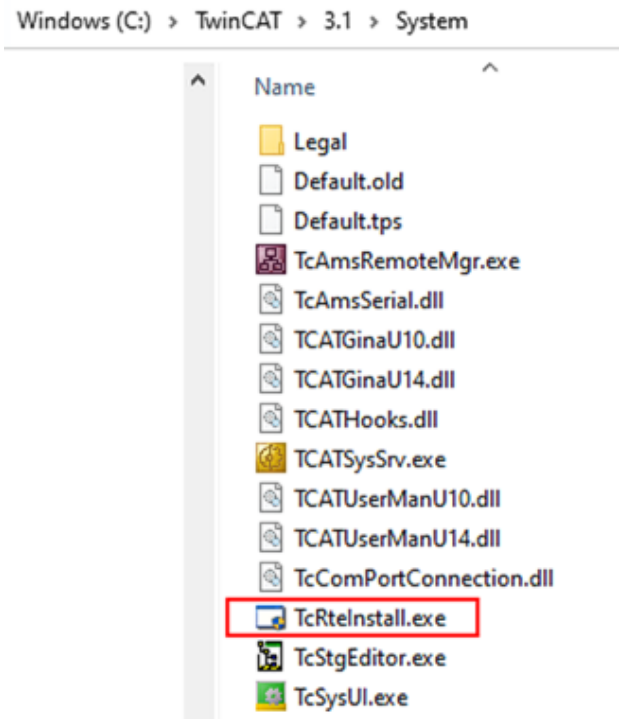


Fig. 61: TcRteInstall in the TwinCAT directory

In both cases, the following dialog appears:

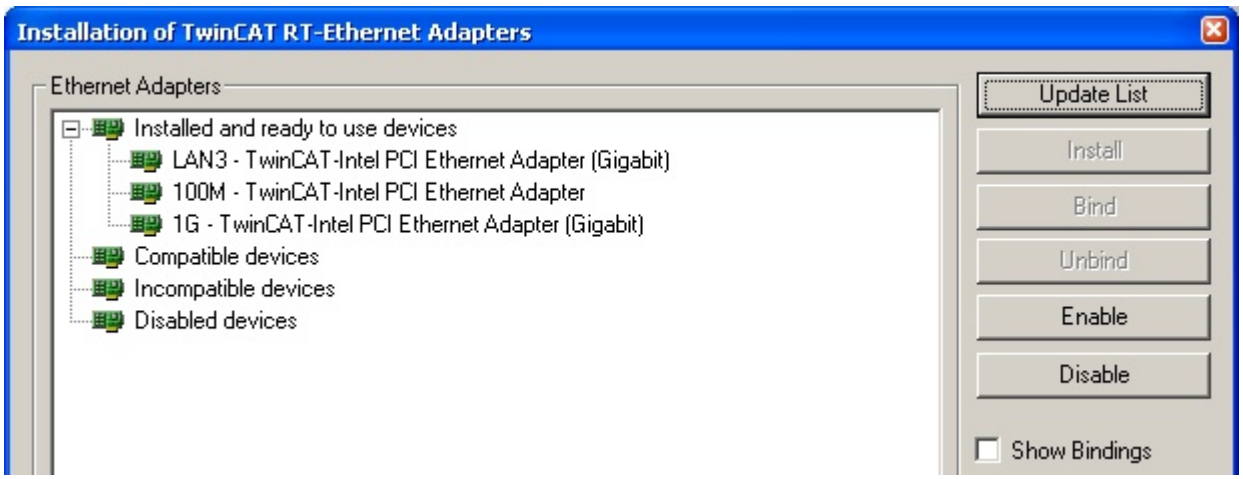


Fig. 62: Overview of network interfaces

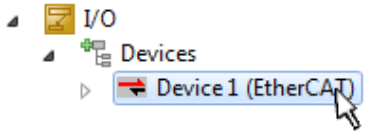
Interfaces listed under “Compatible devices” can be assigned a driver via the “Install” button. A driver should only be installed on compatible devices.

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” [▶ 76] in order to view the compatible ethernet ports via its EtherCAT properties (tab “Adapter”, button “Compatible Devices…”):

Fig. 63: 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)

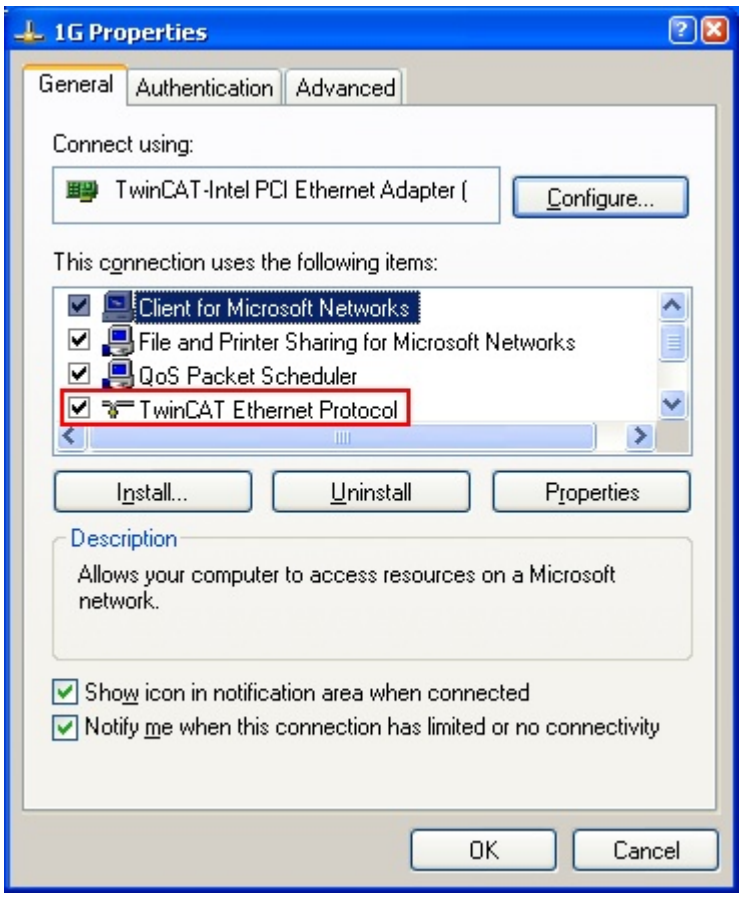


Fig. 64: Windows properties of the network interface

A correct setting of the driver could be:

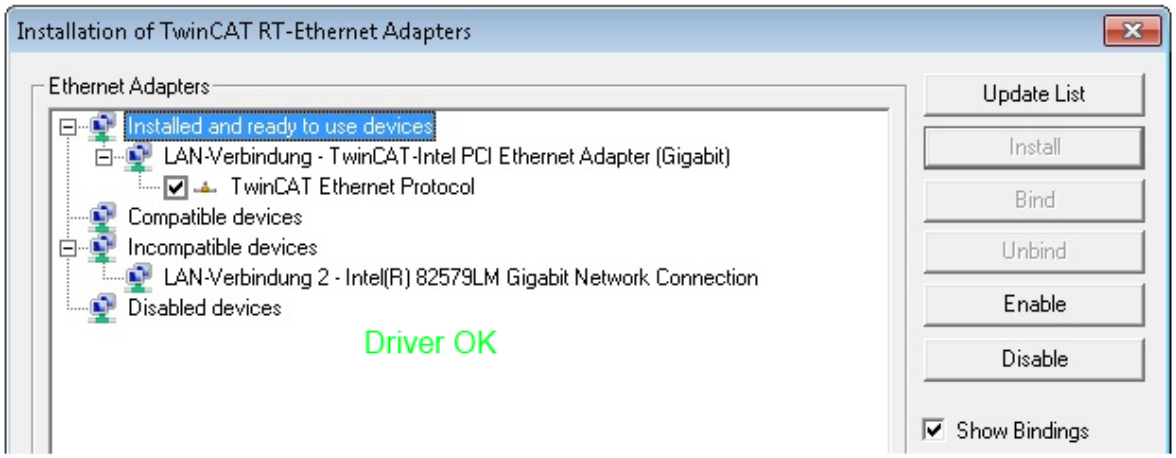


Fig. 65: Exemplary correct driver setting for the Ethernet port

Other possible settings have to be avoided:

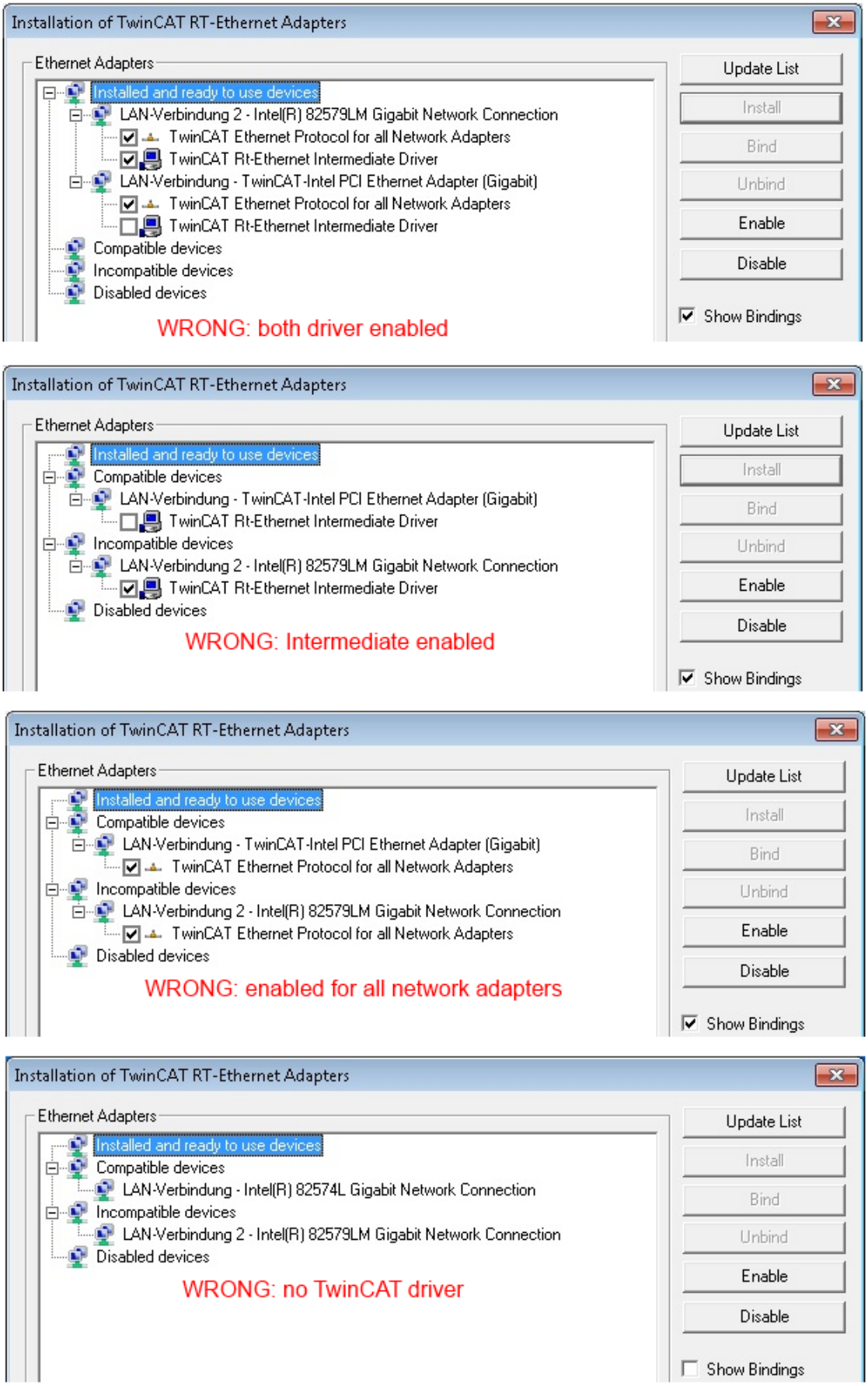


Fig. 66: Incorrect driver settings for the Ethernet port

IP address of the port used

IP address/DHCP

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

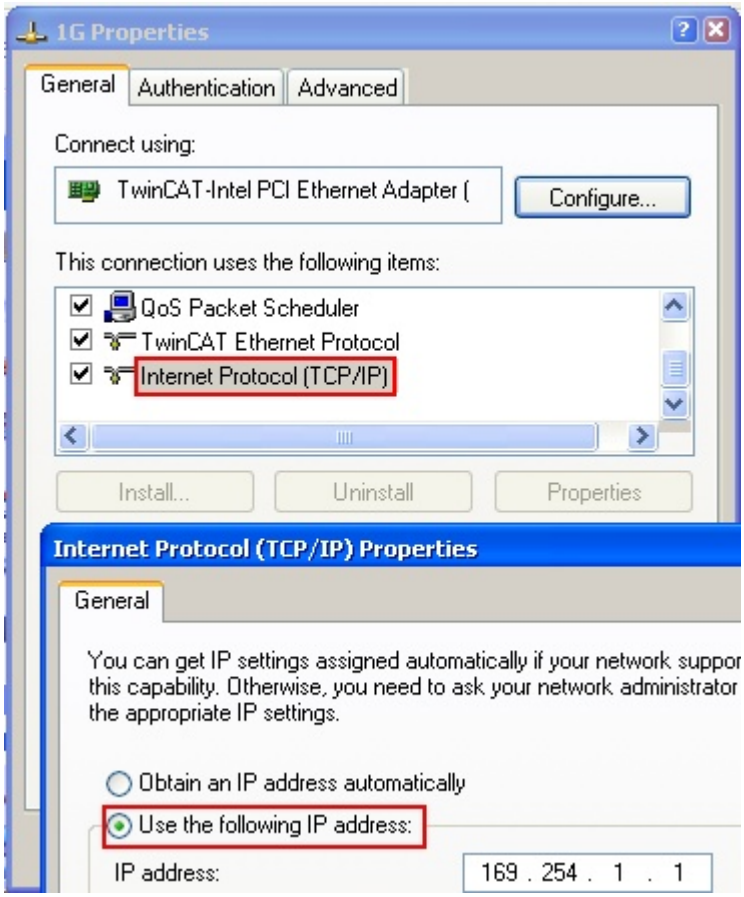


Fig. 67: TCP/IP setting for the Ethernet port

4.2.2 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 [▶ 74] 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”

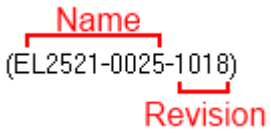


Fig. 68: Identifier structure

The order identifier consisting of name + type (here: EL2521-0025) describes the device function. The revision indicates the technical progress and is managed by Beckhoff. In principle, a device with a higher revision can replace a device with a lower revision, unless specified otherwise, e.g. in the documentation. Each revision has its own ESI description. See further notes [▶ 9].

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.

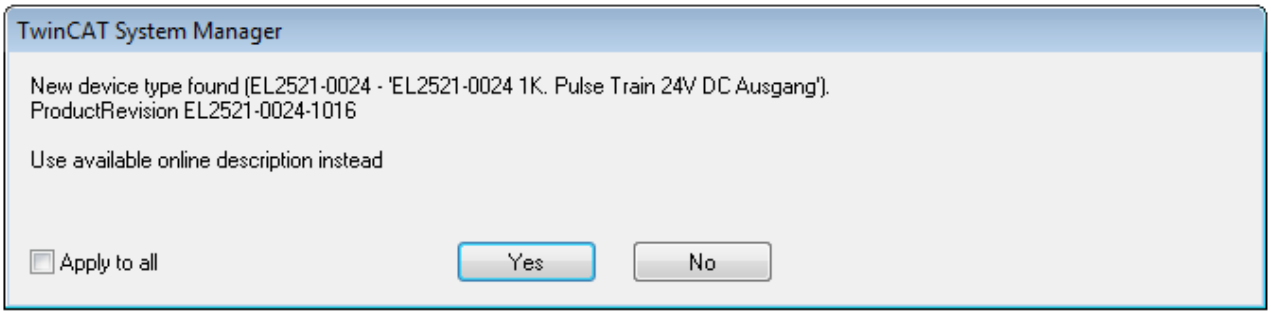


Fig. 69: OnlineDescription information window (TwinCAT 2)

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

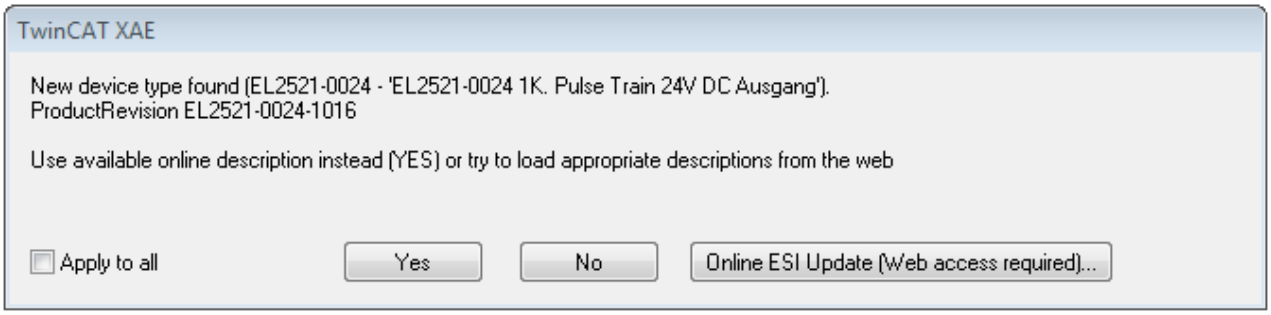


Fig. 70: Information window OnlineDescription (TwinCAT 3)

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

NOTICE

Changing the "usual" configuration through a scan

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

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

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

Refer in particular to the chapter "General notes on the use of Beckhoff EtherCAT IO components" and for manual configuration to the chapter "Offline configuration creation [▶ 76]".

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. 71: File OnlineDescription.xml created by the System Manager

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

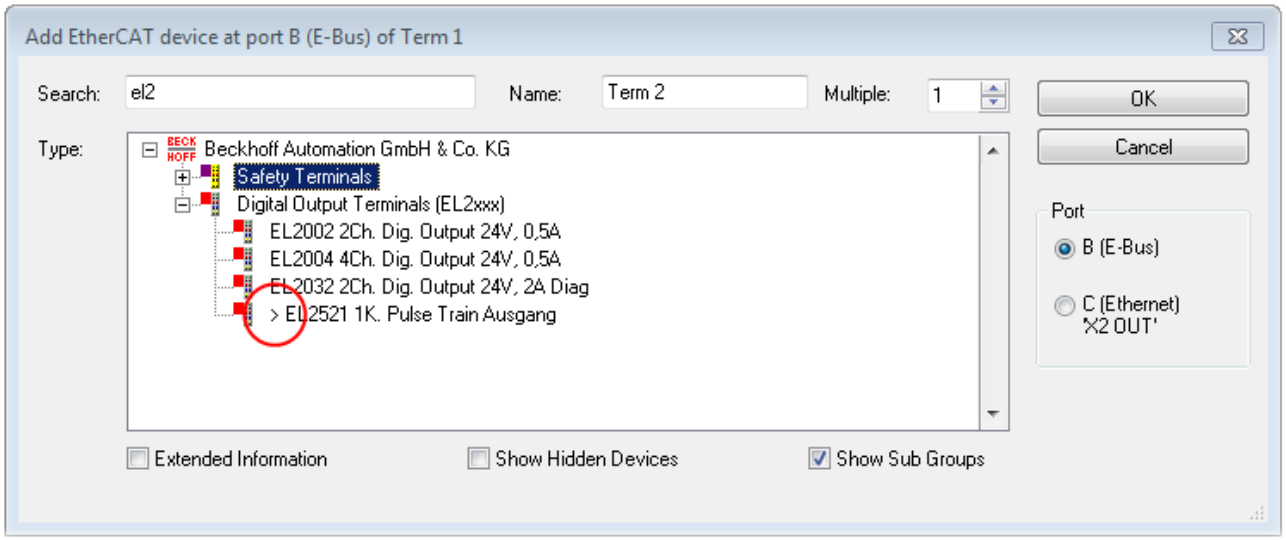


Fig. 72: Indication of an online recorded ESI of EL2521 as an example

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

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

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

OnlineDescription for TwinCAT 3.x

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

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

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

Faulty ESI file

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

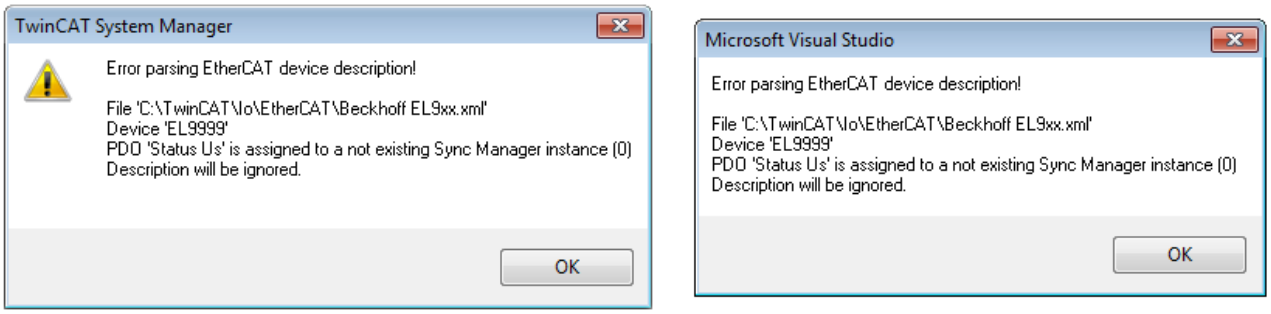


Fig. 73: Information window for faulty ESI file (left: TwinCAT 2; right: TwinCAT 3)

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

4.2.3 TwinCAT ESI Updater

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

4.2.3.1 TwinCAT 3

Step 1) Updating the ESI data sets

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

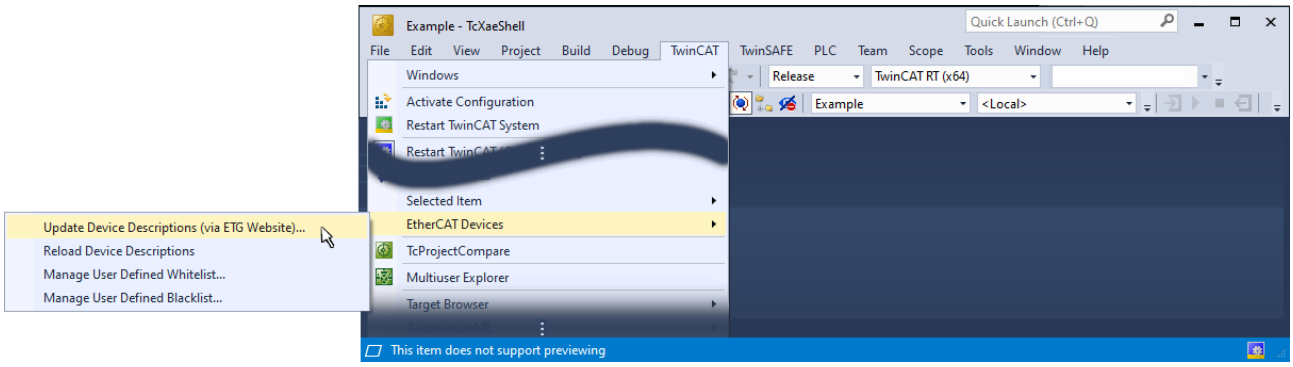
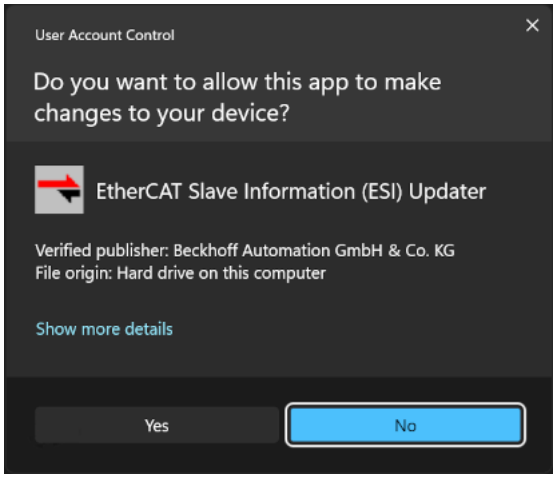


Fig. 74: Calling the ESI Updater (TwinCAT 3)

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



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

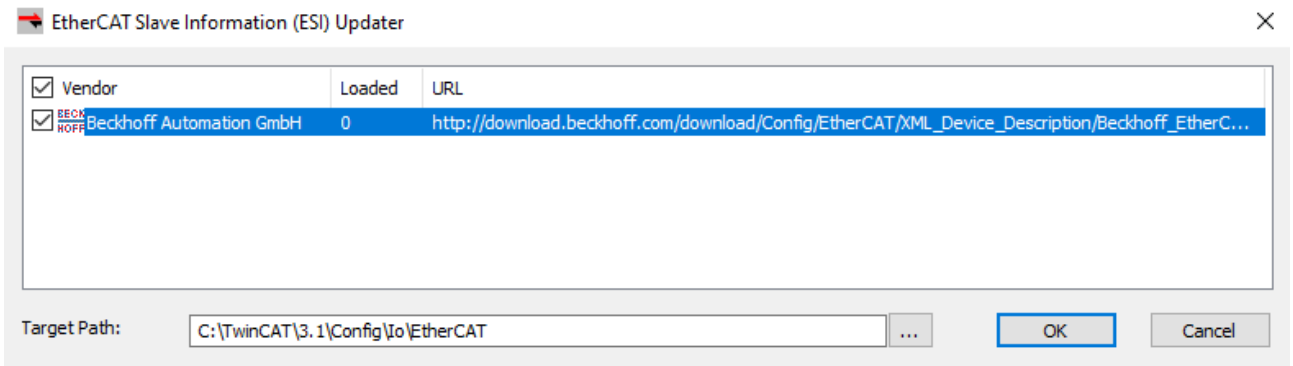
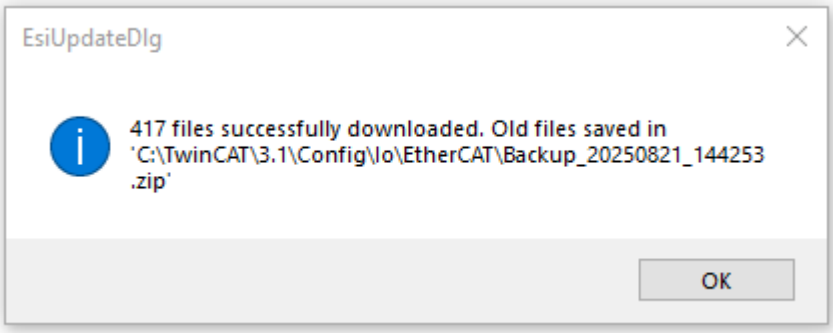


Fig. 75: (ESI) Updater

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

The completion of the download process is indicated by the message



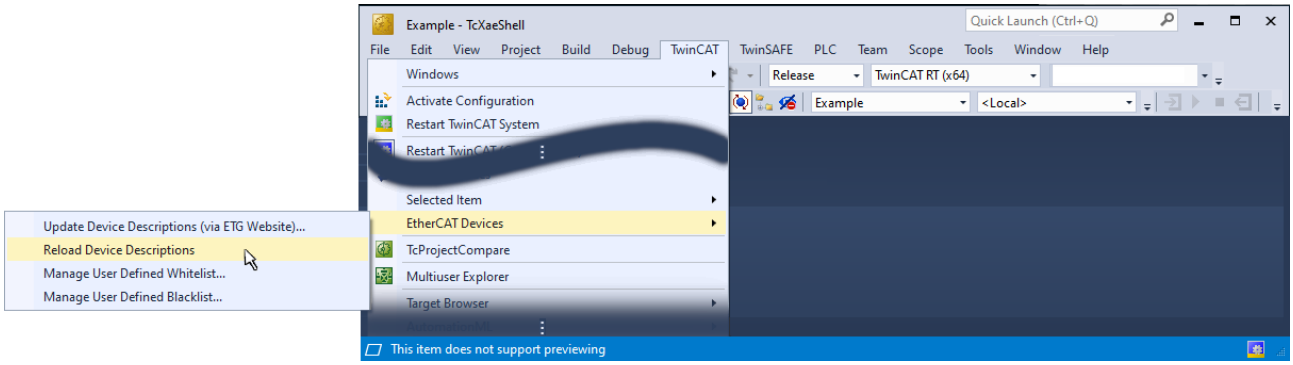
, which shows the number of downloaded files and the storage location or the backup zip folder. Click [OK] to close the window.

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

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

Step 2) Update ESI cache

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



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

4.2.3.2 TwinCAT 2

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

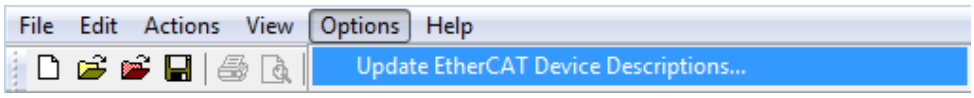


Fig. 76: Using the ESI Updater (>= TwinCAT 2.11)

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

4.2.4 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” [▶ 70].

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 [▶ 82] (Ethernet port at the IPC)
- detecting the connected EtherCAT devices [▶ 83]. This step can be carried out independent of the preceding step
- troubleshooting [▶ 86]

The scan with existing configuration [▶ 87] can also be carried out for comparison.

4.2.5 OFFLINE configuration creation

Creating the EtherCAT device

Create an EtherCAT device in an empty System Manager window.

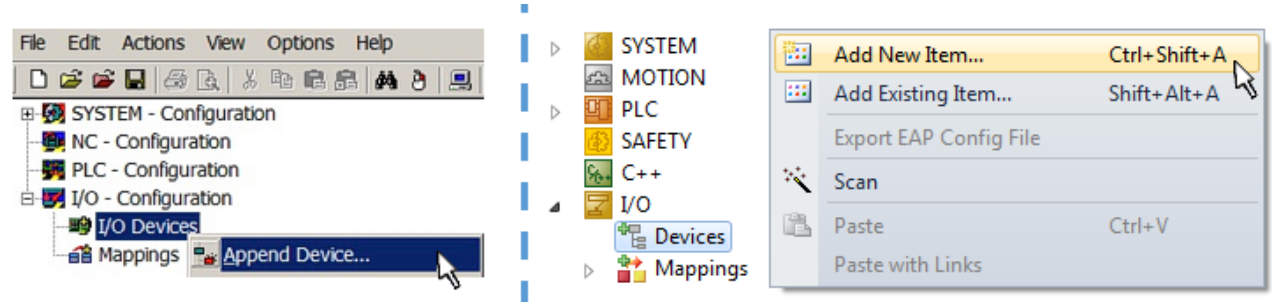


Fig. 77: Append EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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”.

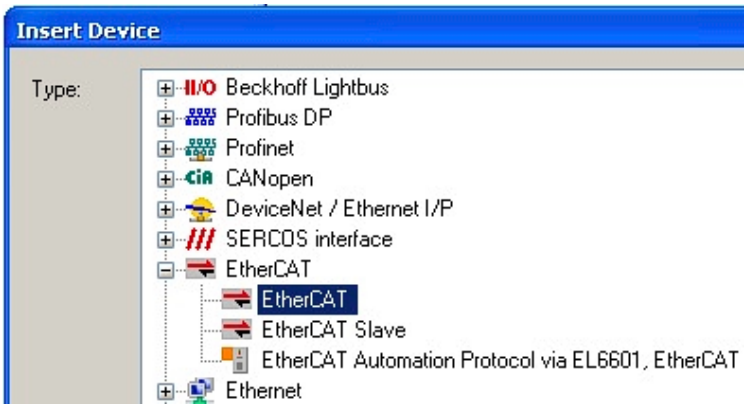


Fig. 78: Selecting the EtherCAT connection (TwinCAT 2.11, TwinCAT 3)

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

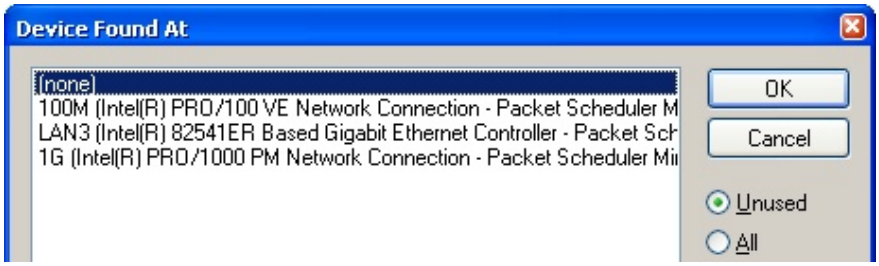


Fig. 79: Selecting the Ethernet port

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)”.

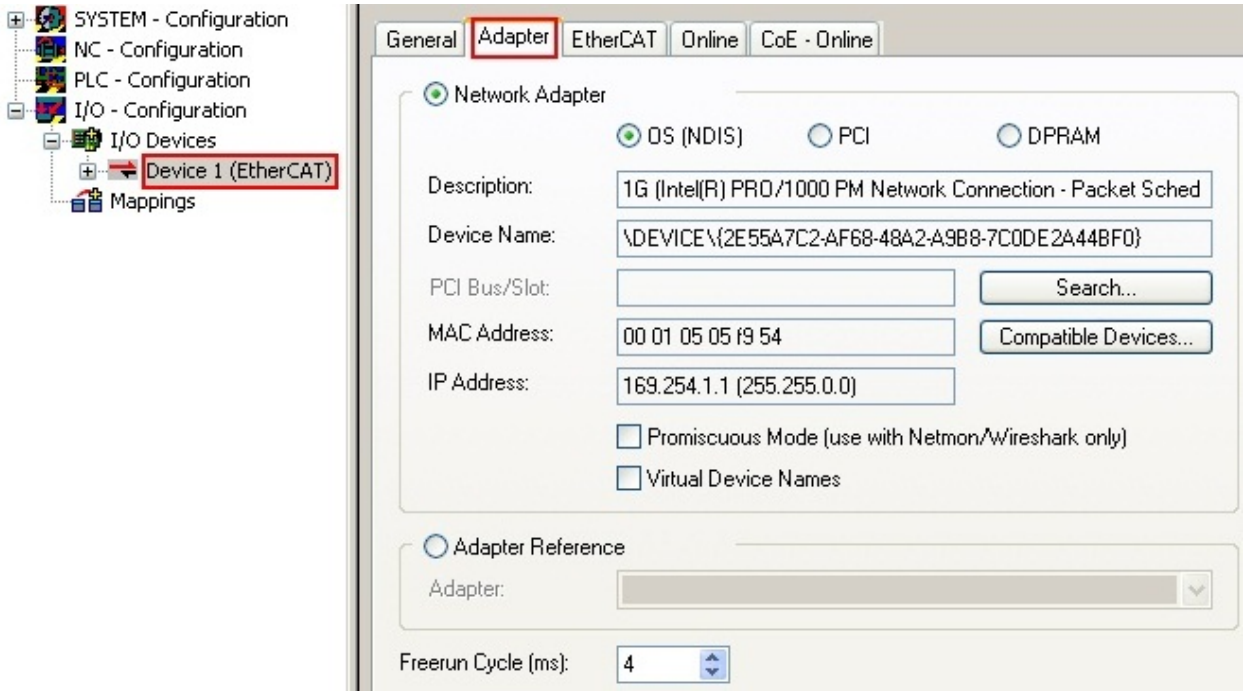
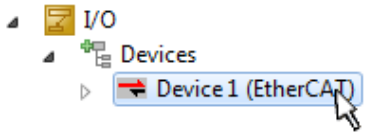


Fig. 80: EtherCAT device properties (TwinCAT 2)

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



Selecting the Ethernet port

Ethernet ports can only be selected for EtherCAT devices for which the TwinCAT real-time driver is installed. This has to be done separately for each port. Please refer to the respective installation page [▶ 64].

Defining EtherCAT slaves

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

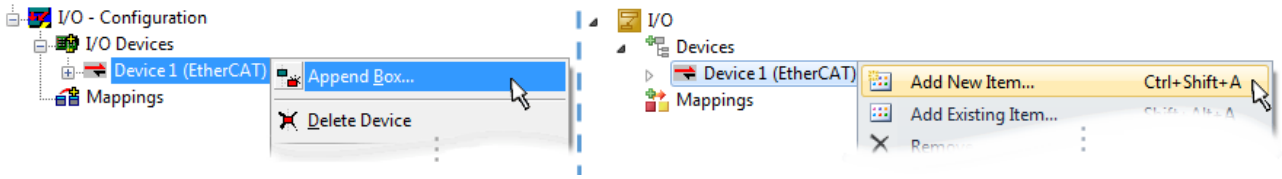


Fig. 81: Appending EtherCAT devices (left: TwinCAT 2; right: TwinCAT 3)

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

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

Overview of physical layer

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

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

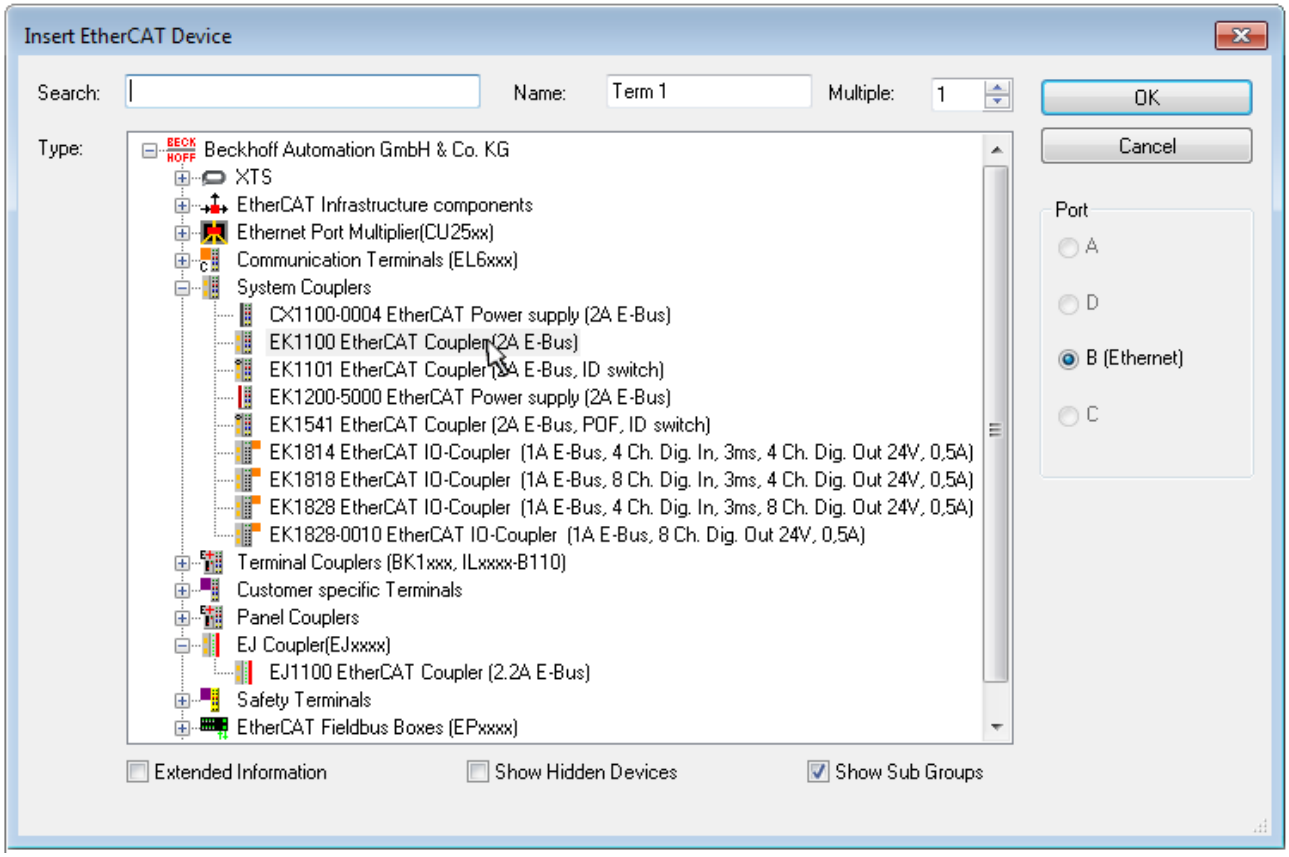


Fig. 82: Selection dialog for new EtherCAT device

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”.

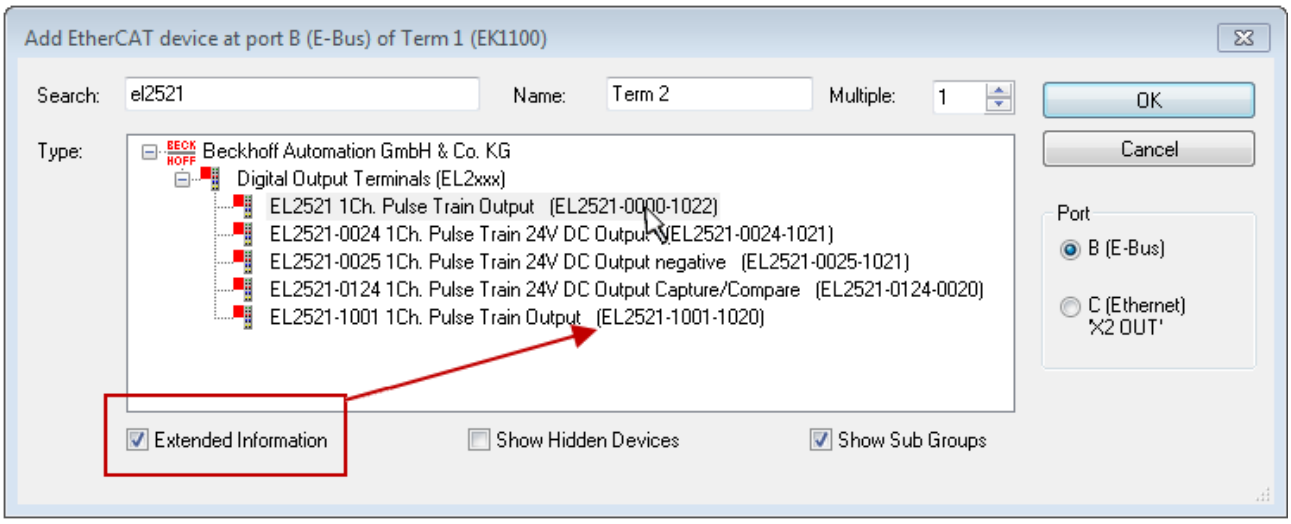


Fig. 83: Display of device revision

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”.

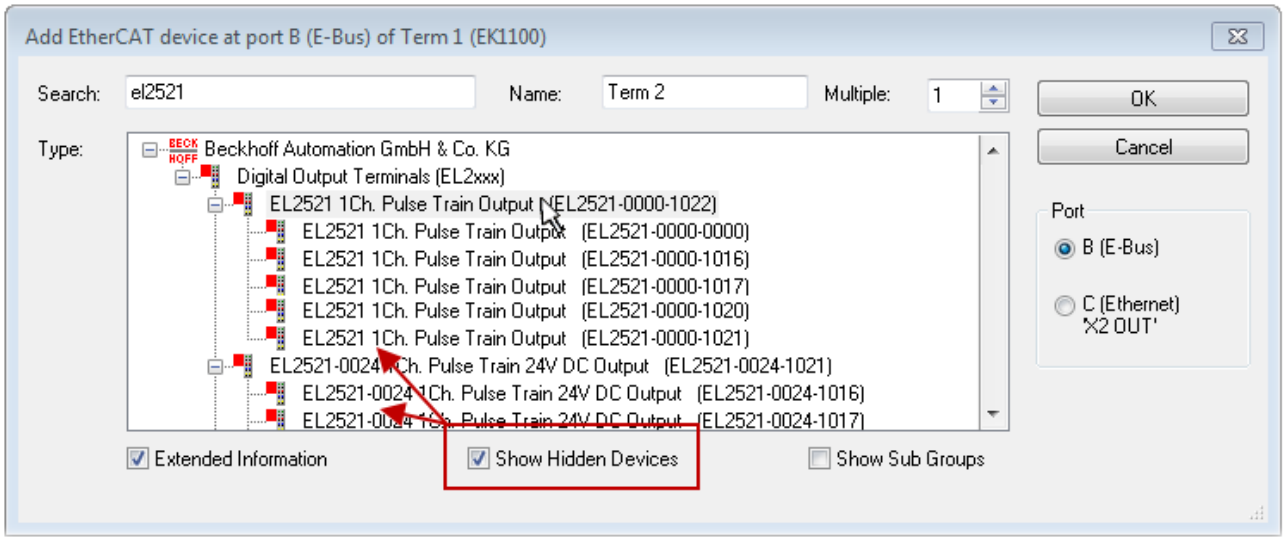


Fig. 84: Display of previous revisions

Device selection based on revision, compatibility

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

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

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

Example

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

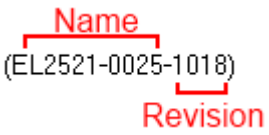


Fig. 85: Name/revision of the terminal

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

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

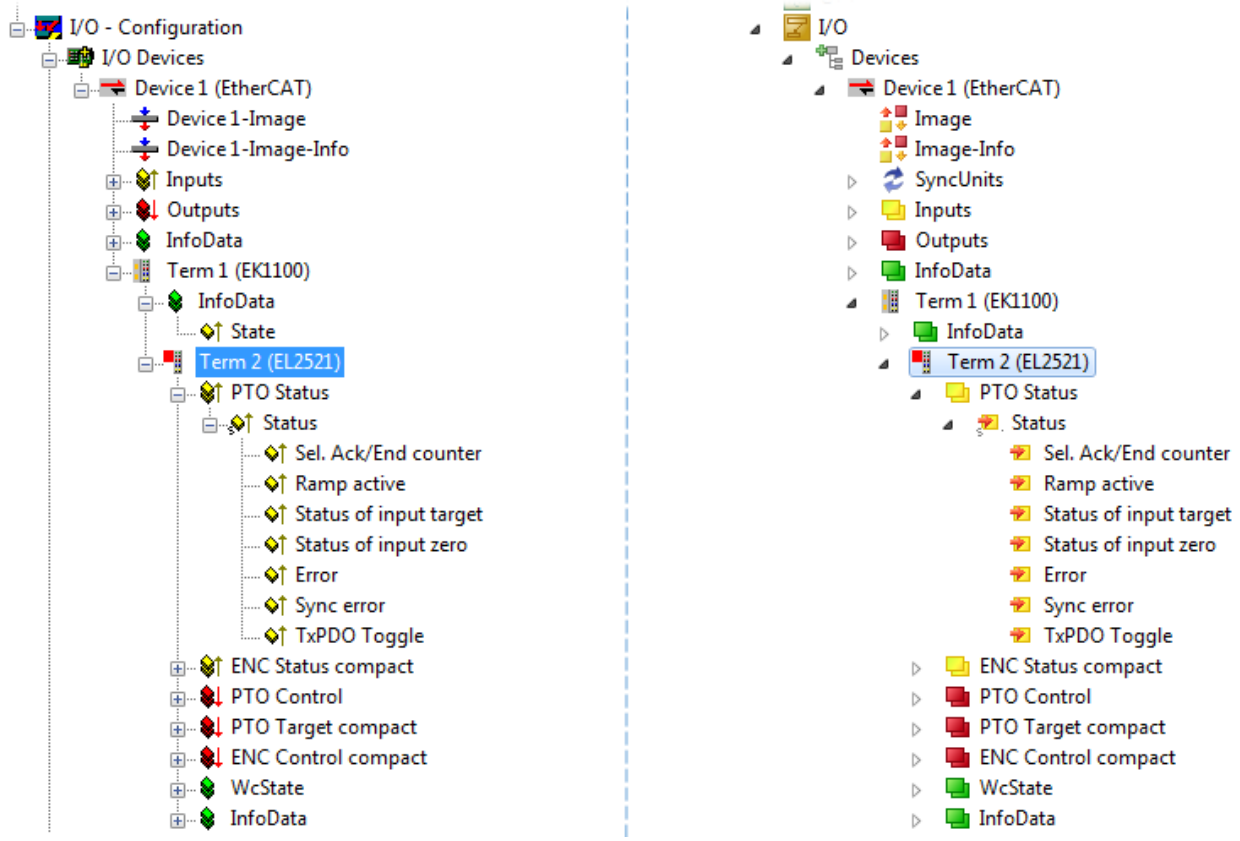


Fig. 86: EtherCAT terminal in the TwinCAT tree (left: TwinCAT 2; right: TwinCAT 3)

4.2.6 ONLINE configuration creation

Detecting/scanning of the EtherCAT device

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

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

TwinCAT can be set into this mode:

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

Online scanning in Config mode

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

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

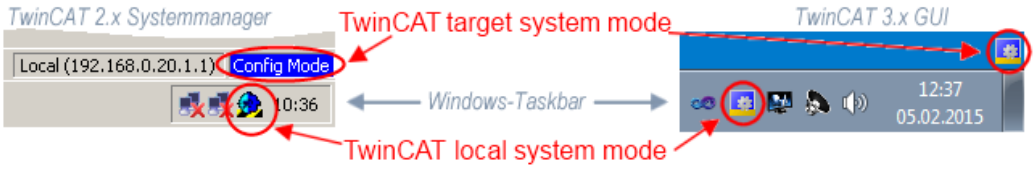


Fig. 87: Differentiation local/target system (left: TwinCAT 2; right: TwinCAT 3)

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

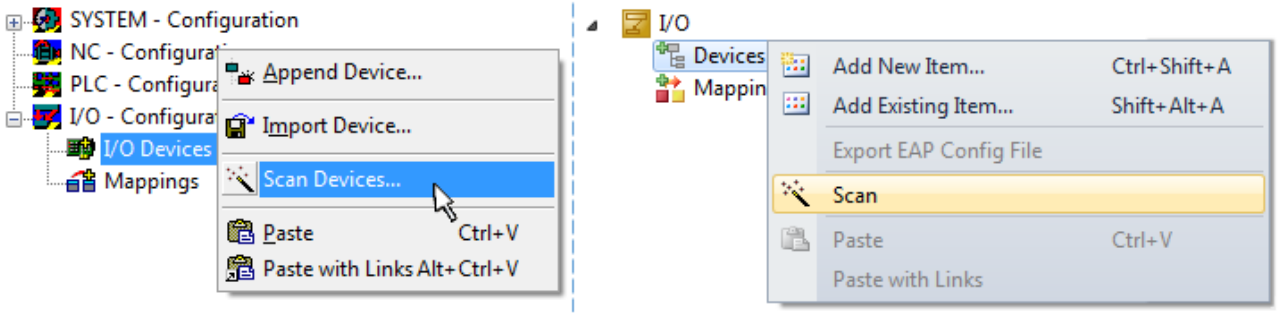


Fig. 88: Scan Devices (left: TwinCAT 2; right: TwinCAT 3)

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

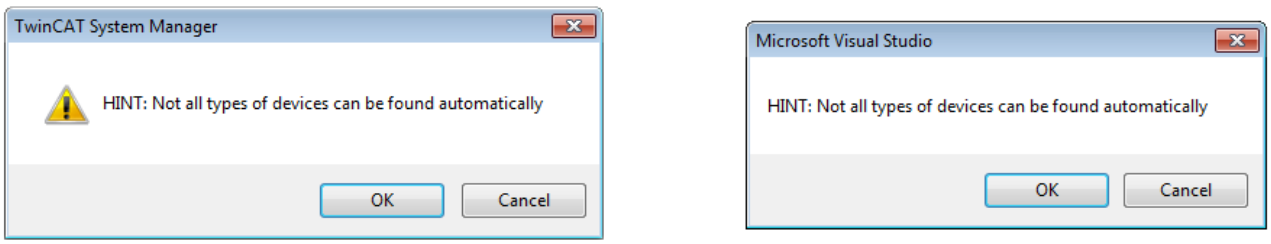


Fig. 89: Note for automatic device scan (left: TwinCAT 2; right: TwinCAT 3)

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” .

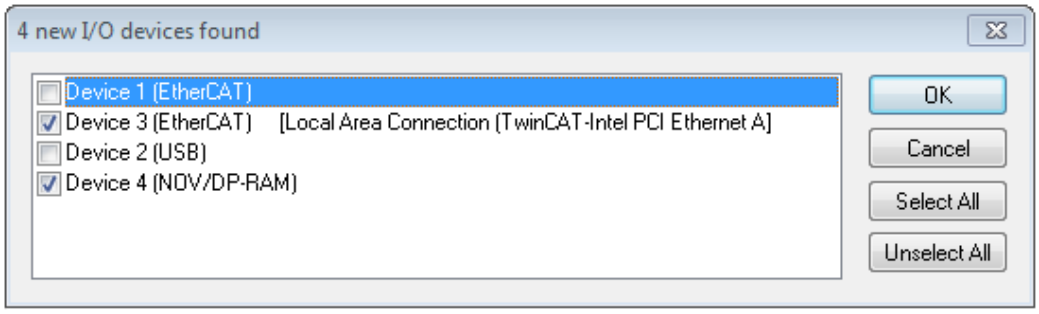


Fig. 90: Detected Ethernet devices

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 [▶ 64].

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.

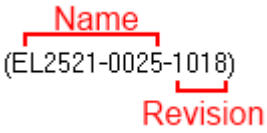


Fig. 91: Example default state

NOTICE

Slave scanning in practice in series machine production

The scanning function should be used with care. It is a practical and fast tool for creating an initial configuration as a basis for commissioning. In series machine production or reproduction of the plant, however, the function should no longer be used for the creation of the configuration, but if necessary for comparison [▶ 87] 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:

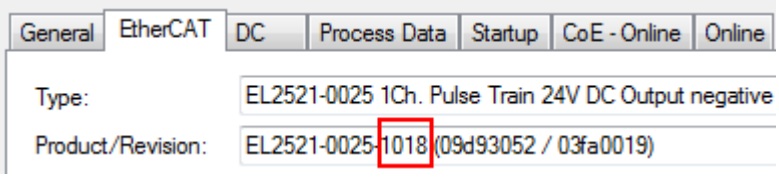


Fig. 92: Installing EthetCAT terminal with revision -1018

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

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

After some time Beckhoff extends the EL2521-0025 by a new feature C. Therefore the FW is changed, outwardly recognizable by a higher FW version and **a new revision -1019**. Nevertheless the new device naturally supports functions and interfaces of the predecessor version(s); an adaptation of “B.tsm” or even “B.pro” is therefore unnecessary. The series-produced machines can continue to be built with “B.tsm” and “B.pro”; it makes sense to perform a comparative scan [▶ 87] 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:

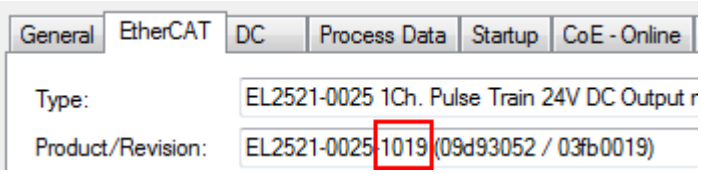


Fig. 93: Detection of EtherCAT terminal with revision -1019

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

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

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

Fig. 94: Scan query after automatic creation of an EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

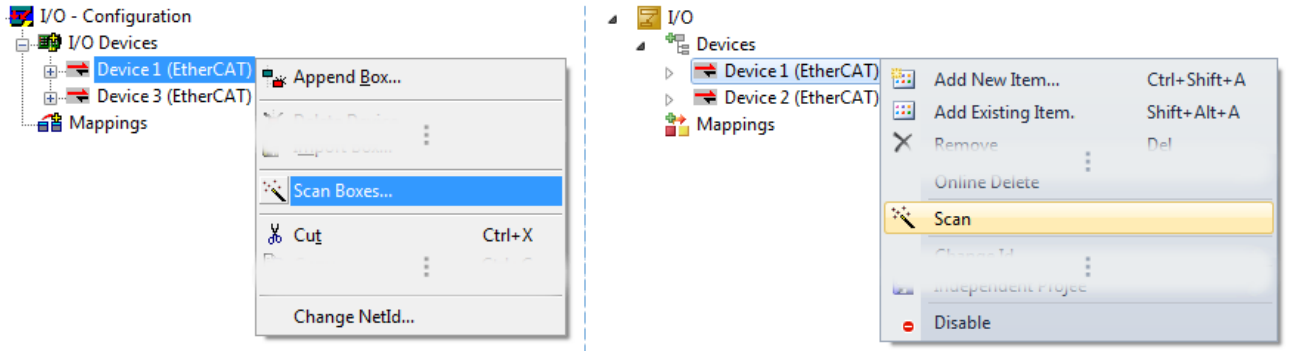


Fig. 95: Manual scanning for devices on a specified EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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

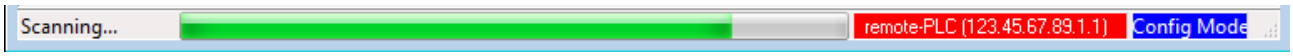


Fig. 96: *Scan progressexemplary by TwinCAT 2*

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

Fig. 97: 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).

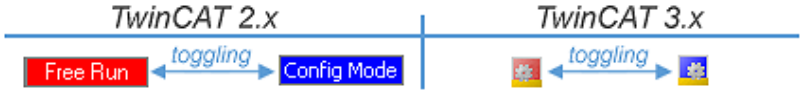


Fig. 98: Displaying of “Free Run” and “Config Mode” toggling right below in the status bar

Fig. 99: 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*.

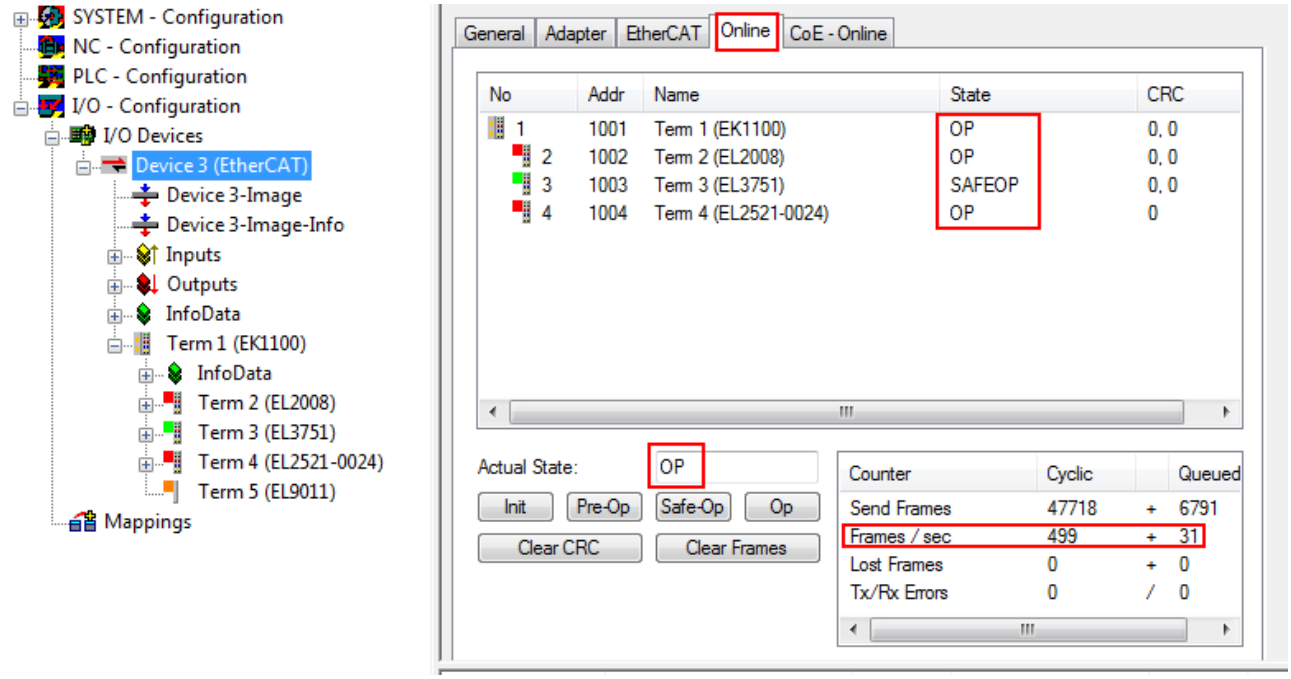


Fig. 100: Online display example

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 [▶ 76].

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.

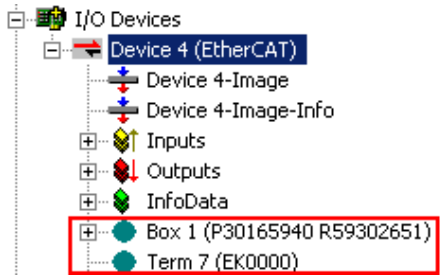


Fig. 101: Faulty identification

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

Scan over existing Configuration

NOTICE

Change of the configuration after comparison

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

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

Fig. 102: 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.

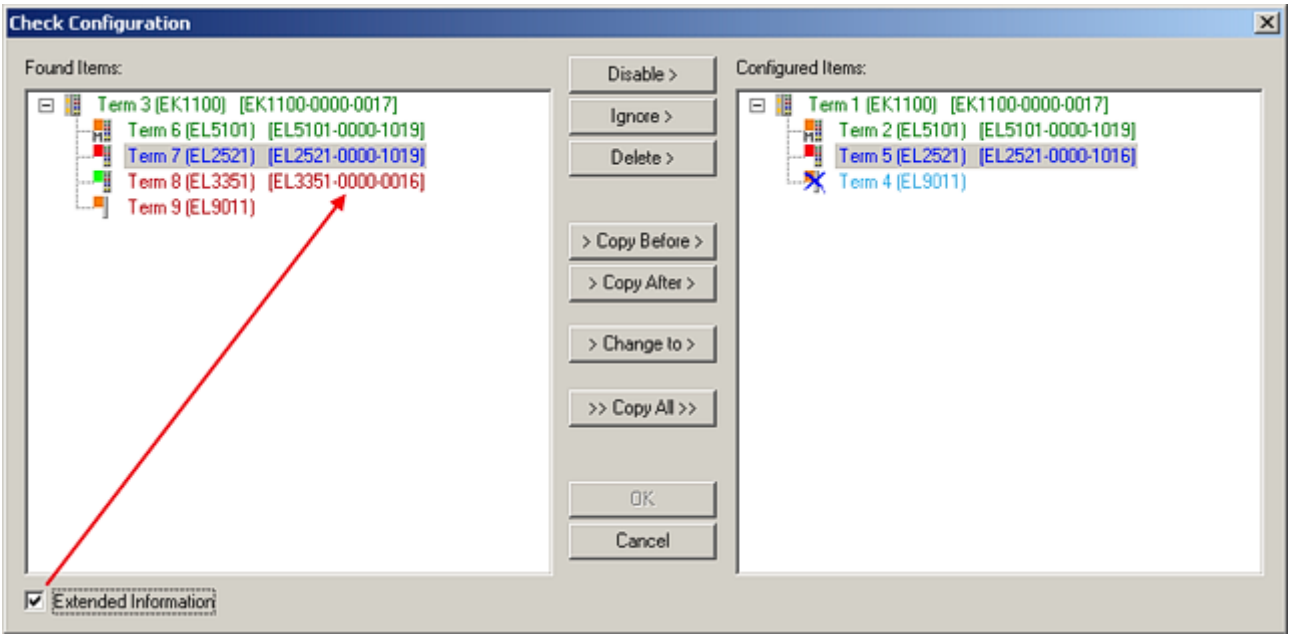


Fig. 103: Correction dialog

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

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

Device selection based on revision, compatibility

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

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

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

Example

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

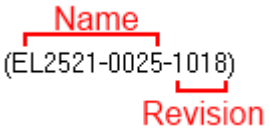


Fig. 104: Name/revision of the terminal

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

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

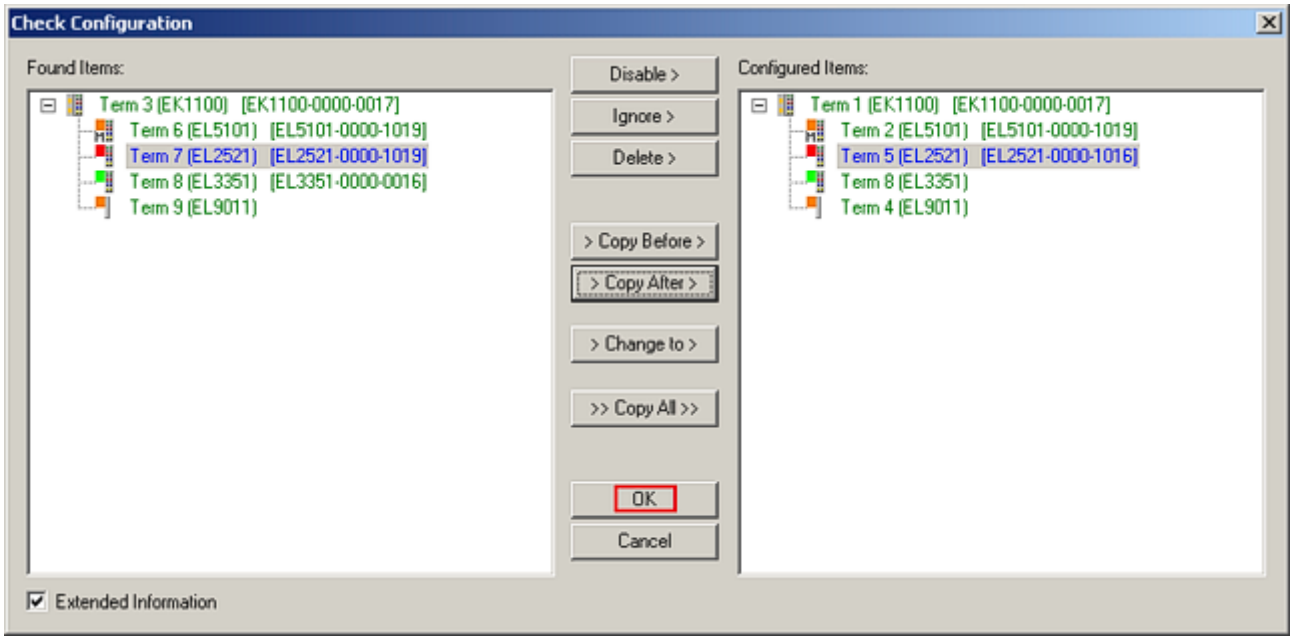


Fig. 105: Correction dialog with modifications

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.

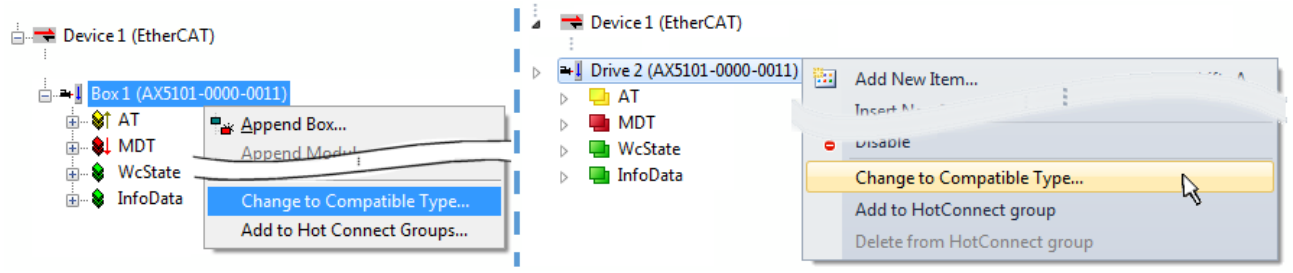


Fig. 106: Dialog “Change to Compatible Type…” (left: TwinCAT 2; right: TwinCAT 3)

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

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

This function is preferably to be used on AX5000 devices.

Change to Alternative Type

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

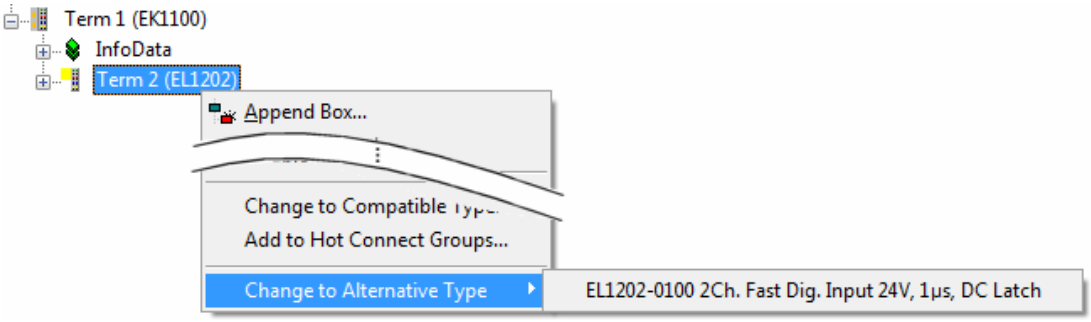


Fig. 107: TwinCAT 2 Dialog Change to Alternative Type

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

4.2.7 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).

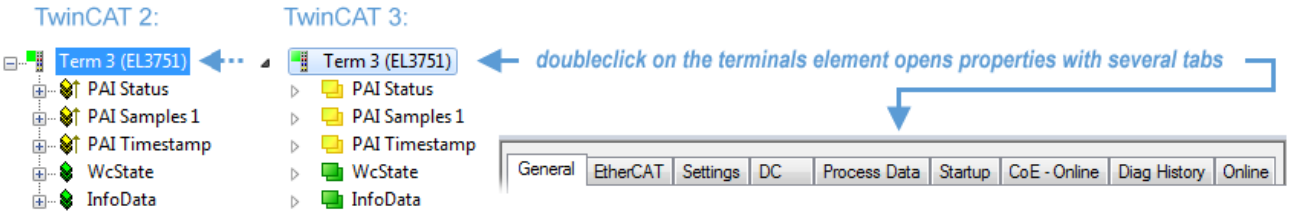


Fig. 108: Branch element as terminal EL3751

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

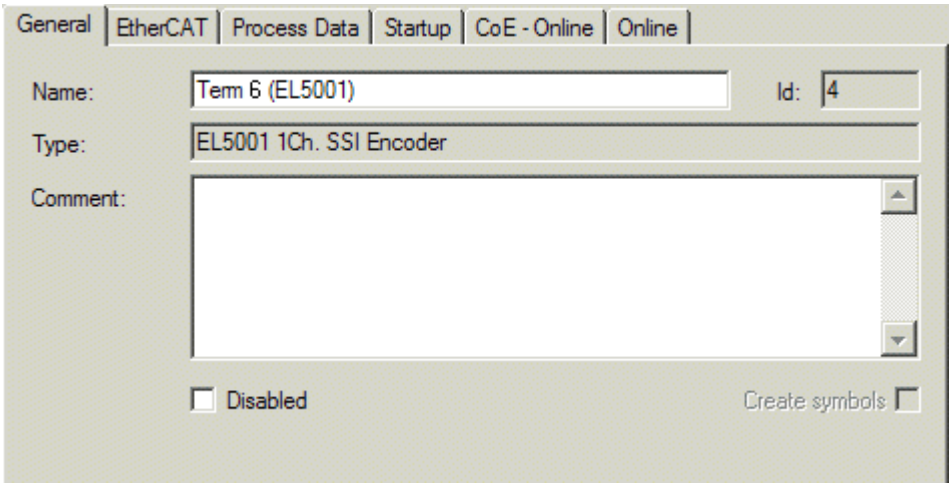


Fig. 109: “General” tab

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

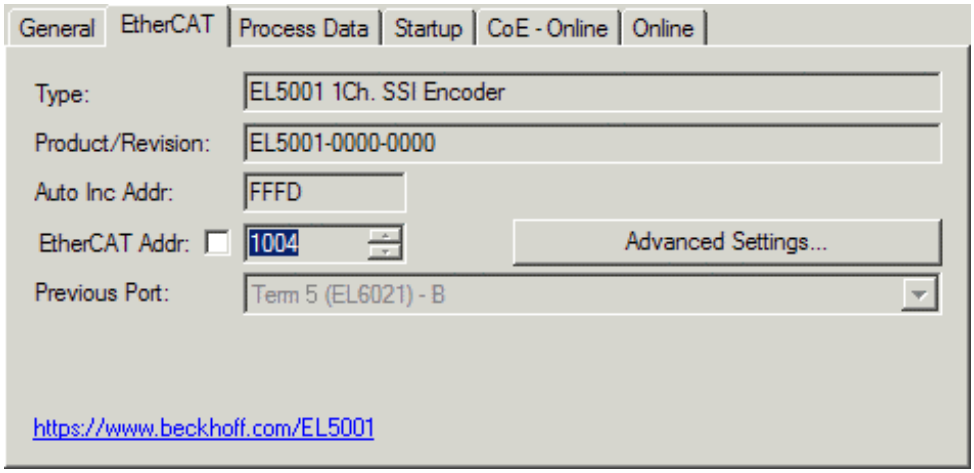


Fig. 110: “EtherCAT” tab

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

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

“Process Data” tab

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

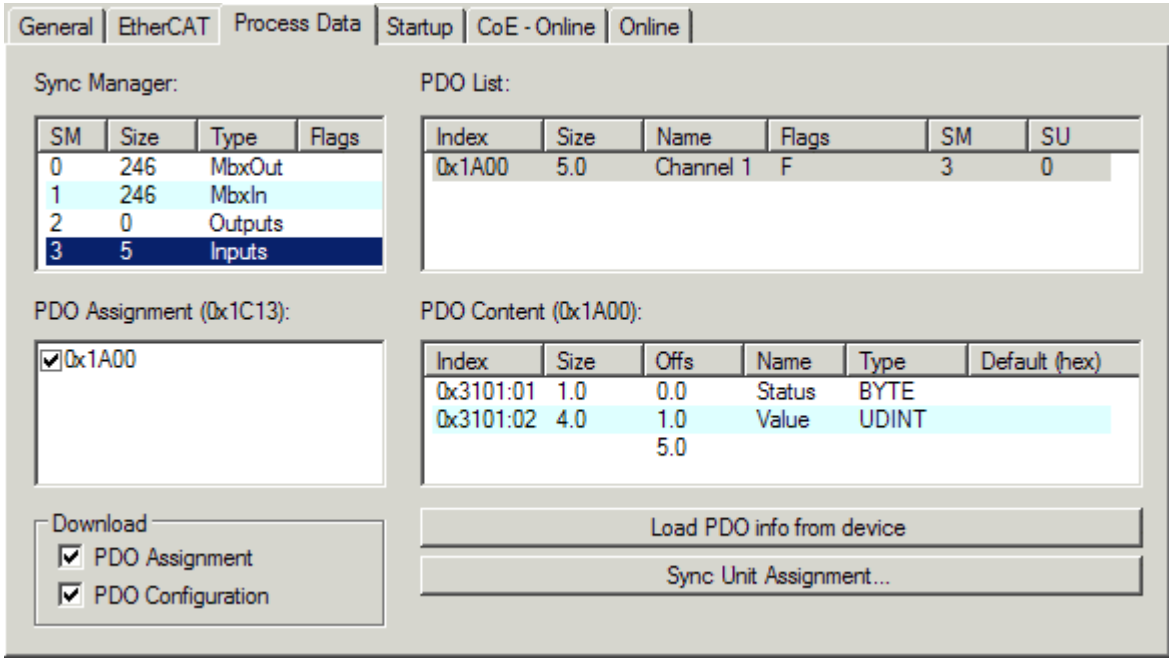


Fig. 111: “Process Data” tab

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”).

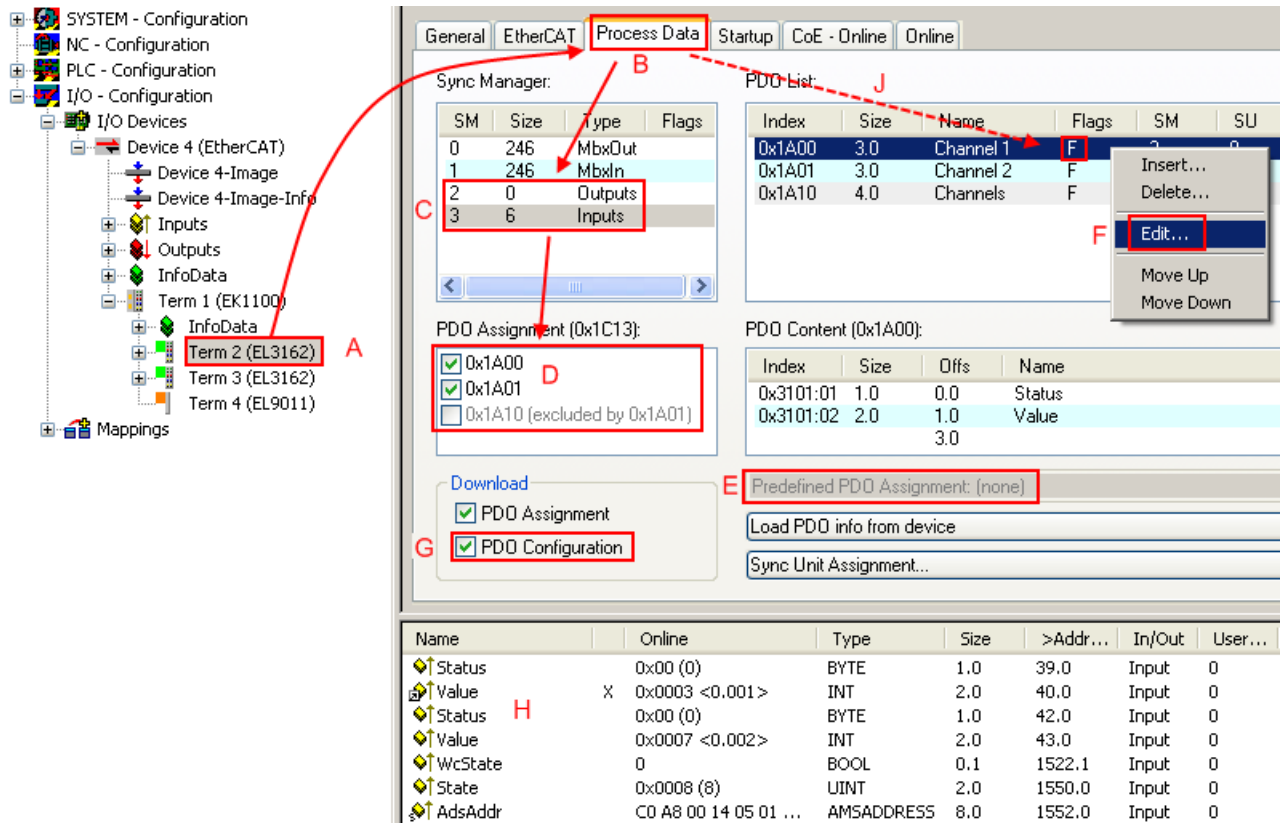


Fig. 112: Configuring the process data

Manual modification of the process data

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

A detailed description [▶ 98] 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.

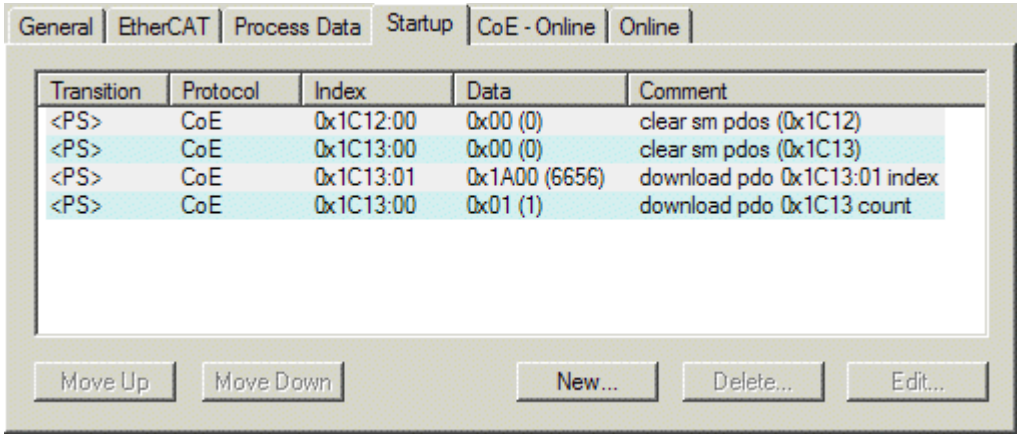


Fig. 113: “Startup” tab

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

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

“CoE - Online” tab

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

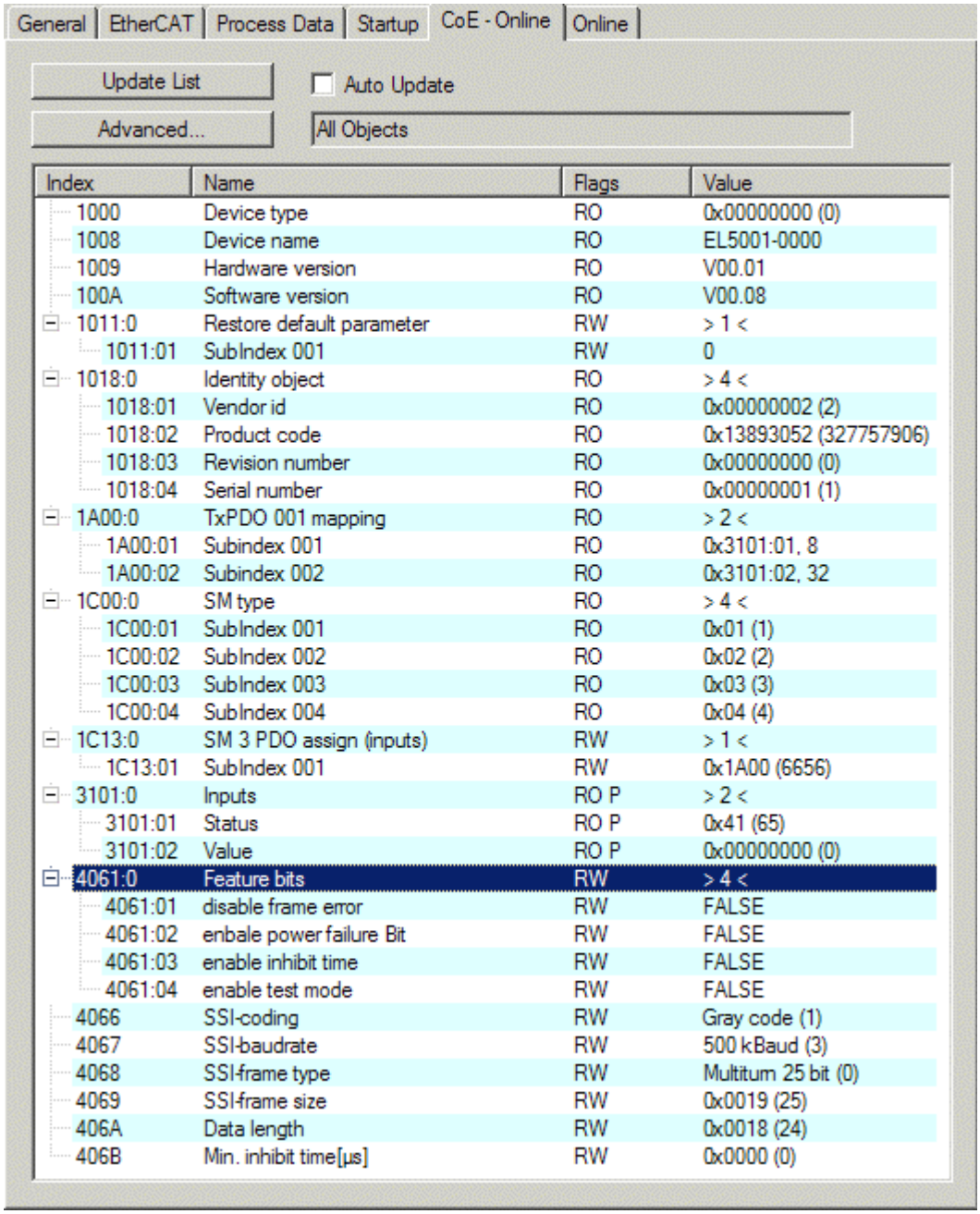


Fig. 114: “CoE - Online” tab

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.

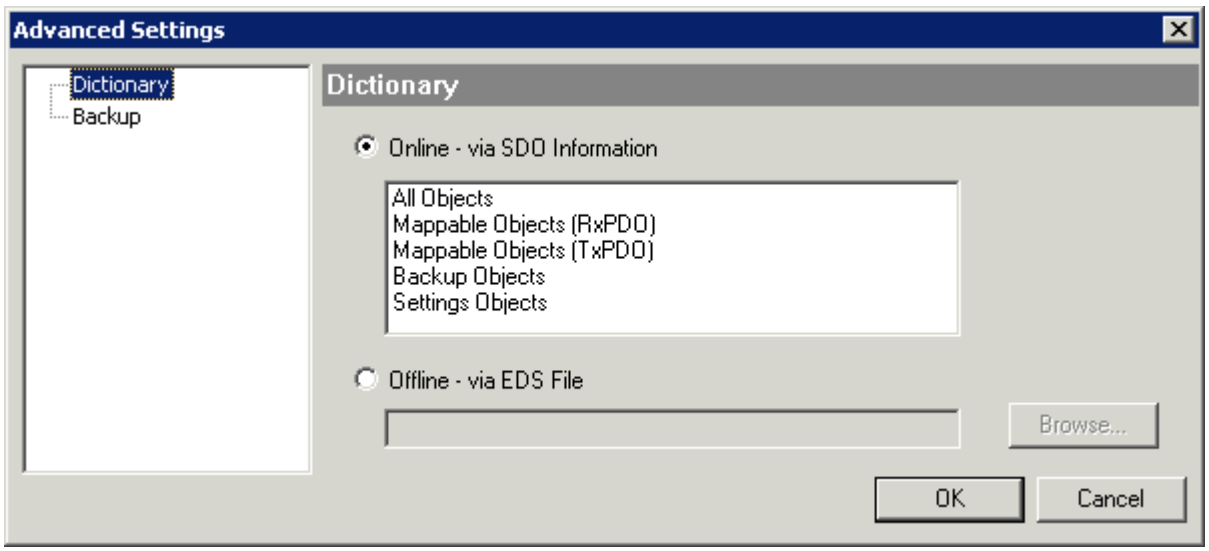


Fig. 115: Dialog “Advanced settings”

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

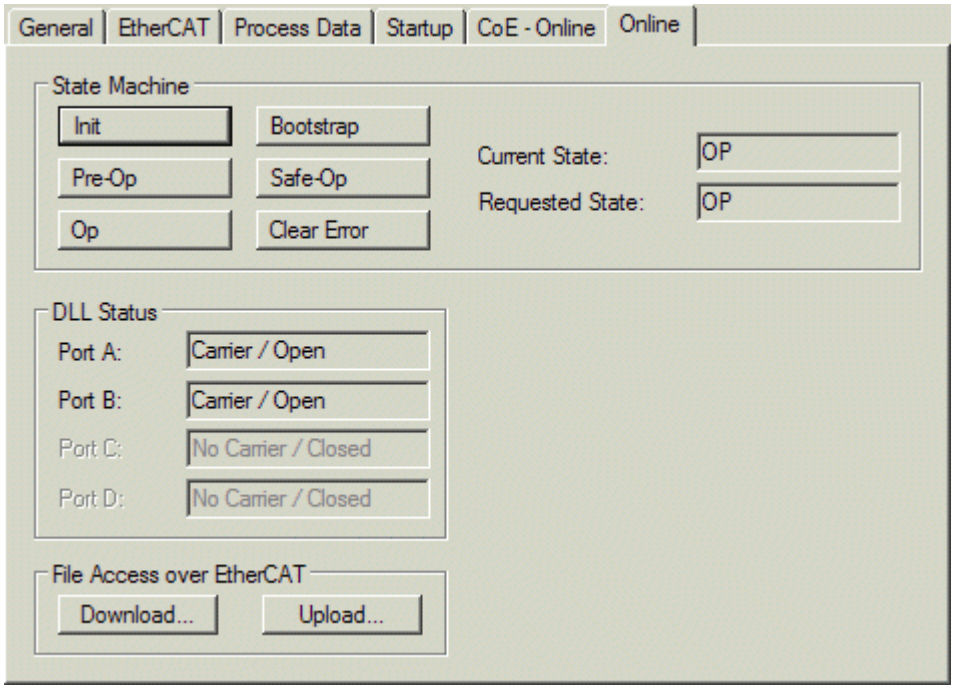


Fig. 116: “Online” tab

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)

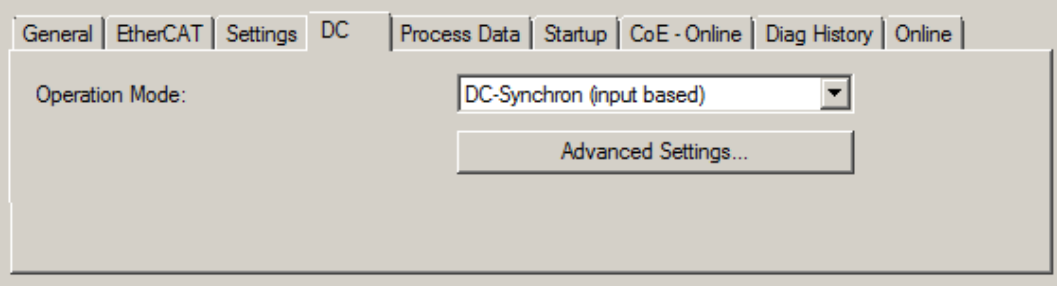


Fig. 117: “DC” tab (Distributed Clocks)

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

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

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

4.2.7.1 Detailed description of Process Data tab

Sync Manager

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

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

PDO Assignment

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

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

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

Activation of PDO assignment

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

a) the EtherCAT slave has to run through the PS status transition cycle (from pre-operational to safe-operational) once (see Online tab [▶ 96]),

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

(button for TwinCAT 2 or button for TwinCAT 3)

PDO list

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

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

PDO Content

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

Download

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

PDO Assignment

If this check box is selected, the PDO assignment that is configured in the PDO Assignment list is downloaded to the device on startup. The required commands to be sent to the device can be viewed in the Startup [▶ 93] 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.

4.2.8 Import/Export of EtherCAT devices with SCI and XTI

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

4.2.8.1 Basic principles

An EtherCAT slave is basically parameterized through the following elements:

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

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

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

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

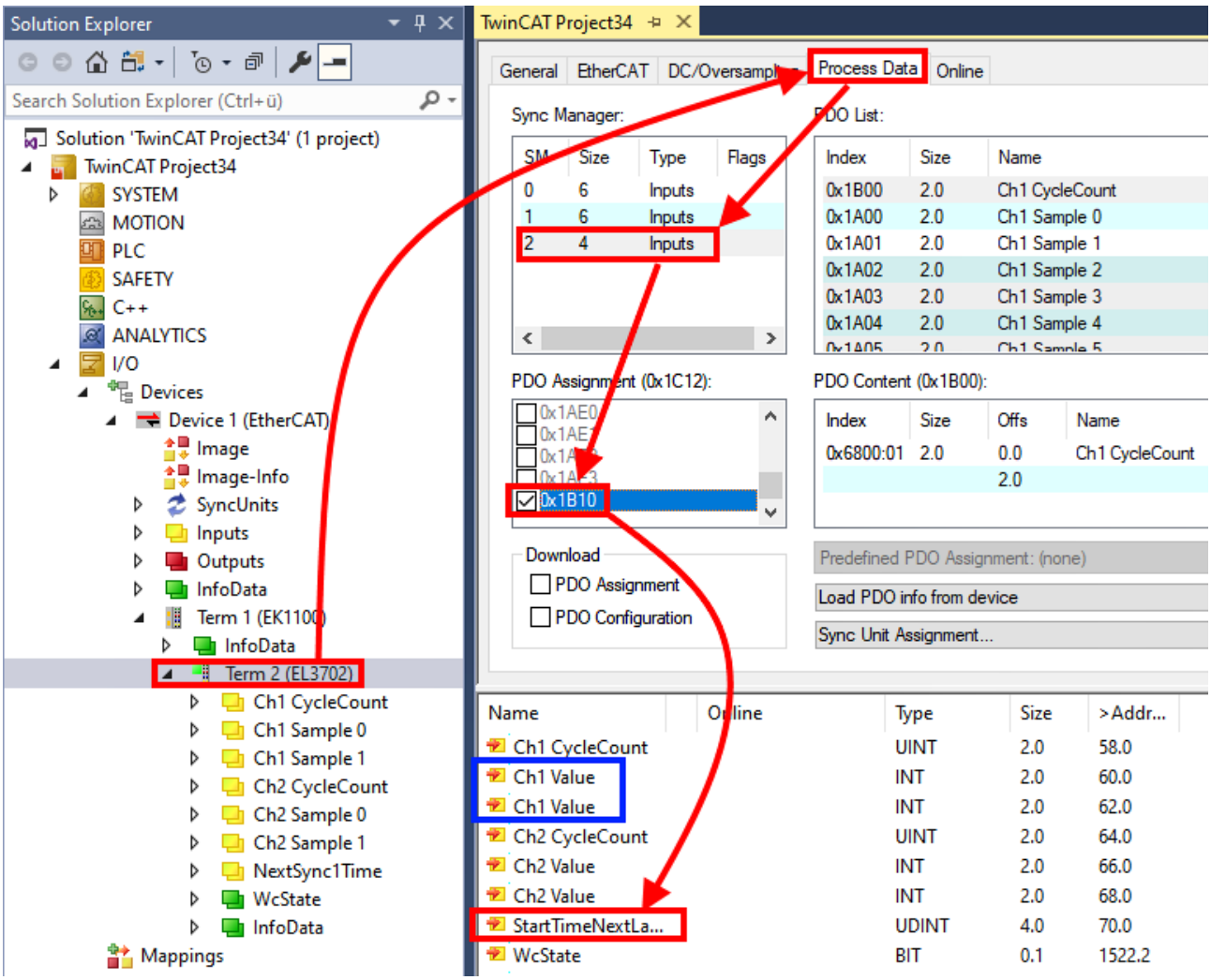
It may become necessary to reuse the modified slave in other projects in this way, without having to make equivalent configuration changes to the slave again. To accomplish this, proceed as follows:

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

TwinCAT offers two methods for this purpose:

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

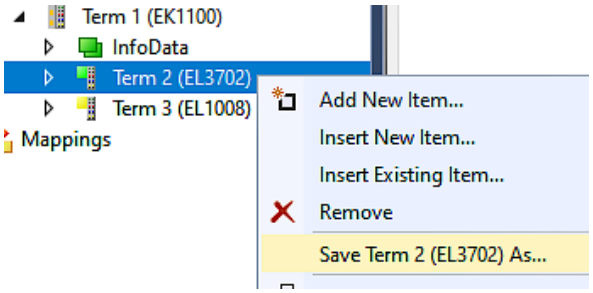
An example is provided below for illustration purposes: an EL3702 terminal with standard setting is switched to 2-fold oversampling (blue) and the optional PDO "StartTimeNextLatch" is added (red):



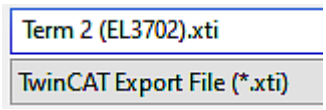
The two methods for exporting and importing the modified terminal referred to above are demonstrated below.

4.2.8.2 Procedure within TwinCAT with xti files

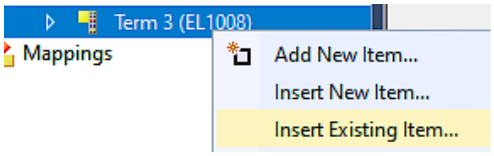
Each IO device can be exported/saved individually:



The xti file can be stored:



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



4.2.8.3 Procedure within and outside TwinCAT with sci file

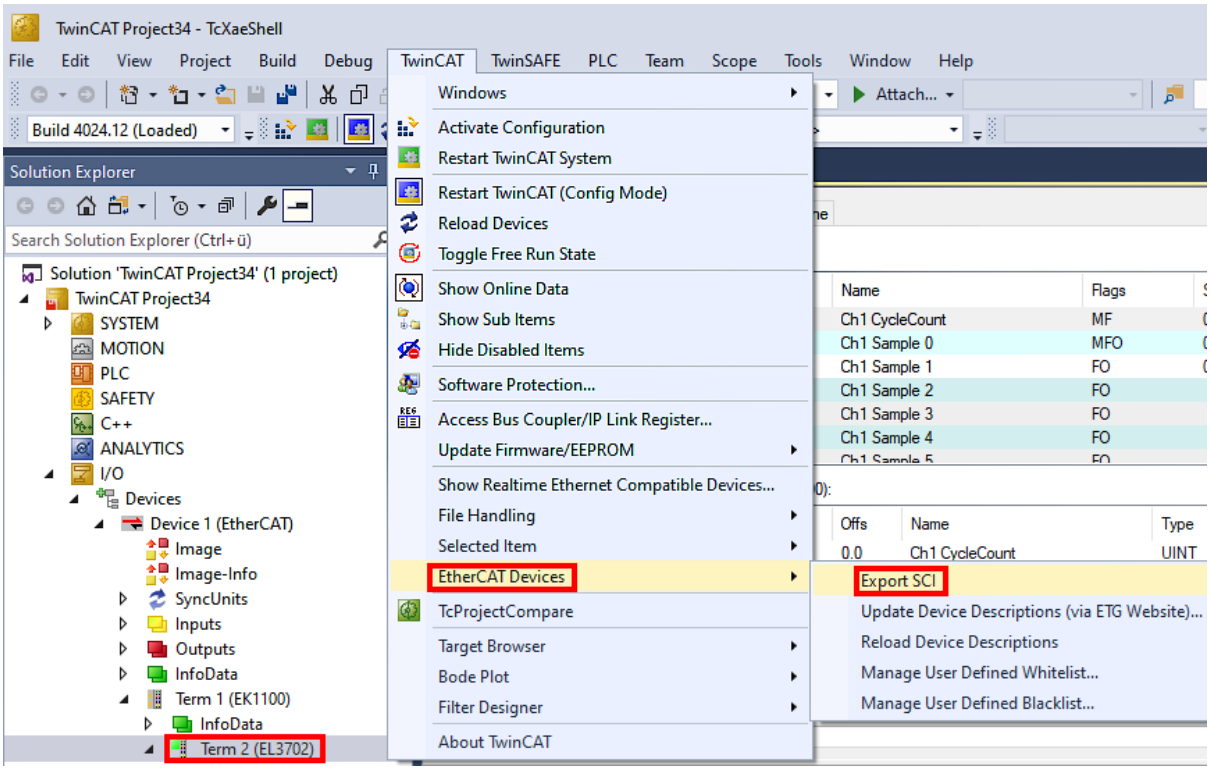
Note regarding availability (2021/01)

The SCI method is available from TwinCAT 3.1 build 4024.14.

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

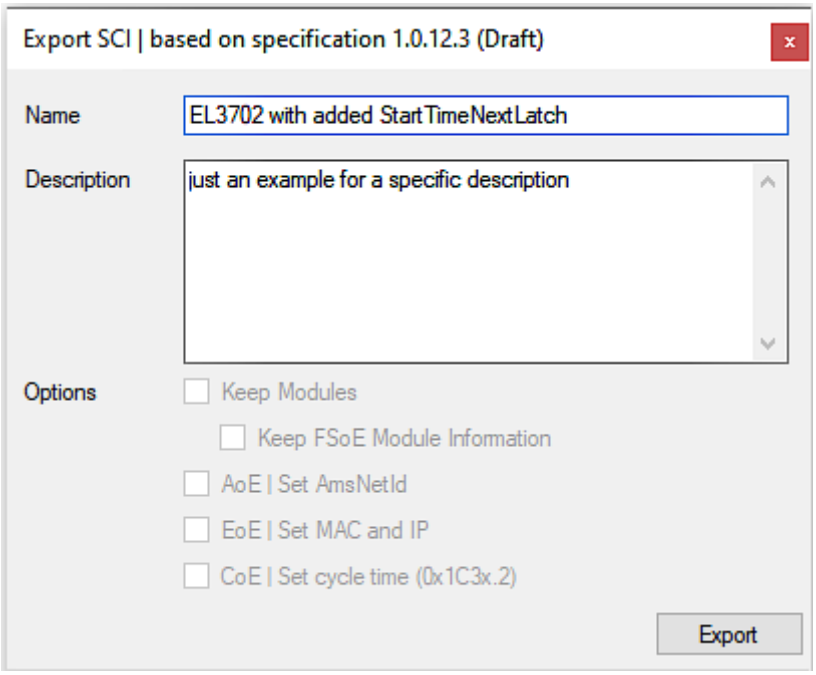
Export:

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



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

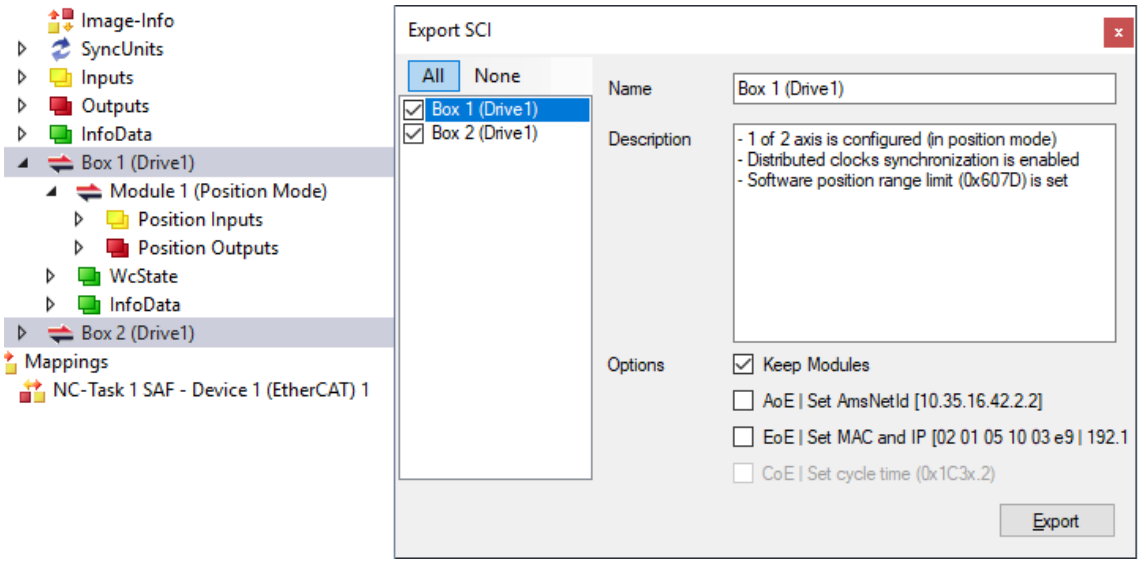
- A description may also be provided:



- Explanation of the dialog box:

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

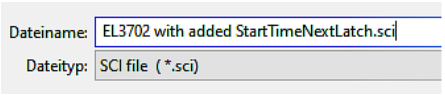
- A list view is available for multiple selections (*Export multiple SCI files)*:



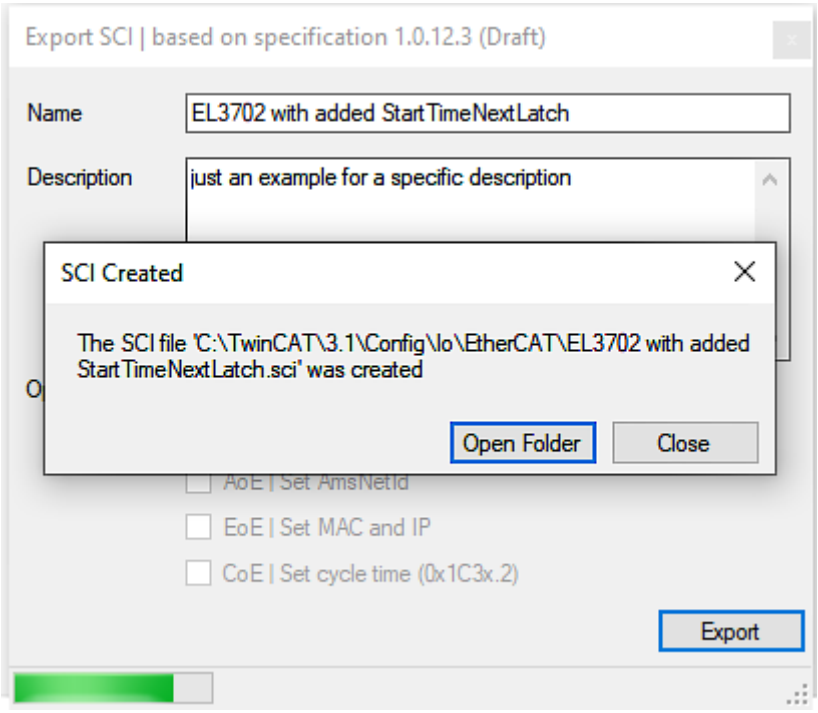
- Selection of the slaves to be exported:
  - All:
    All slaves are selected for export.

- None:
  All slaves are deselected.

- The sci file can be saved locally:

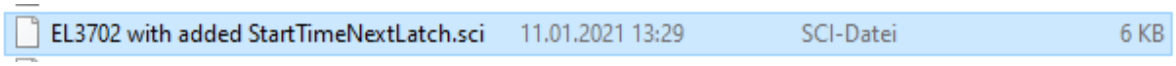


- The export takes place:

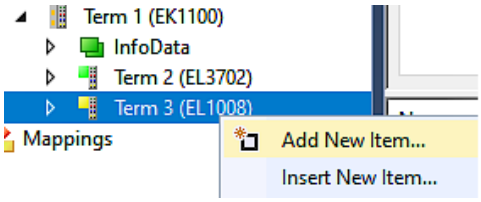


Import

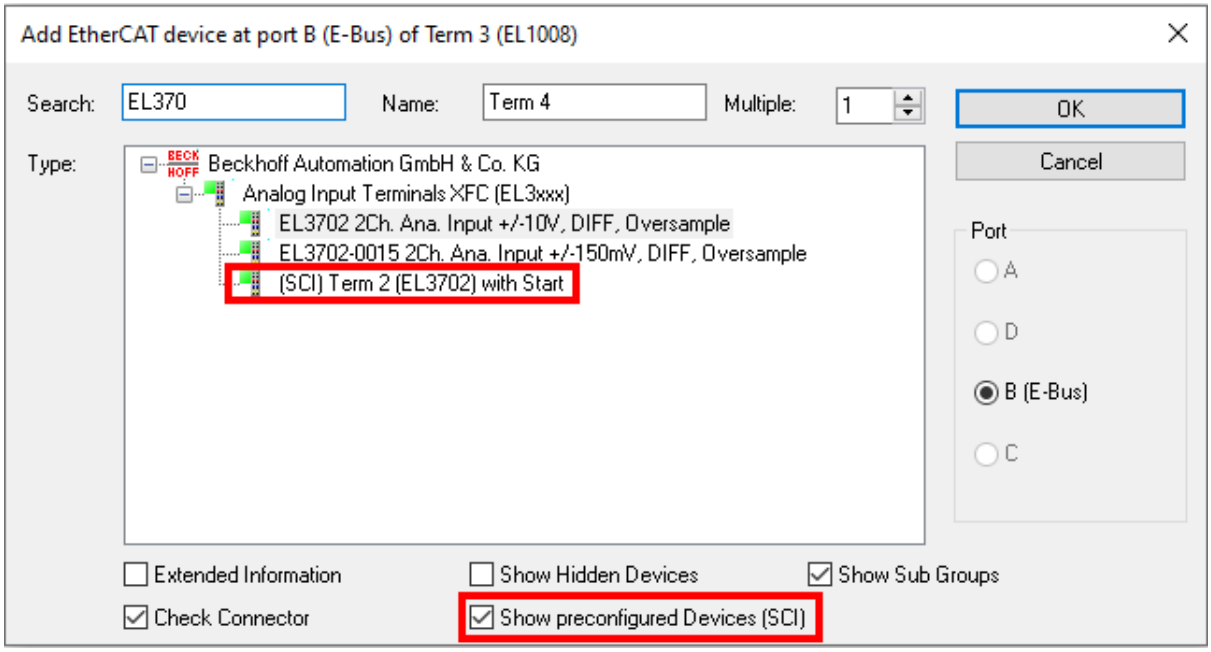
- An sci description can be inserted manually into the TwinCAT configuration like any normal Beckhoff device description.
- The sci file must be located in the TwinCAT ESI path, usually under:
  C:\TwinCAT\3.1\Config\Io\EtherCAT



- Open the selection dialog:

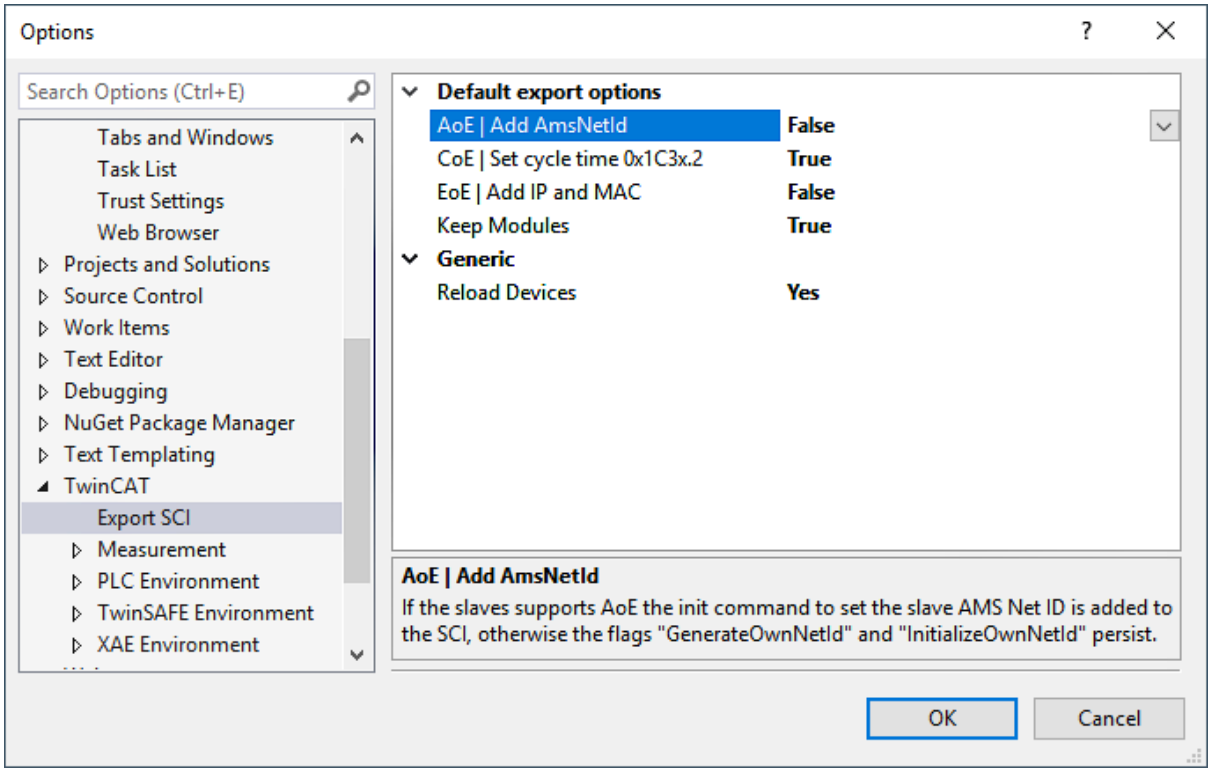


- Display SCI devices and select and insert the desired device:



Additional Notes

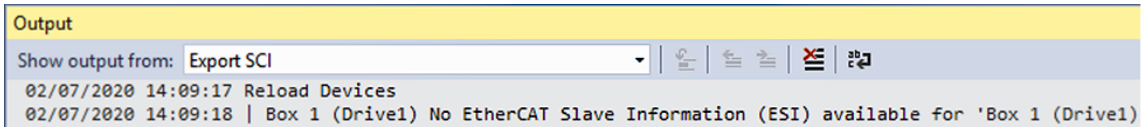
- Settings for the SCI function can be made via the general Options dialog (Tools → Options → TwinCAT → Export SCI):



Explanation of the settings:

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

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



4.3 General Commissioning Instructions for an EtherCAT Slave

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

Diagnosis in real time: WorkingCounter, EtherCAT State and Status

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

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

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

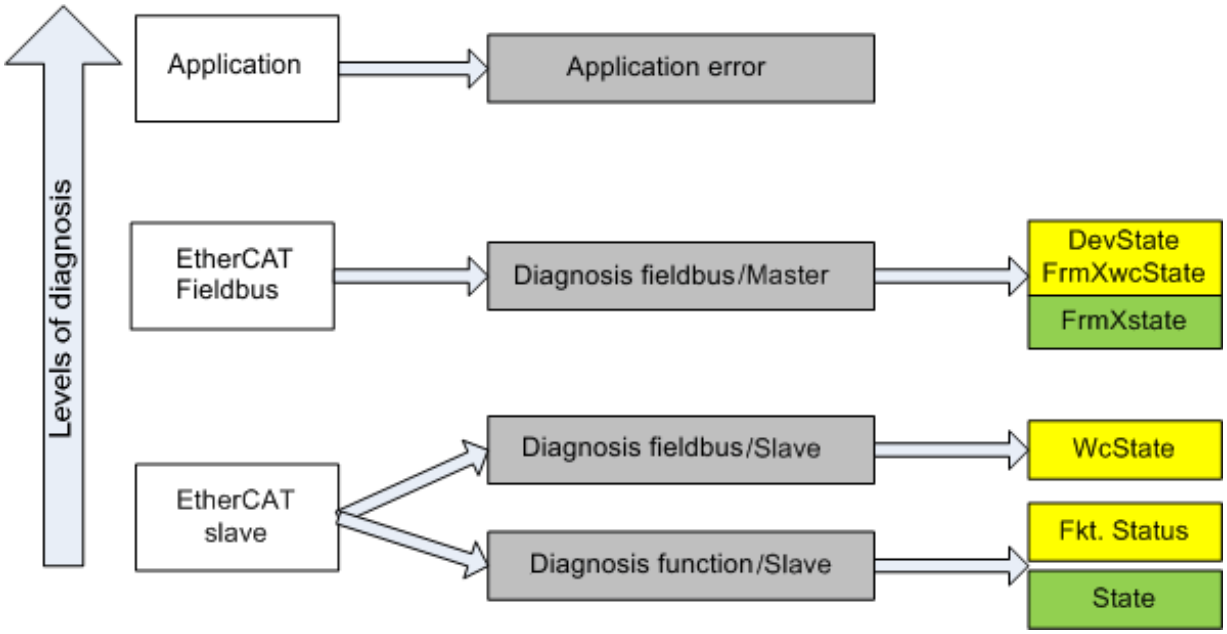


Fig. 118: Selection of the diagnostic information of an EtherCAT Slave

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.

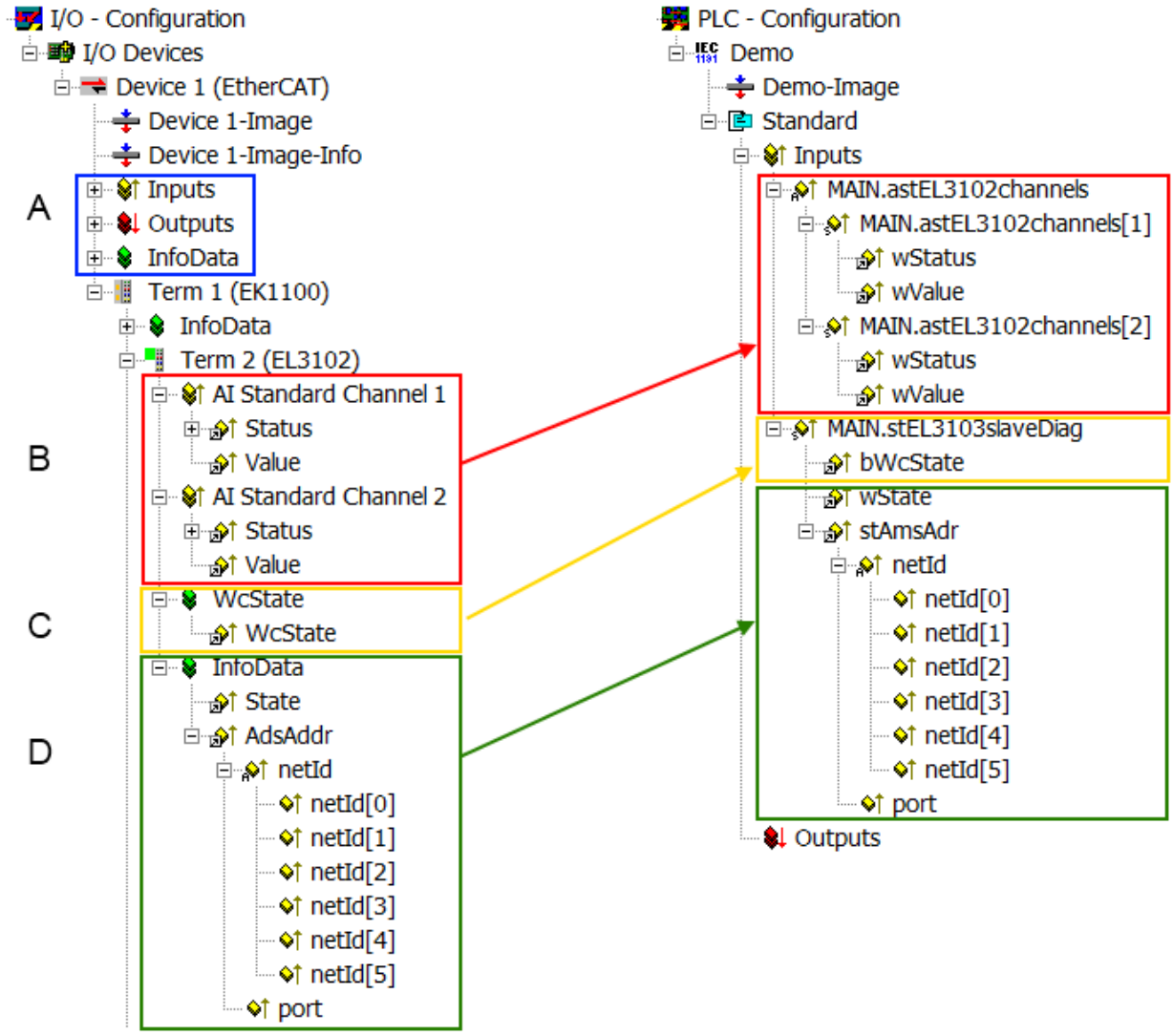


Fig. 119: Basic EtherCAT Slave Diagnosis in the PLC

The following aspects are covered here:

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

NOTICE

Diagnostic information

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

CoE Parameter Directory

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

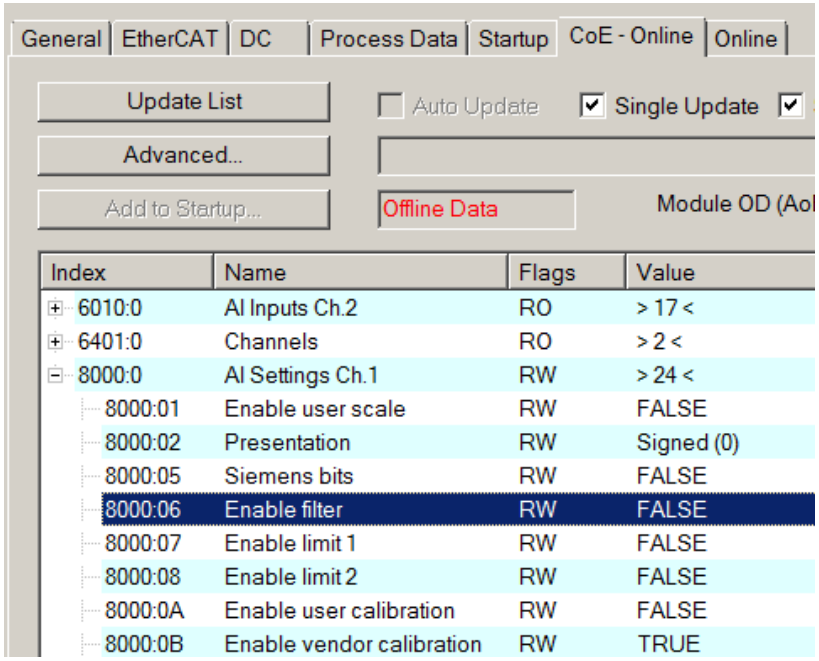


Fig. 120: EL3102, CoE directory

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.

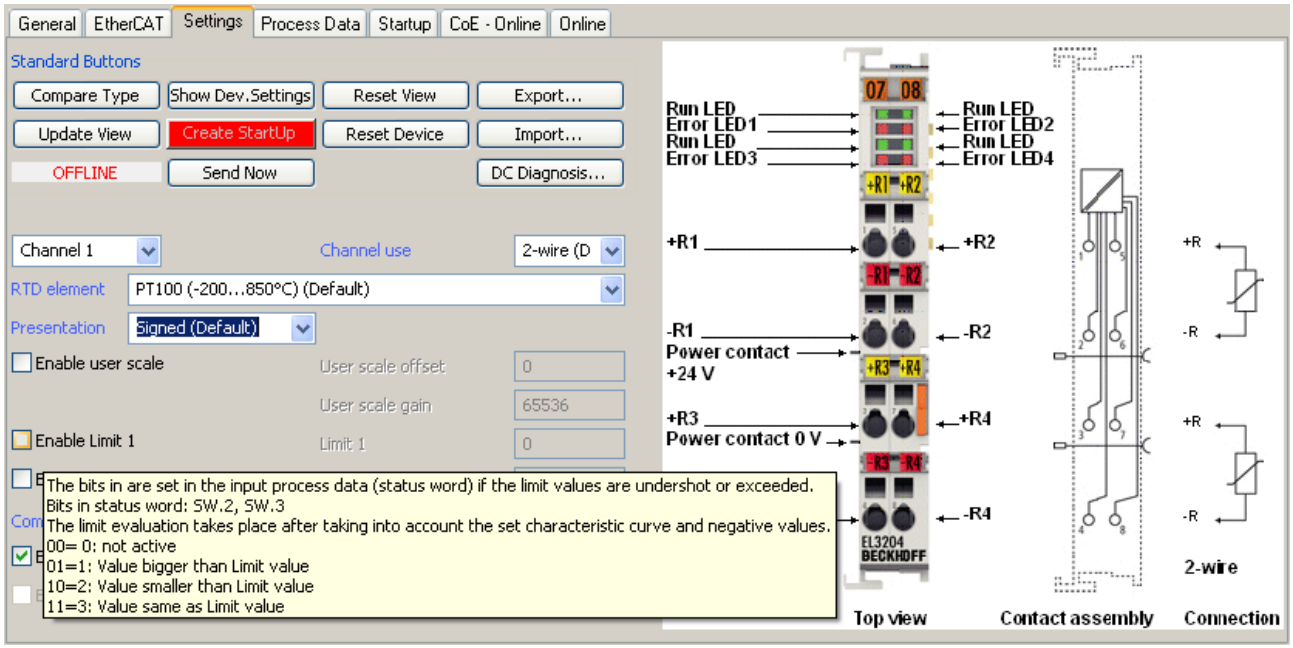


Fig. 121: Example of commissioning aid for a EL3204

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" 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.

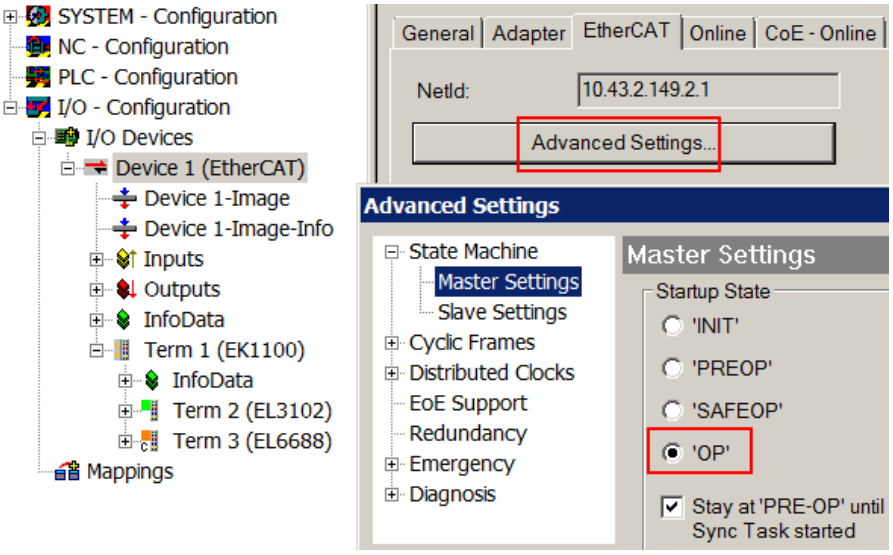


Fig. 122: Default behaviour of the System Manager

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

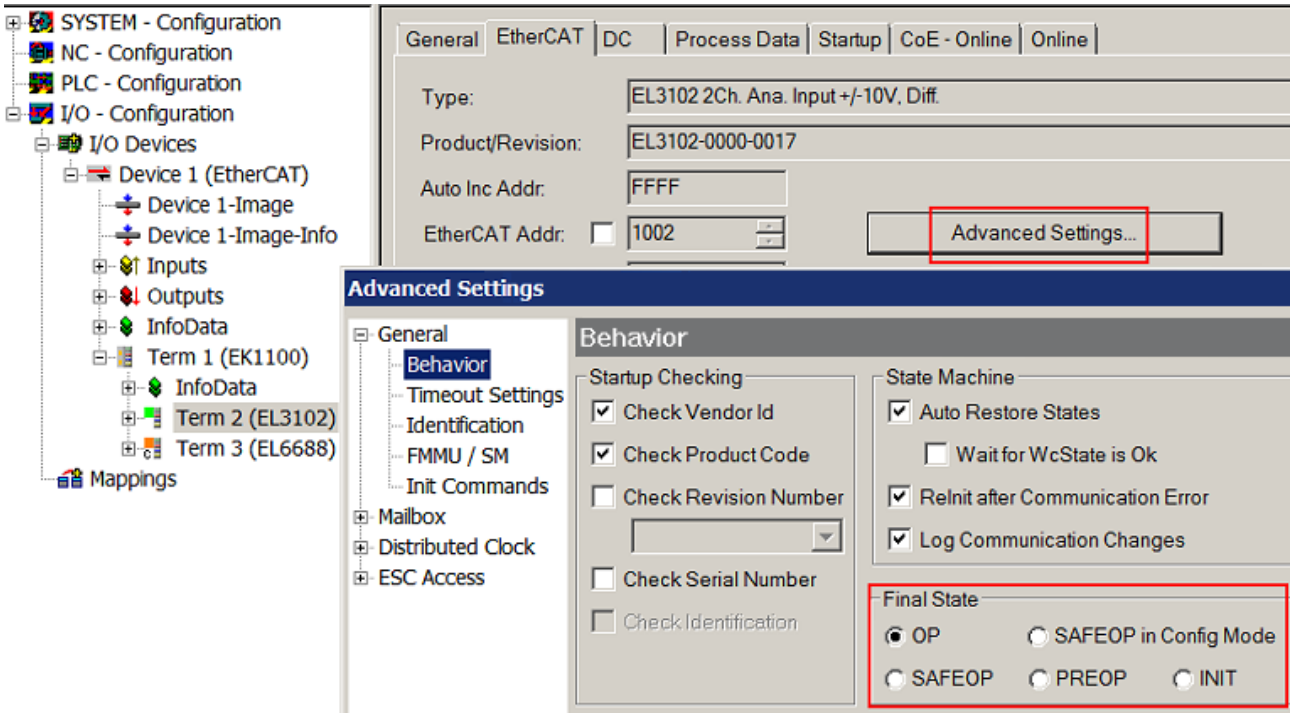


Fig. 123: Default target state in the Slave

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.

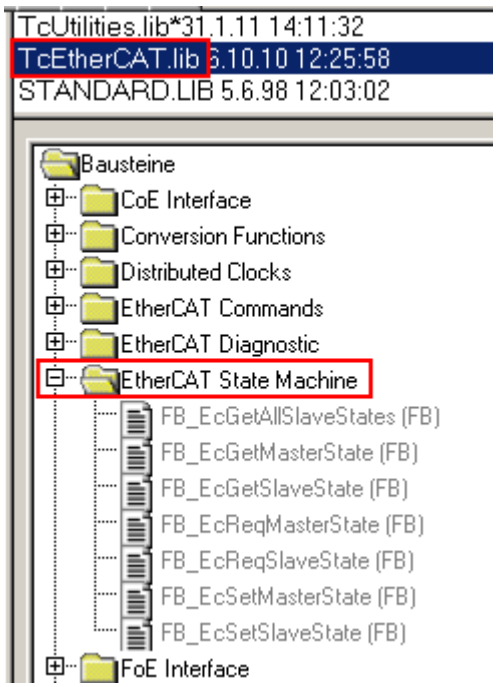
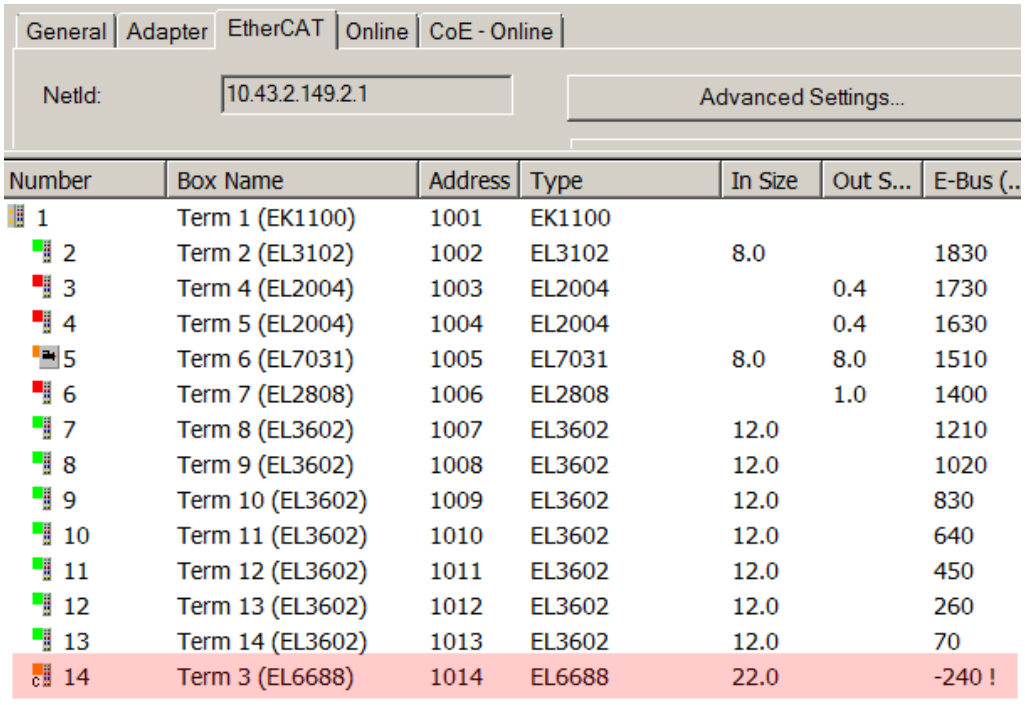


Fig. 124: PLC function blocks

Note regarding E-Bus current

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

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



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

Fig. 125: 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:

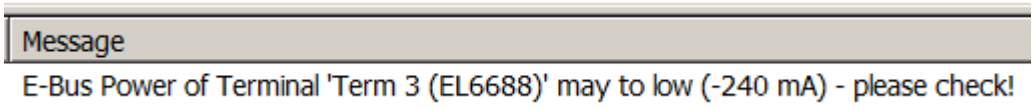


Fig. 126: Warning message for exceeding E-Bus current

NOTICE

Caution! Malfunction possible!

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

5 EL2502-0010 - Commissioning and operation

The pulse width and frequency of a 24 V_{DC} binary signal with a maximum load capacity of 0.5 A can be changed via the digital **p**ulse **w**idth **m**odulation output (PWM).

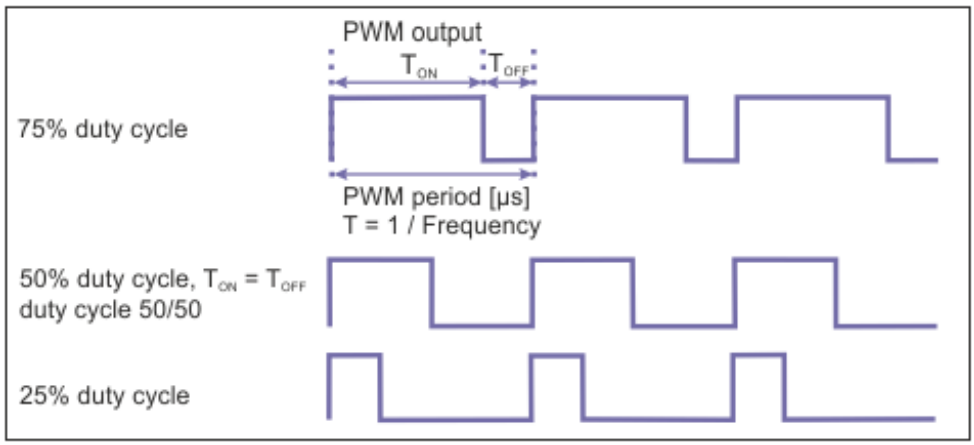


Fig. 127: Setting pulse-pause ratio using the example of duty cycle 25%, 50% and 75%

5.1 Basic operation modes and process data

The EL2502-0010 has three basic operation modes with additional settings for each channel. A detailed description of the operation modes is provided in the respective chapters.

Name	PDO assignment Index - Name (size in byte.bit)	Description
"PWM, DC [▶ 116]":	**Channel 1** 0x1A00 - “PWM Inputs Channel 1” (2.0) 0x1A01 - “PWM Sys Time Channel 1” (8.0) 0x1603 - “PWM Per. DC Outputs Channel 1” (14.0) **Channel 2** Only channel 1 is activated via the Predefined PDO Assignment "PWM DC". The PDOs for channel 2 must also be activated via the SYNC manager (see "Activating the 2nd channel [▶ 115]"). 0x1A02 - “PWM Inputs Channel 2” (2.0) 0x1A03 - “PWM Sys Time Channel 2” (8.0) 0x1607 - “PWM Per. DC Outputs Channel 2” (14.0)	This mode should be used if both the duty cycle and the period value are to be changed quickly in real-time context. • cyclic duty cycle specification via process data "PWM output" (0x7000:11) • cyclic specification of the period value via process data "PWM period" (0x7000:12) • change cyclic specification of a timestamp via process data "DC start time" (0x7000:13) • Switch on PWM output after reaching a timestamp or specifically change/ switch off the state via process data "Activate DC start time" (0x7000:02) = TRUE • Switch off PWM output during current period via process data "Activate DC start time" (0x7000:02) = FALSE
"PWM, Period [▶ 122]":	**Channel 1** 0x1600 - “PWM Outputs Channel 1” (2.0) **Channel 2** 0x1604 - “PWM Outputs Channel 2” (2.0)	This mode should be used if both the duty cycle and the period value are to be changed quickly in real-time context. • cyclic duty cycle specification via process data "PWM output" (0x70n0:11) • cyclic specification of the period value via process data "PWM period" (0x70n0:12)
"PWM [▶ 124]":	**Channel 1** 0x1601 - “PWM Per. Outputs Channel 1” (2.0) **Channel 2** 0x1605 - “PWM Per. Outputs Channel 2” (2.0)	This mode is to be used if only the duty cycle is to be changed quickly in the real-time context. • cyclic duty cycle specification via process data "PWM output" (0x70n0:11) • acyclic specification of the period value via CoE "Period PWM" (0x80n0:15)

1. **Selection of operation mode via "Predefined PDO"**
   The "Predefined PDO Assignment" enables simplified selection of the process data.
   Select the desired function at the bottom of the process data tab.

   ⇨ This automatically enables all required PDOs and disables those that are not required.

2. **Activation of the 2nd channel**
   If the Predefined PDO Assignment “PWM, DC” is selected, only the PDOs for channel 1 are active. Due to the impairment of the time accuracy (see note below), 1-channel operation is recommended for the "PWM, DC" operation mode.
   If required, the following PDOs must be activated for channel 2 via the "Sync Manger" and the "PDO Assignment":
   0x1A02 "PWM Inputs Channel 2"
   0x1A03 "PWM Sys Time Channel 2"
   0x1607 "PWM Per. DC Outputs Channel 2"

3. **Disabling of unused channels**
   Both channels are active when the Predefined PDO Assignments "PWM" and PWM, Period" are selected. If channels are not used, it is recommended to disable the corresponding channels via the "Sync Manger" and the "PDO Assignment" to avoid possible error messages from the terminal.
   If not required, disable the following PDOs for channel 2:
   for "PWM": disable 0x1604 "PWM Outputs Channel 2"
   for "PWM, Period": disable 0x1605 "PWM Per. Outputs Channel 2".

The temporal accuracy is impaired if the second channel is activated

If a second channel is activated by selecting one of the PDO assignments 0x1604 to 0x1607 or 0x1A02, 0x1A03, please note that the additional calculations in the microcontroller slightly impair the temporal accuracy.

4. **Display of process data**
   The scope of the assigned process data is displayed in "PDO List", the contents of the individual PDOs in "PDO Content". Excluded PDOs are highlighted in gray in “PDO Assignment”.

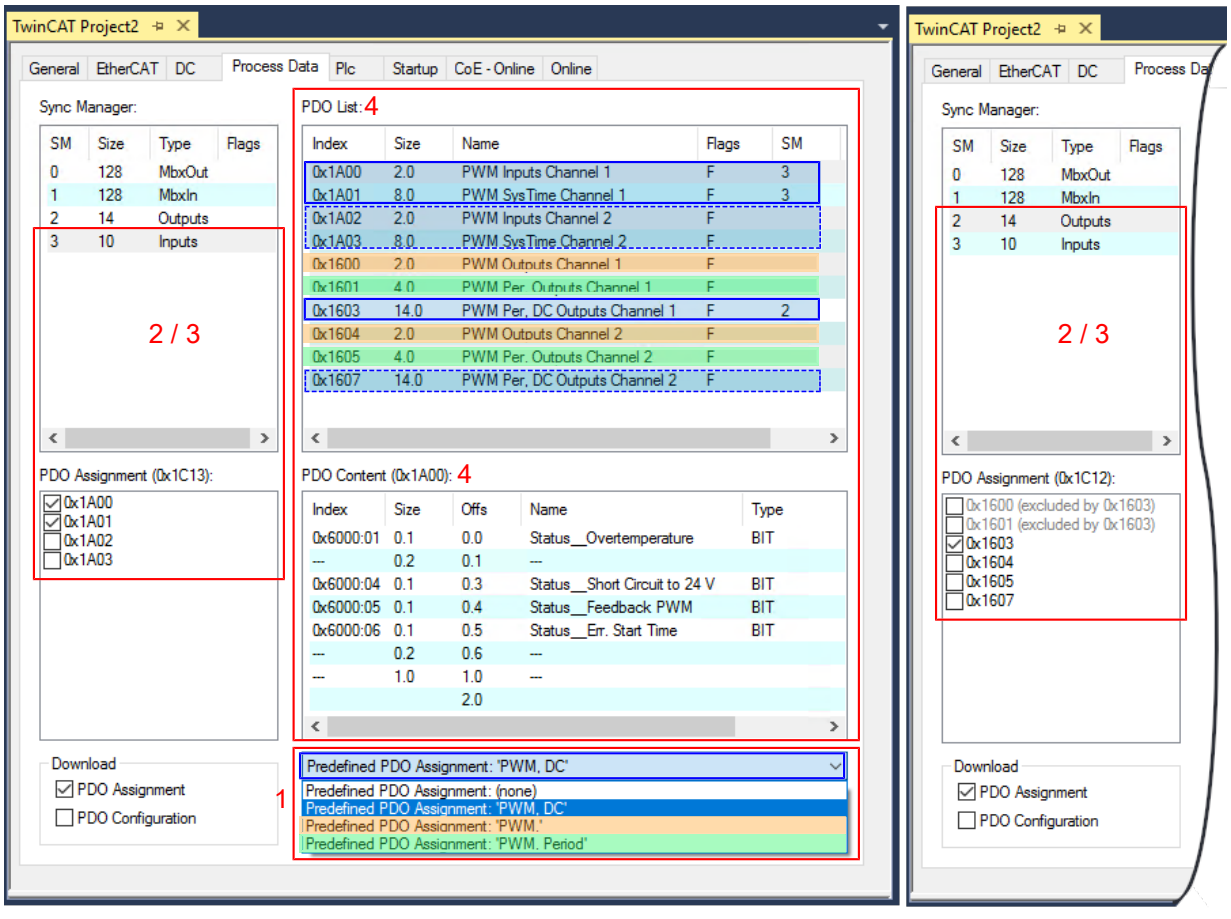


Fig. 128: EL2502-0010 - Process data Selection of predefined PDO "PWM, DC" (default)

5.1.1 Operation mode “PWM, DC”

5.1.1.1 Function and settings

- The duty cycle and the period value (F) can be changed quickly in the real-time context
  - using a timestamp at a specified DC start time [▶ 117] (C) or
  - permanently without a specified time [▶ 120] (B).
- The PWM output can be switched on/off during runtime [▶ 118] (C)
  - Switch on when the DC start time is reached during or after the end of the current period
  - Switch off while the period is running
- DC-synchron output to channel 2 is possible in 1-channel operation [▶ 117] (E).
- Confirmation/error message of the value transfer [▶ 118]
  Successful and failed transfer of the PWM values are displayed (A). The error message and error counter must then be reset (D).
- Exceeding the terminal temperature of 90 °C "Overtemperature" and overload of the outputs "Short Circuit to 24 V" are displayed (A).
- The "Presentation" of the PWM PDO (16-bit) is displayed as a signed integer (G) with the value range: $0...7FFF_{hex}/32767_{dec}$ for 0 ... 100 % duty cycle
- The following additional settings are possible
  - Set safe state in case of bus error (H) (see chapter "Watchdog [▶ 126]")
  - Switch output to high-resistance (I) (see chapter "Tristate [▶ 126]")

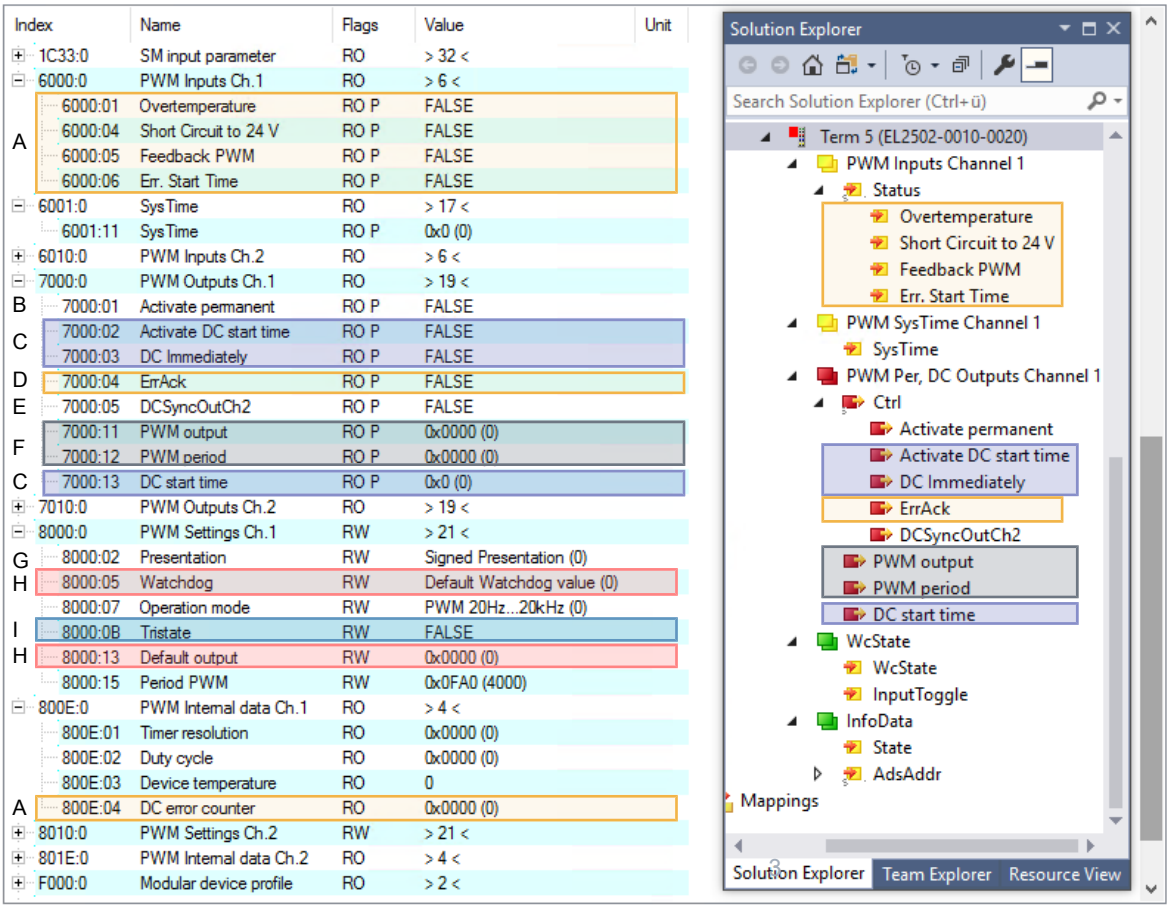


Fig. 129: EL2502-0010 - CoE and process data for Predefined PDO Assignment "PWM, DC"

5.1.1.1.1 Changing PWM period value and duty cycle at DC start time T1

The PWM period value and the duty cycle can be set to predefined values during runtime when a DC timestamp is reached and thus used for synchronization with other processes.
The default values can be activated as described below (n = 0 for channel 1, n = 1 for channel 2).

Specification of duty cycle, period value and start time

- **0x70n0:11 "PWM output":**
  The percentage duty cycle (Duty cycle: T_{ON} / T) is specified in "PWM output" [0 ... 100 %] to change the pulse-pause ratio. The pulse-pause T_{OFF} is calculated automatically as the difference to the 100 % pulse value.

Desired pulse-pause ratio(Duty cycle)	Value range in hex	Value range in dec
100 %	7FFF	32767
50 %	3FFF	16383
0 %	0	0

- **0x70n0:12 "PWM period":**
  Specification of the PWM period in µs (corresponds to a frequency specification).
  The resolution of the output signal depends on the frequency or period value specification.
  The valid value range in this mode corresponds to:
  Output frequency: min. approx. 20 Hz, max. 300 kHz

  The number of steps changes depending on the set frequency. The resolution decreases with decreasing period value or increasing PWM frequency.
  The resolution or step size can be read in CoE object 0x80nE:01 "Timer resolution".

Common specification of the "PWM output" and "PWM period" parameters

To ensure a time-synchronized transition of the PWM parameters, the parameters (0x70n0:11 "PWM output", 0x70n0:12 "PWM period") must always be transferred or changed as a pair.

- **0x70n0:13 "DC start time" (T1)**
  Corresponds to the T1 time, it is a 64-bit DC start time in ns.
  The time in 0x6001:11 "Sys Time" corresponds to the system time of the µController and can be used as a reference point. However, this time is subject to jitter.
- **0x7000:05 "DCSyncOutCh2":**
  If there is no PWM output on channel 2 (PDOs for the second channel are not assigned), it can be used for a DC-synchron output. After activation via 0x7000:05 “DCSyncOutCh2” = TRUE, the output of channel 2 is toggled with the DC event and the switching of the PWM unit with each new valid start time.

Application of values immediately or after the end of the current period

When the specified values are to be applied is defined via

- **0x70n0:03 "DC Immediately" = TRUE** (see example 1 [▶ 119])
  When a DC timestamp is reached, the specified PWM output takes place immediately.
  The current period is aborted. This causes glitches in the output.
- **0x70n0:03 "DC Immediately" = FALSE** (see example 2 [▶ 119])
  When a DC timestamp is reached, the PWM output only takes place after the current period has been completed. This ensures glitch-free conversion of the PWM output.

Activating the transfer, switching PWM output on or off

- **Transferring the values, switching on PWM output**
  The transfer of the values must then be confirmed via "Activate DC start time".
  0x70n0:02 "Activate DC start time" = TRUE
  If this bit is set, a PWM output with the transferred parameters (0x70n0:11 "PWM output", 0x70n0:12 "PWM period") is initiated with each new DC start time.
- **Switching off PWM output**
  0x70n0:02 "Activate DC start time" = FALSE
  The PWM output is stopped while the period is running.

Confirmation of the transferred values

- **0x60n0:05 "Feedback PWM" = TRUE, with correct DC event**
  If the desired "DC start time" (T1) is reached, this is confirmed via bit 0x60n0:05 "Feedback PWM". This bit toggles with every correct DC event and the next event can be activated.
- **0x60n0:06 "Err. Start Time", 0x80nE:04 "DC error counter", if start time is in the past**
  If the "DC start time" set is already in the past, the PWM output takes place immediately, taking into account the conditions set in 0x70n0:13 "DC Immediately".
  - The transfer is reported via 0x60n0:06 "Err. Start time" = TRUE and
  - the error counter is incremented under 0x80nE:04 "DC error counter".
  - The error counter is reset and the error message acknowledged by setting 0x70n0:04 "ErrAck" = TRUE.

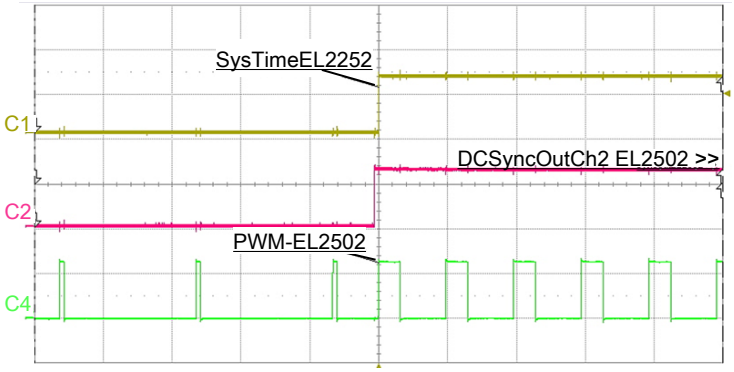
5.1.1.1.1.1 Examples of the "PWM, DC" operation mode

Legend for the diagrams in the following examples:

C1: EL2252 DC-Output Ch1 with the same "Time Stamp" as EL2502-0010
C2: EL2502-0010 Output of the process data "DCSyncOutCh2" (0x7000:05)
C4: PWM signal output to channel 1

Example 1: "DC Immediately" = TRUE

When the DC timestamp is reached, the specified PWM output takes place immediately. The current period is aborted, resulting in "glitches" in the output.

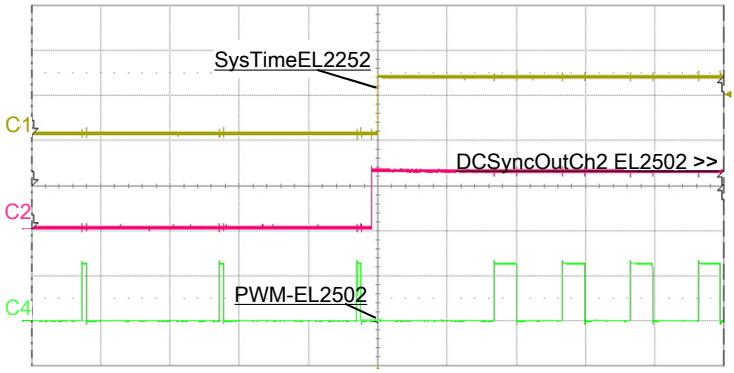


- "PWM output" = new duty cycle value
- "PWM period" = new period value
- "DC start time" = new start time value (T1)
- "DCSyncOutCh2" = TRUE
- "DC Immediately" = TRUE
- "Activate DC start time" = TRUE

Fig. 130: Diagram “DC Immediately” = TRUE

Example 2: "DC Immediately" = FALSE

When the DC timestamp is reached, the PWM output only takes place after the current period has been completed. The PWM output is "glitch-free".

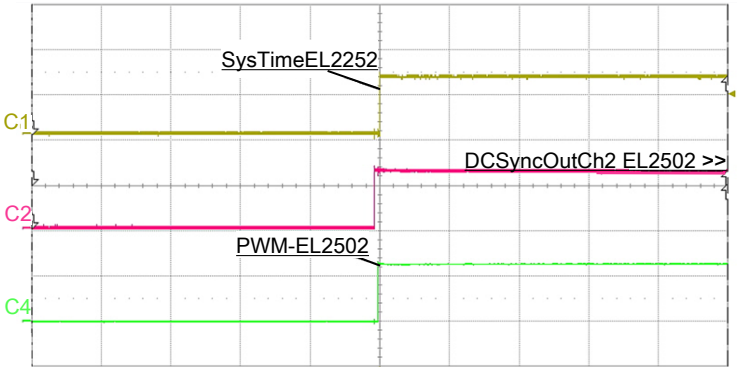


- "PWM output" = new duty cycle value
- "PWM period" = new period value
- "DC start time" = new start time value (T1)
- "DCSyncOutCh2" = TRUE
- "DC Immediately" = FALSE
- "Activate DC start time" = TRUE

Fig. 131: Diagram "DC Immediately" = FALSE

Example 3: "DC Immediately" = TRUE, duty cycle changes from 0 to 100%

When the DC timestamp is reached, the duty cycle is immediately changed from 0 to 100%.



- "PWM output" = new duty cycle value
  Duty cycle changes from 0 to 100%
- "PWM period" = new period value
- "DC start time" = new start time value (T1)
- "DCSyncOutCh2" = TRUE
- "DC Immediately" = TRUE
- "Activate DC start time" = TRUE

Fig. 132: Diagram "DC Immediately" = TRUE, duty cycle changes from 0 to 100 %

5.1.1.1.2 Note on advanced DC settings

EL2502-0010 only distributed clocks device in the EtherCAT system

If the EL2502-0010 is the only distributed clocks device in the EtherCAT system, the operation mode must also be enabled via the "Advanced Settings" (see following figure).

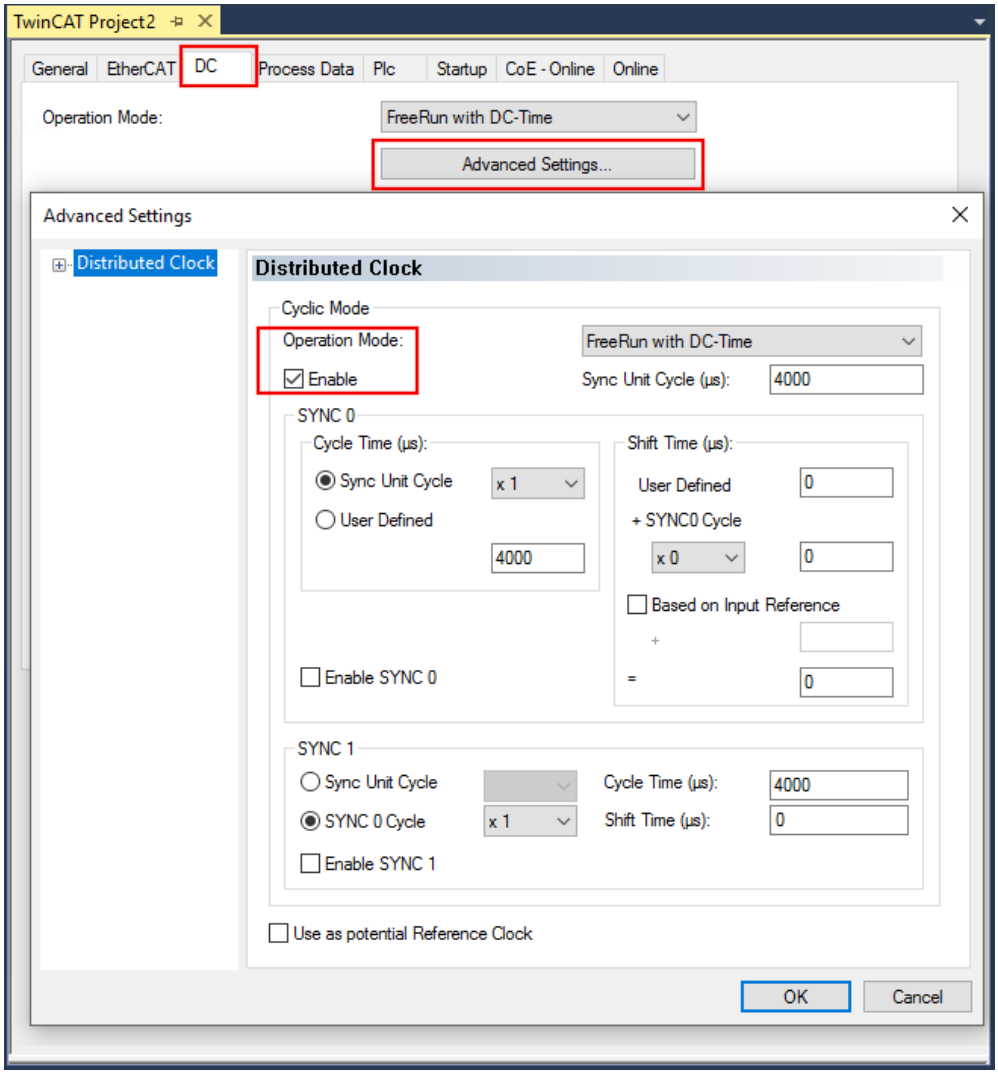


Fig. 133: Enabling "FreeRun with DC-Time" operation mode via Advanced settings in the "DC" tab

5.1.1.1.3 Permanent PWM output, without time specification

A permanent PWM output without time specification is carried out via

- 0x70n0:01 "Activate permanent" = TRUE
  The PWM output is permanent, with the specified parameters (0x70n0:11 "PWM output", 0x70n0:12 "PWM period"). It is not necessary to specify the "DC Start time".
  The PWM parameters (0x70n0:11 "PWM output", 0x70n0:12 "PWM period") can be changed cyclically.

5.1.1.2 Process data assignment in the "PWM, DC" operation mode

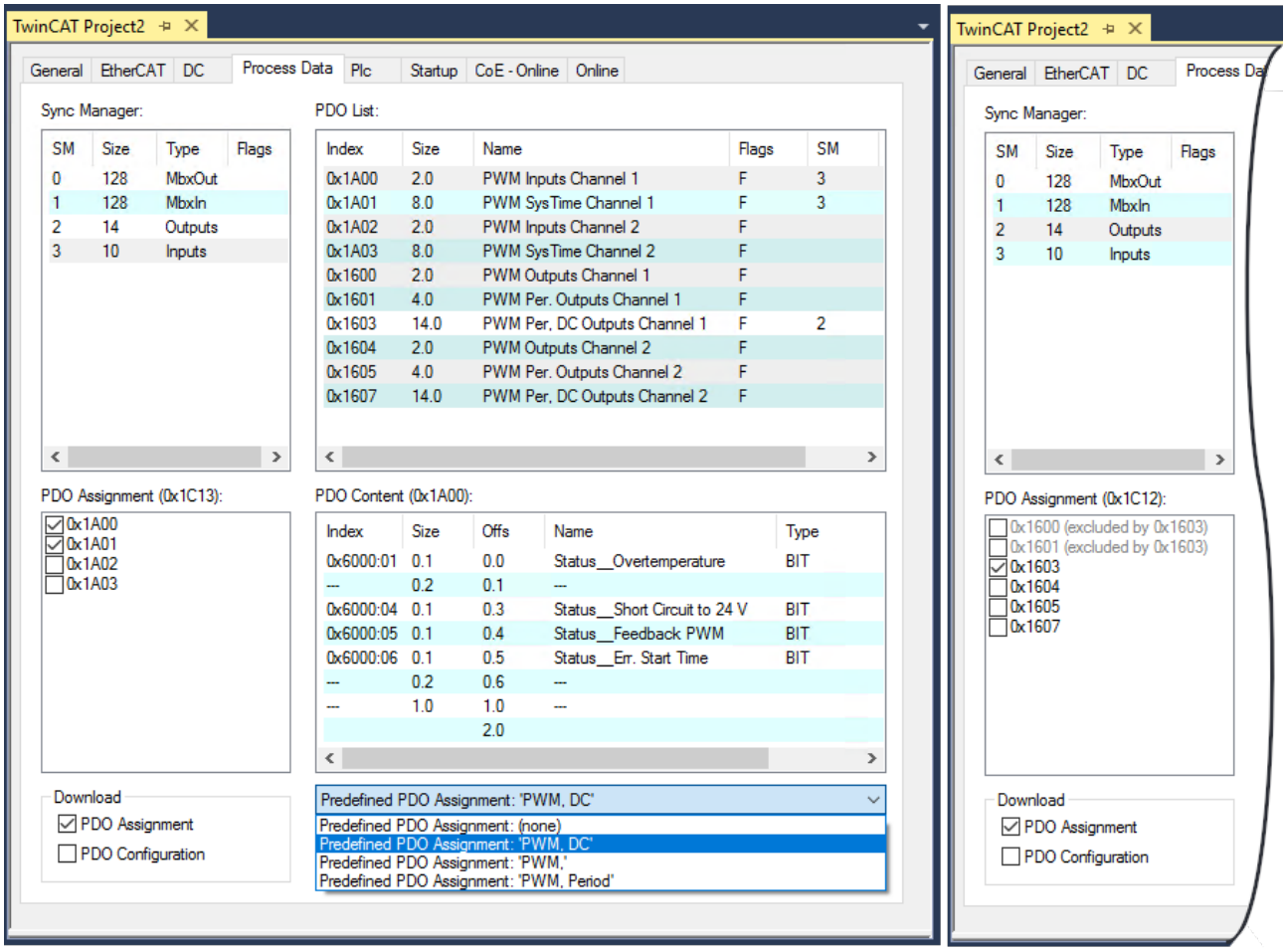


Fig. 134: Process data “PWM, DC”

Predefined PDO "PWM, DC"	
PDOs for channel 1 (index - name)	**Size (byte.bit)**
0x1A00 - "PWM Inputs Channel 1" 0x1A01 - "PWM Sys Time Channel 1"	(2.0) (8.0)
0x1603 - "PWM Per. DC Outputs Channel1"	(14.0)

Process data for channel 2 must be activated separately (see Process data selection [▶ 115])	
PDOs for channel 2 (index - name)	**Size (byte.bit)**
0x1A02 - "PWM Inputs Channel 2" 0x1A03 - "PWM Sys Time Channel 2"	(2.0) (8.0)
0x1607 - "PWM Per. DC Outputs Channel2"	(14.0)

PWM = **P**ulse **W**idth **M**odulation
The contents for the process data objects can be found in the object description (0x16xx [▶ 134], 0x1Axx [▶ 136])

Notice Only the combinations of PDOs described above are permitted. Other settings can lead to irregular process data and error messages in the TwinCAT System Manager Logger window.

5.1.2 Operation mode "PWM, Period"

Function and settings

- This mode allows the duty cycle and the period value (A) to be changed quickly in a real-time context.
- The following additional settings are possible
  - Set safe state in case of bus error (B) (see chapter "Watchdog [▶ 126]")
  - Switch output to high-resistance (C) (see chapter "Tristate [▶ 126]")

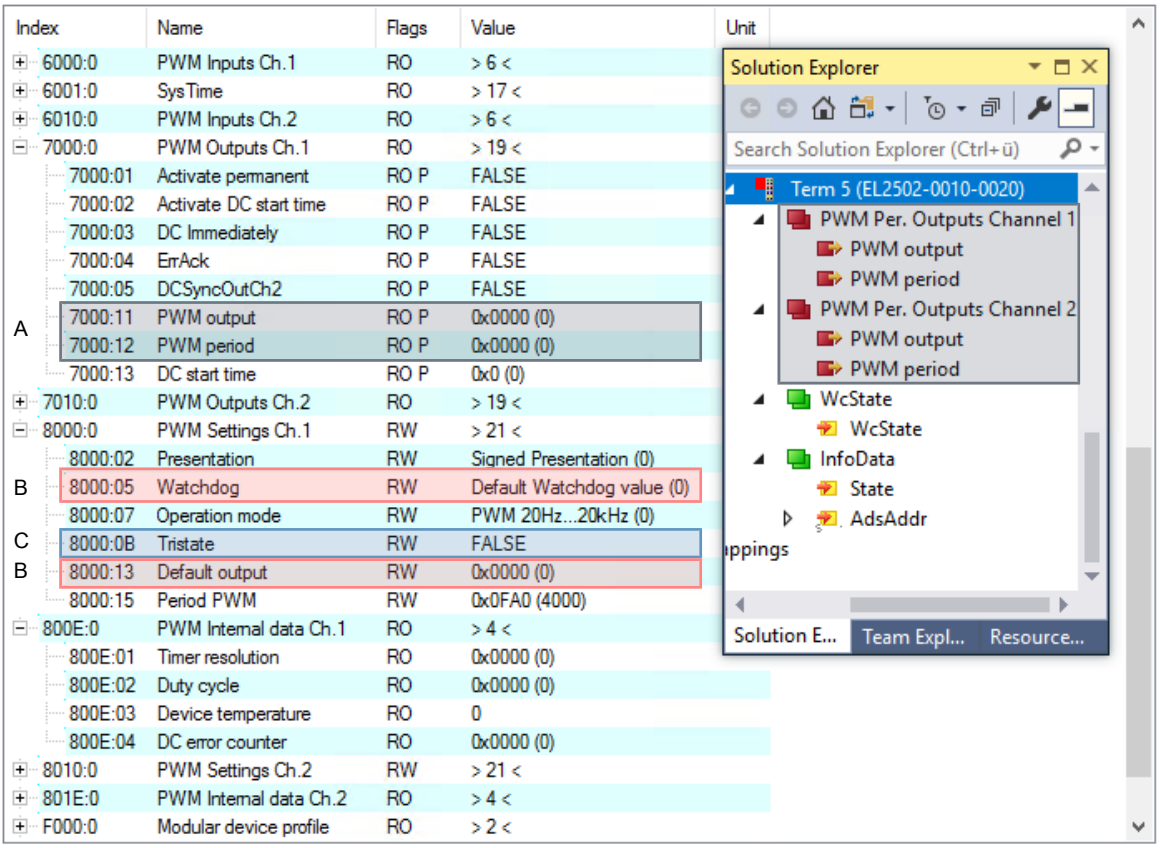


Fig. 135: EL2502-0010 - CoE and process data for Predefined PDO Assignment "PWM, Period"

- **0x70n0:11 "PWM output":**
  The percentage duty cycle (Duty cycle: T_{ON} / T) is specified in "PWM output" [0 ... 100 %] to change the pulse-pause ratio. The pulse-pause T_{OFF} is calculated automatically as the difference to the 100 % pulse value.

Desired pulse-pause ratio(Duty cycle)	Value range in hex	Value range in dec
100 %	7FFF	32767
50 %	3FFF	16383
0 %	0	0

- **0x70n0:12 "PWM period":**
  Specification of the PWM period in µs (corresponds to a frequency specification).
  The resolution of the output signal depends on the frequency or period value specification.
  The valid value range in this mode corresponds to:
  Output frequency: min. approx. 20 Hz, max. 300 kHz

  The number of steps changes depending on the set frequency. The resolution decreases with decreasing period value or increasing PWM frequency.
  The resolution or step size can be read in CoE object 0x80nE:01 "Timer resolution".

After changing the PWM values (0x70n0:11 "PWM output" and 0x70n0:12 "PWM period"), the changed PWM output only takes place after the current period has been completed.

5.1.2.1 Process data assignment in the "PWM, Period" operation mode

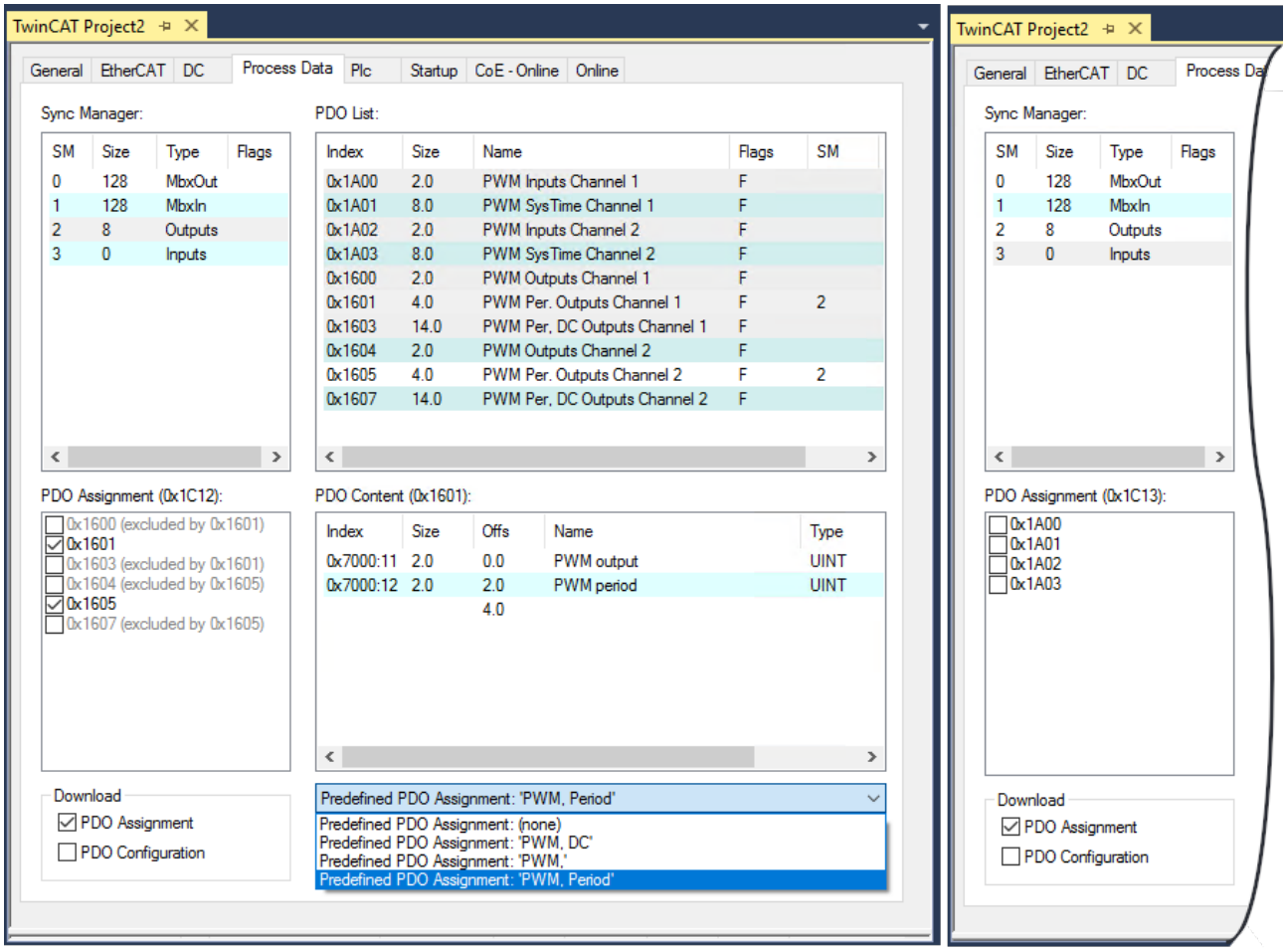


Fig. 136: Process data "PWM, Period"

Predefined PDO "PWM Period" (Index - Name)	Size (byte.bit)
0x1601 - "PWM Per. Outputs Channel 1"	(4.0)
0x1605 - "PWM Per. Outputs Channel 2" Process data for channel 2 should be disabled if not required (see Process data selection [▶ 115])	(4.0)

PWM = **P**ulse **W**idth **M**odulation
The contents for the process data objects can be found in the object description (0x16xx [▶ 134], 0x1Axx [▶ 136])

Notice **Only the combinations of PDOs described above are permitted. Other settings can lead to irregular process data and error messages in the TwinCAT System Manager Logger window.**

5.1.3 PWM” operation mode

Function and settings

- This mode allows the duty cycle to be changed quickly in a real-time context (A).
- In this mode, the period value can only be changed acyclically via the CoE (D).
- The following additional settings are possible:
  - Set safe state in case of bus error (B) (see chapter "Watchdog [▶ 126]")
  - Switch output to high-resistance (C) (see chapter "Tristate [▶ 126]")

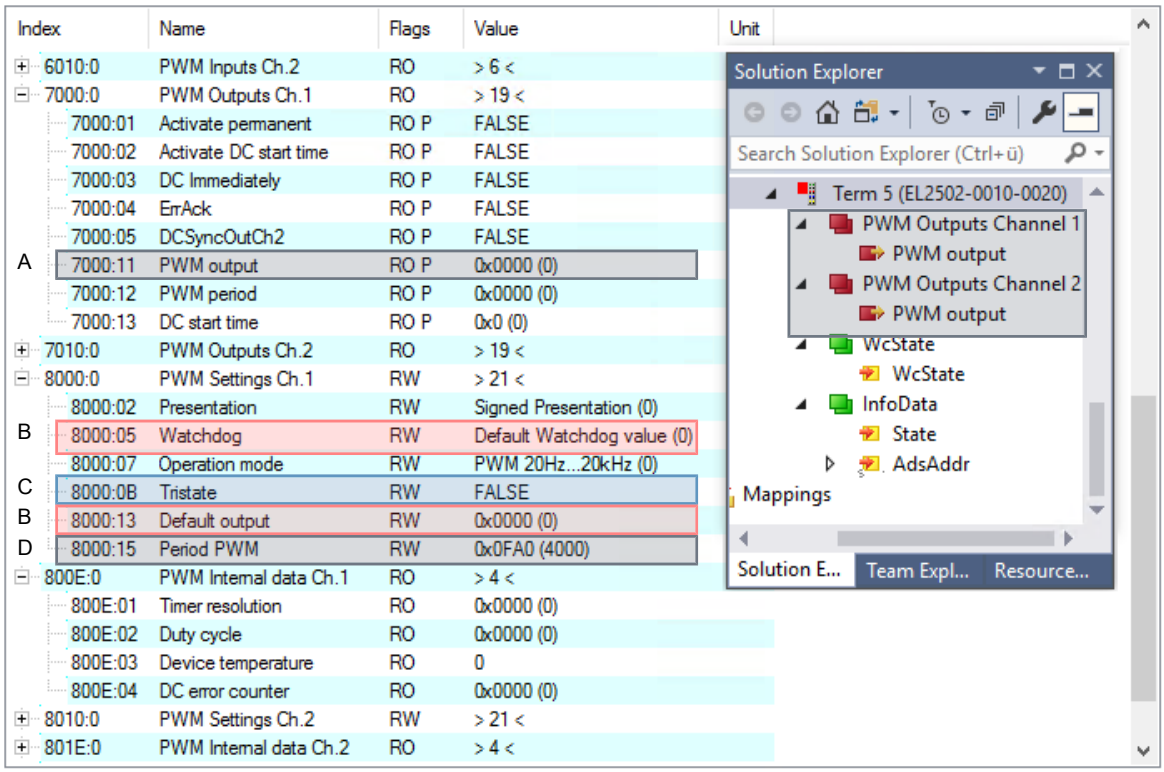


Fig. 137: EL2502-0010 - CoE and process data for Predefined PDO "PWM"

- **0x70n0:11 "PWM output":**
  The percentage duty cycle (Duty cycle: T_{ON} / T) is specified in "PWM output" [0 ... 100 %] to change the pulse-pause ratio. The pulse-pause T_{OFF} is calculated automatically as the difference to the 100 % pulse value.

Desired pulse-pause ratio(Duty cycle)	Value range in hex	Value range in dec
100 %	7FFF	32767
50 %	3FFF	16383
0 %	0	0

- **0x80n0:07 "Operation mode", 0x80n0:15 "Period PWM"**
  - The operation range "PWM 20Hz...20kHz" is available via object 0x80n0:07 "Operation mode".
    Notice The specification in the drop-down menu differs from the actual valid value range. The valid value range is approx. 20 Hz ... 500 kHz.
  - Object 0x80n0:15 "Period PWM" is used to specify the PWM period in µs.
  - The number of steps changes depending on the set frequency. The resolution decreases with decreasing period value or increasing PWM frequency. The resolution or step size can be read in CoE object 0x80nE:01 "Timer resolution".

5.1.3.1 Process data assignment in the "PWM" operation mode

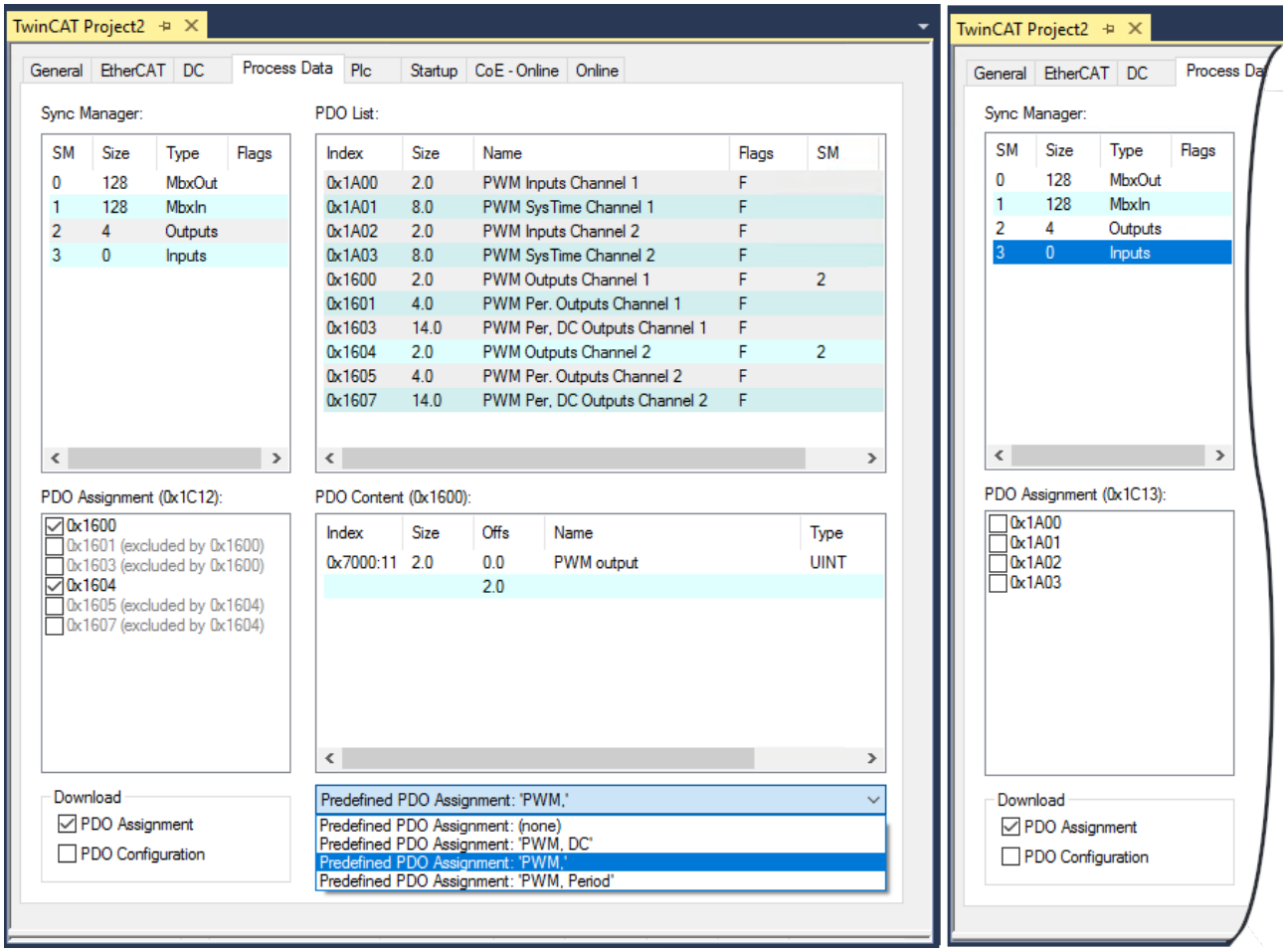


Fig. 138: Process data “PWM”

Predefined PDO "PWM" (Index - Name)	Size (byte.bit)
0x1600 - "PWM Outputs Channel 1"	(2.0)
0x1604 - "PWM Outputs Channel 2" Process data for channel 2 should be disabled if not required (see Process data selection [▶ 115])	(2.0)

PWM = **P**ulse **W**idth **M**odulation
The contents for the process data objects can be found in the object description (0x16xx [▶ 134], 0x1Axx [▶ 136])

Notice Only the combinations of PDOs described above are permitted. Other settings can lead to irregular process data and error messages in the TwinCAT System Manager Logger window.

5.2 Safe state in the event of a bus error (watchdog)

The setting in index 0x80n0:05 "Watchdog" fixes whether the output should assume a safe state in the event of a bus error. The following options are available (n = 0 for channel 1, n = 1 for channel 2):

Desired state of the output in the event of a bus error	Default in 0x80n0:05 "Watchdog"
set to specified value	**0: default watchdog value** The specified pulse width "PWM Output" of the output is set to the value defined in index 0x80n0:13 "Default output". The period value "PWM Period" remains unchanged in the meantime.
Retain current value	**2: Last value** The output retains the current state.

NOTICE

Watchdog parameterization can trigger unintended actions!

Please refer to chapter "Notes for setting the watchdog"!

5.3 Switch channel to high-resistance (Tristate)

Each channel can be switched individually in high-resistance mode via index 0x80n0:0B [▶ 128] (*Tristate*).

This state ensures that the respective output behaves as if it were not connected. This means that the outputs of other outputs/devices that are connected in parallel with these are not affected. The associated output takes on the same output voltage as the other active devices.

If the output is in Tristate mode, the respective Tristate LED lights up yellow.

5.4 Object description and parameterization

EtherCAT XML Device Description

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

Parameterization via the CoE list (CAN over EtherCAT)

The EtherCAT device is parameterized via the CoE-Online tab [▶ 94] (double-click on the respective object) or via the Process Data tab [▶ 91](allocation of PDOs). Please note the following general CoE notes 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 [▶ 154]” for resetting changes

5.4.1 Restore object

Index 1011 Restore default parameters

Index (hex)	Name	Meaning	Data type	Flags	Default
1011:0	Restore default parameters [▶ 154]	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})

5.4.2 0x80xx | Configuration data

Index 8000 PWM Settings Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
8000:0	PWM Settings Ch.1	Max. subindex	UINT8	RO	0x15 (21_{dec})
8000:02	Presentation	• **0: Signed presentation** The value range of the output index 0x7000:11 [▶ 130] is represented as a 16-bit signed integer. 100 % duty cycle corresponds to 0x7FFF, 50 % duty cycle to 0x3FFF. 0% is used for the negative range. • **1: Unsigned presentation** This setting has no effect on the display of the output.	BIT3	RW	0x00 (0_{dec})
8000:05	Watchdog	• **0: Default watchdog value** The default value ("Default output“ Index 0x8000:13) is active. • **2: “Last value” active:** The last process data is output in the event of an error (watchdog drop).	BIT2	RW	0x00 (0_{dec})
8000:07	Operation mode	• **0: PWM 20 Hz..20 kHz** Period setting 1 unit = 1000 ns = 1 µs Deviating from the designation in the selection list, values in the range 20 Hz ... 500 kHz are possible (see chapter "PWM operation mode [▶ 124]")	BIT2	RW	0x00 (0_{dec})
8000:0B	Tristate	• 0: Output activated • 1: The output is switched to high-resistance in Tristate mode [▶ 126]. The Tristate LED lights up yellow.	BOOLEAN	FW	0x00 (0_{dec})
8000:13	Default output	Output value in watchdog case, if activated via index 0x8000:05 = 0 “Default watchdog value”	UINT16	RW	0x0000 (0_{dec})
8000:15	Period PWM	In the "PWM" operation mode, the period value is specified here with a resolution of 1 µs (default). The default setting is 4000 µs (corresponds to 250 Hz).	UINT16	RW	0x0FA0 (4000_{dec})

Index 8010 PWM Settings Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
8010:0	PWM Settings Ch.2	Max. subindex	UINT8	RO	0x15 (21_{dec})
8010:02	Presentation	• **0: Signed presentation** The value range of the output (index 0x7010:11 [▶ 130]) is represented as a 16-bit signed integer. 100 % duty cycle corresponds to 0x7FFF, 50 % duty cycle to 0x3FFF. 0% is used for the negative range. • **1: Unsigned presentation** This setting has no effect on the display of the output.	BIT3	RW	0x00 (0_{dec})
8010:05	Watchdog	• **0: Default watchdog value** The default value ("Default output“ Index 0x8010:13) is active. • **2: Last value** The last process data is output in the event of an error (watchdog drop).	BIT2	RW	0x00 (0_{dec})
8010:07	Operation mode	• **0: PWM 20 Hz..20 kHz** Period setting 1 unit = 1000 ns = 1 µs Deviating from the designation in the selection list, values in the range 20 Hz ... 500 kHz are possible (see chapter "PWM operation mode [▶ 124]")	BIT2	RW	0x00 (0_{dec})
8010:0B	Tristate	• 0: Output activated • 1: The output is switched to high-resistance in Tristate mode [▶ 126]. The Tristate LED lights up yellow.	BOOLEAN	FW	0x00 (0_{dec})
8010:13	Default output	Output value in watchdog case, if activated via index 0x8010:05 = 0 “Default watchdog value”	UINT16	RW	0x0000 (0_{dec})
8010:15	Period PWM	In the "PWM" operation mode, the period value is specified here with a resolution of 1 µs (default). The default setting is 4000 µs (corresponds to 250 Hz).	UINT16	RW	0x0FA0 (4000_{dec})

5.4.3 0x60xx | Input data

Index 6000 PWM Inputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
6000:0	PWM Inputs Ch.1	Max. subindex	UINT8	RO	0x06 (6_{dec})
6000:01	Overtemperature	Overtemperature (temperature above 90 °C)	BOOLEAN	RO	0x00 (0_{dec})
6000:04	Short Circuit to 24 V	Short Circuit to 24 V	BOOLEAN	RO	0x00 (0_{dec})
6000:05	Feedback PWM	Bit toggles with each DC event	BOOLEAN	RO	0x00 (0_{dec})
6000:06	Err. Start Time	“DC start time” (0x7000:13) is in the past	BOOLEAN	RO	0x00 (0_{dec})

Index 6001 SysTime

Index (hex)	Name	Meaning	Data type	Flags	Default
6001:0	SysTime	Max. subindex	UINT8	RO	0x11 (17_{dec})
6001:11	Sys Time	System time of µC (jitter)	ULINT64	RO	0x0000000000000000 (0_{dec})

Index 6010 PWM Inputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
6010:0	PWM Inputs Ch.2	Max. subindex	UINT8	RO	0x05 (5_{dec})
6010:01	Overtemperature	Overtemperature (temperature above 90 °C)	BOOLEAN	RO	0x00 (0_{dec})
6010:04	Short Circuit to 24 V	Short Circuit to 24 V	BOOLEAN	RO	0x00 (0_{dec})
6010:05	Feedback PWM	Bit toggles with each DC event	BOOLEAN	RO	0x00 (0_{dec})
6010:06	Err. Start Time	“DC start time” (0x7010:13) is in the past	BOOLEAN	RO	0x00 (0_{dec})

5.4.4 0x70xx | Output data

Index 7000 PWM Outputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
7000:0	PWM Outputs Ch.1	Max. subindex	UINT8	RO	0x13 (19_{dec})
7000:01	Activate permanent	The PWM output signal is permanently active.	BOOLEAN	RO	0x00 (0_{dec})
7000:02	Activate DC start time	The DC start time "DC start time" (0x7000:13) is activated with the PWM signal (from values of the indices 0x7000:11 "PWM output" and 0x7000:12 "PWM period").	BOOLEAN	RO	0x00 (0_{dec})
7000:03	DC Immediately	0: Signal change without glitches. The PWM signal changes after the current period. 1: Glitches occurred during immediate signal change after the "DC start time" (0x7000:13) was reached.	BOOLEAN	RO	0x00 (0_{dec})
7000:04	ErrAck	Reset the error "Err. Start Time" (0x6000:06) and the error counter "DC Error Counter" (0x800E:04)	BOOLEAN	RO	0x00 (0_{dec})
7000:05	DCSyncOutCh2	The second channel toggles with each DC event.	BOOLEAN	RO	0x00 (0_{dec})
7000:11	PWM output	The duty cycle “PWM output” in % 0x7FFF = 100 % 0x3FFF = 50 % 0x0000 = 0 %	UINT16	RO	0x0000 (0_{dec})
7000:12	PWM period	"PWM Period" in µs	UINT16	RO	0x0000 (0_{dec})
7000:13	DC start time	The 64-bit "DC start time" in ns	ULINT64	RO	0x0000000000000000 (0_{dec})

Index 7010 PWM Outputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
7010:0	PWM Outputs Ch.2	Max. subindex	UINT8	RO	0x13 (19_{dec})
7010:01	Activate permanent	The PWM output signal is permanently active.	BOOLEAN	RO	0x00 (0_{dec})
7010:02	Activate DC start time	The DC start time "DC start time" (0x7010:13) is activated with the PWM signal (from values of the indices 0x7010:11 "PWM output" and 0x7010:12 "PWM period").	BOOLEAN	RO	0x00 (0_{dec})
7010:03	DC Immediately	0: Signal change without glitches. The PWM signal changes after the current period. 1: Glitches occurred during immediate signal change after the "DC start time" (0x7010:13) was reached.	BOOLEAN	RO	0x00 (0_{dec})
7010:04	ErrAck	Reset the error "Err. Start Time" (0x6010:06) and the error counter "DC Error Counter" (0x801E:04)	BOOLEAN	RO	0x00 (0_{dec})
7010:05	DCSyncOutCh2	The second channel toggles with each DC event.	BOOLEAN	RO	0x00 (0_{dec})
7010:11	PWM output	The duty cycle “PWM output” in % 0x7FFF = 100 % 0x3FFF = 50 % 0x0000 = 0 %	UINT16	RO	0x0000 (0_{dec})
7000:12	PWM period	"PWM Period" in µs	UINT16	RO	0x0000 (0_{dec})
7010:13	DC start time	The 64-bit "DC start time" in ns	ULINT64	RO	0x0000000000000000 (0_{dec})

5.4.5 0x80nE, 0xFxxx | Information and diagnostic data

Index 800E PWM Internal data Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
800E:0	PWM Internal data Ch.1	Max. subindex	UINT8	RO	0x04 (4_{dec})
800E:01	Timer resolution	Reload value of the PWM timer. The reload value matches the maximum resolution of the PWM unit.	UINT16	RO	0x0000 (0_{dec})
800E:02	Duty cycle	Current duty cycle of the PWM unit. 100% corresponds to the timer resolution (index 0x800E:01)	UINT16	RO	0x0000 (0_{dec})
800E:03	Device temperature	Terminal temperature	UINT16	RO	0x0000 (0_{dec})
800E:04	DC error counter	DC error counter	UINT16	RO	0x0000 (0_{dec})

Index 801E PWM Internal data Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
801E:0	PWM Internal data Ch.2	Max. subindex	UINT8	RO	0x04 (4_{dec})
801E:01	Timer resolution	Reload value of the PWM timer. The reload value matches the maximum resolution of the PWM unit.	UINT16	RO	0x0000 (0_{dec})
801E:02	Duty cycle	Current duty cycle of the PWM unit. 100% corresponds to the timer resolution (index 0x801E:01)	UINT16	RO	0x0000 (0_{dec})
801E:03	Device temperature	Terminal temperature	UINT16	RO	0x0000 (0_{dec})
801E:04	DC error counter	DC error counter	UINT16	RO	0x0000 (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	0x0002 (2_{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	0x02 (2_{dec})
F010:01	Subindex 001	MDP PWM 250	UINT16	RO	0x000000FA (250_{dec})
F010:02	Subindex 002	MDP PWM 250	UINT16	RO	0x000000FA (250_{dec})

Index FB40 Memory interface

Index (hex)	Name	Meaning	Data type	Flags	Default
FB40:0	Memory interface	Max. subindex	UINT8	RW	0x03 (3_{dec})
FB40:01	Adress	reserved	UINT32	RO	0x000000 (0_{dec})
FB40:02	Length	reserved	UINT16	RO	0x0000 (0_{dec})
FB40:03	Data	reserved	OCTET-STRING[8]	RO	{0}

5.4.6 0x10xx - 0x1C33 | Standard objects

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

5.4.6.1 0x10xx | Device

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	0x00FA1389 (16389001_{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	EL2502-0010

Index 1009 Hardware version

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

Index 100A Software version

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

Index 100B Bootloader version

Index (hex)	Name	Meaning	Data type	Flags	Default
100A:0	Bootloader version	Version of Bootloader	STRING	RO	-

Index 1018 Identity

Index (hex)	Name	Meaning	Data type	Flags	Default
1018:0	Identity	Information to identify 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	0x09C63052 (163983442_{dec})
1018:03	Revision	Revision number of the EtherCAT slave: The low word (bit 0-15) identifies the special terminal number. The high word (bit 16-31) refers to the device description.	UINT32	RO	-
1018:04	Serial number	Serial number of the EtherCAT slave: • Low Word ◦ 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	-

Index 10F0 Backup parameter

Index (hex)	Name	Meaning	Data type	Flags	Default
10F0:0	Backup parameter	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 10F8 Actual Time Stamp

Index (hex)	Name	Meaning	Data type	Flags	Default
10F8:0	Actual Time Stamp	Timestamp	UINT64	RO	0x0000000000000000 (0_{dec})

5.4.6.2 0x14xx | Excluded RxPDOs

Index 1400 PWM RxPDO-Par Outputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1400:0	RxPDO-Par Outputs Ch.1	PDO Parameter Outputs Ch.1	UINT8	RO	0x06 (6_{dec})
1400:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with this RxPDO	OCTET-STRING[2]	RO	01 16 02 16 03 16

Index 1401 PWM RxPDO-Par Per. Outputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1401:0	RxPDO-Par Per. Outputs Ch.1	PDO Parameter Per. Outputs Ch.1	UINT8	RO	0x06 (6_{dec})
1401:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with this RxPDO	OCTET-STRING[2]	RO	00 16 02 16 03 16

Index 1402 PWM RxPDO-Par DC Outputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1402:0	RxPDO-Par DC Outputs Ch.1	PDO Parameter DC Outputs Ch.1	UINT8	RO	0x06 (6_{dec})
1402:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with this RxPDO	OCTET-STRING[2]	RO	00 16 01 16 03 16

Index 1403 PWM RxPDO-Par Per, DC Outputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1403:0	RxPDO-Par Per, DC Outputs Ch.1	PDO Parameter Per, DC Outputs Ch.1	UINT8	RO	0x06 (6_{dec})
1403:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with this RxPDO	OCTET-STRING[2]	RO	00 16 01 16 02 16

Index 1404 PWM RxPDO-Par Outputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1404:0	RxPDO-Par Outputs Ch.2	PDO Parameter Outputs Ch.2	UINT8	RO	0x06 (6_{dec})
1404:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with this RxPDO	OCTET-STRING[2]	RO	05 16 06 16 07 16

Index 1405 PWM RxPDO-Par Per. Outputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1405:0	RxPDO-Par Per. Outputs Ch.2	PDO Parameter Per. Outputs Ch.2	UINT8	RO	0x06 (6_{dec})
1405:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with this RxPDO	OCTET-STRING[2]	RO	04 16 06 16 07 16

Index 1406 PWM RxPDO-Par DC Outputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1406:0	RxPDO-Par DC Outputs Ch.2	PDO Parameter DC Outputs Ch.2	UINT8	RO	0x06 (6_{dec})
1406:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with this RxPDO	OCTET-STRING[2]	RO	04 16 05 16 07 16

Index 1407 PWM RxPDO-Par Per, DC Outputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1407:0	RxPDO-Par Per, DC Outputs Ch.2	PDO Parameter Per, DC Outputs Ch.2	UINT8	RO	0x06 (6_{dec})
1407:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with this RxPDO	OCTET-STRING[2]	RO	04 16 05 16 06 16

5.4.6.3 0x16xx | PDO Mapping RxPDOs

Index 1600 PWM RxPDO-Map Outputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1600:0	RxPDO-Map Outputs Ch.1	PDO Mapping Outputs Ch.1	UINT8	RO	0x01 (1_{dec})
1600:01	SubIndex 001	1. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x11 (PWM output))	UINT32	RO	0x7000:11, 16

Index 1601 PWM RxPDO-Map Per. Outputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1601:0	RxPDO-Map Per. Outputs Ch.1	PDO Mapping Per. Outputs Ch.1	UINT8	RO	0x02 (2_{dec})
1601:01	SubIndex 001	1. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x11 (PWM output))	UINT32	RO	0x7000:11, 16
1601:02	SubIndex 002	2. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x12 (PWM period))	UINT32	RO	0x7000:12, 16

Index 1602 PWM RxPDO-Map DC Outputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1602:0	PWM RxPDO-Map DC Outputs Ch.1	PDO Mapping DC Outputs Ch.1	UINT8	RO	0x0A (10_{dec})
1602:01	SubIndex 001	1. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x01 (Activate permanent))	UINT32	RO	0x7000:01, 1
1602:02	SubIndex 002	2. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x02 (Activate DC start time))	UINT32	RO	0x7000:02, 1
1602:03	SubIndex 003	3. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x03 (DC Immediately))	UINT32	RO	0x7000:03, 1
1602:04	SubIndex 004	4. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x04 (ErrAck))	UINT32	RO	0x7000:04, 1
1602:05	SubIndex 005	5. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x05 (DCSyncOutCh2))	UINT32	RO	0x7000:05, 1
1602:06	SubIndex 006	6. PDO Mapping entry (3 bits align)	UINT32	RO	0x0000:00, 3
1602:07	SubIndex 007	7. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8
1602:08	SubIndex 008	8. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x11 (PWM output))	UINT32	RO	0x7000:11, 16
1602:09	SubIndex 009	9. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x12 (PWM period))	UINT32	RO	0x7000:12, 16
1602:0A	SubIndex 010	10. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x13 (DC start time))	UINT32	RO	0x7000:13, 64

Index 1603 PWM RxPDO-Map Per, DC Outputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1603:0	PWM RxPDO-Map Per, DC Outputs Ch.1	PDO Mapping Per, DC Outputs Ch.1	UINT8	RO	0x0A (10_{dec})
1603:01	SubIndex 001	1. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x01 (Activate permanent))	UINT32	RO	0x7000:01, 1
1603:02	SubIndex 002	2. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x02 (Activate DC start time))	UINT32	RO	0x7000:02, 1
1603:03	SubIndex 003	3. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x03 (DC Immediately))	UINT32	RO	0x7000:03, 1
1603:04	SubIndex 004	4. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x04 (ErrAck))	UINT32	RO	0x7000:04, 1
1603:05	SubIndex 005	5. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x05 (DCSyncOutCh2))	UINT32	RO	0x7000:05, 1
1603:06	SubIndex 006	6. PDO Mapping entry (3 bits align)	UINT32	RO	0x0000:00, 3
1603:07	SubIndex 007	7. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8
1603:08	SubIndex 008	8. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x11 (PWM output))	UINT32	RO	0x7000:11, 16
1603:09	SubIndex 009	9. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x12 (PWM period))	UINT32	RO	0x7000:12, 16
1603:0A	SubIndex 010	10. PDO Mapping entry (object 0x7000 (PWM Outputs Ch.1), entry 0x13 (DC start time))	UINT32	RO	0x7000:13, 64

Index 1604 PWM RxPDO-Map Outputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1604:0	RxPDO-Map Outputs Ch.2	PDO Mapping Outputs Ch.2	UINT8	RO	0x01 (1_{dec})
1604:01	SubIndex 001	1. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x11 (PWM output))	UINT32	RO	0x7010:11, 16

Index 1605 PWM RxPDO-Map Per. Outputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1605:0	RxPDO-Map Per. Outputs Ch.2	PDO Mapping Per. Outputs Ch.2	UINT8	RO	0x02 (2_{dec})
1605:01	SubIndex 001	1. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x11 (PWM output))	UINT32	RO	0x7010:11, 16
1605:02	SubIndex 002	2. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x12 (PWM period))	UINT32	RO	0x7010:12, 16

Index 1606 PWM RxPDO-Map DC Outputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1606:0	PWM RxPDO-Map DC Outputs Ch.2	PDO Mapping DC Outputs Ch.2	UINT8	RO	0x0A (10_{dec})
1606:01	SubIndex 001	1. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x01 (Activate permanent))	UINT32	RO	0x7010:01, 1
1606:02	SubIndex 002	2. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x02 (Activate DC start time))	UINT32	RO	0x7010:02, 1
1606:03	SubIndex 003	3. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x03 (DC Immediately))	UINT32	RO	0x7010:03, 1
1606:04	SubIndex 004	4. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x04 (ErrAck))	UINT32	RO	0x7010:04, 1
1606:05	SubIndex 005	5. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x05 (DCSyncOutCh2))	UINT32	RO	0x7010:05, 1
1606:06	SubIndex 006	6. PDO Mapping entry (3 bits align)	UINT32	RO	0x0000:00, 3
1606:07	SubIndex 007	7. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8
1606:08	SubIndex 008	8. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x11 (PWM output))	UINT32	RO	0x7010:11, 16
1606:09	SubIndex 009	9. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x12 (PWM period))	UINT32	RO	0x7010:12, 16
1606:0A	SubIndex 010	10. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x13 (DC start time))	UINT32	RO	0x7010:13, 64

Index 1607 PWM RxPDO-Map Per, DC Outputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1607:0	PWM RxPDO-Map Per, DC Outputs Ch.1	PDO Mapping Per, DC Outputs Ch.2	UINT8	RO	0x09 (9_{dec})
1607:01	SubIndex 001	1. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x01 (Activate permanent))	UINT32	RO	0x7010:01, 1
1607:02	SubIndex 002	2. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x02 (Activate start time))	UINT32	RO	0x7010:02, 1
1607:03	SubIndex 003	3. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x03 (DC Immediately))	UINT32	RO	0x7010:03, 1
1607:04	SubIndex 004	4. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x04 (ErrAck))	UINT32	RO	0x7010:04, 1
1607:05	SubIndex 005	5. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x05 (DCSyncOutCh2))	UINT32	RO	0x7010:05, 1
1607:06	SubIndex 006	6. PDO Mapping entry (5 bits align)	UINT32	RO	0x0000:00, 5
1607:07	SubIndex 007	7. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8
1607:08	SubIndex 008	8. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x11 (PWM output))	UINT32	RO	0x7010:11, 64
1607:09	SubIndex 009	9. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x12 (PWM period))	UINT32	RO	0x7010:12, 64
1607:0A	SubIndex 010	10. PDO Mapping entry (object 0x7010 (PWM Outputs Ch.2), entry 0x13 (DC Start time))	UINT32	RO	0x7010:13, 64

5.4.6.4 0x1Axx | PDO Mapping TxPDOs

Index 1A00 PWM TxPDO-Map Inputs Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1A00:0	TxPDO-Map Inputs Ch.1	PDO Mapping Inputs Ch.1	UINT8	RO	0x07 (7_{dec})
1A00:01	SubIndex 001	1. PDO Mapping entry (object 0x6000 (PWM Inputs Ch.1), entry 0x01 (Overtemperature))	UINT32	RO	0x6000:01, 1
1A00:02	SubIndex 002	2. PDO Mapping entry (2 bits align)	UINT32	RO	0x0000:00, 2
1A00:03	SubIndex 003	3. PDO Mapping entry (object 0x6000 (PWM Inputs Ch.1), entry 0x04 (Short Circuit to 24 V))	UINT32	RO	0x6000:04, 1
1A00:04	SubIndex 004	4. PDO Mapping entry (object 0x6000 (PWM Inputs Ch.1), entry 0x05 (Feedback PWM))	UINT32	RO	0x6000:05, 1
1A00:05	SubIndex 005	5. PDO Mapping entry (object 0x6000 (PWM Inputs Ch.1), entry 0x06 (Err. Start Time))	UINT32	RO	0x6000:06, 1
1A00:06	SubIndex 006	6. PDO Mapping entry (2 bits align)	UINT32	RO	0x0000:00, 2
1A00:07	SubIndex 007	7. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8

Index 1A01 PWM TxPDO-Map Sys Time Ch.1

Index (hex)	Name	Meaning	Data type	Flags	Default
1A01:0	TxPDO-Map Inputs Ch.2	PDO Mapping Inputs Ch.1	UINT8	RO	0x01 (1_{dec})
1A01:01	SubIndex 001	1. PDO Mapping entry (object 0x6001 (Sys Time), entry 0x11 (SysTime))	UINT32	RO	0x6001:11, 64

Index 1A02 PWM TxPDO-Map Inputs Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1A02:0	TxPDO-Map Inputs Ch.2	PDO Mapping Inputs Ch.2	UINT8	RO	0x07 (7_{dec})
1A02:01	SubIndex 001	1. PDO Mapping entry (object 0x6010 (PWM Inputs Ch.2), entry 0x01 (Overtemperature))	UINT32	RO	0x6010:01, 1
1A02:02	SubIndex 002	2. PDO Mapping entry (2 bits align)	UINT32	RO	0x0000:00, 2
1A02:03	SubIndex 003	3. PDO Mapping entry (object 0x6010 (PWM Inputs Ch.2), entry 0x04 (Short Circuit to 24 V))	UINT32	RO	0x6010:04, 1
1A02:04	SubIndex 004	4. PDO Mapping entry (object 0x6010 (PWM Inputs Ch.2), entry 0x05 (Feedback PWM))	UINT32	RO	0x6010:05, 1
1A02:05	SubIndex 005	5. PDO Mapping entry (object 0x6010 (PWM Inputs Ch.2), entry 0x06 (Err. Start Time))	UINT32	RO	0x6010:06, 1
1A02:06	SubIndex 006	6. PDO Mapping entry (2 bits align)	UINT32	RO	0x0000:00, 2
1A02:07	SubIndex 007	7. PDO Mapping entry (4 bits align)	UINT32	RO	0x0000:00, 4
1A02:08	SubIndex 008	8. PDO Mapping entry (object 0x6010 (PWM Inputs Ch.2), entry 0x0D (Diag))	UINT32	RO	0x6010:0D, 1
1A02:09	SubIndex 009	9. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8

Index 1A03 PWM TxPDO-Map Sys Time Ch.2

Index (hex)	Name	Meaning	Data type	Flags	Default
1A03:0	TxPDO-Map Inputs Ch.2	PDO Mapping Inputs Ch.1	UINT8	RO	0x01 (1_{dec})
1A03:01	SubIndex 001	1. PDO Mapping entry (object 0x6001 (Sys Time), entry 0x11 (SysTime))	UINT32	RO	0x6001:11, 64

5.4.6.5 0x1Cxx | Sync Manager

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	RO	0x02 (2_{dec})
1C12:01	Subindex 001	1. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RO	0x1603 (5635_{dec})
1C12:02	Subindex 002	2. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RO	0x1607 (5639_{dec})

Index 1C13 TxPDO assign

Index (hex)	Name	Meaning	Data type	Flags	Default
1C13:0	TxPDO assign	PDO Assign Inputs	UINT8	RO	0x04 (4_{dec})
1C13:01	Subindex 001	1. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RO	0x1A00 (6656_{dec})
1C13:02	Subindex 002	2. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RO	0x1A01 (6657_{dec})
1C13:03	Subindex 003	3. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RO	0x1A02 (6658_{dec})
1C13:04	Subindex 004	4. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RO	0x1A03 (6659_{dec})

Index 1C32 SM output parameter

Index (hex)	Name	Meaning	Data type	Flags	Default
1C32:0	SM output parameter	Synchronization parameters for the outputs	UINT8	RO	0x20 (32_{dec})
1C32:01	Sync mode	Current synchronization mode: • 0: Free Run • 1: Synchron with SM 2 Event	UINT16	RW	0x0000 (0_{dec})
1C32:02	Cycle time	Cycle time (in ns): • Free Run: cycle time of the local timer • Synchron with SM 2 Event: cycle time of the master	UINT32	RW	0x00000000 (0_{dec})
1C32:03	Shift time	Time between SYNC0 Event and output of the outputs (in ns, DC mode only)	UINT32	RO	0x00000000 (0_{dec})
1C32:04	Sync modes supported	Sync modes supported: • Bit 0 = 1: Free Run is supported • Bit 1 = 1: Synchron with SM 2 Event is supported • Bit 2-3 = 01: DC-Mode is supported • Bit 4-5 = 10: Output Shift with SYNC1 Event (only DC mode) • Bit 14 = 1: dynamic times (measurement through writing of 0x1C32:08)	UINT16	RO	0x4001 (16385_{dec})
1C32:05	Minimum cycle time	Minimum cycle time (in ns)	UINT32	RO	0x00000000 (0_{dec})
1C32:06	Calc and copy time	Minimum time between SYNC0 and SYNC1 Event (in ns, DC Mode only)	UINT32	RO	0x00000000 (0_{dec})
1C32:07	Minimum delay time	•	UINT32	RO	0x00000000 (0_{dec})
1C32:08	Command	• 0: Measurement of the local cycle time is stopped • 1: Measurement of the local cycle time is started The entries 0x1C32:03, 0x1C32:05, 0x1C32:06, 0x1C32:09 are updated with the maximum measured values. For a subsequent measurement the measured values are reset	UINT16	RW	0x00000000 (0_{dec})
1C32:09	Delay time	Time between SYNC1 Event and output of the outputs (in ns, DC mode only)	UINT32	RO	0x0000 (0_{dec})
1C32:0B	SM event missed counter	Number of missed SM events in OPERATIONAL (DC Mode only)	UINT16	RO	0x00000000 (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	0x0000 (0_{dec})

Index 1C33 SM input parameter

Index (hex)	Name	Meaning	Data type	Flags	Default
1C33:0	SM input parameter	Synchronization parameters for the inputs	UINT8	RO	0x20 (32_{dec})
1C33:01	Sync mode	Current synchronization mode: • 0: Free Run • 1: Synchron with SM 3 Event (no outputs available) • 2: DC - Synchron with SYNC0-Event • 3: DC - Synchron with SYNC1-Event • 34: Synchron with SM 2 Event (outputs available)	UINT16	RW	0x0000 (0_{dec})
1C33:02	Cycle time	as 0x1C32:02 [▶ 138]	UINT32	RW	0x00000000 (0_{dec})
1C33:03	Shift time	Time between SYNC0 event and reading of the inputs (in ns, DC Mode only)	UINT32	RO	0x00000000 (0_{dec})
1C33:04	Sync modes supported	Sync modes supported: • Bit 0 = 1: Free Run is supported • Bit 1 = 1: Synchron with SM 2 Event is supported • Bit 2-3 = 01: DC-Mode is supported • Bit 14 = 1: dynamic times (measurement by writing 0x1C32:08 [▶ 138]) (for revision no.: 17 – 25)	UINT16	RO	0x4001 (16385_{dec})
1C33:05	Minimum cycle time	as 0x1C32:05 [▶ 138]	UINT32	RO	0x00000000 (0_{dec})
1C33:06	Calc and copy time	Time between reading of the inputs and the inputs being available for the master (in ns, DC Mode only)	UINT32	RO	0x00000000 (0_{dec})
1C33:07	Minimum delay time		UINT32	RO	0x00000000 (0_{dec})
1C33:08	Command	as 0x1C32:08 [▶ 138]	UINT16	RW	0x0000 (0_{dec})
1C33:09	Maximum Delay time	Time between SYNC1 Event and reading of the inputs (in ns, DC Mode only)	UINT32	RO	0x00000000 (0_{dec})
1C33:0B	SM event missed counter	as 0x1C32: [▶ 138]0B	UINT16	RO	0x0000 (0_{dec})
1C33:0C	Cycle exceeded counter	as 0x1C32: [▶ 138]0C	UINT16	RO	0x0000 (0_{dec})
1C33:0D	Shift too short counter	as 0x1C32: [▶ 138]0D	UINT16	RO	0x0000 (0_{dec})
1C33:20	Sync error	as 0x1C32: [▶ 138]20	BOOLEAN	RO	0x00 (0_{dec})

6 Appendix

6.1 EtherCAT AL Status Codes

For detailed information please refer to the EtherCAT system description.

6.2 Firmware compatibility

Beckhoff EtherCAT devices are delivered with the latest available firmware version. Compatibility of firmware and hardware is mandatory; not every combination ensures compatibility. The overview below shows the hardware versions on which a firmware can be operated.

Note

- It is recommended to use the newest possible firmware for the respective hardware
- Beckhoff is not under any obligation to provide customers with free firmware updates for delivered products.

NOTICE

Risk of damage to the device!

Pay attention to the instructions for firmware updates on the separate page [▶ 142].
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!

EL2502-0010			
Hardware (HW)	**Firmware (FW)**	**Revision**	**Release date**
00 - 02*	01	EL2502-0010-0016	20YY/MM
	03	EL2502-0010-0020	2019/04
	04*		2020/10

*) This is the current compatible firmware/hardware version at the time of the preparing this documentation.
Check on the Beckhoff web page whether more up-to-date documentation is available.

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

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

NOTICE

Only use TwinCAT 3 software!

A firmware update of Beckhoff IO devices must only be performed with a TwinCAT 3 installation. It is recommended to build as up-to-date as possible, available for free download on the Beckhoff website.

To update the firmware, TwinCAT can be operated in the so-called FreeRun mode, a paid license is not required.

The device to be updated can usually remain in the installation location, but TwinCAT has to be operated in the FreeRun. Please make sure that EtherCAT communication is trouble-free (no LostFrames etc.).

Other EtherCAT master software, such as the EtherCAT Configurator, should not be used, as they may not support the complexities of updating firmware, EEPROM and other device components.

Storage locations

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

- Each EtherCAT slave has a device description, consisting of identity (name, product code), timing specifications, communication settings, etc.
  This device description (ESI; EtherCAT Slave Information) can be downloaded from the Beckhoff website in the download area as a zip file and used in EtherCAT masters for offline configuration, e.g. in TwinCAT.
  Above all, each EtherCAT slave carries its device description (ESI) electronically readable in a local memory chip, the so-called **ESI EEPROM**. When the slave is switched on, this description is loaded locally in the slave and informs it of its communication configuration; on the other hand, the EtherCAT master can identify the slave in this way and, among other things, set up the EtherCAT communication accordingly.

NOTICE

Application-specific writing of the ESI-EEPROM

The ESI is developed by the device manufacturer according to ETG standard and released for the corresponding product.
- Meaning for the ESI file: Modification on the application side (i.e. by the user) is not permitted.
- Meaning for the ESI EEPROM: Even if a writeability is technically given, the ESI parts in the EEPROM and possibly still existing free memory areas must not be changed beyond the normal update process. Especially for cyclic memory processes (operating hours counter etc.), dedicated memory products such as EL6080 or IPC's own NOVRAM must be used.

- Depending on functionality and performance EtherCAT slaves have one or several local controllers for processing I/O data. The corresponding program is the so-called **firmware** in *.efw format.
- In some EtherCAT slaves the EtherCAT communication may also be integrated in these controllers. In this case the controller is usually a so-called **FPGA** chip with *.rbf firmware.

Customers can access the data via the EtherCAT fieldbus and its communication mechanisms. Acyclic mailbox communication or register access to the ESC is used for updating or reading of these data.

The TwinCAT System Manager offers mechanisms for programming all three parts with new data, if the slave is set up for this purpose. Generally the slave does not check whether the new data are suitable, i.e. it may no longer be able to operate if the data are unsuitable.

Simplified update by bundle firmware

The update using so-called **bundle firmware** is more convenient: in this case the controller firmware and the ESI description are combined in a *.efw file; during the update both the firmware and the ESI are changed in the terminal. For this to happen it is necessary

- for the firmware to be in a packed format: recognizable by the file name, which also contains the revision number, e.g. ELxxxx-xxxx_REV0016_SW01.efw

- for password=1 to be entered in the download dialog. If password=0 (default setting) only the firmware update is carried out, without an ESI update.
- for the device to support this function. The function usually cannot be retrofitted; it is a component of many new developments from year of manufacture 2016.

Following the update, its success should be verified

- ESI/Revision: e.g. by means of an online scan in TwinCAT ConfigMode/FreeRun – this is a convenient way to determine the revision
- Firmware: e.g. by looking in the online CoE of the device

NOTICE

Risk of damage to the device!

✓ Note the following when downloading new device files

a) Firmware downloads to an EtherCAT device must not be interrupted

b) Flawless EtherCAT communication must be ensured. CRC errors or LostFrames must be avoided.

c) The power supply must adequately dimensioned. The signal level must meet the specification.

⇨ In the event of malfunctions during the update process the EtherCAT device may become unusable and require re-commissioning by the manufacturer.

6.3.1 Device description ESI file/XML

NOTICE

Attention regarding update of the ESI description/EEPROM

Some slaves have stored calibration and configuration data from the production in the EEPROM. These are irretrievably overwritten during an update.

The ESI device description is stored locally on the slave and loaded on start-up. Each device description has a unique identifier consisting of slave name (9 characters/digits) and a revision number (4 digits). Each slave configured in the System Manager shows its identifier in the EtherCAT tab:

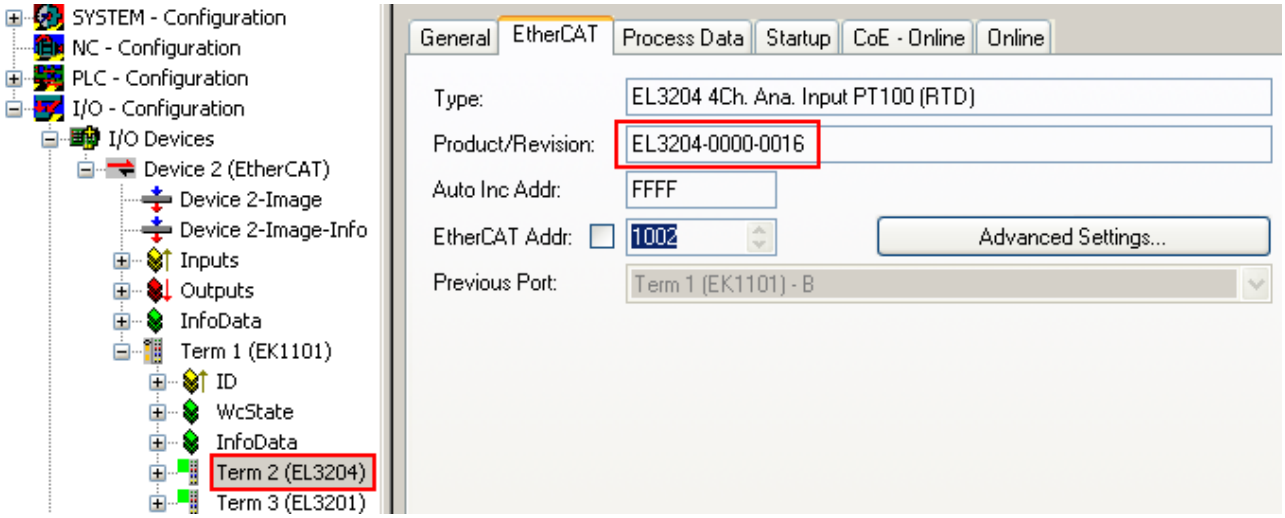


Fig. 139: Device identifier consisting of name EL3204-0000 and revision -0016

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

For further information on this, please refer to the EtherCAT system documentation.

Update of XML/ESI description

The device revision is closely linked to the firmware and hardware used. Incompatible combinations lead to malfunctions or even final shutdown of the device. Corresponding updates should only be carried out in consultation with Beckhoff support.

Display of ESI slave identifier

The simplest way to ascertain compliance of configured and actual device description is to scan the EtherCAT boxes in TwinCAT mode Config/FreeRun:

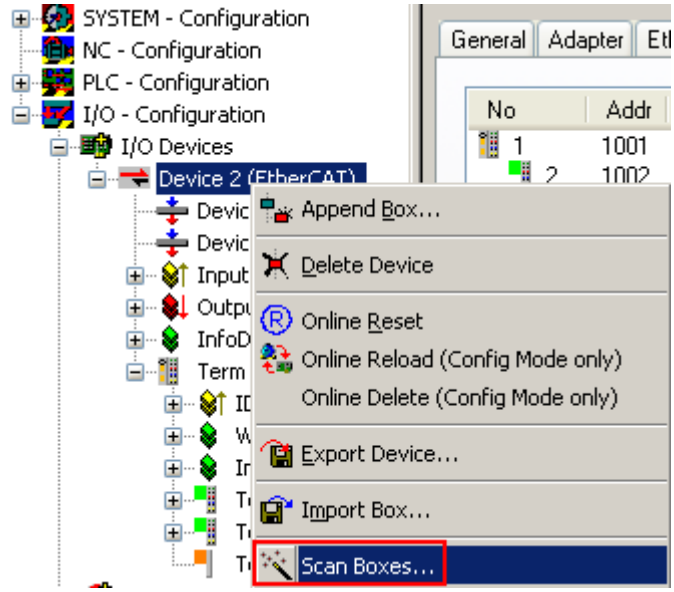


Fig. 140: Scan the subordinate field by right-clicking on the EtherCAT device

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

Fig. 141: Configuration is identical

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

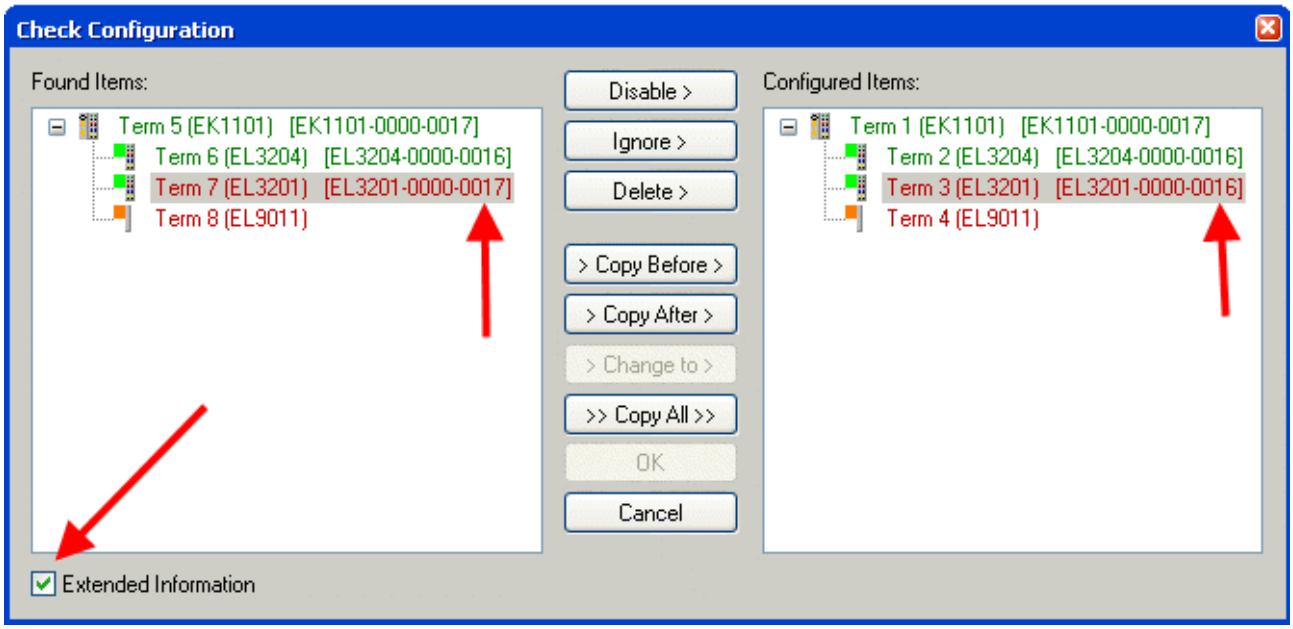


Fig. 142: Change dialog

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*

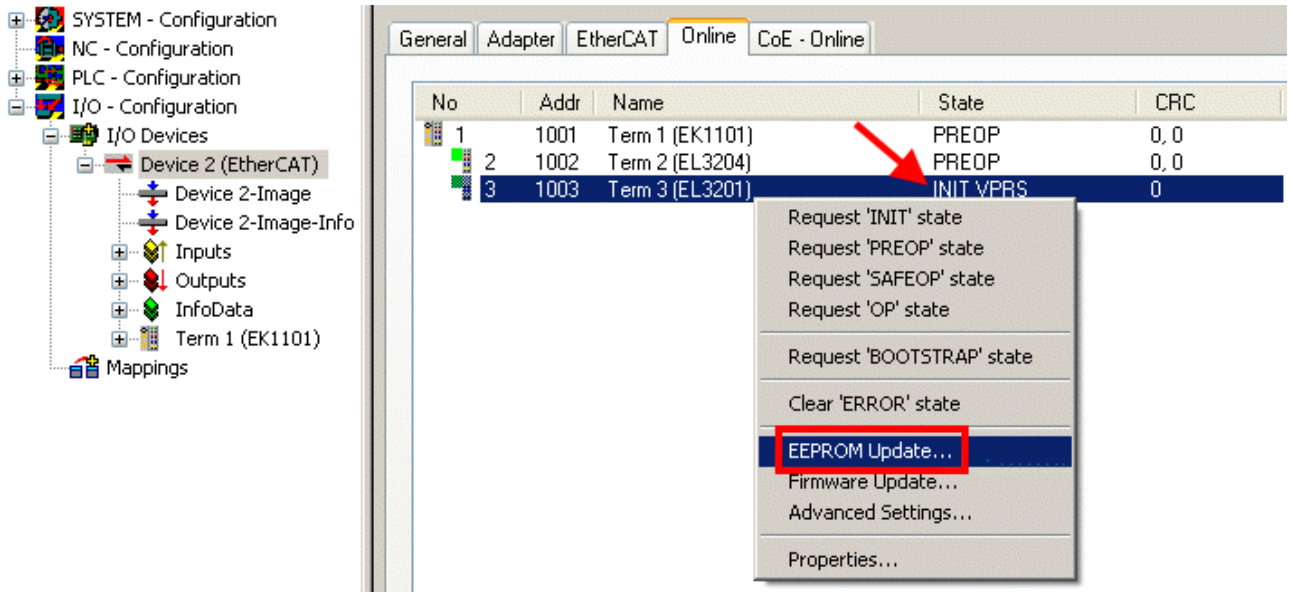


Fig. 143: EEPROM Update

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.

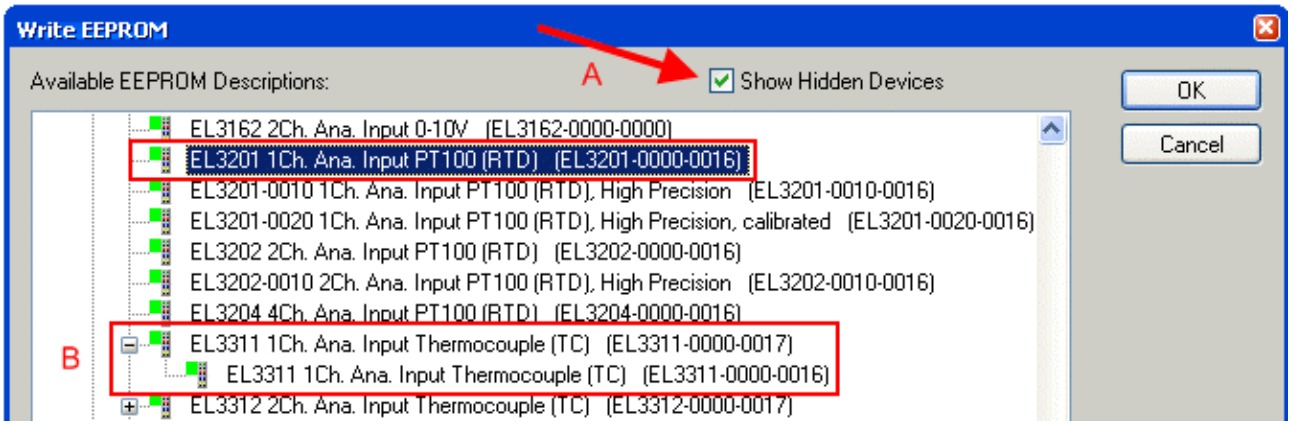


Fig. 144: Selecting the new ESI

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

The change only takes effect after a restart.

Most EtherCAT devices read a modified ESI description immediately or after startup from the INIT. Some communication settings such as distributed clocks are only read during power-on. The EtherCAT slave therefore has to be switched off briefly in order for the change to take effect.

6.3.2 Firmware explanation

Determining the firmware version

Determining the version via the TwinCAT System Manager

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

CoE Online and Offline CoE

Two CoE directories are available:
• **online**: This is offered in the EtherCAT slave by the controller, if the EtherCAT slave supports this. This CoE directory can only be displayed if a slave is connected and operational.
• **offline**: The EtherCAT Slave Information ESI/XML may contain the default content of the CoE. This CoE directory can only be displayed if it is included in the ESI (e.g. “Beckhoff EL5xxx.xml”).

The Advanced button must be used for switching between the two views.

In Fig. *Display of EL3204 firmware version* the firmware version of the selected EL3204 is shown as 03 in CoE entry 0x100A.

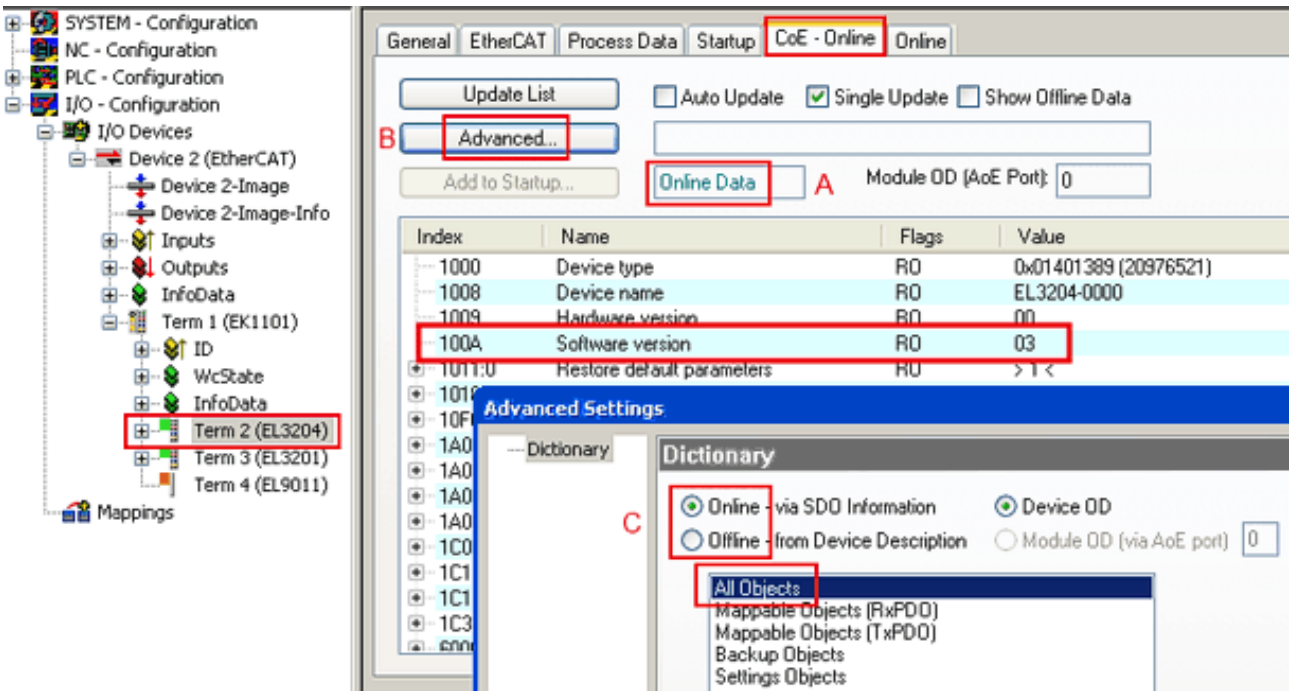


Fig. 145: Display of EL3204 firmware version

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

6.3.3 Updating controller firmware *.efw

CoE directory

The Online CoE directory is managed by the controller and stored in a dedicated EEPROM, which is generally not changed during a firmware update.

Switch to the *Online* tab to update the controller firmware of a slave, see Fig. *Firmware Update.*

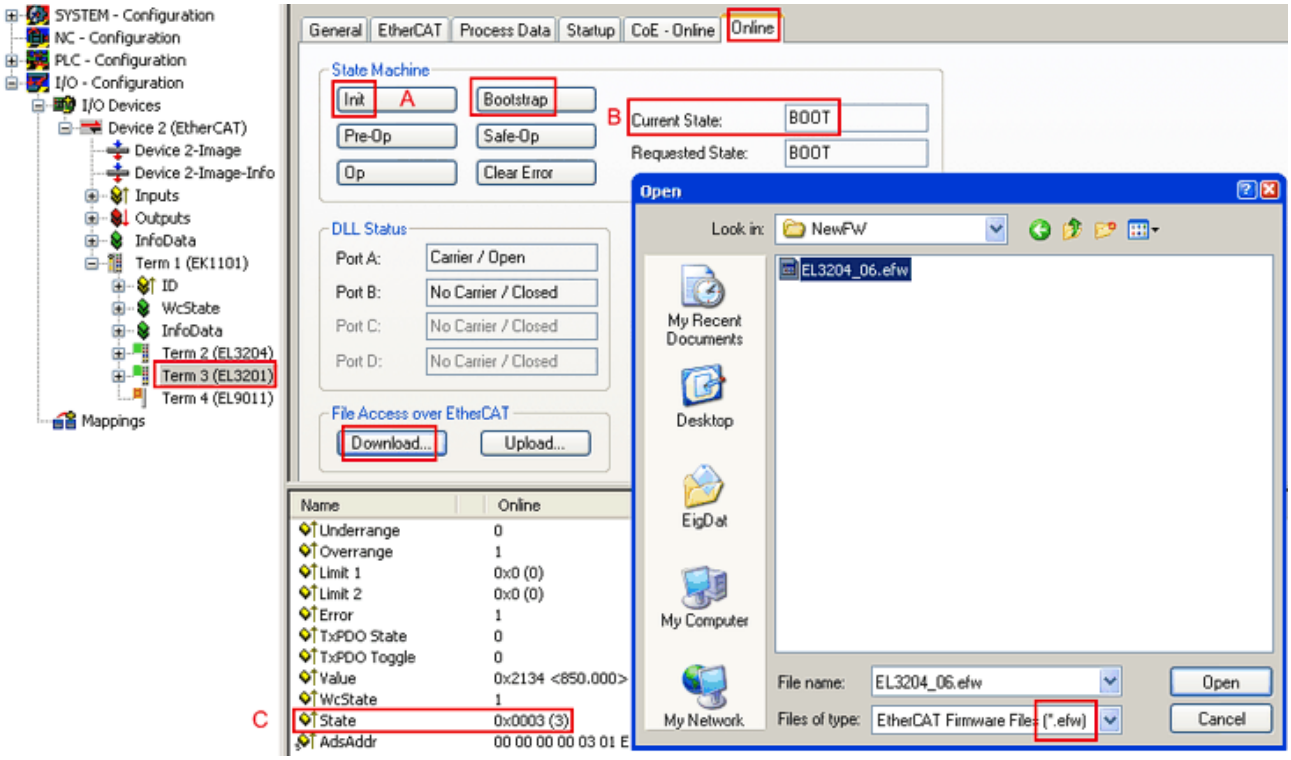
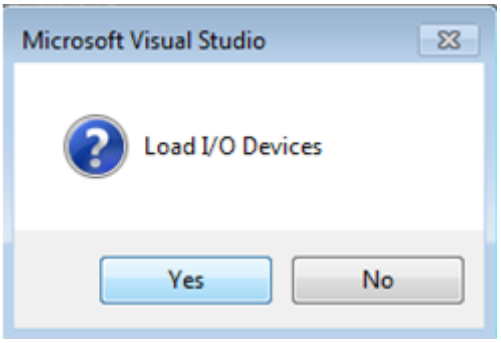
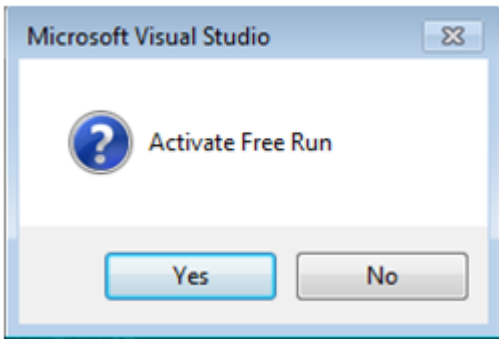


Fig. 146: Firmware Update

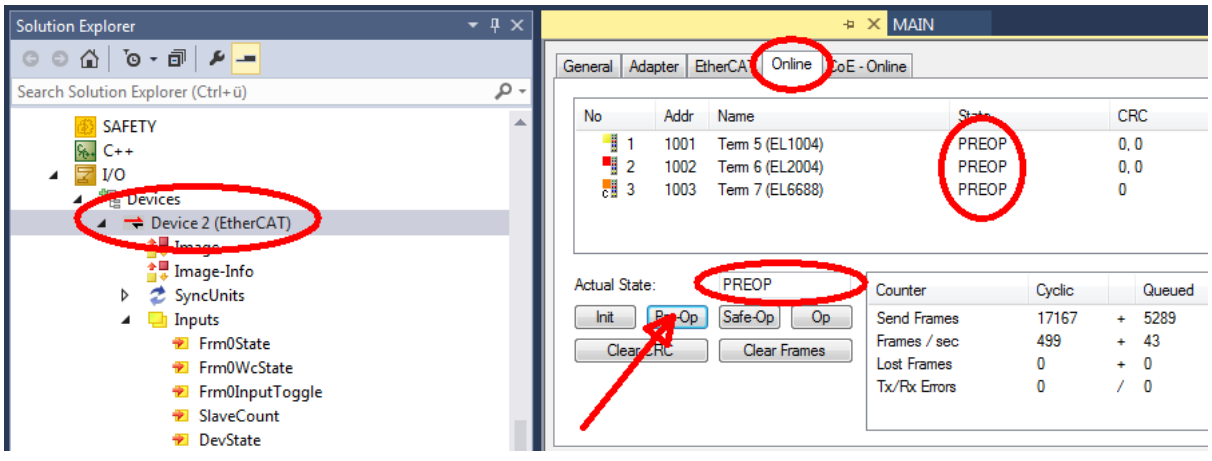
Proceed as follows, unless instructed otherwise by Beckhoff support. Valid for TwinCAT 2 and 3 as EtherCAT master.

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



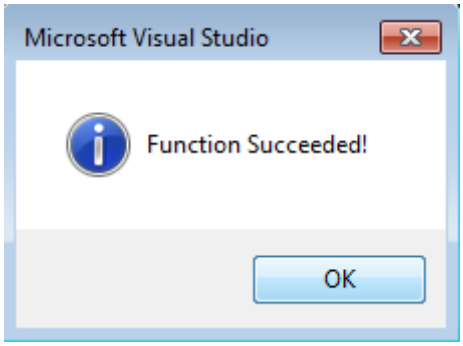


- Switch EtherCAT Master to PreOP



- Switch slave to INIT (A)
- Switch slave to BOOTSTRAP

- Check the current status (B, C)
- Download the new *efw file (wait until it ends). A password will not be necessary usually.



- After the download switch to INIT, then PreOP
- Switch off the slave briefly (don't pull under voltage!)
- Check within CoE 0x100A, if the FW status was correctly overtaken.

6.3.4 FPGA firmware *.rbf

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

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

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

Determining the version via the TwinCAT System Manager

The TwinCAT System Manager indicates the FPGA firmware version. Click on the Ethernet card of your EtherCAT strand (Device 2 in the example) and select the *Online* tab.

The *Reg:0002* column indicates the firmware version of the individual EtherCAT devices in hexadecimal and decimal representation.

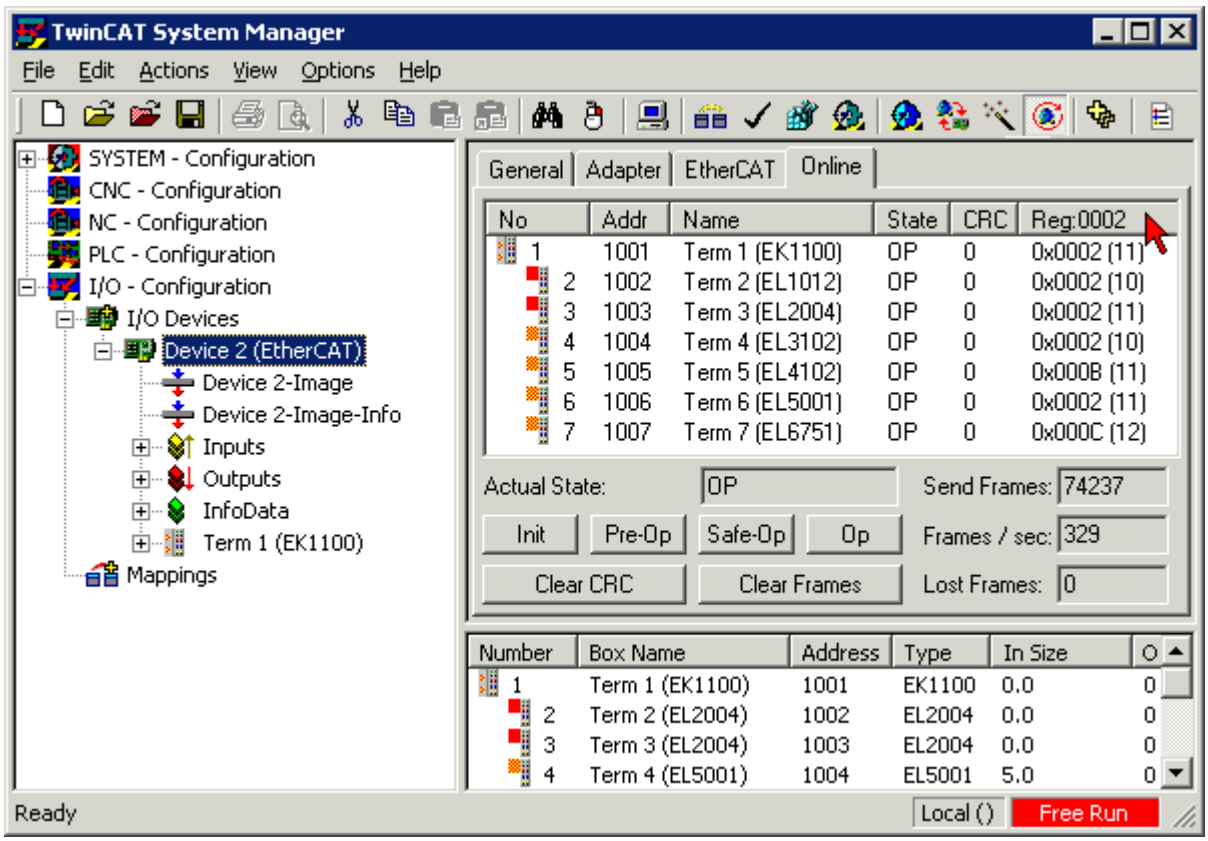


Fig. 147: FPGA firmware version definition

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

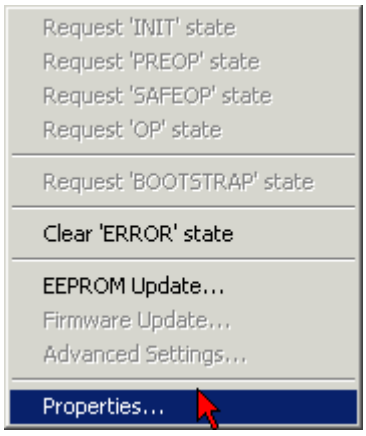


Fig. 148: Context menu *Properties*

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

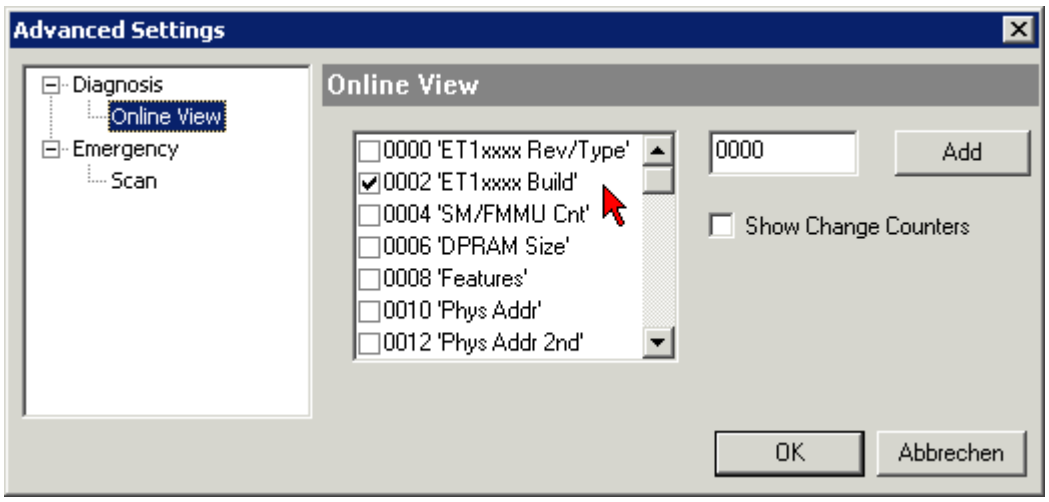


Fig. 149: Dialog *Advanced Settings*

Update

For updating the FPGA firmware

- of an EtherCAT coupler the coupler must have FPGA firmware version 11 or higher;
- of an E-Bus Terminal the terminal must have FPGA firmware version 10 or higher.

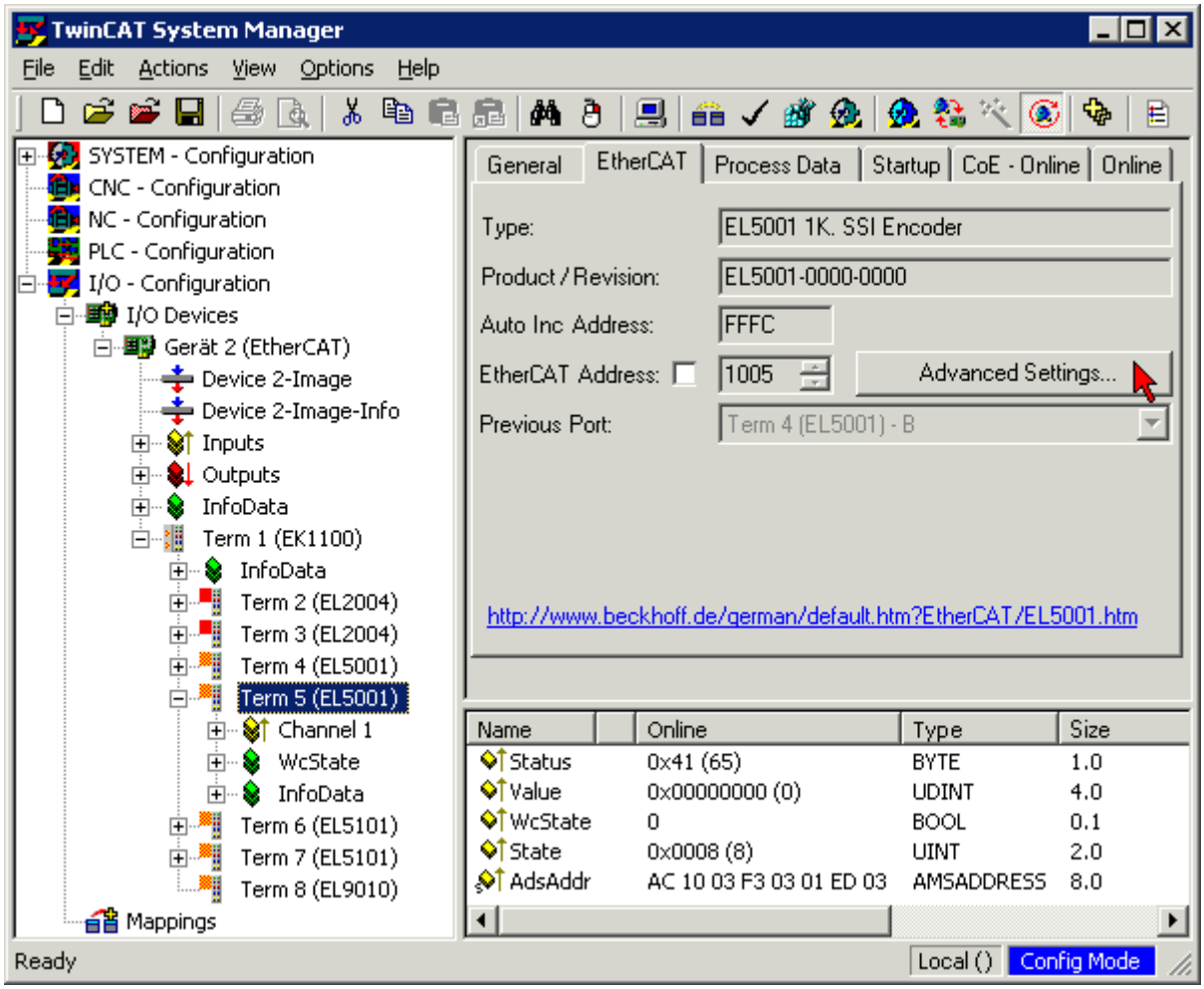
Older firmware versions can only be updated by the manufacturer!

Updating an EtherCAT device

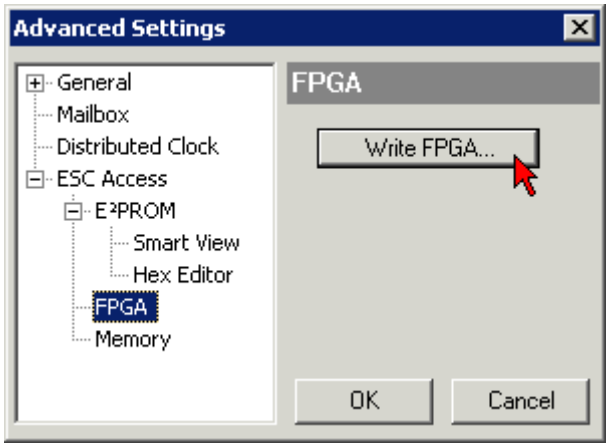
The following sequence order have to be met if no other specifications are given (e.g. by the Beckhoff support):

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

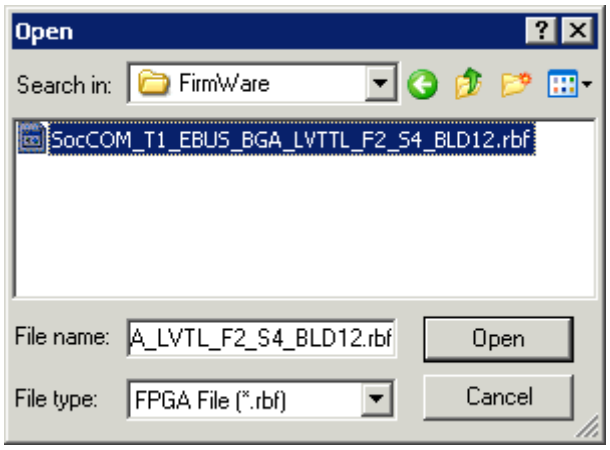
- In the TwinCAT System Manager select the terminal for which the FPGA firmware is to be updated (in the example: Terminal 5: EL5001) and
  click the *Advanced Settings* button in the *EtherCAT* tab:



- The *Advanced Settings* dialog appears. Under *ESC Access/E²PROM*/FPGA click on *Write FPGA* button:



- Select the file (*.rbf) with the new FPGA firmware, and transfer it to the EtherCAT device:



- Wait until download ends
- Switch slave current less for a short time (don't pull under voltage!). In order to activate the new FPGA firmware a restart (switching the power supply off and on again) of the EtherCAT device is required.
- Check the new FPGA status

NOTICE

Risk of damage to the device!

A download of firmware to an EtherCAT device must not be interrupted in any case! If you interrupt this process by switching off power supply or disconnecting the Ethernet link, the EtherCAT device can only be recommissioned by the manufacturer!

6.3.5 Simultaneous updating of several EtherCAT devices

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

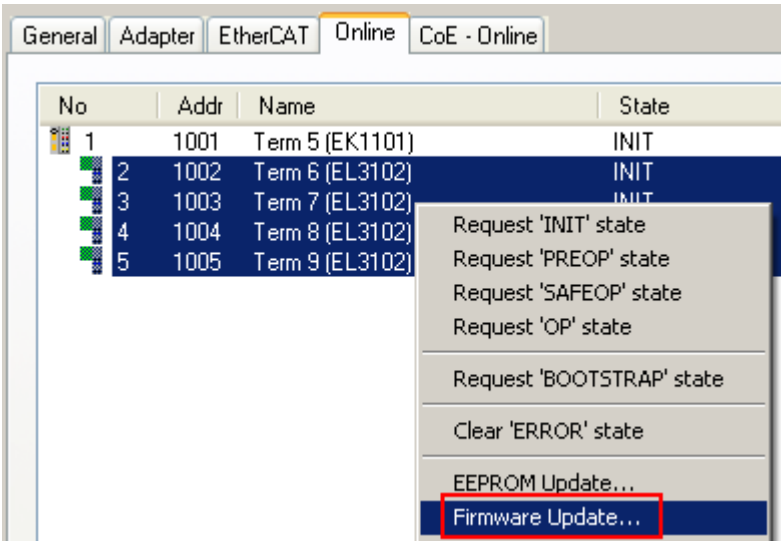


Fig. 150: Multiple selection and firmware update

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

6.4 Restoring the delivery state

To restore the delivery state (factory settings) of CoE objects (object directory) for EtherCAT devices ("slaves"), the CoE object *Restore default parameters*, SubIndex 001 can be used via EtherCAT master (e.g. TwinCAT) (see Fig. *Selecting the Restore default parameters PDO*).

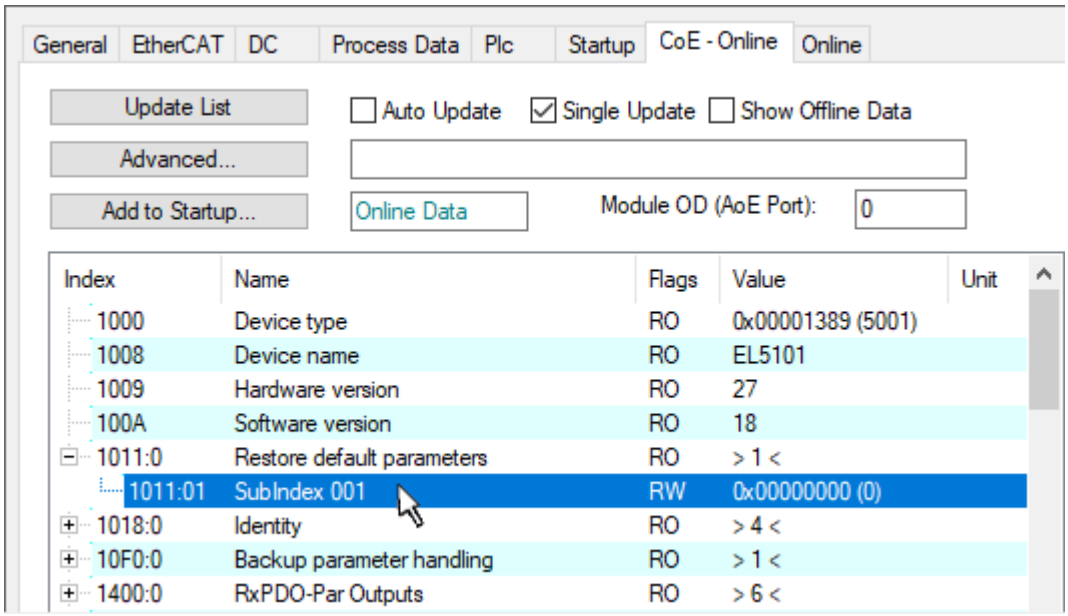


Fig. 151: Selecting the *Restore default parameters* PDO

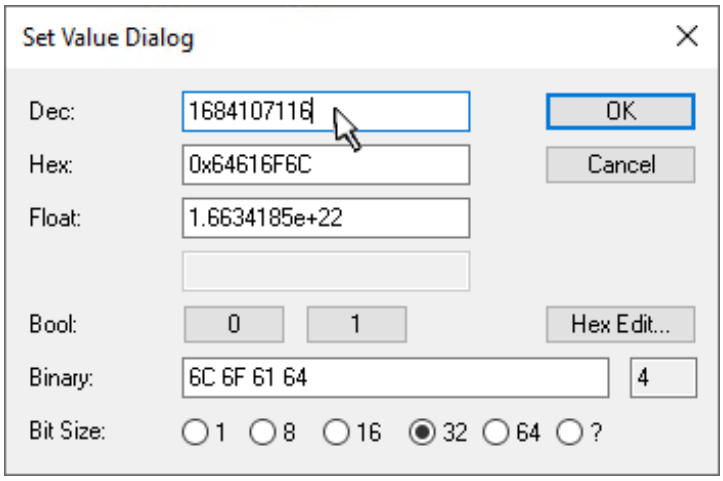


Fig. 152: Entering a restore value in the Set Value dialog

Double-click on *SubIndex 001* to enter the Set Value dialog. Enter the reset value **1684107116** in field *Dec* or the value **0x64616F6C** in field *Hex* (ASCII: "load") and confirm with *OK* (Fig. *Entering a restore value in the Set Value dialog*).

- All changeable entries in the slave are reset to the default values.
  Exception: objects write-protected via password
- Depending on the size of the object directory, this process can take from a few ms to > 1 second.
- The values can only be successfully restored if the reset is directly applied to the online CoE, i.e. to the slave. No values can be changed in the offline CoE.
- TwinCAT must be in the RUN or CONFIG/Freerun state for this; that means EtherCAT data exchange takes place. Ensure error-free EtherCAT transmission.
- No separate confirmation takes place due to the reset. A changeable object can be manipulated beforehand for the purposes of checking.
- This reset procedure can also be adopted as the first entry in the startup list of the slave, e.g. in the state transition PREOP->SAFEOP or, as in Fig. *CoE reset as a startup entry*, in SAFEOP->OP.

All backup objects are reset to the delivery state.

Alternative restore value

In some older terminals (FW creation approx. before 2007) the backup objects can be switched with an alternative restore value: Decimal value: 1819238756, Hexadecimal value: 0x6C6F6164.

An incorrect entry for the restore value has no effect.

6.5 Support and Service

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

Beckhoff's branch offices and representatives

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

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

You will also find further documentation for Beckhoff components there.

Support

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

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

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

Service

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

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

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

Headquarters Germany

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Germany

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

Trademark statements

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

Third-party trademark statements

DeviceNet and EtherNet/IP are trademarks of ODVA, Inc.

DSP System Toolbox, Embedded Coder, MATLAB, MATLAB Coder, MATLAB Compiler, MathWorks, Predictive Maintenance Toolbox, Simscape, Simscape™ Multibody™, Simulink, Simulink Coder, Stateflow and ThingSpeak are registered trademarks of The MathWorks, Inc.

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

More Information:
www.beckhoff.com/EL2502-0010

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