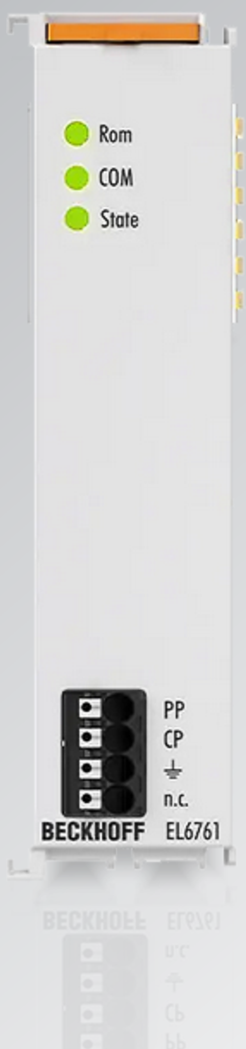
BECKHOFF New Automation Technology

Documentation | EN

EL6761

1-channel communication interface, ISO 15118 powerline, charge controller



2025-04-08 | Version: 1.0.0

Table of contents

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

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
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 Specific safety instructions

NOTICE

CP/PP Loss signal, index 6000:09 [▸ 103] (CP Loss), index 6000:0A [▸ 103] (PP Loss)

It should be noted that the CP/PP Loss signal can be used to comply with the switch-off times required by the standards.
For this purpose, the application must ensure that when the CP/PP Loss signal is disabled, the power flow is interrupted within the standardized switch-off times and the charging line is de-energized.

NOTICE

Maintaining an error state

To ensure that an error state is maintained as required by the standard (see IEC 61851-23) after an interruption of the CP signal, you must prevent an unintentional change to idle mode after the CP error at application level.

NOTICE

Unintended change of state

A change of state from C (charging) to A (plug not inserted) under 200 ms, which is not provided for in the standards, is recognized by the terminal as a wire break.
It must therefore be ensured that state B (charging completed, plug inserted) is always used between states C & A, as is also provided for in the standards. This eliminates the possibility of a misinterpreted wire break.

NOTICE

Using a fan cartridge

The use of a fan cartridge (ZB8610) for the terminal is mandatory from an ambient temperature of 60 °C in the "Optimum installation position (Standard)" (see chapter "Installation positions [▸ 34]").

1.5 Documentation issue status

Version	Comment
1.0.0	• First release
0.4.0	• Complements and corrections
0.3.0	• Complements and corrections
0.2.0	• Complements and corrections
0.1	• Provisional documentation for EL6761

1.6 Version identification of EtherCAT devices

1.6.1 General notes on marking

Designation

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

- family key
- type
- version
- revision

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

Notes

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

1.6.2 Version identification of EL terminals

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

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

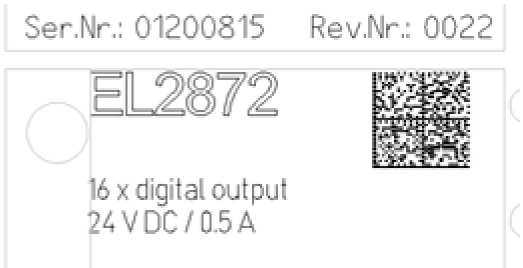


Fig. 1: EL2872 with revision 0022 and serial number 01200815

1.6.3 Beckhoff Identification Code (BIC)

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

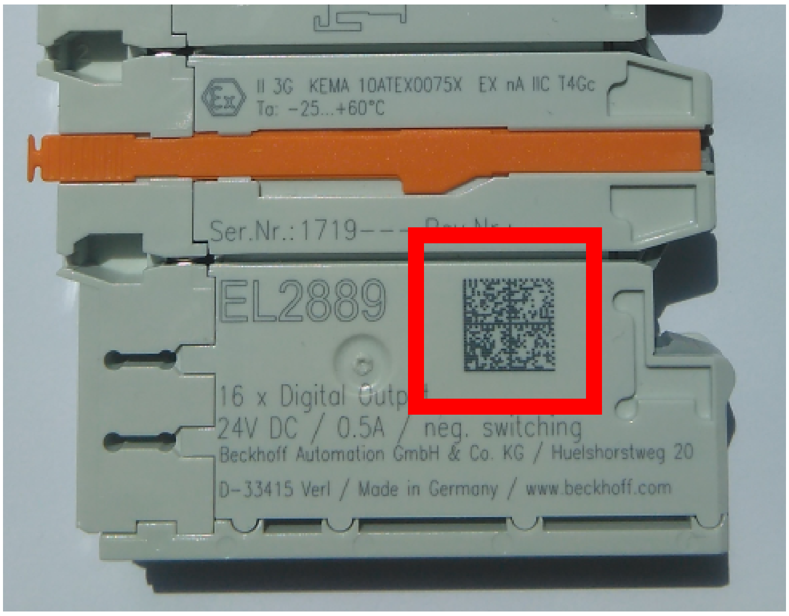


Fig. 2: BIC as data matrix code (DMC, code scheme ECC200)

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

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

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

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

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

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

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

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

Structure of the BIC

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

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

Accordingly as DMC:

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

BTN

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

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

1.6.4 Electronic access to the BIC (eBIC)

Electronic BIC (eBIC)

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

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

K-bus devices (IP20, IP67)

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

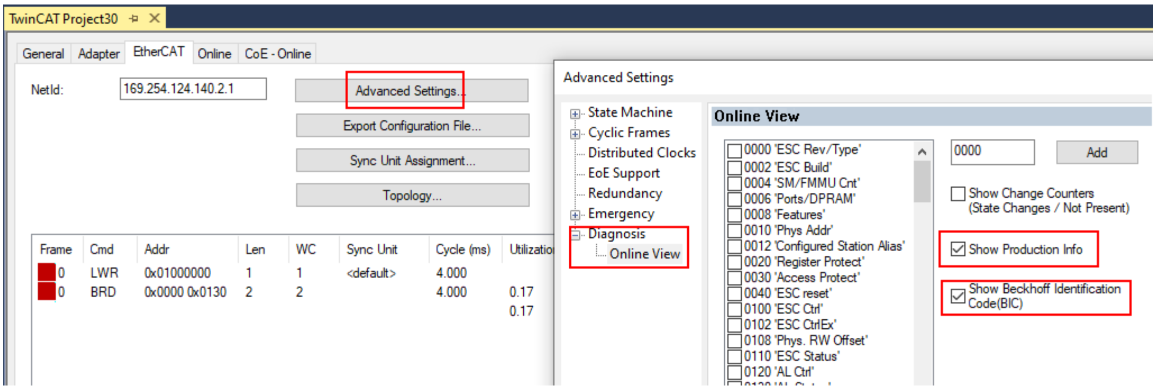
EtherCAT devices (IP20, IP67)

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

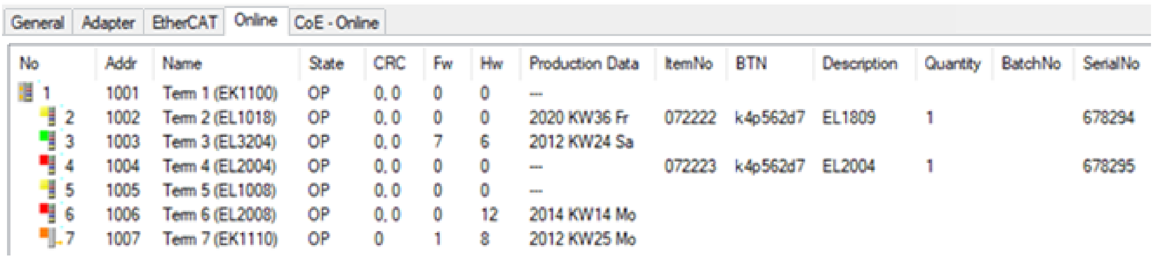
Beckhoff also stores the eBIC in the ESI-EEPROM. The eBIC was introduced into Beckhoff IO production (terminals, box modules) in 2020; as of 2023, implementation is largely complete.

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

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



  - The BTN and its contents are then displayed:



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

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

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

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

PROFIBUS; PROFINET, and DeviceNet devices

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

2 Product description

2.1 Introduction

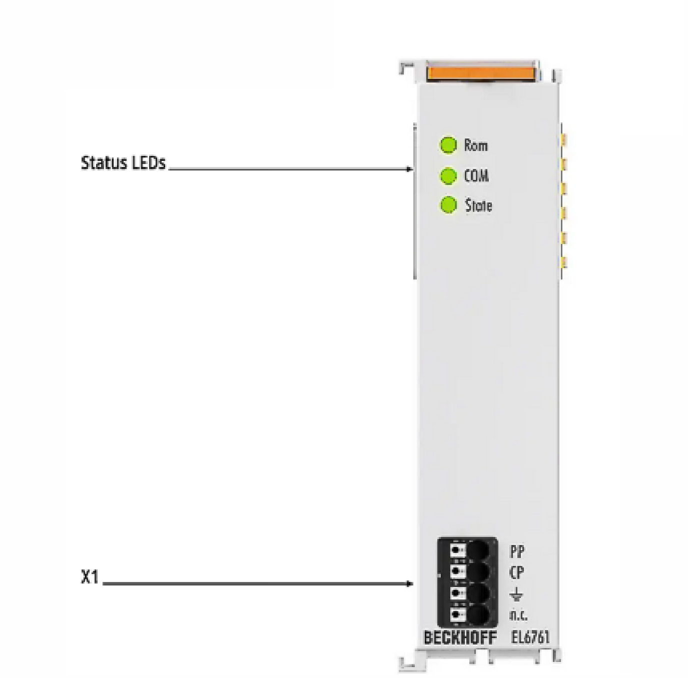


Fig. 4: EL6761 - 1-channel communication interface, ISO 15118 powerline, charge controller

The EL6761 EtherCAT Terminal enables communication for the charging infrastructure both to the EV (Electric Vehicle) and to the higher-level charging management system.

It supports two completely independent communication standards: PWM communication in accordance with IEC 61851 and powerline communication as stated in ISO 15118. ISO 15118 communication provides a flexible communication channel for transporting all necessary data between the car and the charging station.

In combination with TwinCAT 3 IoT OCPP (TF6771) for communication with higher-level controllers, a complete communication system for charging infrastructure applications is created.

2.2 Technical data

Technical data	EL6761
Technology	Powerline Communication
Fieldbus	ISO 15118
Number of fieldbus channels	1
Maximum cable length between terminal and EV	30 m (specified by ISO 15118)
Interfaces	CP, PP, (ground)
Communication	PWM communication in accordance with IEC 61851 and high-level communication in accordance with ISO 15118 and DIN 70121
Hardware diagnostics	Status LEDs
Electrical isolation	1500 V
Current consumption of power contacts	–
Current consumption via E-bus	420 mA typ.
Configuration	via TwinCAT System Manager
Weight	approx. 90 g
Permissible ambient temperature range during operation	-25 °C… +60 °C (extended temperature range)**)
Permissible ambient temperature range during storage	-40 °C… + 85 °C
Permissible relative air humidity	95 %, no condensation
Dimensions (W x H x D)	approx. 24 mm x 100 mm x 52 mm
Installation [▸ 27]	on 35 mm mounting rail, conforms to EN 50022
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
Installation position	variable
Approvals / markings*	CE

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

**) in standard installation position, see chapter "Installation positions [▸ 34]" and "Specific safety instructions [▸ 8]"

3 Basics communication

3.1 EtherCAT basics

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

3.2 General notes for setting the watchdog

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

The EtherCAT Slave Controller has two watchdogs:

- Sync manager (SM) watchdog (default: 100 ms)
- Process data (PDI) watchdog (default: 100 ms)

Their times are parameterized individually in TwinCAT as follows:

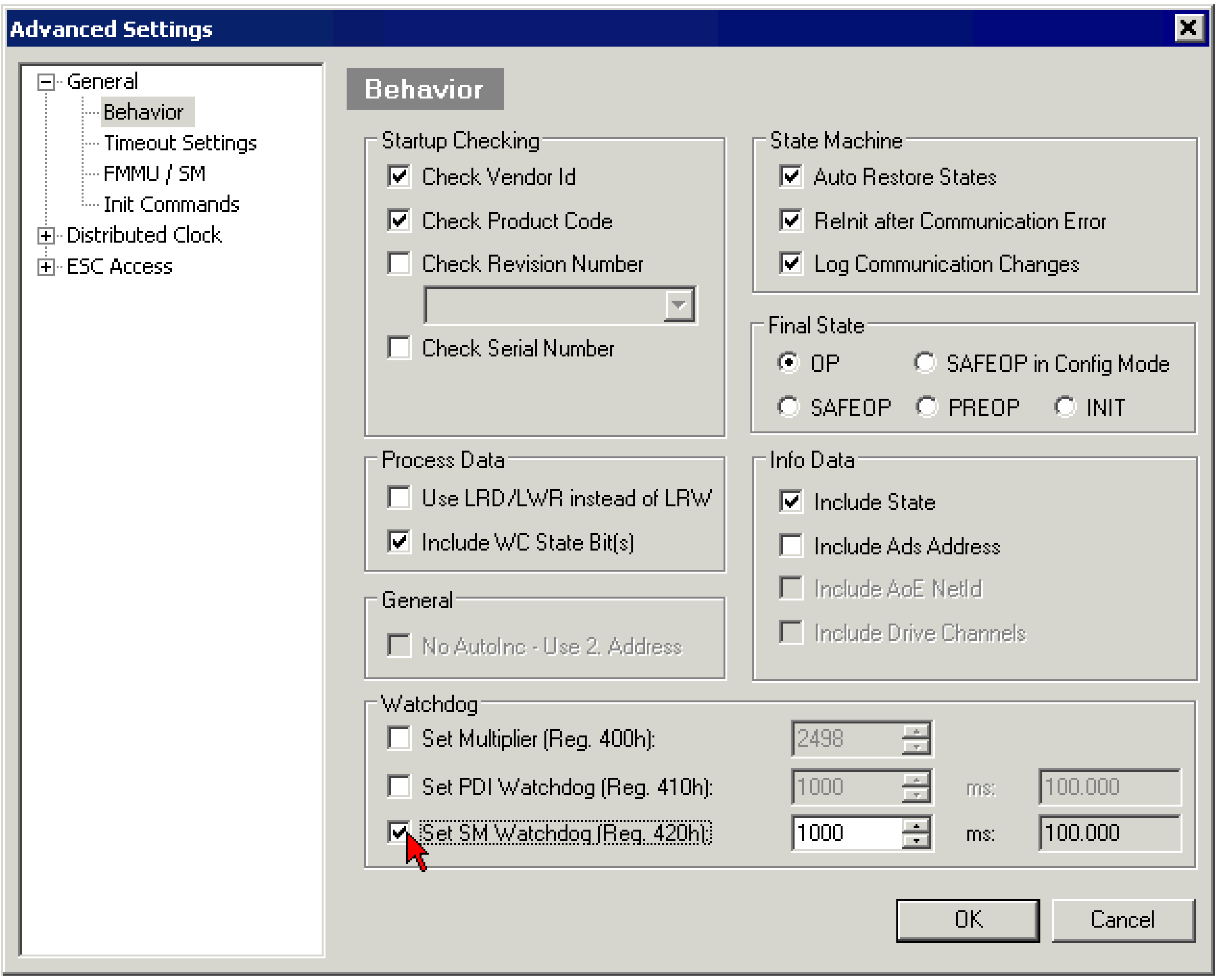


Fig. 5: EtherCAT tab -> Advanced Settings -> Behavior --> Watchdog

Remarks:

- The Multiplier register 400h (hexadecimal, i.e. 0x0400) is valid for both watchdogs.
- Each watchdog has its own timer setting 410h or 420h, which together with the multiplier results in a time.

- Important: the multiplier/timer setting is only loaded into the slave at EtherCAT start if the checkbox in front of it is activated.
  If this is not activated, nothing is downloaded and the setting in the ESC remains unchanged.
- The downloaded values can be viewed in the ESC registers 400h, 410h and 420h: ESC Access -> Memory (ESC Access -> Memory).

SM watchdog (SyncManager watchdog)

The SyncManager watchdog is reset with every successful EtherCAT Terminal process data communication. If no EtherCAT process data communication takes place with the terminal for longer than the set and activated SM watchdog time, e.g. in the event of a line interruption, the watchdog is triggered. The status of the terminal (usually OP) remains unaffected. The watchdog is only reset after a successful EtherCAT process data access.

The SyncManager watchdog therefore monitors correct and timely process data communication between the master and ESC, which takes place exclusively at EtherCAT level.

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

PDI watchdog (Process Data Watchdog)

If there is no PDI (Process Data Interface) communication with the ESC for longer than the set and activated PDI watchdog time, this watchdog is triggered.
PDI is the internal interface of the ESC, e.g. to local processors in the EtherCAT slave. The PDI watchdog can be used to monitor this communication for failure.

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

Calculation

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

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

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

⚠ CAUTION

Damage to the equipment and unintended behavior of the system is possible!

If the SM watchdog is activated and a value of 0 is entered the watchdog switches off completely. This is watchdog deactivation! Outputs are then NOT set to a safe state, in the event of an interruption in communication!

3.3 EtherCAT State Machine

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

A distinction is made between the following states:

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

- Bootstrap

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

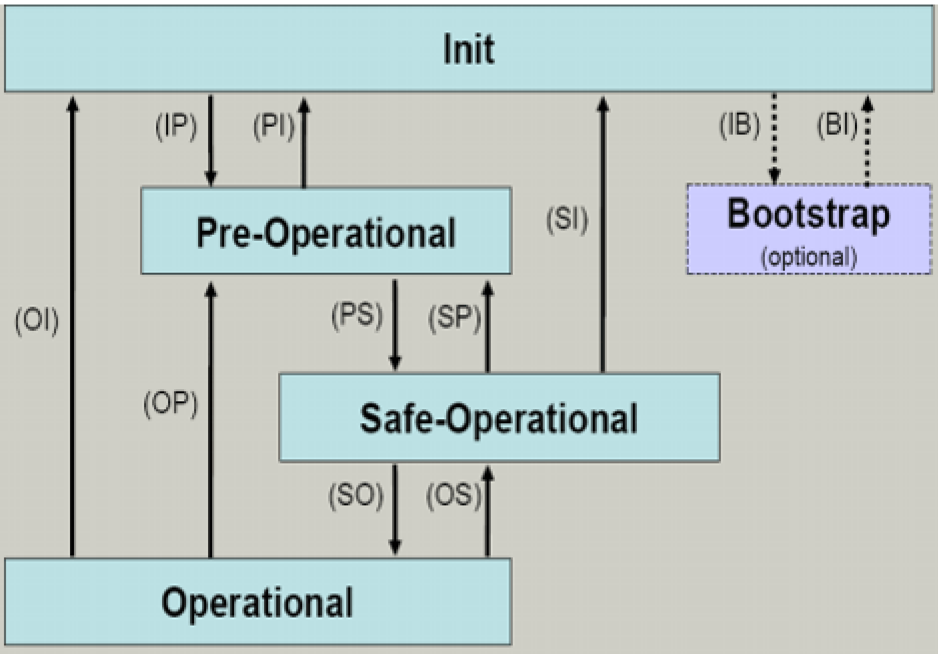


Fig. 6: States of the EtherCAT State Machine

Init

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

Pre-Operational (Pre-Op)

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

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

Safe-Operational (Safe-Op)

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

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

Outputs in SAFEOP state

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

Operational (Op)

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

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

Boot

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

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

3.4 CoE Interface

General description

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

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

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

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

The value ranges are

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

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

The relevant ranges for EtherCAT fieldbus users are:

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

Other important ranges are:

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

Availability

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

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

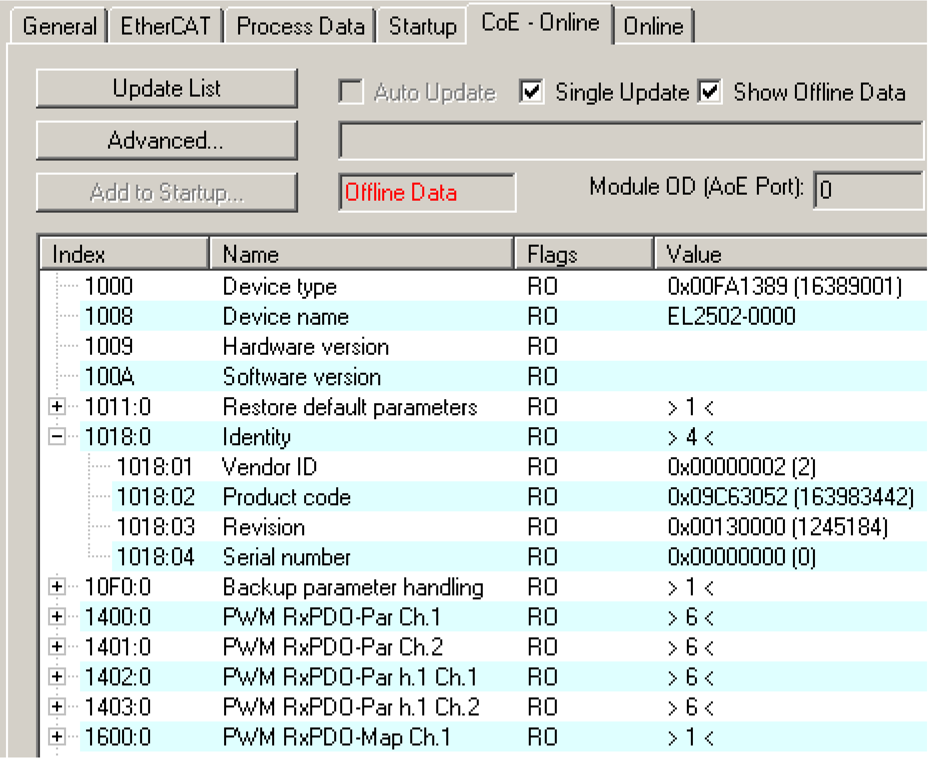


Fig. 7: “CoE Online” tab

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

NOTICE

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

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

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

Data management and function “NoCoeStorage”

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

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

Data management

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

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

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

Startup list

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

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

Recommended approach for manual modification of CoE parameters

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

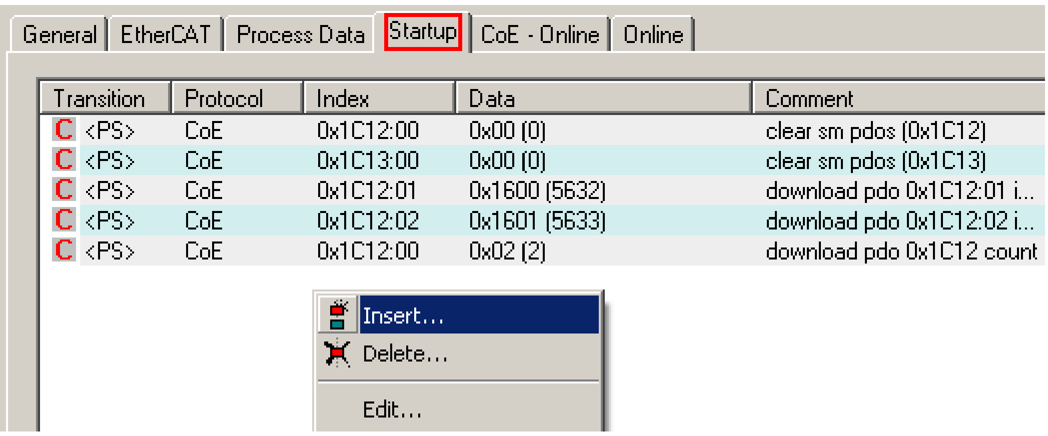


Fig. 8: Startup list in the TwinCAT System Manager

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

Online / offline list

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

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

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

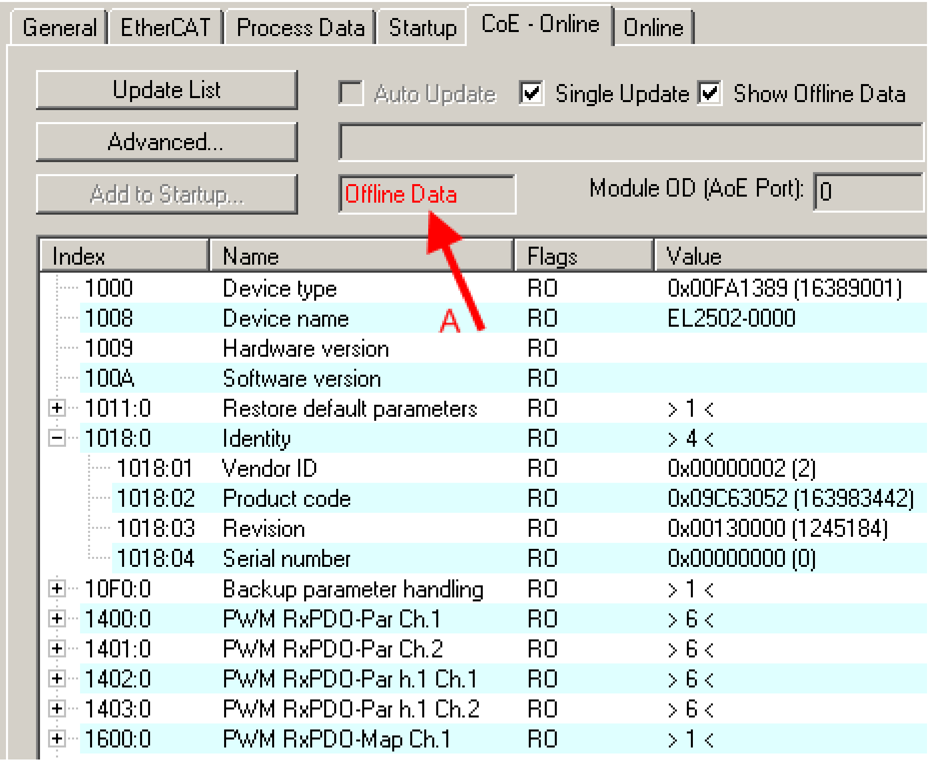


Fig. 9: Offline list

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

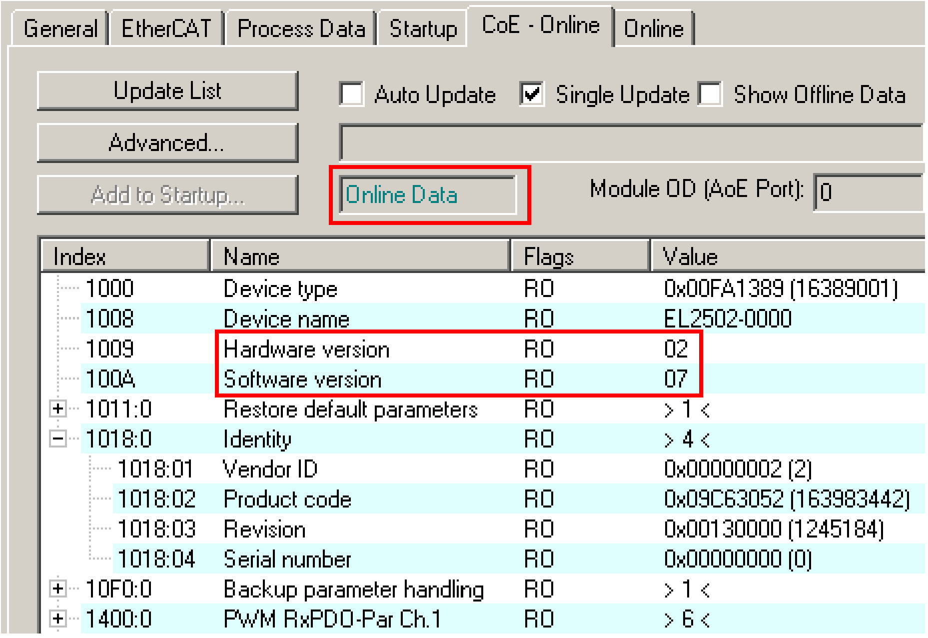


Fig. 10: Online list

Channel-based order

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

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

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

This is generally written as 0x80n0.

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

3.5 Distributed Clock

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

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

For detailed information please refer to the EtherCAT system description.

4 Mounting and wiring

4.1 Instructions for ESD protection

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.

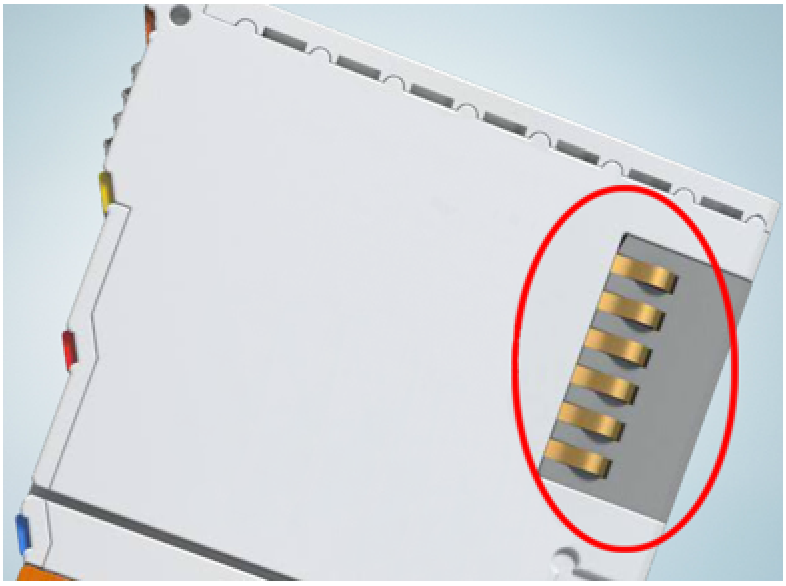


Fig. 11: Spring contacts of the Beckhoff I/O components

4.2 Note - power supply

⚠ WARNING

Power supply from SELV / PELV power supply unit!

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

Notes:

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

4.3 Mounting and demounting - top front unlocking

The terminal modules are fastened to the assembly surface with the aid of a 35 mm mounting rail (e. g. mounting rail TH 35-15).

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 recommended mounting rails under the terminals and couplers, you should use flat mounting connections (e.g. countersunk screws or blind rivets).

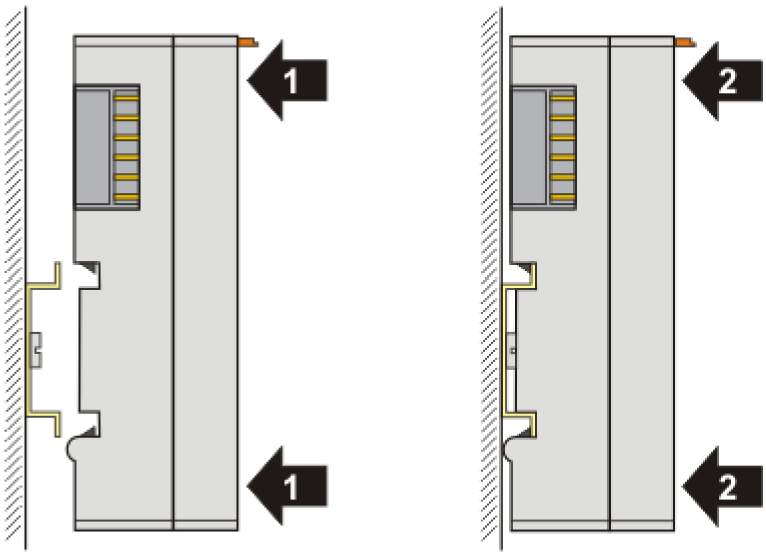
⚠ WARNING

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!

Mounting

- Fit the mounting rail to the planned assembly location.

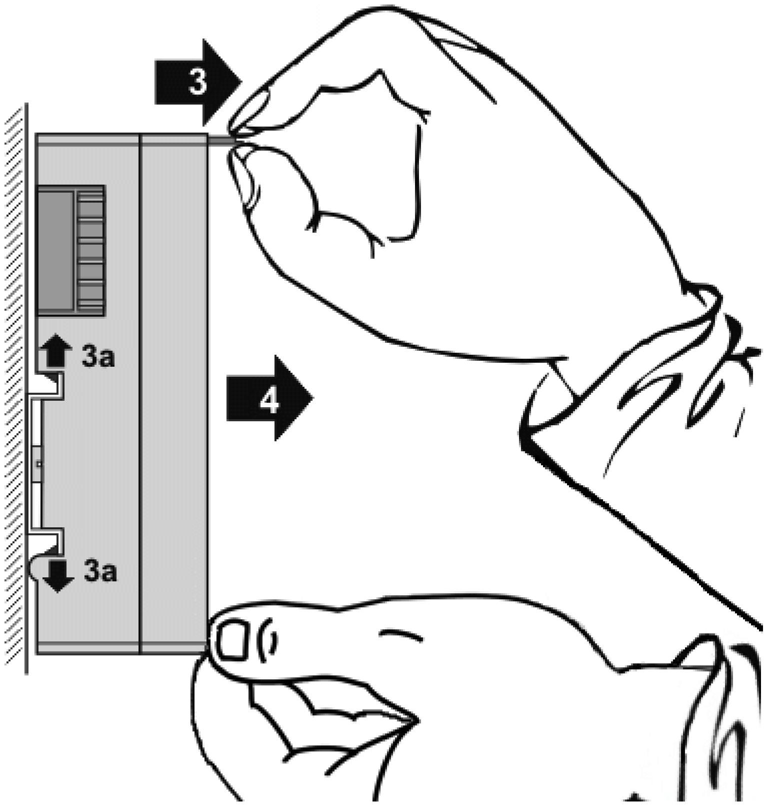


and press (1) the terminal module against the mounting rail until it latches in place on the mounting rail (2).

- Attach the cables.

Demounting

- Remove all the cables.
- Lever the unlatching hook back with thumb and forefinger (3). An internal mechanism pulls the two latching lugs (3a) from the top hat rail back into the terminal module.



- Pull (4) the terminal module away from the mounting surface.
  Avoid canting of the module; you should stabilize the module with the other hand, if required.

4.4 Mounting and demounting - traction lever unlocking

The terminal modules are fastened to the assembly surface with the aid of a 35 mm mounting rail (e. g. mounting rail TH 35-15).

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 recommended mounting rails under the terminals and couplers, you should use flat mounting connections (e.g. countersunk screws or blind rivets).

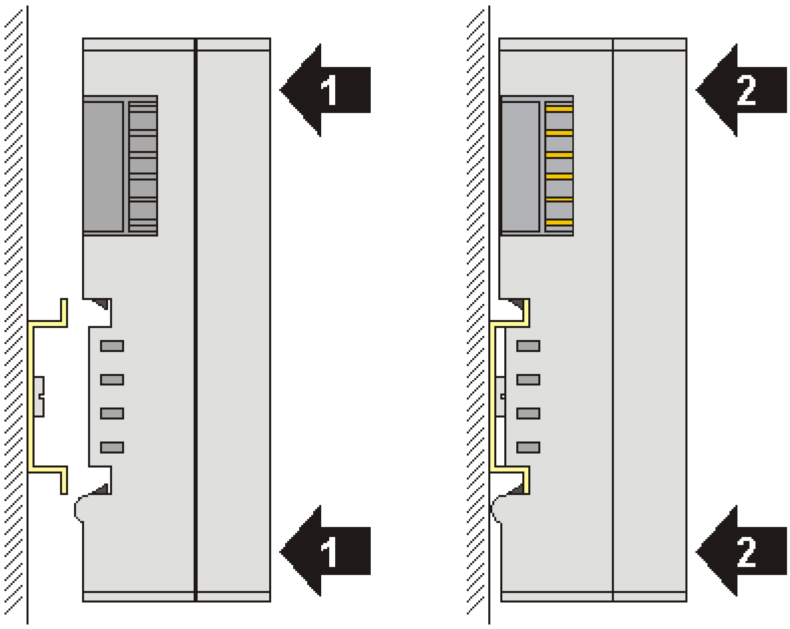
⚠ WARNING

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!

Mounting

- Fit the mounting rail to the planned assembly location.

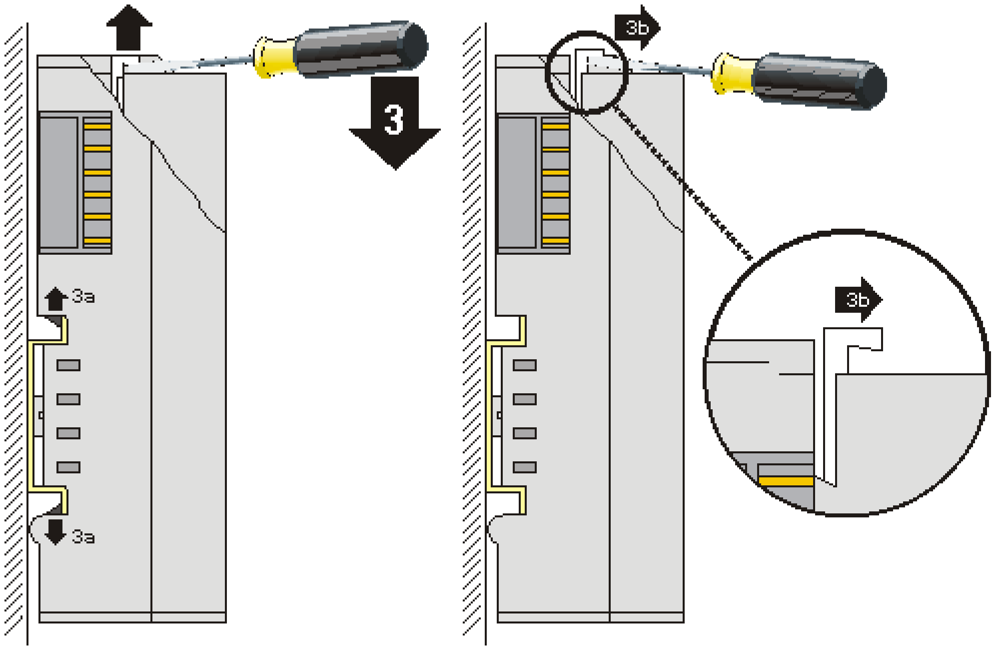


and press (1) the terminal module against the mounting rail until it latches in place on the mounting rail (2).

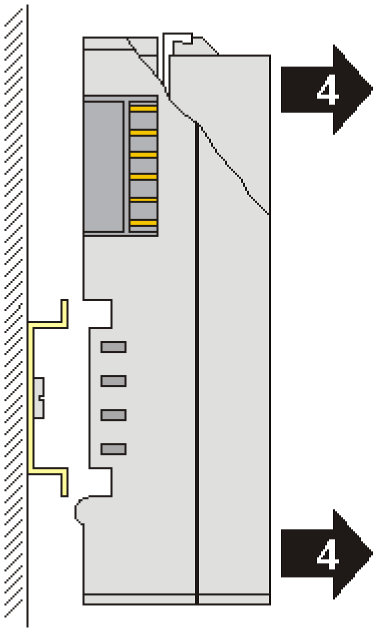
- Attach the cables.

Demounting

- Remove all the cables. Thanks to the KM/EM connector, it is not necessary to remove all the cables separately for this, but for each KM/EM connector simply undo 2 screws so that you can pull them off (fixed wiring)!
- Lever the unlatching hook on the left-hand side of the terminal module upwards with a screwdriver (3). As you do this
    - an internal mechanism pulls the two latching lugs (3a) from the top hat rail back into the terminal module,
    - the unlatching hook moves forwards (3b) and engages



- In the case 32 and 64 channel terminal modules (KMxxx4 and KMxxx8 or EMxxx4 and EMxxx8) you now lever the second unlatching hook on the right-hand side of the terminal module upwards in the same way.
- Pull (4) the terminal module away from the mounting surface.



4.5 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)

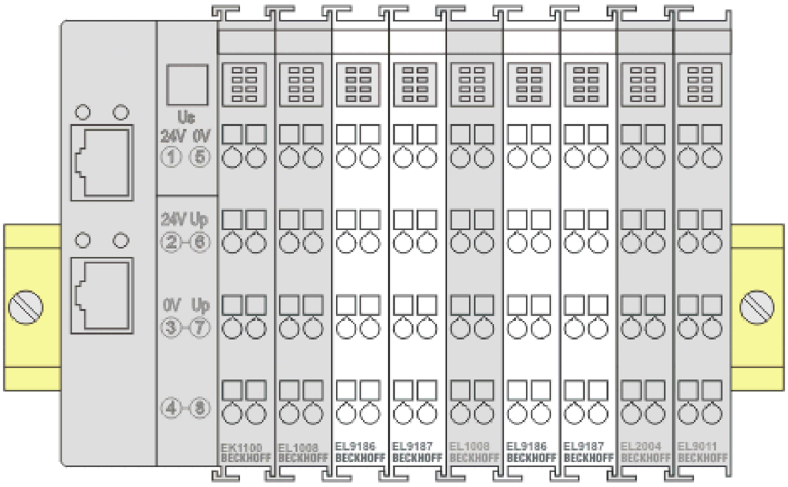


Fig. 12: Correct positioning

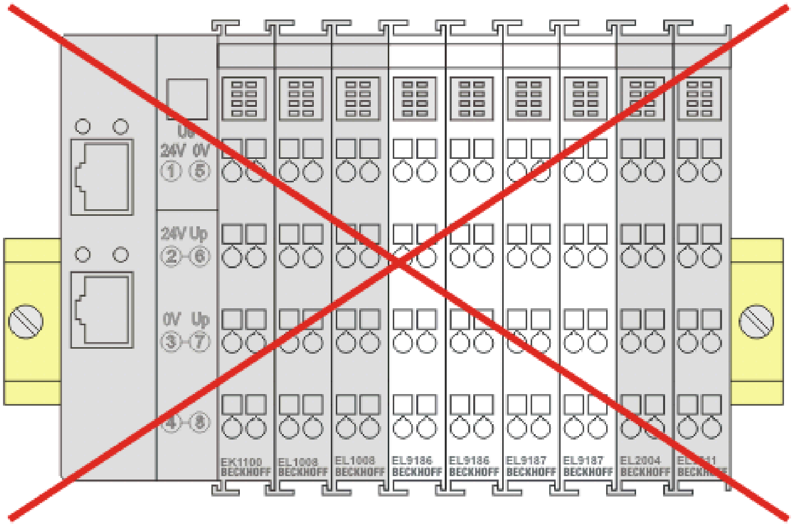


Fig. 13: Incorrect positioning

4.6 Installation positions

NOTICE

Constraints regarding installation position and operating temperature range

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

Optimum installation position (standard)

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

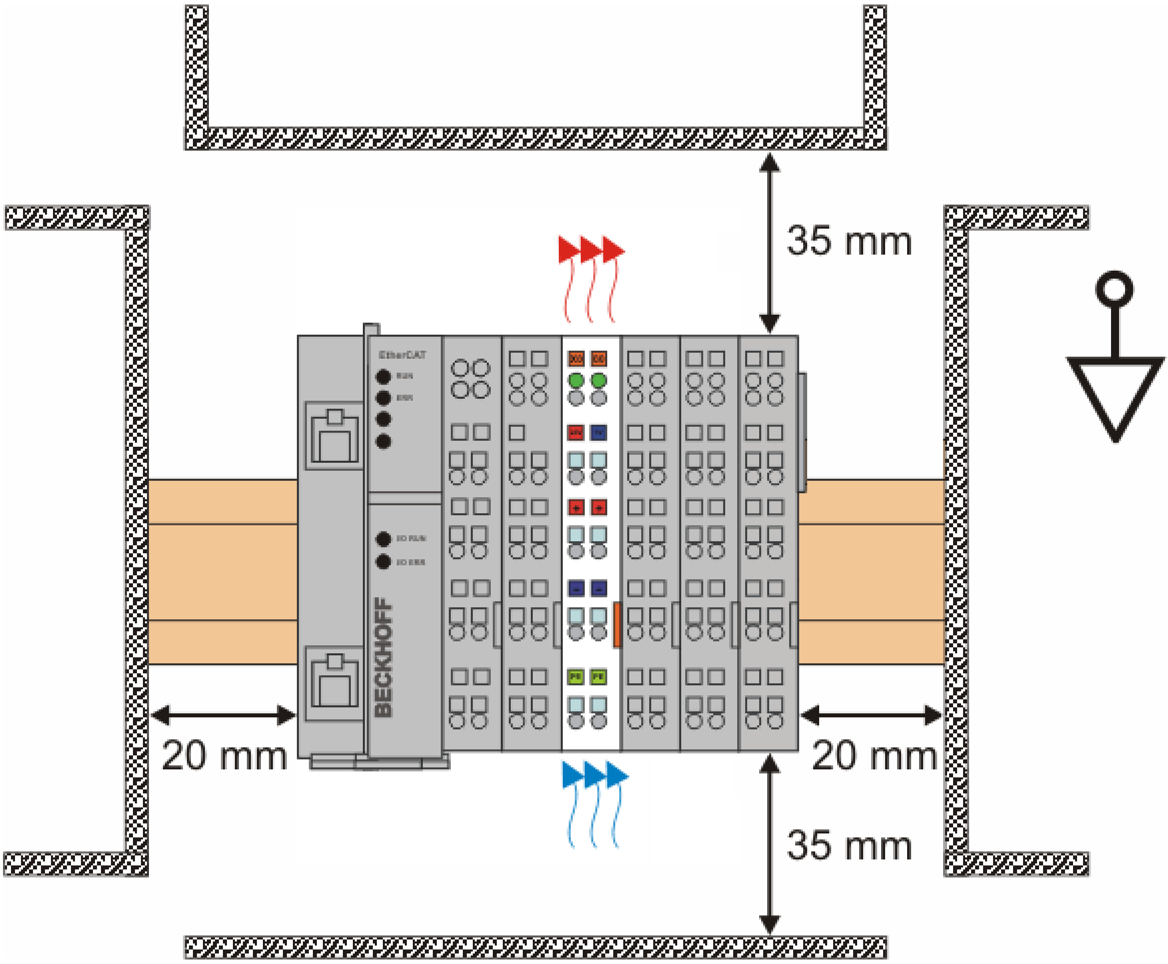


Fig. 14: Recommended distances for standard installation position

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

Other installation positions

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

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

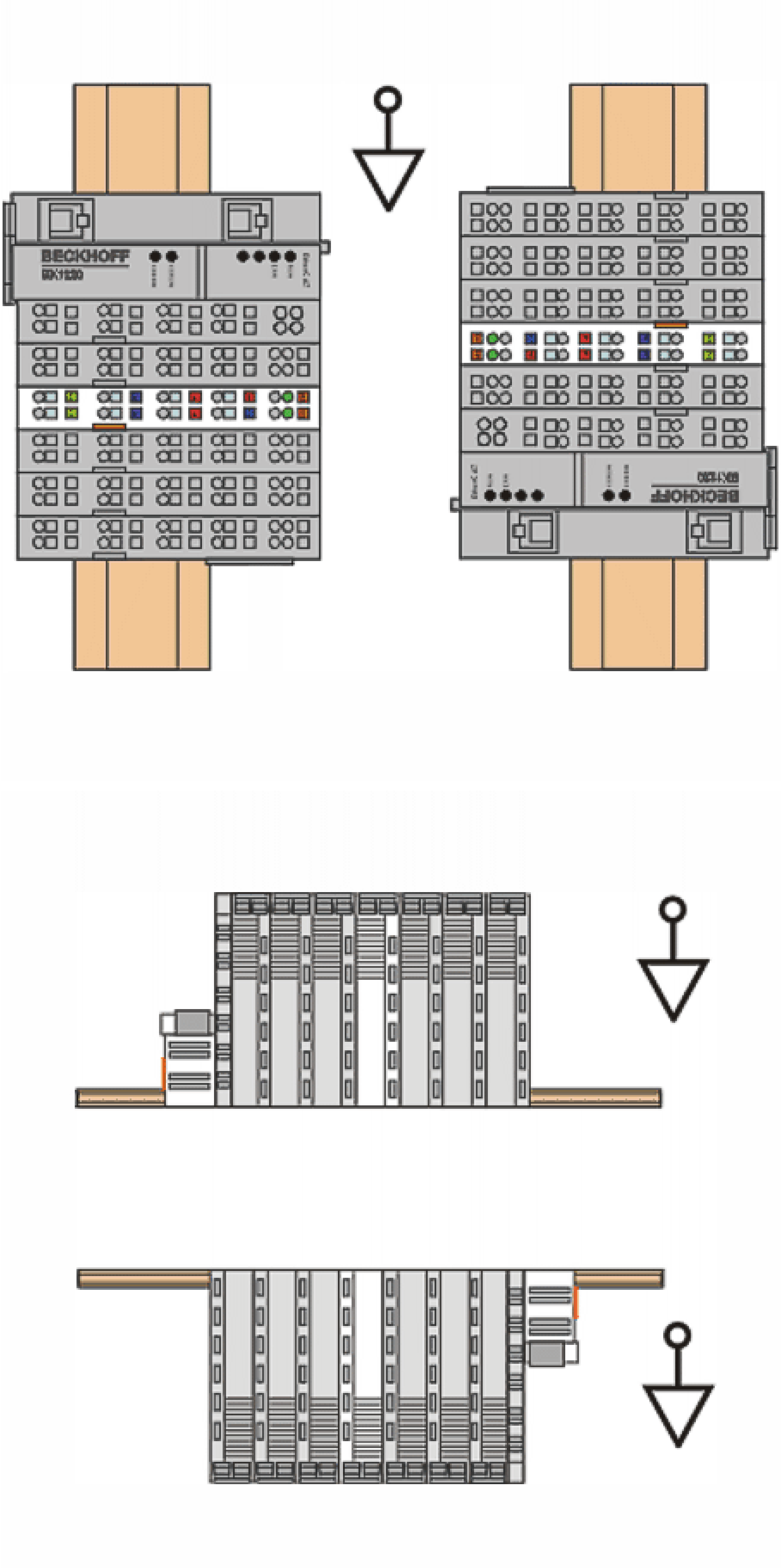


Fig. 15: Other installation positions

4.7 EL6761 - LEDs and connection

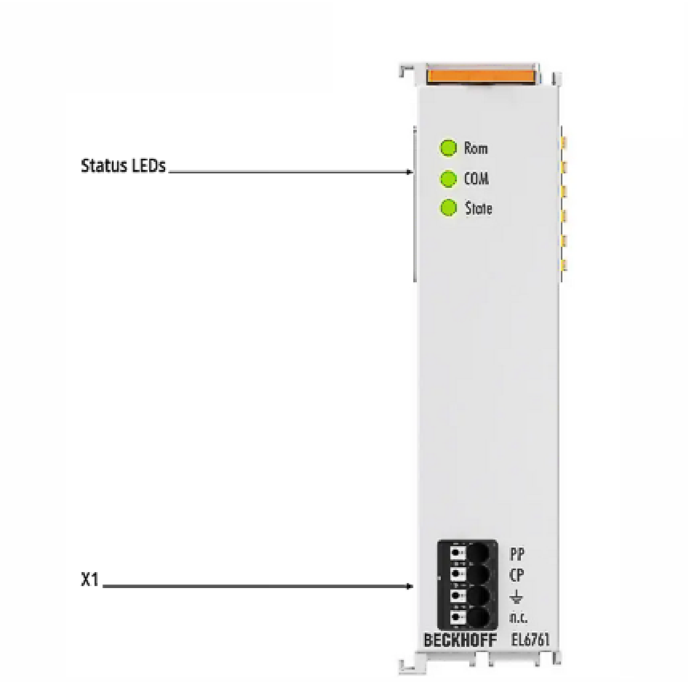


Fig. 16: EL6761

LED

LED	Color	Meaning	
Run	green	These LEDs indicate the terminal's operating state:	
		off	State of the EtherCAT State Machine [▸ 64]: **INIT** = initialization of the terminal or **BOOTSTRAP** = function for terminal firmware updates [▸ 126]
		flashing	State of the EtherCAT State Machine: **PREOP** = function for mailbox communication and different default settings set
		single flash	State of the EtherCAT State Machine: **SAFEOP** = verification of the Sync Manager [▸ 64] 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
Com	Displays the communication to the ISO 15118 module		
	green	off	No communication to the CME active
		on	Communication to the CME active
		flashing	Firmware update of the CME active
	yellow	flashing	File is read or written via FoE
	red	off	No error or reset active
		on	Error present or reset active
		flashing	-
State	Displays the status of communication with the EV		
	green	off	No EV connected
		on	Charge stop/pause
		flashing	Pre-Charge/Charge/Welding
		single flash	Pause (DC only - ISO 15118)
	yellow	off	No EV connected
		on	High level: Initialization phase (DC only - ISO 15118)
		flashing	Authentication/parameter phase
		single flash	Isolation phase
	red	off	-
		on	-
		flashing	Error present

Connections "X1"

Connection	Description	Function
PP	Proximity pin	Presence control pin Type1: - DC: presence recognition + evaluation of the plug lock - AC: optional (fixed cable) Type2: - DC: optional - AC: measures the current coded in the cable (if cable is not fixed)
⏚ "Ground"	Ground	Functional earth (FE), see Example circuit EL6761 [▸ 94]
CP	Control Pilot	Communication pin
n.c.	Not connected	Not connected

4.8 Notes on connection technology

⚠ **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 wires are plugged in directly; for solid wires no tools are required, i.e. after the insulation has been stripped, the wire is simply pushed into the contact point. The same applies for ferrules. Free stranded wire ends can also be connected in this way; in this case the wire clamping mechanism has to be opened by operating the pushing device.

Like in standard terminals, the wires are released via the contact release device, using a screwdriver or pushing device.

The cables must not be live or plugged/unplugged under load.

The permitted wire cross-sections and the strip length are shown in the following table.

Wire cross-section (solid wire)	0.2 ... 1.5 mm^2
Cable cross-section (stranded wire)	0.2 ... 1.5 mm^2
Cable cross-section (stranded wire)	0.25 ... 0.75 mm^2 (with ferrule with plastic collar)
Cable cross-section (stranded wire)	0.25 ... 1.5 mm^2 (with ferrule without plastic collar)
Current carrying capacity, continuous	5 A
Conductor (AWG)	24 – 14 | 14: THHN, THWN
Strip length	8 ... 9 mm / 0.31 – 0.35-inch

4.9 Disposal

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

5 Commissioning

5.1 TwinCAT Development Environment

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

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

Details:

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

Additional features:

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

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

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

5.1.1 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.

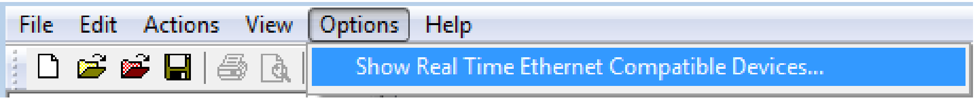


Fig. 17: System Manager “Options” (TwinCAT 2)

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

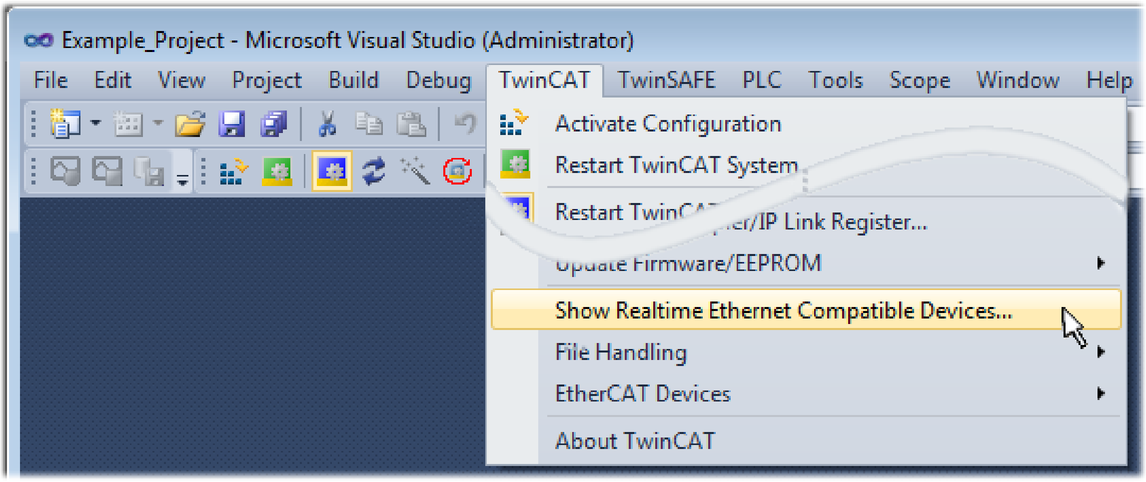


Fig. 18: Call up under VS Shell (TwinCAT 3)

B: Via TcRteInstall.exe in the TwinCAT directory

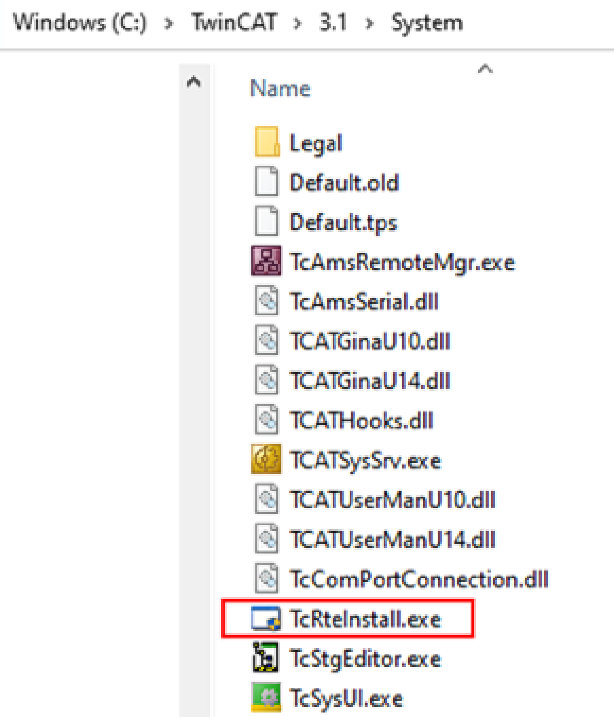


Fig. 19: TcRteInstall in the TwinCAT directory

In both cases, the following dialog appears:

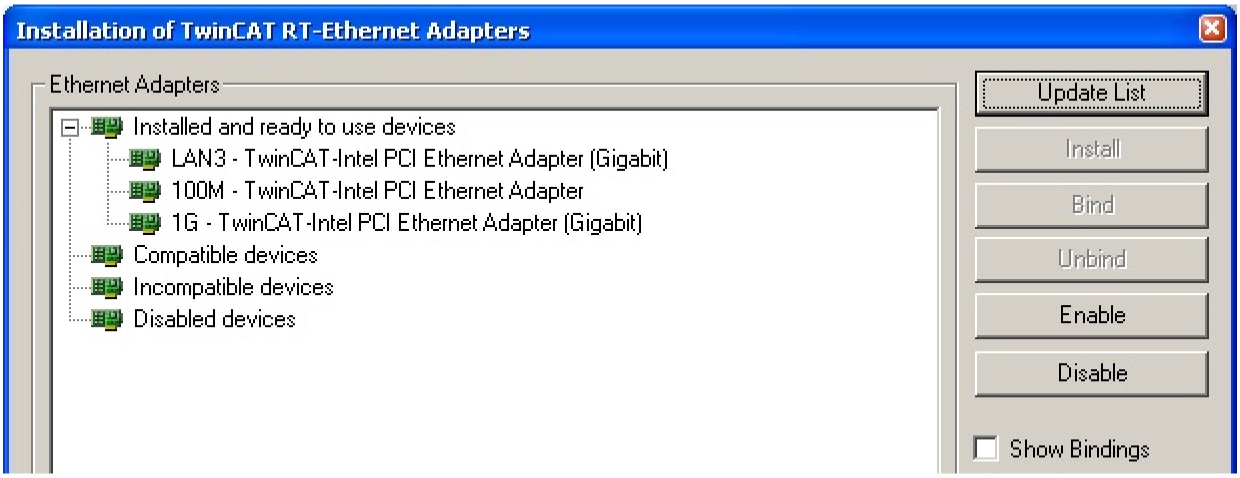


Fig. 20: Overview of network interfaces

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

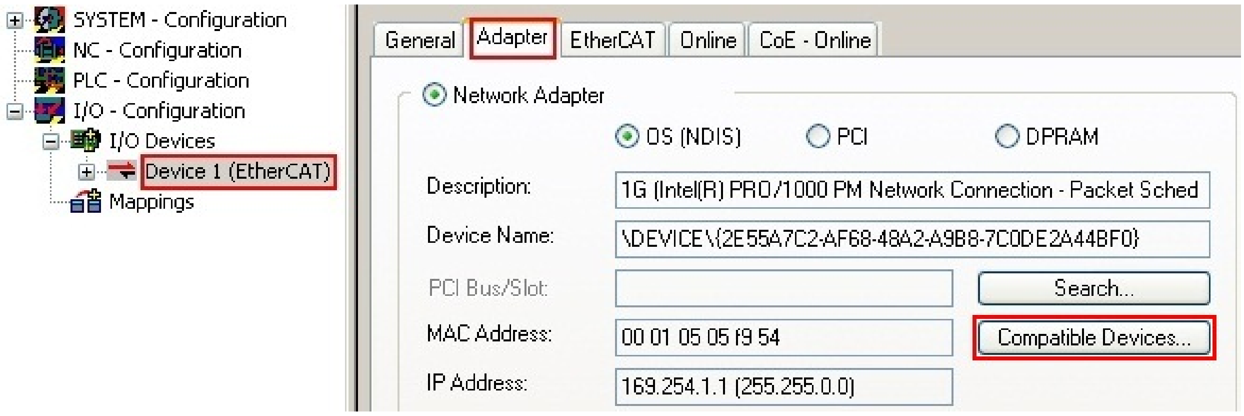


Fig. 21: EtherCAT device properties (TwinCAT 2): click on “Compatible Devices…” of tab “Adapter”

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)

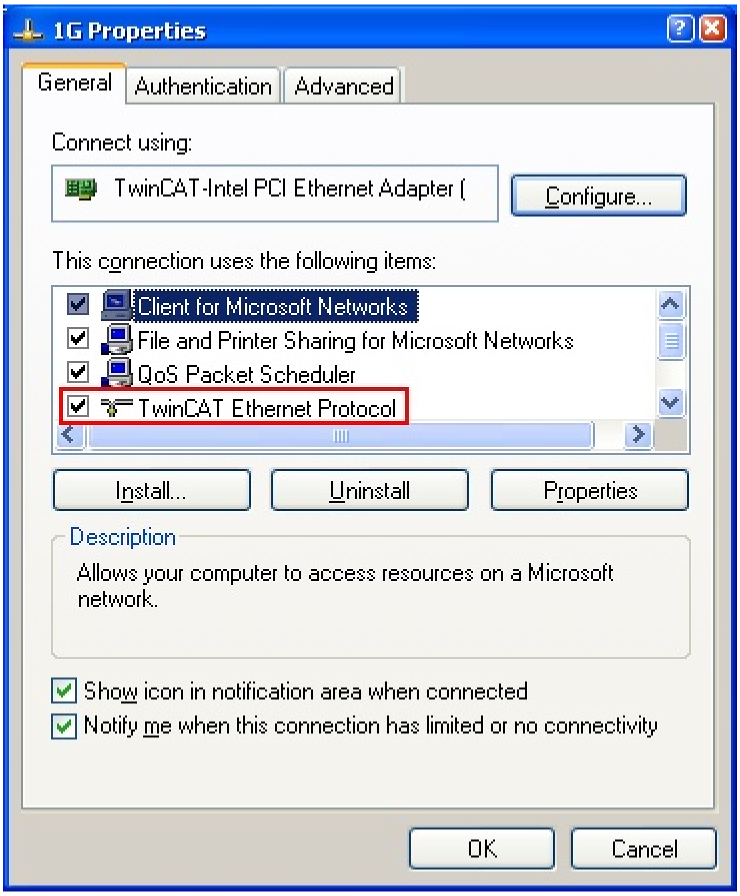


Fig. 22: Windows properties of the network interface

A correct setting of the driver could be:

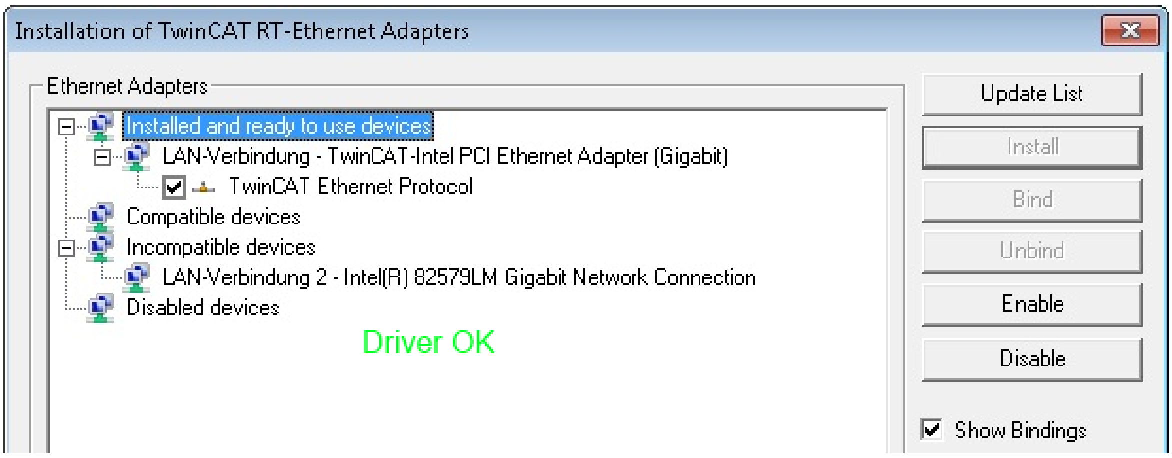


Fig. 23: Exemplary correct driver setting for the Ethernet port

Other possible settings have to be avoided:

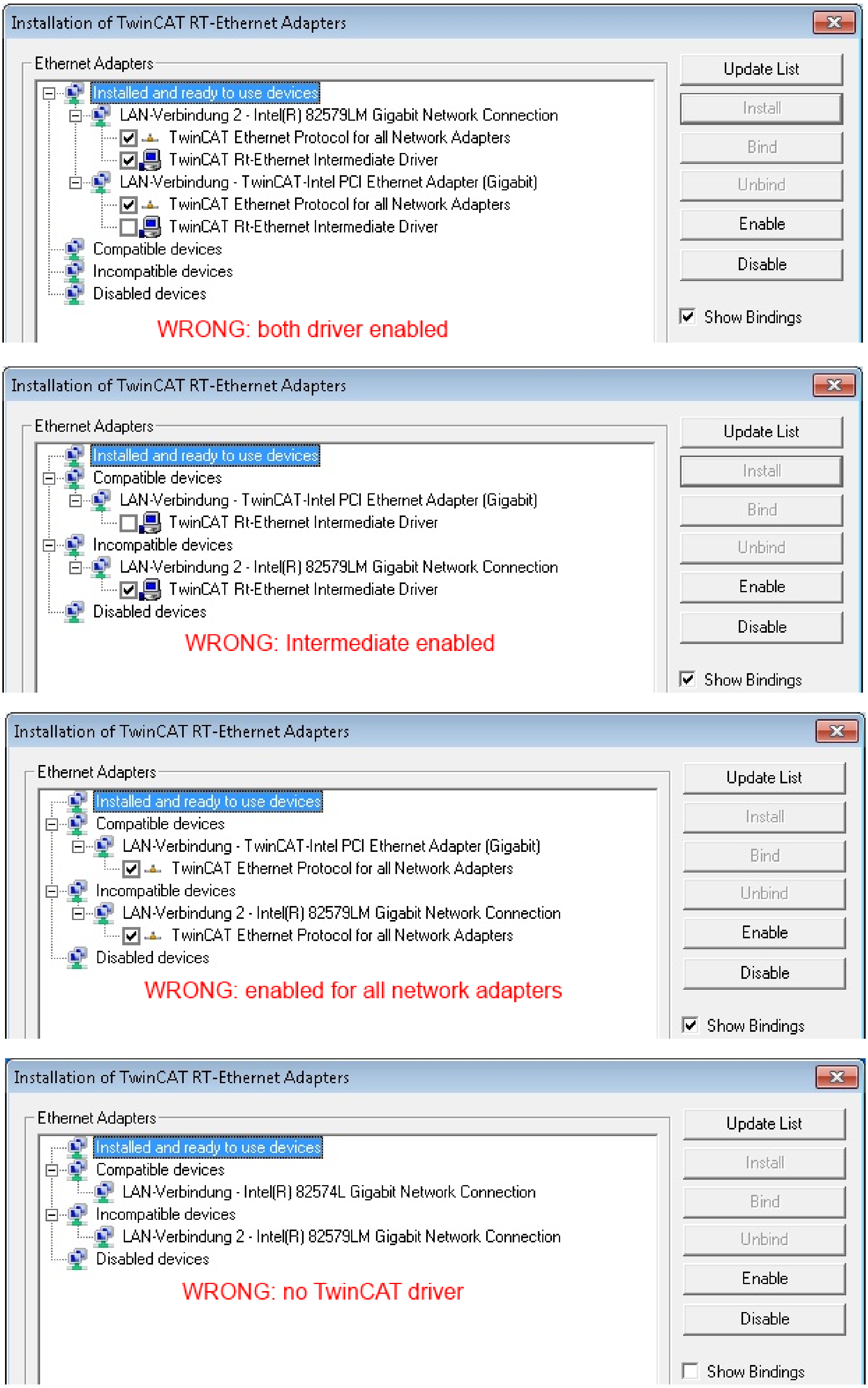


Fig. 24: Incorrect driver settings for the Ethernet port

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.

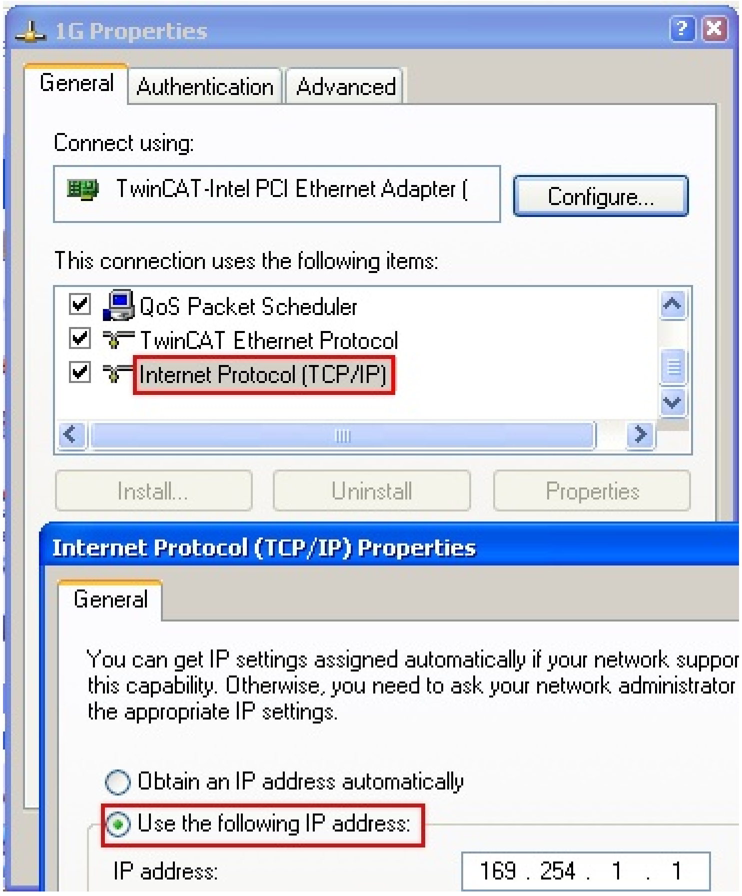


Fig. 25: TCP/IP setting for the Ethernet port

5.1.2 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 [▶ 50] 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”

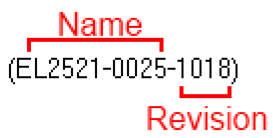


Fig. 26: Identifier structure

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

Online description

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

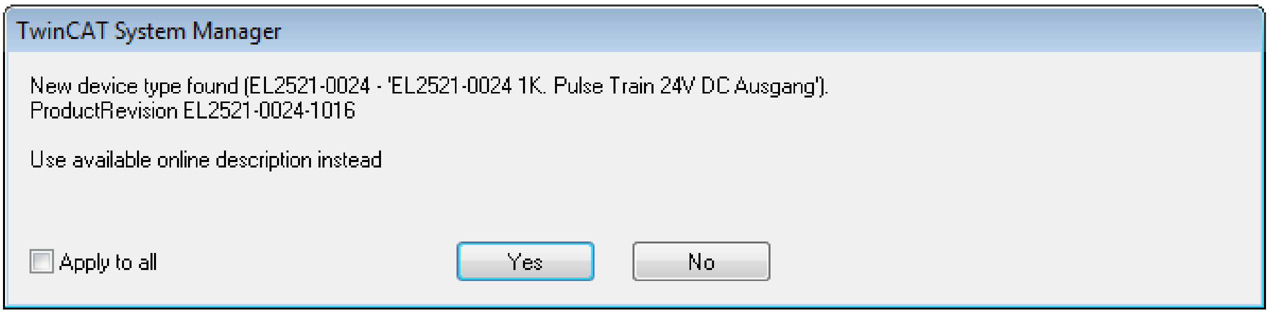


Fig. 27: OnlineDescription information window (TwinCAT 2)

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

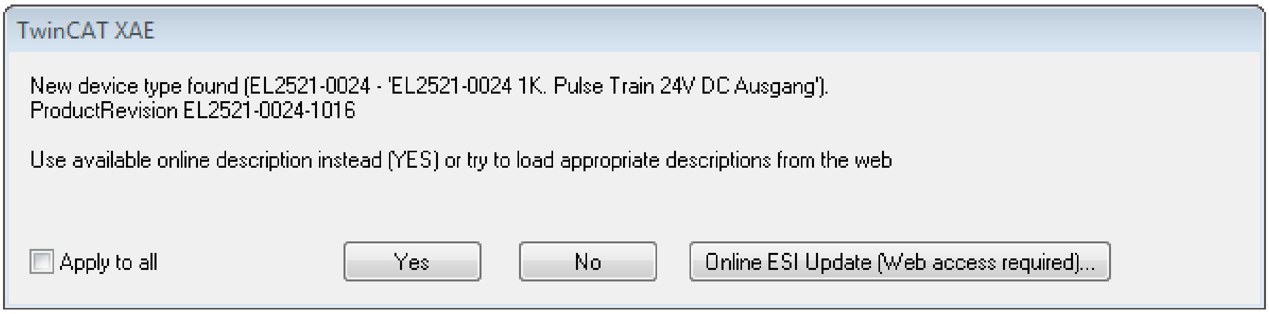


Fig. 28: Information window OnlineDescription (TwinCAT 3)

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 [▶ 51]”.

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.

Fig. 29: 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)*.

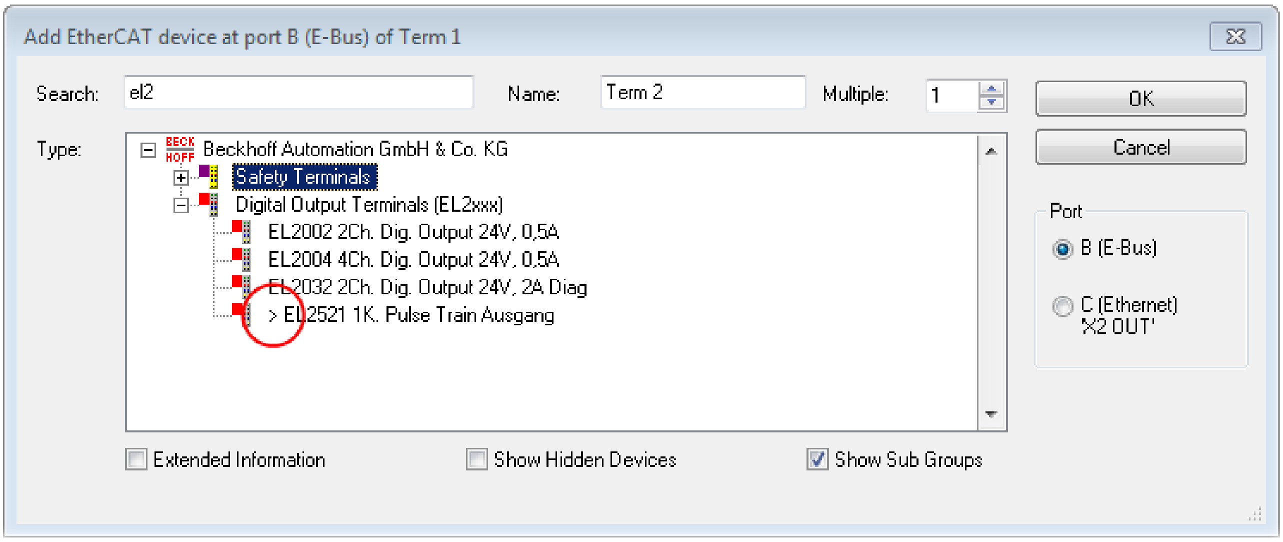


Fig. 30: Indication of an online recorded ESI of EL2521 as an example

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.

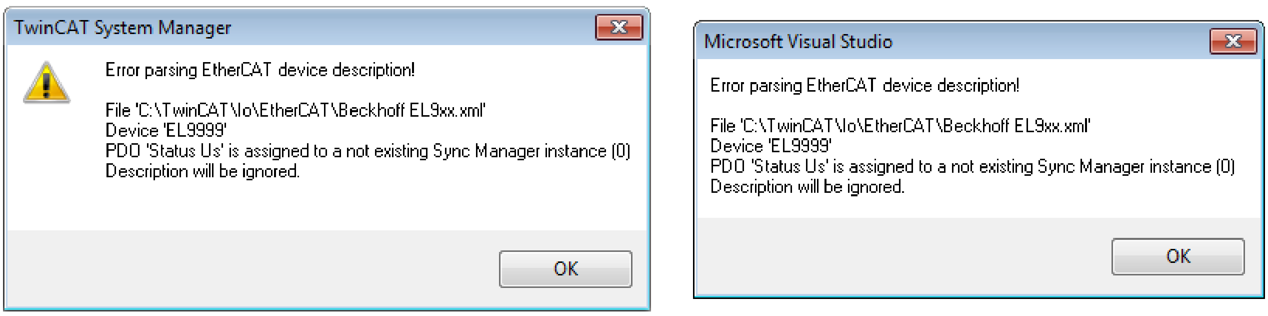


Fig. 31: Information window for faulty ESI file (left: TwinCAT 2; right: TwinCAT 3)

Reasons may include:

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

5.1.3 TwinCAT ESI Updater

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

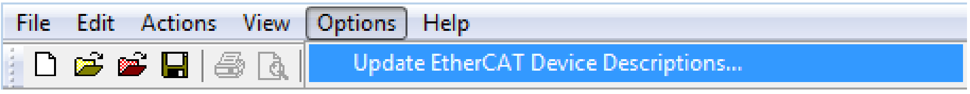


Fig. 32: Using the ESI Updater (>= TwinCAT 2.11)

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

Selection under TwinCAT 3:

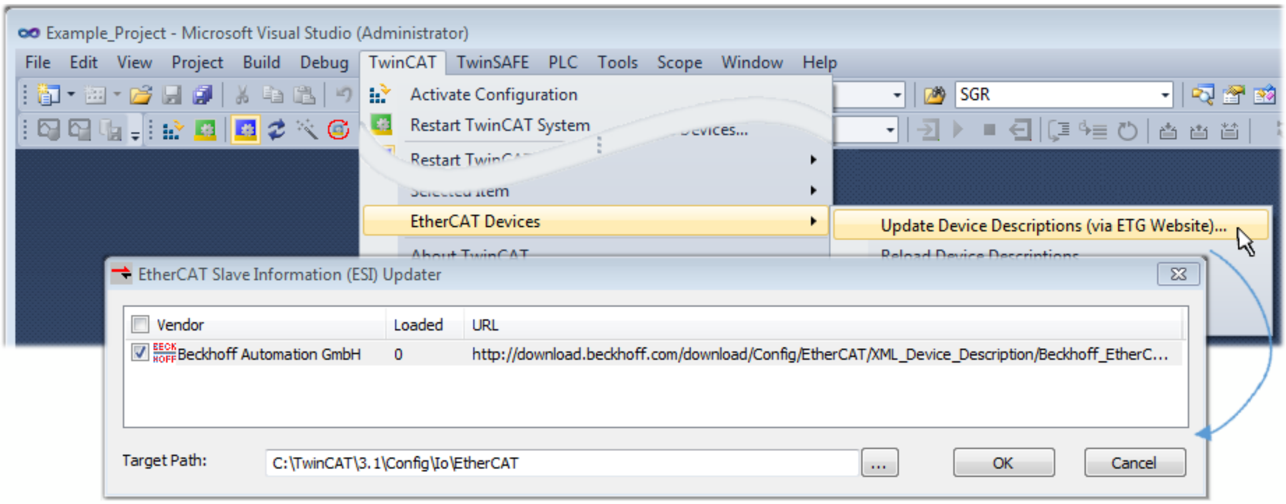


Fig. 33: Using the ESI Updater (TwinCAT 3)

The ESI Updater (TwinCAT 3) is a convenient option for automatic downloading of ESI data provided by EtherCAT manufacturers via the Internet into the TwinCAT directory (ESI = EtherCAT slave information). TwinCAT accesses the central ESI ULR directory list stored at ETG; the entries can then be viewed in the Updater dialog, although they cannot be changed there.

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

5.1.4 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” [▶ 46].

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 [▸ 56] (Ethernet port at the IPC)
- detecting the connected EtherCAT devices [▸ 57]. This step can be carried out independent of the preceding step
- troubleshooting [▸ 60]

The scan with existing configuration [▸ 61] can also be carried out for comparison.

5.1.5 OFFLINE configuration creation

Creating the EtherCAT device

Create an EtherCAT device in an empty System Manager window.

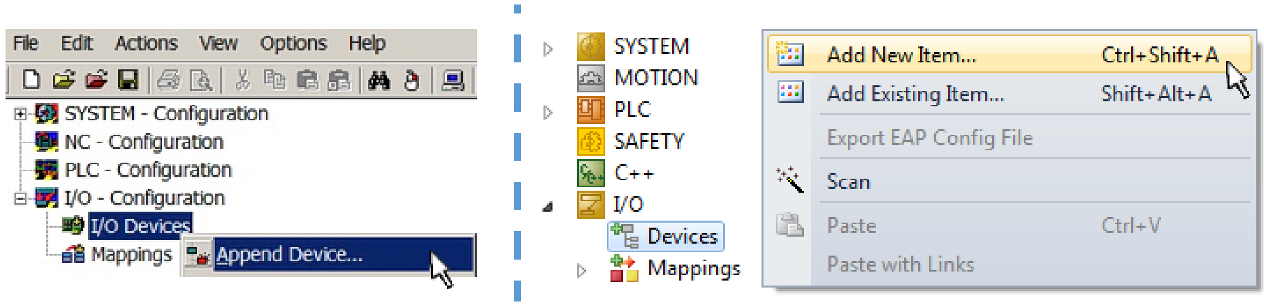


Fig. 34: Append EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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”.

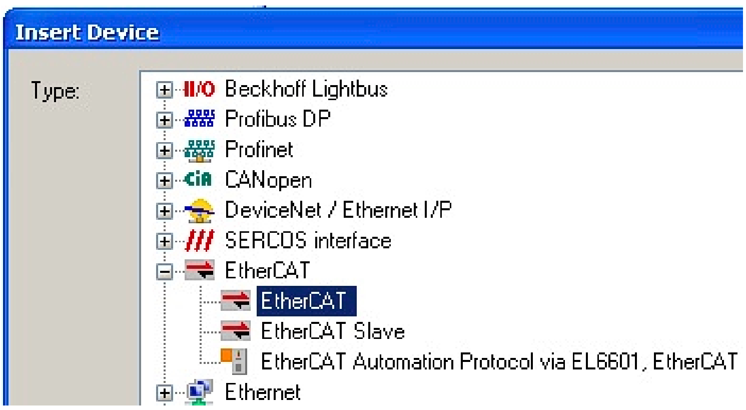


Fig. 35: Selecting the EtherCAT connection (TwinCAT 2.11, TwinCAT 3)

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

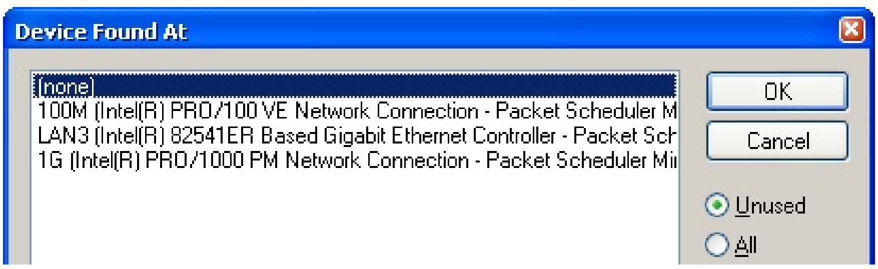


Fig. 36: Selecting the Ethernet port

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)”.

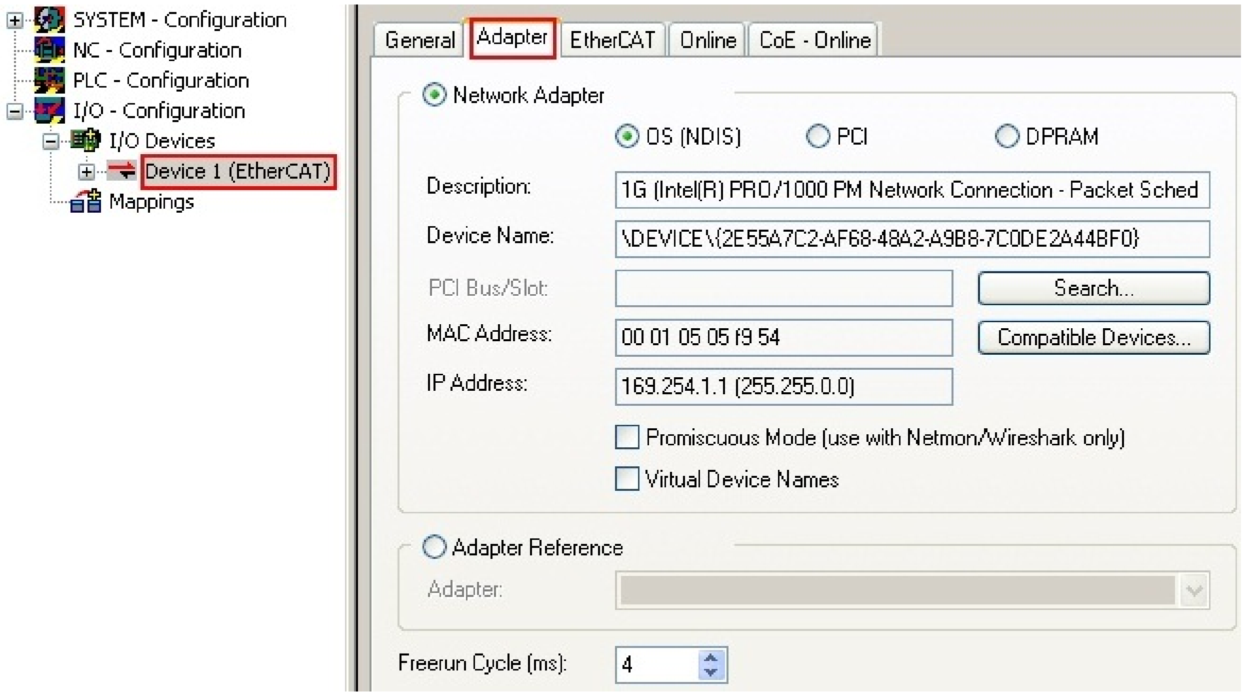
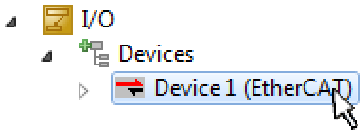


Fig. 37: EtherCAT device properties (TwinCAT 2)

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 [▸ 40].

Defining EtherCAT slaves

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

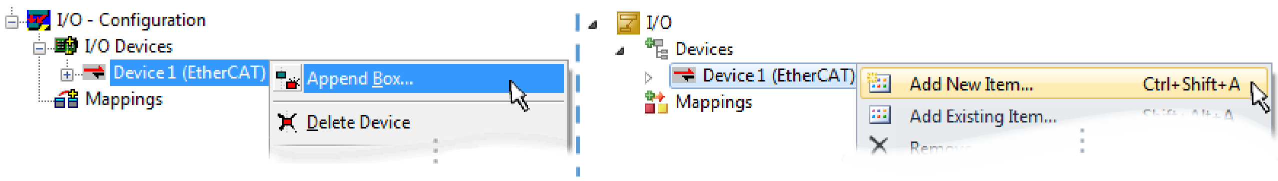


Fig. 38: Appending EtherCAT devices (left: TwinCAT 2; right: TwinCAT 3)

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

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

Overview of physical layer

- “Ethernet”: cable-based 100BASE-TX: couplers, box modules, devices with RJ45/M8/M12 connector

- "E-Bus": LVDS "terminal bus", EtherCAT plug-in modules (EJ), EtherCAT terminals (EL/ES), various modular modules

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

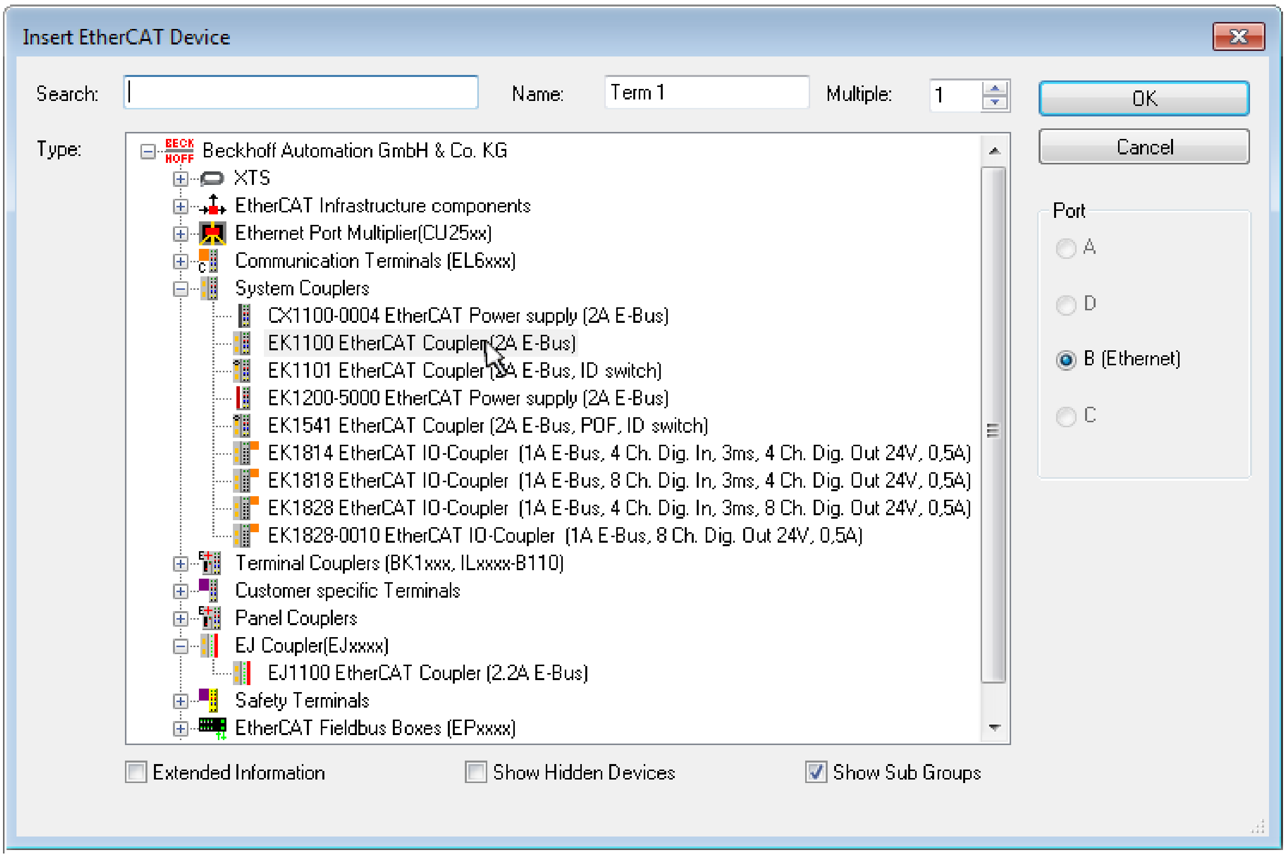


Fig. 39: Selection dialog for new EtherCAT device

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".

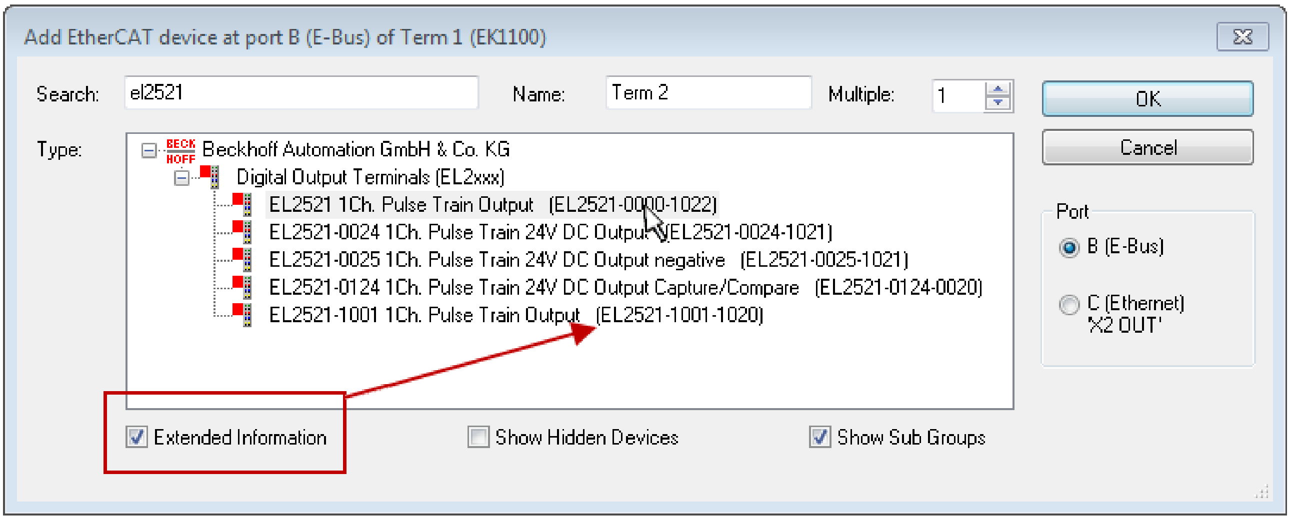


Fig. 40: Display of device revision

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".

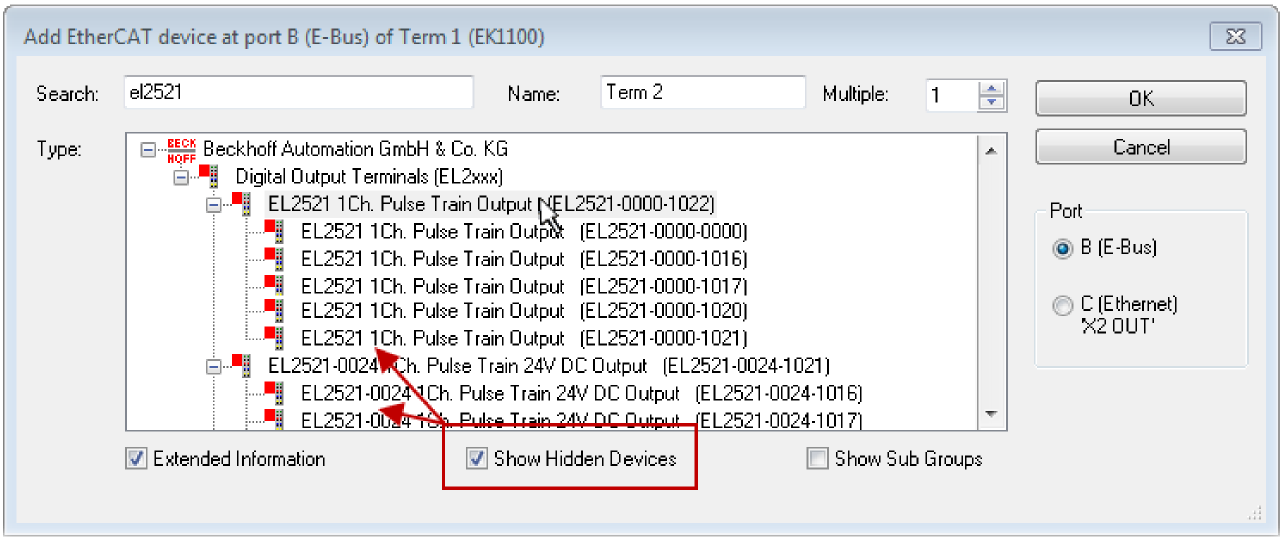


Fig. 41: Display of previous revisions

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.

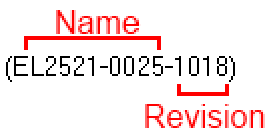


Fig. 42: Name/revision of the terminal

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, ...

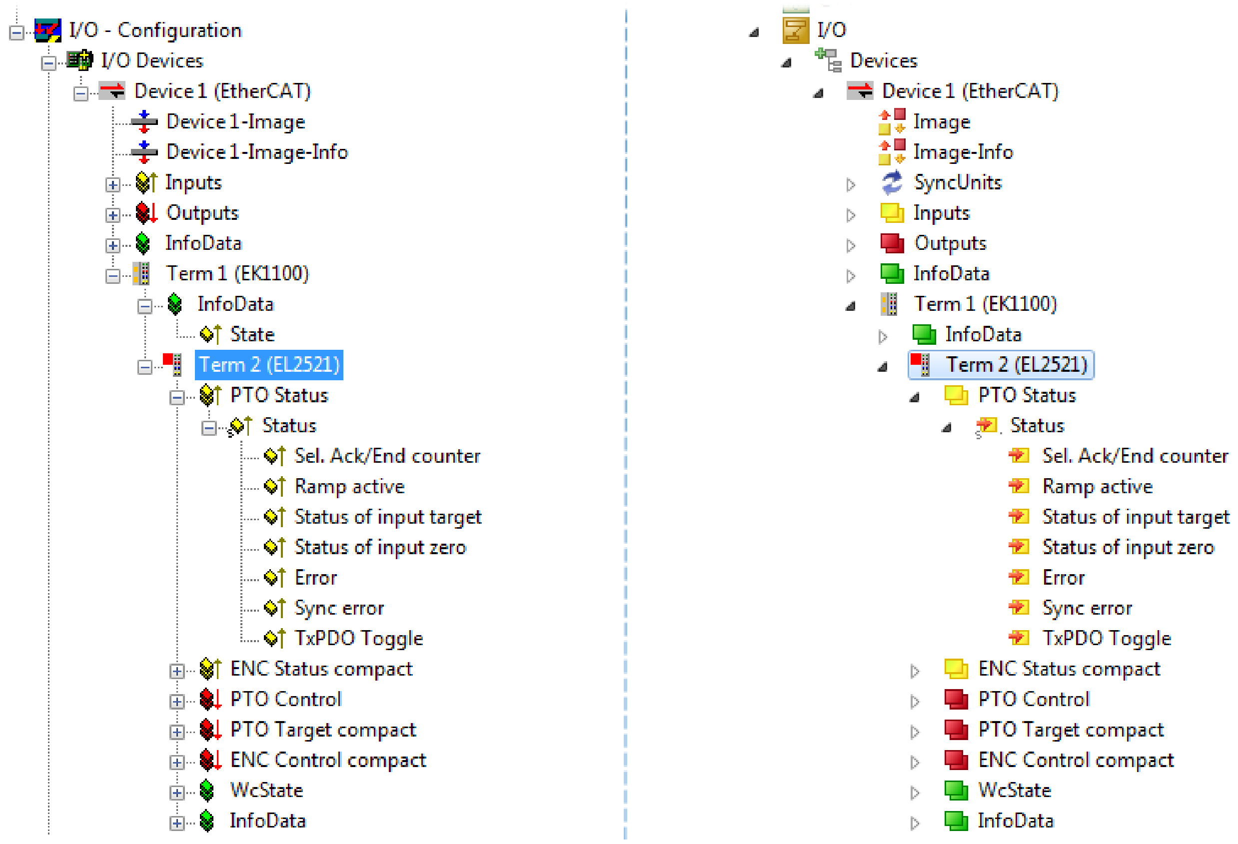


Fig. 43: EtherCAT terminal in the TwinCAT tree (left: TwinCAT 2; right: TwinCAT 3)

5.1.6 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.

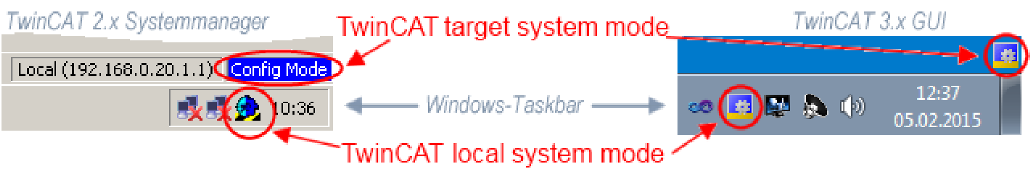


Fig. 44: Differentiation local/target system (left: TwinCAT 2; right: TwinCAT 3)

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

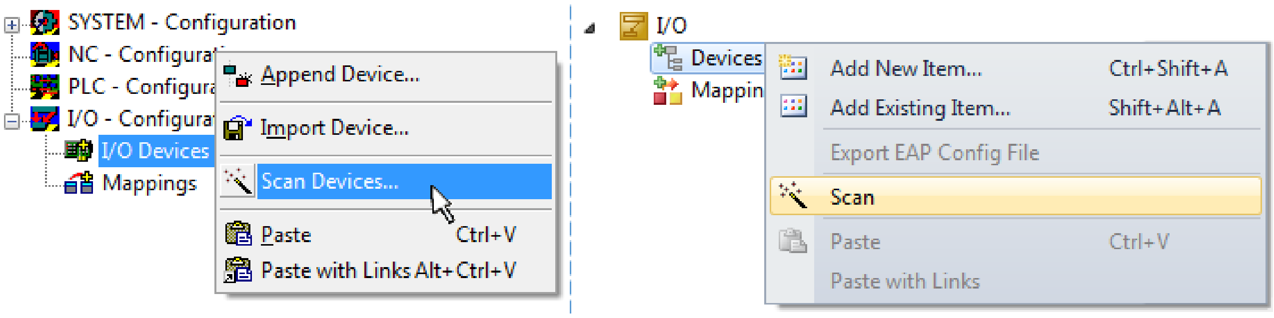


Fig. 45: Scan Devices (left: TwinCAT 2; right: TwinCAT 3)

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.

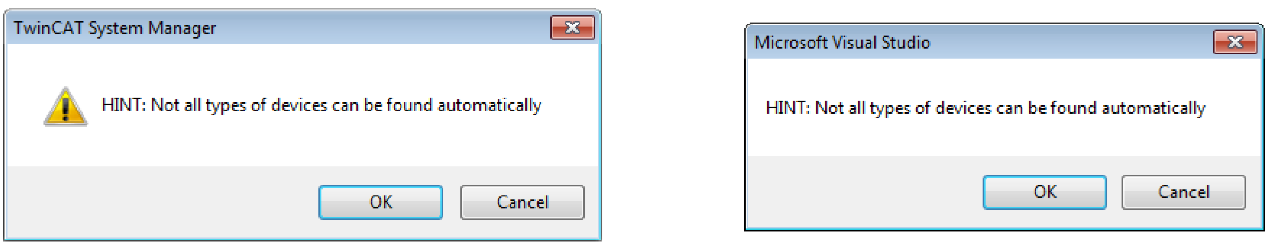


Fig. 46: Note for automatic device scan (left: TwinCAT 2; right: TwinCAT 3)

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” .

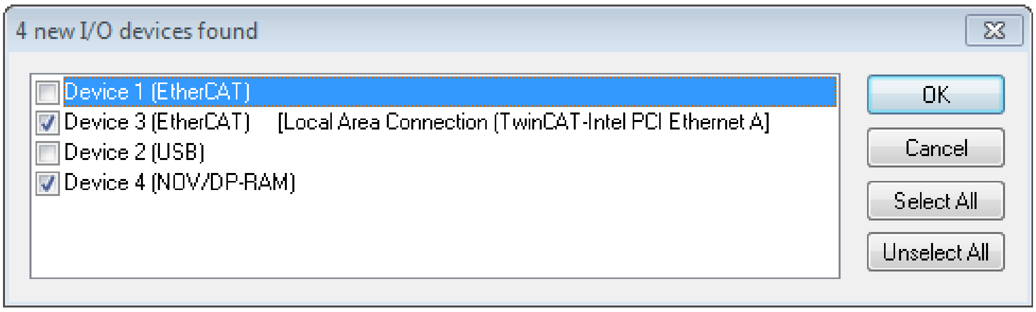


Fig. 47: Detected Ethernet devices

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 [▸ 40].

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.

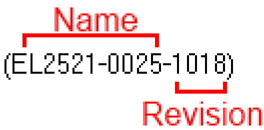


Fig. 48: Example default state

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 [▸ 61] 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:

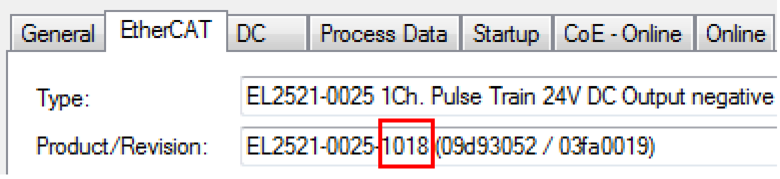


Fig. 49: Installing EthetCAT terminal with revision -1018

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 [▸ 61] 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:

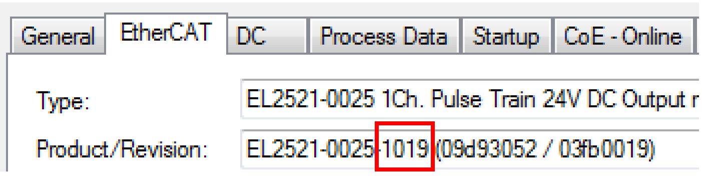


Fig. 50: Detection of EtherCAT terminal with revision -1019

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

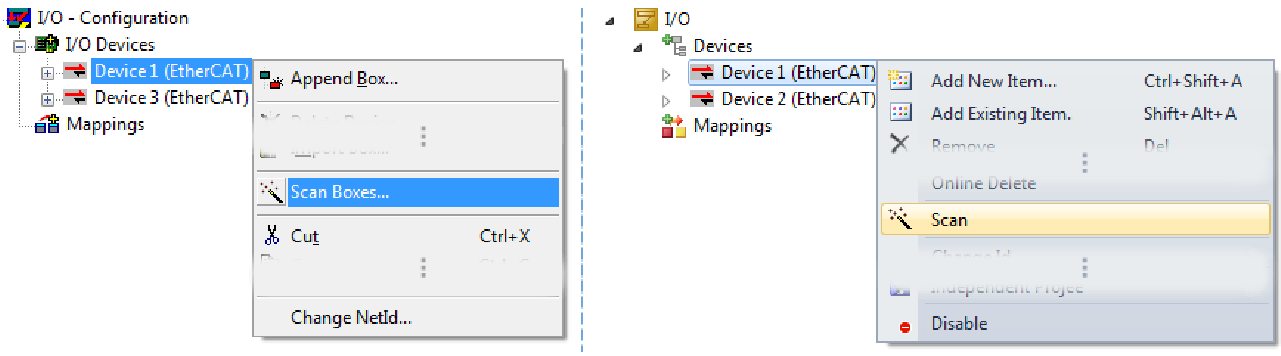


Fig. 52: Manual scanning for devices on a specified EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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. 53: *Scan progressexemplary by TwinCAT 2*

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

Fig. 54: 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).

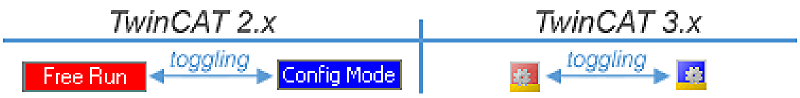


Fig. 55: Displaying of “Free Run” and “Config Mode” toggling right below in the status bar

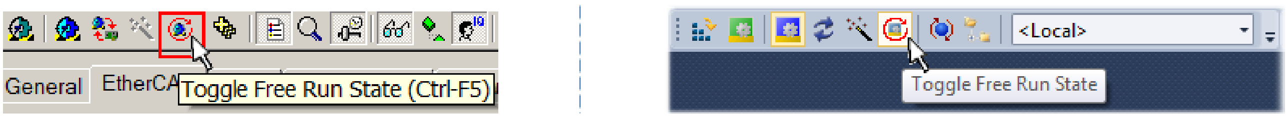


Fig. 56: TwinCAT can also be switched to this state by using a button (left: TwinCAT 2; right: TwinCAT 3)

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

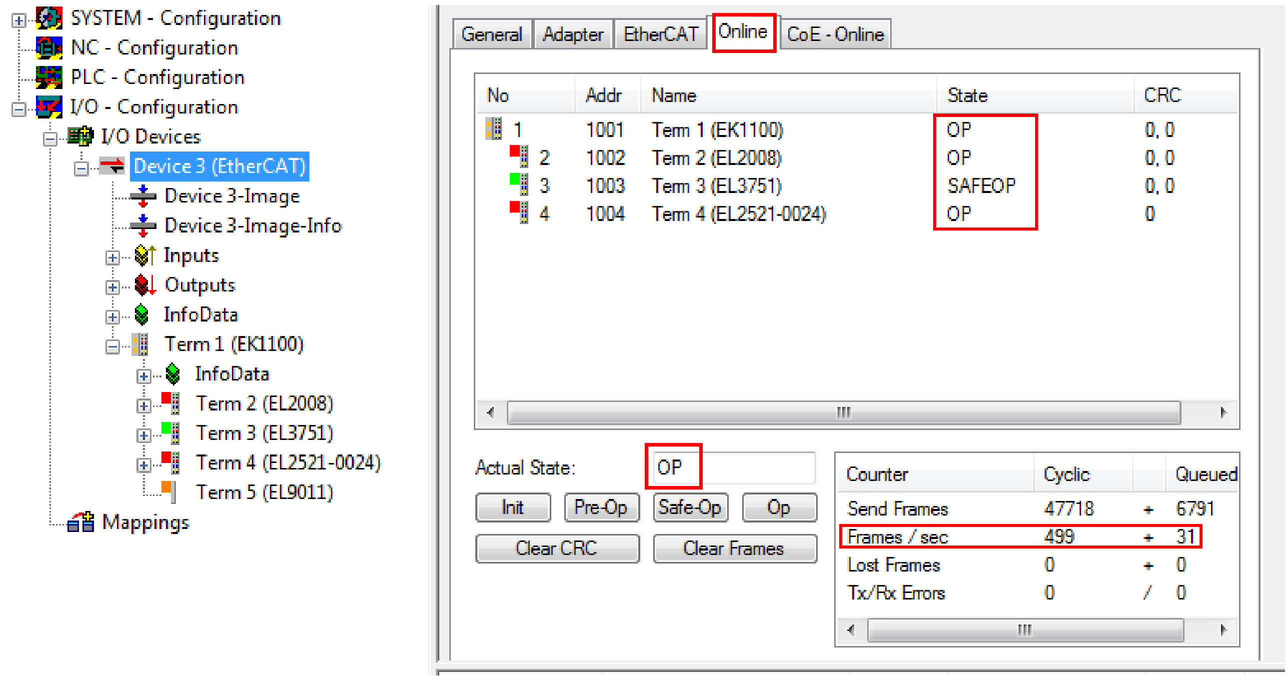


Fig. 57: Online display example

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 [▶ 51].

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.

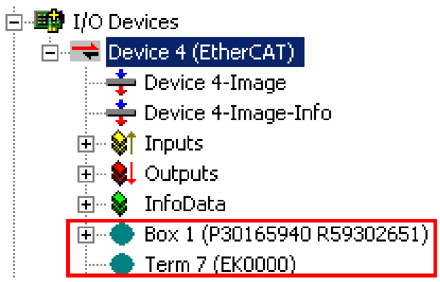


Fig. 58: Faulty identification

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. 59: 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.

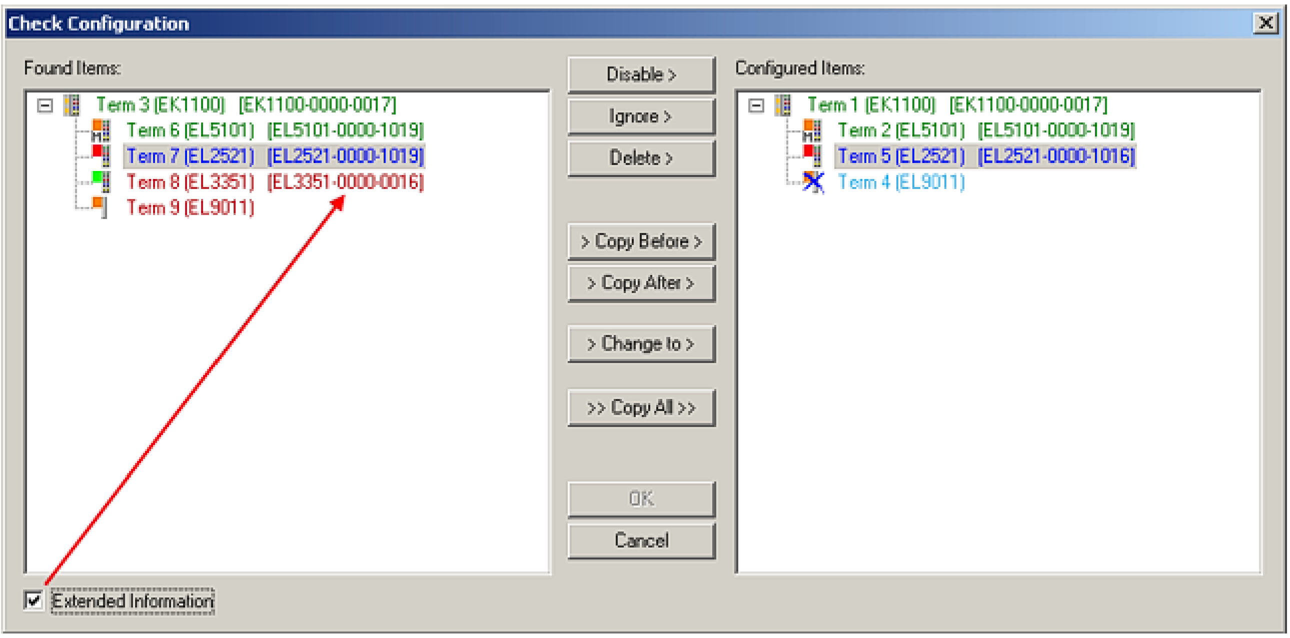


Fig. 60: Correction dialog

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.

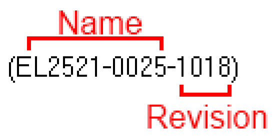


Fig. 61: Name/revision of the terminal

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, ...

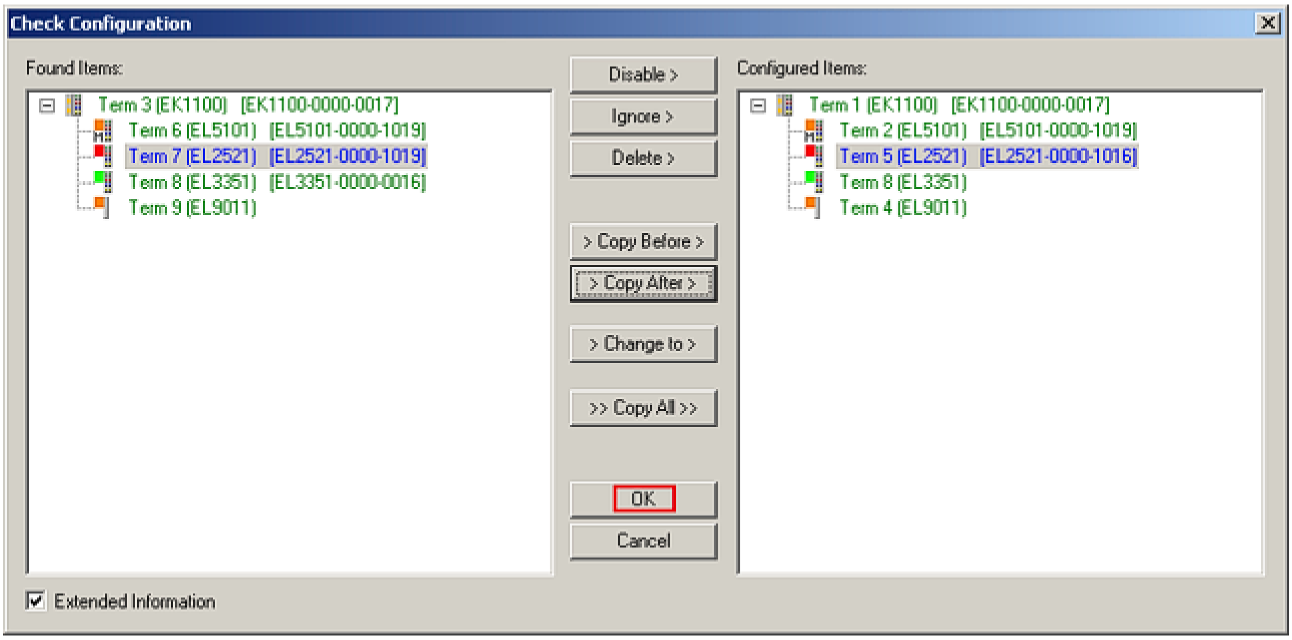


Fig. 62: Correction dialog with modifications

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.

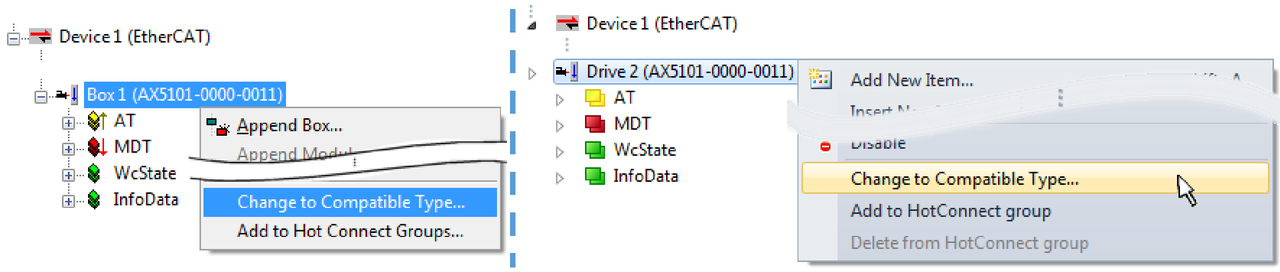


Fig. 63: Dialog “Change to Compatible Type…” (left: TwinCAT 2; right: TwinCAT 3)

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

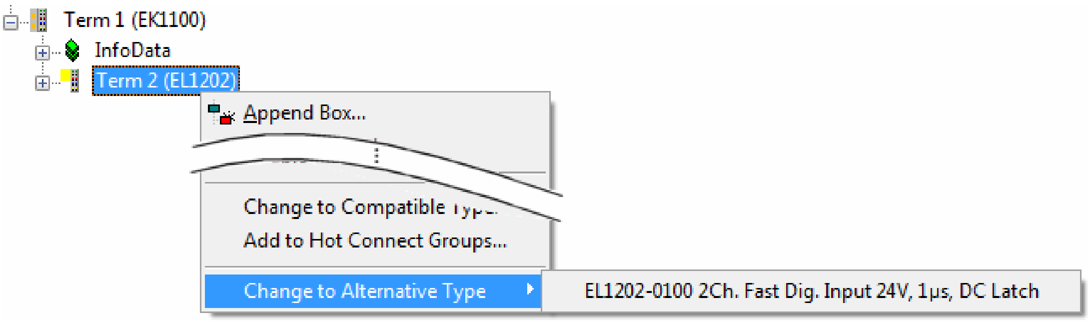


Fig. 64: TwinCAT 2 Dialog Change to Alternative Type

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

5.1.7 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).

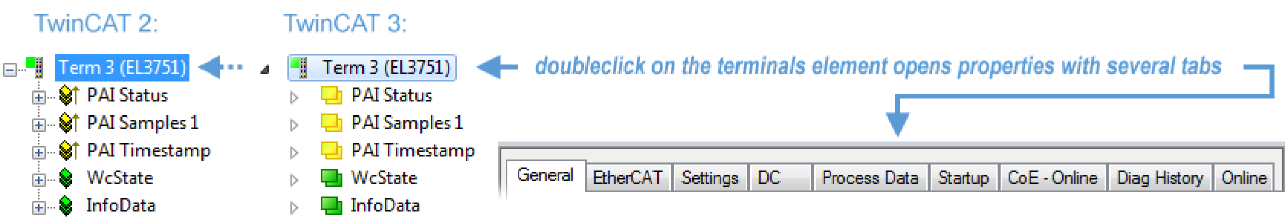


Fig. 65: Branch element as terminal EL3751

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

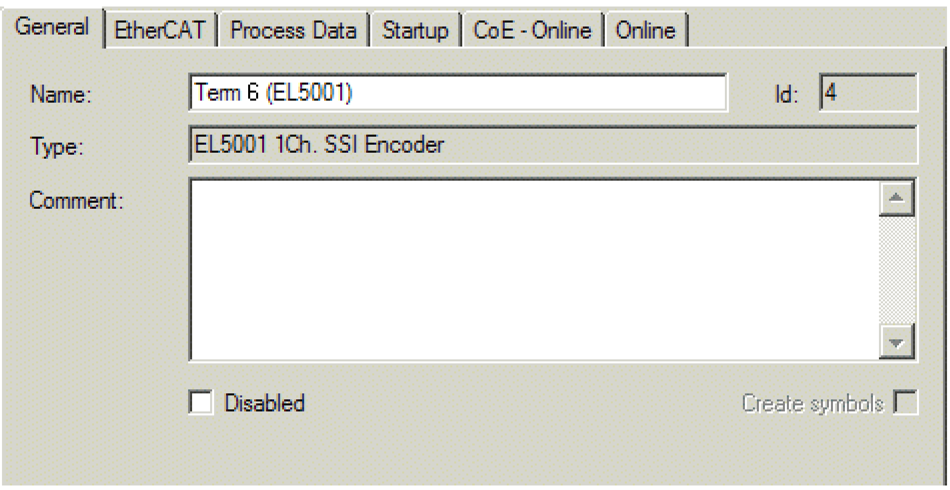


Fig. 66: “General” tab

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

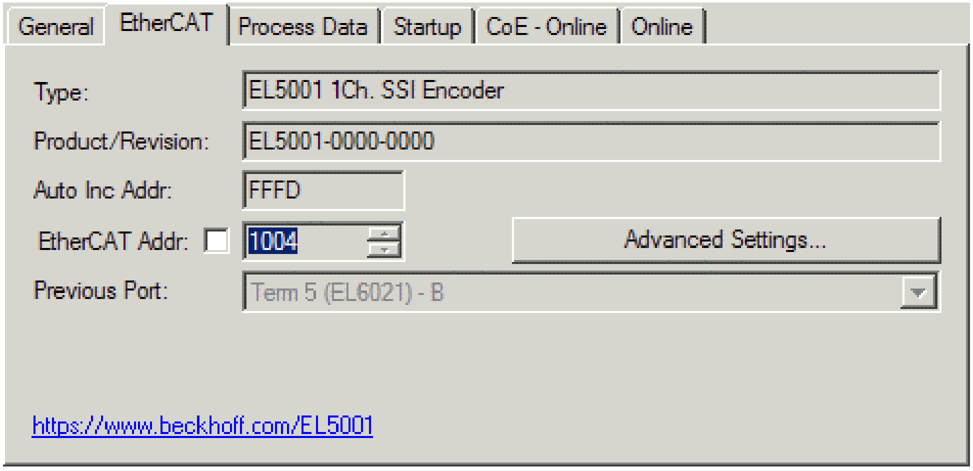


Fig. 67: "EtherCAT" tab

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.

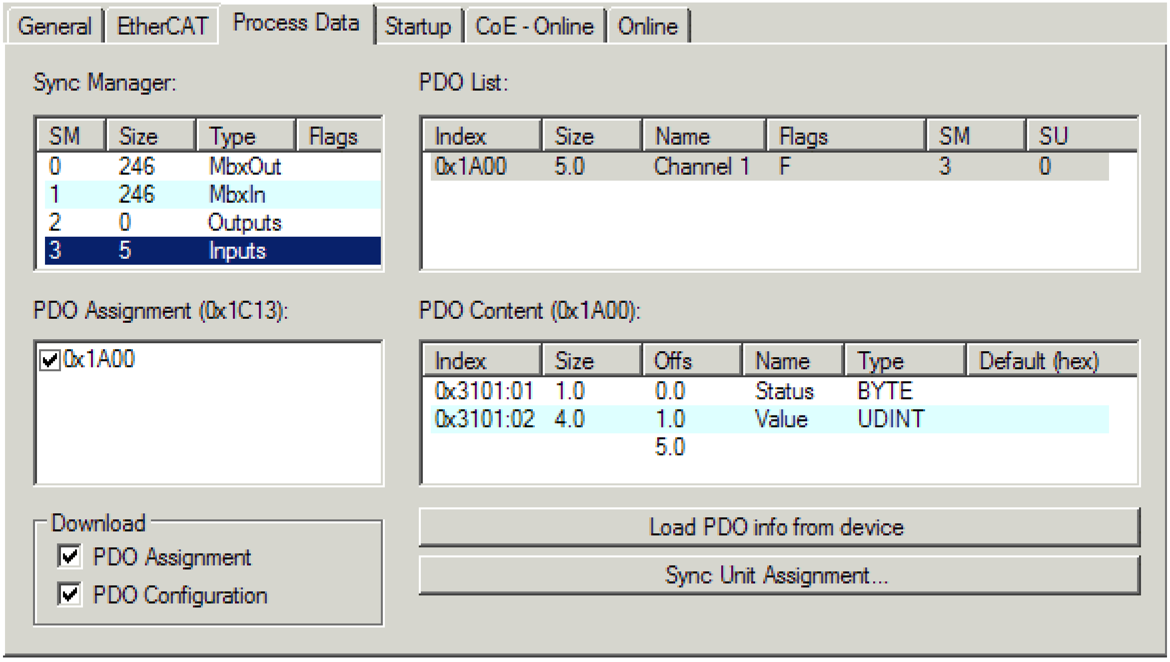


Fig. 68: “Process Data” tab

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”).

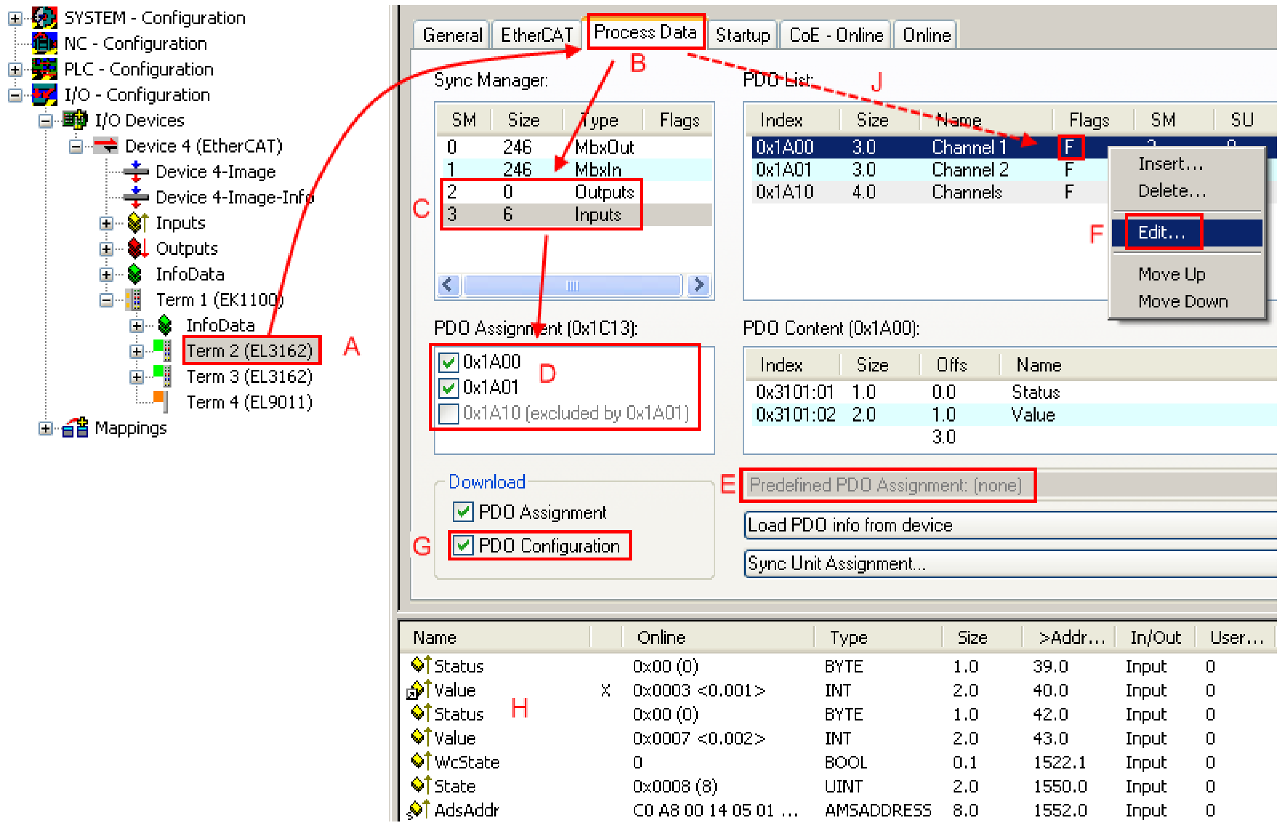


Fig. 69: Configuring the process data

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 [▸ 72] 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.

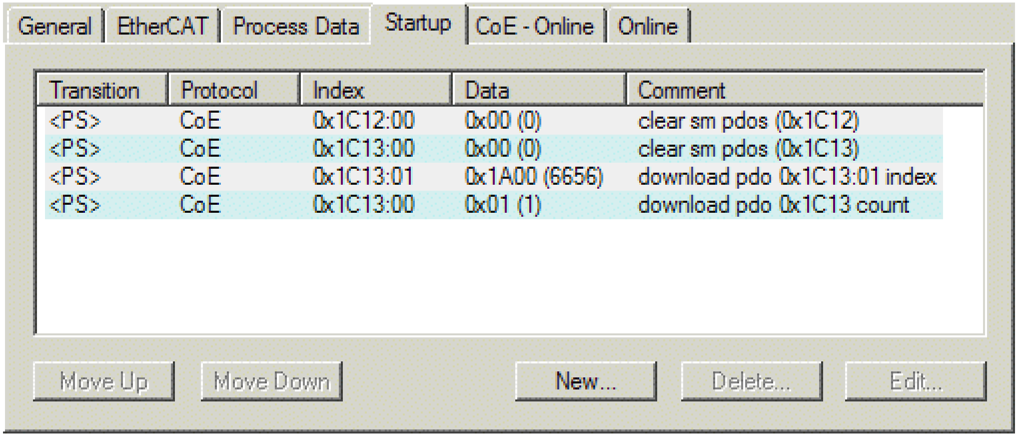


Fig. 70: “Startup” tab

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.

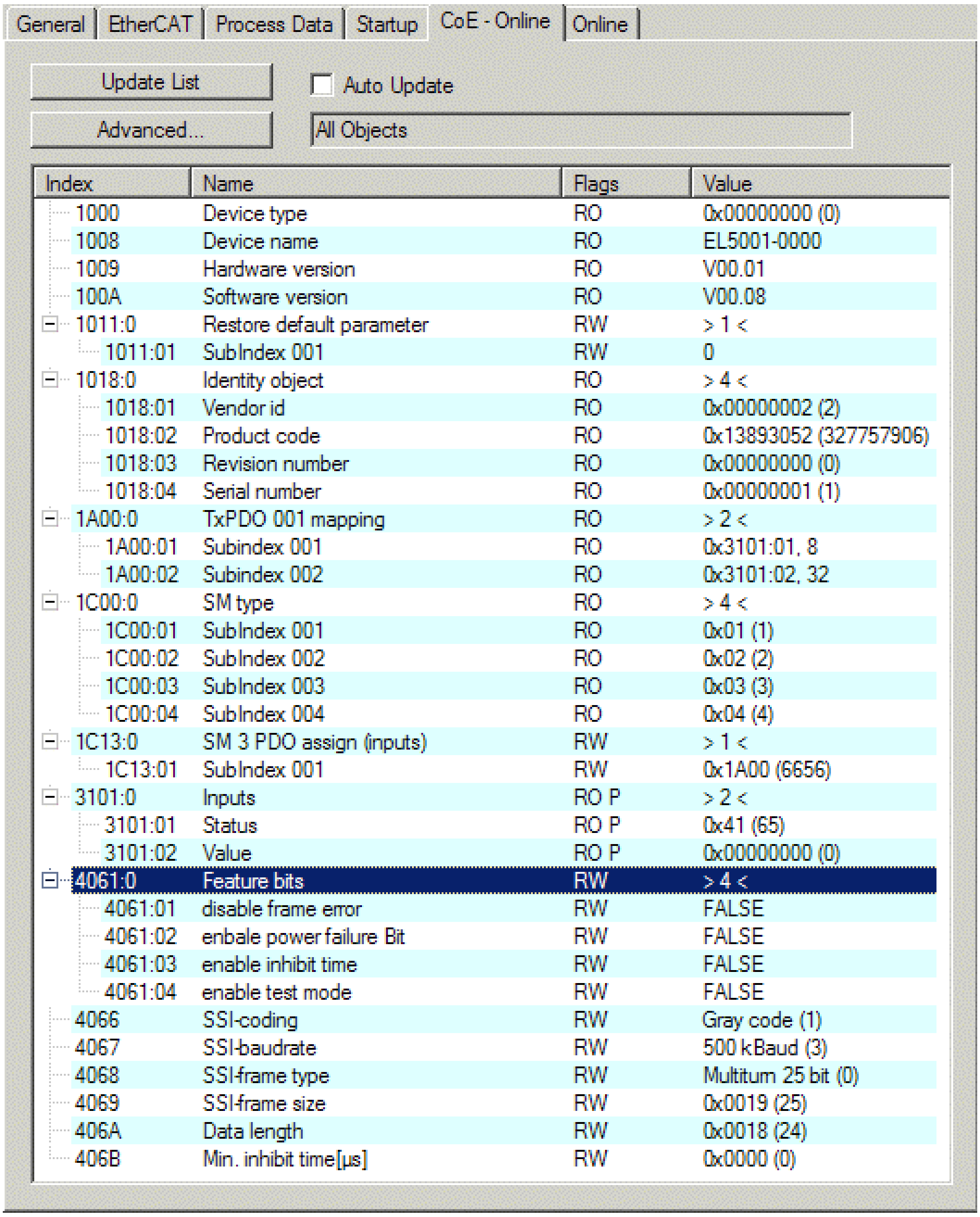


Fig. 71: “CoE - Online” tab

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.

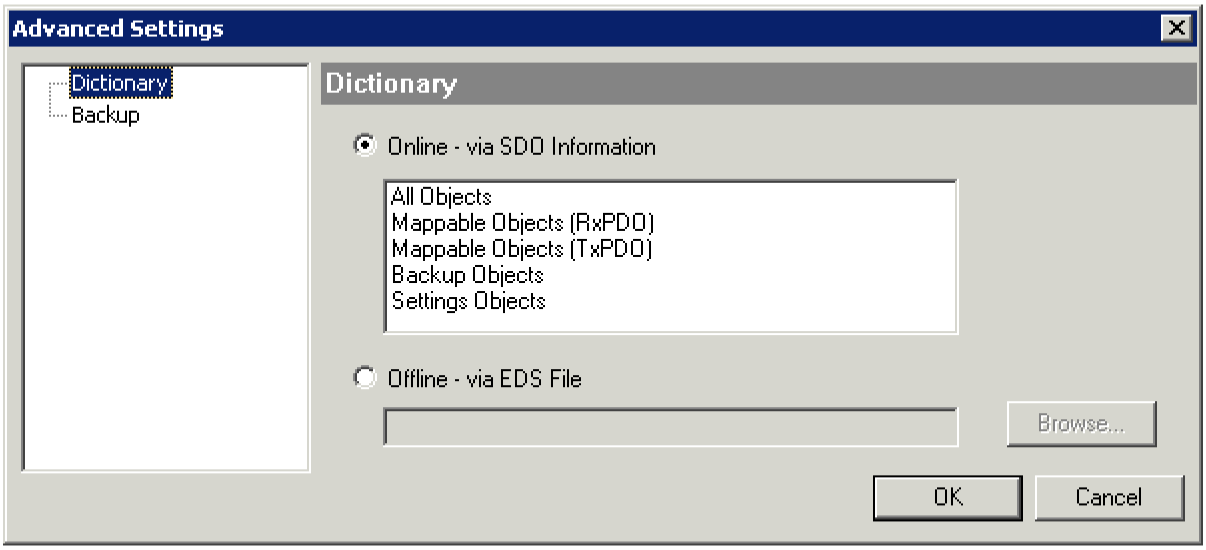


Fig. 72: Dialog “Advanced settings”

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

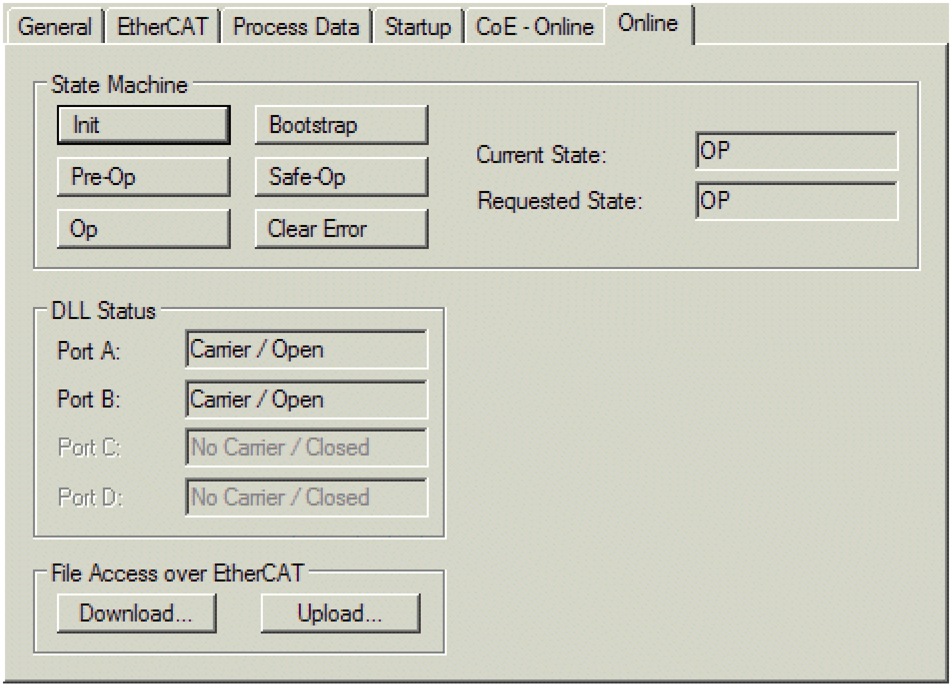


Fig. 73: “Online” tab

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)

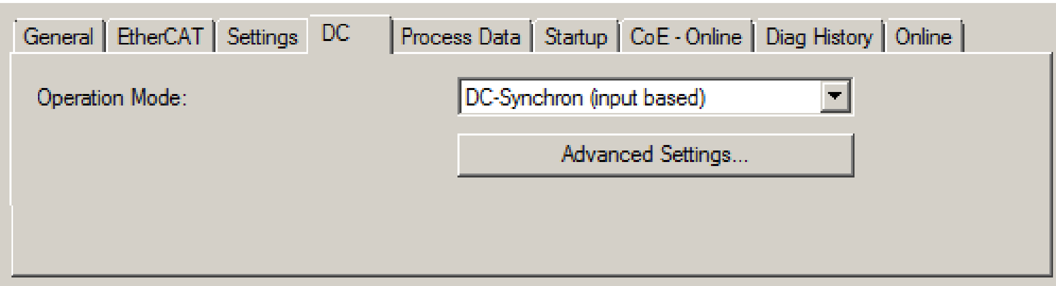


Fig. 74: “DC” tab (Distributed Clocks)

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

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

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

5.1.7.1 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 [▸ 70]),

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 [▶ 67] tab.

PDO Configuration

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

5.1.8 Import/Export of EtherCAT devices with SCI and XTI

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

5.1.8.1 Basic principles

An EtherCAT slave is basically parameterized through the following elements:

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

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

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

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

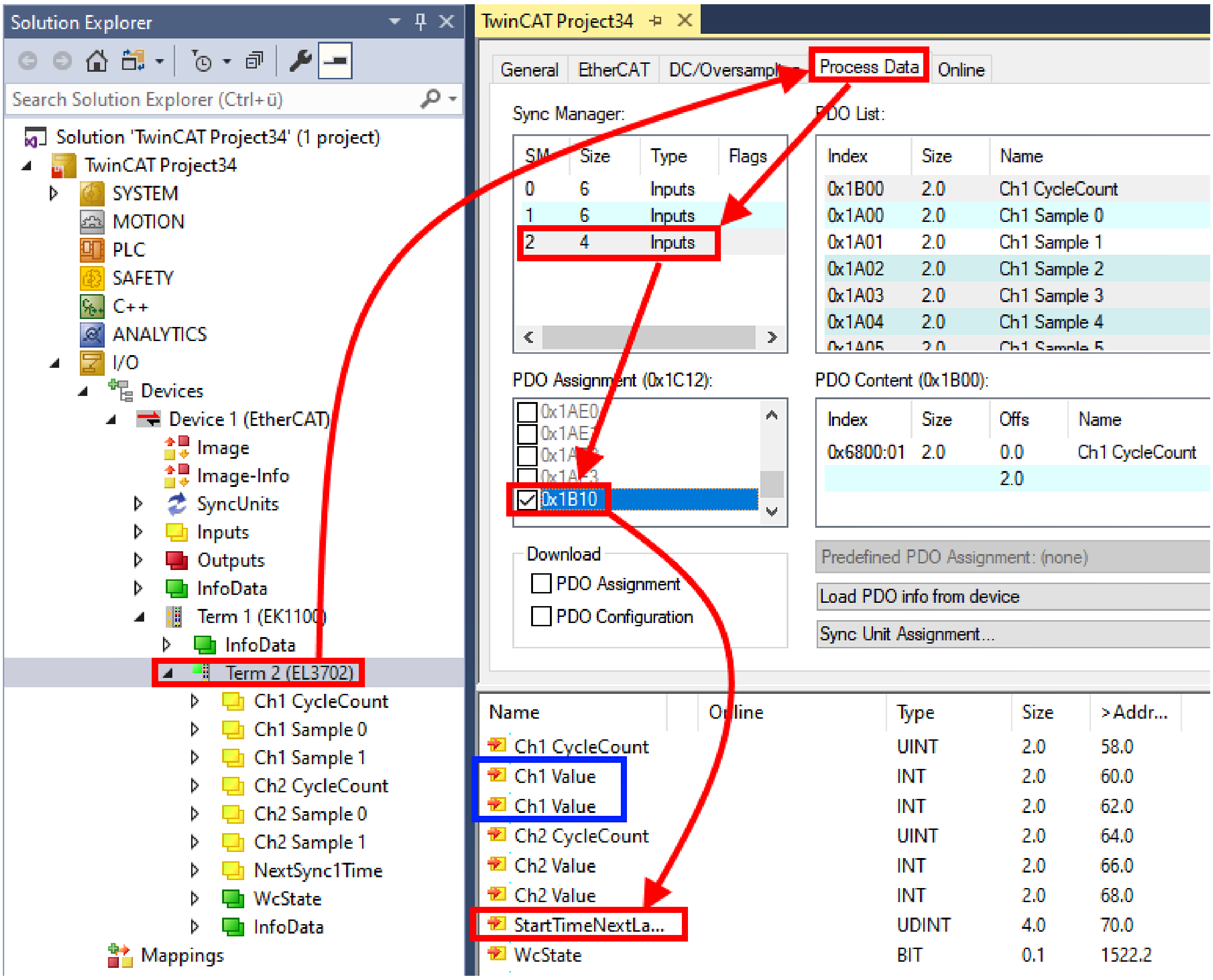
It may become necessary to reuse the modified slave in other projects in this way, without having to make equivalent configuration changes to the slave again. To accomplish this, proceed as follows:

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

TwinCAT offers two methods for this purpose:

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

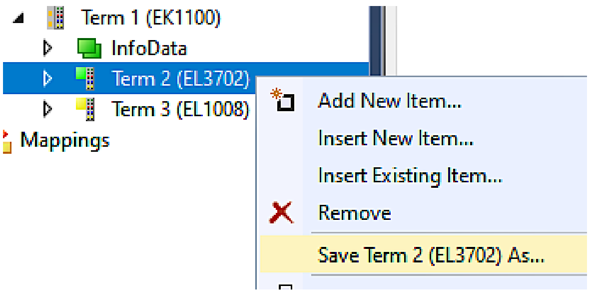
An example is provided below for illustration purposes: an EL3702 terminal with standard setting is switched to 2-fold oversampling (blue) and the optional PDO "StartTimeNextLatch" is added (red):



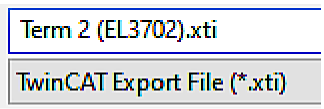
The two methods for exporting and importing the modified terminal referred to above are demonstrated below.

5.1.8.2 Procedure within TwinCAT with xti files

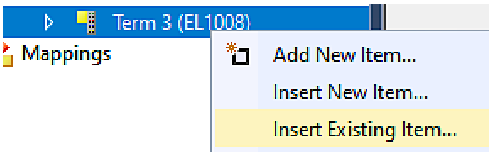
Each IO device can be exported/saved individually:



The xti file can be stored:



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



5.1.8.3 Procedure within and outside TwinCAT with sci file

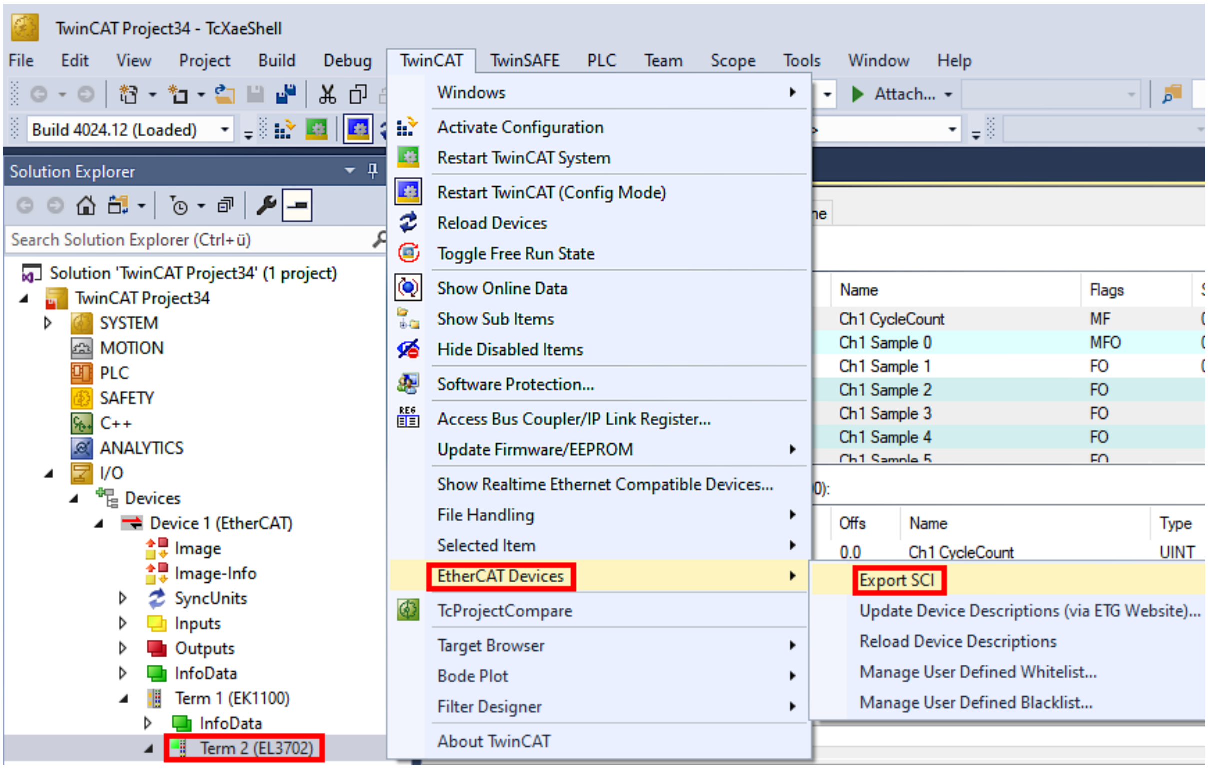
Note regarding availability (2021/01)

The SCI method is available from TwinCAT 3.1 build 4024.14.

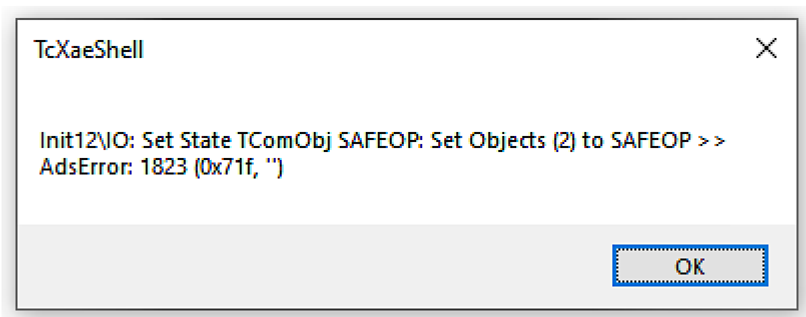
The Slave Configuration Information (SCI) describes a specific complete configuration for an EtherCAT slave (terminal, box, drive...) based on the setting options of the device description file (ESI, EtherCAT Slave Information). That is, it includes PDO, CoE, synchronization.

Export:

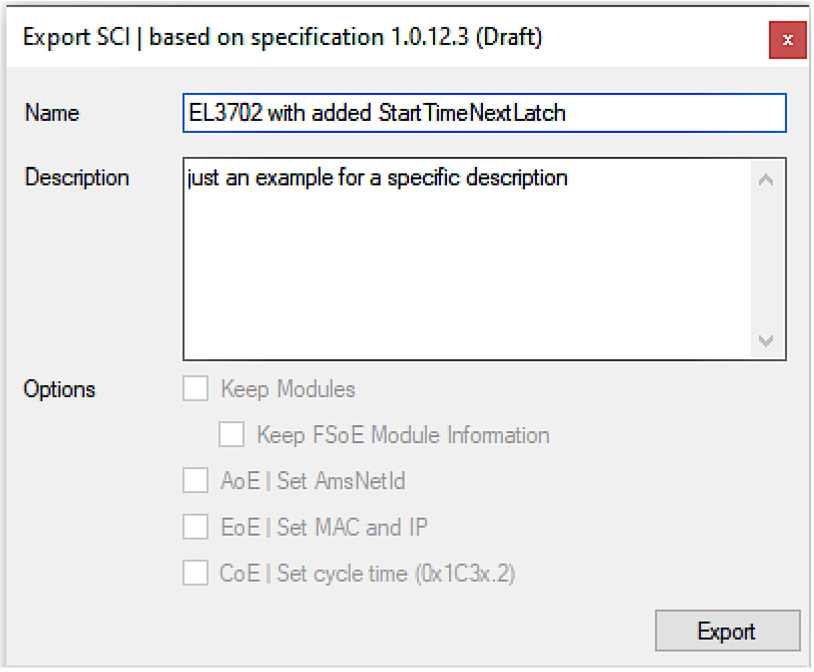
- select a single device via the menu (multiple selection is also possible):
  TwinCAT → EtherCAT Devices → Export SCI.



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



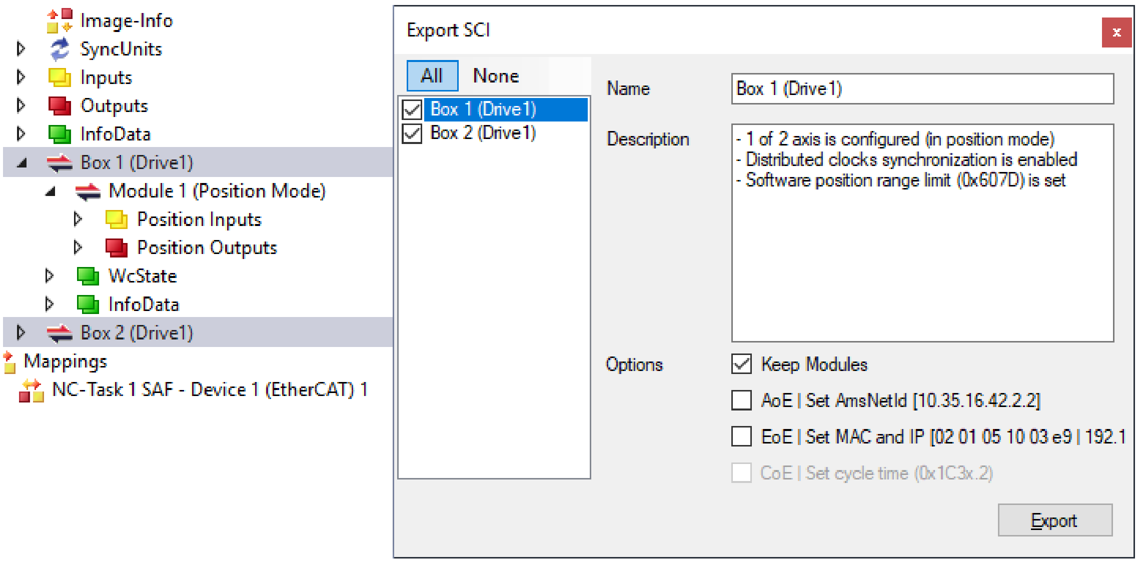
- A description may also be provided:



- Explanation of the dialog box:

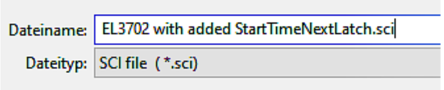
Name		Name of the SCI, assigned by the user.
Description		Description of the slave configuration for the use case, assigned by the user.
Options	Keep modules	If a slave supports modules/slots, the user can decide whether these are to be exported or whether the module and device data are to be combined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

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

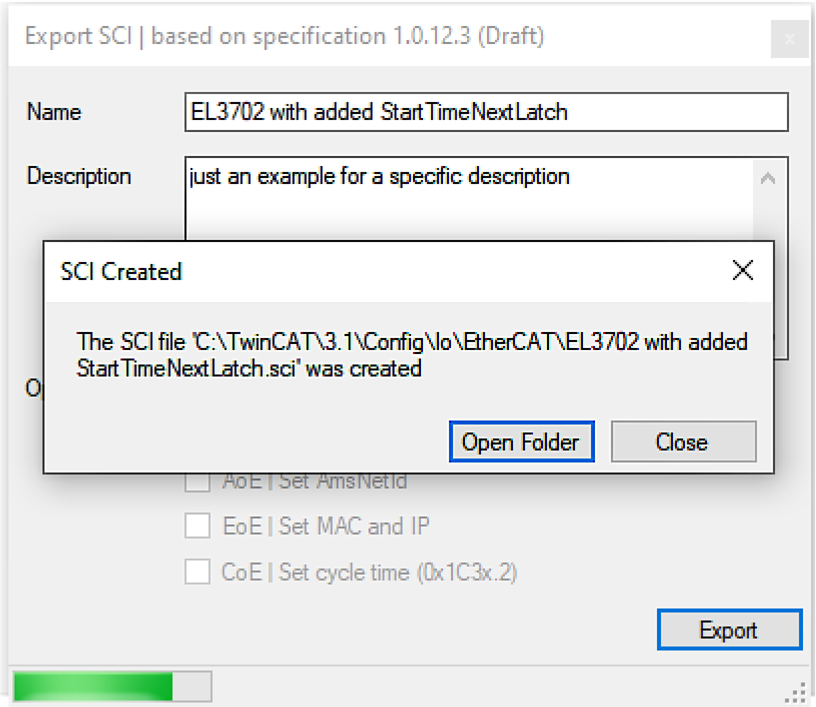


- Selection of the slaves to be exported:
  - All:
    All slaves are selected for export.

  - None:
    All slaves are deselected.
- The sci file can be saved locally:



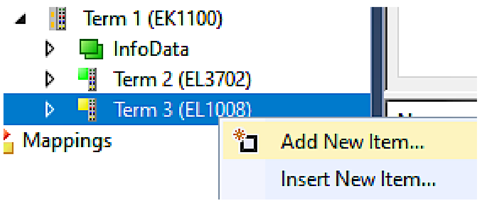
- The export takes place:



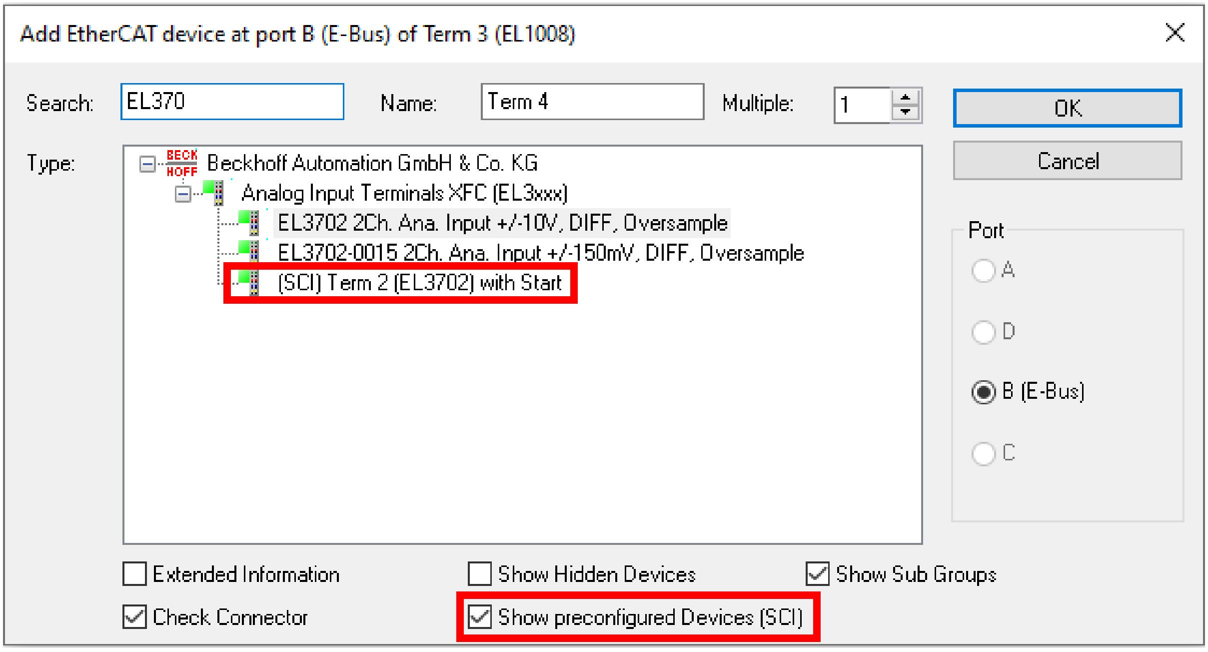
Import

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

- Open the selection dialog:

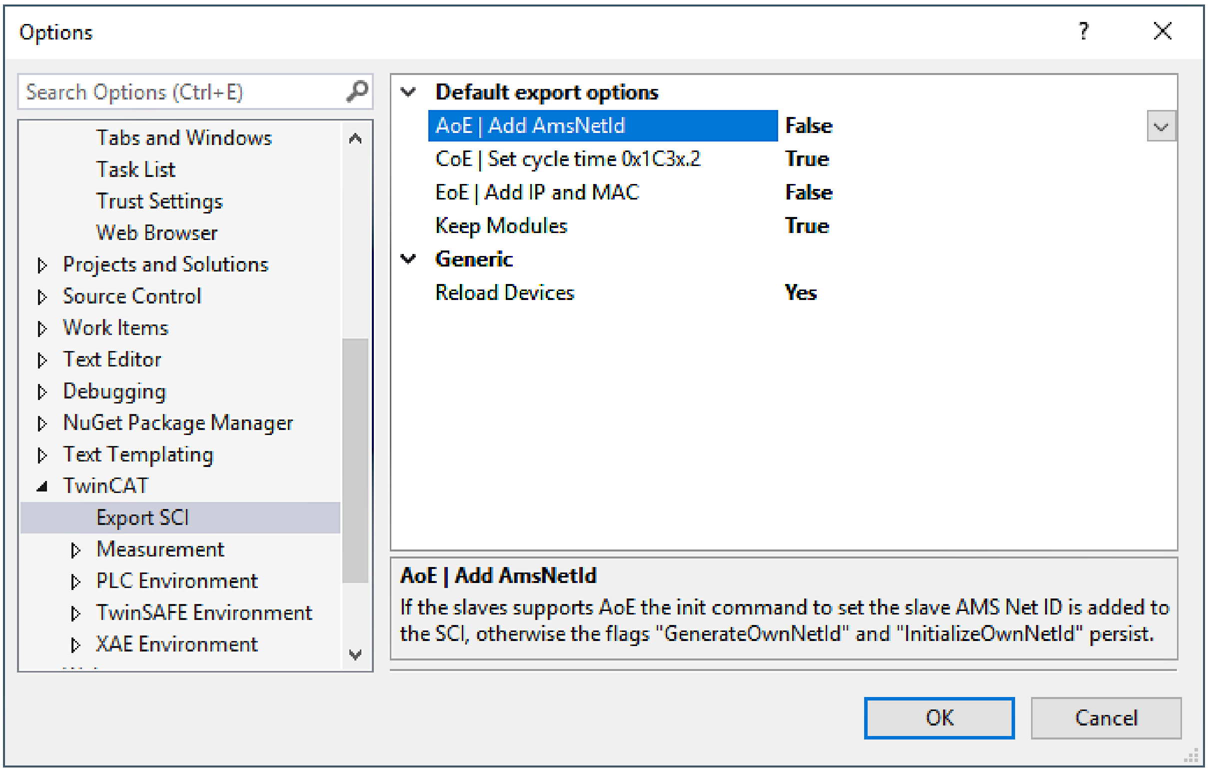


- Display SCI devices and select and insert the desired device:



Additional Notes

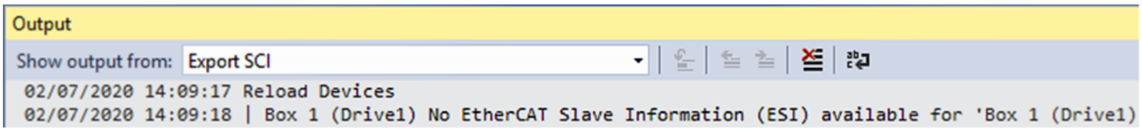
- Settings for the SCI function can be made via the general Options dialog (Tools → Options → TwinCAT → Export SCI):



Explanation of the settings:

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

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



5.2 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.

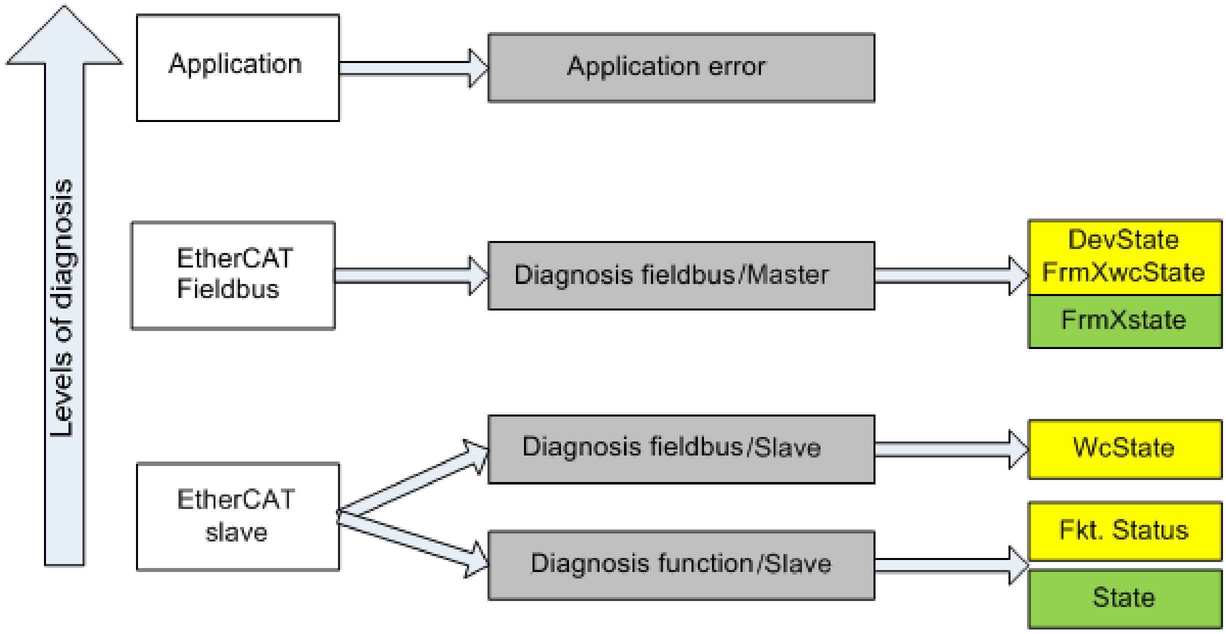


Fig. 75: Selection of the diagnostic information of an EtherCAT Slave

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.

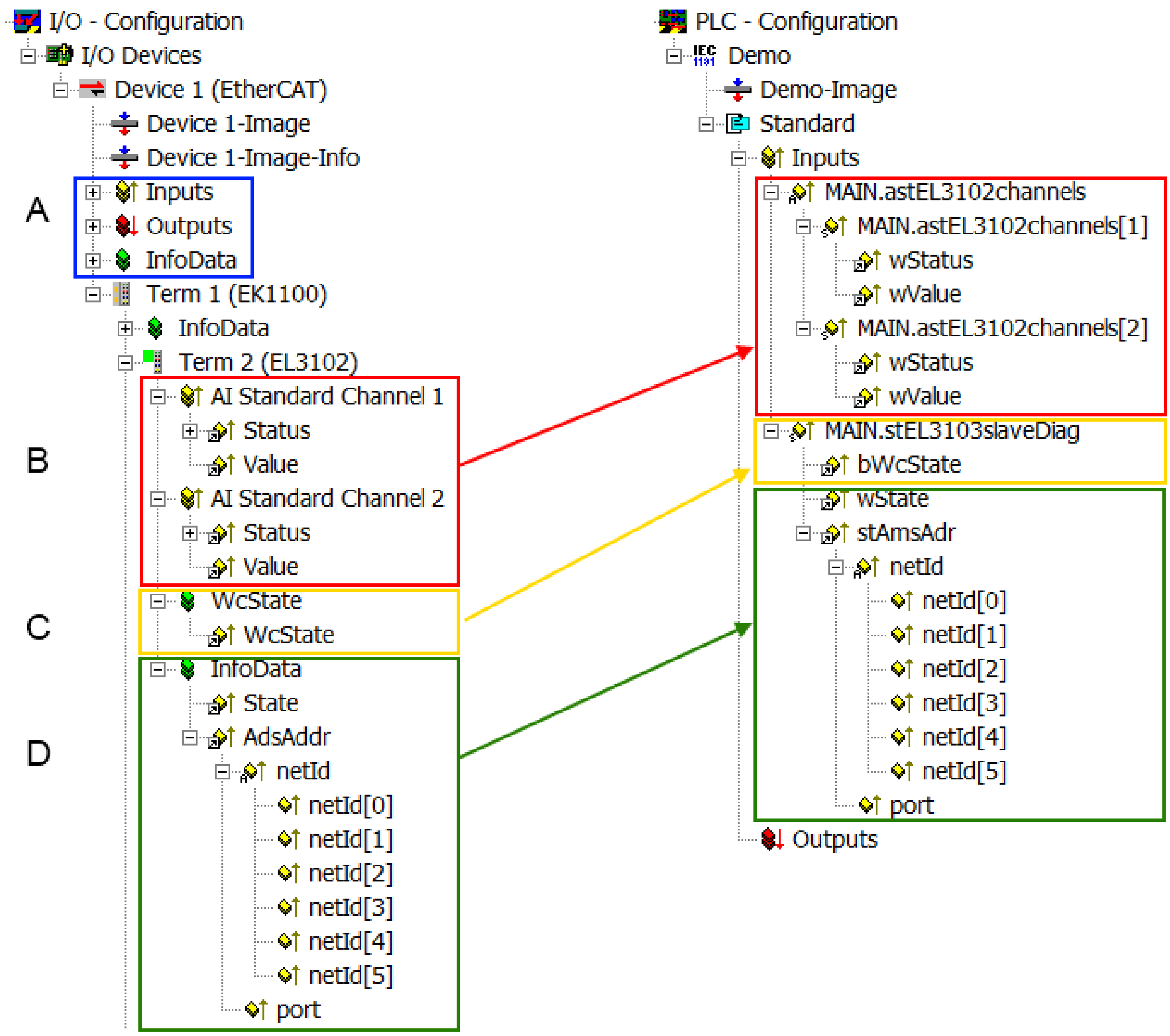


Fig. 76: Basic EtherCAT Slave Diagnosis in the PLC

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*:

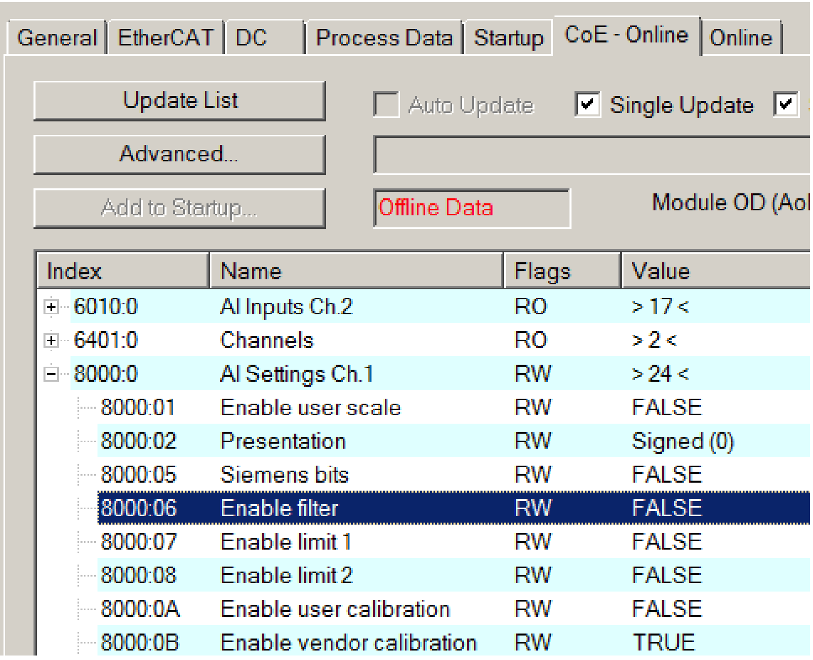


Fig. 77: EL3102, CoE directory

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.

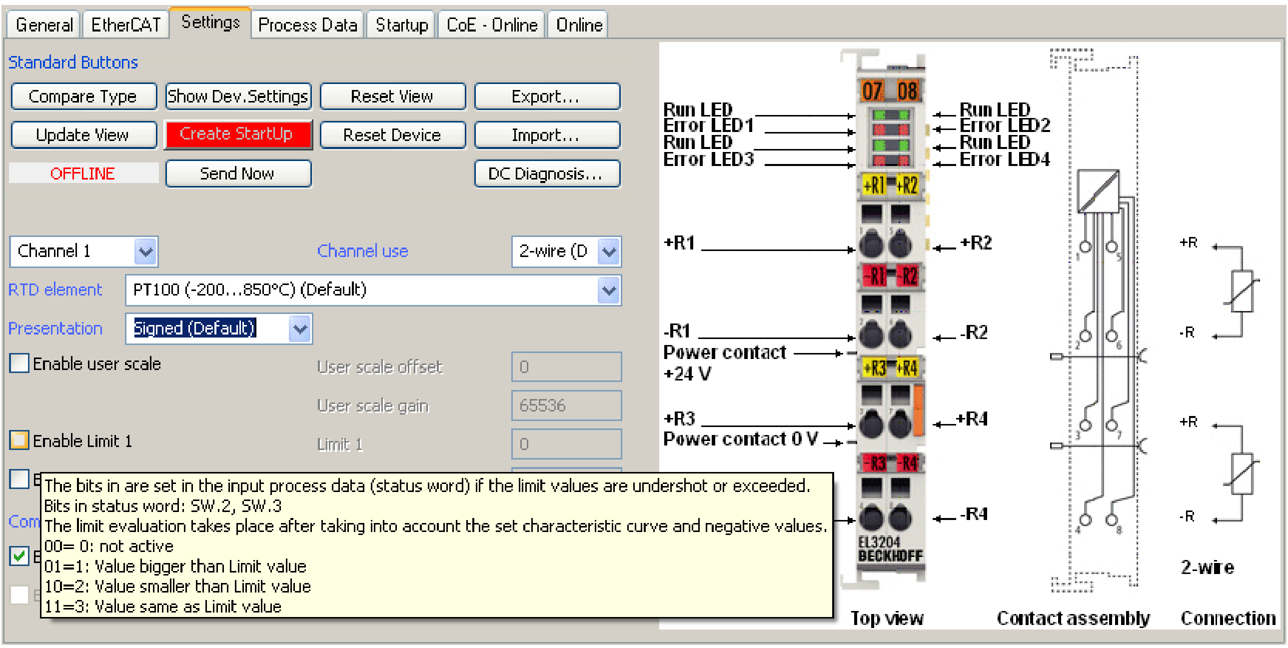


Fig. 78: Example of commissioning aid for a EL3204

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 [▶ 19]" 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.

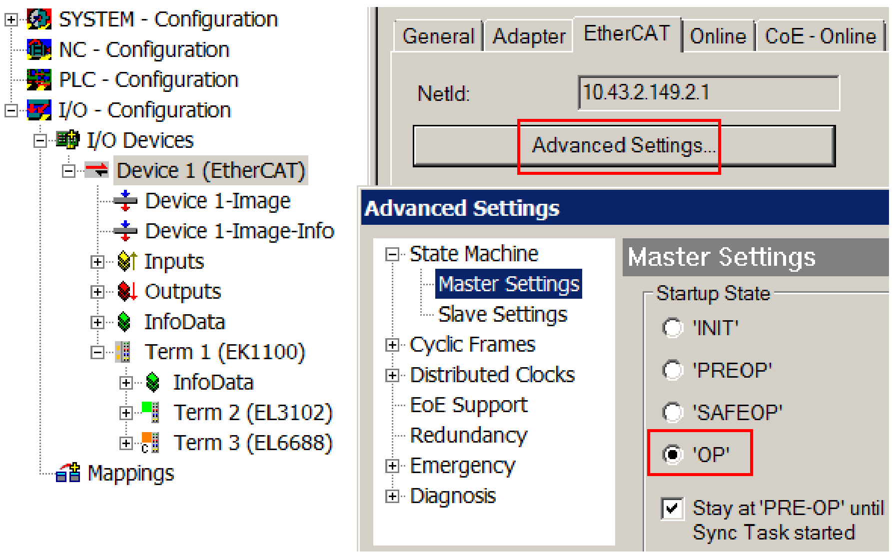


Fig. 79: Default behaviour of the System Manager

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

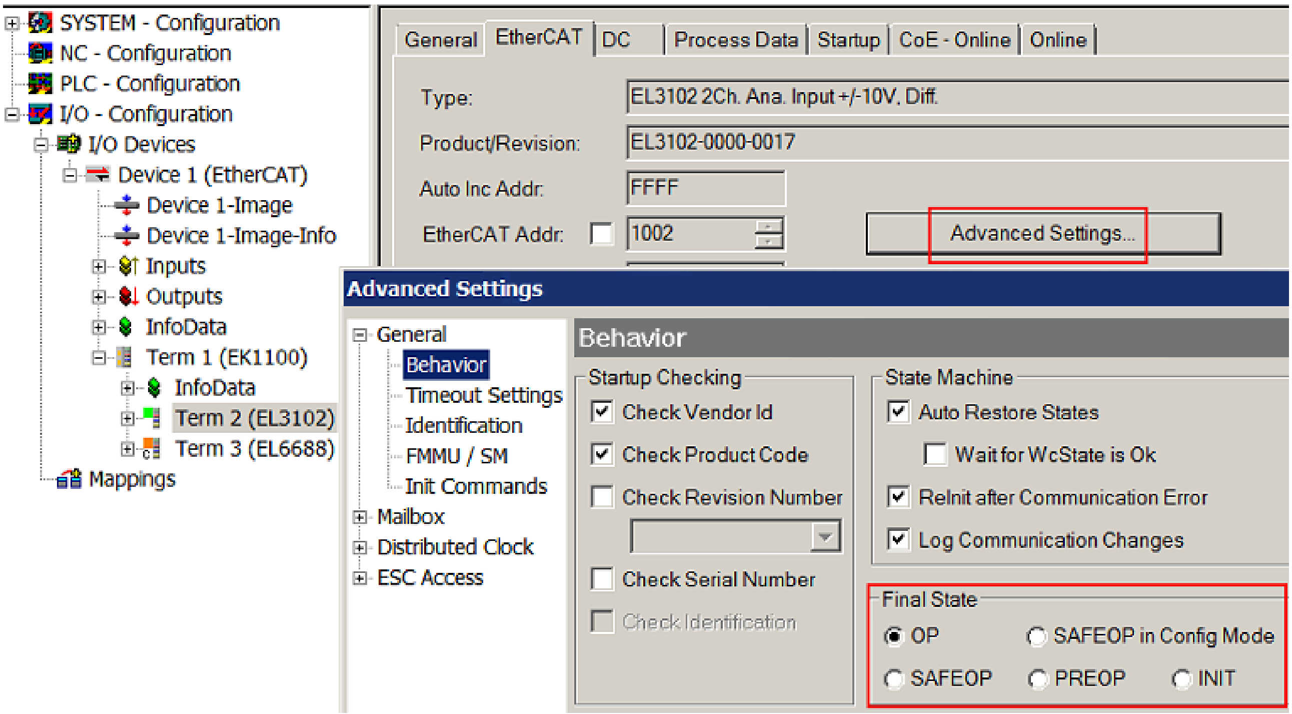


Fig. 80: Default target state in the Slave

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.

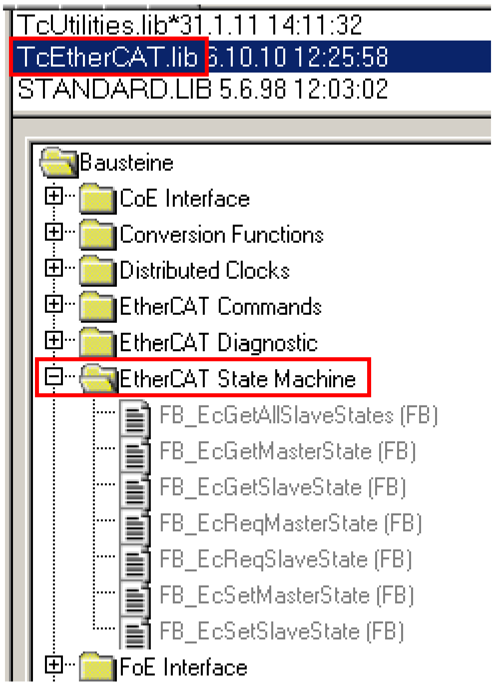
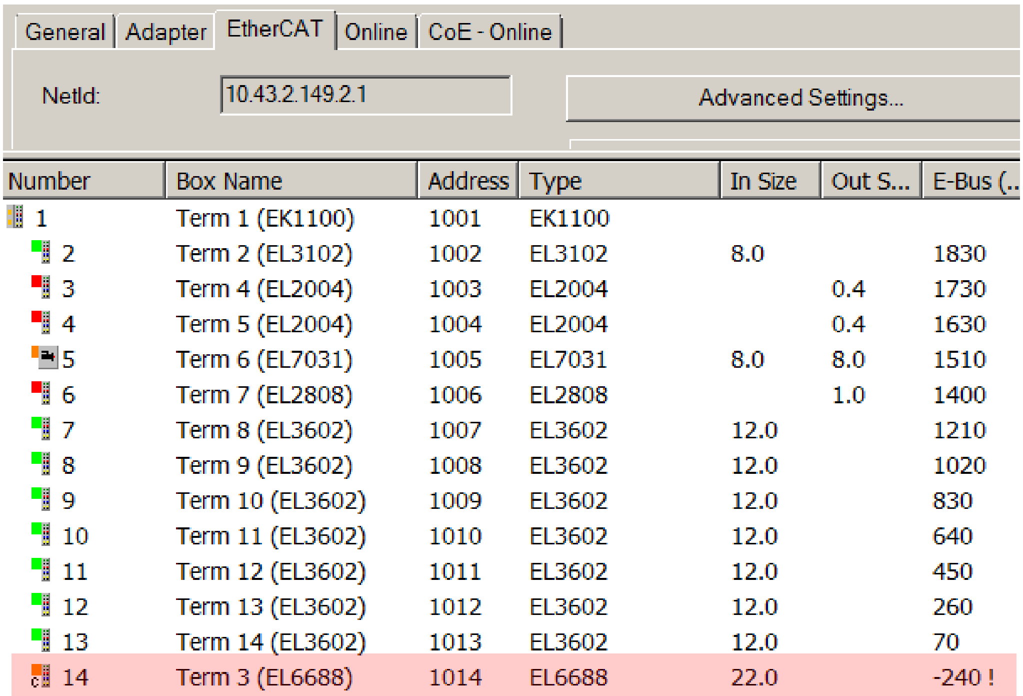


Fig. 81: PLC function blocks

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. 82: 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:

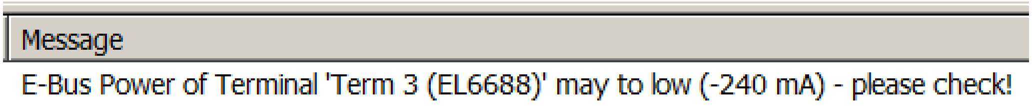


Fig. 83: Warning message for exceeding E-Bus current

NOTICE

Caution! Malfunction possible!

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

5.3 Basic function principles

5.3.1 IEC 61851 - Low-level communication

The standard is divided into several parts, each of which focuses on a different aspect of charging electric vehicles:

1. IEC 61851-1: This part contains general requirements for conductive charging systems for electric vehicles and covers the basic topics of safety, functionality and interoperability.
2. IEC 61851-21: This part deals with the requirements for electric vehicles for conductive connection to an AC/DC supply. It is further divided into two parts:
    - IEC 61851-21-1: Covers on-board measuring devices for electric vehicles (EV) for measuring the electricity transmitted from the grid to the vehicle.
    - IEC 61851-21-2: Focuses on the requirements for the vehicle's internal power supply circuit for connection to the charging station.
3. IEC 61851-22: This part specifies the requirements for the charging station for electric vehicles and outlines the testing and safety requirements for the equipment.
4. IEC 61851-23: This part defines the digital communication between the electric vehicle and the electric vehicle supply device for DC charging.
5. IEC 61851-24: This part deals with digital communication between a DC EV charging station and an electric vehicle to control DC charging.

The IEC 61851 standard is of central importance in ensuring that charging stations and vehicles for electric vehicles are compatible and safe across manufacturers and countries. It aims to promote the widespread introduction of electric vehicles by providing a standardized framework for the charging infrastructure. For this reason, it has fixed four charging modes (see table Charging modes)

Charging modes table

Charging mode	Description
Charging mode 1	In this mode, the EV is connected directly to a household socket.
Charging mode 2	A special charging cable equipped with an internal cable control and protection device (IC-CPD) is used for charging mode 2.
Charging mode 3	In this mode, a special EVSE is used together with the EV on-board charger. The alternating current from the charging station is forwarded to the on-board circuit to charge the battery. To ensure public safety, several control and protection functions are used, including checking the protective earth and the connection between the EVSE and EV.
Charging mode 4	This is the only charging mode that includes an external charger with a DC output. The direct current is supplied directly to the battery and the on-board charger is bypassed. This mode can supply several hundred volts DC with a maximum current of 400 A. The high performance in this mode requires a higher level of communication and more stringent security features.

As far as the primary communication between the EVSE and the EV is concerned, IEC 61851-1 specifies state-based communication via the CP connection contact. The PP connection provides information on the capacity of the line between EVSE, as well as information on whether a car or cable is plugged in.

This state-based communication on the CP connection provides for six states at different voltage levels that represent the charging readiness of the EVSE and EV. The states are listed in the IEC 61851 communication states table below.

Table IEC 61851 Communication states

State	CP voltage level	Description
State A	+ 12 V	EV not connected
State B	+ 9 V	EV connected; EV not ready
State C	+6 V	EV connected; EV ready for charging; without ventilation
State D	+3 V	EV connected; EV ready for charging; with ventilation
State E	+0 V	Error state (e.g. lack of energy from the grid)
State F	-12 V	Forced by the charging station, e.g. due to maintenance work

In addition to the various voltage levels, the ESVE supplies a PWM signal with a specific duty cycle. An overview of the different duty cycle levels specified by IEC 61851 is shown below.

Table IEC 61851 Duty cycle

Duty Cycle D in %	Maximum permissible current	Description
D < 3 %	0 A	Invalid
3 % ≤ D ≤ 7 %	HLC	A nominal duty cycle of 5% indicates high-level communication in accordance with ISO 15118 / DIN 70121. Power consumption in this duty cycle range without this communication is not permitted.
7 % ≤ D < 8 %	0 A	Invalid
8 % ≤ D < 10 %	6 A	Some vehicles use this duty cycle to use a low current of D x 0.6 A, i.e. 4.8 A to 6 A.
10 % ≤ D ≤ 85 %	D x 0.6 A	6 A to 51 A
85 % < D ≤ 96 %	(D - 64) x 2.5 A	51 A to 80 A
96 % < D ≤ 97 %	80 A	Maximum value
97 % < D ≤ 100 %	0 A	Invalid

After the EL6761 has recognized that the CP line is connected to the EV, the customer has the option of manipulating the PWM via 0x7001:16 Setpoint Current.

Setting the signal to "0" results in a PWM of 0% and charging is not possible. The value can only be set prior to plugging in to indicate that the EVSE is in maintenance mode. Any other valid value for this signal results in a PWM of 100 % if the EVSE is not connected.

If the terminal is connected to an EV, setting the signal to "1" results in a PWM of 100 %, while setting the signal to "2" HLC results in AC high-level communication (PWM of 5 %). Any value between "2" and "60" is ignored. Any value between "60" and "800" is calculated according to the table above and values above "800" are ignored.

The unit of this value is [A] with a factor of 0.1 of the values (from 60 to 800). If the signal is set to "0" during the charging process, the charging process is aborted. However, the charging process can be resumed at any time by setting 0x7001:16 Setpoint Current to a valid value without having to remove the plug.

Example: If the value "250" is sent via 0x7001:16 Setpoint Current, this means a maximum current of 25 A per phase and corresponds to a duty cycle of 41 % according to the calculation in the previous table. The duty cycle is then automatically calculated and set by the terminal.

To establish a charging connection between EVSE and EV, the figure below shows an exemplary communication process.

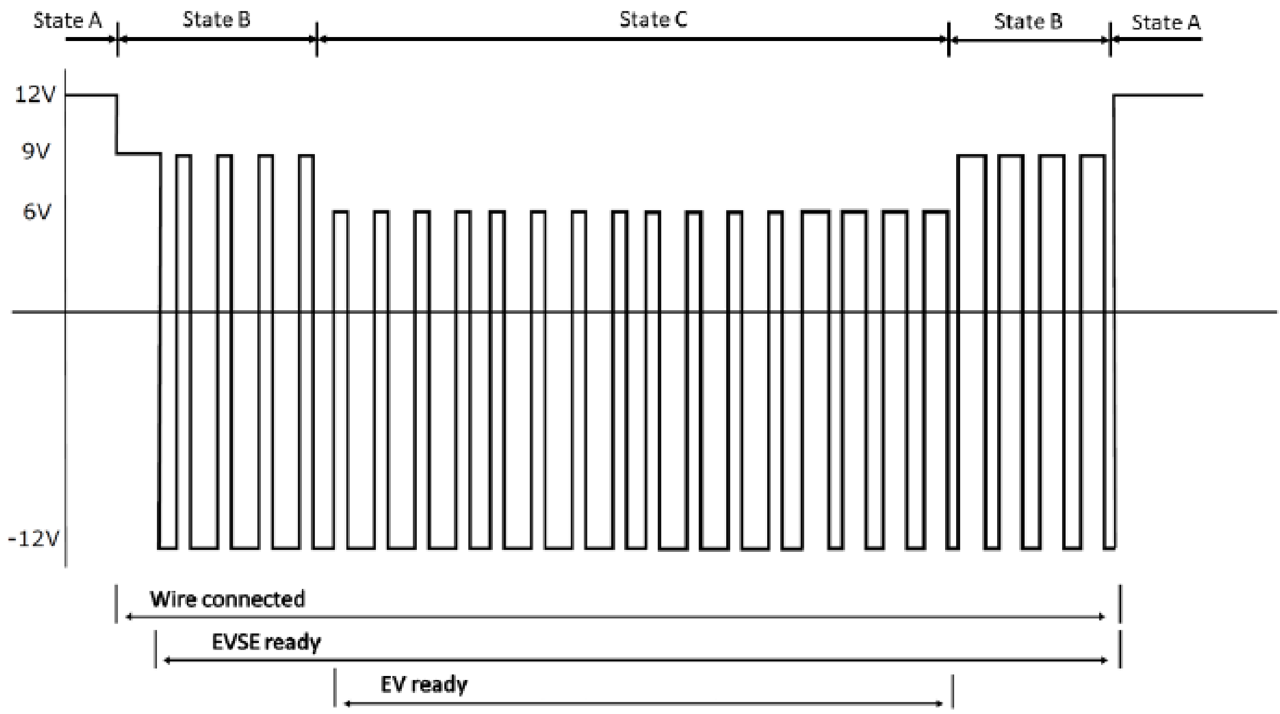


Fig. 84: IEC 61851 Communication example. The charging power can be adjusted by changing the duty cycle in state C (see table "IEC 61851 Communication states"

5.3.2 ISO 15118 - High-level communication

An extended communication scheme is defined in ISO 15118. It offers a flexible communication channel for transmitting more information between the vehicle and the charging station compared to IEC 61851 communication. It therefore enables advanced functions for charging stations, such as:

1. **Plug & Charge:** The vehicle can authenticate itself and start the charging process without the need for additional user authentication. This simplifies the user experience, as the vehicle handles the payment and billing processes automatically.
2. **Bi-directional charging**: The standard supports vehicle-to-grid (V2G) communication, which enables not only the charging of the electric vehicle, but also the discharging of the vehicle battery into the grid. This can help to balance demand on the electrical grid.
3. **Intelligent charging:** This includes the ability of the charging station and vehicle to exchange information about the state of charge, battery condition and grid requirements to optimize charging schedules, electricity rates and power distribution.
4. **Security:** ISO 15118 contains provisions for secure communication between the vehicle and the charging infrastructure to protect against unauthorized access and data misuse.

Communication stack ISO 15118

To enable extended functions compared to low-level communication, ISO 15118 uses communication via powerline communication (PLC) between EVSE and EV (Figure 2). Physically, the same CP connection is used as for IEC 61851 communication. However, to enable higher communication data rates, the PLC data stream is modulated to the PWM signal. As far as the higher communication layers are concerned, a Transmission Control Protocol / Internet Protocol (TCP/IP) stack is used for communication.

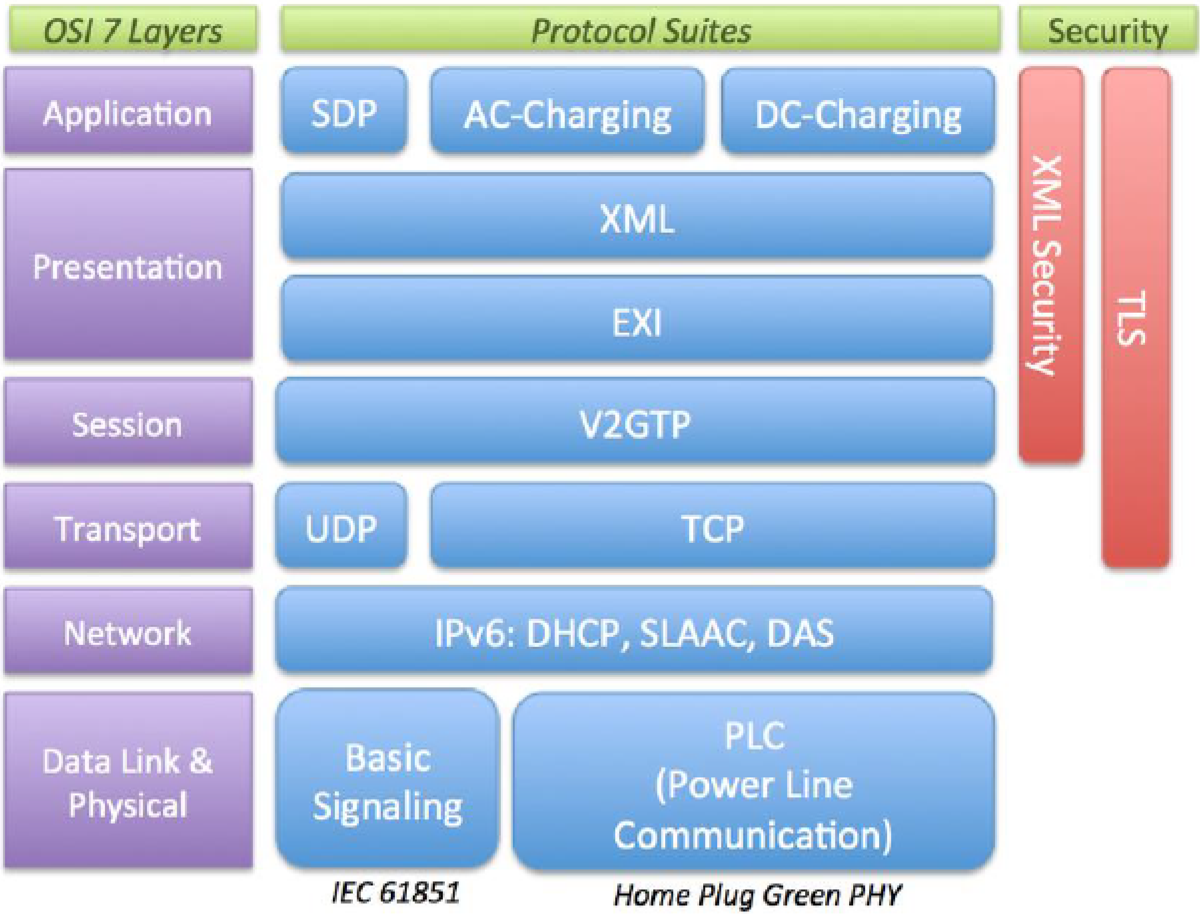


Fig. 85: ISO 15118 communication stack

In conjunction with the EL6761 EtherCAT Terminal, high-level communication can be used to obtain information about the current state of the vehicle, e.g. the current SoC (State of Charge), the remaining charging time, etc., and to enable DC charging.

5.4 Notes to Commissioning

Software functions can be accessed via TwinCAT on the basis of the EtherCAT protocol. The process data objects (PDOs) provide access to the module's important data outputs. Further settings and information can be read or written in places via CANopen over EtherCAT (CoE).

AC operation

The terminal handles the communication for an AC charging process between it (or the charging station function implemented in the customer application, Electrical Vehicle Supply Equipment (EVSE)) and an electrical vehicle (EV) via PWM communication in accordance with IEC 61851 and high-level communication in accordance with ISO 15118.

DC operation

The terminal handles the communication for a DC charging process between it (or the charging station function implemented in the customer application, Electrical Vehicle Supply Equipment (EVSE)) and an electrical vehicle (EV) via PWM communication in accordance with IEC 61851 and high-level communication in accordance with ISO 15118.

Plug Type setting

This setting option can be used in the application to select which type of charging configuration is used: Type 1 or Type 2.

High/low-level communication

The PDO setting can be used to set the representation of the communication levels between high and low level or direct current (DC) and alternating current (AC) charging:

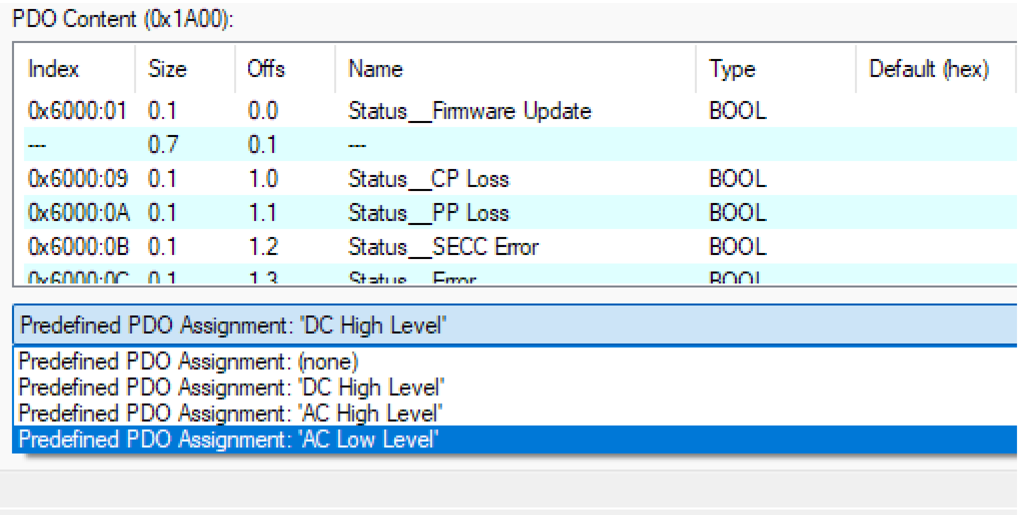


Fig. 86: Predefined PDO

Behavior in the event of PP/CP loss

The amplitude of the pilot signal CP (Control Pilot) and the Proximity Pilot (PP) is evaluated. As the CP/PP-Loss functions are largely implemented in hardware, state transitions and error states (e.g. wire break) in the charging process can be detected and signaled very quickly.

You can use the CP/PP-Loss functions to comply with the switch-off times required by the standards. To do this, make sure that the power flow is interrupted within the standardized switch-off times after the CP/PP-Loss signals are triggered and that the charging line is de-energized (see IEC 61851-23).

Example circuit EL6761

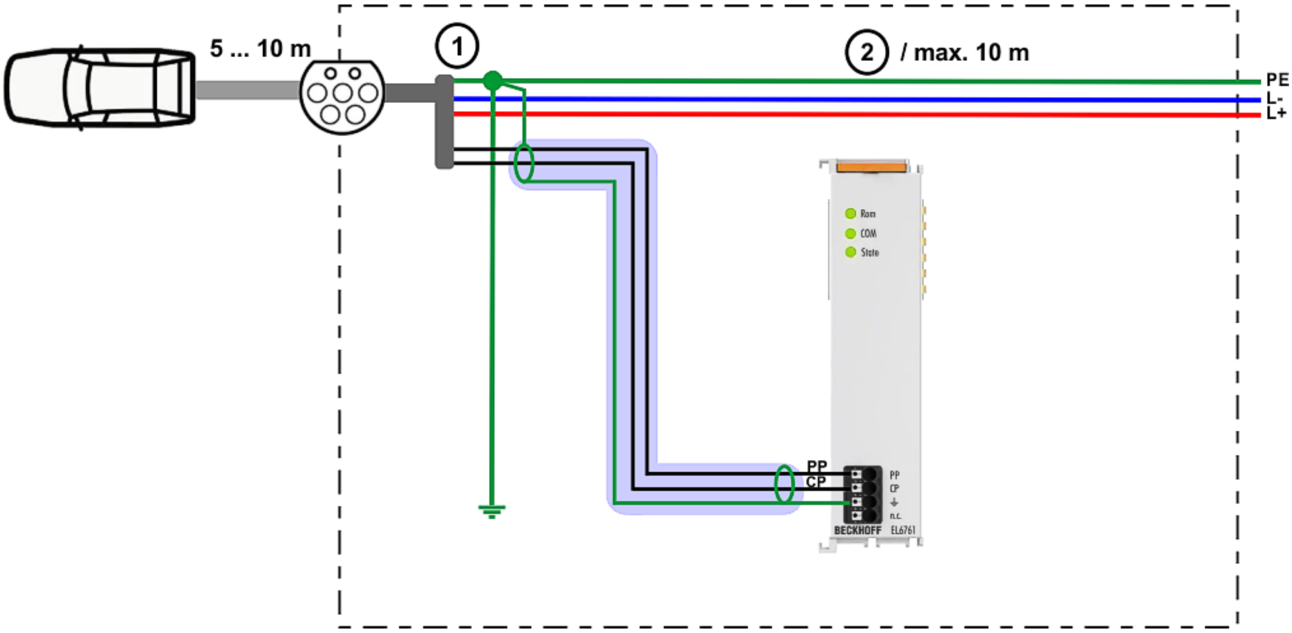


Fig. 87: Example circuit EL6761

Point 1 - common grounding

Point 2 - the shielded power for forwarding the CP signal should not be longer than 15 m, as the total line is limited by standards to 30 m. The remaining length can be used for the cable to the EV.

The EL6761 terminal is intended exclusively for providing communication between it (or the charging station function implemented in the customer application, EVSE) and an EV. Many other functions are missing for a charging station function, such as the provision of power electronics to ensure the desired energy transfer.

NOTICE

EtherCAT cycle time

For optimum diagnostics, the additional diagnostic PDOs should be collected with a cycle time of 1 ms.

Cable Type CoE parameters

	Type1	Type2
DC	PP detection mandatory (CCS1) in accordance with SAE J1772 -> Parameter is ignored and PP detection is executed	No PP detection (IEC 61851) -> Parameter is ignored and PP detection is not executed
AC	Type 1 cable with AC permanently connected: PP detection optional according to SAE J1772 -> depends on the configuration whether PP is recognized or not	Permanently connected cable: No PP detection (IEC 61851) (Maximum current of the cable should be known to the host controller, which sends the maximum current) -> Parameter must be set to PP detection "off" ——— Loose cable: PP detection to determine the coding resistance in the cable for the maximum current of the cable (IEC 61851) -> Parameter must be set to PP detection "on"

CP-Loss is saving

- Is evaluated in state C or D and output as an error if necessary
- CP must be unplugged and go through state B again, then the error is cleared

Authentication phase and PreCharge phase

The following must be observed:

The authentication phase must not exceed 55 s.
The PreCharge phase must not exceed 7 s.

Parameter setting only in "PreOP" mode

Please note the following:

The parameters can only be set in "PreOp" mode.

5.4.1 Example 1

In the first example, basic communication between EVSE and EV is implemented on the basis of IEC 61851 low-level communication using the EL6761 EtherCAT Terminal.

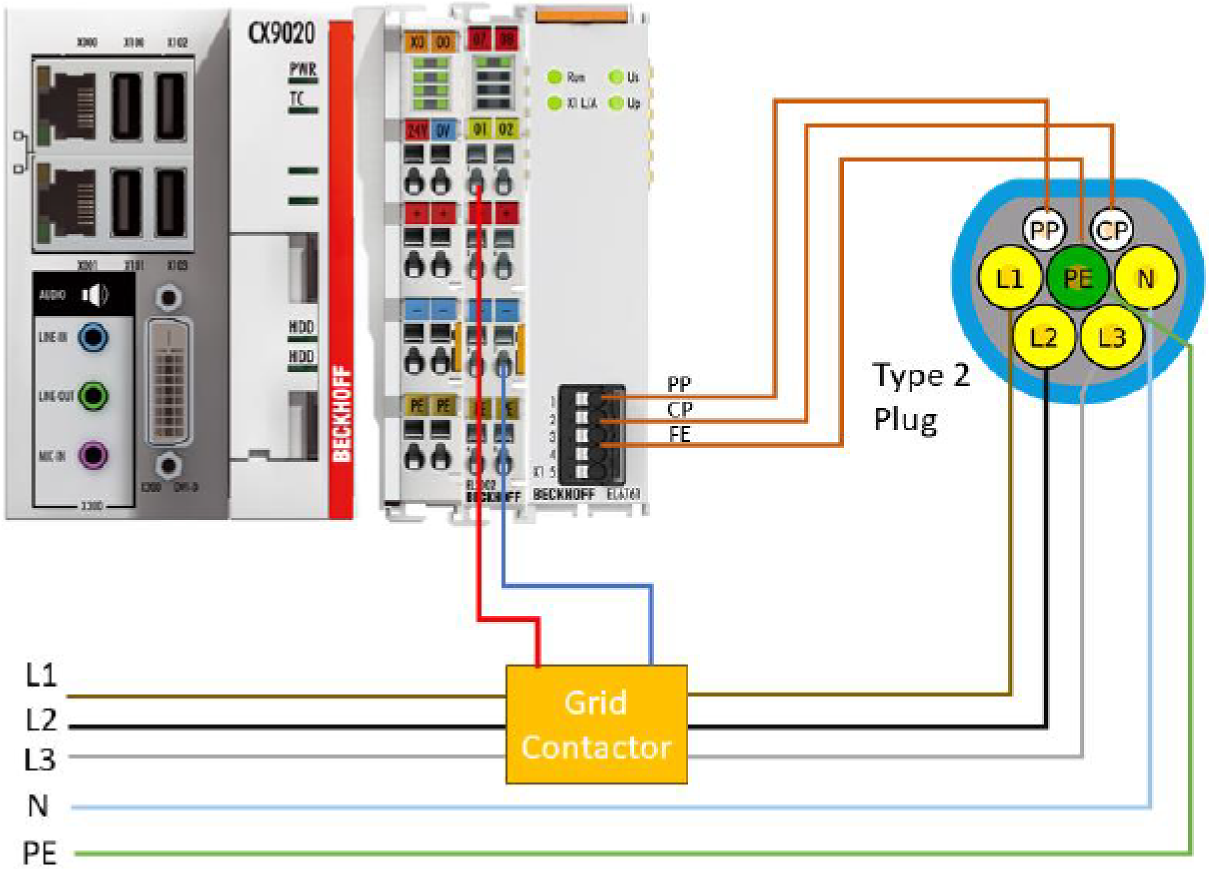
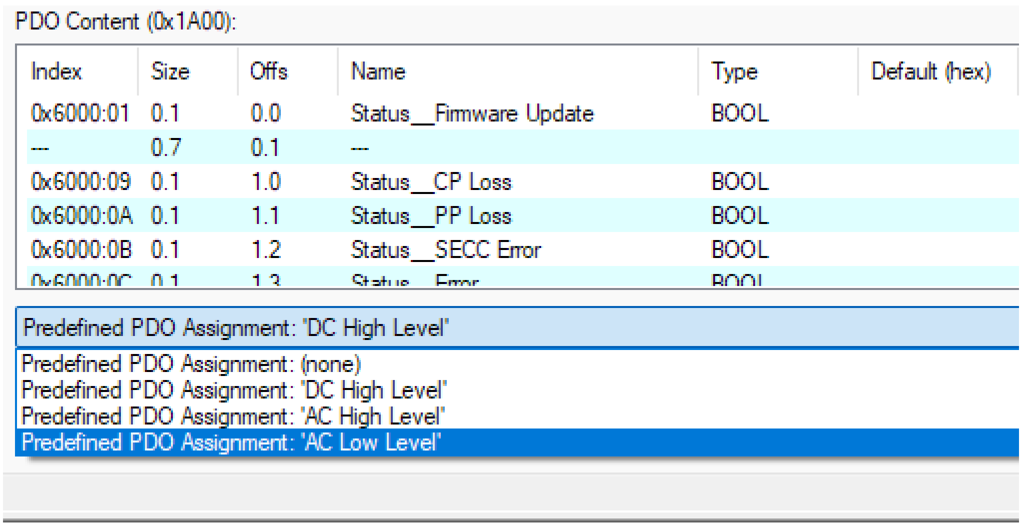


Fig. 88: Exemplary setup: the structure is simplified with regard to safety features such as residual current protection.

To use the terminal for low-level communication, the predefined PDO setting "AC Low Level" is selected in TwinCAT.


PDO Content (0x1A00):

Index	Size	Offs	Name	Type	Default (hex)
0x6000:01	0.1	0.0	Status__Firmware Update	BOOL	
---	0.7	0.1	---		
0x6000:09	0.1	1.0	Status__CP Loss	BOOL	
0x6000:0A	0.1	1.1	Status__PP Loss	BOOL	
0x6000:0B	0.1	1.2	Status__SECC Error	BOOL	
0x6000:0C	0.1	1.3	Status__Error	BOOL	

Predefined PDO Assignment: 'DC High Level'

Predefined PDO Assignment: (none)
Predefined PDO Assignment: 'DC High Level'
Predefined PDO Assignment: 'AC High Level'
Predefined PDO Assignment: 'AC Low Level'

Fig. 89: Predefined PDO "AC Low Level"

In addition, the terminal is configured via CoE Index 8000 (SECC settings) as shown in the "Index 8000 settings" figure.

8000:0	SECC Settings		> 5 <
8000:01	Pilot Control	RW	Automatic (1)
8000:02	Plug Type	RW	Type 2 (2)
8000:03	PP Detection	RW	Off (0)
8000:04	Charge Protocol	RW	Automatic (0)
8000:05	EVSE ID	RW	DE*BEC*EL6761*0000

Fig. 90: Index 8000 settings

To establish communication in TwinCAT, a state machine realized by switch-case determination handles the states of the Control Pilot State input PDO. The resulting state diagram is shown in Figure "State sequence of the Pilot State Machine".

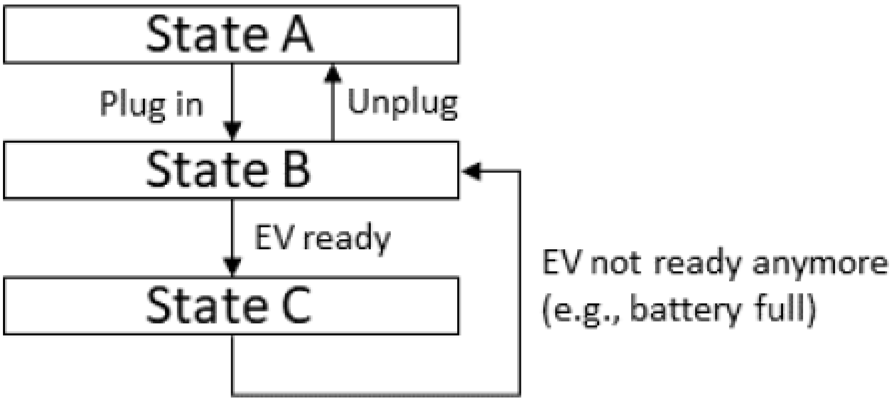


Fig. 91: State sequence of the Pilot State Machine

As soon as the terminal is connected to the EV, it changes the PDO from state A to state B, as shown above. As soon as the connection has been successfully established and the vehicle is ready for charging (state machine changes to state C), the mains contactor can be switched on and the charging process can be started. Once the battery is fully charged, the state changes back to B.

5.4.2 Example 2

In the second example, a DC charging session is set up with the Beckhoff EL6761 EtherCAT Terminal and high-level communication.

The Sevenstax V2G Simulator was used to test the implementation. The simulator is connected to the EL6761 terminal via the CP connection. In addition, the power supply is provided via one of the USB interfaces. The LAN interface enables interaction with the simulator via HTTP.

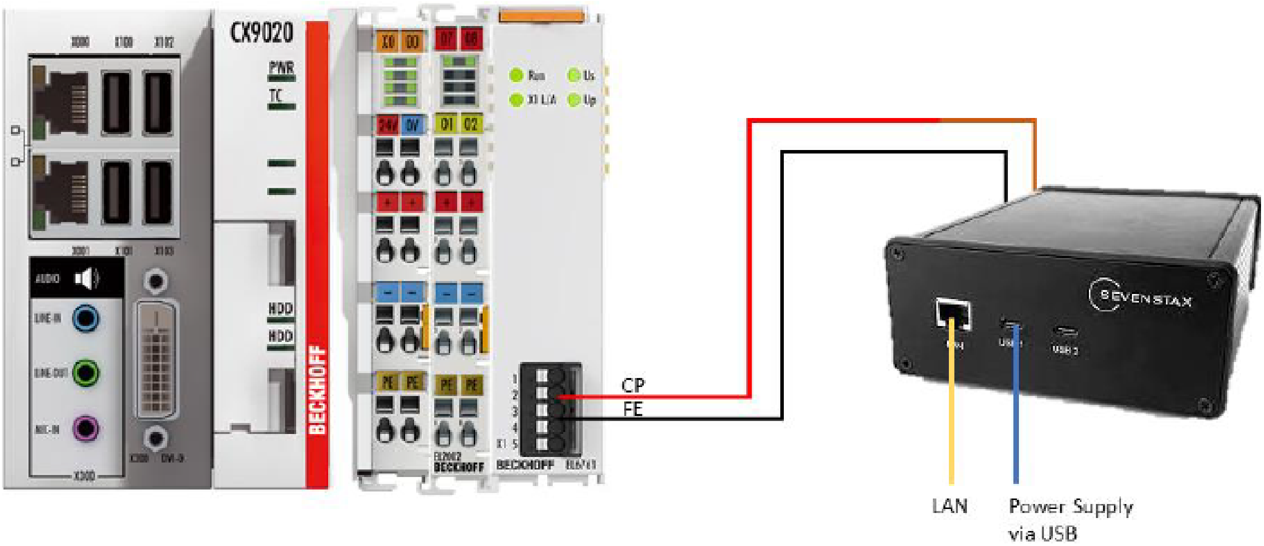


Fig. 92: Exemplary setup 2 using the Sevenstax simulator

To use the terminal for high-level communication, the predefined PDO setting "DC High Level" is selected.

PDO Content (0x1A00):

Index	Size	Offs	Name	Type	Default (hex)
0x6000:01	0.1	0.0	Status__Firmware Update	BOOL	
---	0.7	0.1	---		
0x6000:09	0.1	1.0	Status__CP Loss	BOOL	
0x6000:0A	0.1	1.1	Status__PP Loss	BOOL	
0x6000:0B	0.1	1.2	Status__SECC Error	BOOL	
0x6000:0C	0.1	1.3	Status__Error	BOOL	

Predefined PDO Assignment: 'DC High Level'

Predefined PDO Assignment: (none)
Predefined PDO Assignment: 'DC High Level'
Predefined PDO Assignment: 'AC High Level'
Predefined PDO Assignment: 'AC Low Level'

Fig. 93: Predefined PDO "DC High Level"

In addition, the terminal is configured via CoE Index 8000 (SECC settings) as shown in the "Index 8000 settings" figure.

8000:0	SECC Settings		> 5 <
8000:01	Pilot Control	RW	Automatic (1)
8000:02	Plug Type	RW	Type 2 (2)
8000:03	PP Detection	RW	Off (0)
8000:04	Charge Protocol	RW	Automatic (0)
8000:05	EVSE ID	RW	DE*BEC*EL6761*0000

Fig. 94: Index 8000 settings

For the transmission of high-level communication, the terminal uses an extended state machine, which is provided via the state machine state PDO.

Here too, the state transitions between the state machine states are performed by the terminal, and the application must process the states in the TwinCAT PLC. The states are shown below.

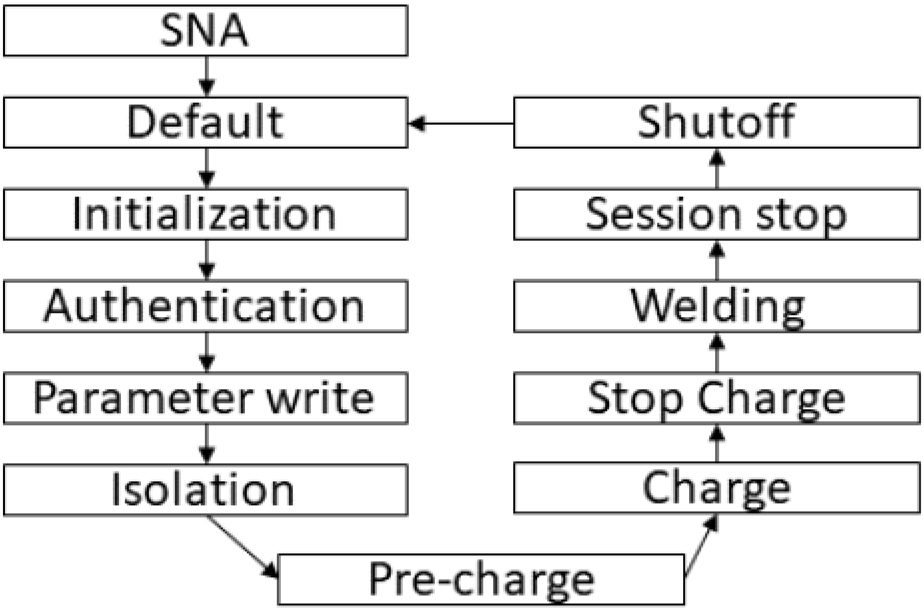


Fig. 95: State sequence of the state machine

After starting, the application initializes its outputs to defined values (SNA state), reaches the default state and remains there until the EV is connected.

When the EV is connected, the application must set current limits (minimum, maximum, ripple and regulation tolerance) as well as voltage and power limits. In addition, the responses for the processing of CA (Control Authentication), CPD (Control Parameters) and CC (Control Isolation) must be set to "ongoing" as they have not yet been completed.

In the following authentication step, the CA response is set to completed. In the parameter step, this is repeated for the CPD response and for the CC response in the "Isolation" state. In addition, the isolation step sets the current and voltage PDOs to zero to enable "ramping" in the following state.

5.5 Overview of parameter objects (CoE)

EtherCAT XML Device Description

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

Parameterization via the CoE list (CAN over EtherCAT)

The EtherCAT device is parameterized via the CoE-Online tab [▸ 68] (double-click on the respective object) or via the Process Data tab [▸ 65](allocation of PDOs). Please note the following general CoE notes [▸ 21] 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 [▸ 138]" for resetting changes

5.5.1 Restore objects

Index 1011 Restore default parameters

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

5.5.2 Configuration data

Index 8000 SECC Settings

Index (hex)	Name	Meaning		Data type	Flags	Default
8000:0	SECC Settings	Max. subindex		UINT8	RO	0x05 (5_{dec})
8000:01	Pilot Control	0	Manual	UINT8	RW	0x01 (1_{dec})
		1	Automatic			
8000:02	Plug Type	1	Type 1	UINT8	RW	0x02 (2_{dec})
		2	Type 2			
8000:03	PP Detection	0	Off	UINT8	RW	0x00 (0_{dec})
		1	On			
8000:04	Charge Protocol	0	Automatic	UINT8	RW	0x00 (0_{dec})
		1	DIN 70121			
		2	ISO 15118			
8000:05	EVSE ID	This signal must contain the ID selected by the EVSE, which it transmits to the vehicle.		STRING	RW	DE*BEC*EL6761*0000

Index 8001 SECC Slac Settings

Index (hex)	Name	Meaning	Data type	Flags	Default
8001:0	SECC Slac Settings	Max. subindex	UINT8	RO	0x01 (1_{dec})
8001:01	Slac Rx Attenuation Correction	This value corrects the attenuation profile calculated with SLAC in [dB]	UINT8	RW	0x00 (0_{dec})
		0			
		3			
		6			
		9			
		12			
		15			
		18			
		21			

5.5.3 Configuration data (vendor-specific)

Index 8002 SECC Thresholds

Index (hex)	Name	Meaning	Data type	Flags	Default
8002:0	SECC Thresholds	Max. subindex	UINT8	RO	0x14 (20_{dec})
8002:01	CP Threshold High State A	CP signal threshold for state A (High)	UINT16	RW	0x32C8 (13000_{dec})
8002:02	CP Threshold Low State A	CP signal threshold for state A (Low)	UINT16	RW	0x29FE (10750_{dec})
8002:03	CP Threshold High State B	CP signal threshold for state B (High)	UINT16	RW	0x280A (10250_{dec})
8002:04	CP Threshold Low State B	CP signal threshold for state B (Low)	UINT16	RW	0x1E46 (7750_{dec})
8002:05	CP Threshold High State C	CP signal threshold for state C (High)	UINT16	RW	0x1C52 (7250_{dec})
8002:06	CP Threshold Low State C	CP signal threshold for state C (Low)	UINT16	RW	0x128E (4750_{dec})
8002:07	CP Threshold High State D	CP signal threshold for state D (High)	UINT16	RW	0x109A (4250_{dec})
8002:08	CP Threshold Low State D	CP signal threshold for state D (Low)	UINT16	RW	0x06D6 (1750_{dec})
8002:09	CP Threshold High State E	CP signal threshold for state E (High)	UINT16	RW	0x04E2 (1250_{dec})
8002:0A	CP Threshold Low State E	CP signal threshold for state E (Low)	INT16	RW	0x0000 (0_{dec})
8002:0B	CP Threshold High State F	CP signal threshold for state F (High)	INT16	RW	0xD8F0 (55536_{dec})
8002:0C	CP Threshold Low State F	CP signal threshold for state F (Low)	INT16	RW	0xCD38 (52536_{dec})
8002:0D	PP Threshold High 13A	PP signal threshold for 13A (High)	UINT16	RW	0x0BE0 (3040_{dec})
8002:0E	PP Threshold Low 13A	PP signal threshold for 13A (Low)	UINT16	RW	0x0A5A (2650_{dec})
8002:0F	PP Threshold High 20A	PP signal threshold for 20A (High)	UINT16	RW	0x09F6 (2550_{dec})
8002:10	PP Threshold Low 20A	PP signal threshold for 20A (Low)	UINT16	RW	0x0758 (1880_{dec})
8002:11	PP Threshold High 32A	PP signal threshold for 32A (High)	UINT16	RW	0x0686 (1670_{dec})
8002:12	PP Threshold Low 32A	PP signal threshold for 32A (Low)	UINT16	RW	0x0474 (1140_{dec})
8002:13	PP Threshold High 63A	PP signal threshold for 63A (High)	UINT16	RW	0x0406 (1030_{dec})
8002:14	PP Threshold Low 63A	PP signal threshold for 63A (Low)	UINT16	RW	0x029E (670_{dec})

5.5.4 Input data

Index 6000 SECC State

Index (hex)	Name	Meaning	Data type	Flags	Default
6000:0	SECC State	Max subindex	UINT8	RO	0x0F (15_{dec})
6000:01	Firmware Update	Firmware update of the CME active	BOOLEAN	RO	0x00 (0_{dec})
6000:09	CP Loss	CP-Loss HW detection	BOOLEAN	RO	0x00 (0_{dec})
6000:0A	PP Loss	PP-Loss HW detection	BOOLEAN	RO	0x00 (0_{dec})
6000:0B	SECC Error	EL6761 error (without CME)	BOOLEAN	RO	0x00 (0_{dec})
6000:0C	Error	Collective error bit of the EL6761	BOOLEAN	RO	0x00 (0_{dec})
6000:0D	Diag	Indicates that a new message is available in the "Diag History".	BOOLEAN	RO	0x00 (0_{dec})
6000:0E	TxPDO State	Validity of the data of the associated TxPDO (0 = valid, 1 = invalid).	BOOLEAN	RO	0x00 (0_{dec})
6000:0F	Input cycle counter	This 2-bit counter is incremented on each process data cycle and toggles to 0 after reaching its maximum value of 3.	BIT2	RO	0x00 (0_{dec})

Index 6001 SECC Slac

Index (hex)	Name	Meaning	Data type	Flags	Default
6001:0	SECC Slac	Max subindex	UINT8	RO	0x02 (2_{dec})
6001:01	Link Status	Displays the status of the SLAC process.	UINT8	RO	0x00 (0_{dec})
6001:02	Measured attenuation	Measured attenuation	UINT8	RO	0x00 (0_{dec})

Index 6002 SECC Charge Status

Index 02 units

Currents are always in the unit [0.1 A]
Voltages are always in the unit [0.1 V]
Power is always in the unit [100 W]
Energy is always in the unit [100 Wh]

Index (hex)	Name	Meaning	Data type	Flags	Default
6002:0	SECC Charge Status	Max subindex	UINT8	RO	0x1C (28_{dec})
6002:01	State Machine State	Current state of the charging process	UINT8	RO	0x00 (0_{dec})
6002:02	CP State	CP state	UINT8	RO	0x00 (0_{dec})
6002:03	PP State	State of the PP	UINT8	RO	0x00 (0_{dec})
6002:04	CP Duty Cycle	CP duty cycle [%]	UINT8	RO	0x00 (0_{dec})
6002:05	Diode Presence	Indicates whether the diode of an EV is detected in the CP circuit.	UINT8	RO	0x00 (0_{dec})
6002:06	Charging Complete	If it is set to true, the EV indicates that it is fully charged.	UINT8	RO	0x00 (0_{dec})
6002:07	Bulk Charging Complete	If it is set to true, the EV indicates that bulk charging is complete.	UINT8	RO	0x00 (0_{dec})
6002:08	Actual Charge Protocol	The charging protocol used for the current connection.	UINT8	RO	0x00 (0_{dec})
6002:09	TCP Status	Indicates whether a TCP connection is established between EV and EVSE	UINT8	RO	0x00 (0_{dec})
6002:0A	Cabin Conditioning	Vehicle cabin air conditioning, If true, the EV uses EV energy from the EVSE to heat or cool the passenger compartment.	UINT8	RO	0x00 (0_{dec})
6002:0B	Ress Conditioning	Vehicle RESS conditioning, the vehicle uses the energy from the DC charger to condition the RESS to a target temperature.	UINT8	RO	0x00 (0_{dec})
6002:0C	EV Ready	Indicates that the vehicle is ready for further charging (TRUE). This is a display parameter that must not affect the charging process.	UINT8	RO	0x00 (0_{dec})
6002:0D	Target Current	The charging current requested by the vehicle. [A]	UINT16	RO	0x0000 (0_{dec})
6002:0E	Target Voltage	The target voltage of the vehicle during the charging process. [V]	UINT16	RO	0x0000 (0_{dec})
6002:0F	Target Pre Charge Voltage	This is the desired pre-charging voltage of the vehicle. [V]	UINT16	RO	0x0000 (0_{dec})
6002:10	Min Current	This is the minimum current of the vehicle battery. [A]	UINT16	RO	0x0000 (0_{dec})
6002:11	Max Current	This is the maximum current of the vehicle battery. [A]	UINT16	RO	0x0000 (0_{dec})
6002:13	Max Voltage	This is the maximum voltage of the vehicle battery. [V]	UINT16	RO	0x0000 (0_{dec})
6002:14	Max Power	This is the maximum power of the vehicle battery. [W]	UINT16	RO	0x0000 (0_{dec})
6002:15	SoC	The current SoC of the vehicle for display at the charging station. [%]	UINT8	RO	0x00 (0_{dec})
6002:16	Full SoC	This is the SoC of the EV battery, which should be displayed as fully charged. [%]	UINT8	RO	0x00 (0_{dec})
6002:17	Bulk SoC	This is the charge level at which the vehicle battery is fully charged and the vehicle wants to end the fast charging process. [%]	UINT8	RO	0x00 (0_{dec})
6002:18	Energy Transfer Mode Request	Displays the requested energy transfer mode: "energyTransferType" ISO 15118-2 DC-HL: 0 = DC-extended AC-HL: 0 = AC-Single-phase + AC-three-phase 1 = AC-single phase 3 = AC-three phase	UINT8	RO	0x00 (0_{dec})
6002:19	Energy Capacity	This is the maximum energy capacity supported by the EV. [Wh]	UINT16	RO	0x0000 (0_{dec})
6002:1A	Energy Request	This is the amount of energy that the EV requests from the EVSE. [Wh]	UINT16	RO	0x0000 (0_{dec})
6002:1B	Time to Full SoC	Estimated time until the vehicle is fully charged. [s]	UINT32	RO	0x00000000 (0_{dec})
6002:1C	Time to Bulk SoC	Estimated time for the vehicle to complete fast charging. [s]	UINT32	RO	0x00000000 (0_{dec})

Index 6003 SECC Diagnostics Data

Index (hex)	Name	Meaning	Data type	Flags	Default
6003:0	SECC Diagnostics Data	Max. subindex	UINT8	RO	0x16 (22_{dec})
6003:01	Send__Send Counter	reserved	UINT8	RO	0x00 (0_{dec})
6003:02	Send__Send Data	reserved	OCTET-STRING[19]	RO	{0}
6003:03	Receive__Receive Byte Counter	reserved	UINT16	RO	0x0000 (0_{dec})
6003:04	Receive__Receive Block Counter	reserved	UINT8	RO	0x00 (0_{dec})
6003:05	Receive__Alive Counter	reserved	UINT8	RO	0x00 (0_{dec})
6003:06	Diag__Diag Code 0	reserved	UINT32	RO	0x00000000 (0_{dec})
6003:07	Diag__Diag Code 1	reserved	UINT32	RO	0x00000000 (0_{dec})
6003:08	Diag__Diag Code 2	reserved	UINT32	RO	0x00000000 (0_{dec})
6003:09	Diag__Diag Code 3	reserved	UINT32	RO	0x00000000 (0_{dec})
6003:0A	Debug__Slac Status	reserved	UINT8	RO	0x00 (0_{dec})
6003:0B	Debug__Pilot States Status	reserved	UINT8	RO	0x00 (0_{dec})
6003:0C	Debug__PP Voltage	reserved	UINT16	RO	0x0000 (0_{dec})
6003:0D	Debug__CP Voltage	reserved	UINT16	RO	0x0000 (0_{dec})
6003:0E	Debug__V2G Status DCISO	reserved	UINT8	RO	0x00 (0_{dec})
6003:0F	Debug__V2G Status ACISO	reserved	UINT8	RO	0x00 (0_{dec})
6003:10	Debug__V2G Status DIN	reserved	UINT8	RO	0x00 (0_{dec})
6003:11	Watchdog__Watchdog Reset Counter	reserved	UINT8	RO	0x00 (0_{dec})
6003:12	Watchdog__Watchdog Reset Cause	reserved	UINT8	RO	0x00 (0_{dec})
6003:13	Watchdog__Watchdog Reset Log 0	reserved	UINT8	RO	0x00 (0_{dec})
6003:14	Watchdog__Watchdog Reset Log 1	reserved	UINT8	RO	0x00 (0_{dec})
6003:15	Watchdog__Watchdog Reset Log 2	reserved	UINT8	RO	0x00 (0_{dec})
6003:16	Watchdog__Watchdog Reset Log 3	reserved	UINT8	RO	0x00 (0_{dec})

Index 6004 SECC Plug and Charge

Index (hex)	Name	Meaning	Data type	Flags	Default
6004:0	SECC Plug and Charge	Max. subindex	UINT8	RO	0x04 (4_{dec})
6004:01	EMAID	reserved	STRING	RO	
6004:02	EMAID Validation Status	reserved	UINT8	RO	0x00 (0_{dec})
6004:03	Certificate Validation Status	reserved	UINT8	RO	0x00 (0_{dec})
6004:04	Signature Validation Status	reserved	UINT8	RO	0x00 (0_{dec})

5.5.5 Output data

Index 7000 SECC Control

Index (hex)	Name	Meaning	Data type	Flags	Default
7000:0	SECC Control	Maximum subindex	UINT8	RO	0x01 (1_{dec})
7000:01	Reset	Restart terminal (only use in the event of an error)	BOOLEAN	RO	0x00 (0_{dec})

Index 7001 SECC Charge Control

Index (hex)	Name	Meaning	Data type	Flags	Default
7001:0	SECC Charge Control	Maximum subindex	UINT8	RO	0x1C (28_{dec})
7001:01	Control Code	This can be used to send charging commands: Possible values: 0 - EVSE Not Ready 1 - EVSE Ready 2 - EVSE Shutdown 5 - EVSE Emergency Shutdown 6 - EVSE Malfunction	UINT8	RO	0x00 (0_{dec})
7001:02	CP State	Displays the detected CP state (can only be used if Pilot Control (0x8000:01 [▸ 100]) is set to Manual (0)).	UINT8	RO	0x00 (0_{dec})
7001:03	PP State	Specifies the type of cable that is connected via the proximity resistor coding (can only be used if Pilot Control (0x8000:01 [▸ 100]) is set to Manual (0)).	UINT8	RO	0x00 (0_{dec})
7001:04	Free Service	This signal can be used to offer a service without payment. Set to true if the service is free, otherwise the negotiated payment method will be used for billing.	UINT8	RO	0x00 (0_{dec})
7001:05	CP Duty Cycle	Displays the currently set positive duty cycle of the charger on the control pilot line [%], can only be used if Pilot Control (0x8000:01 [▸ 100]) is set to Manual (0).	UINT16	RO	0x0000 (0_{dec})
7001:06	Isolation Status	Shows the current isolation status of the EVSE: 0 - Invalid 1 - Valid	UINT8	RO	0x00 (0_{dec})
7001:07	Processing Authentication	Displays the authentication status: 0 – Ongoing 1 – Finished 2 - Waiting For Customer	UINT8	RO	0x00 (0_{dec})
7001:08	Processing Parameter	Displays the processing parameter status: 0 – Ongoing 1 – Finished 2 - Waiting For Customer	UINT8	RO	0x00 (0_{dec})
7001:09	Processing Isolation	Displays the isolation status: 0 – Ongoing 1 – Finished 2 - Waiting For Customer	UINT8	RO	0x00 (0_{dec})
7001:0A	Current Limit Achieved	Indicates whether the EVSE has reached its current limit.	UINT8	RO	0x00 (0_{dec})
7001:0B	Voltage Limit Achieved	Indicates whether the EVSE has reached its voltage limit.	UINT8	RO	0x00 (0_{dec})
7001:0C	Power Limit Achieved	Indicates whether the EVSE has reached its power limit.	UINT8	RO	0x00 (0_{dec})
7001:0D	Current Regulation Tolerance	This is the maximum size of the current control tolerance of the EVSE.	UINT8	RO	0x00 (0_{dec})
7001:0E	Current Peak Ripple	This is the peak-to-peak magnitude of the current ripple of the EVSE.	UINT8	RO	0x00 (0_{dec})
7001:0F	Energy Transfer Mode Support	Displays the energy transfer mode according to the set PDO setting: DC-HL: 0 = DC-extended AC-HL: 0 = AC-Single-phase + AC-three-phase 1 = AC-single phase 3 = AC-three phase	UINT8	RO	0x00 (0_{dec})
7001:10	Min Current	This is the minimum current that the EVSE can supply. [A]	UINT16	RO	0x0000 (0_{dec})
7001:11	Max Current	This is the maximum current that the EVSE can supply. [A]	UINT16	RO	0x0000 (0_{dec})
7001:12	Min Voltage	This is the minimum voltage that the EVSE can supply. [V]	UINT16	RO	0x0000 (0_{dec})
7001:13	Max Voltage	This value indicates the maximum voltage that the vehicle can request. [V]	UINT16	RO	0x0000 (0_{dec})
7001:14	Max Power	This value indicates the maximum power that the vehicle can request. [W]	UINT16	RO	0x0000 (0_{dec})
7001:15	Nominal Voltage	Displays the supported phase voltage of the EVSE in volts - measured between phase and neutral conductor.	UINT16	RO	0x0000 (0_{dec})

Index (hex)	Name	Meaning	Data type	Flags	Default
7001:16	Setpoint Current	Charging current setpoint For operation mode: AC-HighLevel: Setpoint charging current INT scaled in [0.1A] For operation mode: AC-LowLevel: see chapter Basic function principles	UINT16	RO	0x0000 (0_{dec})
7001:17	Present Current	This is the current output current of the EVSE. [A]	UINT16	RO	0x0000 (0_{dec})
7001:18	Present Voltage	This is the current output voltage of the EVSE. [V]	UINT16	RO	0x0000 (0_{dec})
7001:19	Energy to be Delivered	This is the amount of energy that EVSE can supply. [Wh]	UINT16	RO	0x0000 (0_{dec})
7001:1C	Actual Timestamp	This is the current timestamp of the charger as a UNIX® timestamp (i.e. the number of seconds since January 1, 1970) [s]	UINT64	RO	

5.5.6 Information and diagnostic data

Index 9000 SECC Info Data

Index (hex)	Name	Meaning	Data type	Flags	Default
9000:0	SECC Info Data	Max. subindex	UINT8	RO	0x0E (14_{dec})
9000:01	SECC Firmware Update Progress	Percentage progress of the CME firmware updates	UINT8	RO	0x00 (0_{dec})
9000:02	SECC Firmware Version Major	Firmware Version Major of the CME	UINT8	RO	0x00 (0_{dec})
9000:03	SECC Firmware Version Minor	Firmware Version Minor of the CME	UINT8	RO	0x00 (0_{dec})
9000:04	SECC Firmware Version Patch	Firmware Version Patch of the CME	UINT8	RO	0x00 (0_{dec})
9000:05	SECC Firmware Version Hash	Firmware Version Hash of the CME	UINT32	RO	0x00000000 (0_{dec})
9000:06	SECC Firmware Version Config	Firmware Version Config of the CME	UINT8	RO	0x00 (0_{dec})
9000:07	SECC Firmware Version Bootloader	Firmware Version Bootloader of the CME	UINT8	RO	0x00 (0_{dec})
9000:08	SECC Mac Address	Mac Adress Host	OCTET-STRING[6]	RO	{0}
9000:09	SECC Serial Number	CME serial number	STRING	RO	
9000:0A	PLC Firmware Version Major	Firmware Version Major of the PowerLineCommunication chip	UINT8	RO	0x00 (0_{dec})
9000:0B	PLC Firmware Version Minor	Firmware Version Minor of the PowerLineCommunication chip	UINT8	RO	0x00 (0_{dec})
9000:0C	PLC Firmware Version Patch	Firmware Version Patch of the PowerLineCommunication chip	UINT8	RO	0x00 (0_{dec})
9000:0D	PLC Firmware Enum	Firmware Version Enum of the PowerLineCommunication chip	UINT8	RO	0x00 (0_{dec})
9000:0E	PLC Mac Address	Mac address of the PowerLineCommunication chip	OCTET-STRING[6]	RO	{0}

Index 9001 SECC EV Info Data

<table>
<tr><th>Index (hex)</th><th>Name</th><th colspan="2">Meaning</th><th>Data type</th><th>Flags</th><th>Default</th></tr>
<tr><td>9001:0</td><td>SECC EV Info Data</td><td colspan="2">Max. subindex</td><td>UINT8</td><td>RO</td><td>0x01 (1_{dec})</td></tr>
<tr><td rowspan="6">9001:01</td><td rowspan="6">EVCC ID</td><td colspan="2">EV identifier</td><td rowspan="6">UINT64</td><td rowspan="6">RO</td><td rowspan="6"></td></tr>
<tr><td>Diag Code 0:</td><td>Latest diagnostic code from the CME</td></tr>
<tr><td>Diag Code 1:</td><td>Old diagnostic code from the CME</td></tr>
<tr><td>Diag Code 2:</td><td>Older diagnostic code from the CME</td></tr>
<tr><td>Diag Code 3:</td><td>Oldest diagnostic code from the CME</td></tr>
<tr><td>Alive-Counter</td><td>Alive counter of the CME</td></tr>
</table>

Index A000 SECC Diag Data

Index (hex)	Name	Meaning	Data type	Flags	Default
A000:0	SECC Diag Data	Max. Subindex Diagnostic data from the last charging processes. 0 = most recent data 3 = oldest data	UINT8	RO	0x15 (21_{dec})
A000:01	Diag Code 0	Complete diagnostic code of the last charging process	UINT32	RO	0x00000000 (0_{dec})
A000:02	Interface 0	Interface of the diagnostic code	STRING	RO	
A000:03	Severity 0	Severity of the diagnostic code	STRING	RO	
A000:04	Category 0	Category of the diagnostic code	STRING	RO	
A000:05	Error Code 0	Error code of the diagnostic code	STRING	RO	
A000:06	Diag Code 1	Complete diagnostic code of the penultimate charging process	UINT32	RO	0x00000000 (0_{dec})
A000:07	Interface 1	Interface of the diagnostic code	STRING	RO	
A000:08	Severity 1	Severity of the diagnostic code	STRING	RO	
A000:09	Category 1	Category of the diagnostic code	STRING	RO	
A000:0A	Error Code 1	Error code of the diagnostic code	STRING	RO	
A000:0B	Diag Code 2	Complete diagnostic code of the penultimate charging process	UINT32	RO	0x00000000 (0_{dec})
A000:0C	Interface 2	Interface of the diagnostic code	STRING	RO	
A000:0D	Severity 2	Severity of the diagnostic code	STRING	RO	
A000:0E	Category 2	Category of the diagnostic code	STRING	RO	
A000:0F	Error Code 2	Error code of the diagnostic code	STRING	RO	
A000:10	Diag Code 3	Complete diagnostic code of the third last charging process	UINT32	RO	0x00000000 (0_{dec})
A000:11	Interface 3	Interface of the diagnostic code	STRING	RO	
A000:12	Severity 3	Severity of the diagnostic code	STRING	RO	
A000:13	Category 3	Category of the diagnostic code	STRING	RO	
A000:14	Error Code 3	Error code of the diagnostic code	STRING	RO	
A000:15	Alive Counter	Live signal of the charge controller: Counts up continuously (0-15) to signal activity of the charge controller.	UINT8	RO	0x00 (0_{dec})

Index A001 SECC EV Diag Data

Index (hex)	Name	Meaning	Data type	Flags	Default
A001:0	SECC EV Diag Data	Max. subindex	UINT8	RO	0x01 (1_{dec})
A001:01	Diag Code	EVErrorCode - status values are defined as: 0 = EV no error 1 = EV failed ress temperature inhibit 2 = EV failed EV shift position 3 = EV failed charger connector lock fault 4 = EV failed EV ress malfunction 5= EV failed charging current differential 6 = EV failed charging voltage out of range 7 = EV reserved A 8 = EV reserved B 9 = EV reserved C 10 = EV failed charging system incompatibility 11 = EV no data	UINT32	RO	0x00000000 (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 Profiles (MDP) Organizational information on the profiles used in the device and listed in 0xF010	UINT8	RO	0x02 (2_{dec})
F000:01	Module index distance	Index distance of the objects of the individual channels	UINT16	RO	0x0010 (16_{dec})
F000:02	Maximum number of modules	Number of channels	UINT16	RO	0x0001 (1_{dec})

Index F915 LED Status

Index (hex)	Name	Meaning	Data type	Flags	Default
F915:0	LED Status	Max. subindex	UINT8	RO	0x03 (3_{dec})
F915:01	Run	reserved	UINT32	RO	0x00000000 (0_{dec})
F915:02	Com_Err	reserved	UINT32	RO	0x00000000 (0_{dec})
F915:03	State	reserved	UINT32	RO	0x00000000 (0_{dec})

Index FB00 Command

The command object is used for triggering an action in the terminal. The command is started by writing subindex 1 (request). Write access is disabled until the current command is completed.

Index (hex)	Name	Meaning	Data type	Flags	Default
FB00:0	Command	Largest subindex of this object	UINT8	RO	0x03 (3_{dec})
FB00:01	Request	Command value, for use see corresponding application chapter	OCTET-STRING [2]	RW	0x0000 (0_{dec})
FB00:02	State	Command state, for use see corresponding application chapter	UINT8	RW	0x00 (0_{dec})
FB00:03	Response	Command response, for use see corresponding application chapter	OCTET-STRING [2]	RW	0x00000000 (0_{dec})

5.5.7 Standard objects

Index 1000 Device type

Index (hex)	Name	Meaning	Data type	Flags	Default
1000:0	Device type	Device type of the EtherCAT slave: the Lo-Word contains the used CoE profile (5001). The Hi-Word contains the module profile according to the modular device profile.	UINT32	RO	0x02A41389 (44307337_{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	EL6761

Index 1009 Hardware version

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

Index 100A Software version

Index (hex)	Name	Meaning	Data type	Flags	Default
100A:0	Software version	Firmware version of the EtherCAT slave	STRING	RO	00

Index 100B Bootloader version

Index (hex)	Name	Meaning	Data type	Flags	Default
100B:0	Bootloader version	Bootloader version	STRING	RO	n/a

Index 1018 Identity

Index (hex)	Name	Meaning	Data type	Flags	Default
1018:0	Identity	Information for identifying the slave	UINT8	RO	0x04 (4_{dec})
1018:01	Vendor ID	Vendor ID of the EtherCAT slave	UINT32	RO	0x00000002 (2_{dec})
1018:02	Product code	Product code of the EtherCAT slave	UINT32	RO	0x1A693052 (443101266_{dec})
1018:03	Revision	Revision number of the EtherCAT slave; the Low Word (bit 0-15) indicates the special terminal number, the High Word (bit 16-31) refers to the device description	UINT32	RO	0x00000000 (0_{dec})
1018:04	Serial number	Serial number of the EtherCAT slave; the Low Byte (bit 0-7) of the Low Word contains the year of production, the High Byte (bit 8-15) of the Low Word contains the week of production, the High Word (bit 16-31) is 0	UINT32	RO	0x00000000 (0_{dec})

Index 1020 Device Statistics

Index (hex)	Name	Meaning	Data type	Flags	Default
1020:0	Device Statistics	Max. subindex	UINT8	RO	0x03 (3_{dec})
1020:01	Time since power on	Operating time since last power on	UINT32	RO	0x00000000 (0_{dec})
1020:02	Total time powered	Total operating time	UINT32	RO	0x00000000 (0_{dec})
1020:03	Number of power cycles	Number of power cycles	UINT32	RO	0x00000000 (0_{dec})

Index 10E2 Manufacturer-specific Identification Code*)

Index (hex)	Name	Meaning	Data type	Flags	Default
10E2:0	Manufacturer-specific Identification Code	Manufacturer specific Identification Code	UINT8	RO	0x01 (1_{dec})
10E2:01	SubIndex 001	reserved	STRING	RO	

Index 10F0 Backup parameter handling

Index (hex)	Name	Meaning	Data type	Flags	Default
10F0:0	Backup parameter handling	Information for standardized loading and saving of backup entries	UINT8	RO	0x01 (1_{dec})
10F0:01	Checksum	Checksum across all backup entries of the EtherCAT slave	UINT32	RO	0x00000000 (0_{dec})

Index 10F3 Diagnosis History

Index (hex)	Name	Meaning	Data type	Flags	Default
10F3:0	Diagnosis History	Maximum subindex	UINT8	RO	0x15 (21_{dec})
10F3:01	Maximum Messages	Maximum number of stored messages A maximum of 16 messages can be stored	UINT8	RO	0x00 (0_{dec})
10F3:02	Newest Message	Subindex of the latest message	UINT8	RO	0x00 (0_{dec})
10F3:03	Newest Acknowledged Message	Subindex of the last confirmed message	UINT8	RW	0x00 (0_{dec})
10F3:04	New Messages Available	Indicates that a new message is available	BOOLEAN	RO	0x00 (0_{dec})
10F3:05	Flags	not used	UINT16	RW	0x0000 (0_{dec})
10F3:06	Diagnosis Message 001	Message 1	OCTET-STRING[24]	RO	{0}
…	…	….	…	…	…
10F3:15	Diagnosis Message 016	Message 16	OCTET-STRING[24]	RO	{0}

Index 10F8 Timestamp Object

Index (hex)	Name	Meaning	Data type	Flags	Default
10F8:0	Timestamp Object	Timestamp Object [ns] For SM synchronous operation: Time since power-on For DC synchronous operation: copy of the DC time Time can be used by the device e.g. for timestamps of the DiagMessage	UINT64	RO	

Index 1401 SECC RxPDO-Par Charge Control DC High Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1401:0	SECC RxPDO-Par Charge Control DC High Level	PDO Parameter RxPDO 2	UINT8	RO	0x06 (6_{dec})
1401:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 2	OCTET-STRING[4]	RO	02 16 03 16

Index 1402 SECC RxPDO-Par Charge Control AC High Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1402:0	SECC RxPDO-Par Charge Control AC High Level	PDO Parameter RxPDO 3	UINT8	RO	0x06 (6_{dec})
1402:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 3	OCTET-STRING[4]	RO	01 16 03 16

Index 1403 SECC RxPDO-Par Charge Control AC Low Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1403:0	SECC RxPDO-Par Charge Control AC Low Level	PDO Parameter RxPDO 4	UINT8	RO	0x06 (6_{dec})
1403:06	Exclude RxPDOs	Specifies the RxPDOs (index of RxPDO mapping objects) that must not be transferred together with RxPDO 4	OCTET-STRING[4]	RO	01 16 02 16

Index 1600 SECC RxPDO-Map Control

Index (hex)	Name	Meaning	Data type	Flags	Default
1600:0	SECC RxPDO-Map Control	PDO Mapping RxPDO 1	UINT8	RO	0x02 (2_{dec})
1600:01	SubIndex 001	1. PDO Mapping entry (object 0x7000 (SECC Control), entry 0x01 (Reset))	UINT32	RO	0x7000:01, 1
1600:02	SubIndex 002	2. PDO Mapping entry (15 bits align)	UINT32	RO	0x0000:00, 15

Index 1601 SECC RxPDO-Map Charge Control DC High Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1601:0	SECC RxPDO-Map Charge Control DC High Level	PDO Mapping RxPDO 2	UINT8	RO	0x18 (24_{dec})
1601:01	SubIndex 001	1. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x01 (Control Code))	UINT32	RO	0x7001:01, 8
1601:02	SubIndex 002	2. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x02 (CP State))	UINT32	RO	0x7001:02, 8
1601:03	SubIndex 003	3. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x03 (PP State))	UINT32	RO	0x7001:03, 8
1601:04	SubIndex 004	4. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x04 (Free Service))	UINT32	RO	0x7001:04, 8
1601:05	SubIndex 005	5. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x05 (CP Duty Cycle))	UINT32	RO	0x7001:05, 16
1601:06	SubIndex 006	6. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x06 (Isolation Status))	UINT32	RO	0x7001:06, 8
1601:07	SubIndex 007	7. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x07 (Processing Authentication))	UINT32	RO	0x7001:07, 8
1601:08	SubIndex 008	8. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x08 (Processing Parameter))	UINT32	RO	0x7001:08, 8
1601:09	SubIndex 009	9. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x09 (Processing Isolation))	UINT32	RO	0x7001:09, 8
1601:0A	SubIndex 010	10. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x0A (Current Limit Achieved))	UINT32	RO	0x7001:0A, 8
1601:0B	SubIndex 011	11. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x0B (Voltage Limit Achieved))	UINT32	RO	0x7001:0B, 8
1601:0C	SubIndex 012	12. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x0C (Power Limit Achieved))	UINT32	RO	0x7001:0C, 8
1601:0D	SubIndex 013	13. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x0D (Current Regulation Tolerance))	UINT32	RO	0x7001:0D, 8
1601:0E	SubIndex 014	14. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x0E (Current Peak Ripple))	UINT32	RO	0x7001:0E, 8
1601:0F	SubIndex 015	15. PDO Mapping entry (8 bits align)	UINT32	RO	0x7001:0F, 8
1601:10	SubIndex 016	16. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x10 (Min Current))	UINT32	RO	0x7001:10, 16
1601:11	SubIndex 017	17. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x11 (Max Current))	UINT32	RO	0x7001:11, 16
1601:12	SubIndex 018	18. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x12 (Min Voltage))	UINT32	RO	0x7001:12, 16
1601:13	SubIndex 019	19. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x13 (Max Voltage))	UINT32	RO	0x7001:13, 16
1601:14	SubIndex 020	20. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x14 (Max Power))	UINT32	RO	0x7001:14, 16
1601:15	SubIndex 021	21. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x15 (Present Current))	UINT32	RO	0x7001:17, 16
1601:16	SubIndex 022	22. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x16 (Present Voltage))	UINT32	RO	0x7001:18, 16
1601:17	SubIndex 023	23. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x17 (Energy to be Delivered))	UINT32	RO	0x7001:19, 16
1601:18	SubIndex 024	24. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x18 (Actual Timestamp))	UINT32	RO	0x7001:1C, 64

Index 1602 SECC RxPDO-Map Charge Control AC High Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1602:0	SECC RxPDO-Map Charge Control AC High Level	PDO Mapping RxPDO 3	UINT8	RO	0x0C (12_{dec})
1602:01	SubIndex 001	1. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x05 (CP Duty Cycle))	UINT32	RO	0x7001:01, 8
1602:02	SubIndex 002	2. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x04 (Free Service))	UINT32	RO	0x7001:02, 8
1602:03	SubIndex 003	3. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x07 (Processing Authentication))	UINT32	RO	0x7001:03, 8
1602:04	SubIndex 004	4. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x08 (Processing Parameter))	UINT32	RO	0x7001:04, 8
1602:05	SubIndex 005	5. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x11 (Max Current))	UINT32	RO	0x7001:05, 16
1602:06	SubIndex 006	6. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x14 (Nominal Voltage))	UINT32	RO	0x7001:07, 8
1602:07	SubIndex 007	7. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x08 (Processing Parameter))	UINT32	RO	0x7001:08, 8
1602:08	SubIndex 008	8. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x15 (Nominal Voltage))	UINT32	RO	0x7001:15, 16
1602:09	SubIndex 009	9. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x16 (Threshold Current))	UINT32	RO	0x7001:16, 16
1602:0A	SubIndex 010	10. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x1A (Actual Timestamp))	UINT32	RO	0x7001:0F, 8
1602:0B	SubIndex 011	11. PDO Mapping entry (24 bits align)	UINT32	RO	0x0000:00, 24
1602:0C	SubIndex 012	12. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x1C (Actual Timestamp))	UINT32	RO	0x7001:1C, 64

Index 1603 SECC RxPDO-Map Charge Control AC Low Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1603:0	SECC RxPDO-Map Charge Control AC Low Level	PDO Mapping RxPDO 4	UINT8	RO	0x01 (1_{dec})
1603:01	SubIndex 001	1. PDO Mapping entry (object 0x7001 (SECC Charge Control), entry 0x11 (Max Current))	UINT32	RO	0x7001:16, 16

Index 1801 SECC TxPDO-Par Slac Status

Index (hex)	Name	Meaning	Data type	Flags	Default
1801:0	SECC TxPDO-Par Slac Status	PDO Parameter TxPDO 2	UINT8	RO	0x06 (6dec)
1801:06	Exclude TxPDOs	Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 2	OCTET-STRING[8]	RO	04 1A 00 00 00 00 00 00

Index 1802 SECC TxPDO-Par Charge Status DC High Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1802:0	SECC TxPDO-Par Charge Status DC High Level	PDO Parameter TxPDO 3	UINT8	RO	0x06 (6_{dec})
1802:06	Exclude TxPDOs	Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 3	OCTET-STRING[4]	RO	03 1A 04 1A 00 00 00 00

Index 1803 SECC TxPDO-Par Charge Status AC High Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1803:0	SECC TxPDO-Par Charge Status AC High Level	PDO Parameter TxPDO 4	UINT8	RO	0x06 (6_{dec})
1803:06	Exclude TxPDOs	Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 4	OCTET-STRING[4]	RO	02 1A 04 1A 00 00 00 00

Index 1804 SECC TxPDO-Par Charge Status AC Low Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1804:0	SECC TxPDO-Par Charge Status AC Low Level	PDO Parameter TxPDO 5	UINT8	RO	0x06 (6_{dec})
1804:06	Exclude TxPDOs	Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 5	OCTET-STRING[4]	RO	01 1A 02 1A 03 1A 06 1A

Index 1A00 SECC TxPDO-Map Status

Index (hex)	Name	Meaning	Data type	Flags	Default
1A00:0	SECC TxPDO-Map Status	PDO Mapping TxPDO 1	UINT8	RO	0x09 (9_{dec})
1A00:01	SubIndex 001	1. PDO Mapping entry (object 0x6000 (SECC Status), entry 0x01 (Firmware Update))	UINT32	RO	0x6000:01, 1
1A00:02	SubIndex 002	2. PDO Mapping entry (7 bits align)	UINT32	RO	0x0000:00, 7
1A00:03	SubIndex 003	3. PDO Mapping entry (object 0x6000 (SECC Status), entry 0x09 (CP Loss))	UINT32	RO	0x6000:09, 1
1A00:04	SubIndex 004	4. PDO Mapping entry (object 0x6000 (SECC Status), entry 0x0A (PP Loss))	UINT32	RO	0x6000:0A, 1
1A00:05	SubIndex 005	5. PDO Mapping entry (object 0x6000 (SECC Status), entry 0x0B (SECC Error))	UINT32	RO	0x6000:0B, 1
1A00:06	SubIndex 006	6. PDO Mapping entry (object 0x6000 (SECC Status), entry 0x0C (Error))	UINT32	RO	0x6000:0C, 1
1A00:07	SubIndex 007	7. PDO Mapping entry (object 0x6000 (SECC Status), entry 0x0D (Diag))	UINT32	RO	0x6000:0D, 1
1A00:08	SubIndex 008	8. PDO Mapping entry (object 0x6000 (SECC Status), entry 0x0E (TxPDO State))	UINT32	RO	0x6000:0E, 1
1A00:09	SubIndex 009	9. PDO Mapping entry (object 0x6000 (SECC Status), entry 0x0F (Input cycle counter))	UINT32	RO	0x6000:0F, 2

Index 1A01 SECC TxPDO-Map Slac Status

Index (hex)	Name	Meaning	Data type	Flags	Default
1A01:0	SECC TxPDO-Map Slac Status	PDO Mapping TxPDO 2	UINT8	RO	0x02 (2_{dec})
1A01:01	SubIndex 001	1. PDO Mapping entry (object 0x6001 (SECC Slac), entry 0x01 (Link Status))	UINT32	RO	0x6001:01, 8
1A01:02	SubIndex 002	2. PDO Mapping entry (object 0x6001 (SECC Slac), entry 0x02 (Measured Attenuation))	UINT32	RO	0x6001:02, 8

Index 1A02 SECC TxPDO-Map Charge Status DC High Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1A02:0	SECC TxPDO-Map Charge Status DC High Level	PDO Mapping TxPDO 3	UINT8	RO	0x1A (26$_{dec}$)
1A02:01	SubIndex 001	1. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x01 (State Machine State))	UINT32	RO	0x6002:01, 8
1A02:02	SubIndex 002	2. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x02 (CP State))	UINT32	RO	0x6002:02, 8
1A02:03	SubIndex 003	3. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x03 (PP State))	UINT32	RO	0x6002:03, 8
1A02:04	SubIndex 004	4. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x04 (CP Duty Cycle))	UINT32	RO	0x6002:04, 8
1A02:05	SubIndex 005	5. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x05 (Diode Presence))	UINT32	RO	0x6002:05, 8
1A02:06	SubIndex 006	6. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x06 (Charging Complete))	UINT32	RO	0x6002:06, 8
1A02:07	SubIndex 007	7. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x07 (Bulk Charging Complete))	UINT32	RO	0x6002:07, 8
1A02:08	SubIndex 008	8. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x08 (Actual Charge Protocol))	UINT32	RO	0x6002:08, 8
1A02:09	SubIndex 009	9. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x09 (TCP Status))	UINT32	RO	0x6002:09, 8
1A02:0A	SubIndex 010	10. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x0A (Cabin Conditioning))	UINT32	RO	0x6002:0A, 8
1A02:0B	SubIndex 011	11. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x0B (Ress Conditioning))	UINT32	RO	0x6002:0B, 8
1A02:0C	SubIndex 012	12. PDO Mapping entry (8 bits align)	UINT32	RO	0x6002:0C, 8
1A02:0D	SubIndex 013	13. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x0D (Target Current))	UINT32	RO	0x6002:0D, 16
1A02:0E	SubIndex 014	14. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x0E (Target Voltage))	UINT32	RO	0x6002:0E, 16
1A02:0F	SubIndex 015	15. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x0F (Target Pre Charge Voltage))	UINT32	RO	0x6002:0F, 16
1A02:10	SubIndex 016	16. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x10 (Max Current))	UINT32	RO	0x6002:11, 16
1A02:11	SubIndex 017	17. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x11 (Max Voltage))	UINT32	RO	0x6002:13, 16
1A02:12	SubIndex 018	18. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x12 (Max Power))	UINT32	RO	0x6002:14, 16
1A02:13	SubIndex 019	19. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x13 (SoC))	UINT32	RO	0x6002:15, 8
1A02:14	SubIndex 020	20. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x14 (Full SoC))	UINT32	RO	0x6002:16, 8
1A02:15	SubIndex 021	21. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x15 (Bulk SoC))	UINT32	RO	0x6002:17, 8
1A02:16	SubIndex 022	22. PDO Mapping entry (8 bits align)	UINT32	RO	0x6002:18, 8
1A02:17	SubIndex 023	23. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x17 (Time to Full SoC))	UINT32	RO	0x6002:19, 16
1A02:18	SubIndex 024	24. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x18 (Time to Bulk SoC))	UINT32	RO	0x6002:1A, 16
1A02:19	SubIndex 025	25. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x19 (Energy Capacity))	UINT32	RO	0x6002:1B, 32
1A02:1A	SubIndex 026	26. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x1A (Energy Request))	UINT32	RO	0x6002:1C, 32

Index 1A03 SECC TxPDO-Map Charge Status AC High Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1A03:0	SECC TxPDO-Map Charge Status AC High Level	PDO Mapping TxPDO 4	UINT8	RO	0x0D (13$_{dec}$)
1A03:01	SubIndex 001	1. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x02 (CP State))	UINT32	RO	0x6002:01, 8
1A03:02	SubIndex 002	2. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x03 (PP State))	UINT32	RO	0x6002:02, 8
1A03:03	SubIndex 003	3. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x04 (CP Duty Cycle))	UINT32	RO	0x6002:03, 8
1A03:04	SubIndex 004	4. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x05 (Diode Presence))	UINT32	RO	0x6002:04, 8
1A03:05	SubIndex 005	5. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x05 (Diode Presence))	UINT32	RO	0x6002:05, 8
1A03:06	SubIndex 006	6. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x06 (Charging Complete))	UINT32	RO	0x6002:08, 8
1A03:07	SubIndex 007	7. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x07 (Bulk Charging Complete))	UINT32	RO	0x6002:09, 8
1A03:08	SubIndex 008	8. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x08 (Actual Charge Protocol))	UINT32	RO	0x6002:18, 8
1A03:09	SubIndex 009	9. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x09 (TCP Status))	UINT32	RO	0x6002:10, 16
1A03:0A	SubIndex 010	10. PDO Mapping entry (8 bits align)	UINT32	RO	0x6002:11, 16
1A03:0B	SubIndex 011	11. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x10 (Min Current))	UINT32	RO	0x6002:13, 16
1A03:0C	SubIndex 012	12. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x11 (Max Current))	UINT32	RO	0x6002:1A, 16
1A03:0D	SubIndex 013	13. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x12 (Fallback Status))	UINT32	RO	0x6002:12, 16

Index 1A04 SECC TxPDO-Map Charge Status AC Low Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1A04:0	SECC TxPDO-Map Charge Status AC Low Level	PDO Mapping TxPDO 5	UINT8	RO	0x04 (4$_{dec}$)
1A04:01	SubIndex 001	1. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x02 (CP State))	UINT32	RO	0x6002:02, 8
1A04:02	SubIndex 002	2. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x03 (PP State))	UINT32	RO	0x6002:03, 8
1A04:03	SubIndex 003	3. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x04 (CP Duty Cycle))	UINT32	RO	0x6002:04, 8
1A04:04	SubIndex 004	4. PDO Mapping entry (object 0x6002 (SECC Charge Status), entry 0x05 (Diode Presence))	UINT32	RO	0x6002:05, 8

Index 1C00 Sync manager type

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

Index 1C12 RxPDO assign

Index (hex)	Name	Meaning	Data type	Flags	Default
1C12:0	RxPDO assign	PDO Assign Outputs	UINT8	RW	0x02 (2$_{dec}$)
1C12:01	Subindex 001	1. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x1600 (5632$_{dec}$)
1C12:02	Subindex 002	2. allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x1601 (5633$_{dec}$)

Index 1C13 TxPDO assign

Index (hex)	Name	Meaning	Data type	Flags	Default
1C13:0	TxPDO assign	PDO Assign Inputs	UINT8	RW	0x03 (3_{dec})
1C13:01	Subindex 001	1. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A00 (6656_{dec})
1C13:02	Subindex 002	2. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A01 (6657_{dec})
1C13:03	Subindex 003	3. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A02 (6658_{dec})
1C13:04	Subindex 004	4. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x0000 (0_{dec})
1C13:05	Subindex 005	5. allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x0000 (0_{dec})

Index 1C32 SM output parameter

Index (hex)	Name	Meaning	Data type	Flags	Default
1C32:0	SM output parameter	Synchronization parameters for the outputs	UINT8	RO	0x20 (32_{dec})
1C32:01	Sync mode	Current synchronization mode: • 0: Free Run • 1: Synchron with SM 2 Event • 2: DC-Mode - Synchron with SYNC0 Event • 3: DC-Mode - Synchron with SYNC1 Event	UINT16	RW	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 • DC-Mode: SYNC0/SYNC1 Cycle Time	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	0x0000 (0_{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	Get Cycle Time	• 0: Measurement of the local cycle time is stopped • 1: Measurement of the local cycle time is started The entries 1C32:03, 1C32:05, 1C32:06, 1C32:09, 1C33:03, 1C33:06, 1C33:09 are updated with the maximum measured values. For a subsequent measurement the measured values are reset	UINT16	RW	0x0000 (0_{dec})
1C32:09	Maximum delay time	Time between SYNC1 event and output of the outputs (in ns, DC mode only)	UINT32	RO	0x00000000 (0_{dec})
1C32:0B	SM event missed counter	Number of missed SM events in OPERATIONAL (DC Mode only)	UINT16	RO	0x0000 (0_{dec})
1C32:0C	Cycle exceeded counter	Number of occasions the cycle time was exceeded in OPERATIONAL (cycle was not completed in time or the next cycle began too early)	UINT16	RO	0x0000 (0_{dec})
1C32:0D	Shift too short counter	Number of intervals between SYNC0 and SYNC1 events that are too short (DC Mode only)	UINT16	RO	0x0000 (0_{dec})
1C32:20	Sync error	The synchronization was not correct in the last cycle (outputs were output too late; DC Mode only)	BOOLEAN	RO	0x00 (0_{dec})

Index 1C33 SM input parameter

Index (hex)	Name	Meaning	Data type	Flags	Default
1C33:0	SM input parameter	Synchronization parameters for the inputs	UINT8	RO	0x20 (32_{dec})
1C33:01	Sync mode	Current synchronization mode: • 0: Free Run • 1: Synchron with SM 3 Event (no outputs available) • 2: DC - Synchron with SYNC0 Event • 3: DC - Synchron with SYNC1 Event • 34: Synchron with SM 2 Event (outputs available)	UINT16	RW	0x0000 (0_{dec})
1C33:02	Cycle time	• as 0x1C32:02	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: Free Run is supported • Bit 1: Synchron with SM 2 Event is supported (outputs available) • Bit 1: Synchron with SM 3 Event is supported (no outputs available) • Bit 2-3 = 01: DC-Mode is supported • Bit 4-5 = 01: Input shift through local event (outputs available) • Bit 4-5 = 10: Input shift with SYNC1 event (no outputs available) • Bit 14 = 1: dynamic times (measurement through writing of 0x1C32:08 or 0x1C33:08)	UINT16	RO	0x0000 (0_{dec})
1C33:05	Minimum cycle time	as 0x1C32:05	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	Get Cycle Time	as 1C32:08	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 0x1C32:11	UINT16	RO	0x0000 (0_{dec})
1C33:0C	Cycle exceeded counter	as 0x1C32:12	UINT16	RO	0x0000 (0_{dec})
1C33:0D	Shift too short counter	as 0x1C32:13	UINT16	RO	0x0000 (0_{dec})
1C33:20	Sync error	as 0x1C32:32	BOOLEAN	RO	0x00 (0_{dec})

Index F008 Code word

Index (hex)	Name	Meaning	Data type	Flags	Default
F008:0	Code word	Code word (currently reserved)	UINT32	RW	0x00000000 (0_{dec})

Index F009 Password protection

Index (hex)	Name	Meaning	Data type	Flags	Default
F009:0	Password protection	Password protection user calibration	UINT32	RW	0x00000000 (0_{dec})

Index F081 Download revision

Index (hex)	Name	Meaning	Data type	Flags	Default
F081:0	Download revision		UINT8	RO	0x01 (1_{dec})
F081:01	Revision number		UINT32	RW	0x00000000 (0_{dec})

Index 1804 SECC TxPDO-Par Charge Status AC Low Level

Index (hex)	Name	Meaning	Data type	Flags	Default
1806:0	SECC TxPDO-Par Plug and Charge	PDO Parameter TxPDO 7	UINT8	RO	0x06 (6_{dec})
1806:06	Exclude TxPDOs	Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 7	OCTET-STRING[8]	RO	04 1A 00 00 00 00 00 00

Index 1A05 SECC TxPDO-Map Diagnostics Data

Index (hex)	Name	Meaning	Data type	Flags	Default
1A05:0	SECC TxPDO-Map Diagnostics Data	PDO Mapping TxPDO 6	UINT8	RO	0x19 (25_{dec})
1A05:01	SubIndex 001	1. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x01 (Send))	UINT32	RO	0x6003:01, 8
1A05:02	SubIndex 002	2. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8
1A05:03	SubIndex 003	3. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x03 (Receive__Alive Counter))	UINT32	RO	0x6003:02, 152
1A05:04	SubIndex 004	4. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8
1A05:05	SubIndex 005	5. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x03 (Receive__Receive Counter))	UINT32	RO	0x6003:03, 16
1A05:06	SubIndex 006	6. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x06 (Error__Diag Code 0))	UINT32	RO	0x6003:04, 8
1A05:07	SubIndex 007	7. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x07 (Error__Diag Code 1))	UINT32	RO	0x6003:05, 8
1A05:08	SubIndex 008	8. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x08 (Error__Diag Code 2))	UINT32	RO	0x6003:06, 32
1A05:09	SubIndex 009	9. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x09 (Error__Diag Code 3))	UINT32	RO	0x6003:07, 32
1A05:0A	SubIndex 010	10. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x0A (Debug__Pilot States Status))	UINT32	RO	0x6003:08, 32
1A05:0B	SubIndex 011	11. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x0B (Debug__Slac Status))	UINT32	RO	0x6003:09, 32
1A05:0C	SubIndex 012	12. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x0C (Debug__V2G Status DIN))	UINT32	RO	0x6003:0A, 8
1A05:0D	SubIndex 013	13. PDO Mapping entry (8 bits align)	UINT32	RO	0x6003:0B, 8
1A05:0E	SubIndex 014	14. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x0E (Debug__PP Voltage))	UINT32	RO	0x6003:0C, 16
1A05:0F	SubIndex 015	15. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x0F (Debug__CP Voltage))	UINT32	RO	0x6003:0D, 16
1A05:10	SubIndex 016	16. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x10 (Debug__V2G Status DCISO))	UINT32	RO	0x6003:0E, 8
1A05:11	SubIndex 017	17. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x11 (Debug__V2G Status ACISO))	UINT32	RO	0x6003:0F, 8
1A05:12	SubIndex 018	18. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x12 (Debug__Watchdog Reset Cause))	UINT32	RO	0x6003:10, 8
1A05:13	SubIndex 019	19. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x14 (Debug__Watchdog Reset Log 0))	UINT32	RO	0x6003:11, 8
1A05:14	SubIndex 020	20. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x15 (Debug__Watchdog Reset Log 1))	UINT32	RO	0x6003:12, 8
1A05:15	SubIndex 021	21. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x17 (Debug__Watchdog Reset Log 2))	UINT32	RO	0x6003:13, 8
1A05:16	SubIndex 022	22. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x18 (Debug__Watchdog Reset Log 3))	UINT32	RO	0x6003:14, 8
1A05:17	SubIndex 023	23. PDO Mapping entry (8 bits align)	UINT32	RO	0x6003:15, 8
1A05:18	SubIndex 024	24. PDO Mapping entry (object 0x6003 (SECC Diagnostics Data), entry 0x16 (Watchdog__Watchdog Reset Log 3))	UINT32	RO	0x6003:16, 8
1A05:19	SubIndex 025	25. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8

Index 1A06 SECC TxPDO-Map Plug and Charge

Index (hex)	Name	Meaning	Data type	Flags	Default
1A06:0	SECC TxPDO-Map Plug and Charge	PDO Mapping TxPDO 7	UINT8	RO	0x05 (5dec)
1A06:01	SubIndex 001	1. PDO Mapping entry (object 0x6004 (SECC Plug and Charge), entry 0x01 (Certificate Validation Status))	UINT32	RO	0x6004:01, 128
1A06:02	SubIndex 002	2. PDO Mapping entry (object 0x6004 (SECC Plug and Charge), entry 0x02 (Signature Validation Status))	UINT32	RO	0x6004:02, 8
1A06:03	SubIndex 003	3. PDO Mapping entry (object 0x6004 (SECC Plug and Charge), entry 0x03 (EMAID Validation Status))	UINT32	RO	0x6004:03, 8
1A06:04	SubIndex 004	4. PDO Mapping entry (8 bits align)	UINT32	RO	0x6004:04, 8
1A06:05	SubIndex 005	5. PDO Mapping entry (8 bits align)	UINT32	RO	0x0000:00, 8

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 [▸ 126].
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!

EL6761			
Hardware (HW)	**Firmware (FW)**	**Revision no.**	**Release date**
00 – 01*	01	EL6761-0000-0016	2025/04

*) 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:

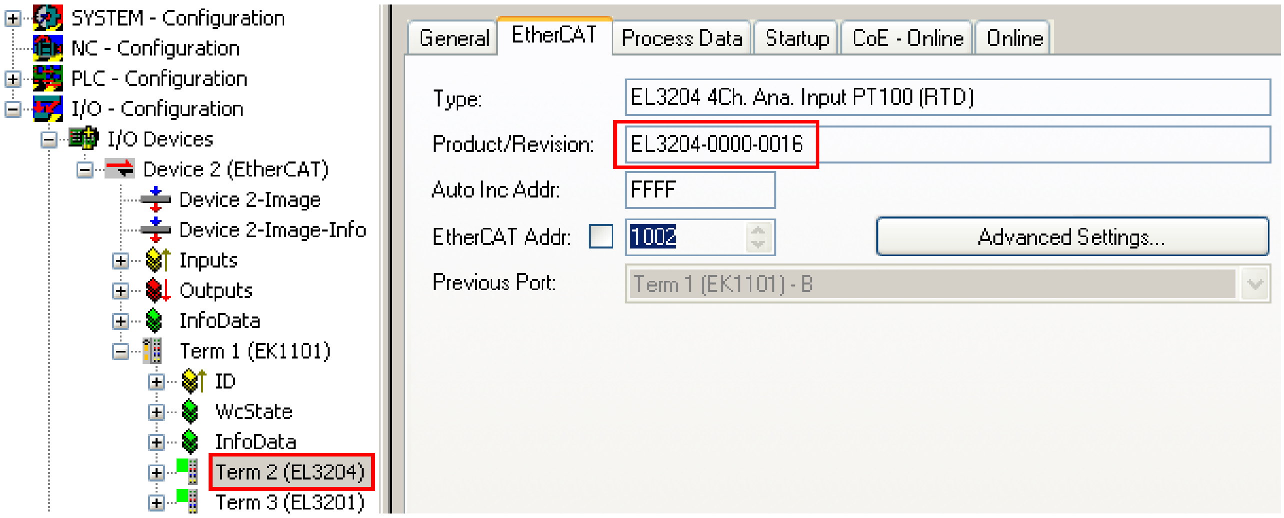


Fig. 96: Device identifier consisting of name EL3204-0000 and revision -0016

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:

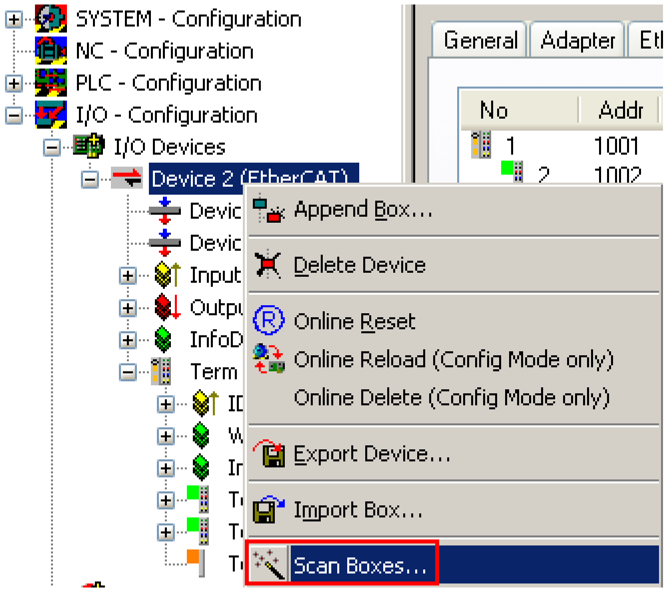


Fig. 97: Scan the subordinate field by right-clicking on the EtherCAT device

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

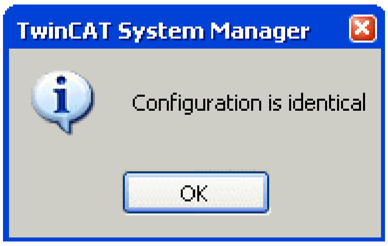


Fig. 98: Configuration is identical

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

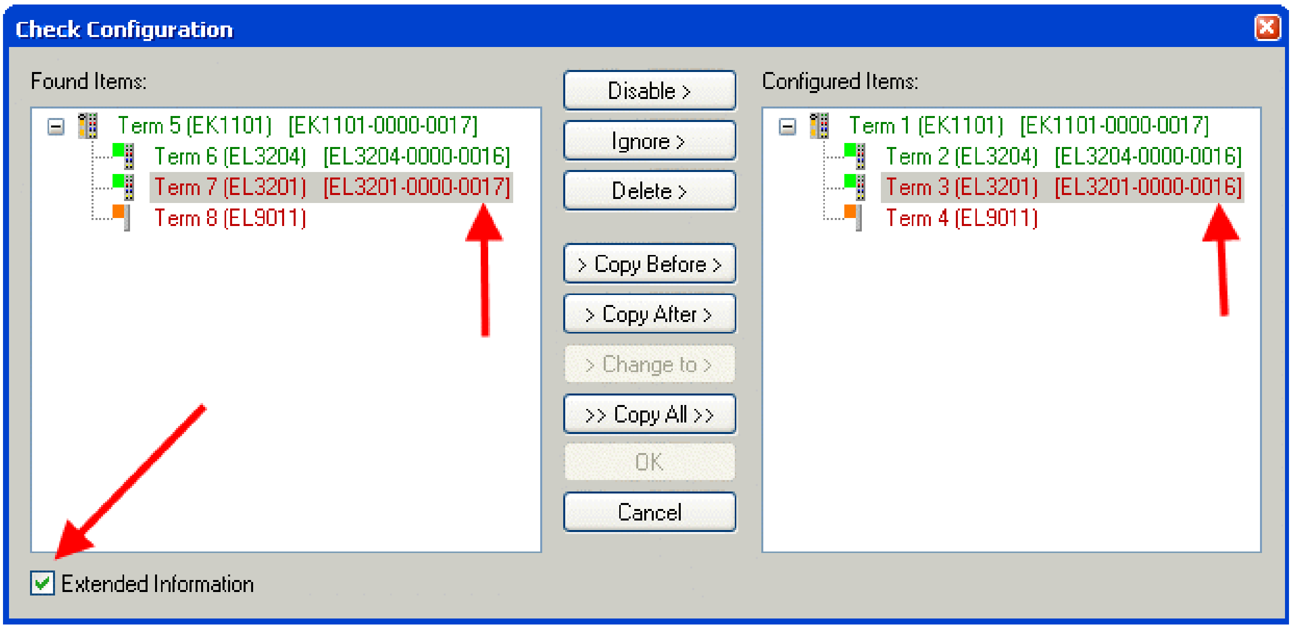


Fig. 99: Change dialog

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*

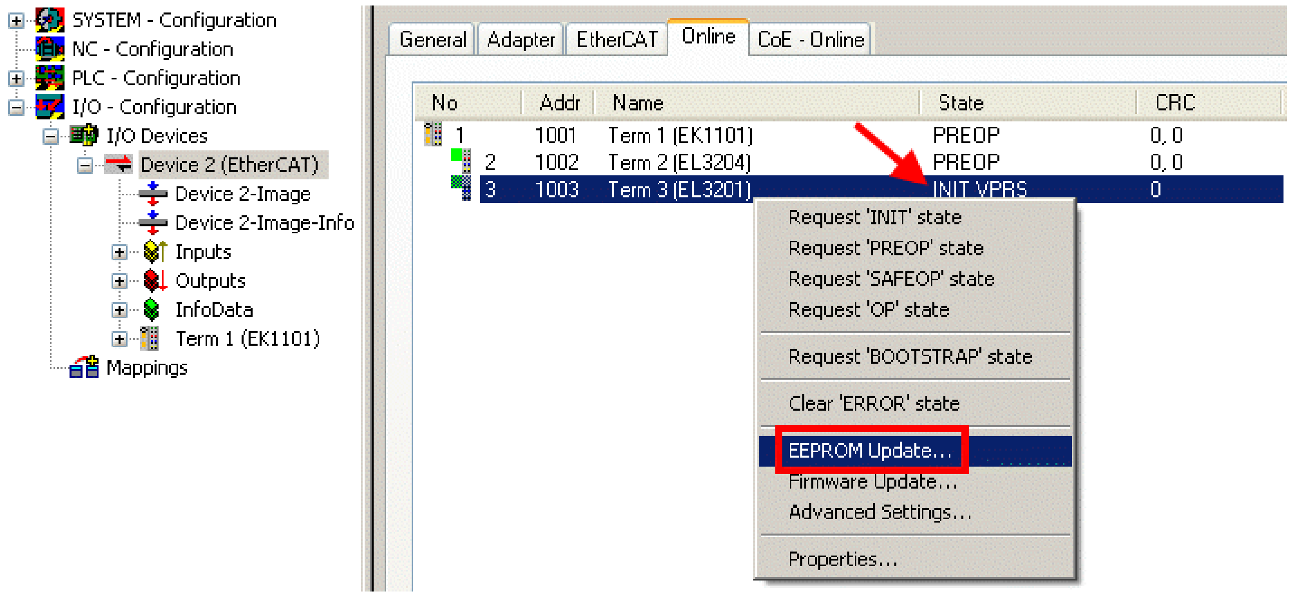


Fig. 100: EEPROM Update

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.

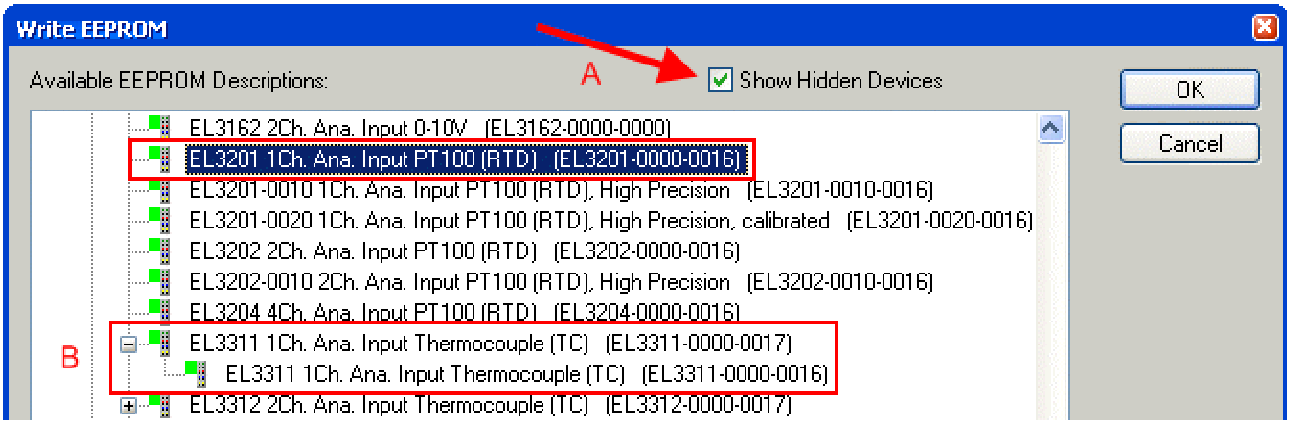


Fig. 101: Selecting the new ESI

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.

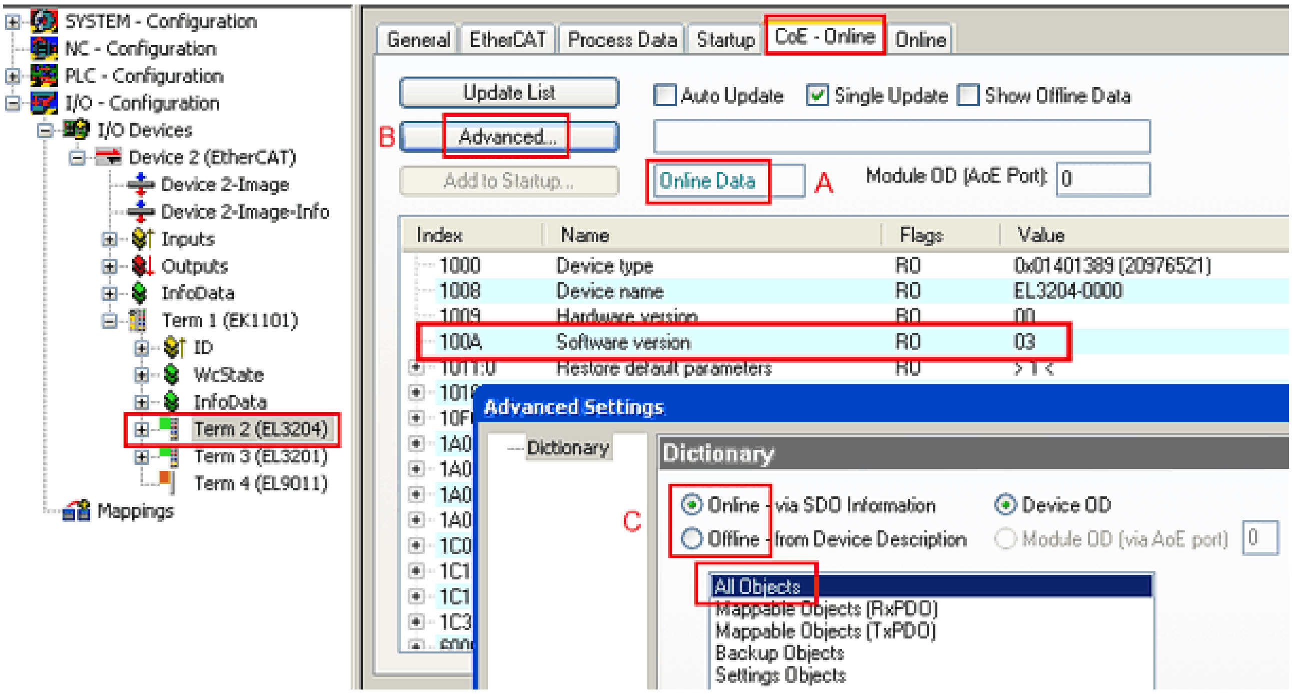


Fig. 102: Display of EL3204 firmware version

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.*

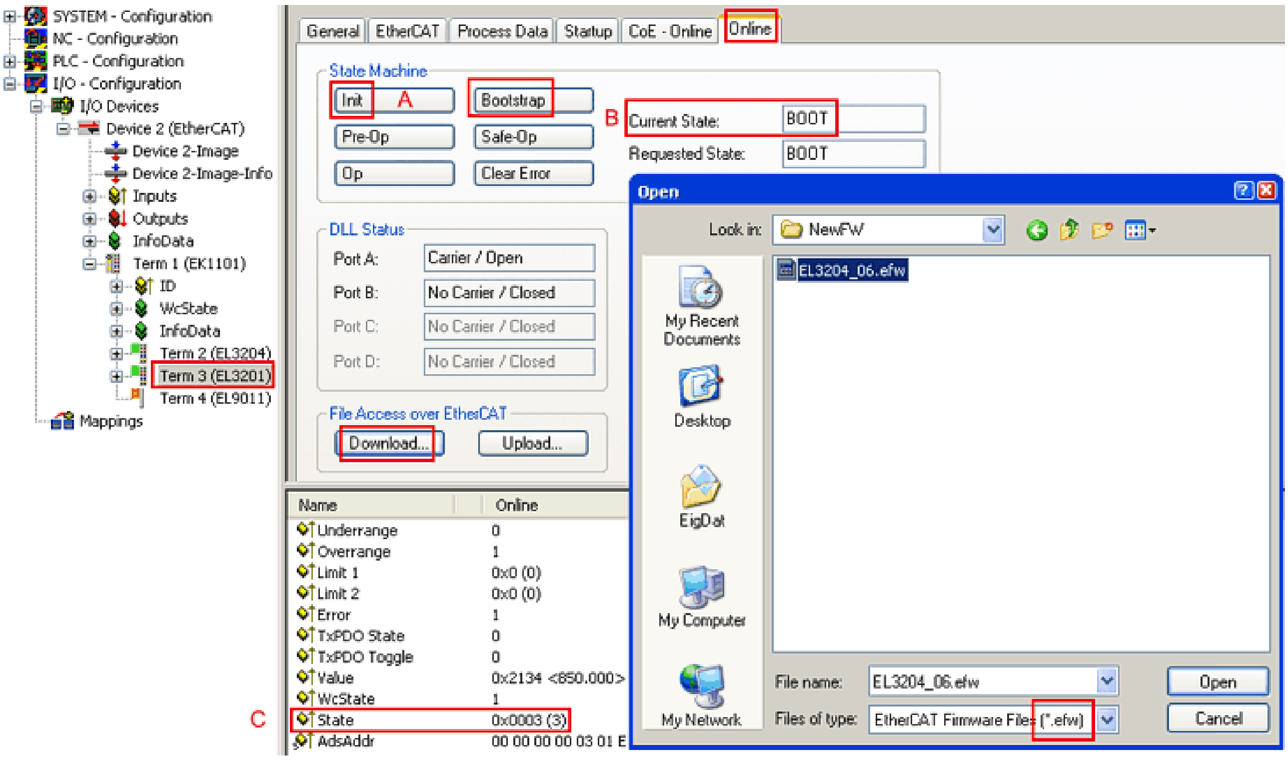
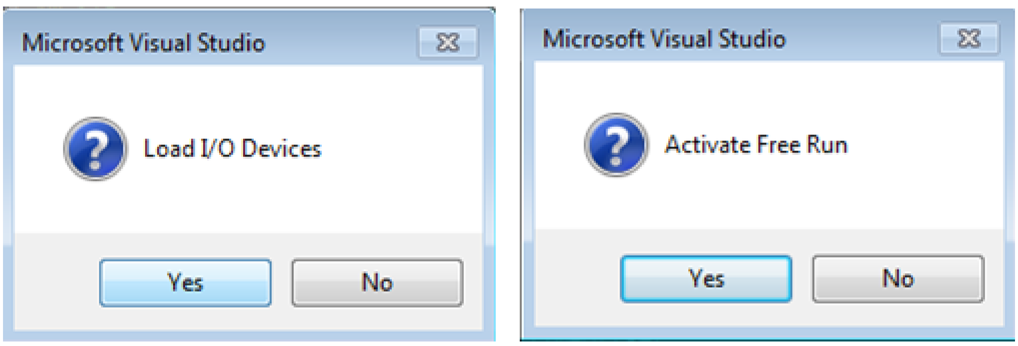


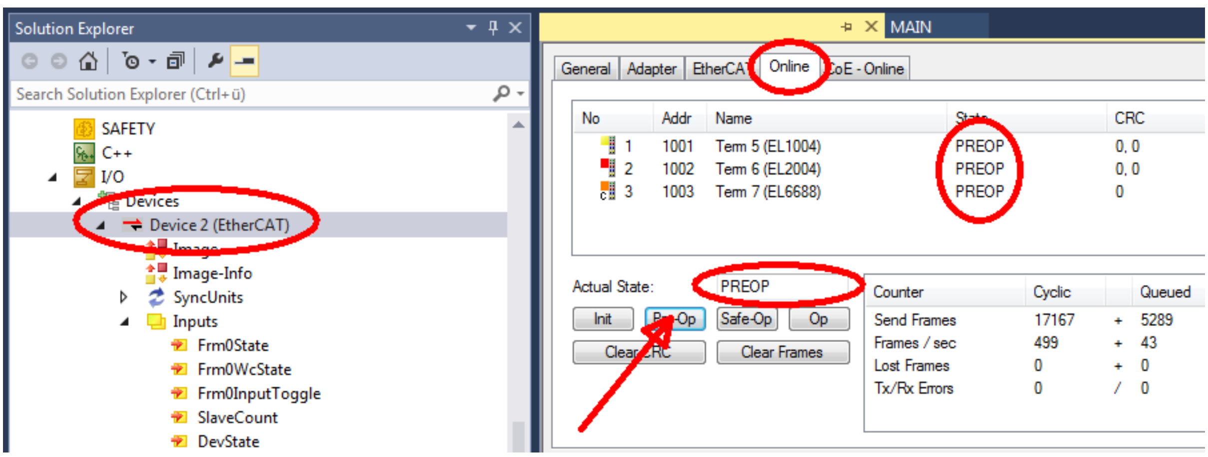
Fig. 103: Firmware Update

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.

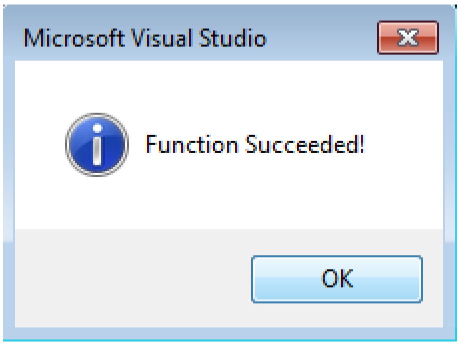


- Switch EtherCAT Master to PreOP



- 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.

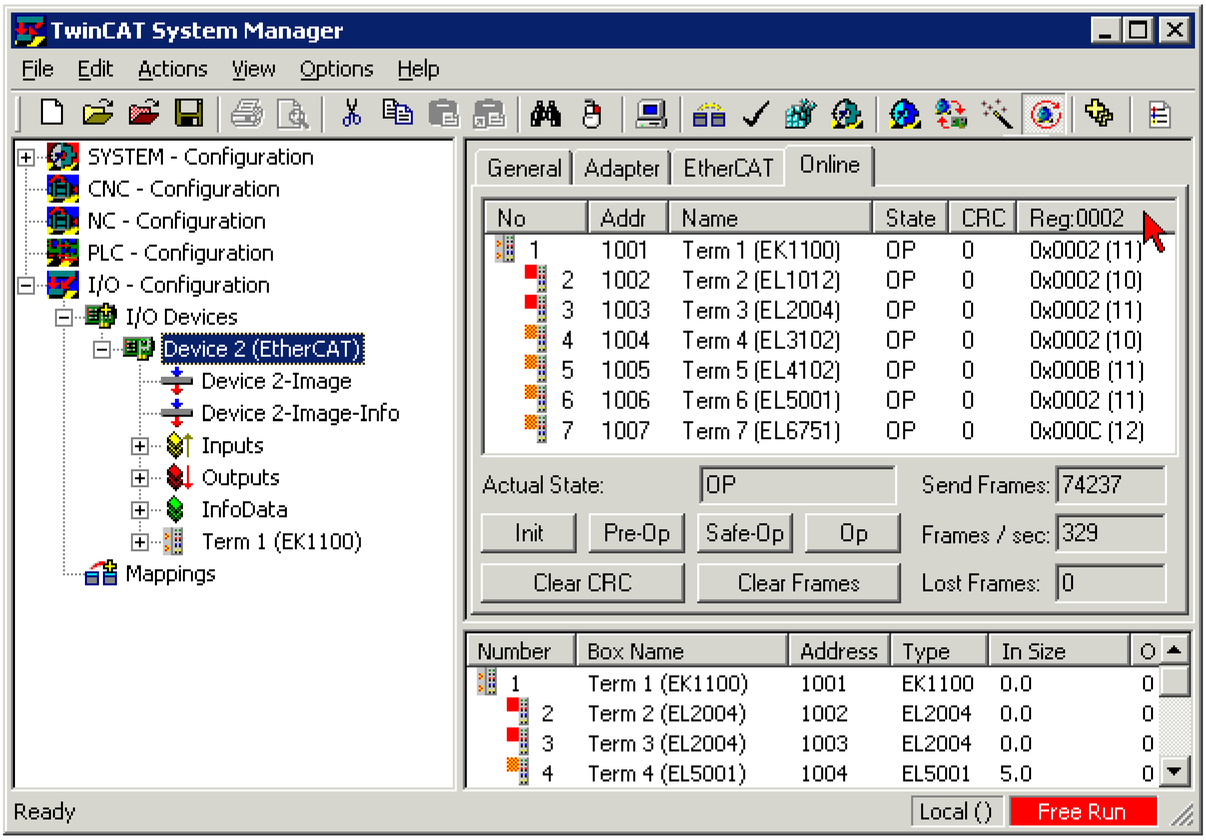


Fig. 104: FPGA firmware version definition

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

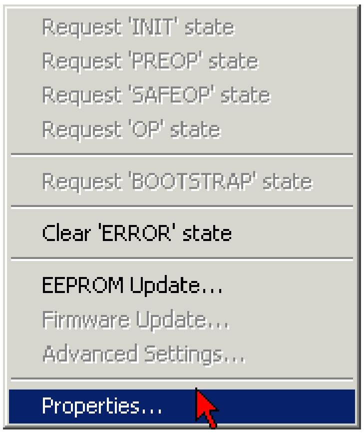


Fig. 105: Context menu *Properties*

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.

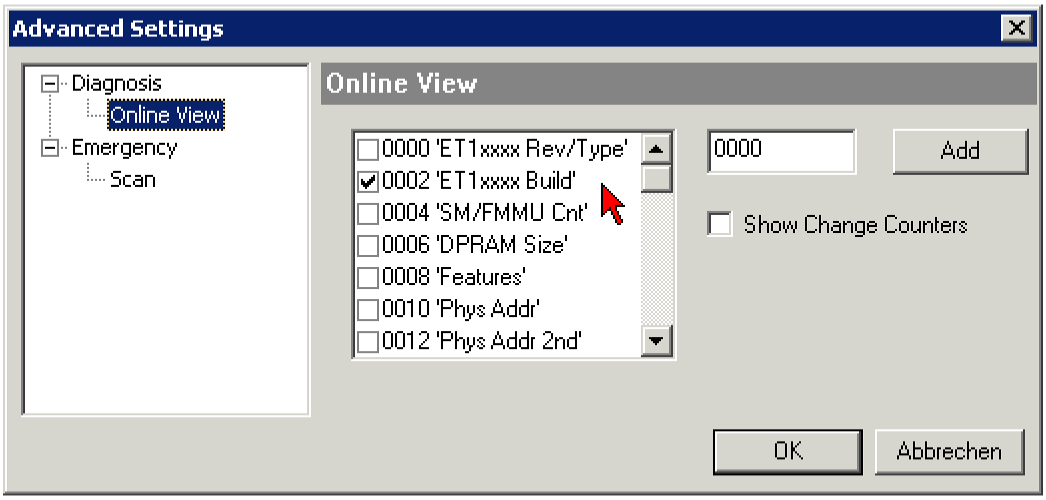


Fig. 106: Dialog *Advanced Settings*

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.

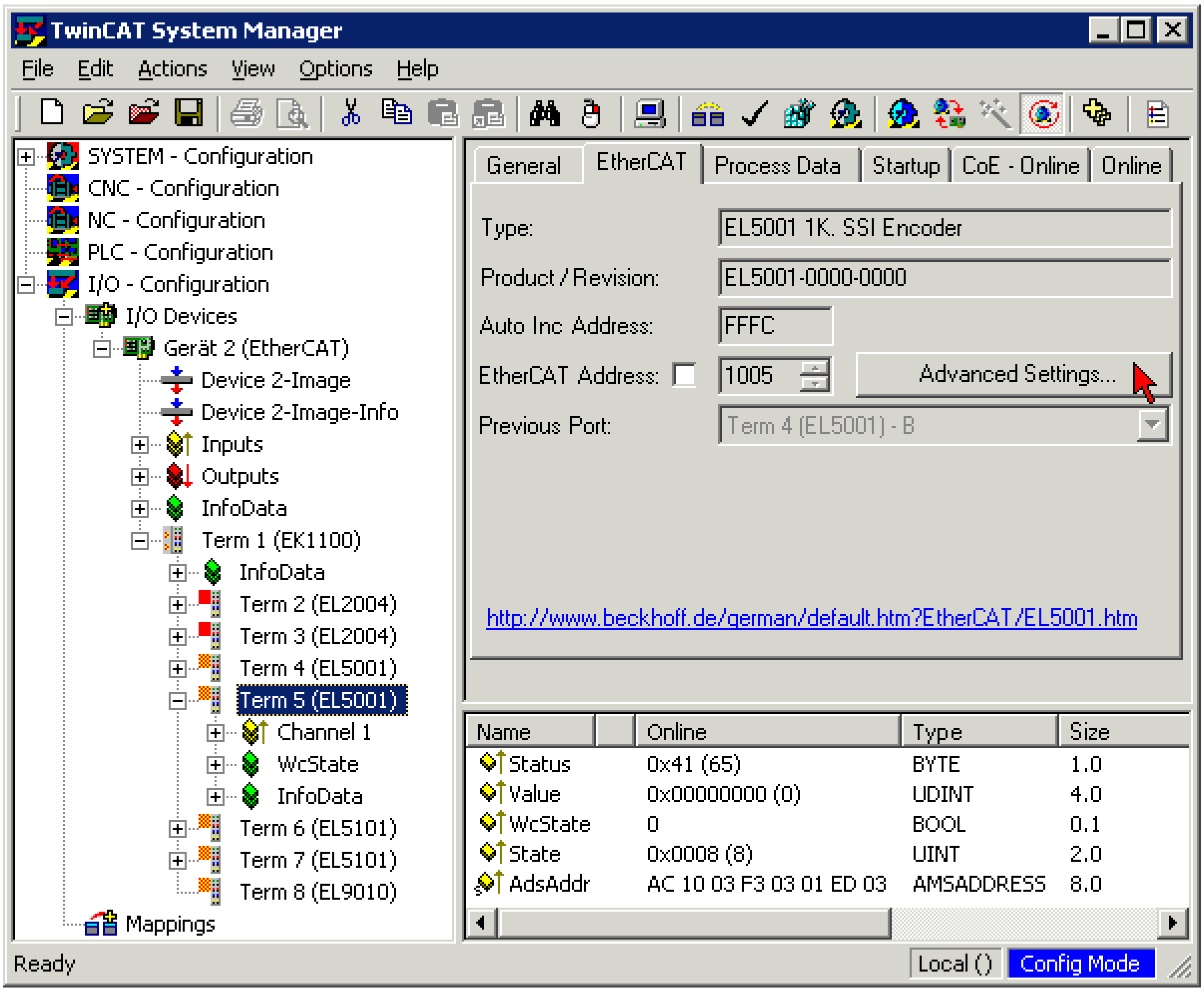
Older firmware versions can only be updated by the manufacturer!

Updating an EtherCAT device

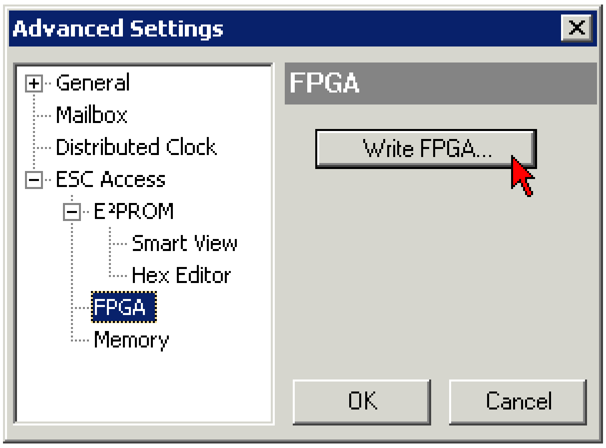
The following sequence order have to be met if no other specifications are given (e.g. by the Beckhoff support):

- 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.

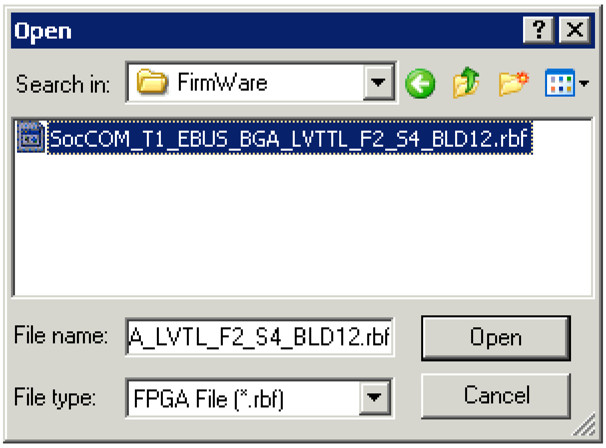
- In the TwinCAT System Manager select the terminal for which the FPGA firmware is to be updated (in the example: Terminal 5: EL5001) and
click the *Advanced Settings* button in the *EtherCAT* tab:



- 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.

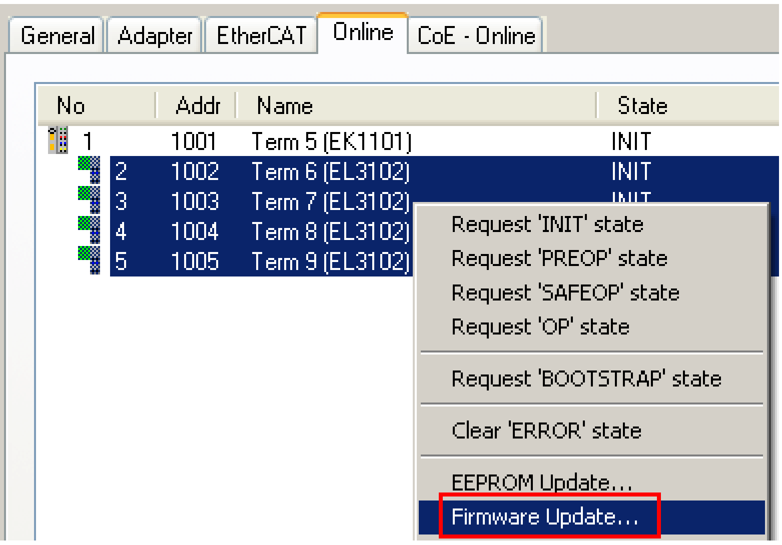


Fig. 107: Multiple selection and firmware update

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 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*).

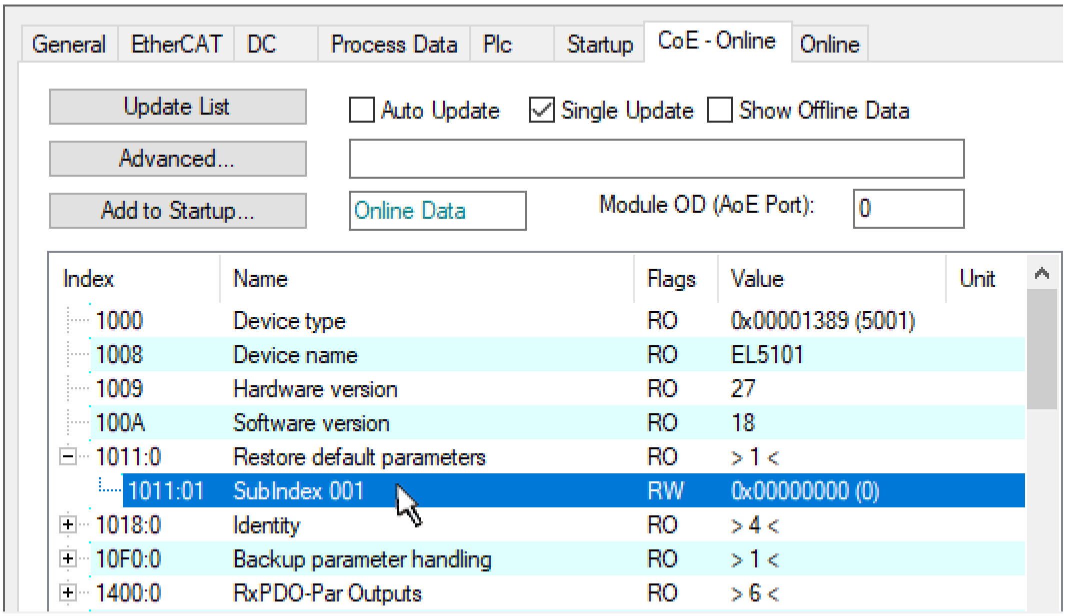


Fig. 108: Selecting the *Restore default parameters* PDO

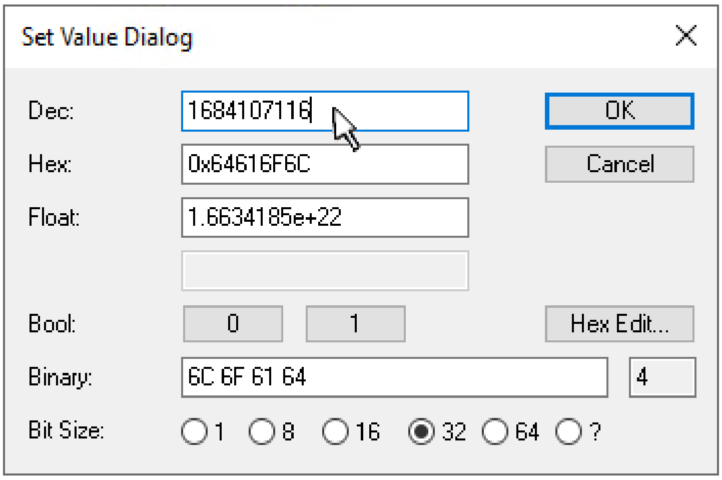


Fig. 109: Entering a restore value in the Set Value dialog

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

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

All backup objects are reset to the delivery state.

Alternative restore value

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

An incorrect entry for the restore value has no effect.

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

Third-party trademark statements

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

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

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

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Limited.

More Information:
www.beckhoff.com/EL6761

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