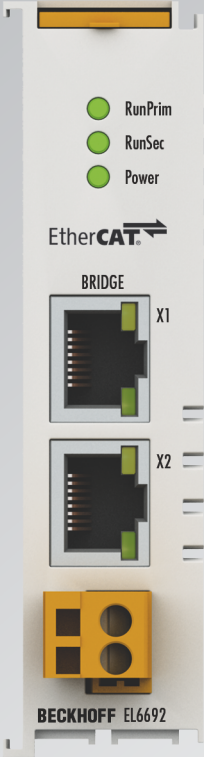


Documentation | EN

# EL6692

EtherCAT Bridge Terminal





# Table of contents

|   |           |
|---|-----------|
| <b>1 Foreword .....</b>   | <b>5</b>  |
| 1.1 Notes on the documentation.....   | 5         |
| 1.2 Safety instructions .....   | 6         |
| 1.3 Documentation issue status .....  | 7         |
| 1.4 Version identification of EtherCAT devices .....                        | 8         |
| 1.4.1 General notes on marking .....  | 8         |
| 1.4.2 Version identification of EL terminals.....                           | 9         |
| 1.4.3 Beckhoff Identification Code (BIC).....                               | 10        |
| 1.4.4 Electronic access to the BIC (eBIC) .....                             | 12        |
| <b>2 Product description.....</b>   | <b>14</b> |
| 2.1 Introduction.....   | 14        |
| 2.2 Technical data .....  | 15        |
| 2.3 Internal and external EtherCAT synchronization .....                    | 16        |
| 2.4 Start .....   | 20        |
| <b>3 Basics communication .....</b>   | <b>21</b> |
| 3.1 EtherCAT basics.....  | 21        |
| 3.2 EtherCAT cabling – wire-bound.....                                      | 21        |
| 3.3 General notes for setting the watchdog.....                             | 22        |
| 3.4 EtherCAT State Machine .....  | 24        |
| 3.5 CoE Interface.....  | 25        |
| 3.6 Distributed Clock .....   | 30        |
| <b>4 Mounting and wiring.....</b>   | <b>31</b> |
| 4.1 Instructions for ESD protection.....                                    | 31        |
| 4.2 Explosion protection .....  | 32        |
| 4.2.1 ATEX - Special conditions (extended temperature range) .....          | 32        |
| 4.2.2 Continuative documentation for ATEX and IECEx .....                   | 33        |
| 4.3 UL notice .....   | 34        |
| 4.4 Recommended mounting rails.....   | 35        |
| 4.5 Mounting and demounting - terminals with traction lever unlocking ..... | 36        |
| 4.6 Mounting and demounting - terminals with front unlocking .....          | 38        |
| 4.7 Positioning of passive Terminals .....                                  | 40        |
| 4.8 Installation positions .....  | 41        |
| 4.9 LEDs and connection .....   | 43        |
| 4.10 Disposal.....  | 45        |
| <b>5 Commissioning.....</b>   | <b>46</b> |
| 5.1 TwinCAT Quick Start .....   | 46        |
| 5.1.1 TwinCAT 2.....  | 49        |
| 5.1.2 TwinCAT 3 .....   | 59        |
| 5.2 TwinCAT Development Environment .....                                   | 73        |
| 5.2.1 Installation of the TwinCAT real-time driver.....                     | 73        |
| 5.2.2 Notes regarding ESI device description.....                           | 79        |
| 5.2.3 TwinCAT ESI Updater .....   | 83        |
| 5.2.4 Distinction between Online and Offline.....                           | 83        |

|          |  |            |
|----------|--|------------|
| 5.2.5    | OFFLINE configuration creation .....                     | 84         |
| 5.2.6    | ONLINE configuration creation .....                      | 89         |
| 5.2.7    | EtherCAT subscriber configuration .....                  | 97         |
| 5.2.8    | Import/Export of EtherCAT devices with SCI and XTI ..... | 106        |
| 5.3      | General Notes - EtherCAT Slave Application .....         | 113        |
| 5.4      | Basic principles of function and commissioning .....     | 121        |
| 5.5      | Handling acyclic data .....                              | 131        |
| 5.6      | External TwinCAT synchronization .....                   | 134        |
| 5.7      | Object description and parameterization .....            | 141        |
| 5.7.1    | Objects for commissioning .....                          | 141        |
| 5.7.2    | Objects for regular operation .....                      | 142        |
| 5.7.3    | Input data .....   | 142        |
| 5.7.4    | Output data .....  | 142        |
| 5.7.5    | Information and diagnostic data .....                    | 142        |
| 5.7.6    | Standard objects (0x1000-0x1FFF) .....                   | 143        |
| <b>6</b> | <b>Appendix .....</b>                                    | <b>148</b> |
| 6.1      | EtherCAT AL Status Codes .....                           | 148        |
| 6.2      | Firmware compatibility .....                             | 149        |
| 6.3      | Firmware Update EL/ES/EM/ELM/EPxxxx .....                | 151        |
| 6.3.1    | Device description ESI file/XML .....                    | 152        |
| 6.3.2    | Firmware explanation .....                               | 155        |
| 6.3.3    | Updating controller firmware *.efw .....                 | 155        |
| 6.3.4    | FPGA firmware *.rbf .....                                | 157        |
| 6.3.5    | Simultaneous updating of several EtherCAT devices .....  | 161        |
| 6.4      | Restoring the delivery state .....                       | 162        |
| 6.5      | Support and Service .....                                | 163        |

# 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.

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

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

### Disclaimer

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

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

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

### Trademarks

Beckhoff®, TwinCAT®, 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.

### Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents: EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702 with corresponding applications or registrations in various other countries.

The logo for EtherCAT, featuring the word "EtherCAT" in a bold, black, sans-serif font. A red arrow points from the top of the "A" towards the right, ending above the "T". A registered trademark symbol (®) is located to the right of the "T".

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

### Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.

The reproduction, distribution and utilization of this document as well as the communication of its contents to others without express authorization are prohibited.

Offenders will be held liable for the payment of damages. All rights reserved in the event of the grant of a patent, utility model or design.

## 1.2 Safety instructions

### Safety regulations

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

### Exclusion of liability

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

### Personnel qualification

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

### Description of instructions

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

#### **DANGER**

##### **Serious risk of injury!**

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

#### **WARNING**

##### **Risk of injury!**

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

#### **CAUTION**

##### **Personal injuries!**

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

#### **NOTE**

##### **Damage to environment/equipment or data loss**

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



##### **Tip or pointer**

This symbol indicates information that contributes to better understanding.

## 1.3 Documentation issue status

| Version | Comment  |
|---------|--|
| 3.4     | <ul style="list-style-type: none"> <li>• Update chapter "Technical data"</li> <li>• Update chapter "Handling of acyclic data"</li> <li>• Update structure</li> </ul>   |
| 3.3     | <ul style="list-style-type: none"> <li>• Update chapter "Technical data"</li> <li>• Update chapter "Version identification of EtherCAT devices"</li> <li>• Update structure</li> <li>• Update notes</li> <li>• Update revision status</li> <li>• Chapter Disposal added</li> </ul>   |
| 3.3     | <ul style="list-style-type: none"> <li>• Update chapter "Technical data"</li> <li>• Update chapter "Version identification of EtherCAT devices"</li> <li>• Update structure</li> <li>• Update notes</li> <li>• Update revision status</li> <li>• Chapter Disposal added</li> </ul>   |
| 3.2     | <ul style="list-style-type: none"> <li>• Update chapter "UL notice"</li> <li>• Update chapter "Firmware compatibility"</li> <li>• Update chapter "Technical data"</li> <li>• Update structure</li> </ul>   |
| 3.1     | <ul style="list-style-type: none"> <li>• Update chapter "Notes on the documentation"</li> <li>• Correction of Technical data</li> <li>• Update chapter "TwinCAT 2.1x" -&gt; "TwinCAT Development Environment" and "TwinCAT Quick Start"</li> <li>• Update revision status</li> </ul> |
| 3.0     | <ul style="list-style-type: none"> <li>• Migration</li> <li>• Update revision status</li> <li>• Structural update</li> </ul>   |
| 2.1     | <ul style="list-style-type: none"> <li>• Structural update</li> <li>• "Technical data" section updated</li> </ul>  |
| 2.0     | <ul style="list-style-type: none"> <li>• Update chapter "Basic principles of function and commissioning"</li> </ul>  |
| 1.9     | <ul style="list-style-type: none"> <li>• Update chapter "External TwinCAT synchronization"</li> </ul>  |
| 1.8     | <ul style="list-style-type: none"> <li>• Firmware compatibility list and technical data updated</li> </ul>   |
| 1.7     | <ul style="list-style-type: none"> <li>• Firmware compatibility list and technical notes added</li> </ul>  |
| 1.6     | <ul style="list-style-type: none"> <li>• Firmware compatibility list and technical notes added</li> </ul>  |
| 1.5     | <ul style="list-style-type: none"> <li>• Samples of external synchronization added</li> </ul>  |
| 1.4     | <ul style="list-style-type: none"> <li>• Chapter AoE, Eoe added</li> </ul>   |
| 1.3     | <ul style="list-style-type: none"> <li>• Technical notes added</li> </ul>  |
| 1.2     | <ul style="list-style-type: none"> <li>• Notes on synchronization added</li> </ul>   |
| 1.1     | <ul style="list-style-type: none"> <li>• Current screenshots added</li> </ul>  |
| 1.0     | <ul style="list-style-type: none"> <li>• Addendum description of cyclic protocols, initial publication</li> </ul>  |

## 1.4 Version identification of EtherCAT devices

### 1.4.1 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<br>(12 mm, non-pluggable connection level) | 3314 (4-channel thermocouple terminal) | 0000 (basic type)             | 0016     |
| ES3602-0010-0017 | ES terminal<br>(12 mm, pluggable connection level)     | 3602 (2-channel voltage measurement)   | 0010 (high-precision version) | 0017     |
| CU2008-0000-0000 | CU device  | 2008 (8-port fast ethernet switch)     | 0000 (basic type)             | 0000     |

#### Notes

- The elements mentioned above result in the **technical designation**. EL3314-0000-0016 is used in the example below.
- EL3314-0000 is the order identifier, in the case of “-0000” usually abbreviated to EL3314. “-0016” is the EtherCAT revision.
- The **order identifier** is made up of
  - family key (EL, EP, CU, ES, KL, CX, etc.)
  - type (3314)
  - version (-0000)
- The **revision** -0016 shows the technical progress, such as the extension of features with regard to the EtherCAT communication, and is managed by Beckhoff.  
In principle, a device with a higher revision can replace a device with a lower revision, unless specified otherwise, e.g. in the documentation.  
Associated and synonymous with each revision there is usually a description (ESI, EtherCAT Slave Information) in the form of an XML file, which is available for download from the Beckhoff web site.  
From 2014/01 the revision is shown on the outside of the IP20 terminals, see Fig. “EL5021 EL terminal, standard IP20 IO device with batch number and revision ID (since 2014/01)”.
- The type, version and revision are read as decimal numbers, even if they are technically saved in hexadecimal.



## 1.4.2 Version identification of EL terminals

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

Structure of the serial number: **KK YY FF HH**

KK - week of production (CW, calendar week)

YY - year of production

FF - firmware version

HH - hardware version

Example with serial number 12 06 3A 02:

12 - production week 12

06 - production year 2006

3A - firmware version 3A

02 - hardware version 02



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

### 1.4.3 Beckhoff Identification Code (BIC)

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

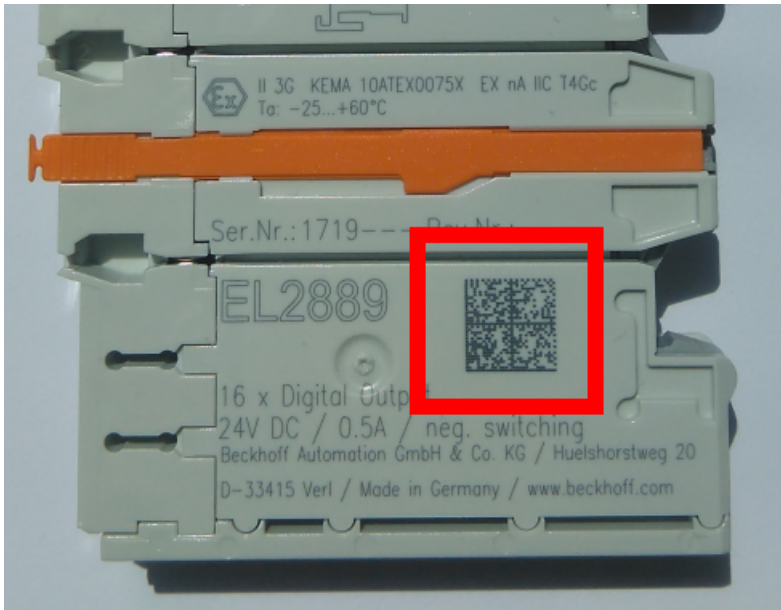


Fig. 2: BIC as data matrix code (DMC, code scheme ECC200)

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

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

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

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

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

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

| Position | Type of information                | Explanation   | Data identifier | Number of digits incl. data identifier | Example                 |
|----------|------------------------------------|---|-----------------|--|-------------------------|
| 1        | Beckhoff order number              | <b>Beckhoff order number</b>  | 1P              | 8                                      | <b>1P</b> 072222        |
| 2        | Beckhoff Traceability Number (BTN) | <b>Unique serial number, see note below</b>                           | SBTN            | 12                                     | <b>SBTN</b> k4p562d7    |
| 3        | Article description                | <b>Beckhoff article description, e.g. EL1008</b>                      | 1K              | 32                                     | <b>1K</b> EL1809        |
| 4        | Quantity                           | <b>Quantity in packaging unit, e.g. 1, 10, etc.</b>                   | Q               | 6                                      | <b>Q1</b>               |
| 5        | Batch number                       | Optional: Year and week of production                                 | 2P              | 14                                     | <b>2P</b> 401503180016  |
| 6        | ID/serial number                   | Optional: Present-day serial number system, e.g. with safety products | 51S             | 12                                     | <b>51S</b> 678294       |
| 7        | Variant number                     | Optional: Product variant number on the basis of standard products    | 30P             | 32                                     | <b>30P</b> 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:

**1P**072222**SBTN**k4p562d7**1K**EL1809 **Q1** **51S**678294

Accordingly as DMC:



Fig. 3: Example DMC **1P**072222**SBTN**k4p562d7**1K**EL1809 **Q1** **51S**678294

**BTN**

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

**NOTE**

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 information.

## 1.4.4 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.

Decisive for the electronic readout is the interface via which the product can be electronically addressed.

### K-bus devices (IP20, IP67)

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

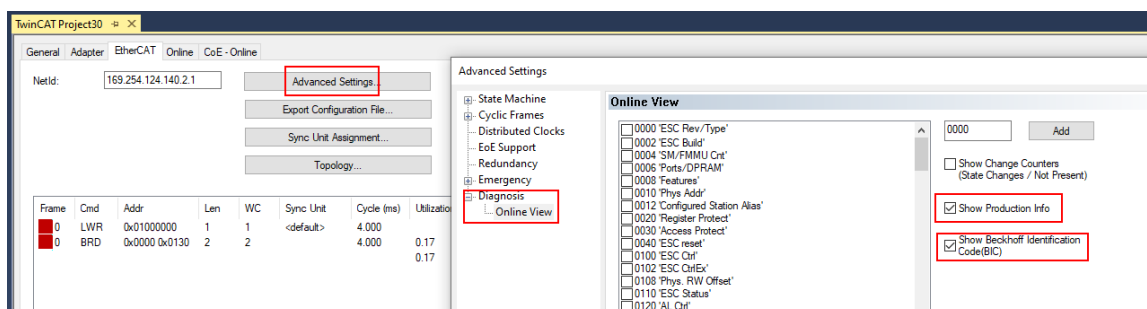
### EtherCAT devices (IP20, IP67)

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

The eBIC is also stored in the ESI-EEPROM. The eBIC was introduced into the Beckhoff I/O production (terminals, boxes) from 2020; widespread implementation is expected in 2021.

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

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



- The BTN and its contents are then displayed:

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

- Note: as can be seen in the illustration, the production data HW version, FW version and production date, which have been programmed since 2012, can also be displayed with "Show Production Info".
- In the case of EtherCAT devices with CoE directory, the object 0x10E2:01 can additionally be used to display the device's own eBIC; the PLC can also simply access the information here:

- The device must be in 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    | 1P158442SBTN0008jepk1KELM3704 Q1 2P482001000016 |
| 10F0:0  | Backup parameter handling                 | RO    | > 1 <   |
| 10F3:0  | Diagnosis History                         | RO    | > 21 <  |
| 10F8    | Actual Time Stamp                         | RO    | 0x170fb277e                                     |

- the object 0x10E2 will be introduced into stock products in the course of a necessary firmware revision.
- Note: in the case of electronic further 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 additionally written as a category in the ESI-EEPROM during the device production. The structure of the ESI content is largely dictated by the ETG specifications, therefore the additional vendor-specific content is stored with the help of a category according to ETG.2010. ID 03 indicates to all EtherCAT masters that they must not overwrite these data in case of an update or restore the data after an ESI update.  
The structure follows the content of the BIC, see there. This results in a memory requirement of approx. 50..200 bytes in the EEPROM.
- 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 with 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/DeviceNet... Devices**

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

## 2 Product description

### 2.1 Introduction

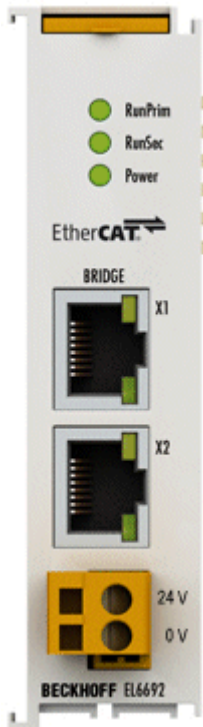


Fig. 4: EL6692

#### EtherCAT bridge terminal with distributed clocks support

The EtherCAT bridge terminal enables data exchange between EtherCAT strands and different masters. It also enables synchronization of the distributed clocks of the individual strands. The power supply on the primary side (E-Bus) comes from the E-Bus, on the secondary side (RJ 45) via an external connection. If several bridge terminals are used, the data traffic continues if the power supply to a device fails. The full functionality of the distributed clocks synchronization is provided from TwinCAT version 2.11.

#### Quick links

- [EtherCAT basics](#)
- [Commissioning \[▶ 46\]](#)
- [Basic function principles \[▶ 121\]](#)
- [CoE object description \[▶ 141\]](#)

## 2.2 Technical data

| Technical data  | EL6692   |
|---|--|
| Connection technology   | Primary side: E-bus<br>Secondary side: RJ45  |
| Width in the process image  | max. 480 bytes in each direction   |
| Data transport time from one EtherCAT side to the opposite side   | typical 1..4 ms  |
| Data transport method   | Asynchronous to the fieldbus   |
| Protocols supported   | Cyclic process data AoE, EoE   |
| <u>Support for distributed clocks synchronization</u><br><a href="#">[▶ 121]</a>  | yes (from TwinCAT 2.11)  |
| Supply voltage for electronics  | via the E-bus  |
| Current consumption from the E-bus - supply, primary side   | typ. 120 mA  |
| Current consumption 24 V - supply, secondary side   | typ. 60 mA   |
| Electrical isolation  | 500 V (E-bus/field voltage)  |
| Configuration   | via TwinCAT System Manager   |
| Weight  | approx. 70 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 humidity   | 95 %, no condensation  |
| Dimensions (W x H x D)  | approx. 26 mm x 100 mm x 52 mm (width aligned: 23 mm)                                      |
| Mounting (housing with <u>front unlocking</u> <a href="#">[▶ 38]</a> / <u>traction lever unlocking</u> <a href="#">[▶ 36]</a> ) | on 35 mm mounting rail conforms to EN 60715  |
| Vibration/shock resistance  | conforms to EN 60068-2-6 / EN 60068-2-27   |
| EMC immunity/emission   | conforms to EN 61000-6-2 / EN 61000-6-4  |
| Protection class  | IP20   |
| Installation position   | variable   |
| Marking / Approval <sup>*)</sup>  | CE, UKCA, EAC<br><u>ATEX</u> <a href="#">[▶ 32]</a><br><u>cULus</u> <a href="#">[▶ 34]</a> |

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

### Ex marking

| Standard | Marking                |
|----------|------------------------|
| ATEX     | II 3 G Ex nA IIC T4 Gc |

## 2.3 Internal and external EtherCAT synchronization

In a machine control with distributed components (I/O, drives, several masters) it may be useful for the components to operate with a close time link to each other. The components must therefore have a local “time”, to which the component (e.g. an I/O terminal) has access at all times.

Associated requirements may include:

1. Several outputs in a control system have to be set simultaneously, irrespective of when the respective station receives the output data.
2. Drives/axes in a control system must read their axis positions synchronized, irrespective of the topology or cycle time.

Both requirements necessitate a synchronization mechanism between the local times of the components of a control system.

3. If inputs affect the control system, the (absolute) time must be recorded. This can be helpful for subsequent analysis, if such an analysis is required for determining the sequence of events in functional chains.

This means that time running in the components must be coupled to a globally valid time, e.g. Greenwich world time or a network clock.

4. tasks on different controllers should run synchronous and without phase shift.

The terms “close temporal reference” or “simultaneous” can be interpreted depending on requirements: for a “simultaneity” in the 10 ms range a serial communication structure may be adequate, while in other ranges 100 ns or less are required.

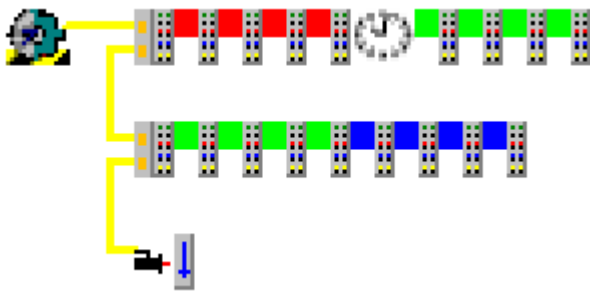


Fig. 5: Simple I/O topology

Fig. *Simple I/O topology* shows a simple EtherCAT topology consisting of a master, several I/Os and an axis. A local time is to be applied in different components. The tasks

- synchronization of local clocks
- coupling to a higher-level reference time
- task synchronization

are discussed below.

### Requirement 1 + 2: synchronization

In an EtherCAT system the distributed clocks concept (DC) is used for synchronization of local clocks in the EtherCAT components. Further information can be found in the separate documentation.

### Synchronization of local EtherCAT devices

General:

- 1 ns time resolution corresponds to 1 digit, scope of 64 bits corresponds to approx. 584 years
- The EtherCAT master must keep the distributed clocks synchronous within the system accuracy (EtherCAT: <100 ns) using synchronization datagrams.



- Not all EtherCAT devices have to support this feature. If a slave does not support this concept, the master will not include it in the synchronization. If the EtherCAT master does not support this feature, DC is also ineffective in all slaves.
- such a clock also runs in the EtherCAT master, in this case software-based.
- In the system *one* of the existing clocks is selected as reference clock and used for synchronizing all other clocks. This reference clock is usually one of the EtherCAT slave clocks, not the EtherCAT master clock. The clock of the first EtherCAT slave in the topology that supports distributed clocks is usually automatically selected as reference clock.
- A distinction is made between
  - the EtherCAT master (the software that “manages” the EtherCAT slaves with Ethernet frames) and the EtherCAT slaves managed by it.
  - the reference clock, which is usually located in the first DC slave, and the slave clocks whose time is based on it, including the clock in the EtherCAT master.

Master:

- During the system start phase the EtherCAT master must set the local time of the reference clock and the other slave clocks to the current time and subsequently minimize deviations between the clocks through cyclic synchronization datagrams.
- In the event of topology changes the EtherCAT master must re-synchronize the clocks accordingly.
- Not all EtherCAT masters support this procedure.
- The EtherCAT master in the Beckhoff TwinCAT automation suite fully supports distributed clocks.

Slave:

- Due to the high precision required this local clock is implemented in hardware (ASIC, FPGA).
- Distributed clocks are managed in the EtherCAT slave controller (ESC) in registers 0x0900 - 0x09FF. Specifically, the local synchronized time runs in the 8 byte from 0x0910.

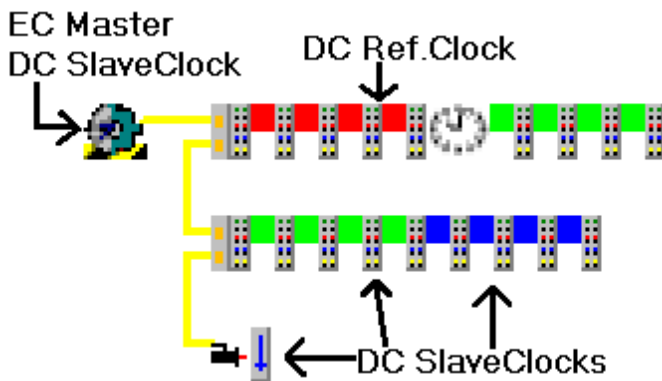


Fig. 6: mapping of DC to the topology

In Fig. *Mapping of DC to the topology* the 3<sup>rd</sup> EtherCAT slave was selected as DC reference clock as a sample. The local time of this slave is now used to adjust all other distributed clocks, i.e. all other EtherCAT slaves and the clock of the EtherCAT master. This is achieved through synchronization datagrams, which the EtherCAT master sends cyclically.

This procedure ensures that all devices supporting DC always have local access to a time that is identical (within the DC synchronization precision) in all devices.

The system now operates based on the timebase of the selected DC reference clock and its local clock generator/quartz with  $T_{DC}$ . Due to production tolerances this timebase is rarely the same as the official sidereal time/coordinated world time UTC  $T_{UTC}$  or another reference time. This means that  $1\text{ ms}_{UTC}$  is never exactly  $1\text{ ms}_{DC}$ ,  $T_{DC} \neq T_{UTC}$ . Over longer periods also drift processes may also change the ratio. As long as DC is used for relative processes within the EtherCAT system, the deviation from the UTC is irrelevant. However, if the DC time is to be used for data logging with a global timebase, for sample, the  $_{DC}$  timebase must be synchronized with the  $_{UTC}$  timebase. This is described in section Requirement 3.

### Requirement 3: higher-level global time, absolute time

If the timebase  $T_{DC}$  is to be adjusted based on a higher-level timebase, the timebase and the associated procedure must be selected. Generally common synchronization protocols are used for the synchronization. Samples for time sources and synchronization procedures are listed below.

- Sources: UTC world time, network time, adjacent control system, radio clocks (in Central Europe: DCF77)
- Procedures: GPS, radio clocks, NTP (NetworkTimeProtocol), SNTP (Simple NTP), PTP (IEEE1588), distributed clocks DC

The following synchronization precisions can be achieved (depending on the hardware)

- NTP/SNTP: ms range
- PTP:  $< 1 \mu\text{s}$
- DC:  $< 100 \text{ ns}$

The following two control aims must be achieved:

- The frequency of the subordinate timebase must be adjusted to the higher-level timebase.
- Any offset between two absolute times does not have necessarily to be controlled to 0. It is sufficient for it to be announced and kept constant. The maximum offset adjustment is  $\pm\frac{1}{2}$  cycle time.

### External EtherCAT synchronization

**i** External synchronization sources (e.g. EL6688, EL6692) can only be used from TwinCAT 2.11 used. In older versions of TwinCAT such EtherCAT slaves have no meaningful function.

If a higher-level master clock is integrated in an EtherCAT system, a special EtherCAT device is generally used for the physical connection. The device monitors both timebases and is therefore able to determine the time difference.

Please refer to [www.beckhoff.de](http://www.beckhoff.de) for suitable products currently available for this purpose.

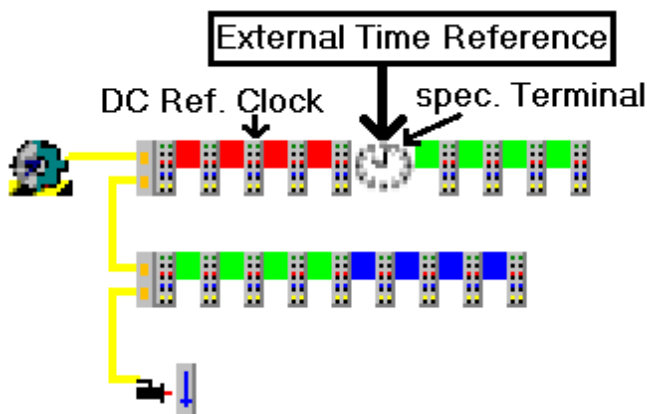


Fig. 7: EtherCAT topology with external reference clock

The different timebases can be arranged hierarchically, so that at the start of the respective system the current absolute time is taken from the subordinate system. If necessary top-down synchronization is used, if external timebases or DC components are present in the system.

### Readjustment of local time vs. higher-level absolute time

For the purpose of synchronization the local DC time is not adjusted based on the higher-level absolute time, but only to a constant offset. This offset is made available to the user as a process data. The offset is corrected by  $\pm\frac{1}{2}$  cycle time to ensure both tasks run in phase.

- When TwinCAT starts the EtherCAT master, the local DC system in the slaves is started and synchronized immediately.

- However, an external reference slave such as EL6688 (IEEE1588 PTP) takes a few seconds before it can supply a reference time that is synchronized with the higher-level clock.
- As soon as the external reference time is available, the offset to the local time is calculated and corrected by  $\pm\frac{1}{2}$  cycle time to ensure that both tasks run in phase, and the EtherCAT master Info Data are made available to the user for reconciliation with the local time values.
- From this time the offset is kept constant, depending on the selected control direction.

## TwinCAT system behavior

### External reference clock outage

If the external reference clock signal fails, both timebases will naturally drift apart again. Once the signal is available again, the system will once again be controlled based on the previous offset values.

TwinCAT can start without external clock signal. In this case the offset is calculated and maintained as described above, as soon as a stable external reference clock signal is received.

### Settings in TwinCAT 2.11

External synchronization via EtherCAT is supported from TwinCAT 2.11. The synchronization direction can be set in the associated dialog.

### Distributed clock timing settings

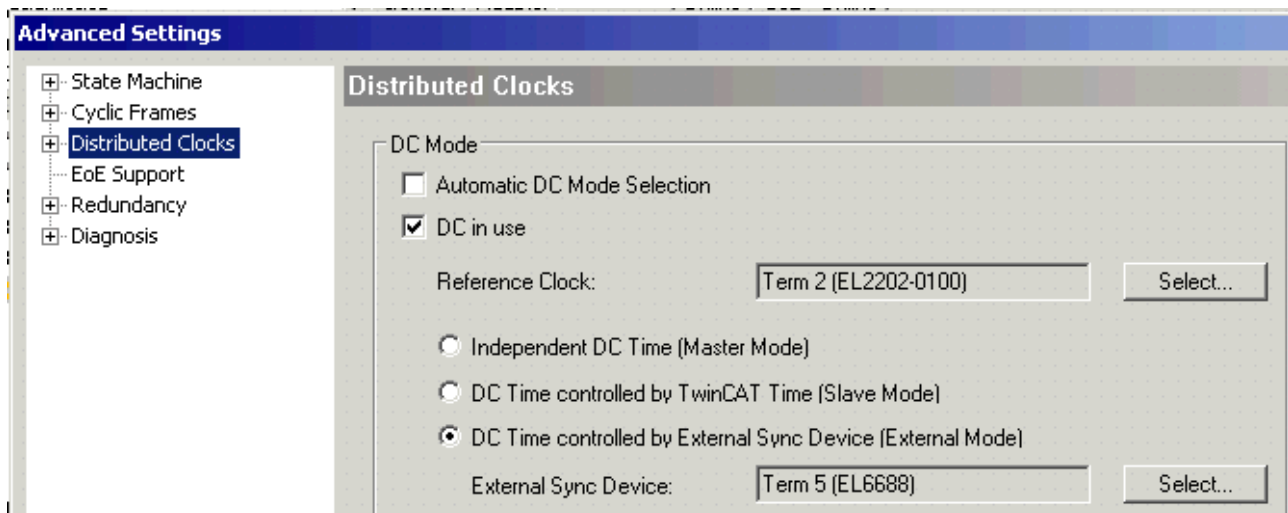


Fig. 8: TwinCAT 2.11 distributed clock settings - Sample for EL6688 in PTP slave mode as time reference for the local EtherCAT system

- **Independent DC Time:** one of the EL terminals (generally the first terminal supporting DC) is the reference clock to which all other DC terminals are adjusted. Selection of the reference clock in the dialog above.
- **DC Time controlled by TwinCAT:** the DC reference clock is adjusted to the local TwinCAT time.
- **DC Time controlled by External Sync Device:** if the EtherCAT system is to be adjusted to a higher-level clock, the external sync device can be selected here.

### Process data settings

TwinCAT 2.11 can display the current offsets in [ns] in the EtherCAT master info data.

- These offsets are calculated once after EtherCAT has started.
- The synchronization control keeps these offsets constant.
- If local DC time values in the synchronized EtherCAT system are to be related to the absolute reference from the higher-level EtherCAT system (e.g. from EL1252 timestamp terminals), the user must adjust this offset with each local timestamp.

Sample:  $t_{\text{EL1252 timestamp channel 1, absolute time}} = t_{\text{EL1252 timestamp channel 1, local DC time}} + t_{\text{ExtToDcOffset}}$

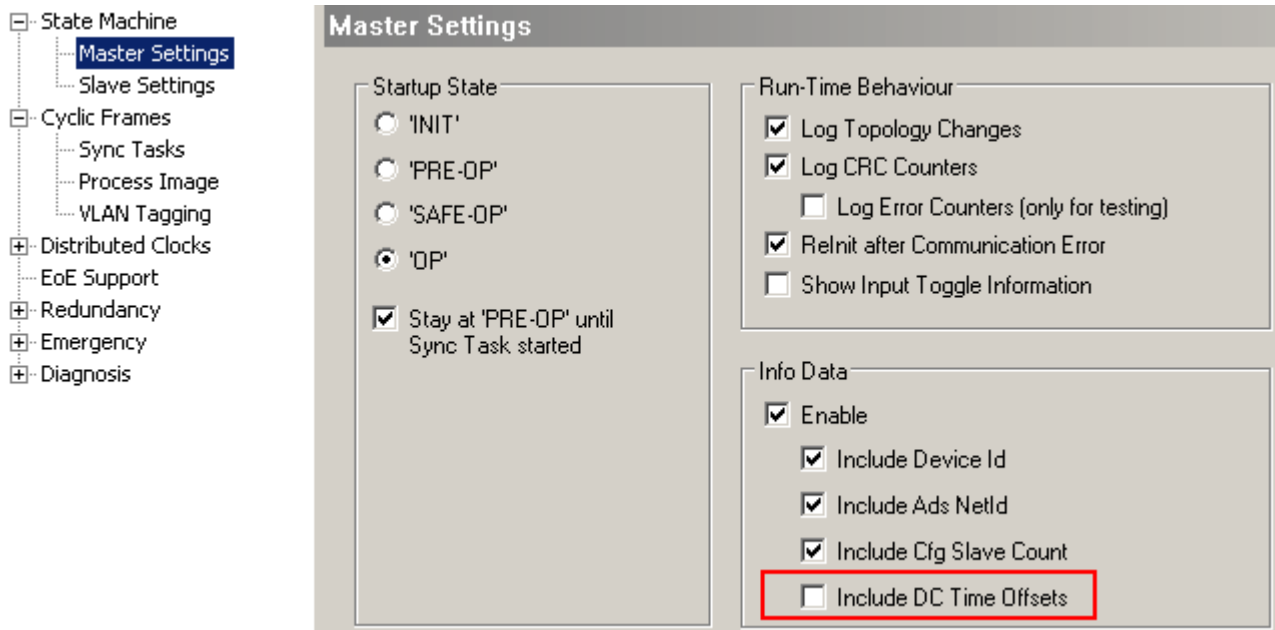


Fig. 9: display current offsets

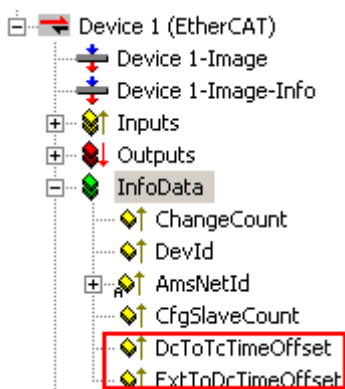


Fig. 10: Current offsets

## 2.4 Start

For commissioning:

- Install the EL6692 as described in chapter [Installation and wiring](#) [▶ 31].
- Configure the EL6692 in TwinCAT as described in the chapter [Commissioning](#) [▶ 46].

## 3 Basics communication

### 3.1 EtherCAT basics

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

### 3.2 EtherCAT cabling – wire-bound

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

#### Cables and connectors

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

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

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

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

#### Recommended cables

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

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

#### E-Bus supply

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

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

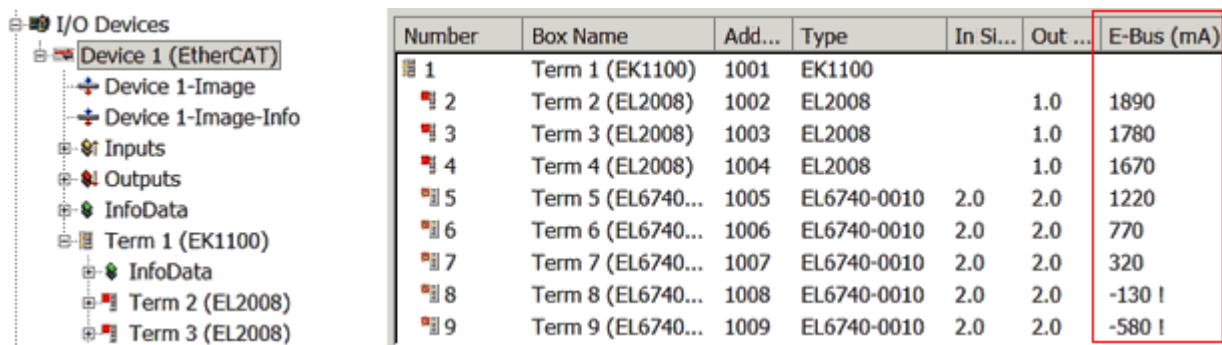


Fig. 11: System manager current calculation

**NOTE**

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

### 3.3 General notes for setting the watchdog

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

The EtherCAT slave controller (ESC) features two watchdogs:

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

#### SM watchdog (SyncManager Watchdog)

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

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

#### PDI watchdog (Process Data Watchdog)

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

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

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

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

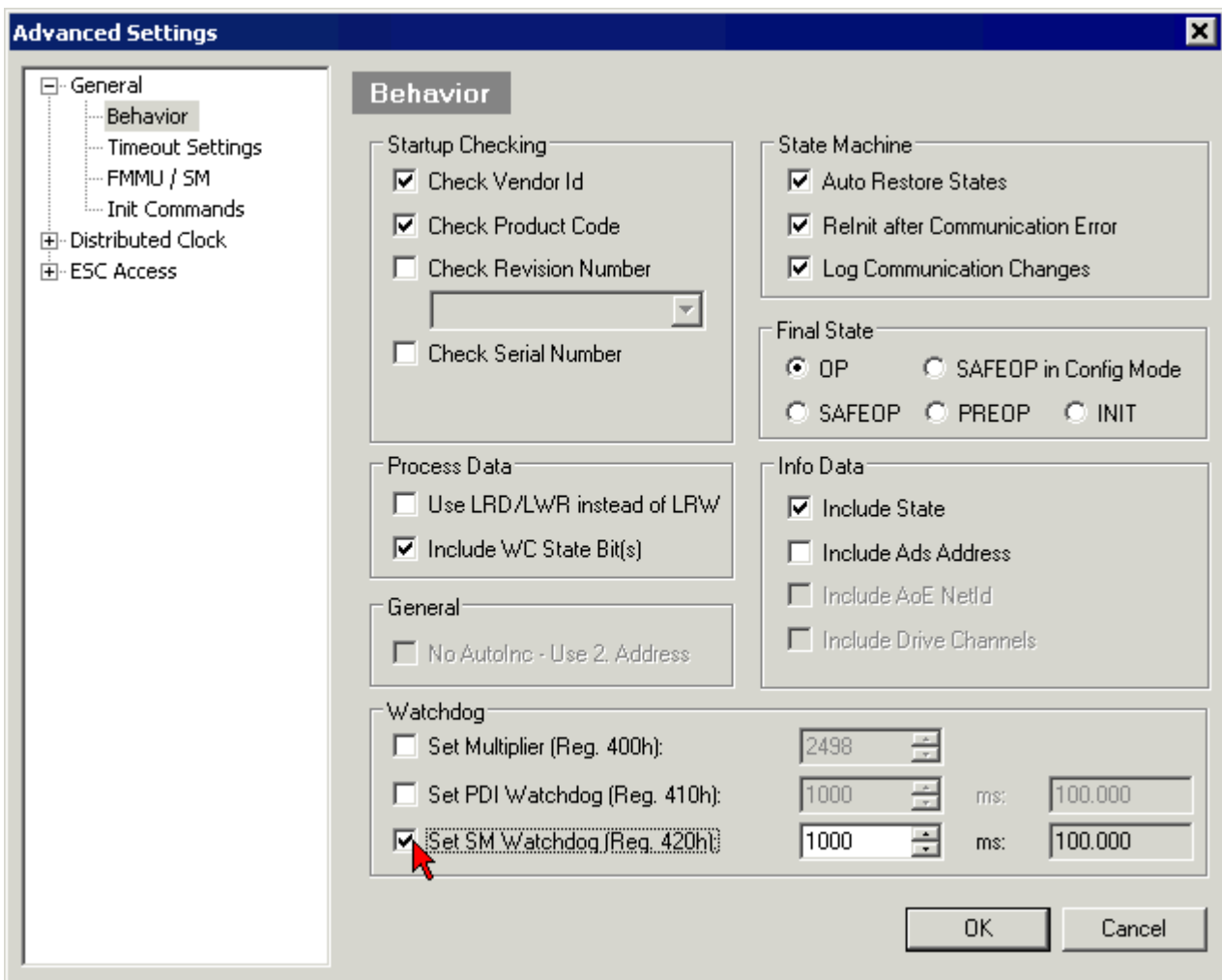


Fig. 12: EtherCAT tab -> Advanced Settings -> Behavior -> Watchdog

Notes:

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

**Multipplier**

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

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

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

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

**Example "Set SM watchdog"**

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

**Calculation**

Multiplier = 2498 → watchdog base time = 1 / 25 MHz \* (2498 + 2) = 0.0001 seconds = 100 μs  
 SM watchdog = 10000 → 10000 \* 100 μs = 1 second watchdog monitoring time

**⚠ CAUTION**

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

**⚠ CAUTION**

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

### 3.4 EtherCAT State Machine

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

A distinction is made between the following states:

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

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

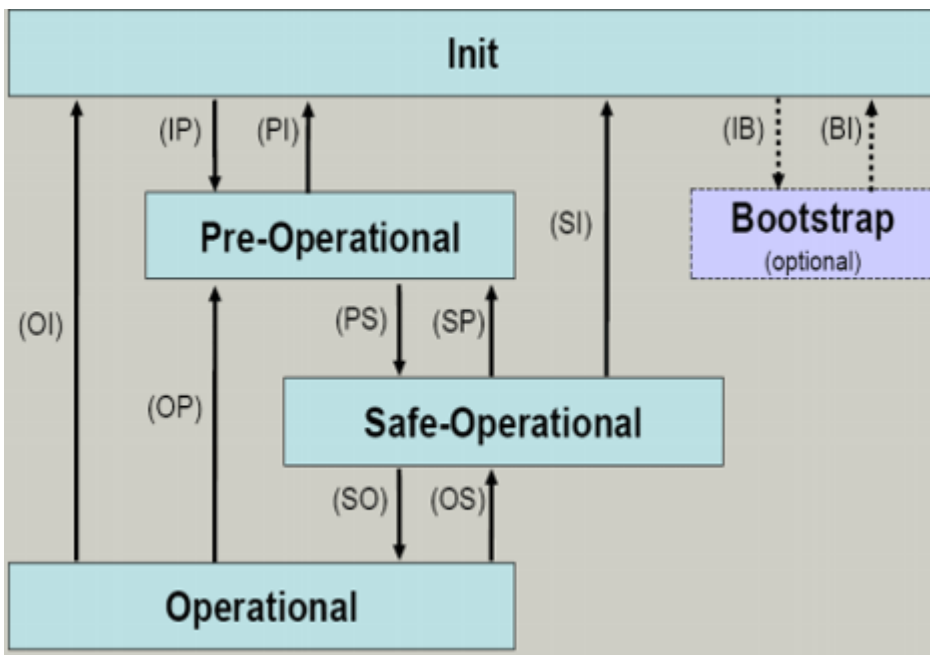


Fig. 13: States of the EtherCAT State Machine



**Init**

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

**Pre-Operational (Pre-Op)**

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

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

**Safe-Operational (Safe-Op)**

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

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

---

**● Outputs in SAFEOP state**

**i** The default set [watchdog \[▶ 22\]](#) monitoring sets the outputs of the module in a safe state - depending on the settings in SAFEOP and OP - e.g. in OFF state. If this is prevented by deactivation of the watchdog monitoring in the module, the outputs can be switched or set also in the SAFEOP state.

---

**Operational (Op)**

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

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

**Boot**

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

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

## 3.5 CoE Interface

**General description**

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

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

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

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

- Index: 0x0000 ...0xFFFF (0...65535<sub>dec</sub>)
- SubIndex: 0x00...0xFF (0...255<sub>dec</sub>)

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 ("input" from the perspective of the EtherCAT master)
- 0x7000: Output PDOs ("output" 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:

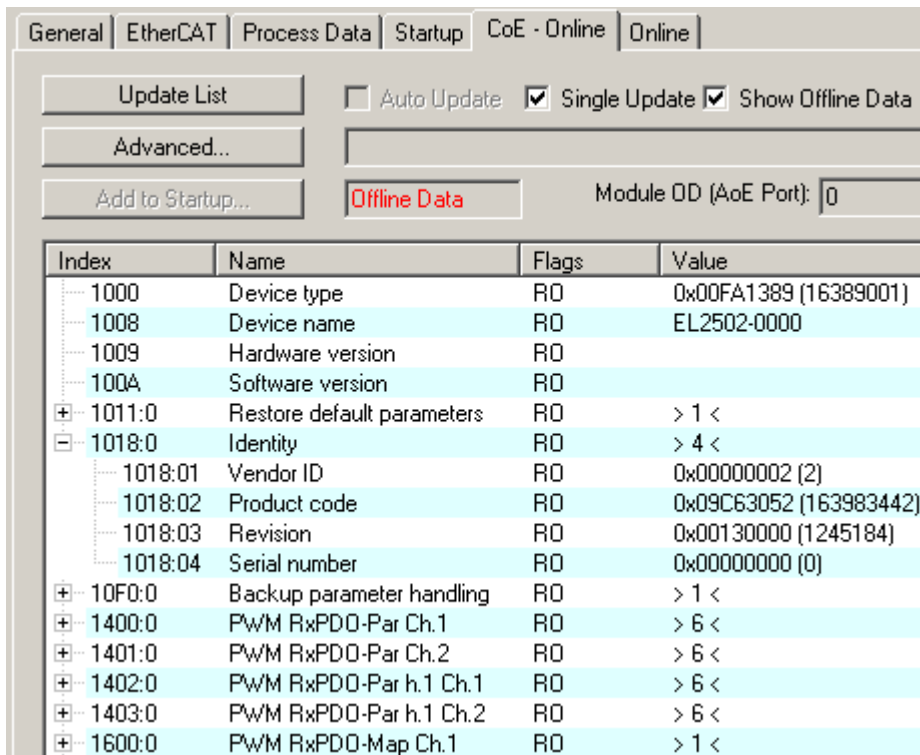


Fig. 14: "CoE Online" tab

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

### Data management and function “NoCoeStorage”

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

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

#### ● Data management

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

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

Please refer to the technical data in this documentation as to whether this applies to the respective device.

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

#### ● Startup list

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

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

### Recommended approach for manual modification of CoE parameters

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

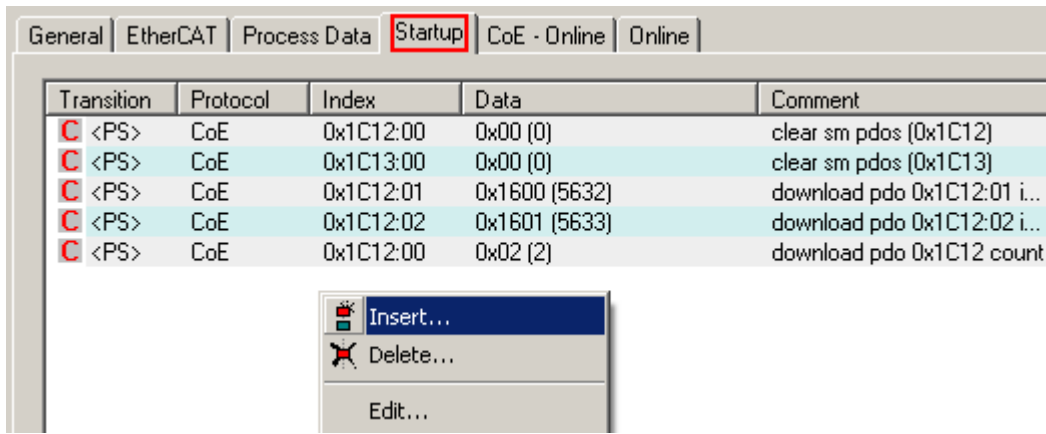


Fig. 15: Startup list in the TwinCAT System Manager

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

**Online/offline list**

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

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

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

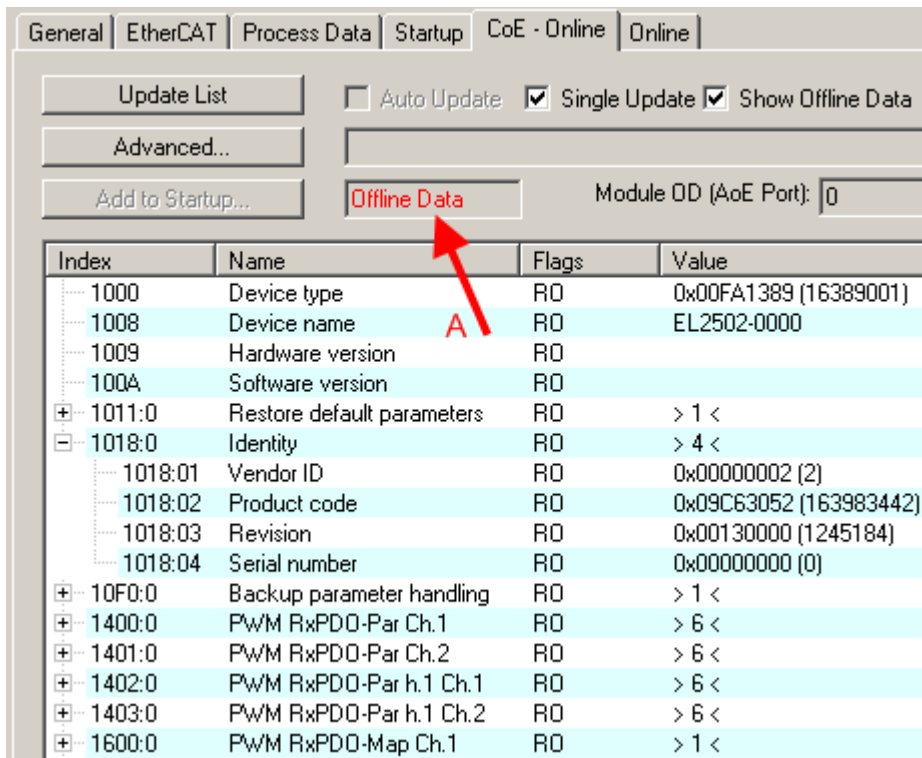


Fig. 16: Offline list

- If the slave is online
  - The actual current slave list is read. This may take several seconds, depending on the size and cycle time.
  - The actual identity is displayed
  - The firmware and hardware version of the equipment according to the electronic information is displayed
  - **Online** is shown in green.

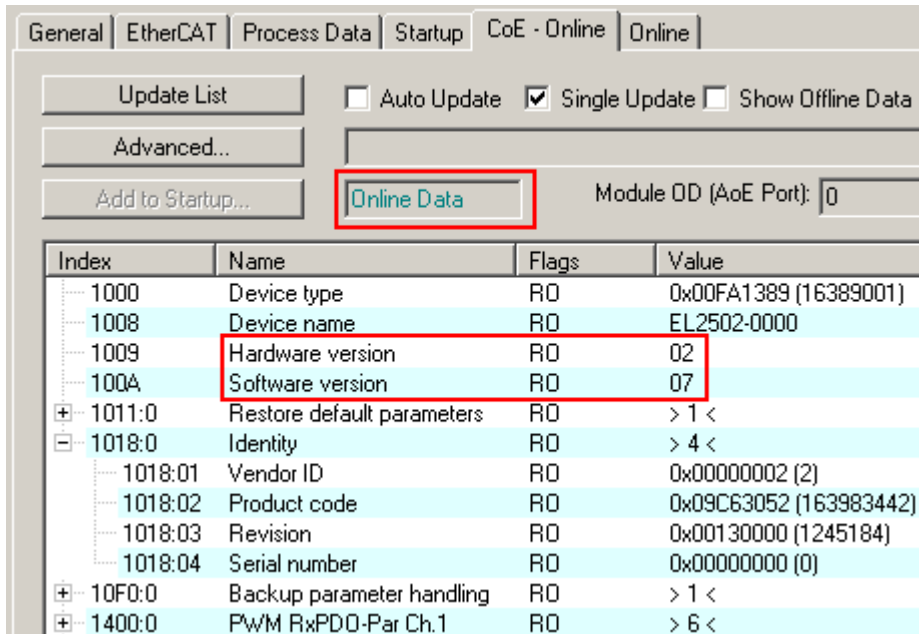


Fig. 17: Online list

**Channel-based order**

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

## 3.6 Distributed Clock

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

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

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

## 4 Mounting and wiring

### 4.1 Instructions for ESD protection

#### NOTE

##### **Destruction of the devices by electrostatic discharge possible!**

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

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

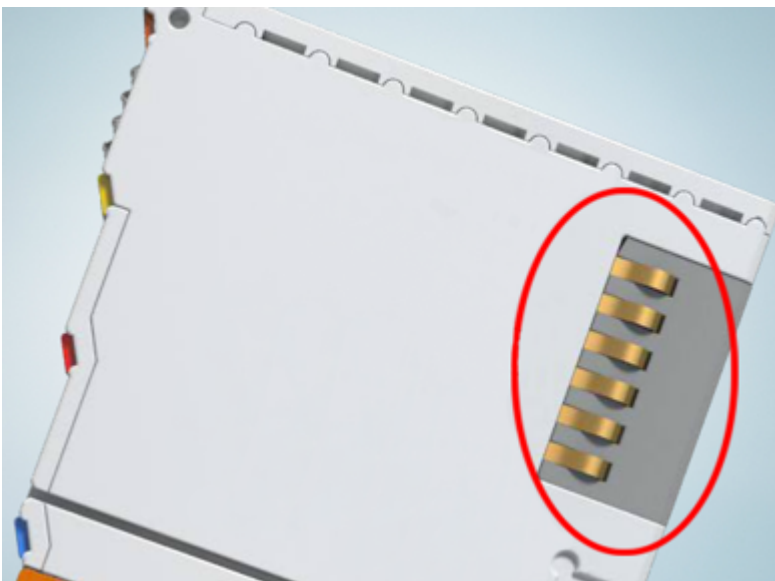


Fig. 18: Spring contacts of the Beckhoff I/O components

## 4.2 Explosion protection

### 4.2.1 ATEX - Special conditions (extended temperature range)

#### ⚠ WARNING

**Observe the special conditions for the intended use of Beckhoff fieldbus components with extended temperature range (ET) in potentially explosive areas (directive 2014/34/EU)!**

- The certified components are to be installed in a suitable housing that guarantees a protection class of at least IP54 in accordance with EN 60079-15! The environmental conditions during use are thereby to be taken into account!
- For dust (only the fieldbus components of certificate no. KEMA 10ATEX0075 X Issue 9): The equipment shall be installed in a suitable enclosure providing a degree of protection of IP54 according to EN 60079-31 for group IIIA or IIIB and IP6X for group IIIC, taking into account the environmental conditions under which the equipment is used!
- If the temperatures during rated operation are higher than 70°C at the feed-in points of cables, lines or pipes, or higher than 80°C at the wire branching points, then cables must be selected whose temperature data correspond to the actual measured temperature values!
- Observe the permissible ambient temperature range of -25 to 60°C for the use of Beckhoff fieldbus components with extended temperature range (ET) in potentially explosive areas!
- Measures must be taken to protect against the rated operating voltage being exceeded by more than 40% due to short-term interference voltages!
- The individual terminals may only be unplugged or removed from the Bus Terminal system if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- The connections of the certified components may only be connected or disconnected if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- The fuses of the KL92xx/EL92xx power feed terminals may only be exchanged if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!
- Address selectors and ID switches may only be adjusted if the supply voltage has been switched off or if a non-explosive atmosphere is ensured!

#### Standards

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

- EN 60079-0:2012+A11:2013
- EN 60079-15:2010
- EN 60079-31:2013 (only for certificate no. KEMA 10ATEX0075 X Issue 9)

#### Marking

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



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

II 3D KEMA 10ATEX0075 X Ex tc IIIC T135°C Dc Ta: -25 ... +60°C  
(only for fieldbus components of certificate no. KEMA 10ATEX0075 X Issue 9)

or



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

II 3D KEMA 10ATEX0075 X Ex tc IIIC T135°C Dc Ta: -25 ... +60°C  
(only for fieldbus components of certificate no. KEMA 10ATEX0075 X Issue 9)



## 4.2.2 Continuative documentation for ATEX and IECEx

---



### **Continuative documentation about explosion protection according to ATEX and IECEx**

Pay also attention to the continuative documentation




#### **Ex. Protection for Terminal Systems**

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

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

---

### 4.3 UL notice

|   |   |
|---|---|
|  | <b>Application</b><br>Beckhoff EtherCAT modules are intended for use with Beckhoff's UL Listed EtherCAT System only.  |
|  | <b>Examination</b><br>For cULus examination, the Beckhoff I/O System has only been investigated for risk of fire and electrical shock (in accordance with UL508 and CSA C22.2 No. 142). |
|  | <b>For devices with Ethernet connectors</b><br>Not for connection to telecommunication circuits.  |

#### Basic principles

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



## 4.4 Recommended mounting rails

Terminal Modules und EtherCAT Modules of KMxxxx and EMxxxx series, same as the terminals of the EL66xx and EL67xx series can be snapped onto the following recommended mounting rails:

DIN Rail TH 35-7.5 with 1 mm material thickness (according to EN 60715)

DIN Rail TH 35-15 with 1,5 mm material thickness

---

**● Pay attention to the material thickness of the DIN Rail**

**i** Terminal Modules und EtherCAT Modules of KMxxxx and EMxxxx series, same as the terminals of the EL66xx and EL67xx series does not fit to the DIN Rail TH 35-15 with 2,2 to 2,5 mm material thickness (according to EN 60715)!

---

## 4.5 Mounting and demounting - terminals with 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

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

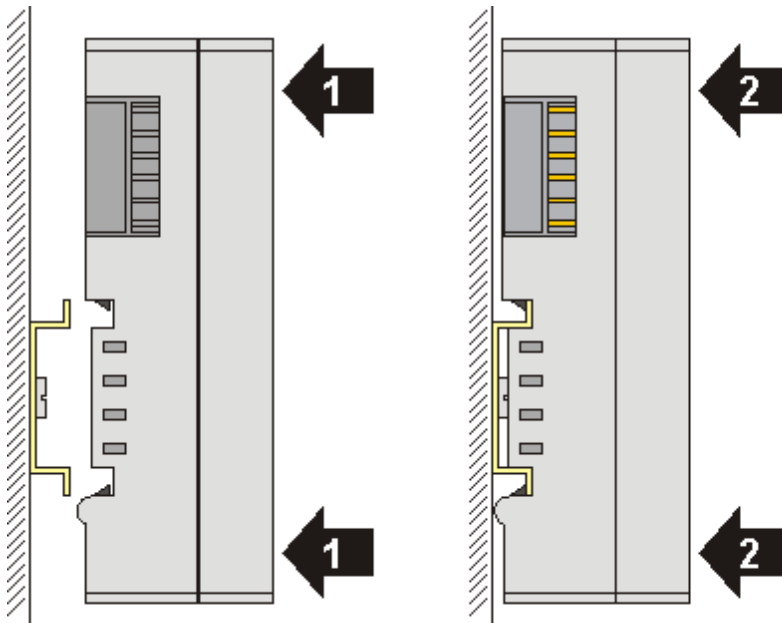
### ⚠ WARNING

#### 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.

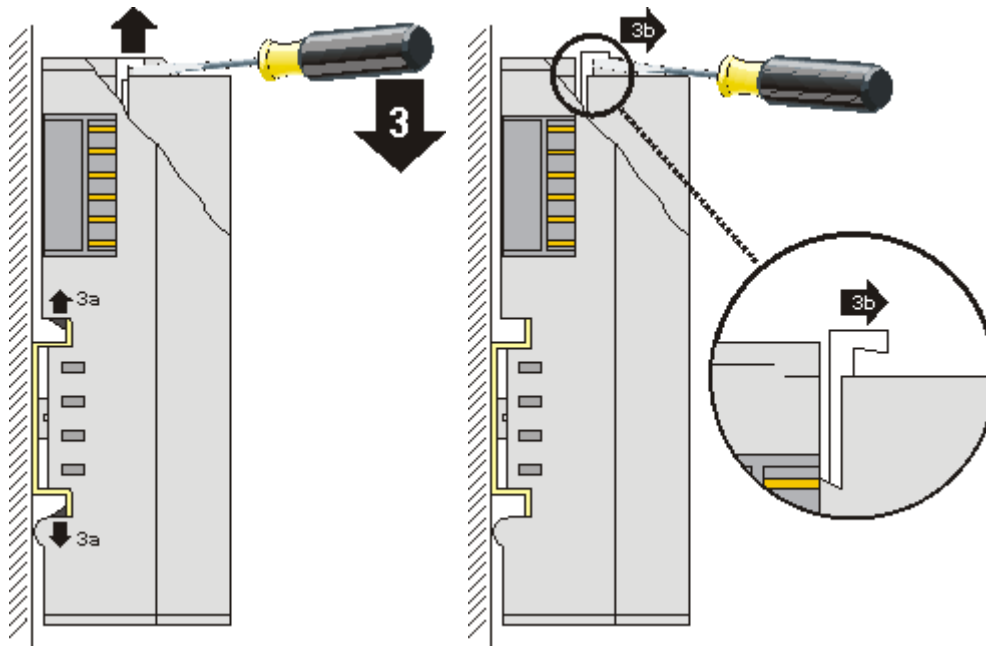


and press (1) the terminal module against the mounting rail until it latches in place on the mounting rail (2).

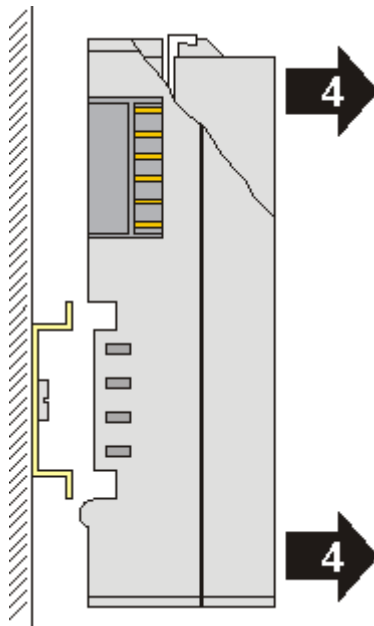
- Attach the cables.

### 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.



## 4.6 Mounting and demounting - terminals with 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

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

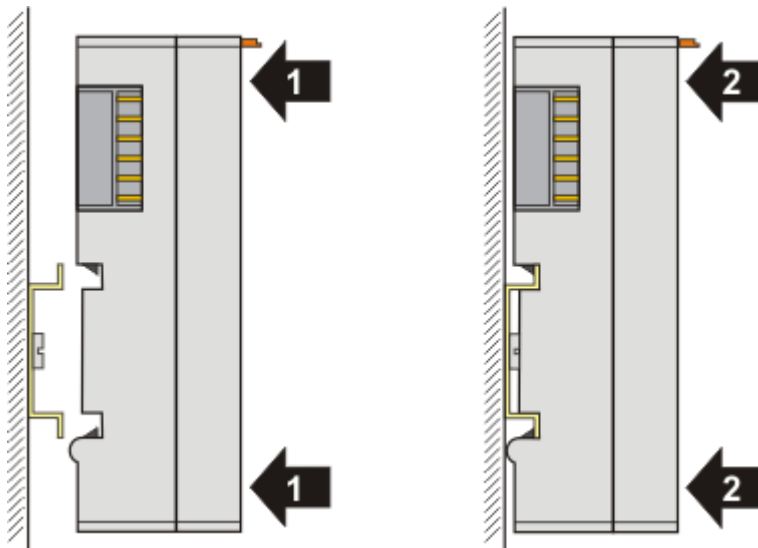
### ⚠ WARNING

#### 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.

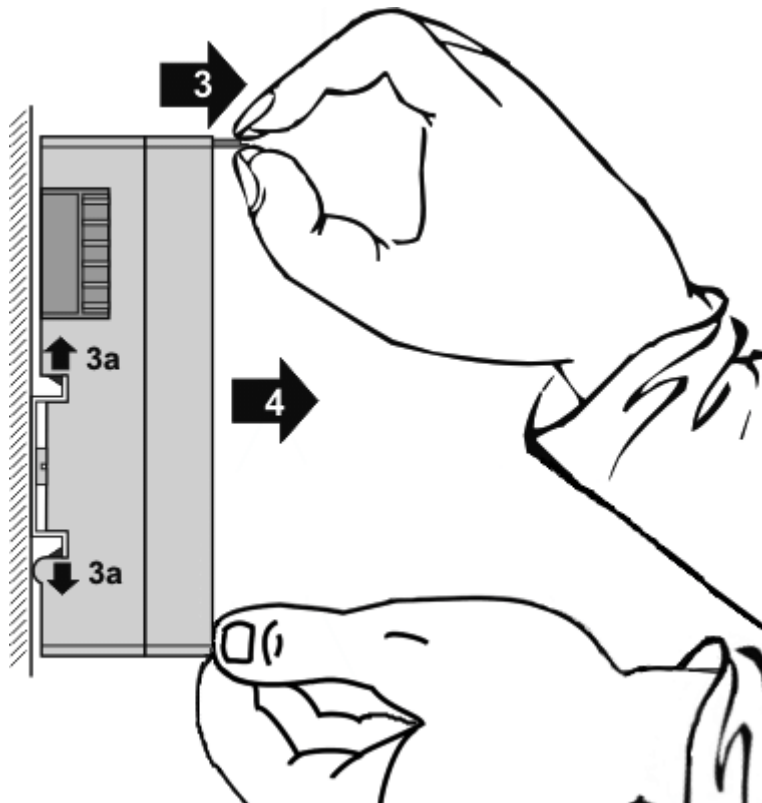


and press (1) the terminal module against the mounting rail until it latches in place on the mounting rail (2).

- 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.

## 4.7 Positioning of passive Terminals

**Hint for positioning of passive terminals in the bus terminal block**

**i** 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)**

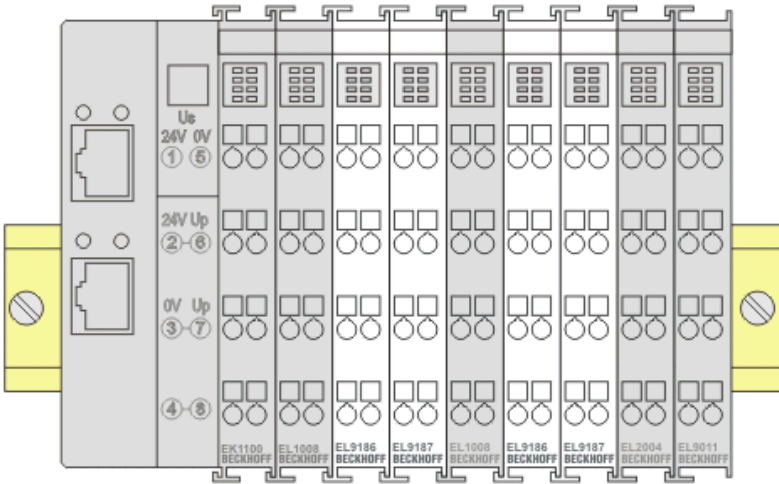


Fig. 19: Correct positioning

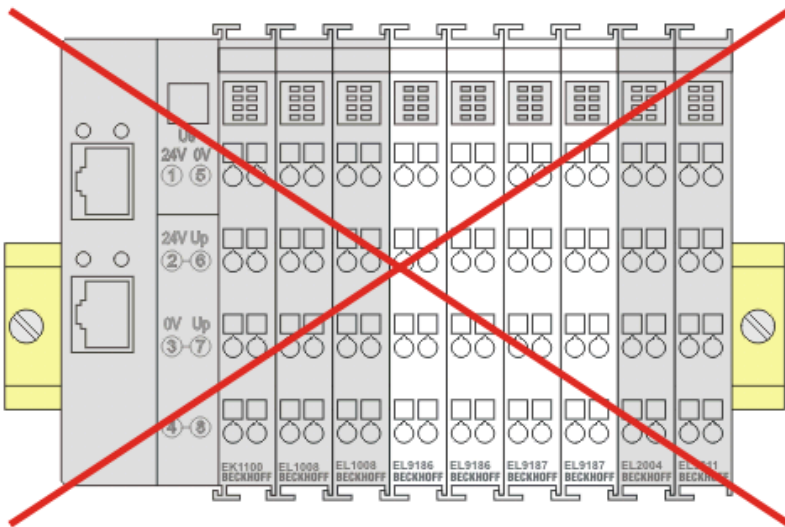


Fig. 20: Incorrect positioning



## 4.8 Installation positions

**NOTE**

**Constraints regarding installation position and operating temperature range**

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

**Optimum installation position (standard)**

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

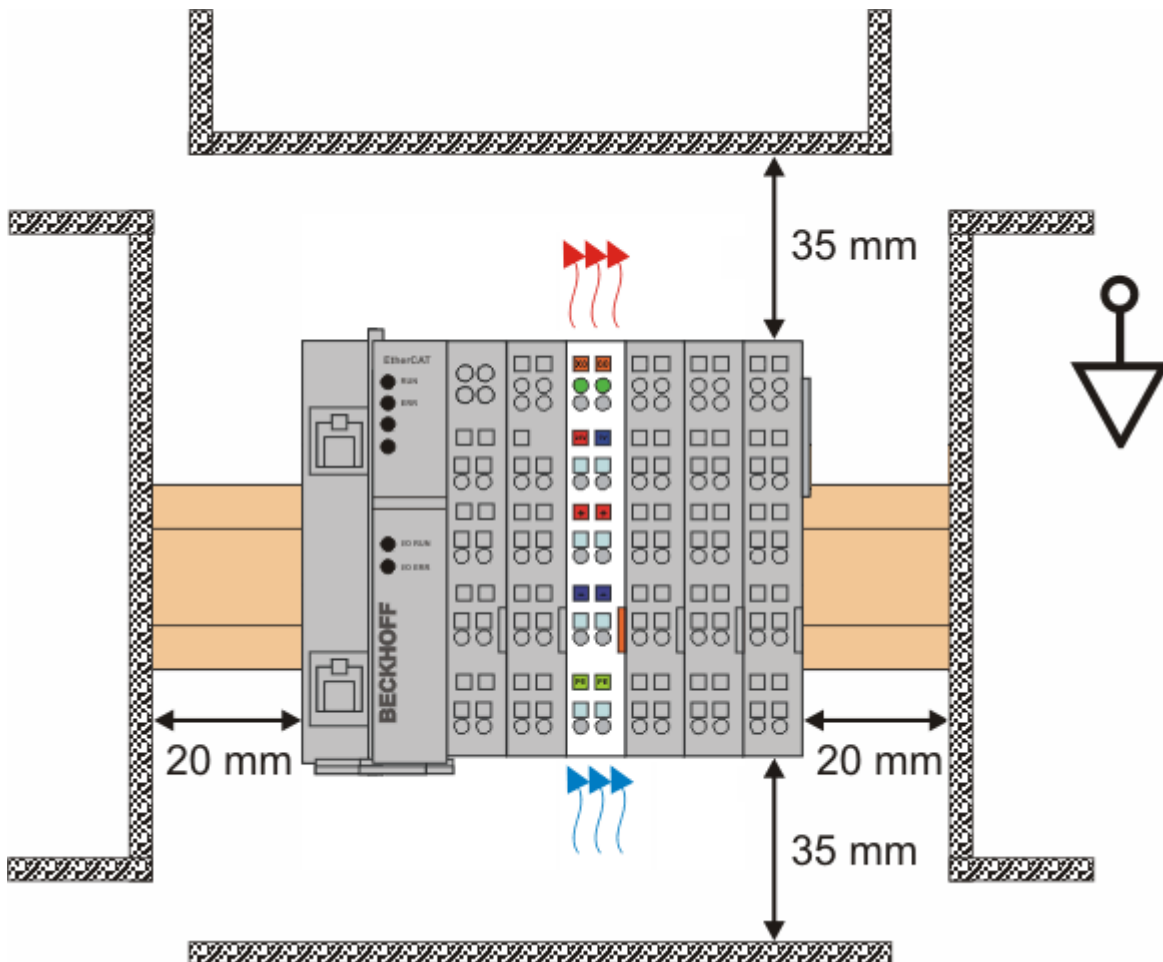


Fig. 21: Recommended distances for standard installation position

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.

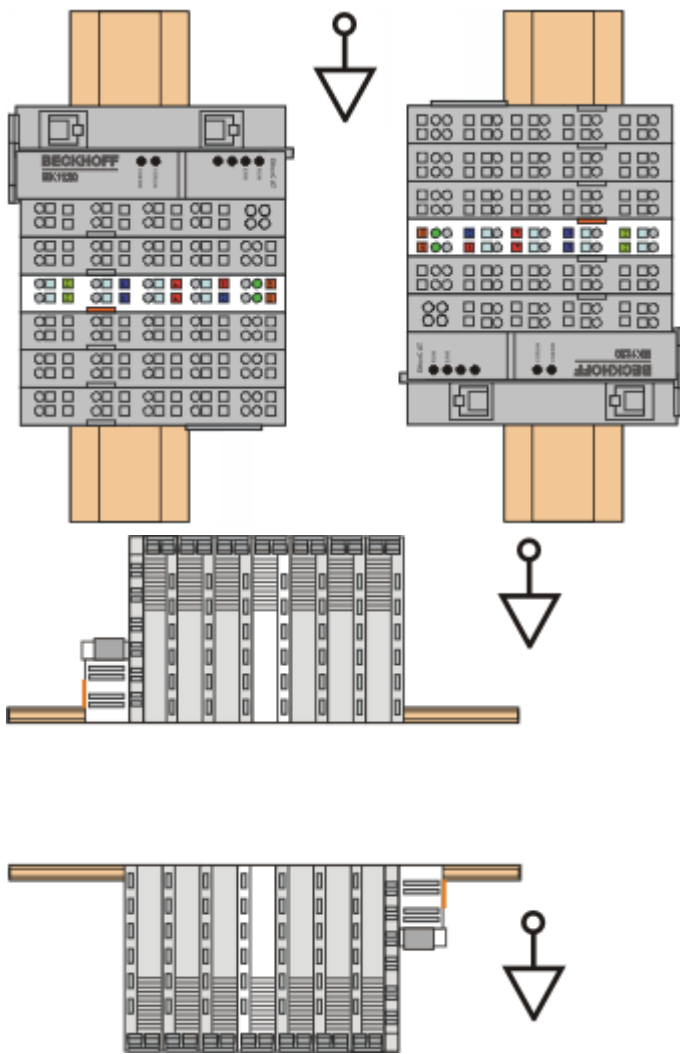


Fig. 22: Other installation positions

## 4.9 LEDs and connection

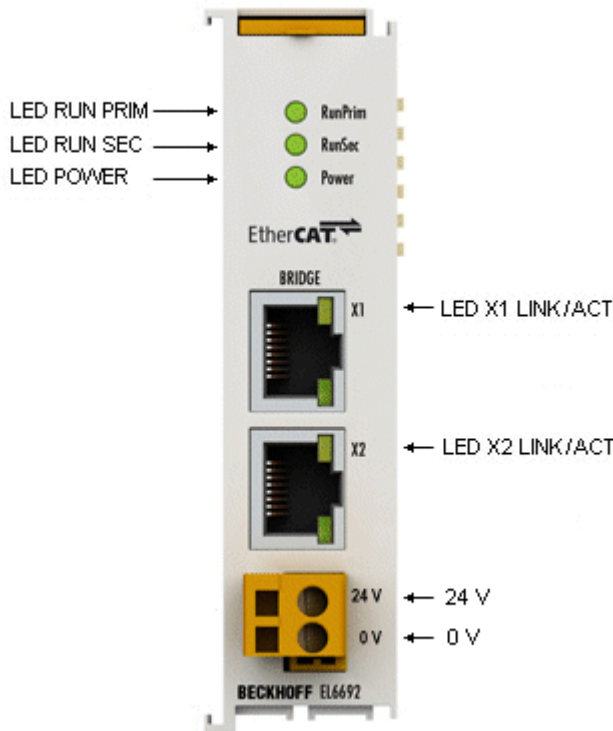


Fig. 23: EL6692 LEDs

### LEDs

| LED      | Color | Meaning  |  |
|----------|-------|--|--|
| RUN PRIM | green | LED for indicating the operating state of the <b>primary</b> EtherCAT strand (terminal): |  |
|          |       | off  | State of the <a href="#">EtherCAT State Machine [► 97]</a> : <b>INIT</b> = initialization of the terminal  |
|          |       | flashing   | State of the EtherCAT State Machine: <b>PREOP</b> = function for mailbox communication and different standard-settings set   |
|          |       | single flash   | State of the EtherCAT State Machine: <b>SAFEOP</b> = verification of the <a href="#">Sync Manager [► 98]</a> channels and the distributed clocks. Outputs remain in safe state |
|          |       | on   | State of the EtherCAT State Machine: <b>OP</b> = normal operating state; mailbox and process data communication is possible  |
| RUN SEC  | green | LED for indicating the operating state of the <b>primary</b> EtherCAT strand:            |  |
|          |       | off  | State of the <a href="#">EtherCAT State Machine [► 97]</a> : <b>INIT</b> = initialization of the terminal  |
|          |       | flashing   | State of the EtherCAT State Machine: <b>PREOP</b> = function for mailbox communication and different standard-settings set   |
|          |       | single flash   | State of the EtherCAT State Machine: <b>SAFEOP</b> = verification of the <a href="#">Sync Manager [► 98]</a> channels and the distributed clocks. Outputs remain in safe state |
|          |       | on   | State of the EtherCAT State Machine: <b>OP</b> = normal operating state; mailbox and process data communication is possible  |
| X1/X2    | green | off  | No connection to EtherCAT network  |
|          |       | on   | Connection to EtherCAT network established (LINK)  |
|          |       | flickering   | Data exchange with devices in the EtherCAT network (ACT)   |

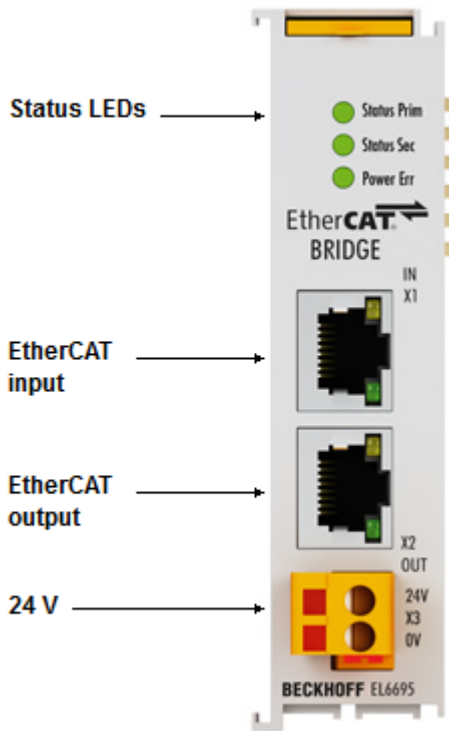


Fig. 24: Connection EL6695

| Terminal point | Description   |
|----------------|---|
| X1             | EtherCAT Input (RJ45 with 10BASE-T/100BASE-TX Ethernet)                 |
| X2             | EtherCAT output (RJ45 with 10BASE-T/100BASE-TX Ethernet)                |
| X3             | 2-pole socket terminal connection (24 VDC), secondary side power supply |

**i Diagnostic LEDs**

For the LED description please refer to the chapter Diagnostic LEDs

## 4.10 Disposal



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

## 5 Commissioning

### 5.1 TwinCAT Quick Start

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

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

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

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

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

The following relationship is envisaged from user PC to the individual control elements:

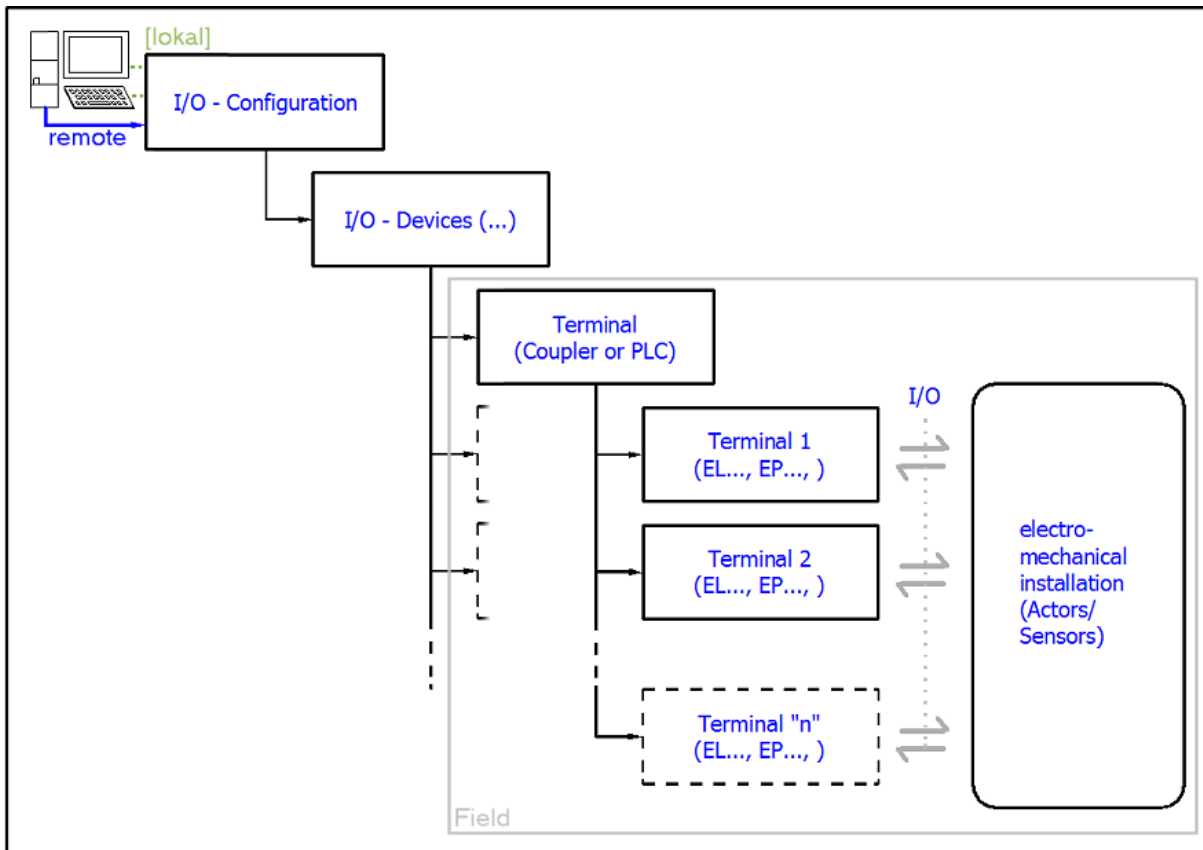


Fig. 25: Relationship between user side (commissioning) and installation

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

**Sample configuration (actual configuration)**

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

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

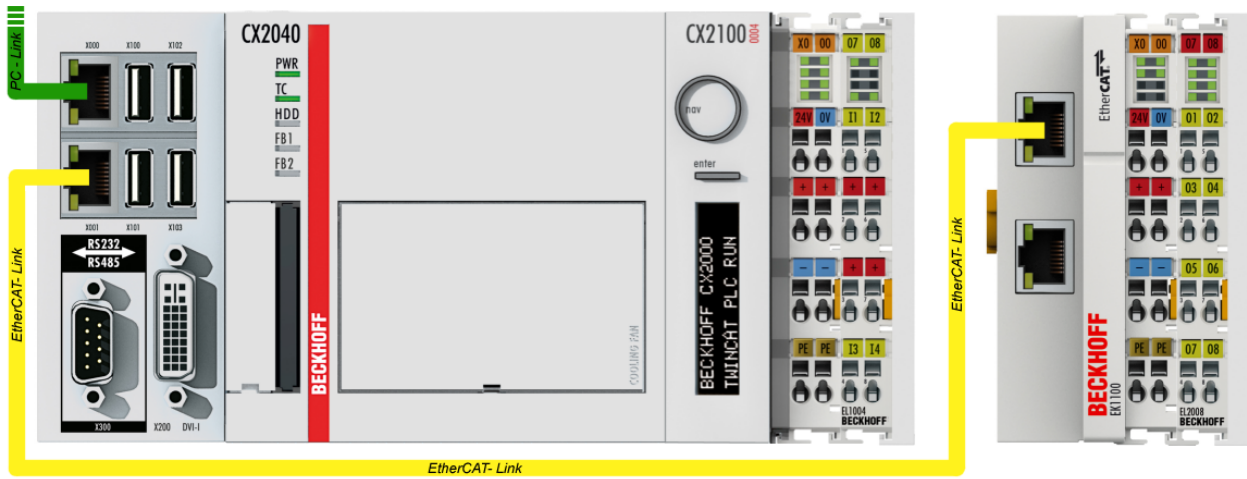


Fig. 26: Control configuration with Embedded PC, input (EL1004) and output (EL2008)

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



## 5.1.1 TwinCAT 2

### Startup

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

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

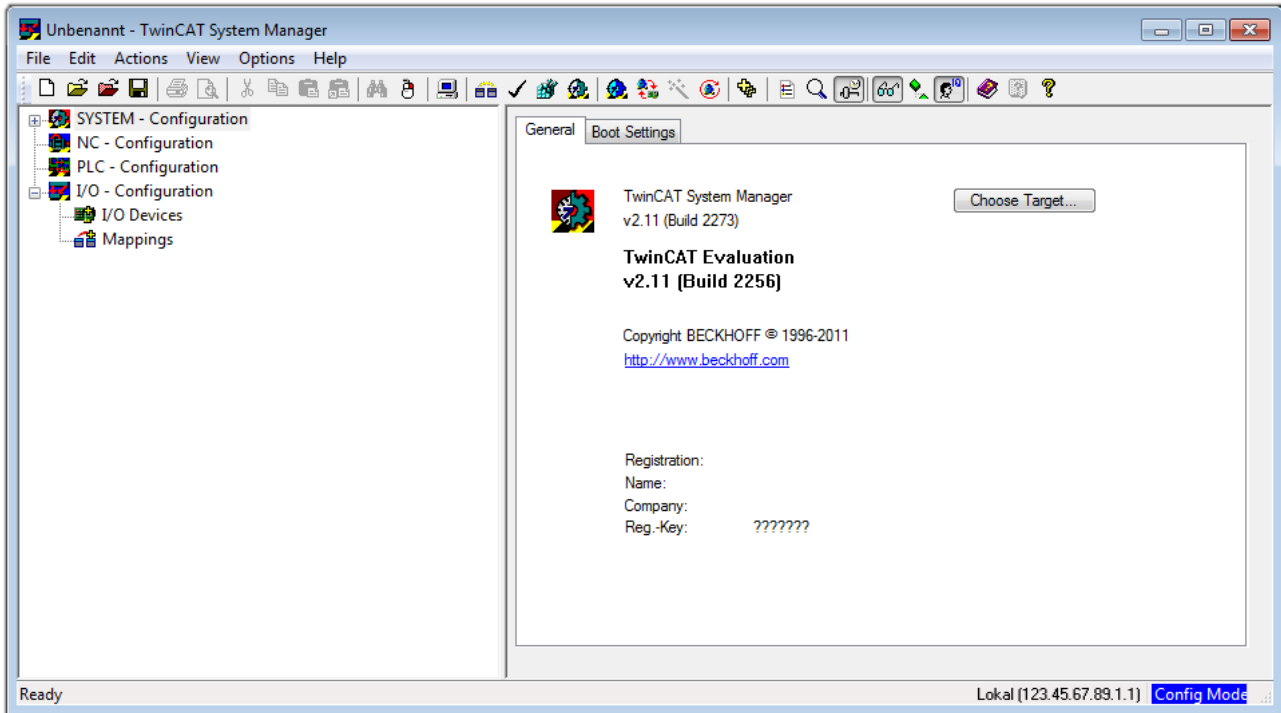



Fig. 27: Initial TwinCAT 2 user interface

Generally, TwinCAT can be used in local or remote mode. Once the TwinCAT system including the user interface (standard) is installed on the respective PLC, TwinCAT can be used in local mode and thereby the next step is “[Insert Device](#) [▶ 51]”.

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

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

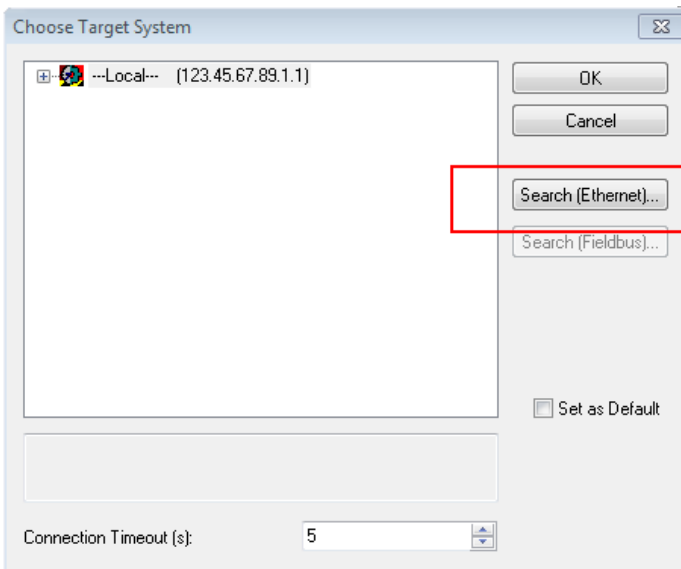


Fig. 28: Selection of the target system

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

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

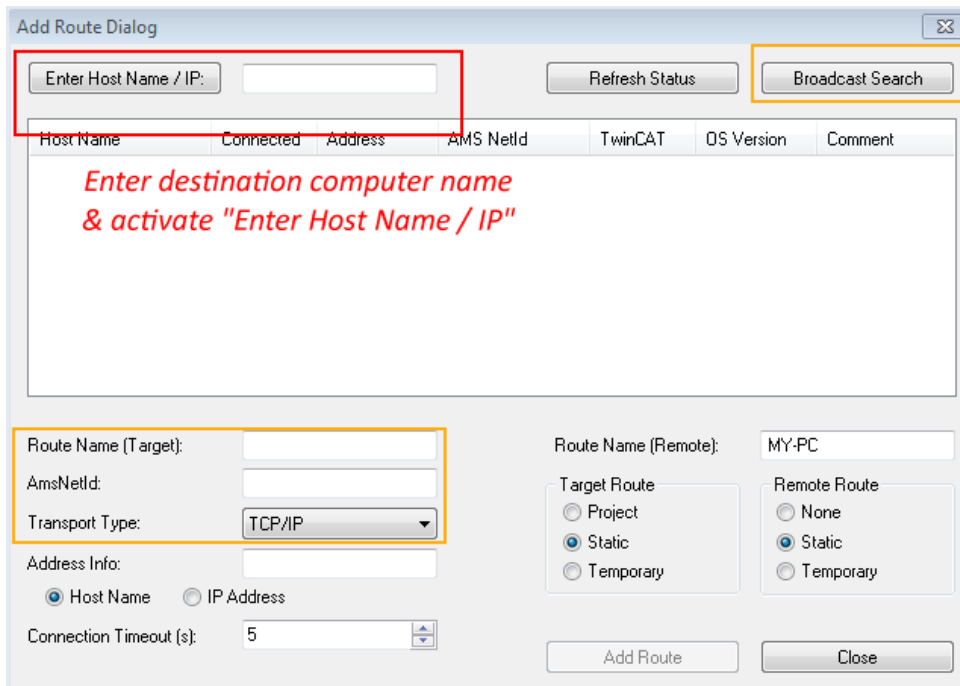
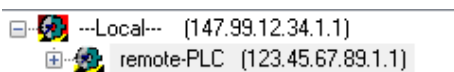


Fig. 29: Specify the PLC for access by the TwinCAT System Manager: selection of the target system



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



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

### Adding devices

In the configuration tree of the TwinCAT 2 System Manager user interface on the left, select “I/O Devices” and then right-click to open a context menu and select “Scan Devices...”, or start the action in the menu bar

via . The TwinCAT System Manager may first have to be set to “Config mode” via  or via menu “Actions” → “Set/Reset TwinCAT to Config Mode...” (Shift + F4).

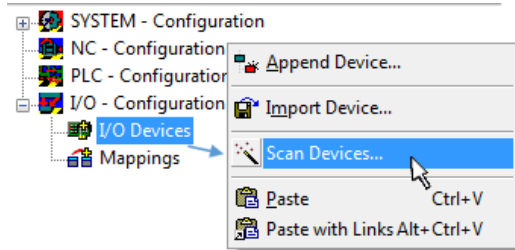


Fig. 30: Select “Scan Devices...”

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

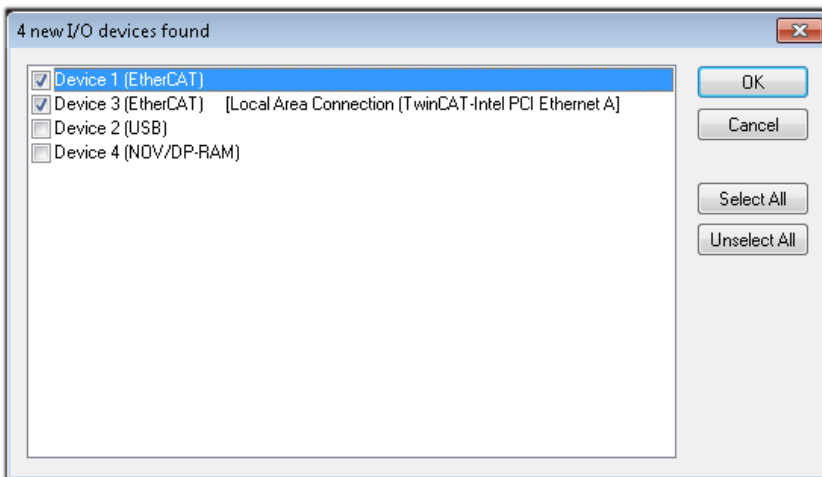


Fig. 31: Automatic detection of I/O devices: selection the devices to be integrated

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

Based on the [sample configuration \[▶ 47\]](#) described at the beginning of this section, the result is as follows:

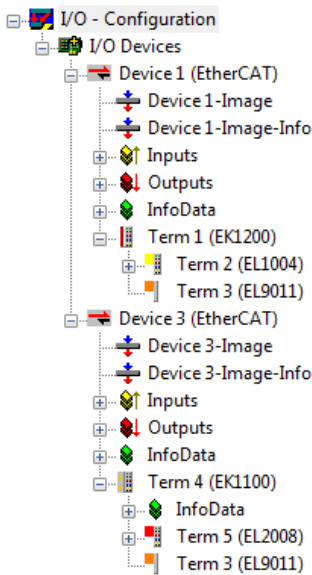


Fig. 32: Mapping of the configuration in the TwinCAT 2 System Manager

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

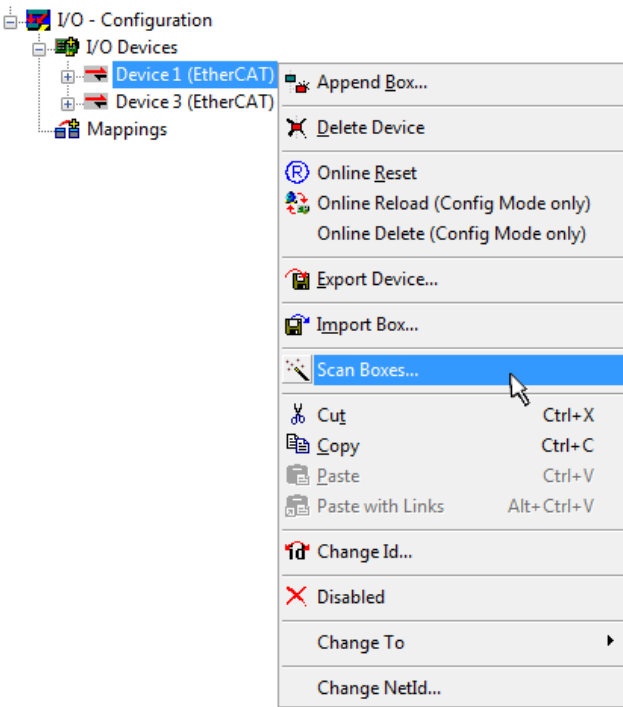


Fig. 33: Reading of individual terminals connected to a device

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

**Programming and integrating the PLC**

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

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

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

The following section refers to Structured Text (ST).

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

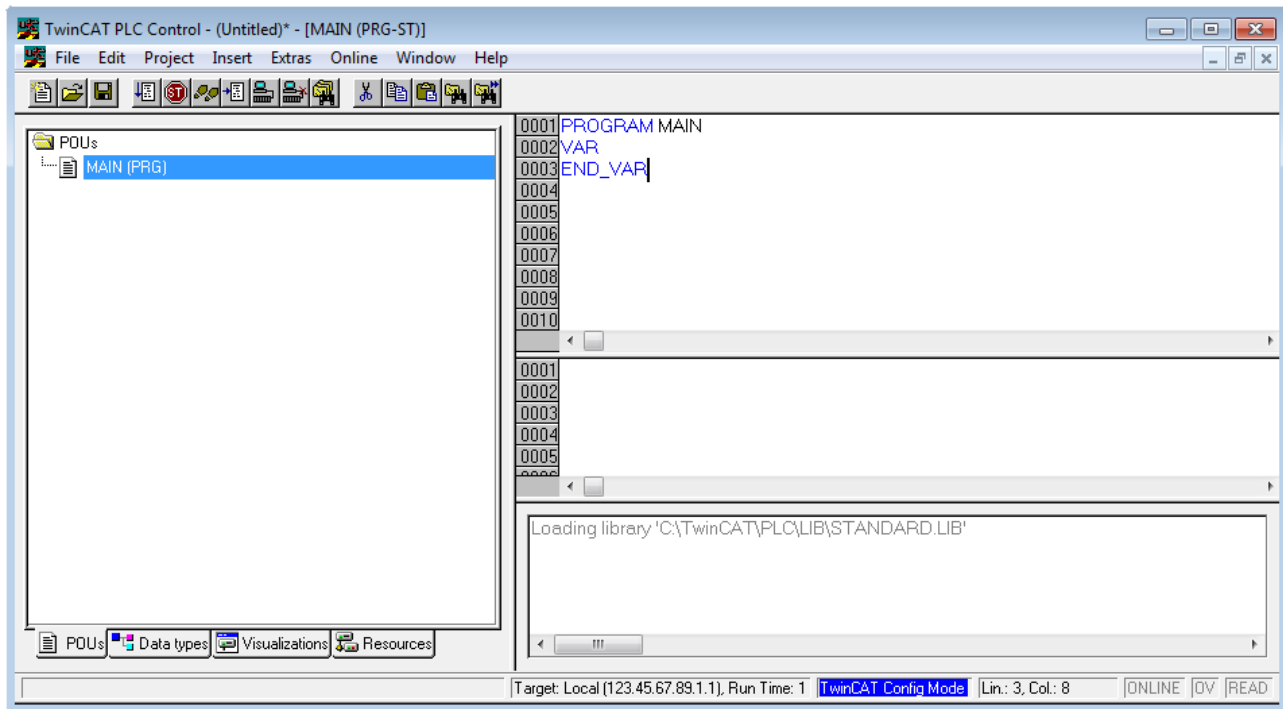


Fig. 34: TwinCAT PLC Control after startup

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

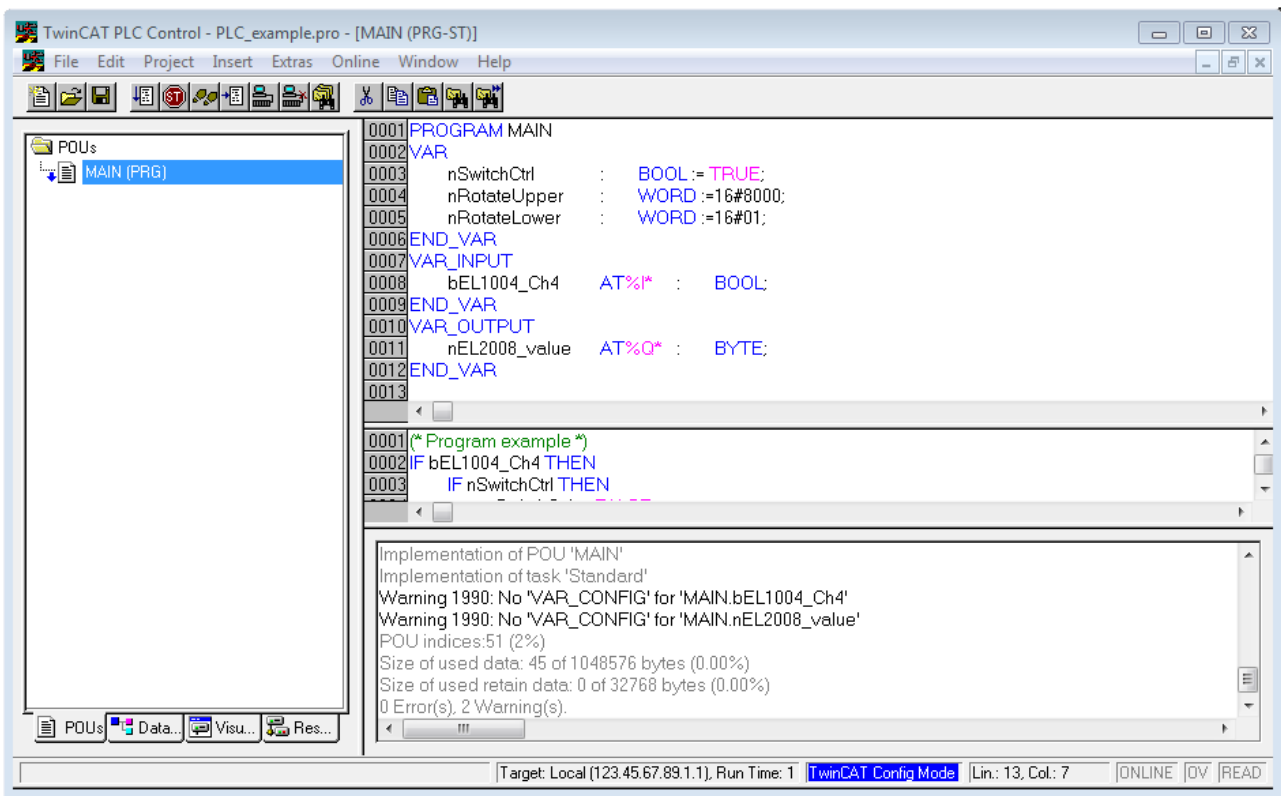


Fig. 35: Sample program with variables after a compile process (without variable integration)

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

First, integrate the TwinCAT PLC Control project in the **System Manager** via the context menu of the PLC configuration; right-click and select “Append PLC Project...”:

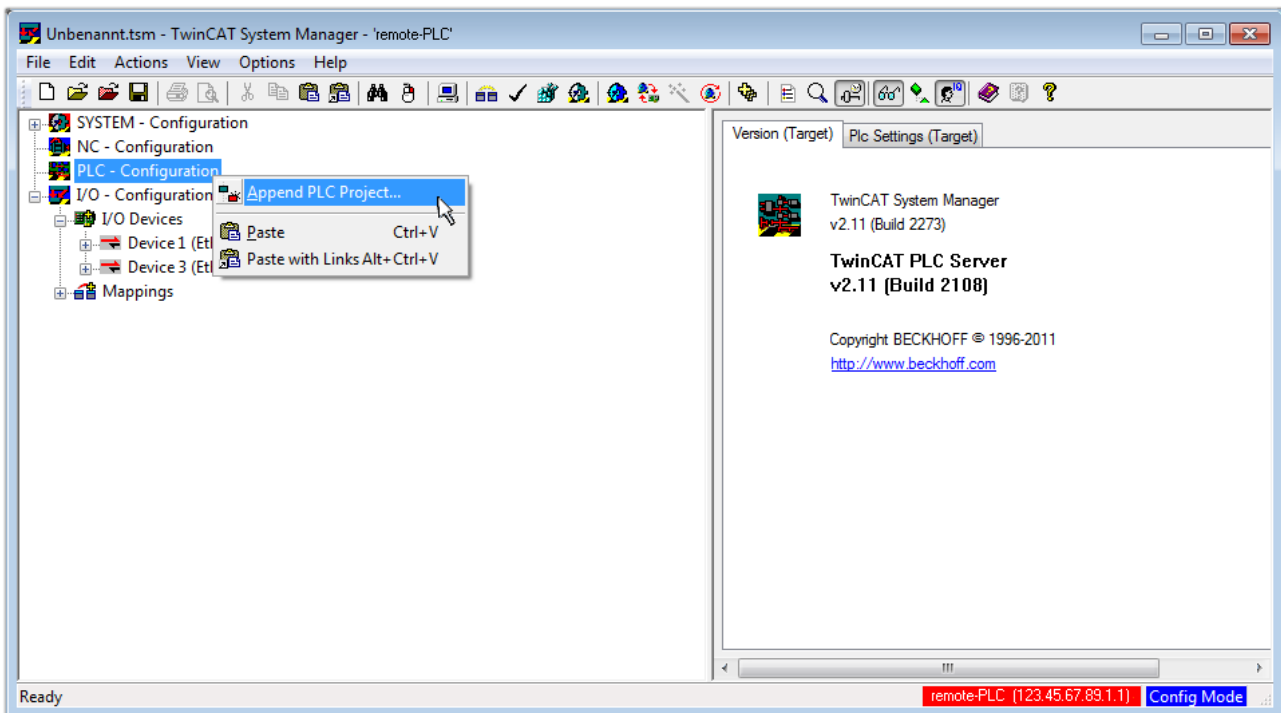


Fig. 36: Appending the TwinCAT PLC Control project

Select the PLC configuration “PLC\_example.tpy” in the browser window that opens. The project including the two variables identified with “AT” are then integrated in the configuration tree of the System Manager:

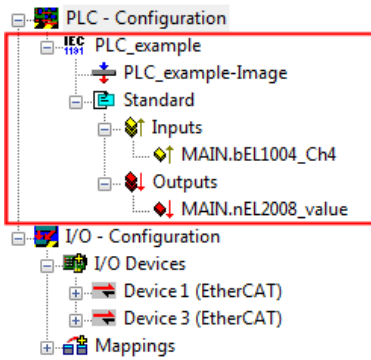


Fig. 37: PLC project integrated in the PLC configuration of the System Manager

The two variables “bEL1004\_Ch4” and “nEL2008\_value” can now be assigned to certain process objects of the I/O configuration.

### Assigning variables

Open a window for selecting a suitable process object (PDO) via the context menu of a variable of the integrated project “PLC\_example” and via “Modify Link...” “Standard”:

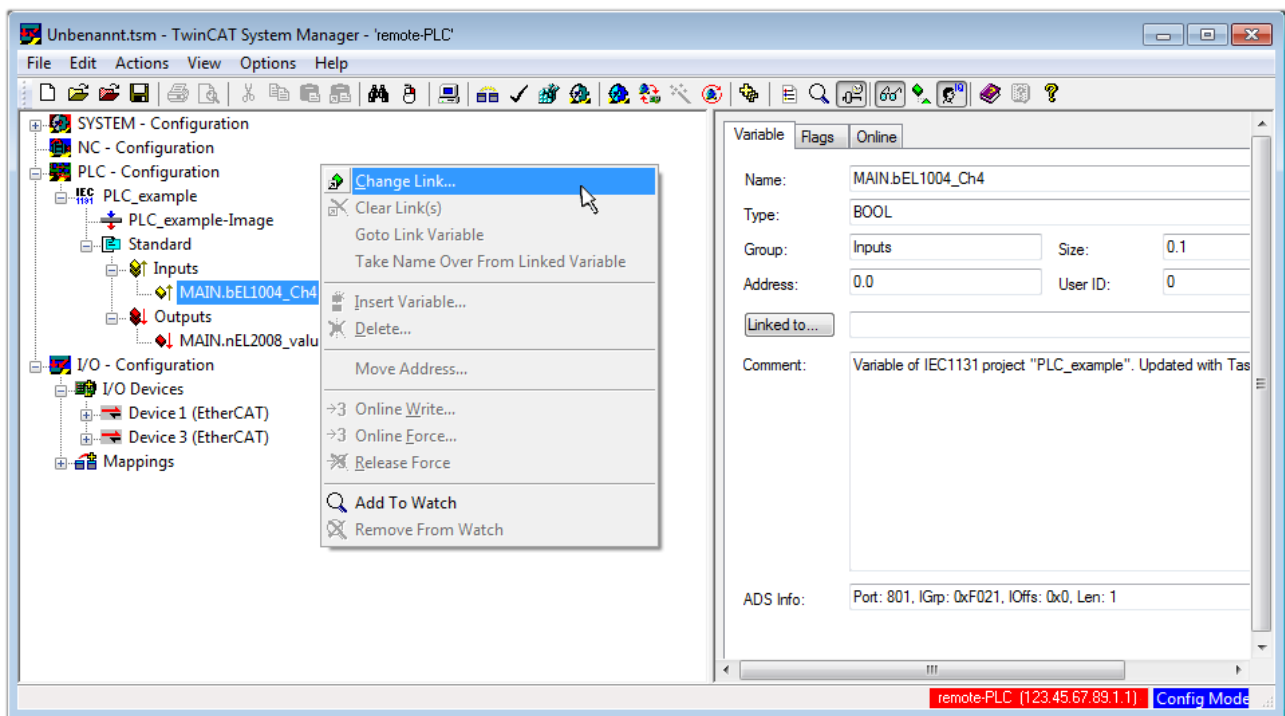


Fig. 38: Creating the links between PLC variables and process objects

In the window that opens, the process object for the variable “bEL1004\_Ch4” of type BOOL can be selected from the PLC configuration tree:

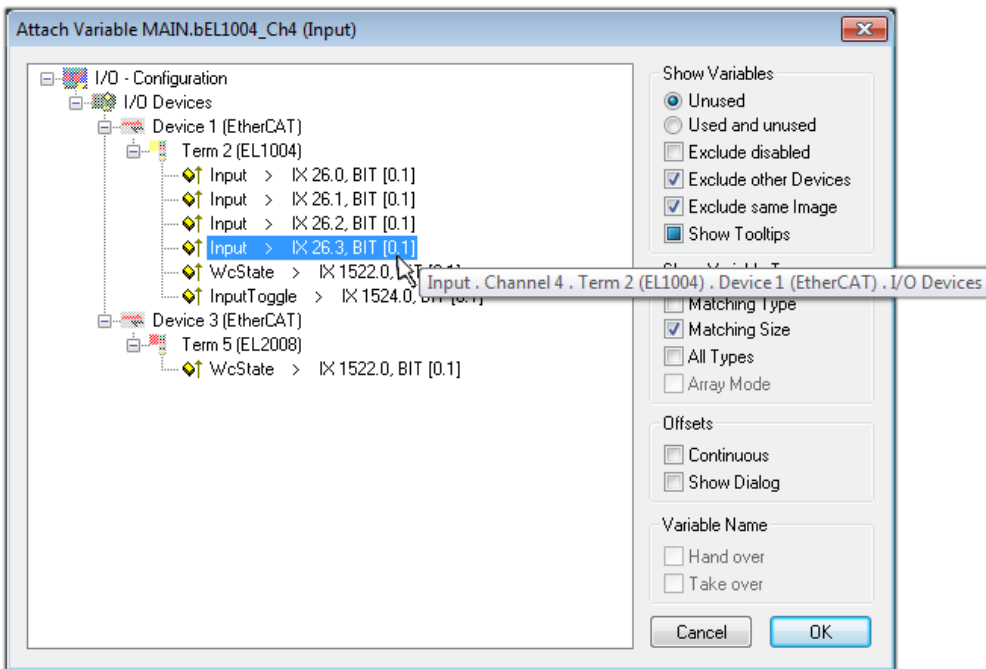


Fig. 39: Selecting PDO of type BOOL

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

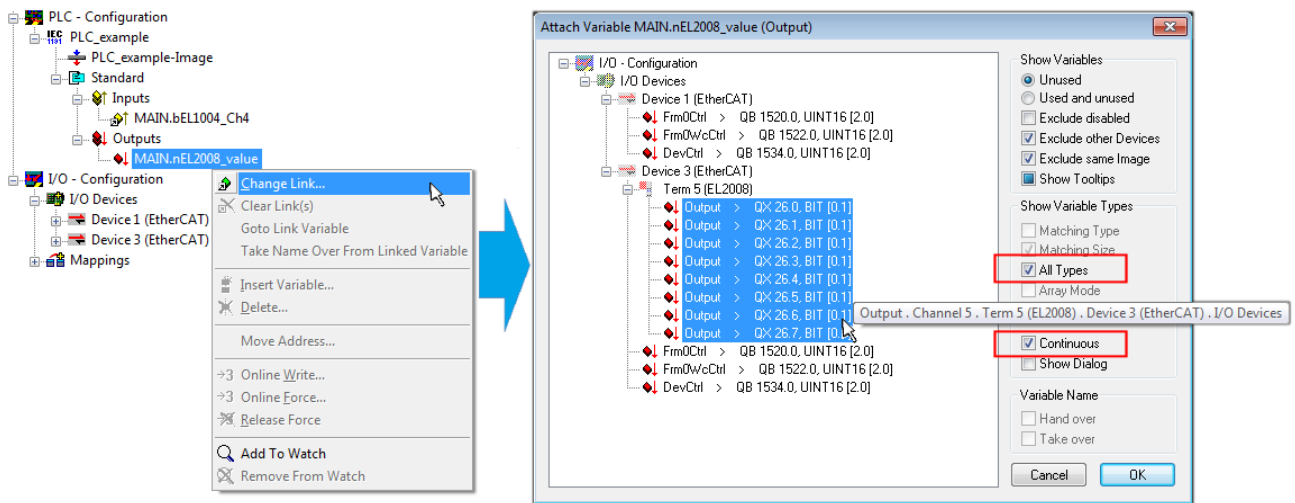



Fig. 40: Selecting several PDOs simultaneously: activate “Continuous” and “All types”

Note that the “Continuous” checkbox was also activated. This is designed to allocate the bits contained in the byte of the variable “nEL2008\_value” sequentially to all eight selected output bits of the EL2008 terminal. In this way it is possible to subsequently address all eight outputs of the terminal in the program with a byte corresponding to bit 0 for channel 1 to bit 7 for channel 8 of the PLC. A special symbol (  ) at the yellow or red object of the variable indicates that a link exists. The links can also be checked by selecting a “Goto Link Variable” from the context menu of a variable. The object opposite, in this case the PDO, is automatically selected:



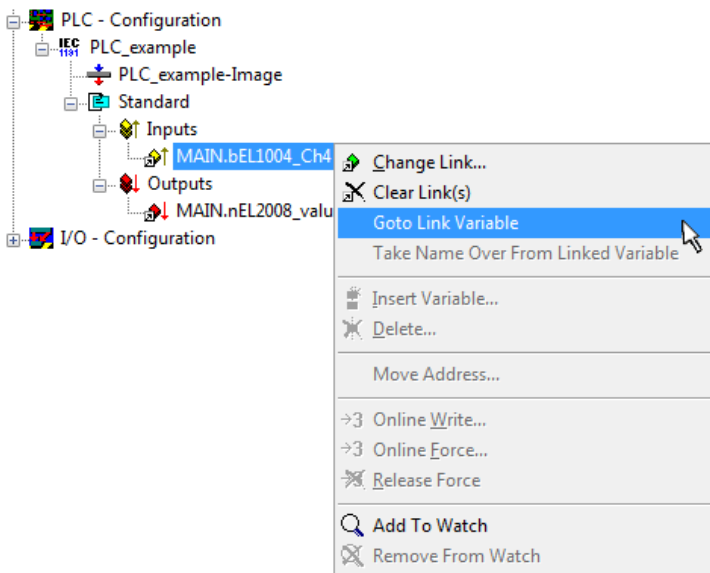

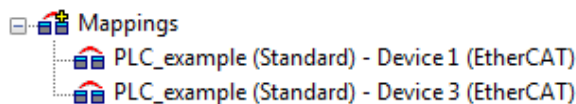


Fig. 41: Application of a “Goto Link” variable, using “MAIN.bEL1004\_Ch4” as a sample

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

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


This can be visualized in the configuration:




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

### Activation of the configuration

The allocation of PDO to PLC variables has now established the connection from the controller to the inputs and outputs of the terminals. The configuration can now be activated. First, the configuration can be verified

via  (or via “Actions” → “Check Configuration”). If no error is present, the configuration can be

activated via  (or via “Actions” → “Activate Configuration...”) to transfer the System Manager settings to the runtime system. Confirm the messages “Old configurations are overwritten!” and “Restart TwinCAT system in Run mode” with “OK”.

A few seconds later the real-time status **RTime 0%** is displayed at the bottom right in the System Manager. The PLC system can then be started as described below.

### Starting the controller

Starting from a remote system, the PLC control has to be linked with the Embedded PC over Ethernet via “Online” → “Choose Run-Time System...”:

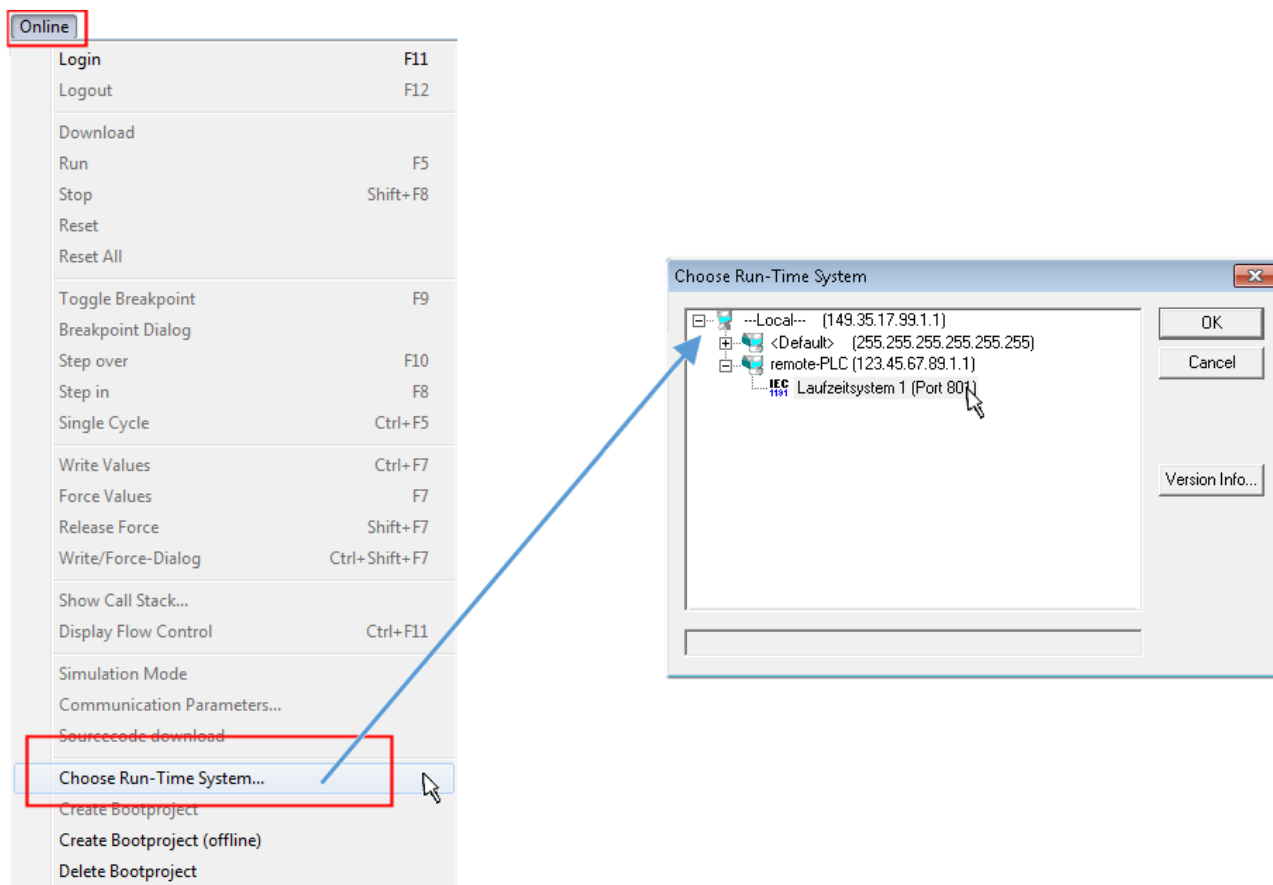



Fig. 42: Choose target system (remote)

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

system via menu option “Online” → “Login”, the F11 key or by clicking on the symbol . The control program can then be loaded for execution. This results in the message “No program on the controller! Should the new program be loaded?”, which should be acknowledged with “Yes”. The runtime environment is ready for the program start:

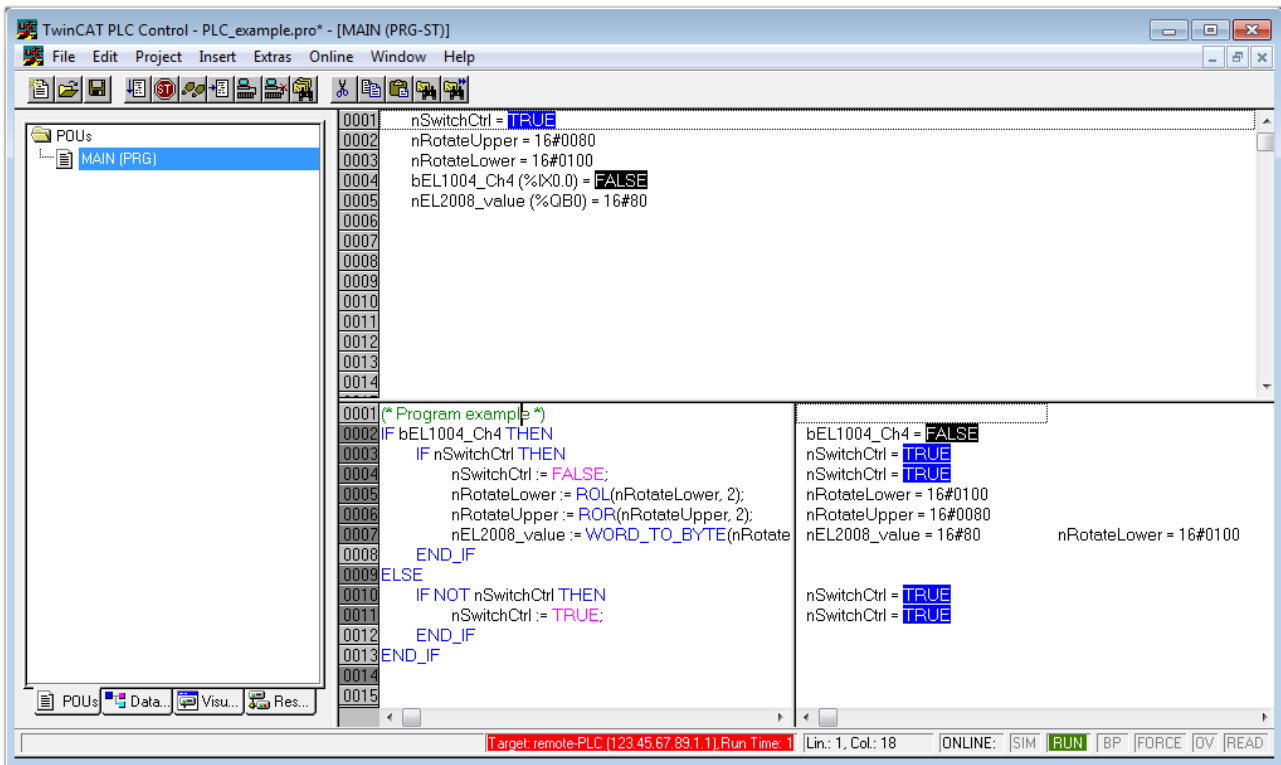


Fig. 43: PLC Control logged in, ready for program startup

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

### 5.1.2 TwinCAT 3

#### Startup

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

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

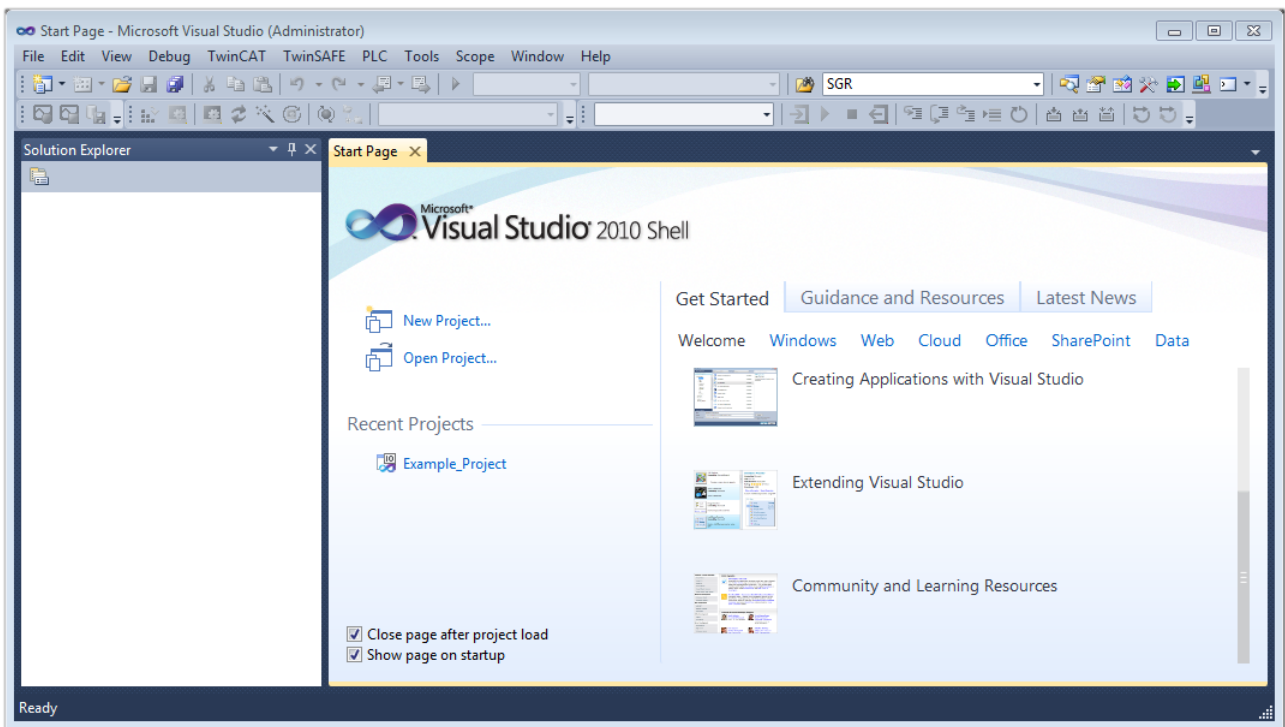



Fig. 44: Initial TwinCAT 3 user interface

First create a new project via  **New TwinCAT Project...** (or under “File”→“New”→“Project...”). In the following dialog make the corresponding entries as required (as shown in the diagram):

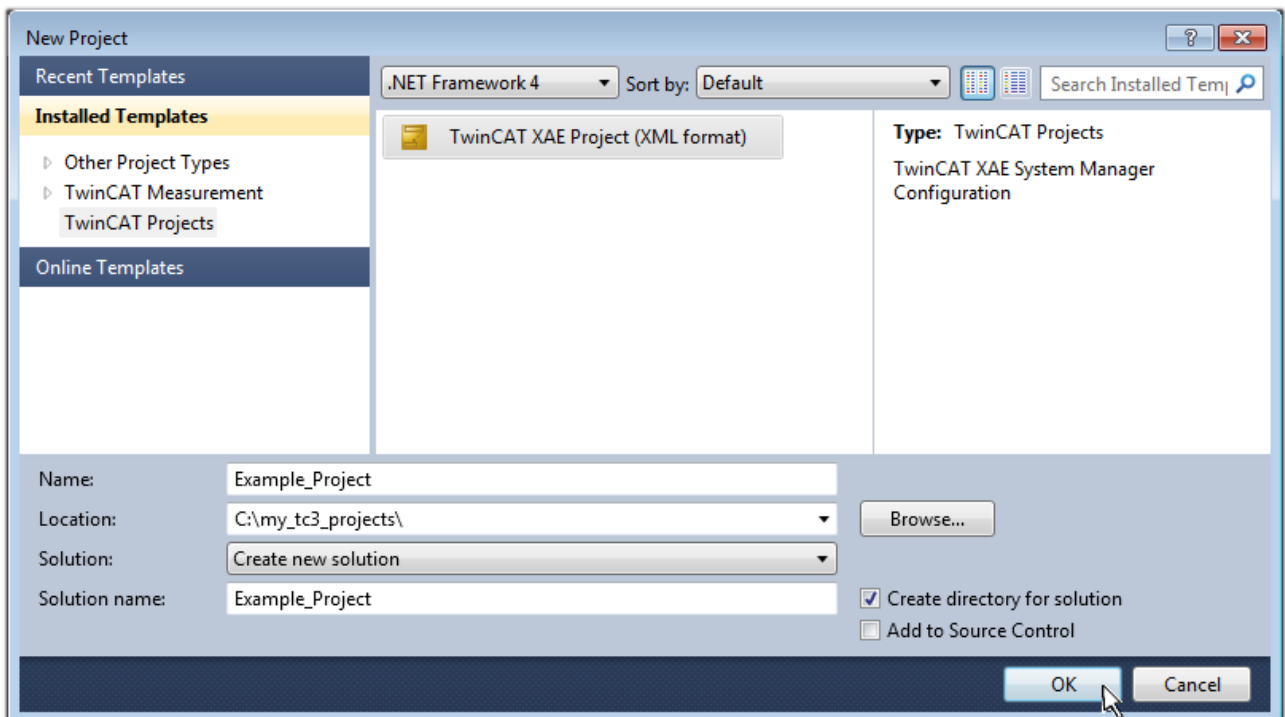


Fig. 45: Create new TwinCAT project

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

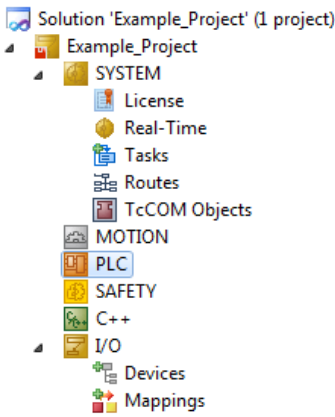
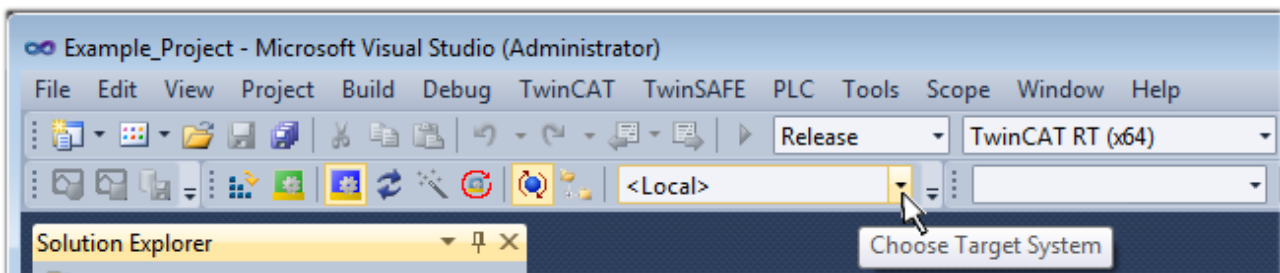


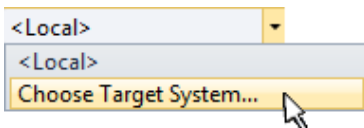
Fig. 46: New TwinCAT3 project in the project folder explorer

Generally, TwinCAT can be used in local or remote mode. Once the TwinCAT system including the user interface (standard) is installed on the respective PLC, TwinCAT can be used in local mode and thereby the next step is “Insert Device [▶ 62]”.

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



expand the pull-down menu:



and open the following window:

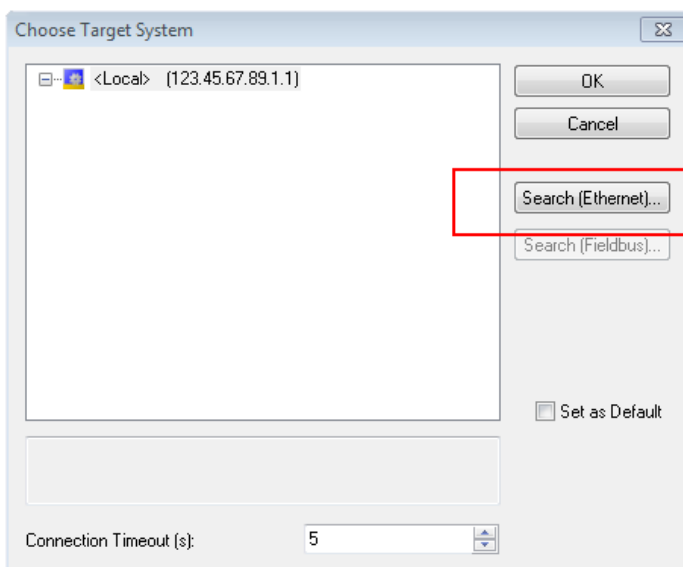


Fig. 47: Selection dialog: Choose the target system

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

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

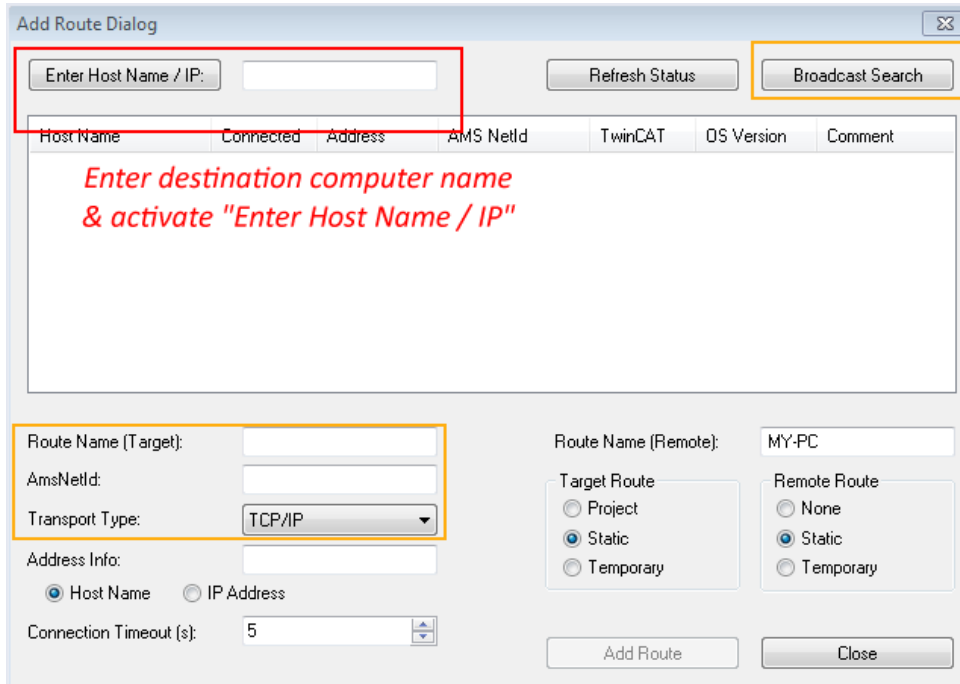
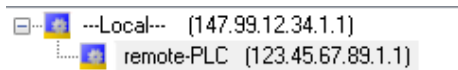


Fig. 48: Specify the PLC for access by the TwinCAT System Manager: selection of the target system


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




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

**Adding devices**

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

element “I/O”, then right-click to open a context menu and select “Scan” or start the action via  in the

menu bar. The TwinCAT System Manager may first have to be set to “Config mode” via  or via the menu “TwinCAT” → “Restart TwinCAT (Config mode)”.

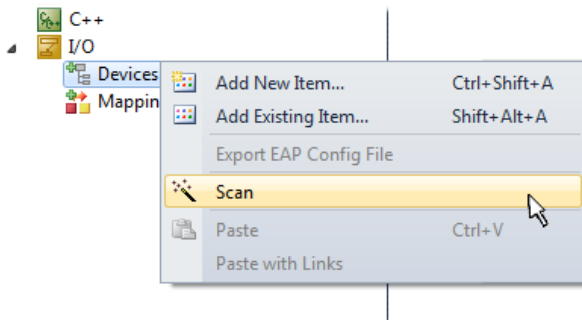


Fig. 49: Select “Scan”

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

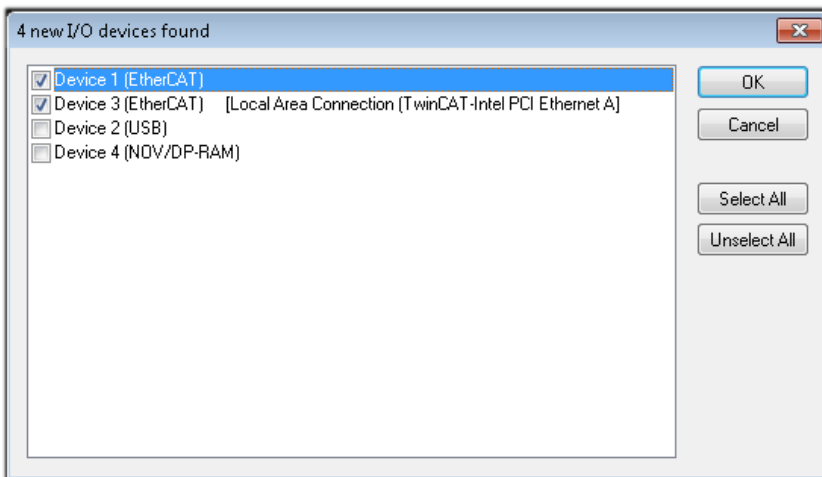


Fig. 50: Automatic detection of I/O devices: selection the devices to be integrated

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

Based on the [sample configuration \[▶ 47\]](#) described at the beginning of this section, the result is as follows:

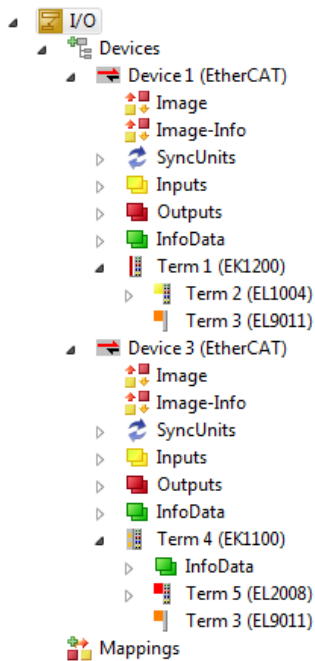


Fig. 51: Mapping of the configuration in VS shell of the TwinCAT3 environment

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

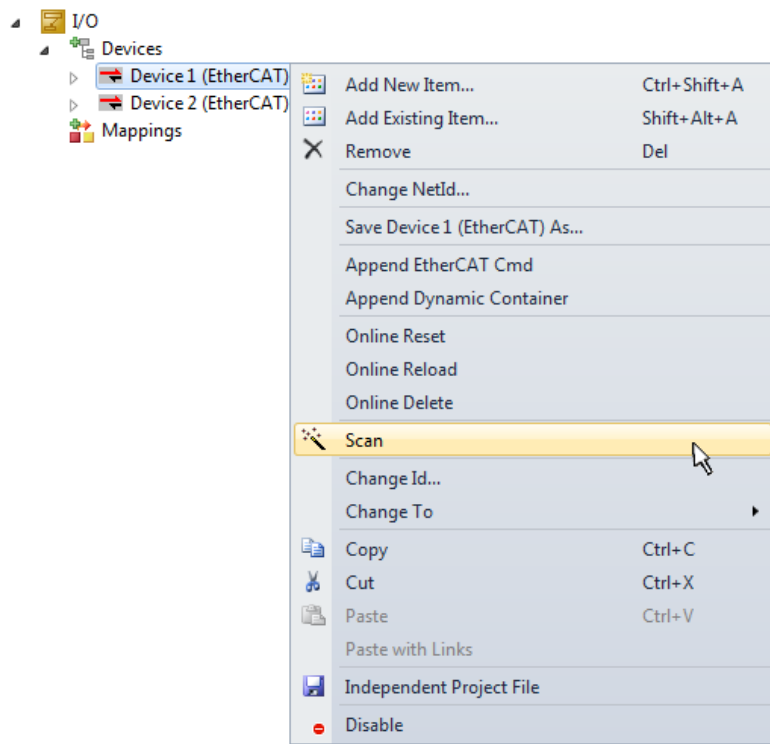


Fig. 52: Reading of individual terminals connected to a device

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

### Programming the PLC

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

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

The following section refers to Structured Text (ST).

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



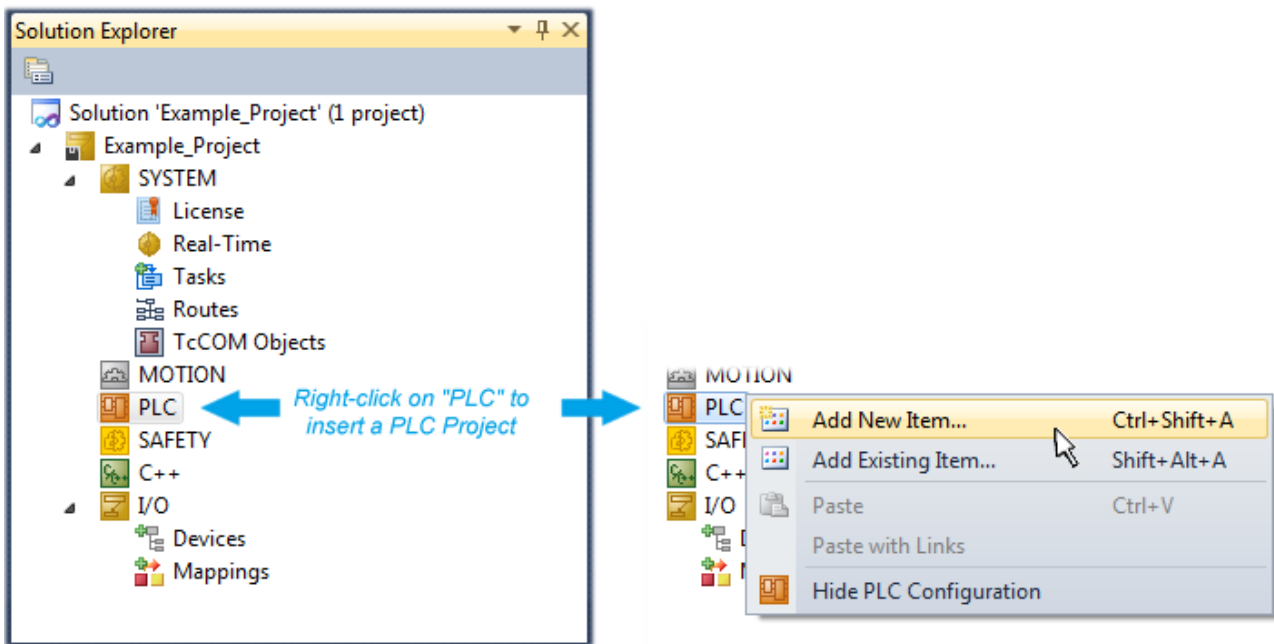


Fig. 53: Adding the programming environment in “PLC”

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

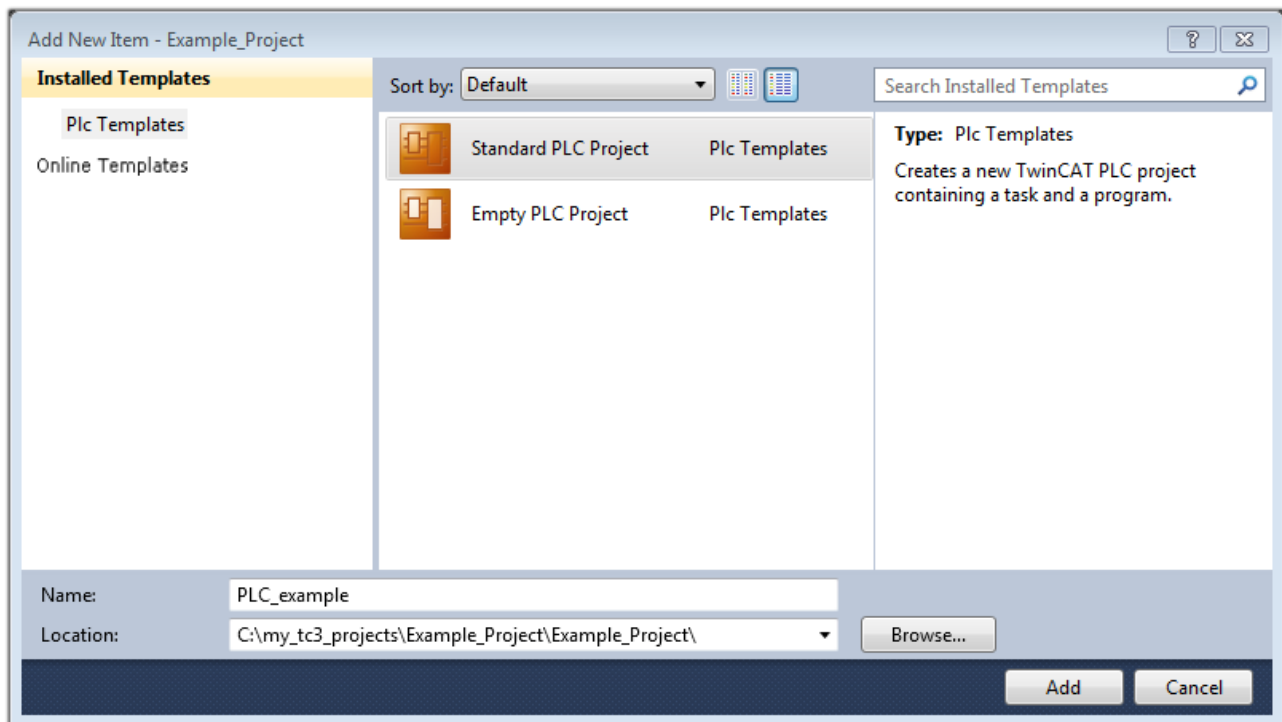


Fig. 54: Specifying the name and directory for the PLC programming environment

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

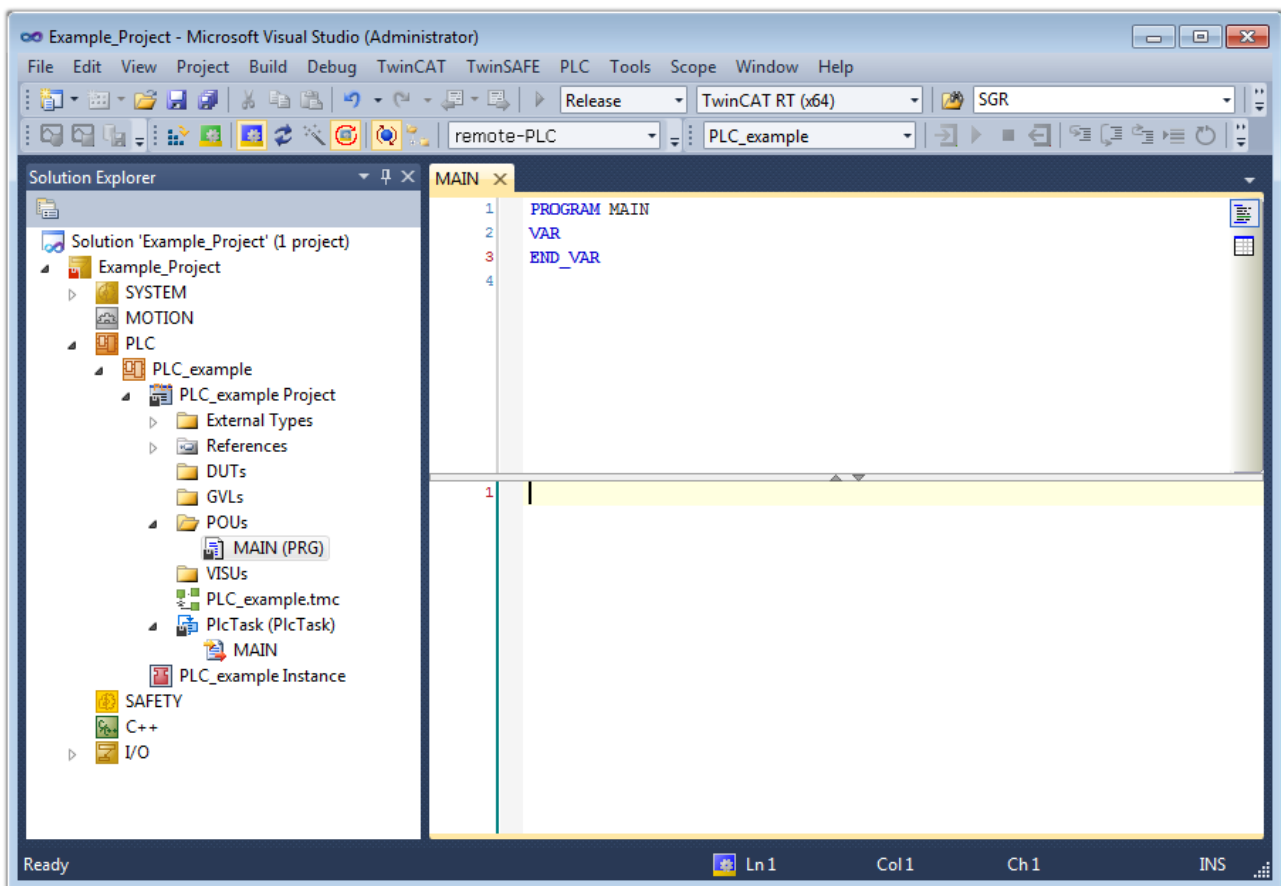


Fig. 55: Initial “Main” program of the standard PLC project

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

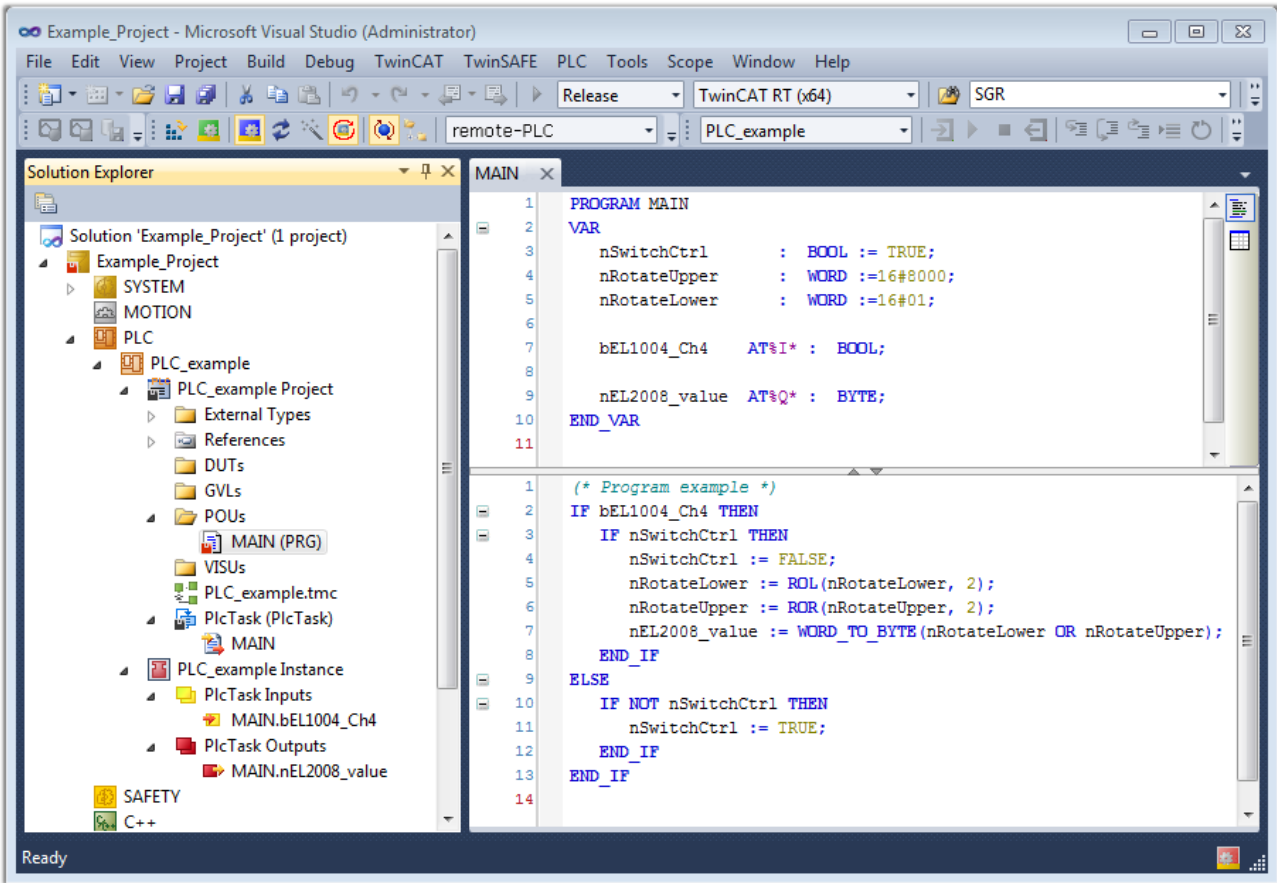


Fig. 56: Sample program with variables after a compile process (without variable integration)

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

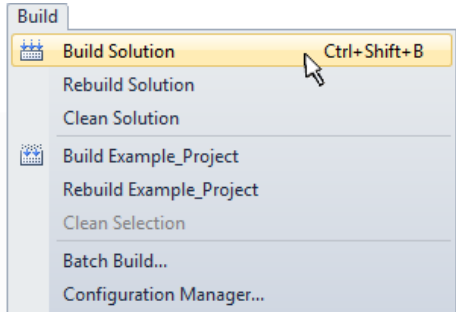
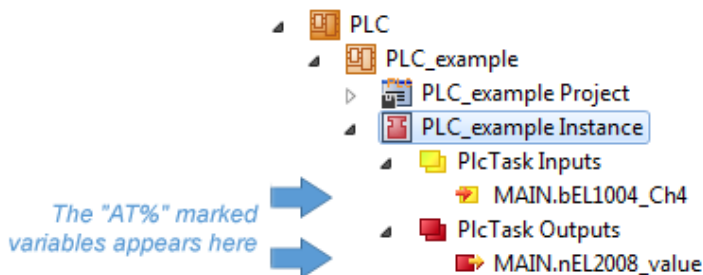


Fig. 57: Start program compilation

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



**Assigning variables**

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

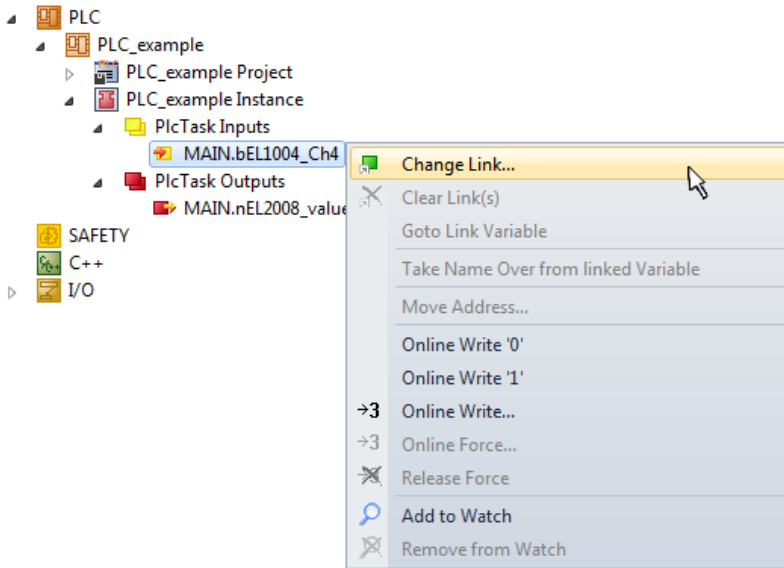


Fig. 58: Creating the links between PLC variables and process objects

In the window that opens, the process object for the variable “bEL1004\_Ch4” of type BOOL can be selected from the PLC configuration tree:

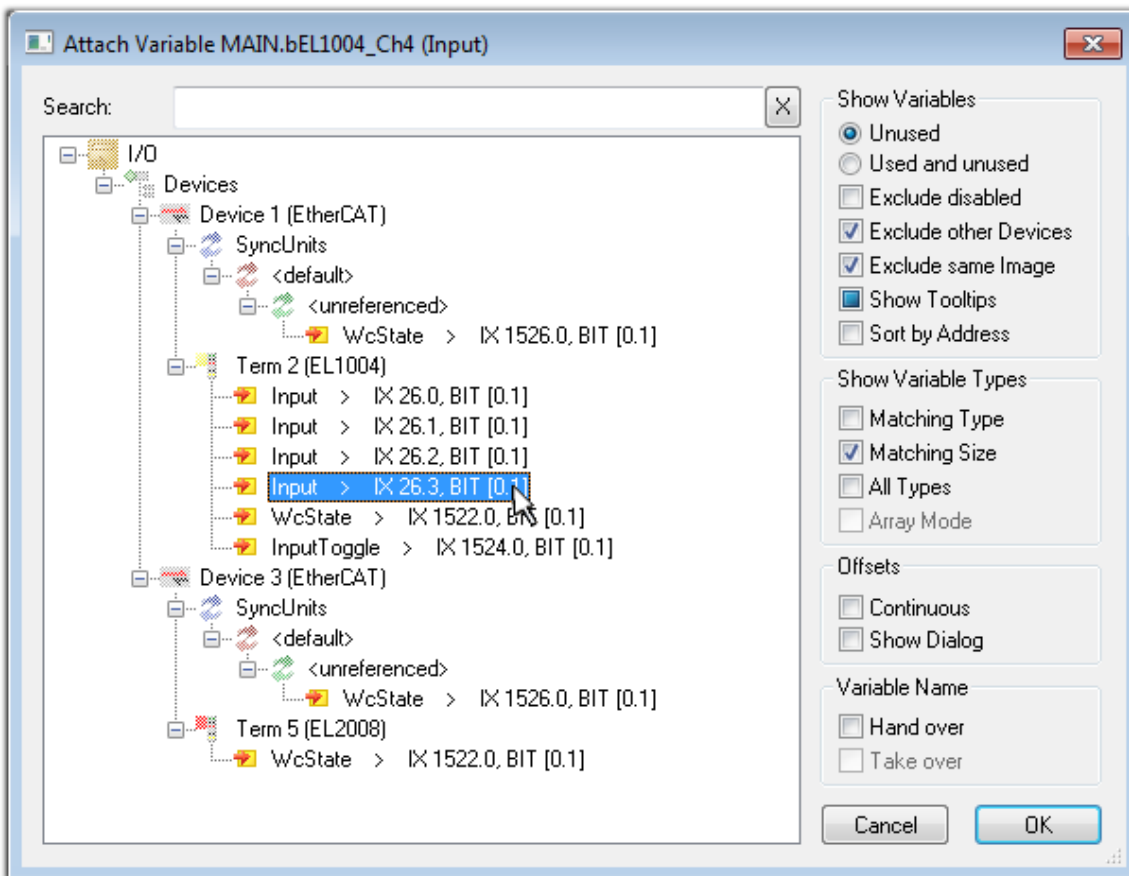


Fig. 59: Selecting PDO of type BOOL

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

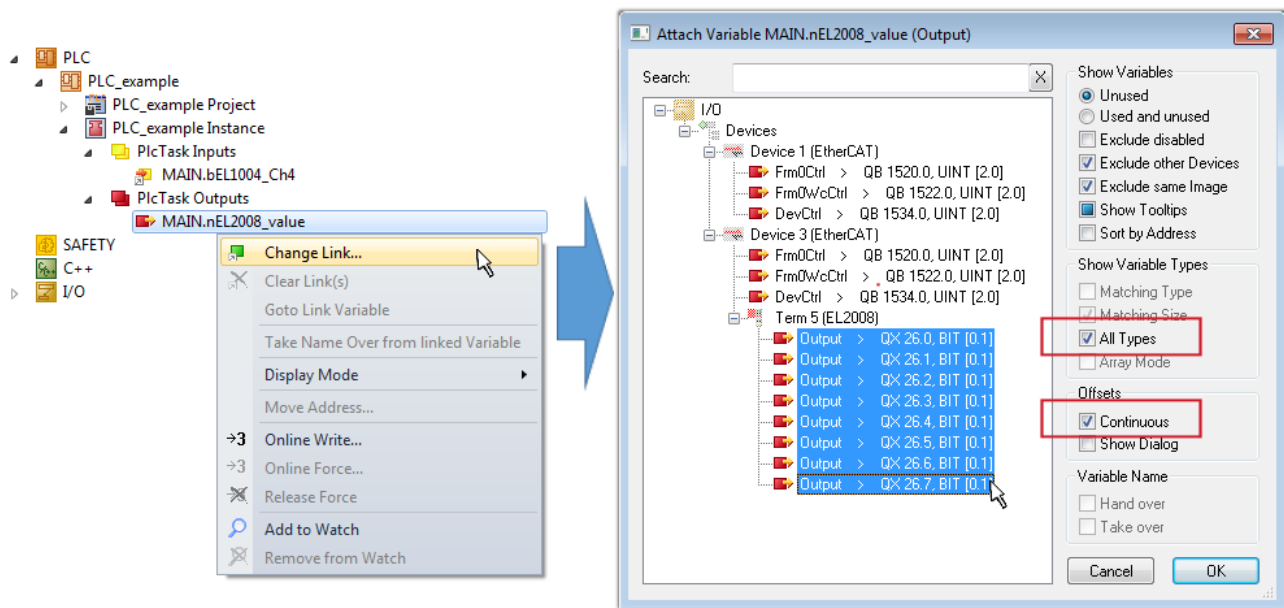



Fig. 60: Selecting several PDOs simultaneously: activate “Continuous” and “All types”

Note that the “Continuous” checkbox was also activated. This is designed to allocate the bits contained in the byte of the variable “nEL2008\_value” sequentially to all eight selected output bits of the EL2008 terminal. In this way it is possible to subsequently address all eight outputs of the terminal in the program with a byte corresponding to bit 0 for channel 1 to bit 7 for channel 8 of the PLC. A special symbol (  ) at the yellow or red object of the variable indicates that a link exists. The links can also be checked by selecting a “Goto Link Variable” from the context menu of a variable. The object opposite, in this case the PDO, is automatically selected:

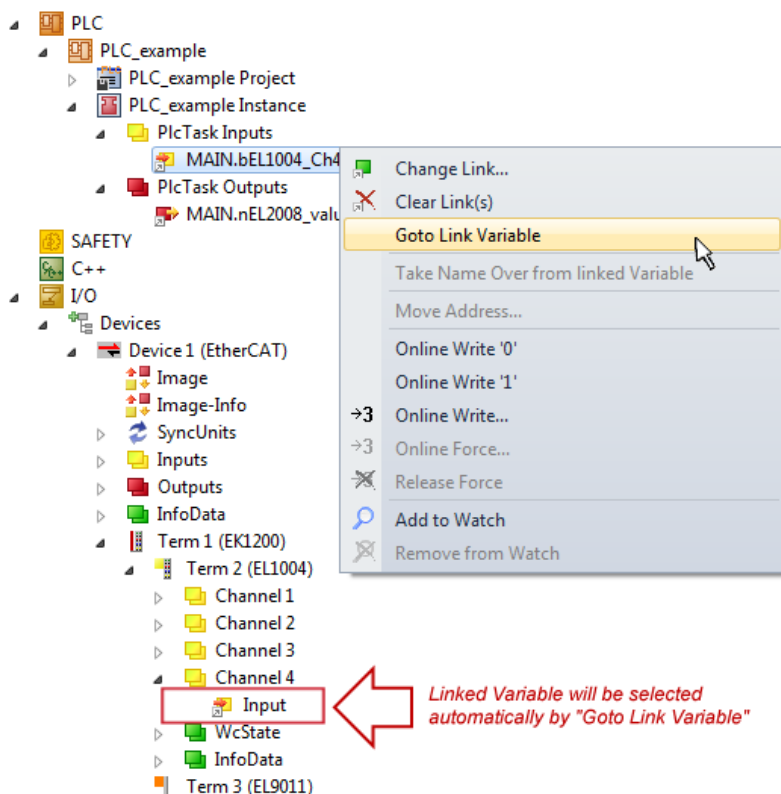


Fig. 61: Application of a “Goto Link” variable, using “MAIN.bEL1004\_Ch4” as a sample

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

similar PDO, it is possible to allocate this a set of bit-standardized variables (type "BOOL"). Here, too, a "Goto Link Variable" from the context menu of a PDO can be executed in the other direction, so that the respective PLC instance can then be selected.

### ● Note on the type of variable assignment



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

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

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

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

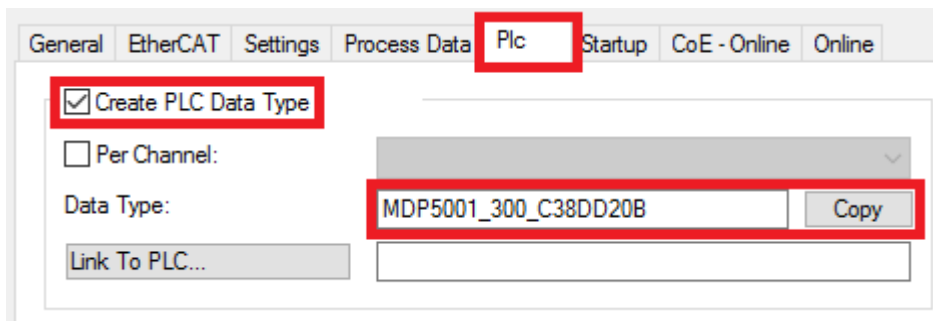


Fig. 62: Creating a PLC data type

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

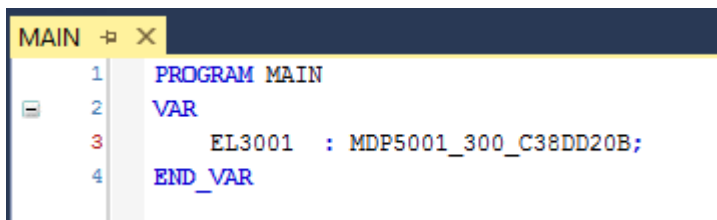


Fig. 63: Instance\_of\_struct

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

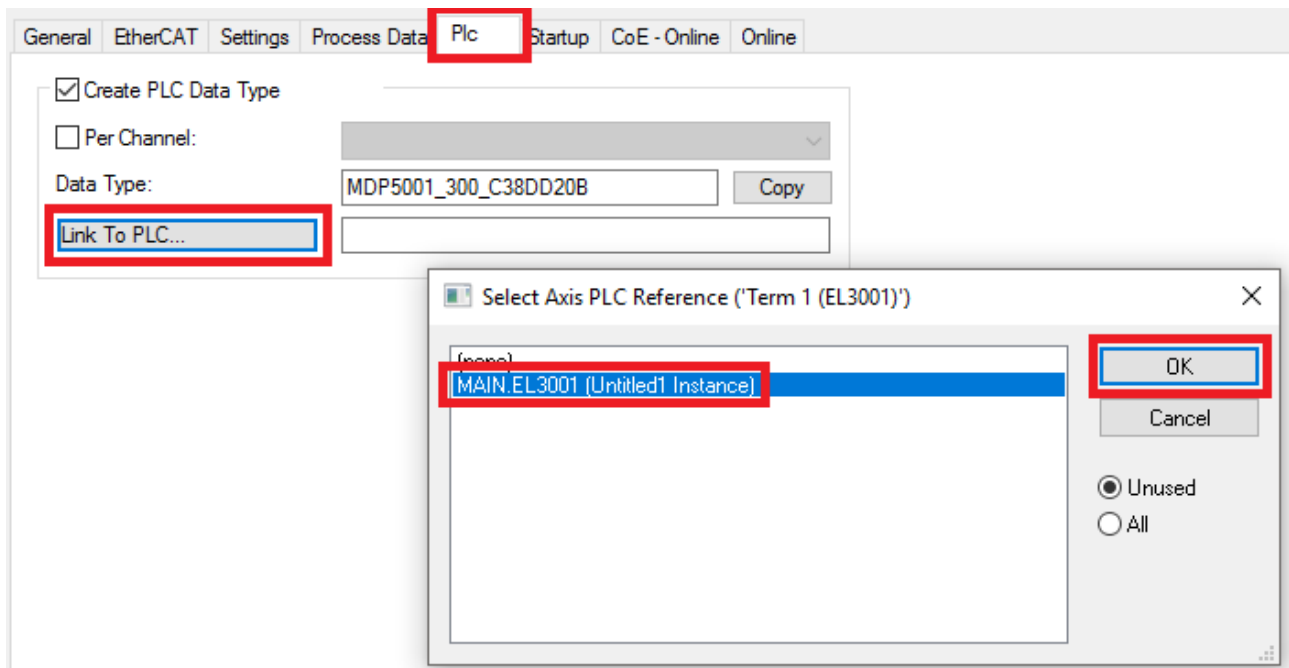


Fig. 64: Linking the structure

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

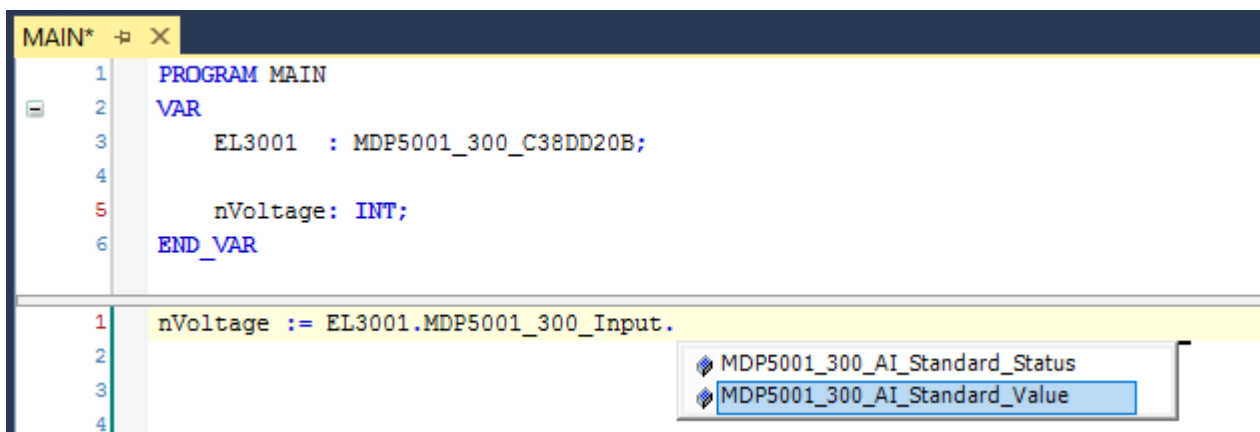
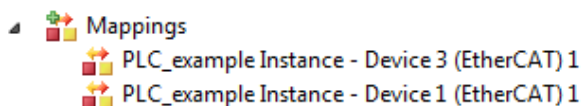


Fig. 65: Reading a variable from the structure of the process data


**Activation of the configuration**

The allocation of PDO to PLC variables has now established the connection from the controller to the inputs


and outputs of the terminals. The configuration can now be activated with  or via the menu under “TwinCAT” in order to transfer settings of the development environment to the runtime system. Confirm the messages “Old configurations are overwritten!” and “Restart TwinCAT system in Run mode” with “OK”. The corresponding assignments can be seen in the project folder explorer:




A few seconds later the corresponding status of the Run mode is displayed in the form of a rotating symbol

 at the bottom right of the VS shell development environment. The PLC system can then be started as described below.

### Starting the controller

Select the menu option “PLC” → “Login” or click on  to link the PLC with the real-time system and load the control program for execution. This results in the message *No program on the controller! Should the new program be loaded?*, which should be acknowledged with “Yes”. The runtime environment is ready for

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

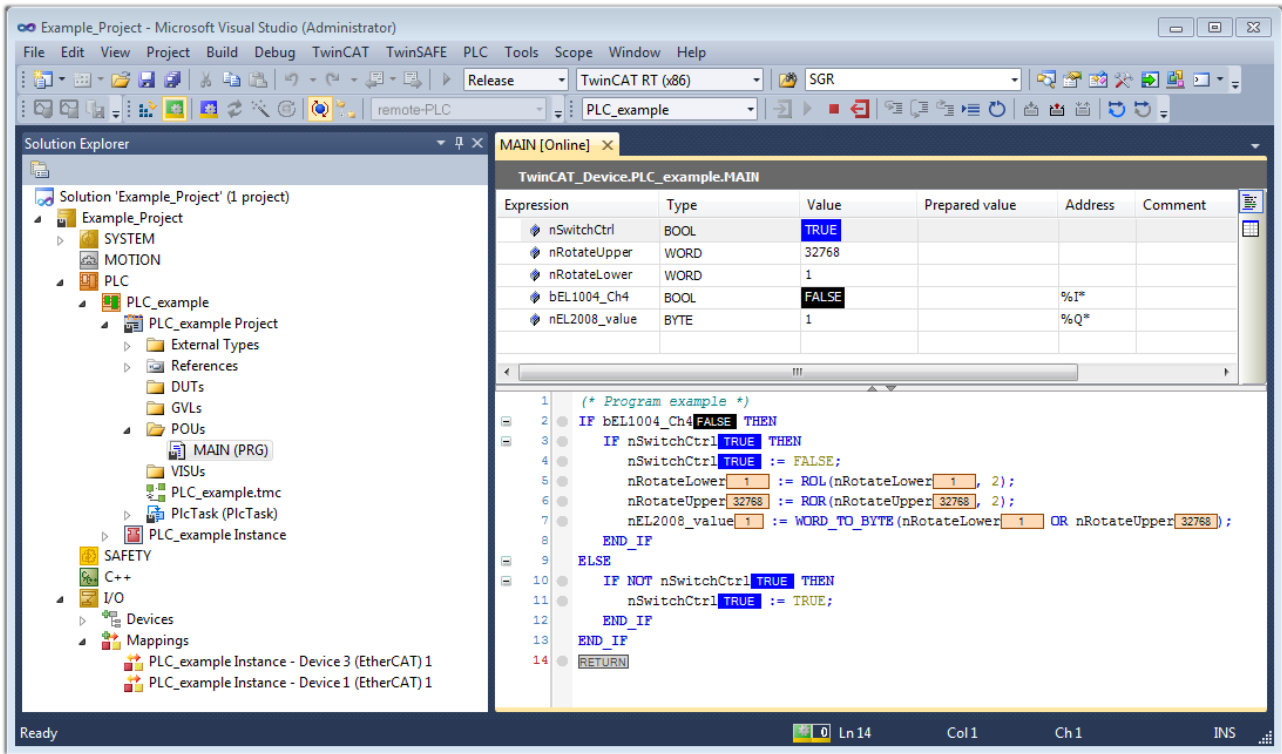


Fig. 66: TwinCAT development environment (VS shell): logged-in, after program startup

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



## 5.2 TwinCAT Development Environment

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

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

### Details:

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

### Additional features:

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

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

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

### 5.2.1 Installation of the TwinCAT real-time driver

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

This can be done in several ways. One option is described here.

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

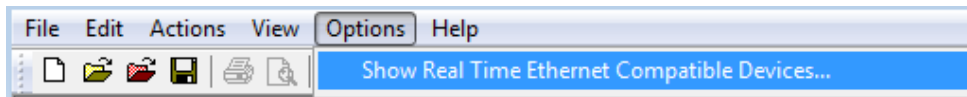


Fig. 67: System Manager “Options” (TwinCAT 2)

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

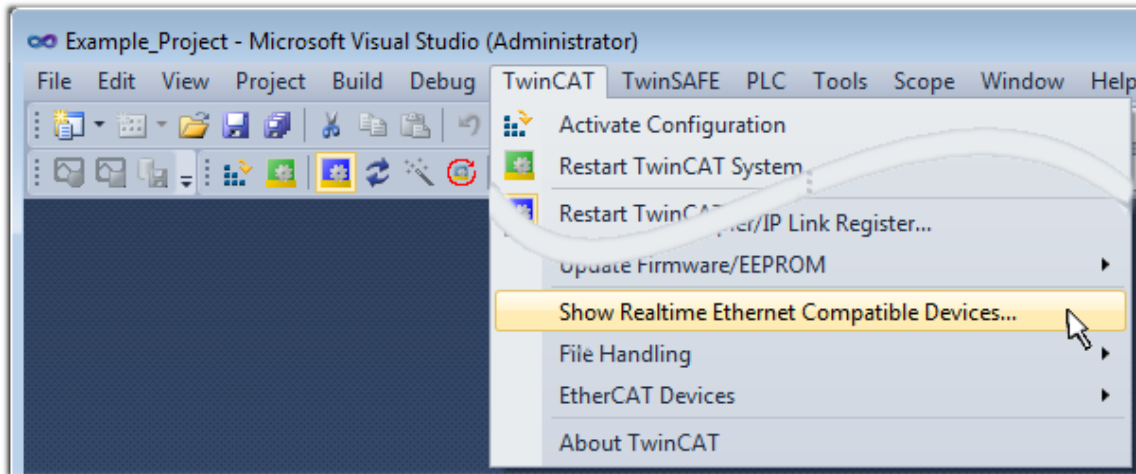


Fig. 68: Call up under VS Shell (TwinCAT 3)

The following dialog appears:

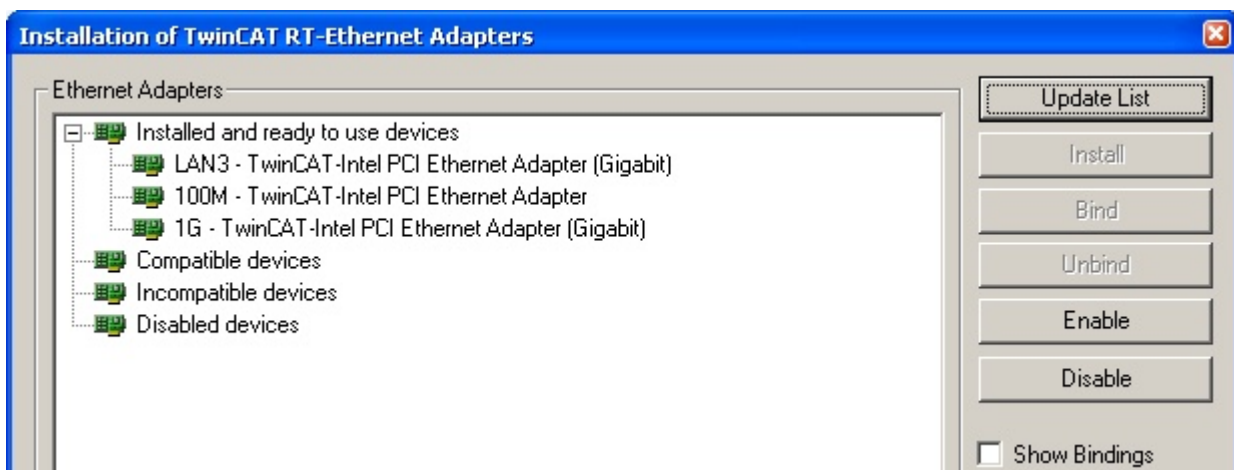


Fig. 69: Overview of network interfaces

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

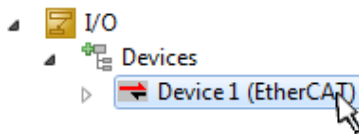
A Windows warning regarding the unsigned driver can be ignored.

**Alternatively** an EtherCAT-device can be inserted first of all as described in chapter [Offline configuration creation, section “Creating the EtherCAT device” \[▶ 84\]](#) in order to view the compatible ethernet ports via its EtherCAT properties (tab “Adapter”, button “Compatible Devices...”):



Fig. 70: EtherCAT device properties(TwinCAT 2): click on “Compatible Devices...” of tab “Adapte”

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)

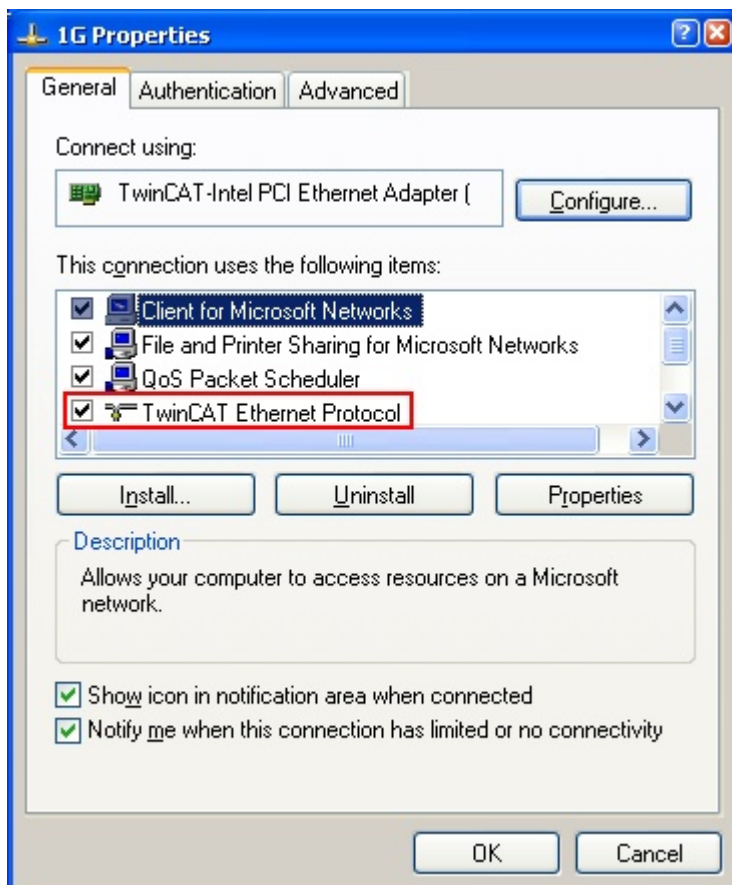


Fig. 71: Windows properties of the network interface

A correct setting of the driver could be:

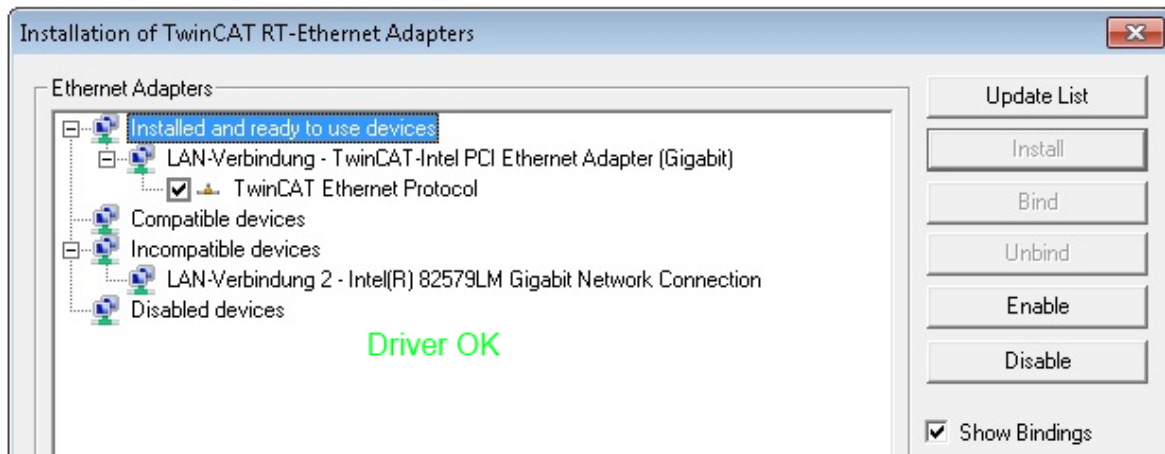


Fig. 72: Exemplary correct driver setting for the Ethernet port

Other possible settings have to be avoided:

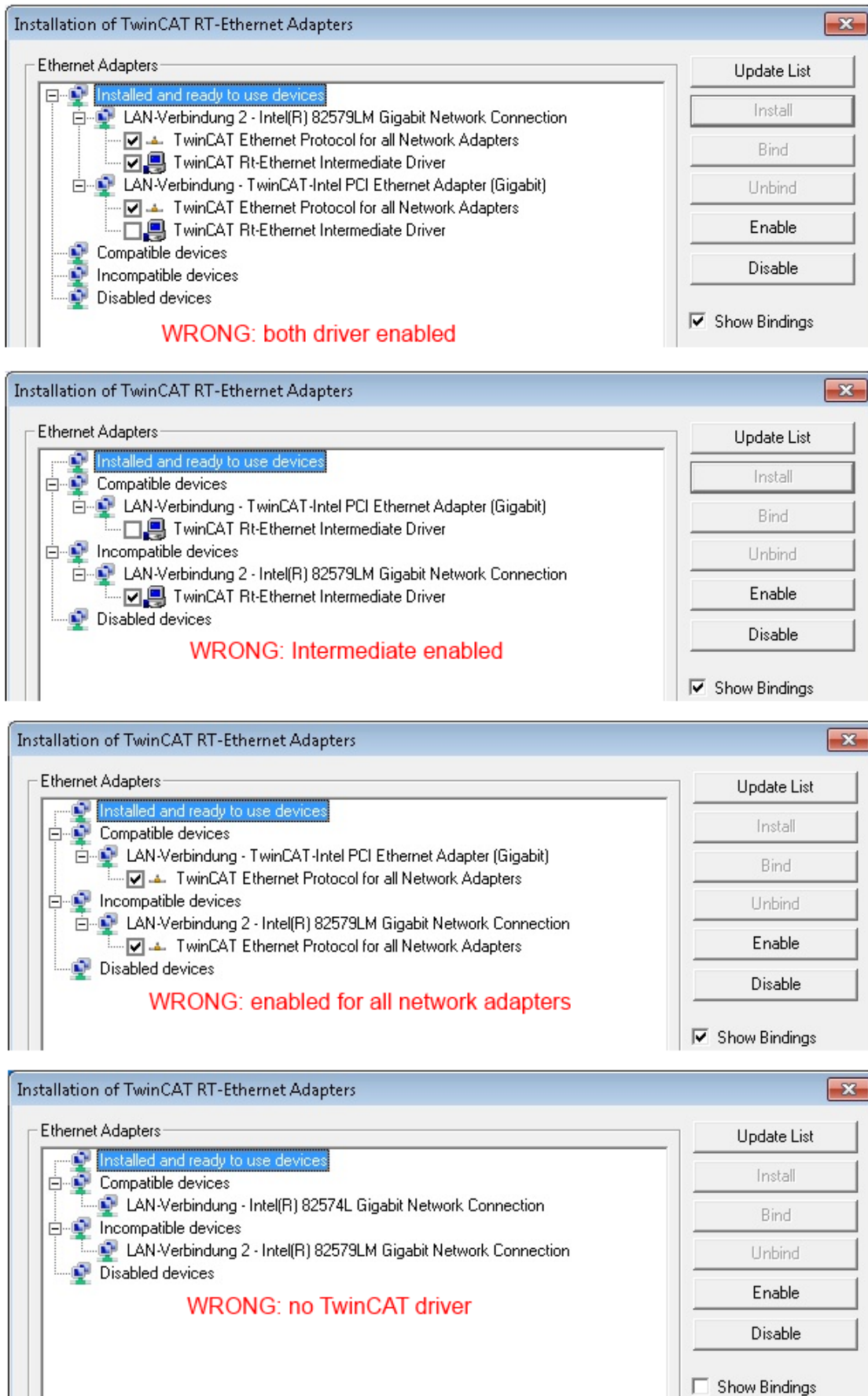


Fig. 73: Incorrect driver settings for the Ethernet port

## IP address of the port used

### **i** IP address/DHCP

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

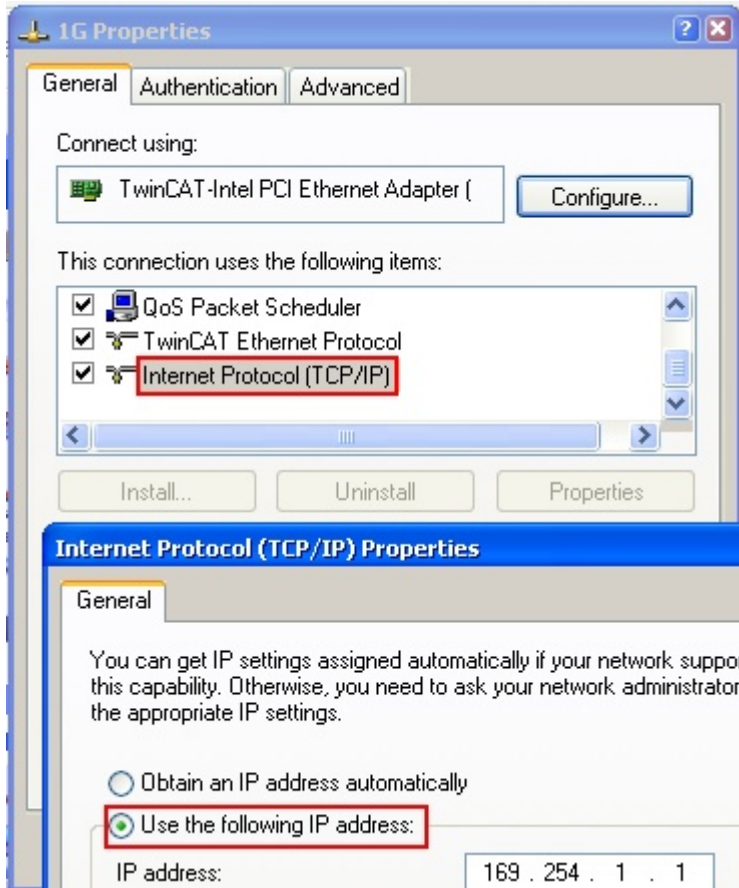


Fig. 74: TCP/IP setting for the Ethernet port

## 5.2.2 Notes regarding ESI device description

### Installation of the latest ESI device description

The TwinCAT EtherCAT master/System Manager needs the device description files for the devices to be used in order to generate the configuration in online or offline mode. The device descriptions are contained in the so-called ESI files (EtherCAT Slave Information) in XML format. These files can be requested from the respective manufacturer and are made available for download. An \*.xml file may contain several device descriptions.

The ESI files for Beckhoff EtherCAT devices are available on the [Beckhoff website](#).

The ESI files should be stored in the TwinCAT installation directory.

Default settings:

- **TwinCAT 2:** C:\TwinCAT\IO\EtherCAT
- **TwinCAT 3:** C:\TwinCAT\3.1\Config\Io\EtherCAT

The files are read (once) when a new System Manager window is opened, if they have changed since the last time the System Manager window was opened.

A TwinCAT installation includes the set of Beckhoff ESI files that was current at the time when the TwinCAT build was created.

For TwinCAT 2.11/TwinCAT 3 and higher, the ESI directory can be updated from the System Manager, if the programming PC is connected to the Internet; by

- **TwinCAT 2:** Option → “Update EtherCAT Device Descriptions”
- **TwinCAT 3:** TwinCAT → EtherCAT Devices → “Update Device Descriptions (via ETG Website)...”

The [TwinCAT ESI Updater \[► 83\]](#) 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”

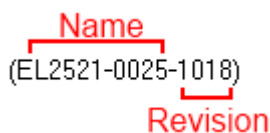


Fig. 75: Identifier structure

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

## 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.

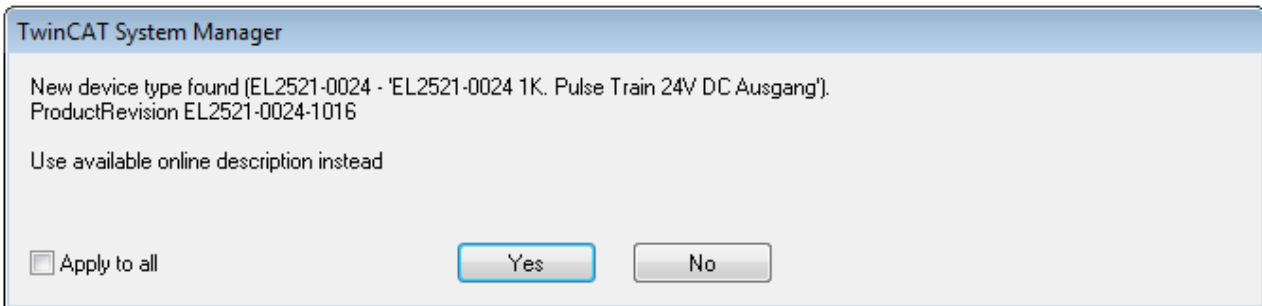


Fig. 76: OnlineDescription information window (TwinCAT 2)

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

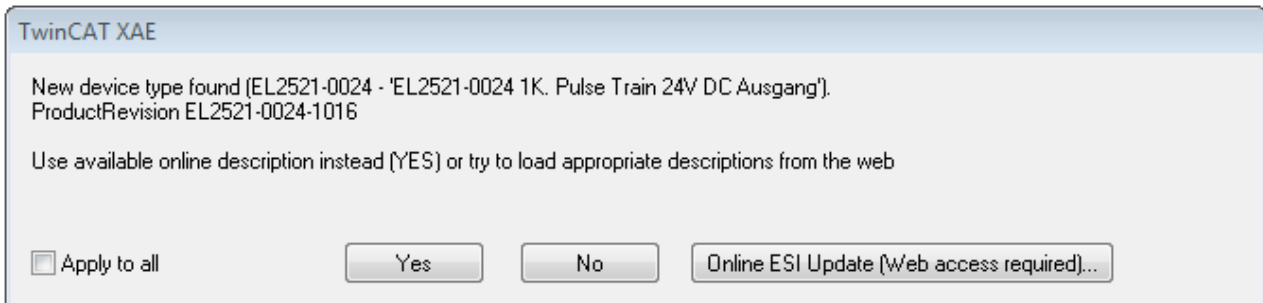


Fig. 77: Information window OnlineDescription (TwinCAT 3)

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

### NOTE

#### Changing the “usual” configuration through a scan

- ✓ If a scan discovers a device that is not yet known to TwinCAT, distinction has to be made between two cases. Taking the example here of the EL2521-0000 in the revision 1019
  - a) no ESI is present for the EL2521-0000 device at all, either for the revision 1019 or for an older revision. The ESI must then be requested from the manufacturer (in this case Beckhoff).
  - b) an ESI is present for the EL2521-0000 device, but only in an older revision, e.g. 1018 or 1017. In this case an in-house check should first be performed to determine whether the spare parts stock allows the integration of the increased revision into the configuration at all. A new/higher revision usually also brings along new features. If these are not to be used, work can continue without reservations with the previous revision 1018 in the configuration. This is also stated by the Beckhoff compatibility rule.

Refer in particular to the chapter “[General notes on the use of Beckhoff EtherCAT IO components](#)” and for manual configuration to the chapter “[Offline configuration creation \[► 84\]](#)”.

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

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



OnlineDescriptionCache00000002.xml

Fig. 78: File OnlineDescription.xml created by the System Manager

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

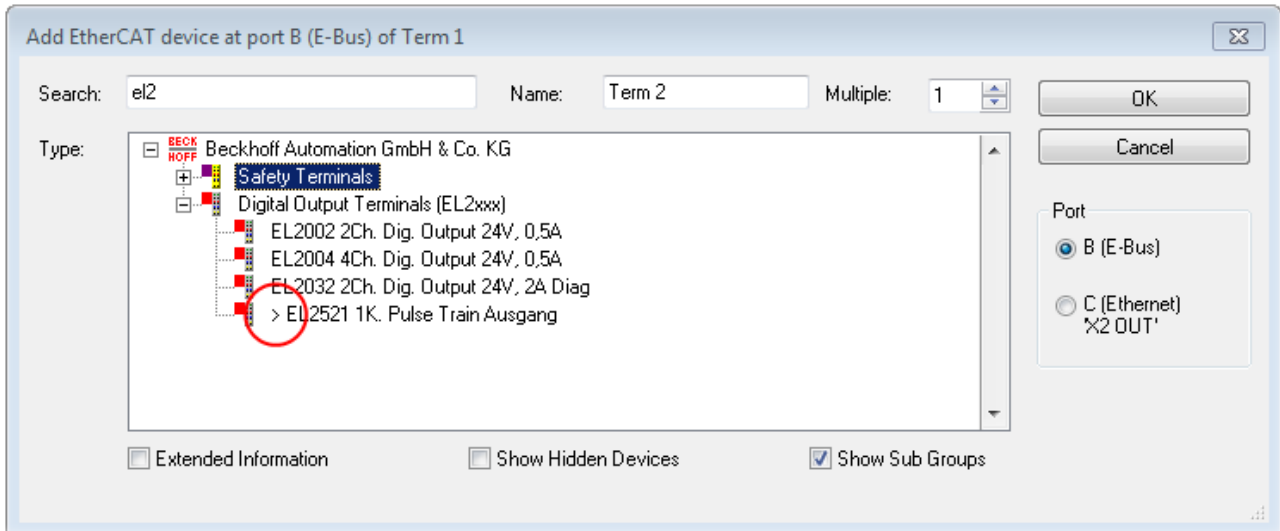


Fig. 79: Indication of an online recorded ESI of EL2521 as an example

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

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

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

**i OnlineDescription for TwinCAT 3.x**

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

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

(Please note the language settings of the OS!)

You have to delete this file, too.

**Faulty ESI file**

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

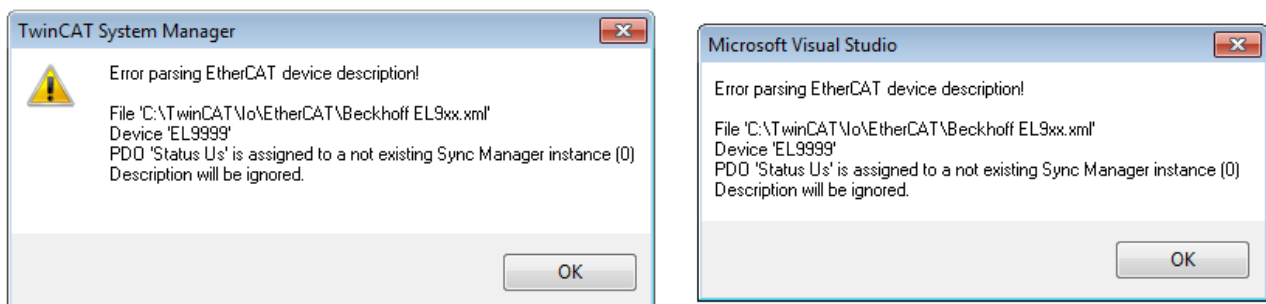


Fig. 80: Information window for faulty ESI file (left: TwinCAT 2; right: TwinCAT 3)

Reasons may include:

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

### 5.2.3 TwinCAT ESI Updater

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

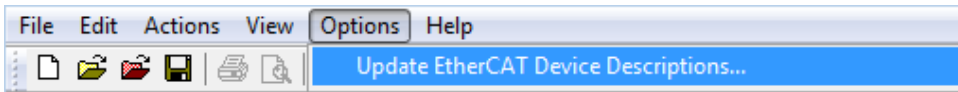


Fig. 81: Using the ESI Updater (>= TwinCAT 2.11)

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

Selection under TwinCAT 3:

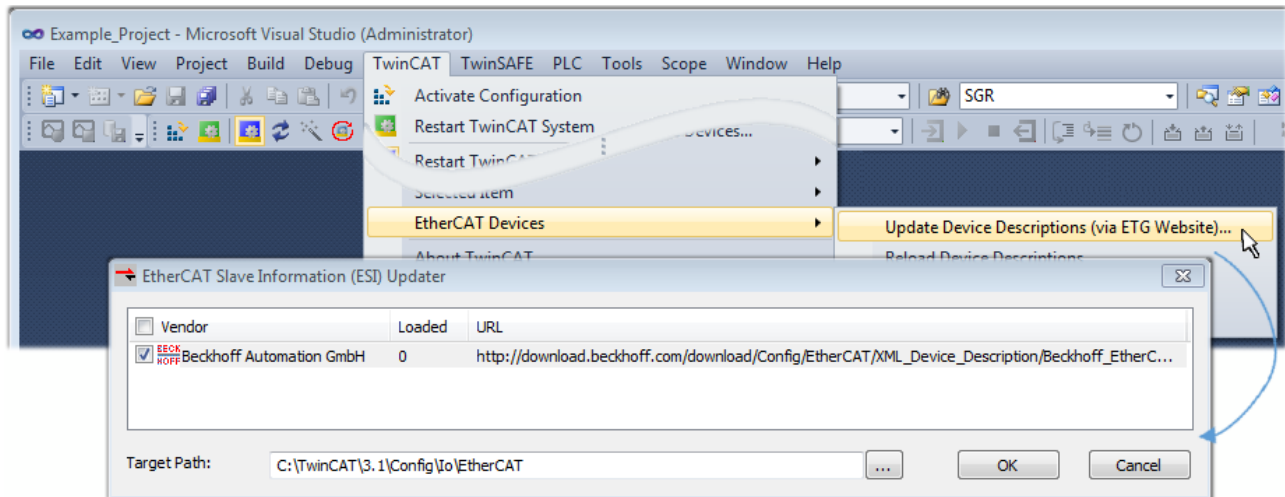


Fig. 82: Using the ESI Updater (TwinCAT 3)

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

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

### 5.2.4 Distinction between Online and Offline

The distinction between online and offline refers to the presence of the actual I/O environment (drives, terminals, EJ-modules). If the configuration is to be prepared in advance of the system configuration as a programming system, e.g. on a laptop, this is only possible in “Offline configuration” mode. In this case all components have to be entered manually in the configuration, e.g. based on the electrical design.

If the designed control system is already connected to the EtherCAT system and all components are energised and the infrastructure is ready for operation, the TwinCAT configuration can simply be generated through “scanning” from the runtime system. This is referred to as online configuration.

In any case, during each startup the EtherCAT master checks whether the slaves it finds match the configuration. This test can be parameterised in the extended slave settings. Refer to note “Installation of the latest ESI-XML device description” [▶ 79].

**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 [▶ 89] (Ethernet port at the IPC)
- detecting the connected EtherCAT devices [▶ 90]. This step can be carried out independent of the preceding step
- troubleshooting [▶ 93]

The scan with existing configuration [▶ 94] can also be carried out for comparison.

## 5.2.5 OFFLINE configuration creation

### Creating the EtherCAT device

Create an EtherCAT device in an empty System Manager window.

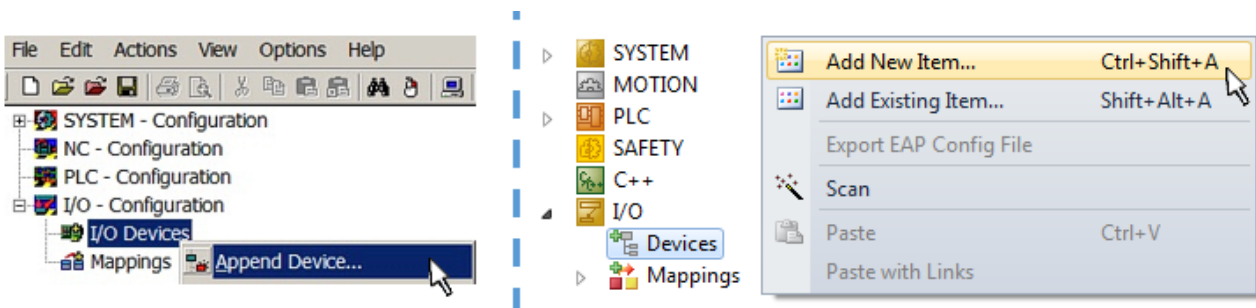


Fig. 83: Append EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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”.

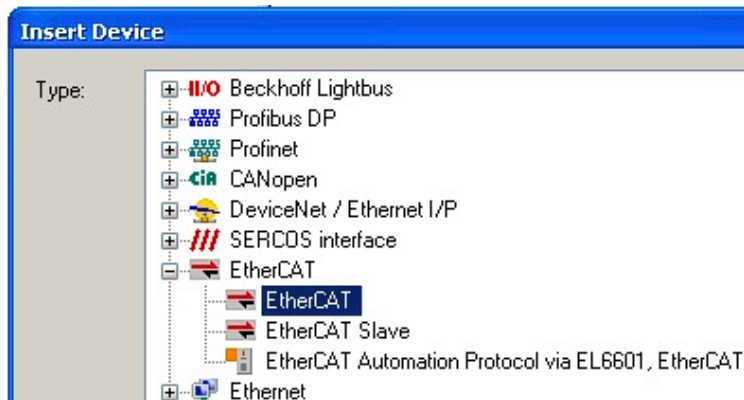


Fig. 84: Selecting the EtherCAT connection (TwinCAT 2.11, TwinCAT 3)

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

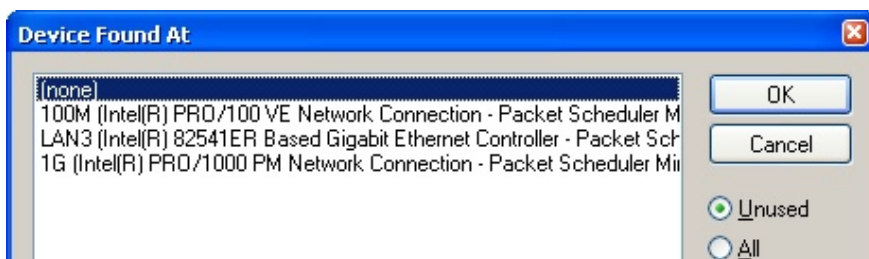


Fig. 85: Selecting the Ethernet port

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)”.

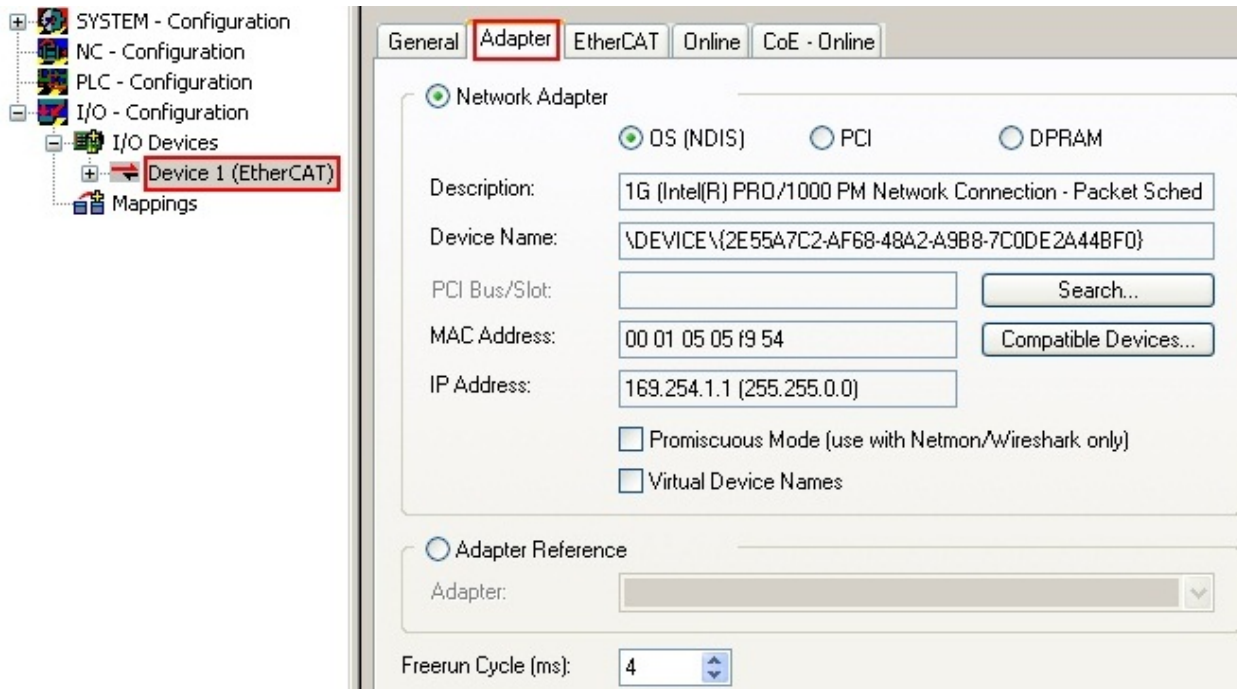
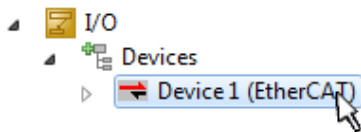


Fig. 86: EtherCAT device properties (TwinCAT 2)

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



**i** **Selecting the Ethernet port**

Ethernet ports can only be selected for EtherCAT devices for which the TwinCAT real-time driver is installed. This has to be done separately for each port. Please refer to the respective [installation page \[p. 73\]](#).

**Defining EtherCAT slaves**

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

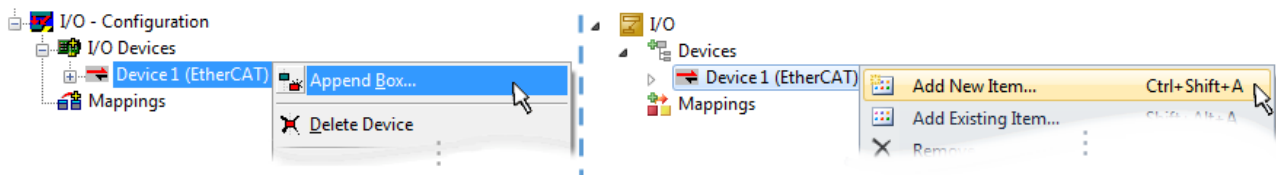


Fig. 87: Appending EtherCAT devices (left: TwinCAT 2; right: TwinCAT 3)

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

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

Overview of physical layer

- “Ethernet”: cable-based 100BASE-TX: EK couplers, EP boxes, devices with RJ45/M8/M12 connector

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

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

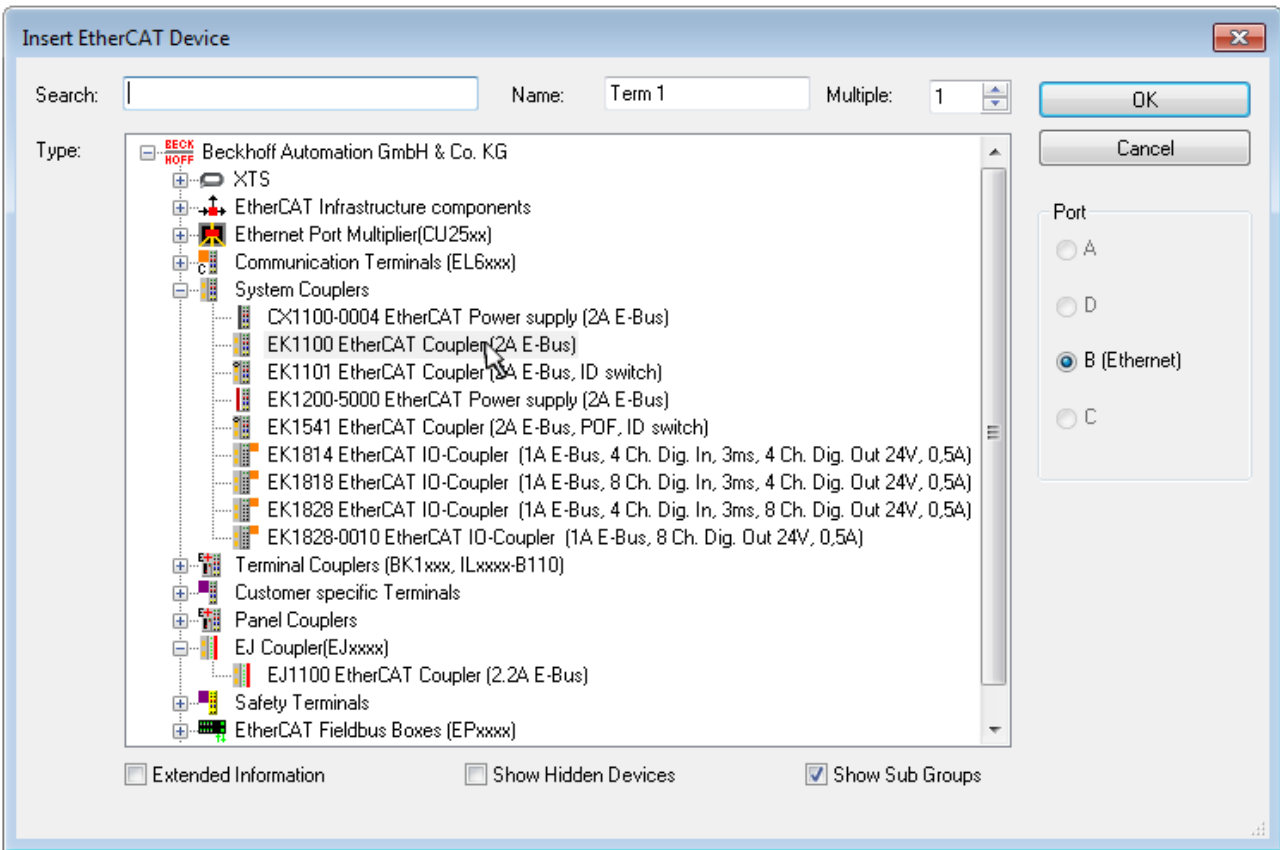


Fig. 88: Selection dialog for new EtherCAT device

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”.

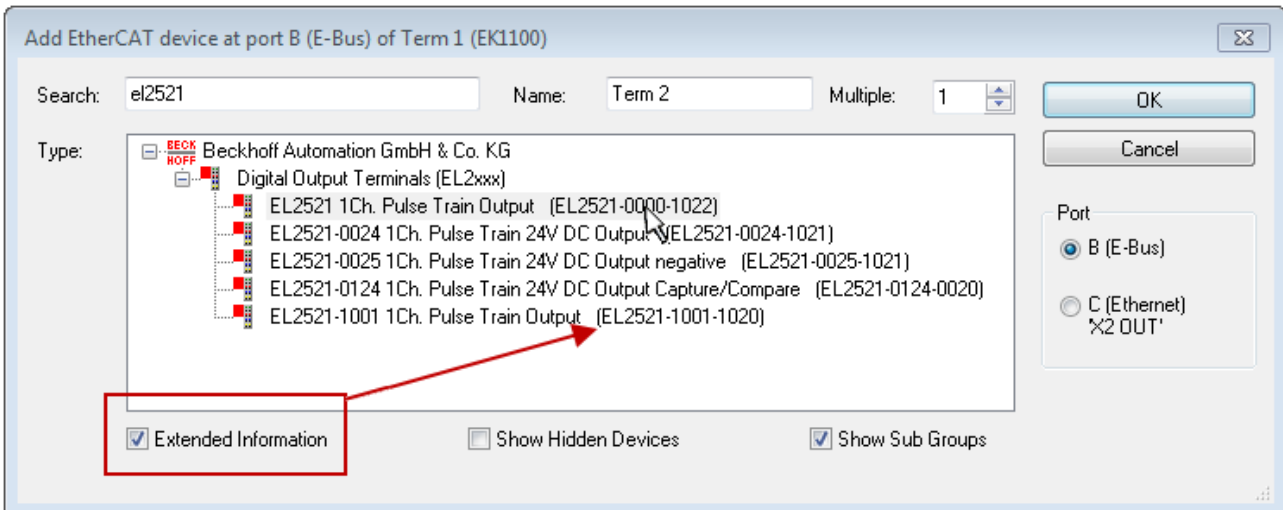


Fig. 89: Display of device revision

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”.

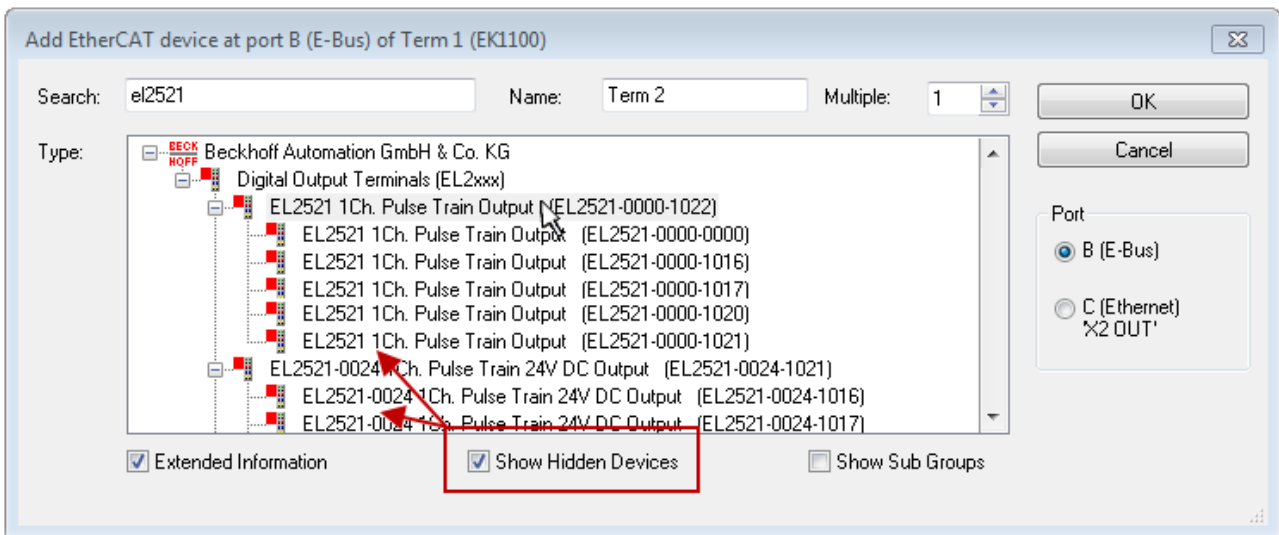


Fig. 90: Display of previous revisions

**i Device selection based on revision, compatibility**

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

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

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

**Example**

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

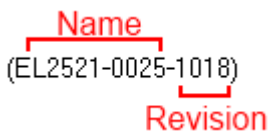


Fig. 91: Name/revision of the terminal

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, ...

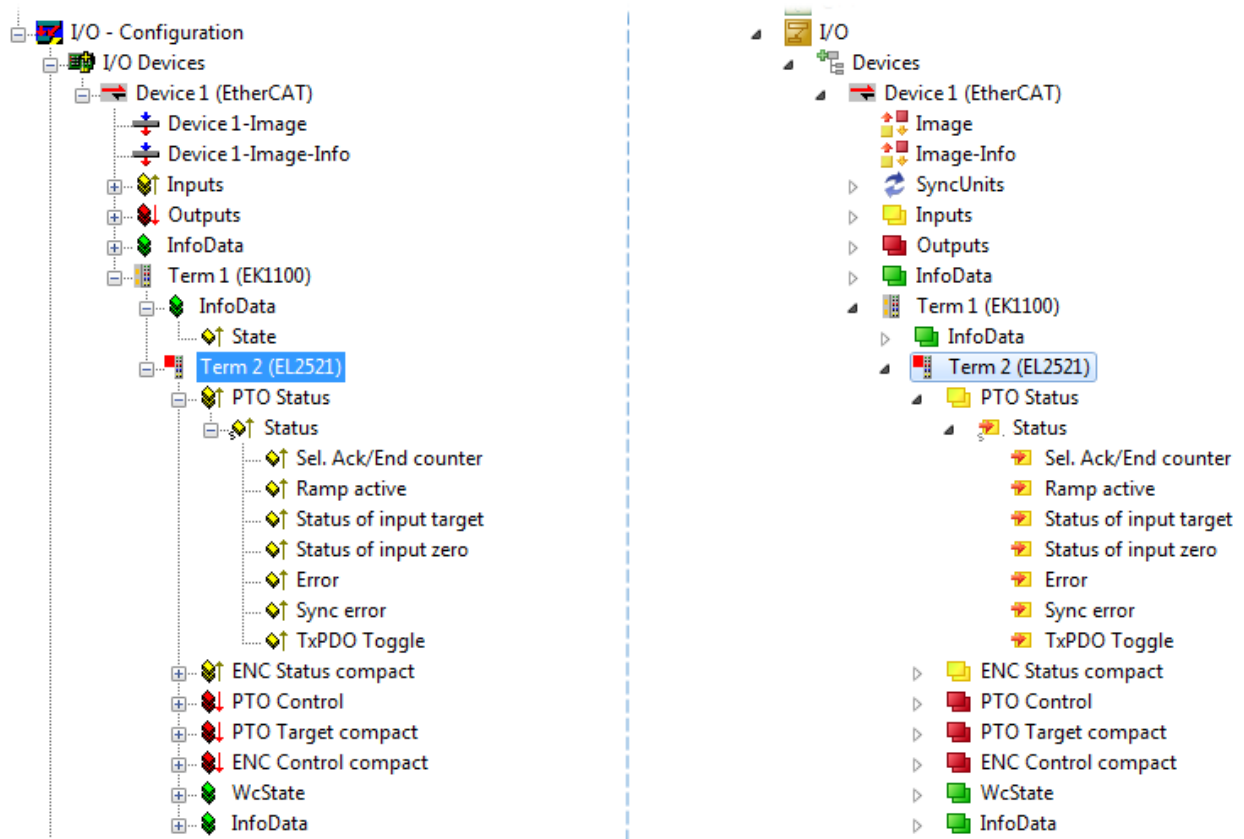




Fig. 92: EtherCAT terminal in the TwinCAT tree (left: TwinCAT 2; right: TwinCAT 3)





## 5.2.6 ONLINE configuration creation

### Detecting/scanning of the EtherCAT device

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



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

TwinCAT can be set into this mode:

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

### Online scanning in Config mode

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.

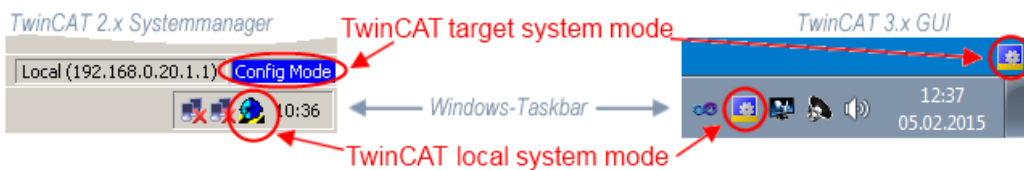


Fig. 93: Differentiation local/target system (left: TwinCAT 2; right: TwinCAT 3)

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

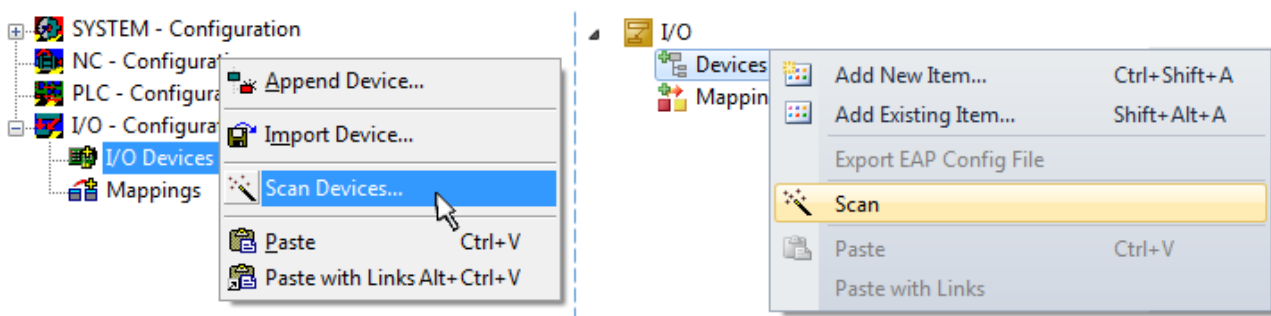


Fig. 94: Scan Devices (left: TwinCAT 2; right: TwinCAT 3)

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

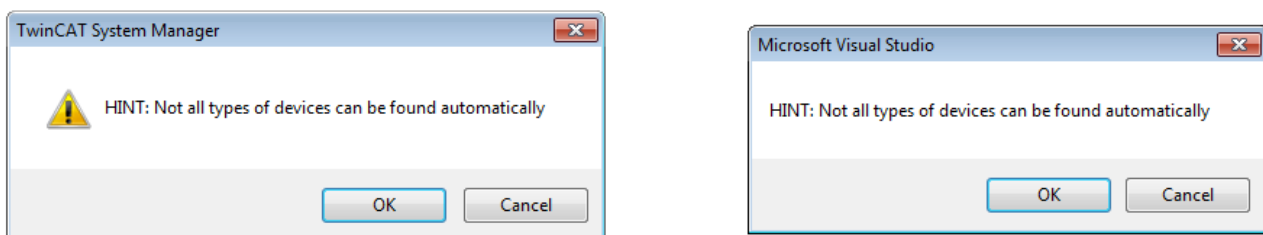


Fig. 95: Note for automatic device scan (left: TwinCAT 2; right: TwinCAT 3)

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” .

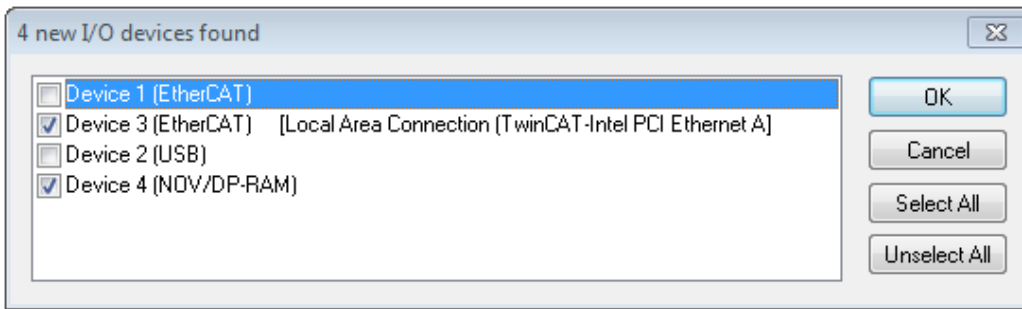


Fig. 96: Detected Ethernet devices

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](#) [▶ 73].

## 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.

**Name**  
(EL2521-0025-1018)  
**Revision**

Fig. 97: Example default state

## NOTE

### Slave scanning in practice in series machine production

The scanning function should be used with care. It is a practical and fast tool for creating an initial configuration as a basis for commissioning. In series machine production or reproduction of the plant, however, the function should no longer be used for the creation of the configuration, but if necessary for [comparison](#) [▶ 94] 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:

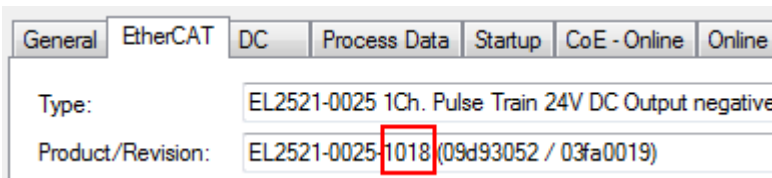


Fig. 98: Installing EthetCAT terminal with revision -1018

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 [► 94] 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:

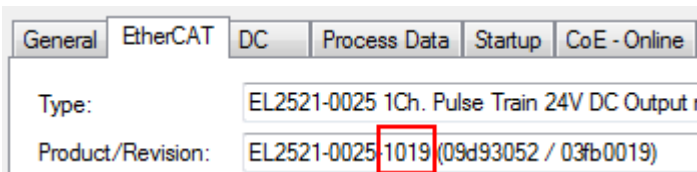


Fig. 99: Detection of EtherCAT terminal with revision -1019

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

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

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



Fig. 100: Scan query after automatic creation of an EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

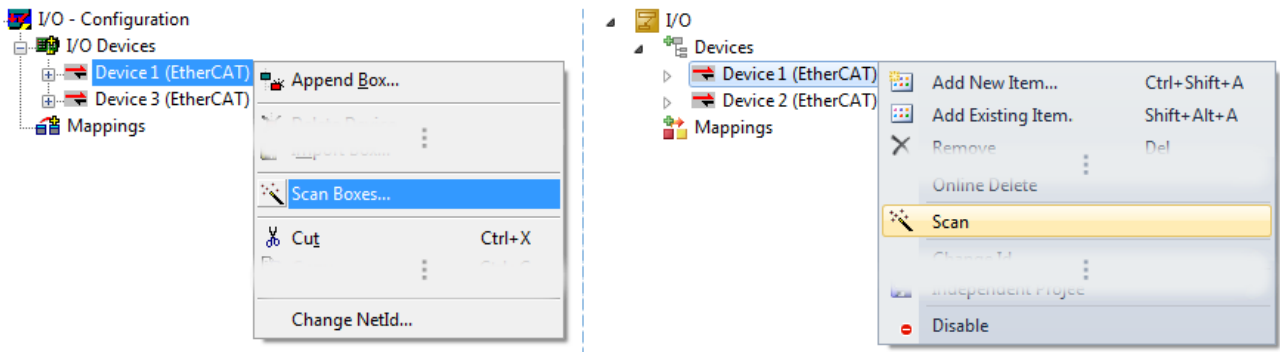


Fig. 101: Manual triggering of a device scan on a specified EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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

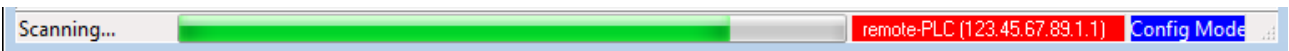


Fig. 102: Scan progress example by TwinCAT 2

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



Fig. 103: 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. 104: Displaying of “Free Run” and “Config Mode” toggling right below in the status bar



Fig. 105: 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*.

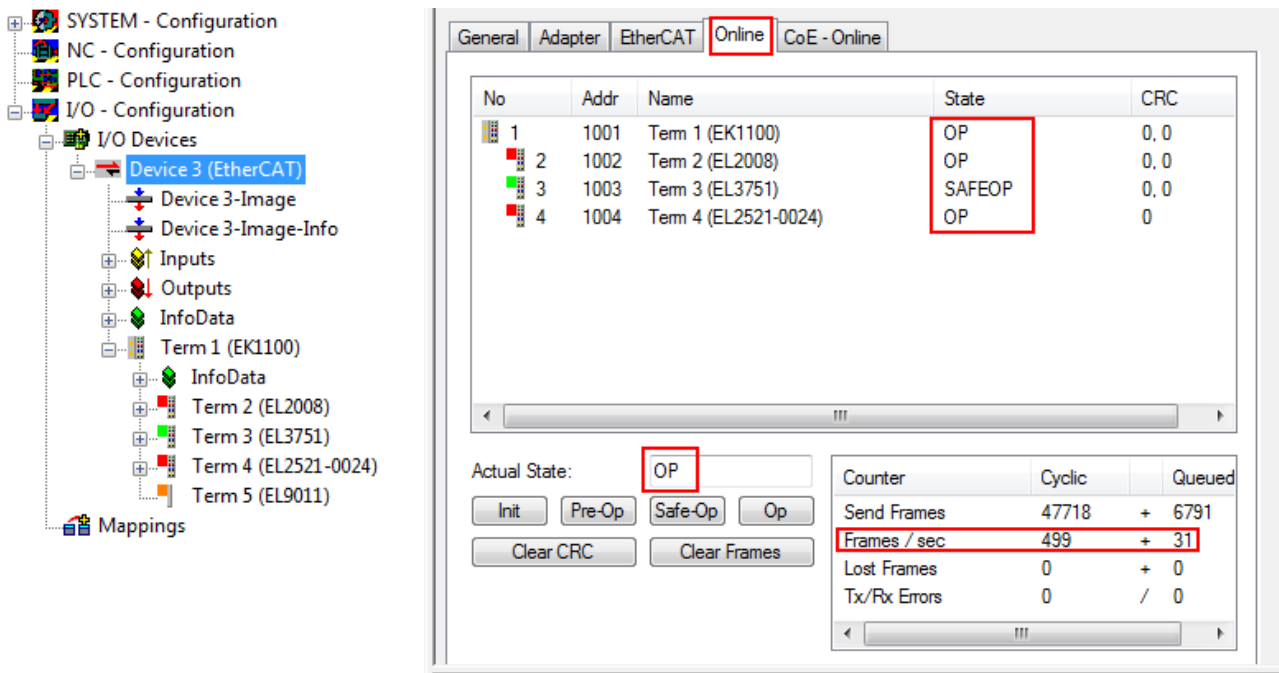


Fig. 106: Online display example

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 \[► 84\]](#).

### 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.

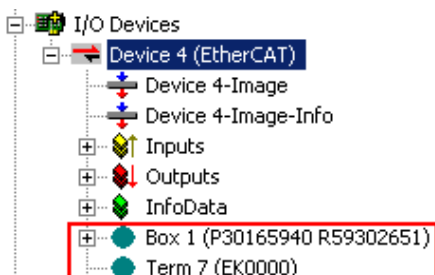


Fig. 107: Faulty identification

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

**Scan over existing Configuration**

**NOTE**

**Change of the configuration after comparison**

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

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



Fig. 108: 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.

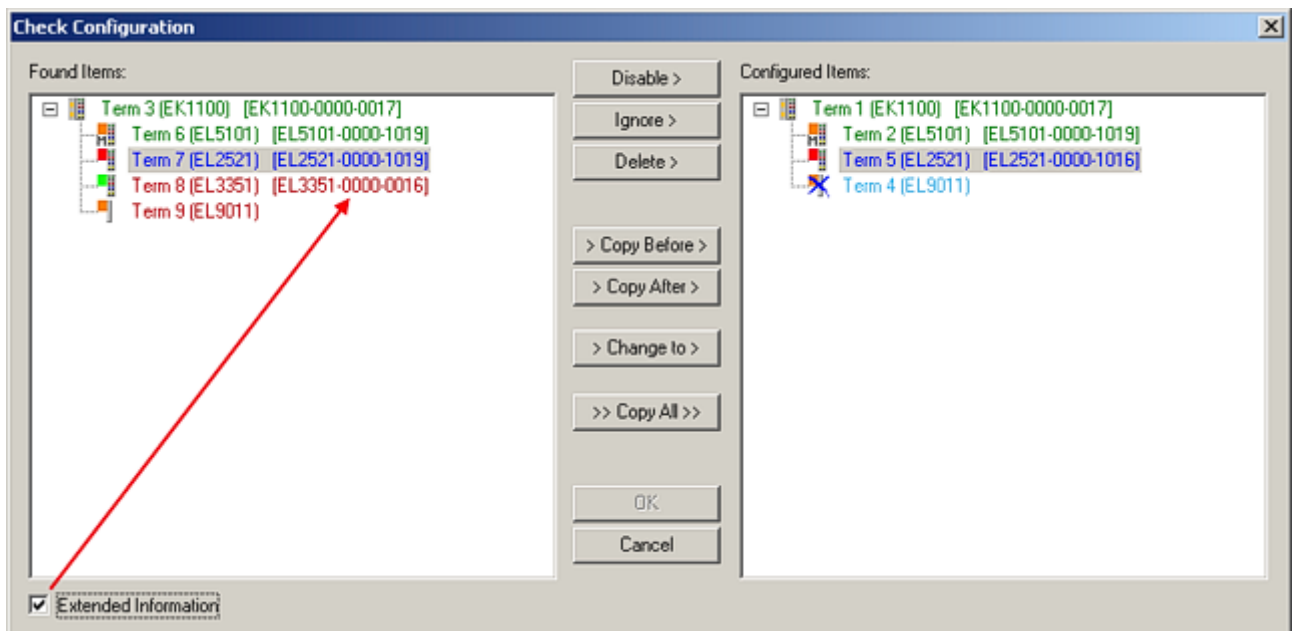


Fig. 109: Correction dialog

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.<br>If the found revision is higher than the configured revision, the slave may be used provided compatibility issues are taken into account.<br><br>If the found revision is lower than the configured revision, it is likely that the slave cannot be used. The found device may not support all functions that the master expects based on the higher revision number.   |
| light blue | This EtherCAT slave is ignored ("Ignore" button)  |
| red        | <ul style="list-style-type: none"> <li>This EtherCAT slave is not present on the other side.</li> <li>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.</li> </ul> |

**i Device selection based on revision, compatibility**

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

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

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

**Example**

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

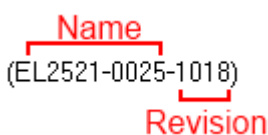


Fig. 110: Name/revision of the terminal

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, ...

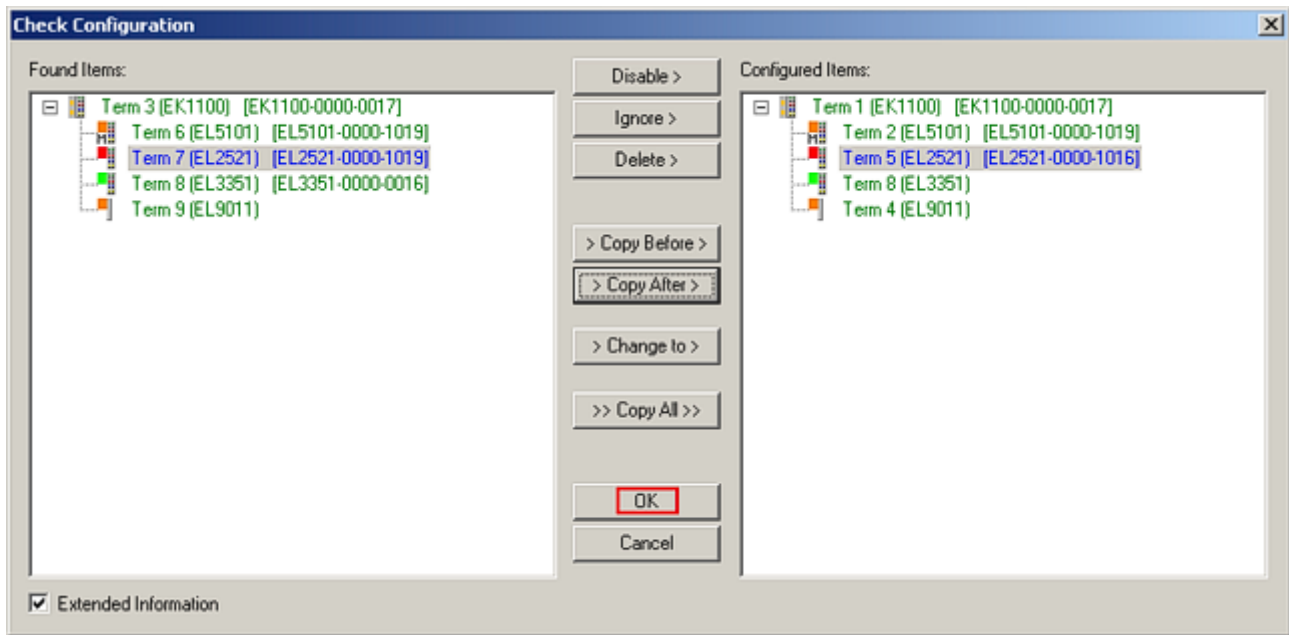


Fig. 111: Correction dialog with modifications

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.

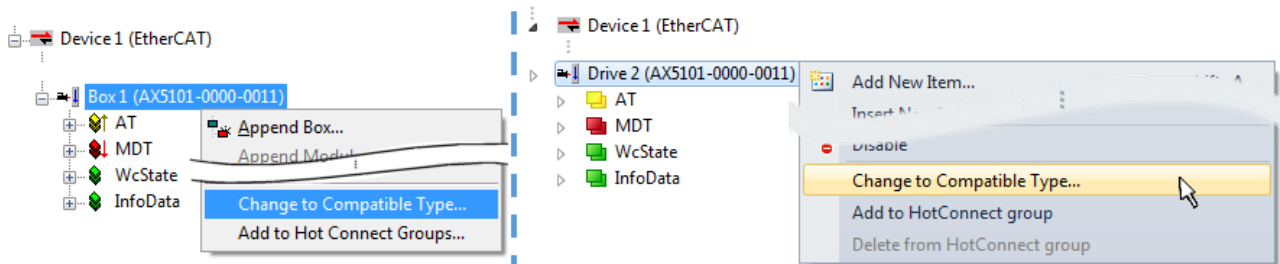


Fig. 112: Dialog “Change to Compatible Type...” (left: TwinCAT 2; right: TwinCAT 3)

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



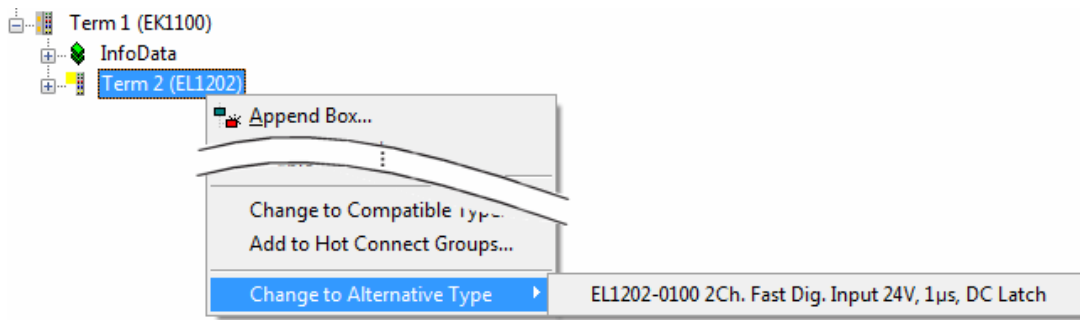


Fig. 113: TwinCAT 2 Dialog Change to Alternative Type

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).

### 5.2.7 EtherCAT subscriber configuration

In the left-hand window of the TwinCAT 2 System Manager or the Solution Explorer of the TwinCAT 3 Development Environment respectively, click on the element of the terminal within the tree you wish to configure (in the example: EL3751 Terminal 3).

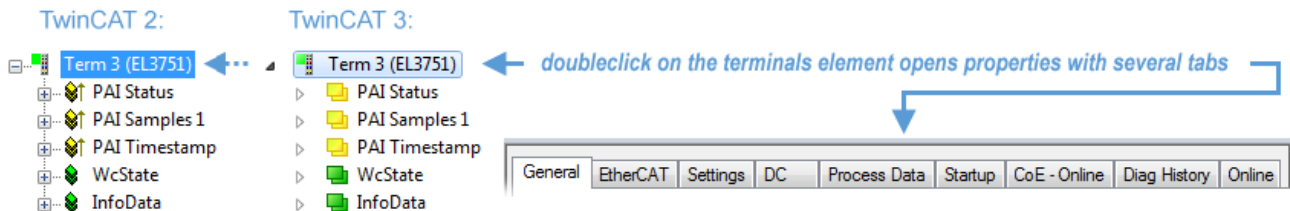


Fig. 114: Branch element as terminal EL3751

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

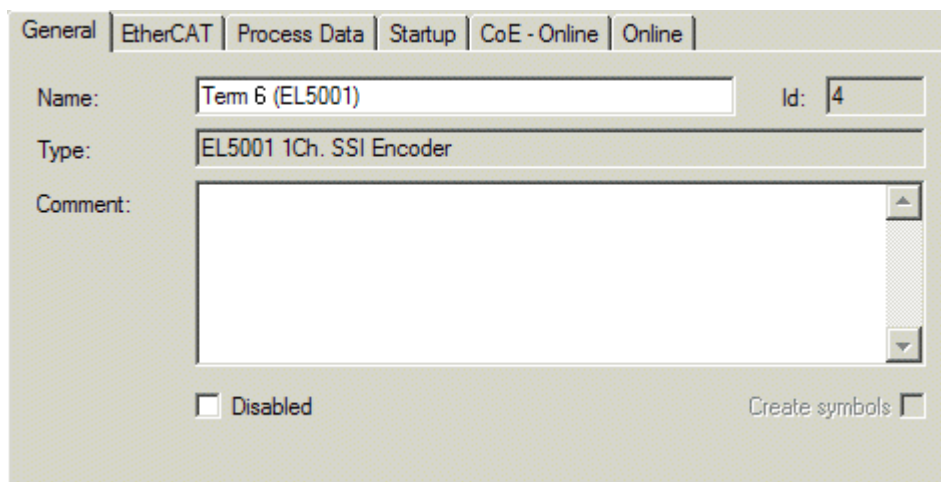


Fig. 115: “General” tab

|                       |   |
|-----------------------|---|
| <b>Name</b>           | Name of the EtherCAT device   |
| <b>Id</b>             | Number of the EtherCAT device   |
| <b>Type</b>           | EtherCAT device type  |
| <b>Comment</b>        | Here you can add a comment (e.g. regarding the system).                                   |
| <b>Disabled</b>       | Here you can deactivate the EtherCAT device.  |
| <b>Create symbols</b> | Access to this EtherCAT slave via ADS is only available if this control box is activated. |

#### “EtherCAT” tab

Fig. 116: “EtherCAT” tab

|                          |   |
|--------------------------|---|
| <b>Type</b>              | EtherCAT device type  |
| <b>Product/Revision</b>  | Product and revision number of the EtherCAT device  |
| <b>Auto Inc Addr.</b>    | 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 <sub>hex</sub> . For each further slave the address is decremented by 1 (FFFF <sub>hex</sub> , FFFE <sub>hex</sub> etc.). |
| <b>EtherCAT Addr.</b>    | 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.   |
| <b>Previous Port</b>     | 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.   |
| <b>Advanced Settings</b> | This button opens the dialogs for advanced settings.  |

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

#### “Process Data” tab

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

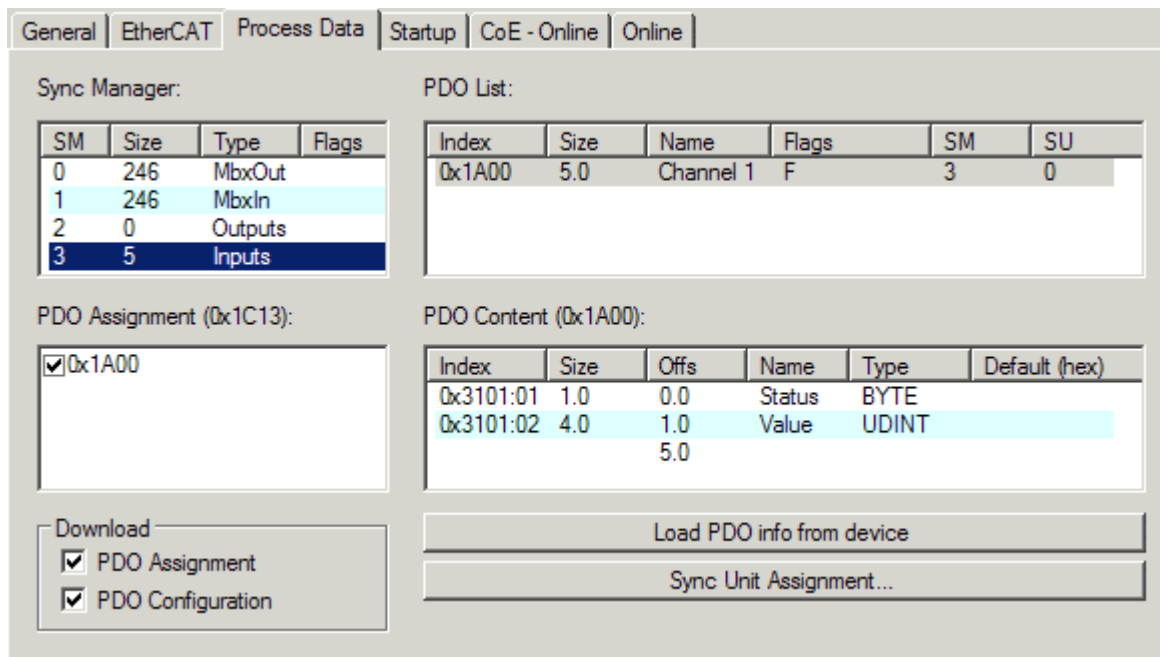


Fig. 117: "Process Data" tab

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").

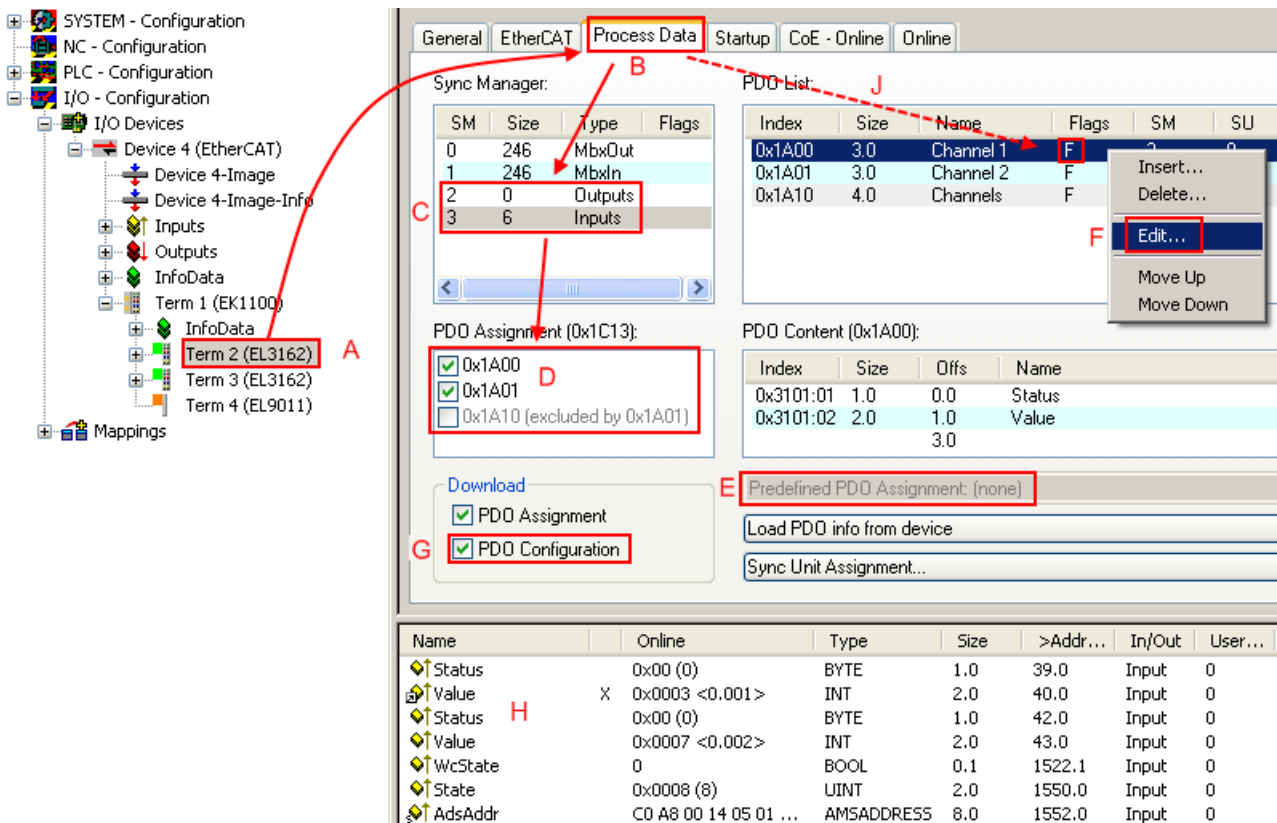


Fig. 118: Configuring the process data

**i Manual modification of the process data**

According to the ESI description, a PDO can be identified as “fixed” with the flag “F” in the PDO overview (Fig. *Configuring the process data*, J). The configuration of such PDOs cannot be changed, even if TwinCAT offers the associated dialog (“Edit”). In particular, CoE content cannot be displayed as cyclic process data. This generally also applies in cases where a device supports download of the PDO configuration, “G”. In case of incorrect configuration the EtherCAT slave usually refuses to start and change to OP state. The System Manager displays an “invalid SM cfg” 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 [▶ 105] 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.

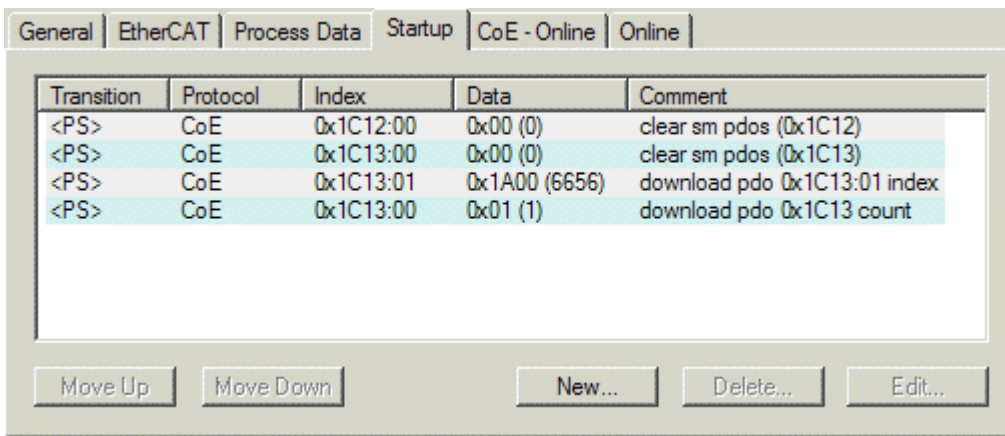


Fig. 119: "Startup" tab

| Column     | Description  |
|------------|--|
| Transition | Transition to which the request is sent. This can either be <ul style="list-style-type: none"> <li>the transition from pre-operational to safe-operational (PS), or</li> <li>the transition from safe-operational to operational (SO).</li> </ul> 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.

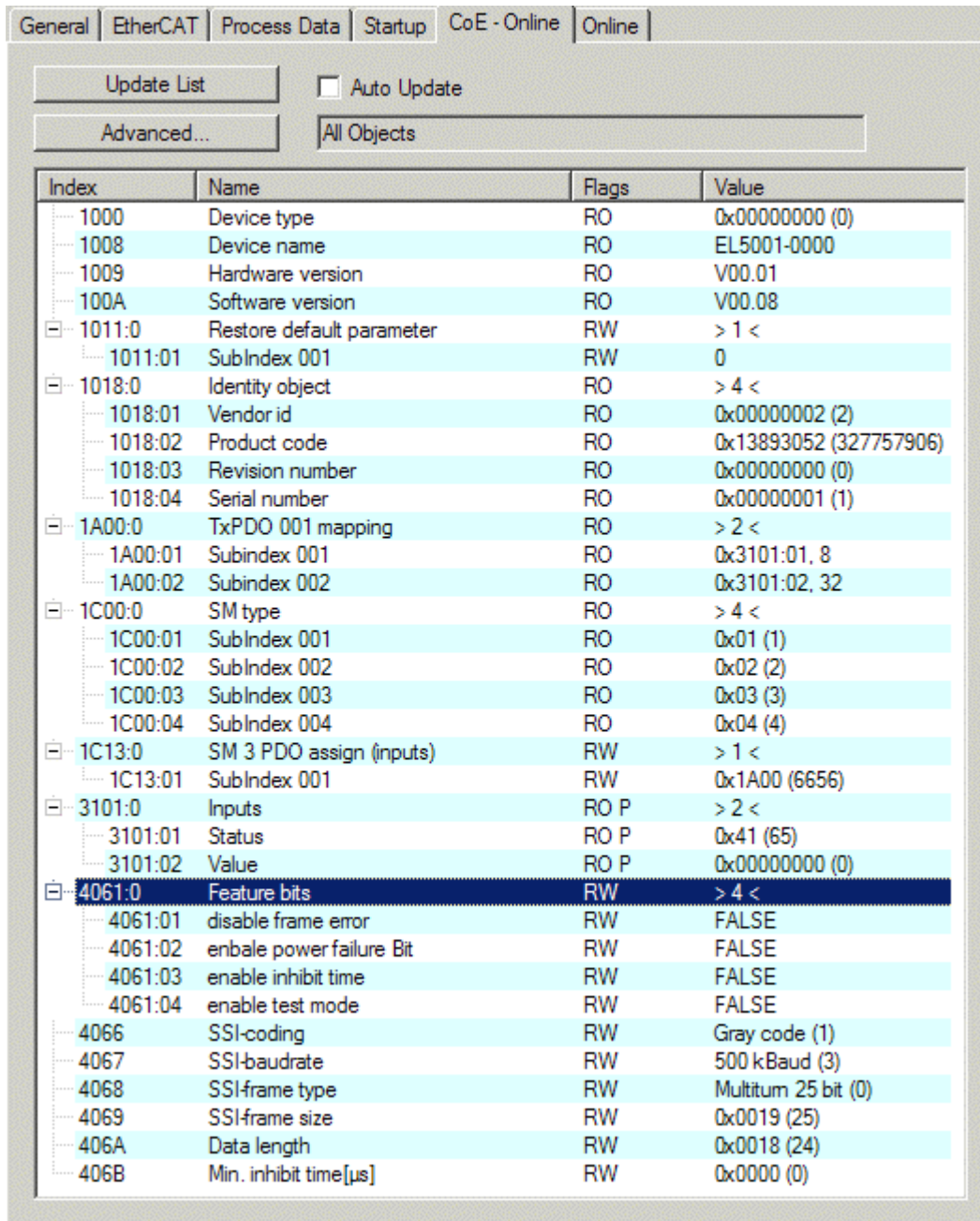


Fig. 120: "CoE - Online" tab

**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.

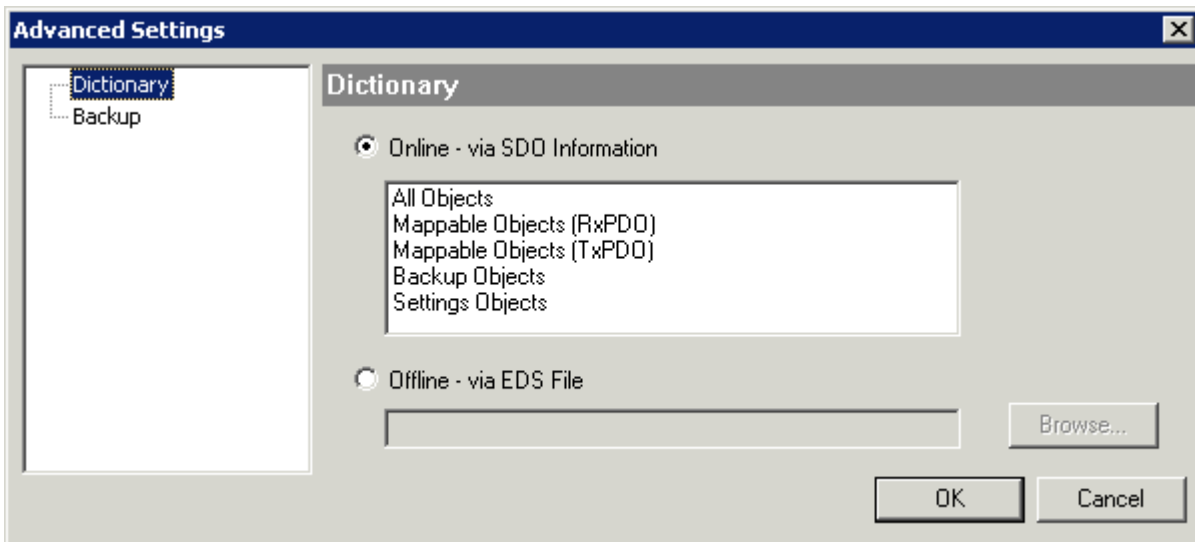


Fig. 121: Dialog “Advanced settings”

**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

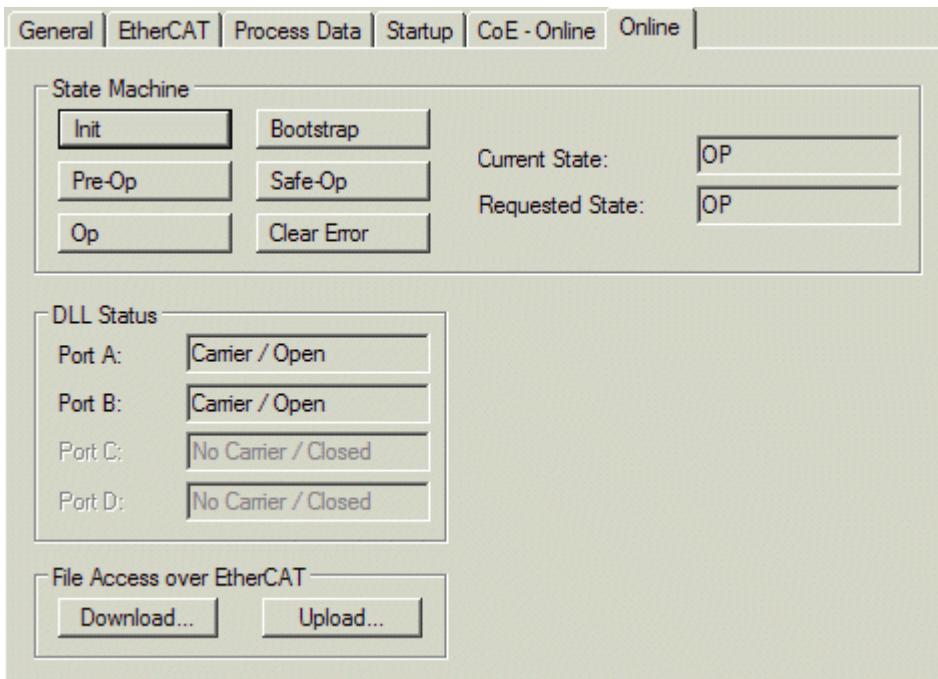


Fig. 122: “Online” tab

## State Machine

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

|                 |  |
|-----------------|--|
| <b>Download</b> | With this button a file can be written to the EtherCAT device. |
| <b>Upload</b>   | With this button a file can be read from the EtherCAT device.  |

## “DC” tab (Distributed Clocks)

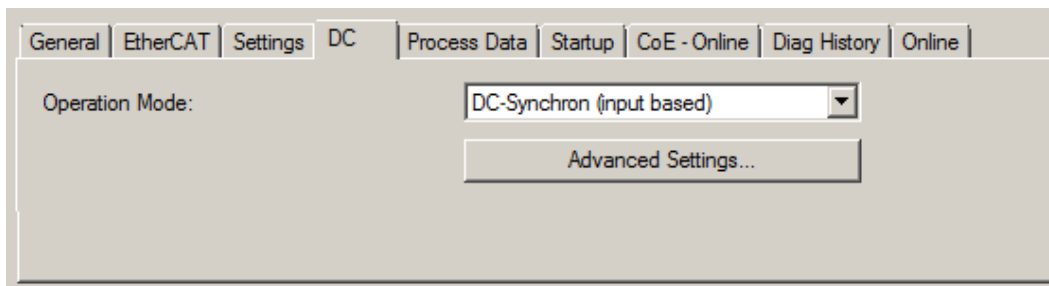


Fig. 123: “DC” tab (Distributed Clocks)

|                             |  |
|-----------------------------|--|
| <b>Operation Mode</b>       | Options (optional): <ul style="list-style-type: none"> <li>• FreeRun</li> <li>• SM-Synchron</li> <li>• DC-Synchron (Input based)</li> <li>• DC-Synchron</li> </ul> |
| <b>Advanced Settings...</b> | 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



### 5.2.7.1 Detailed description of Process Data tab

#### Sync Manager

Lists the configuration of the Sync Manager (SM).

If the EtherCAT device has a mailbox, SM0 is used for the mailbox output (MbxOut) and SM1 for the mailbox input (MbxIn).

SM2 is used for the output process data (outputs) and SM3 (inputs) for the input process data.

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

#### PDO Assignment



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

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

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

#### **i** Activation of PDO assignment

- ✓ If you have changed the PDO assignment, in order to activate the new PDO assignment,
  - a) the EtherCAT slave has to run through the PS status transition cycle (from pre-operational to safe-operational) once (see [Online tab \[▶ 103\]](#)),
  - 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.<br>If this PDO is assigned to a Sync Manager, it appears as a variable of the slave with this parameter as the name. |   |
| Flags  | F   | Fixed content: The content of this PDO is fixed and cannot be changed by the System Manager.  |
|        | M   | Mandatory PDO. This PDO is mandatory and must therefore be assigned to a Sync Manager! Consequently, this PDO cannot be deleted from the <i>PDO Assignment</i> list |
| SM     | Sync Manager to which this PDO is assigned. If this entry is empty, this PDO does not take part in the process data traffic.          |   |
| SU     | Sync unit to which this PDO is assigned.  |   |

#### PDO Content

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

## Download

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

## PDO Assignment

If this check box is selected, the PDO assignment that is configured in the PDO Assignment list is downloaded to the device on startup. The required commands to be sent to the device can be viewed in the [Startup \[► 100\]](#) 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.

## 5.2.8 Import/Export of EtherCAT devices with SCI and XTI

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

#### 5.2.8.1 Basic principles

An EtherCAT slave is basically parameterized through the following elements:

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

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

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

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

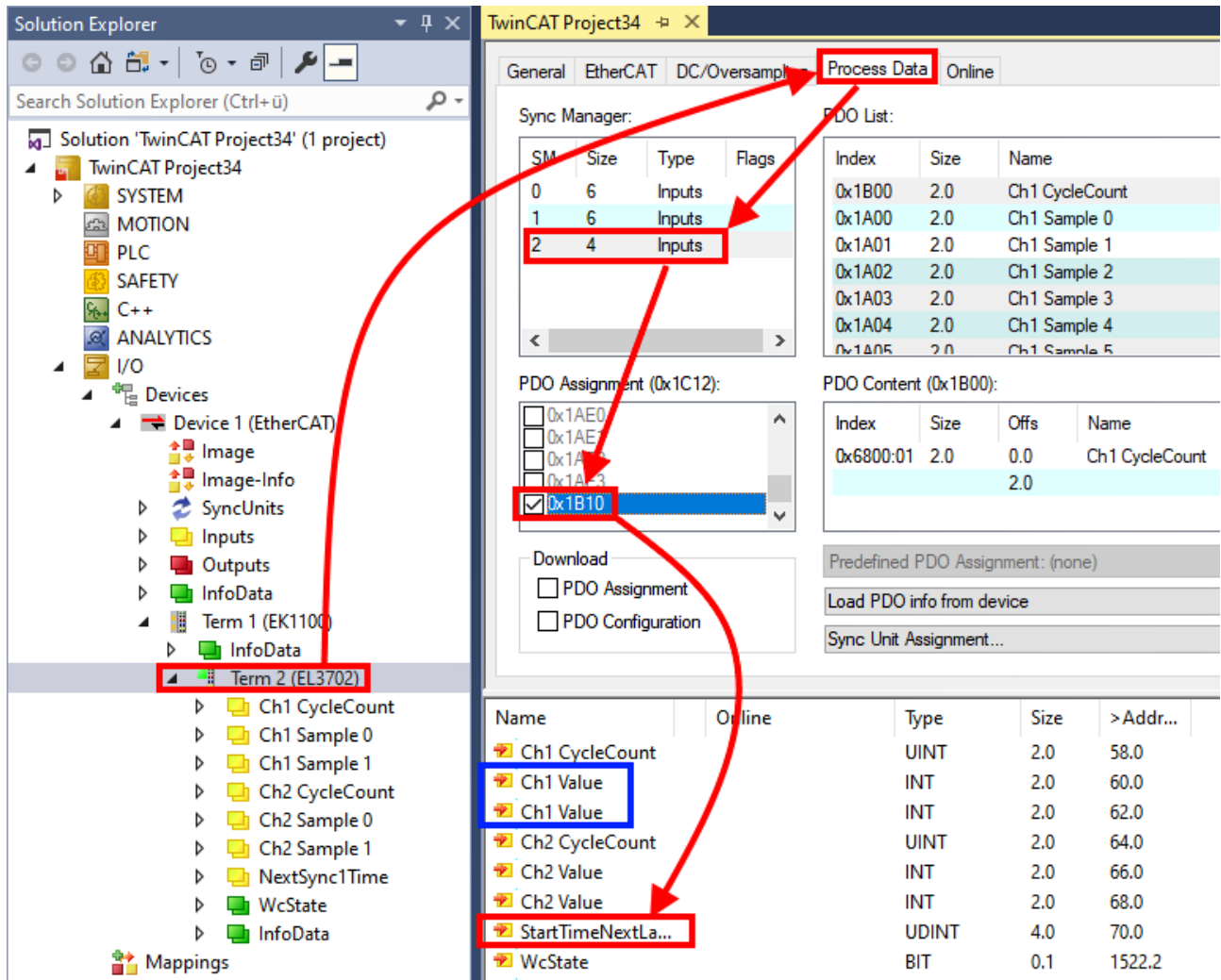
It may become necessary to reuse the modified slave in other projects in this way, without having to make equivalent configuration changes to the slave again. To accomplish this, proceed as follows:

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

TwinCAT offers two methods for this purpose:

- within the TwinCAT environment: Export/Import as **x**ti file or
- outside, i.e. beyond the TwinCAT limits: Export/Import as **s**ci file.

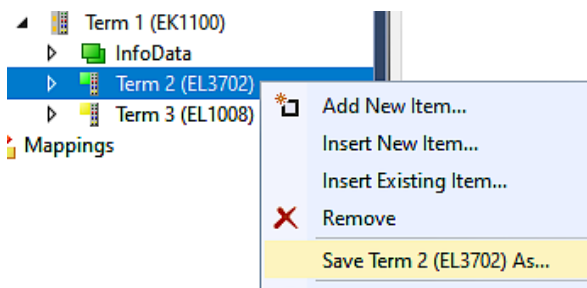
An example is provided below for illustration purposes: an EL3702 terminal with standard setting is switched to 2-fold oversampling (blue) and the optional PDO "StartTimeNextLatch" is added (red):



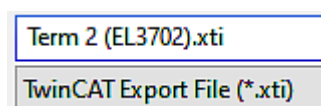
The two methods for exporting and importing the modified terminal referred to above are demonstrated below.

### 5.2.8.2 Procedure within TwinCAT with xti files

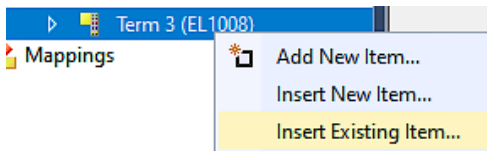
Each IO device can be exported/saved individually:



The xti file can be stored:



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



### 5.2.8.3 Procedure within and outside TwinCAT with sci file

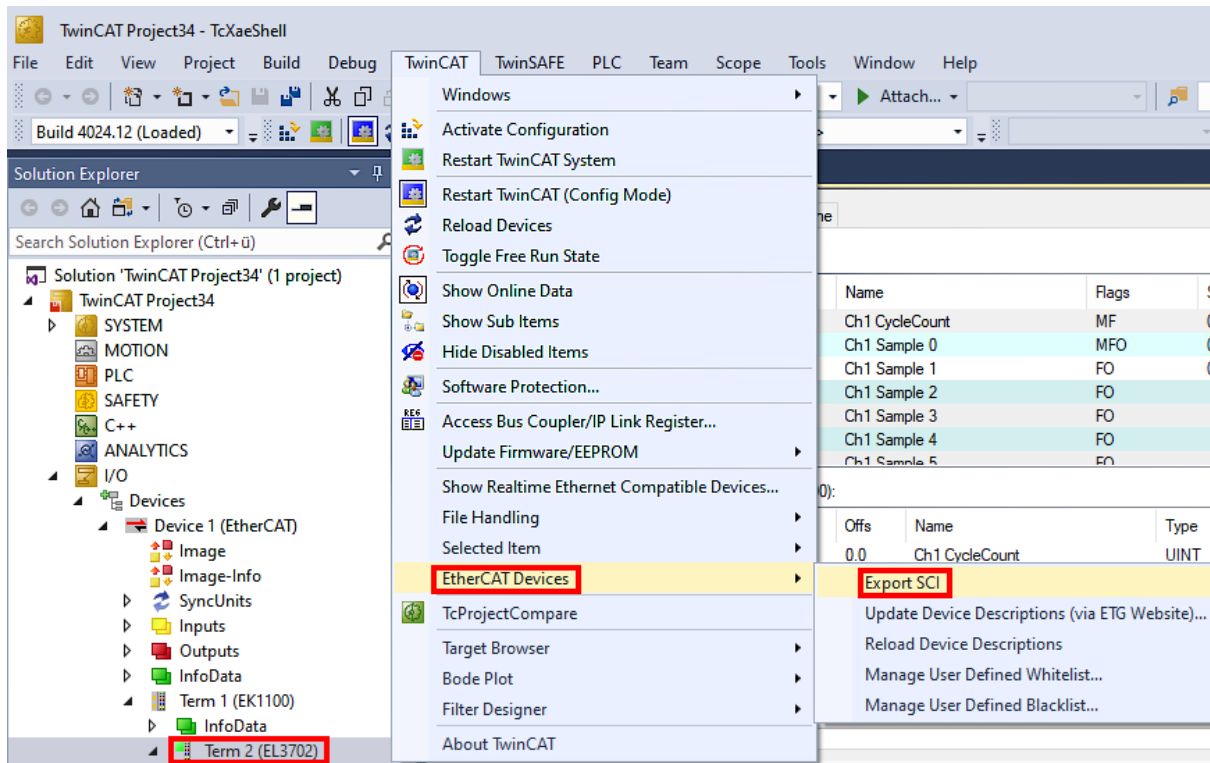
*Note regarding availability (2021/01)*

The SCI method is available from TwinCAT 3.1 build 4024.14.

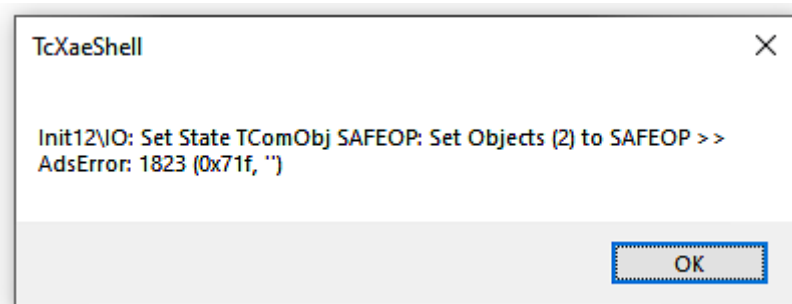
The Slave Configuration Information (SCI) describes a specific complete configuration for an EtherCAT slave (terminal, box, drive...) based on the setting options of the device description file (ESI, EtherCAT Slave Information). That is, it includes PDO, CoE, synchronization.

**Export:**

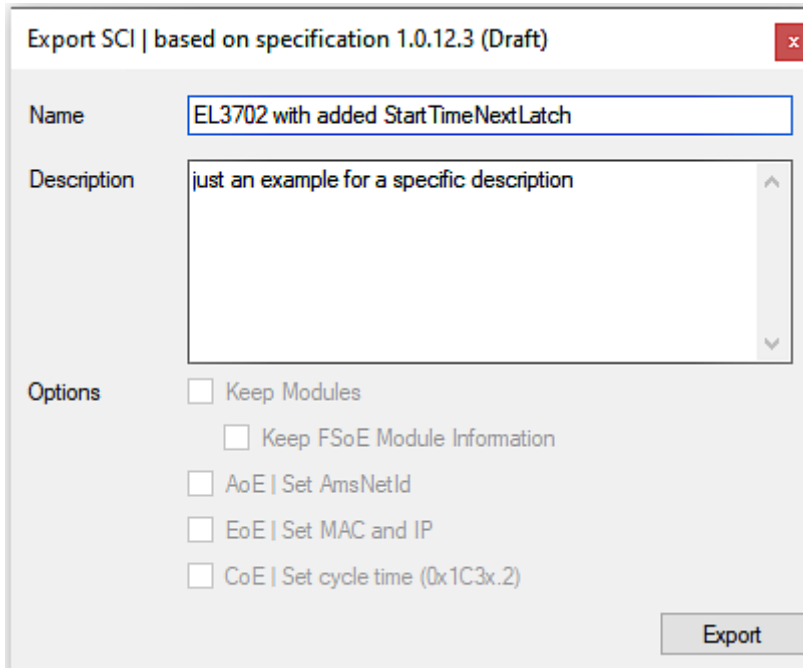
- select a single device via the menu (multiple selection is also possible):  
TwinCAT → EtherCAT Devices → Export SCI.



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



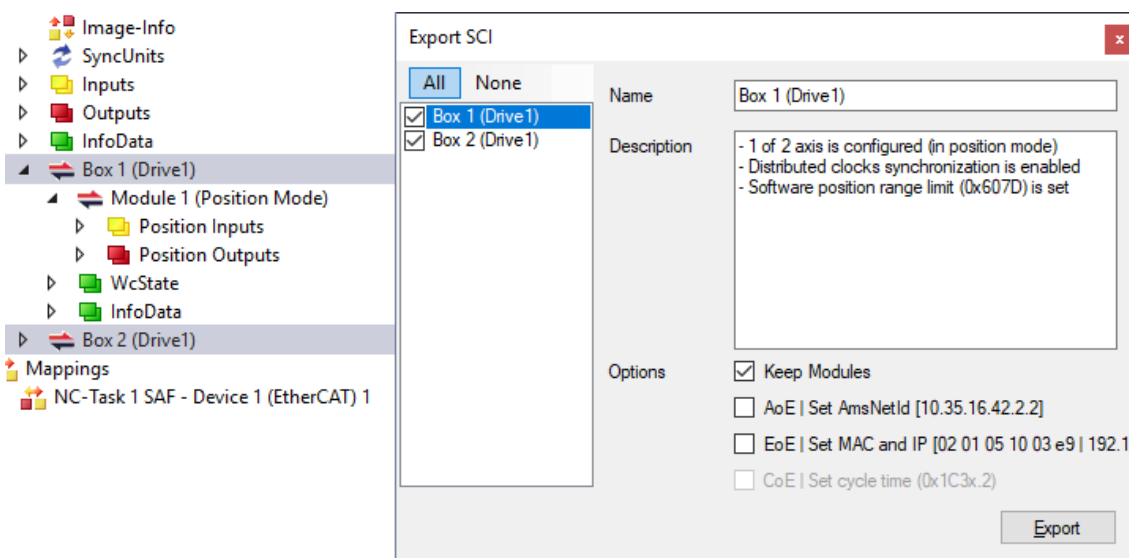
- A description may also be provided:



- Explanation of the dialog box:

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

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

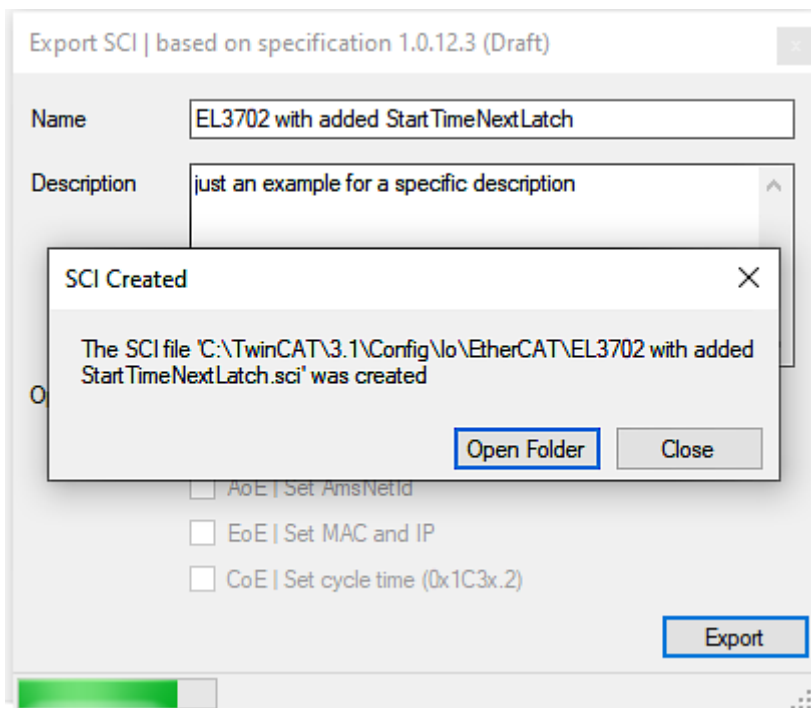


- Selection of the slaves to be exported:
  - All:
    - All slaves are selected for export.

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

Dateiname:   
 Dateityp:

- The export takes place:

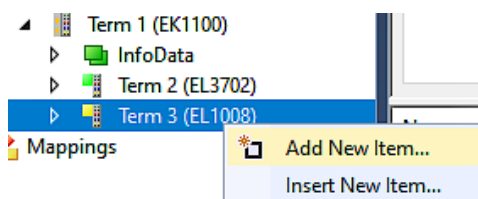


**Import**

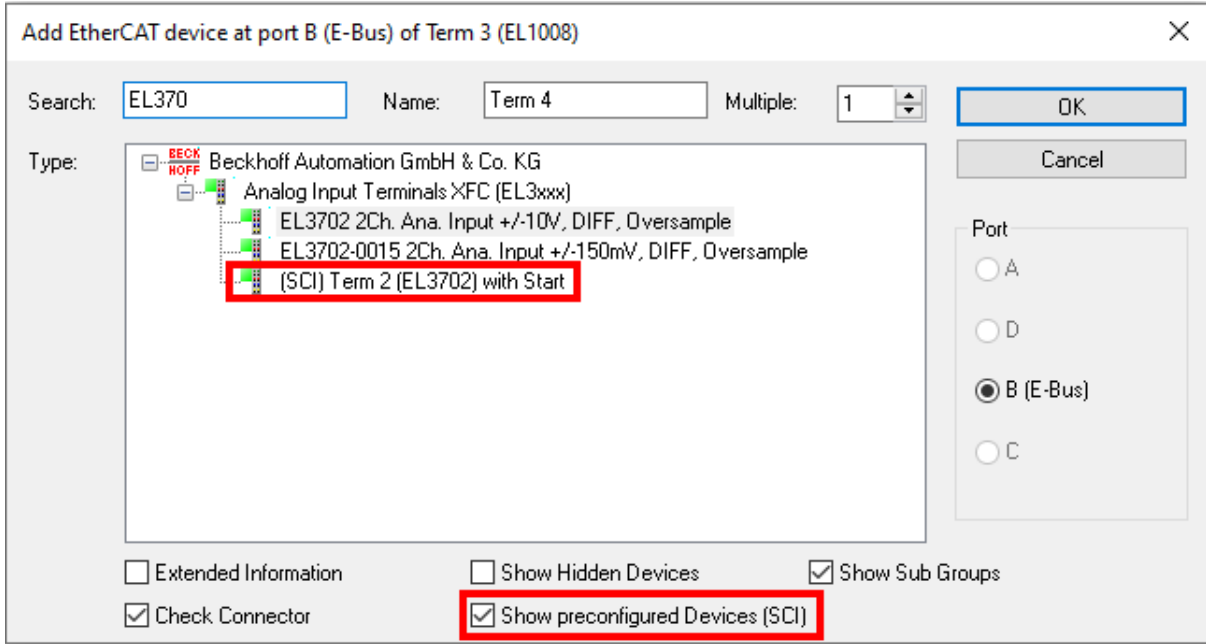
- An sci description can be inserted manually into the TwinCAT configuration like any normal Beckhoff device description.
- The sci file must be located in the TwinCAT ESI path, usually under:  
C:\TwinCAT\3.1\Config\Io\EtherCAT

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

- Open the selection dialog:

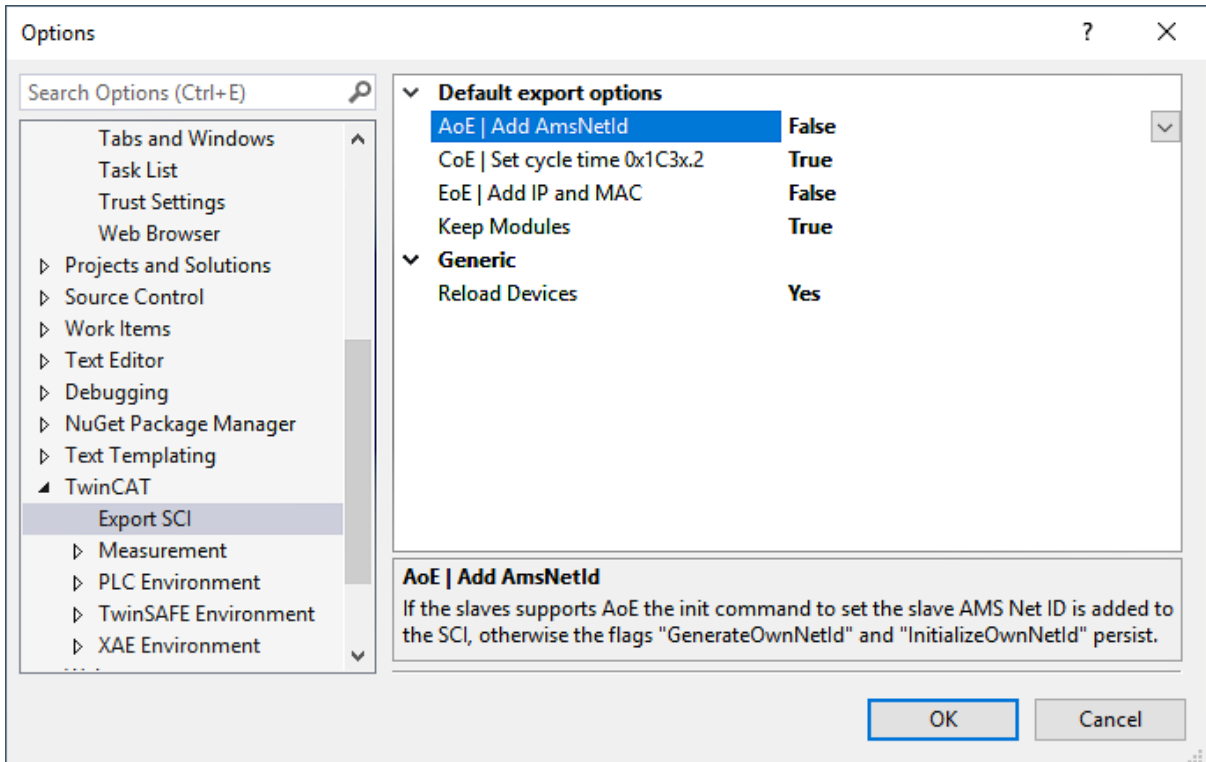


- Display SCI devices and select and insert the desired device:



**Additional Notes**

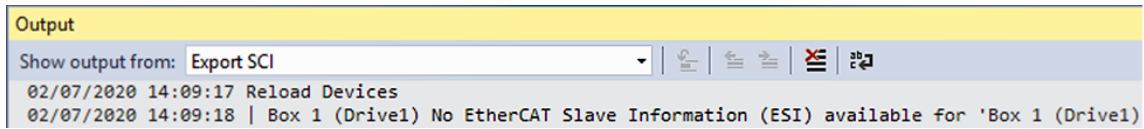
- Settings for the SCI function can be made via the general Options dialog (Tools → Options → TwinCAT → Export SCI):



Explanation of the settings:

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

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





### 5.3 General Notes - EtherCAT Slave Application

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

#### Diagnosis in real time: WorkingCounter, EtherCAT State and Status

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

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

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

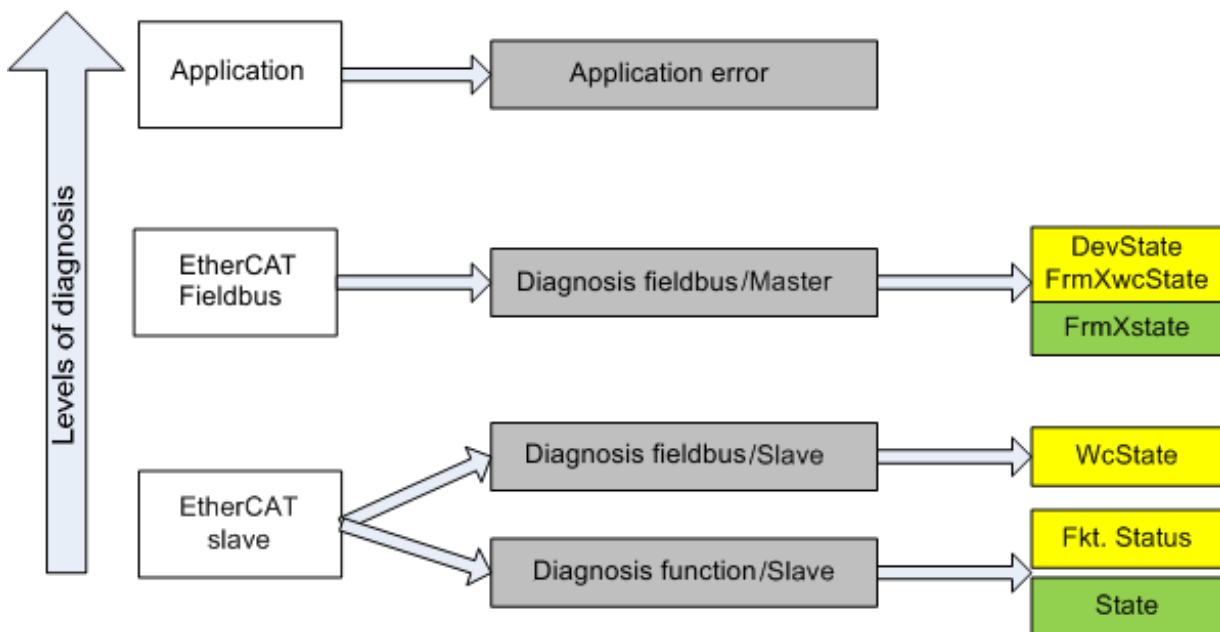


Fig. 124: Selection of the diagnostic information of an EtherCAT Slave

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.

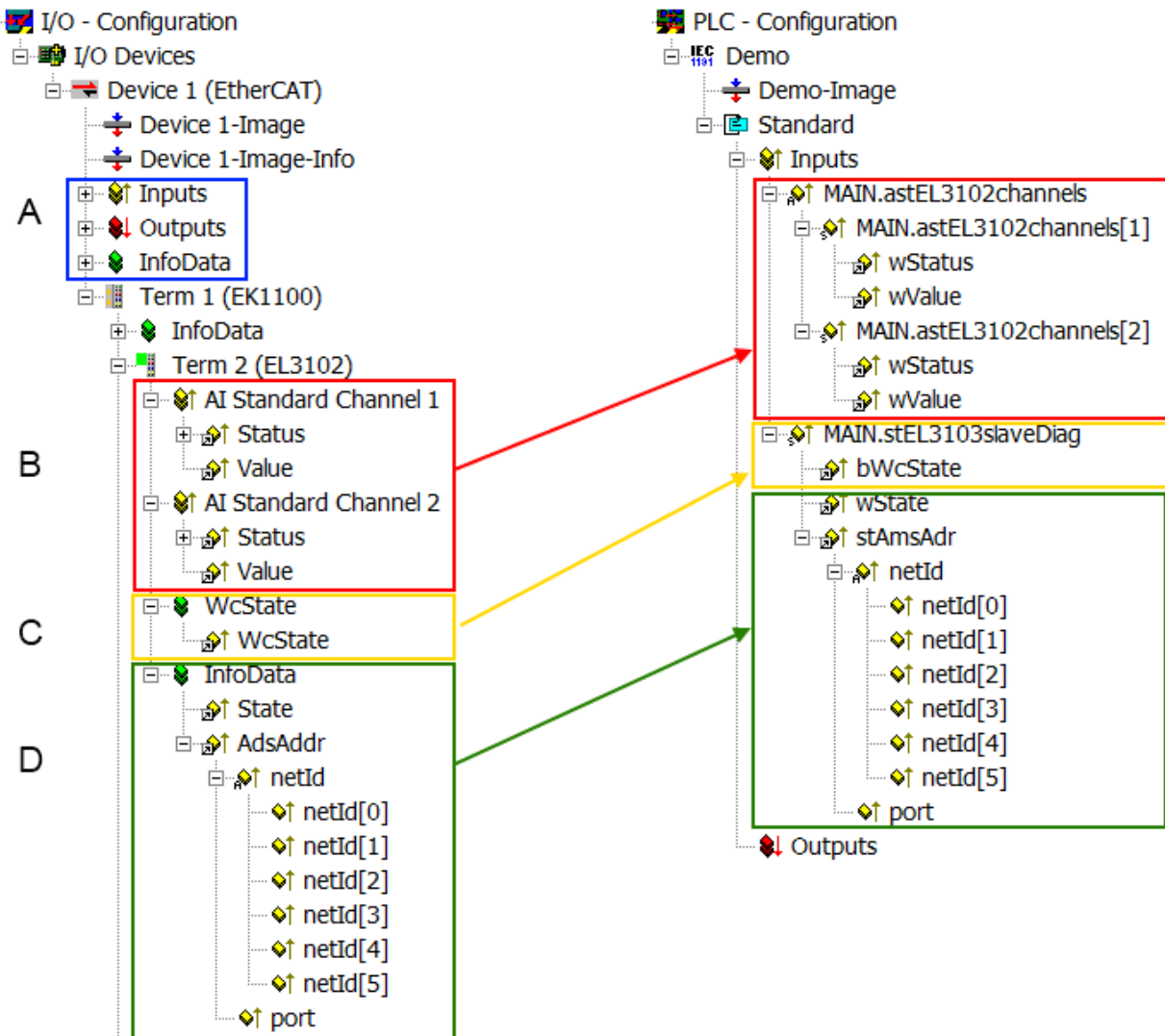


Fig. 125: Basic EtherCAT Slave Diagnosis in the PLC

The following aspects are covered here:

| Code | Function  | Implementation  | Application/evaluation  |
|------|---|---|---|
| A    | The EtherCAT Master's diagnostic information updated acyclically (yellow) or provided acyclically (green).  |   | At least the DevState is to be evaluated for the most recent cycle in the PLC.<br>The EtherCAT Master's diagnostic information offers many more possibilities than are treated in the EtherCAT System Documentation. A few keywords: <ul style="list-style-type: none"> <li>• CoE in the Master for communication with/through the Slaves</li> <li>• Functions from <i>TcEtherCAT.lib</i></li> <li>• Perform an OnlineScan</li> </ul> |
| B    | In the example chosen (EL3102) the EL3102 comprises two analogue input channels that transmit a single function status for the most recent cycle.   | Status <ul style="list-style-type: none"> <li>• the bit significations may be found in the device documentation</li> <li>• other devices may supply more information, or none that is typical of a slave</li> </ul>   | 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 <ol style="list-style-type: none"> <li>1. at the EtherCAT Slave, and, with identical contents</li> <li>2. as a collective variable at the EtherCAT Master (see Point A)</li> </ol> for linking. | WcState (Working Counter)<br>0: valid real-time communication in the last cycle<br>1: invalid real-time communication<br>This may possibly have effects on the process data of other Slaves that are located in the same SyncUnit   | In order for the higher-level PLC task (or corresponding control applications) to be able to rely on correct data, the communication status of the EtherCAT Slave must be evaluated there. Such information is therefore provided with the process data for the most recent cycle.  |
| D    | Diagnostic information of the EtherCAT Master which, while it is represented at the slave for linking, is actually determined by the Master for the Slave concerned and represented there. This information cannot be characterized as real-time, because it <ul style="list-style-type: none"> <li>• is only rarely/never changed, except when the system starts up</li> <li>• is itself determined acyclically (e.g. EtherCAT Status)</li> </ul>  | State<br>current Status (INIT..OP) of the Slave. The Slave must be in OP (=8) when operating normally.<br><i>AdsAddr</i><br>The ADS address is useful for communicating from the PLC/task via ADS with the EtherCAT Slave, e.g. for reading/writing to the CoE. The AMS-NetID of a slave corresponds to the AMS-NetID of the EtherCAT Master; communication with the individual Slave is possible via the <i>port</i> (= EtherCAT address). | Information variables for the EtherCAT Master that are updated acyclically. This means that it is possible that in any particular cycle they do not represent the latest possible status. It is therefore possible to read such variables through ADS.  |

**NOTE**

**Diagnostic information**

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

**CoE Parameter Directory**

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

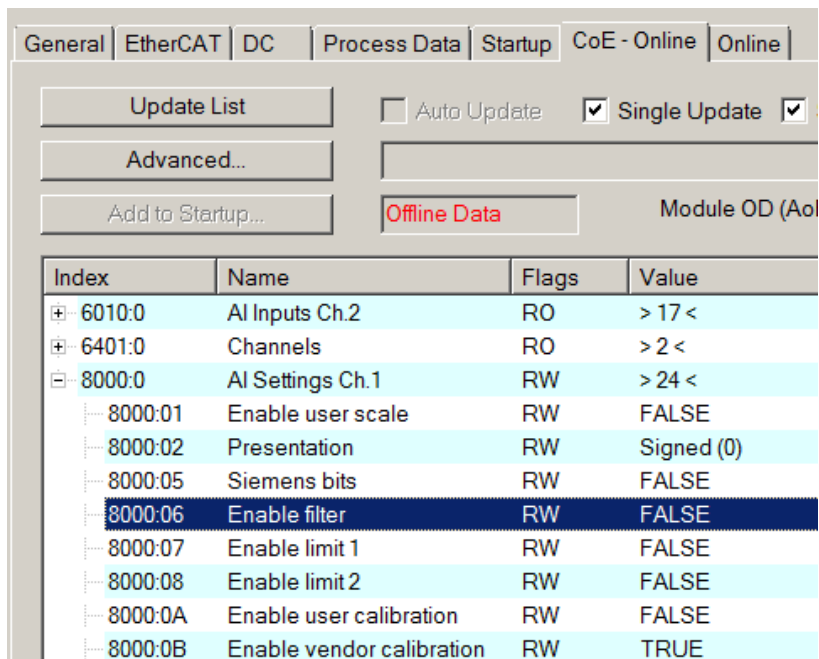


Fig. 126: EL3102, CoE directory

### **i** EtherCAT System Documentation

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

A few brief extracts:

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

### **Commissioning aid in the TwinCAT System Manager**

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

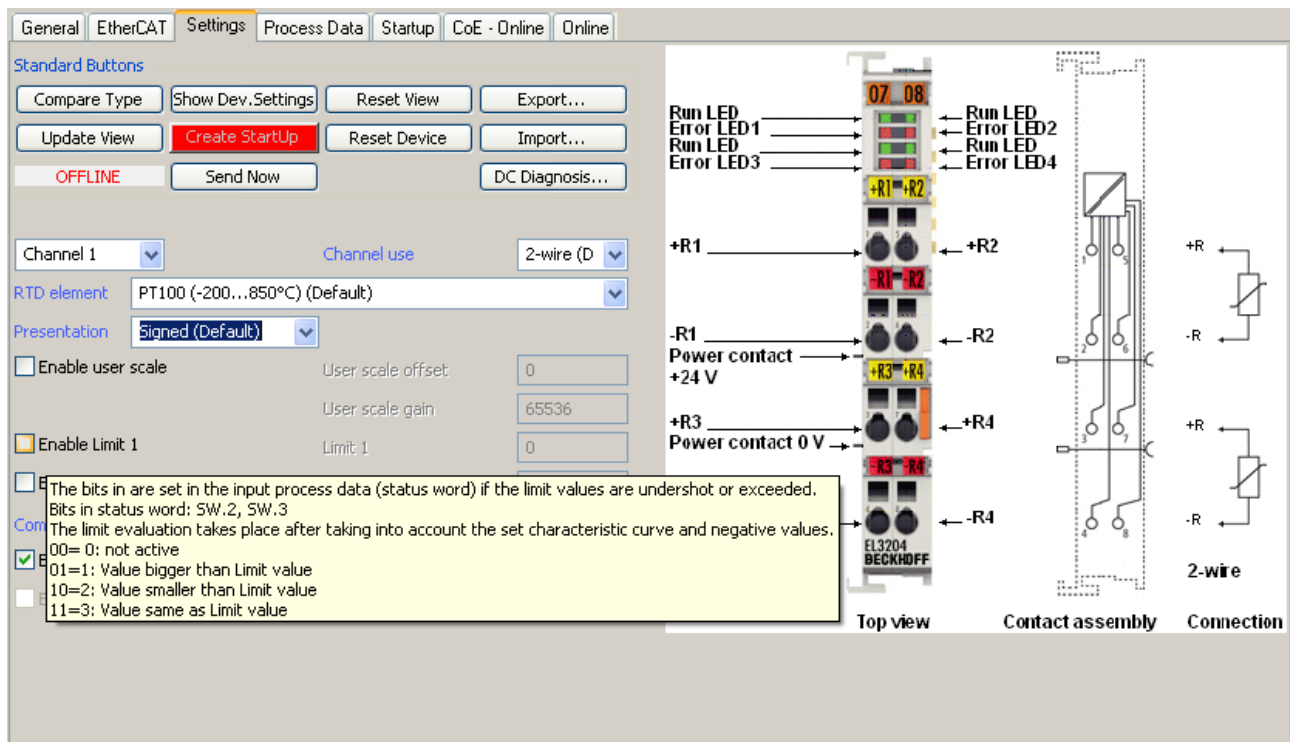


Fig. 127: Example of commissioning aid for a EL3204

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 \[▶ 24\]](#)" 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.

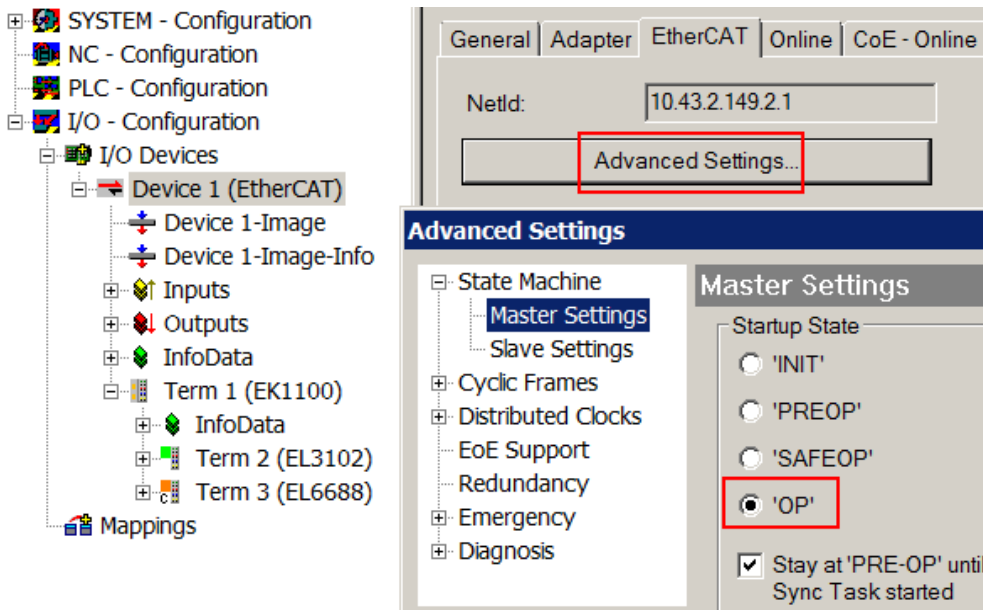


Fig. 128: Default behaviour of the System Manager

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

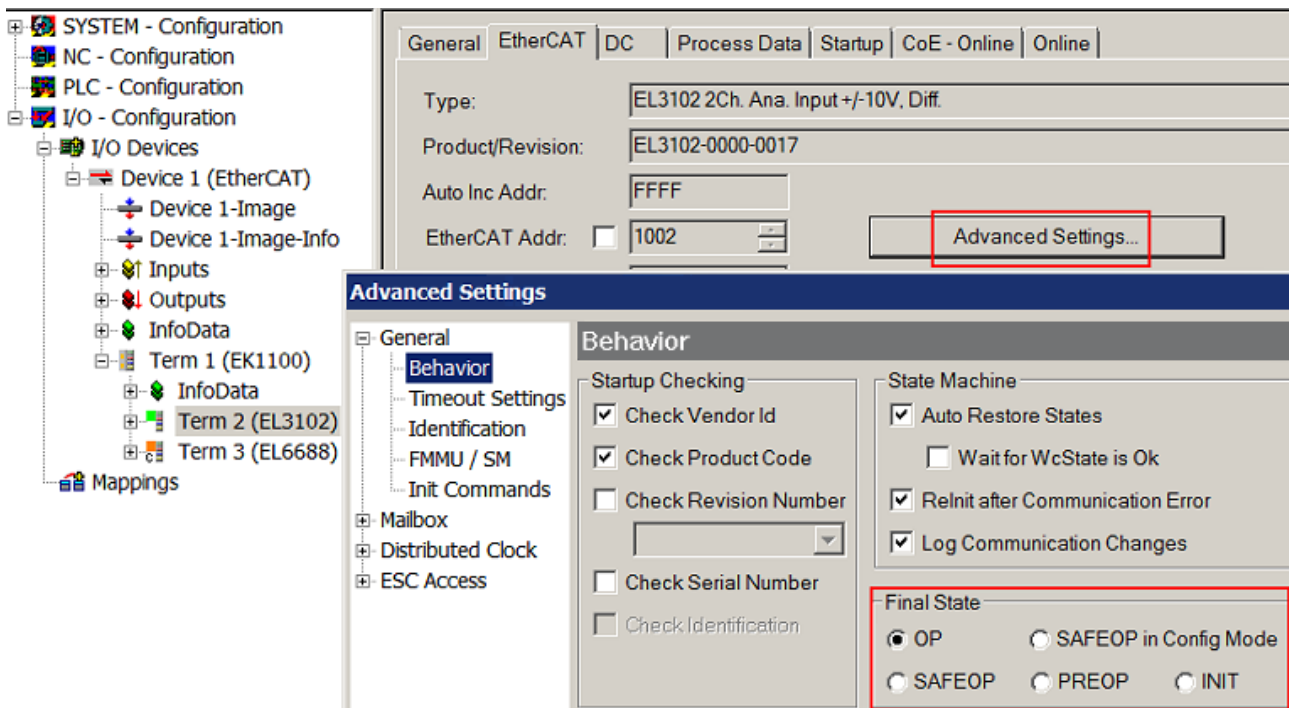


Fig. 129: Default target state in the Slave

**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.

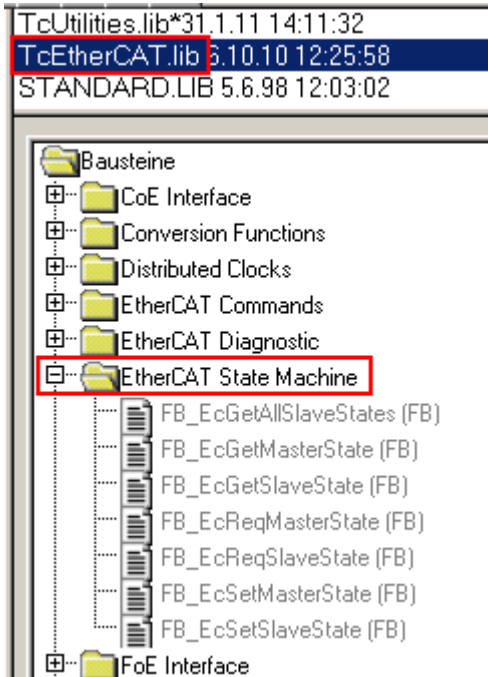


Fig. 130: PLC function blocks

### Note regarding E-Bus current

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

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

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

Fig. 131: 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:

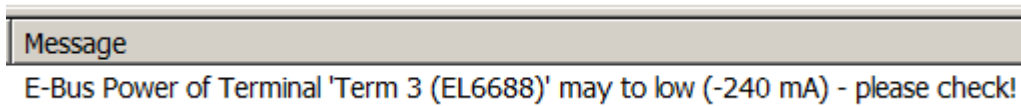


Fig. 132: Warning message for exceeding E-Bus current

| <b>NOTE</b>  |
|--|
| <p><b>Caution! Malfunction possible!</b></p> <p>The same ground potential must be used for the E-Bus supply of all EtherCAT terminals in a terminal block!</p> |



## 5.4 Basic principles of function and commissioning

The EL6692 is used for data exchange between two EtherCAT circuits. Predefined process data can be copied from one fieldbus side to the other. Status variables provide information about missing data and the status of the other side.

If distributed clocks are used in both EtherCAT circuits the EL6692 can also be used for synchronizing the two timebases. In this case the user has to define which side has the reference clock with the higher priority. The EL6692 will then send a reference clock correction value to the EtherCAT master with the lower priority.

Further information about operating principle and the application of distributed clocks can be found in the associated documentation at [www.beckhoff.com](http://www.beckhoff.com).

The EL6692 consists of two EtherCAT slaves in *one* case: the EL6692 on the primary side (terminal bus) and the EL6692-0002 on the secondary side with network cable connection. Both slaves have their own power supplies and are thus independently operational. The primary side is supplied with 5 V via the E-bus, the secondary side with 24 V via the external connection.

Data transfer in the EL6692 is handled by a microcontroller that exchanges data in both directions in free-running mode. The transfer time from one side to the other depends on the number of bytes. The configurable "DC synchronous" mode does not apply to the process data exchange, but to the application of the EL6692 for distributed clock synchronization. The EL6692 does not deal with throughput control, i.e. the user has to ensure (through suitable programming) that the EL6692 is not supplied with more data than it can process, e.g. by using a higher-level handshake. The [limit values \[► 128\]](#) specified below must be complied with.

The EL6692 supports EoE, Ethernet over EtherCAT. This means that not only cyclical process data can be transported from the EtherCAT side to the other side, but also standard Ethernet frames arriving at the EL6692 via mailbox communication. Since TwinCAT and the connected EtherCAT environment act as a virtual network card, Windows handles the routing of IP frames (e.g. 192.168.2.1) to the EL6692, in order to transport the frames to the EtherCAT system on the other side. Further information can be found in the EL6601/EL6614 documentation. No special EL6692 configuration is required.

- [Application notes for EL6692 \[► 121\]](#)
- [Application notes for EL6692-0002 \[► 123\]](#)
- [General application notes \[► 121\]](#)

---

### ● EL6692 support in Beckhoff TwinCAT

**I** Full support for distributed clocks synchronization with the EL6692 is provided from TwinCAT version 2.11 or higher. Previous versions are limited to process data exchange.

---

#### Application notes for EL6692, primary side

The default state of the EL6692 after reading/creation is

- DC support off - FreeRun
- No process data variables preconfigured
- No distributed clocks information data; see *Process Data* tab in the fig. *Default process image for EL6692*.

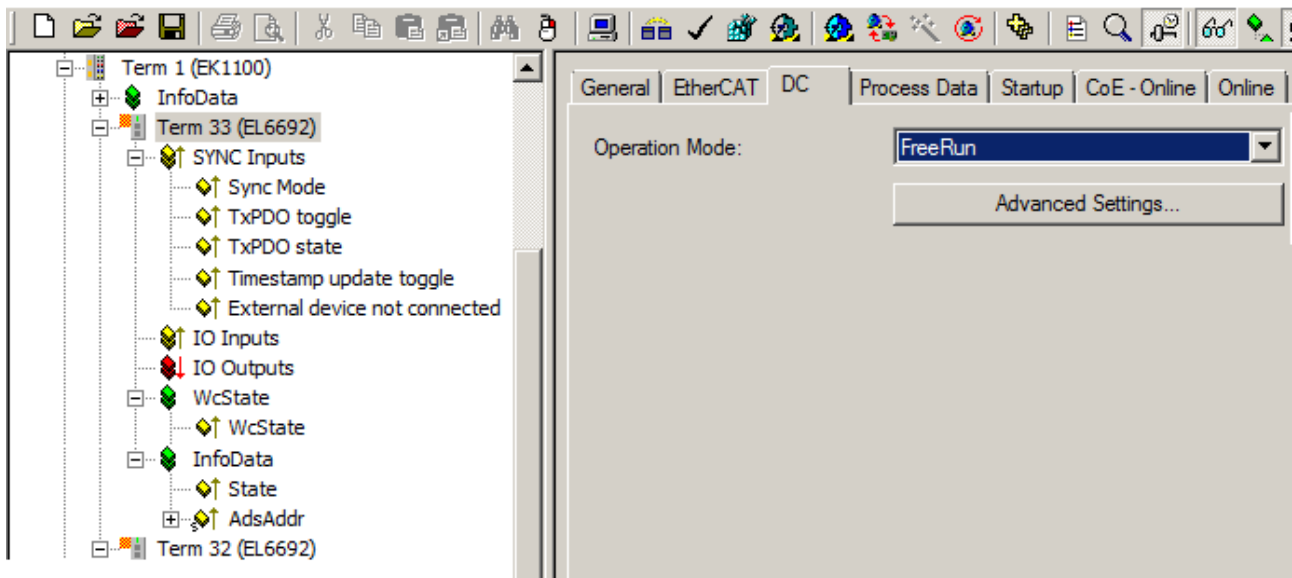


Fig. 133: Default process image for EL6692

The process data to be transferred must now be created on the primary side as required. A right-click on "IO Inputs" or "IO Outputs" opens the dialog box from the fig. *Appending process data variables*.

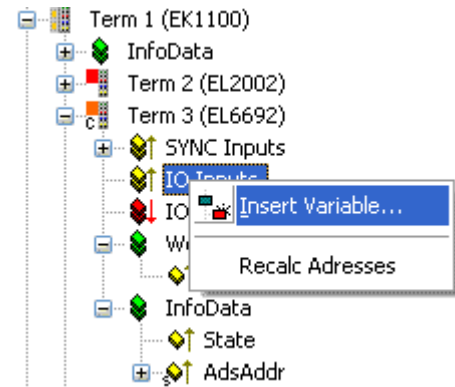


Fig. 134: Appending process data variables

In the full configuration with activated PDO 0x1A02, the EL6692 provides the diagnostic process data from the fig. *Full range of diagnostic process data*

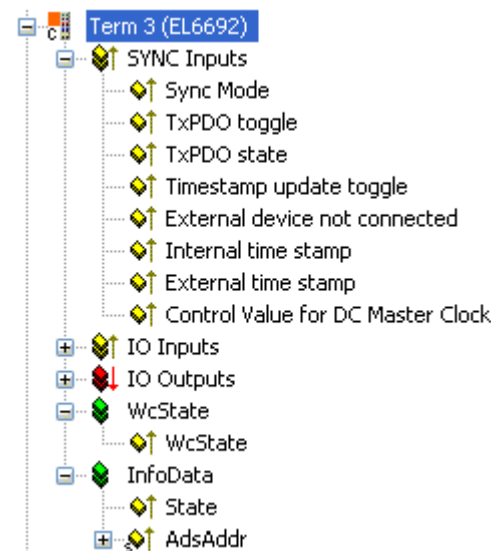


Fig. 135: Full range of diagnostic process data

- **SyncMode:** Information about the synchronization state. 0 = no synchronization, 1 = secondary side is Sync master, 2 = primary side is Sync master.
- **TxPDO toggle:** Bit toggles when a new set of process data has been delivered by the opposite side, but has not yet been collected by the receiving side. In this way the EL6692 also enables operation with EtherCAT cycle times that are greater than the transport time.
- **TxPDO state:** >0 if the other side is not in OP state.
- **Timestamp update toggle:** Bit toggles when new DC data were supplied.
- **External device not connected:** 1 = other side is not connected to its EtherCAT fieldbus.
- **Internal time stamp:** Distributed clocks time on the current side (primary or secondary)
- **External time stamp:** Distributed clocks time on the other side (primary or secondary).
- **Control Value for DC Master Clock:** Offset for correction of the lower priority reference clock.

**Application notes for EL6692-0002, secondary side**

The default state of the EL6692 after reading/creation is

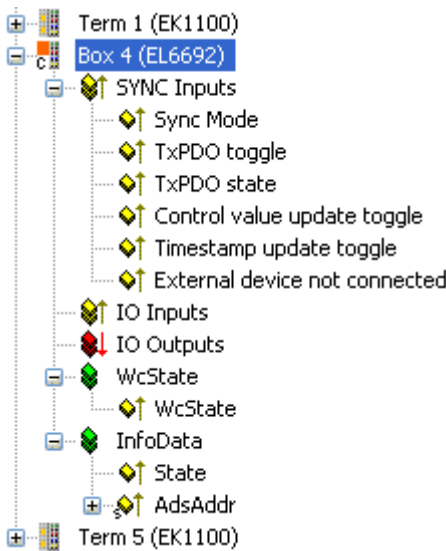


Fig. 136: Default process image for EL6692-0002

In full configuration the EL6692-0002 offers the following variables:

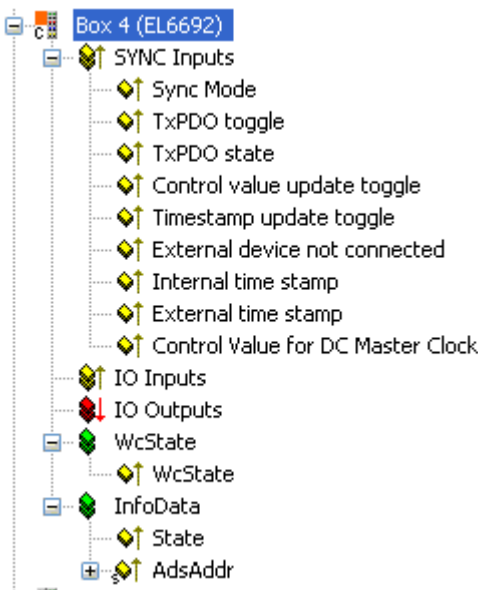


Fig. 137: Full range of diagnostic process data

- **SyncMode:** Information about the synchronization state. 0 = no synchronization, 1 = secondary side is Sync master, 2 = primary side is Sync master.
- **TxPDO toggle:** Bit toggles when a new set of process data has been delivered by the opposite side, but has not yet been collected by the receiving side. In this way the EL6692 also enables operation with EtherCAT cycle times that are greater than the transport time.
- **TxPDO state:** >0 if the other side is not in OP state.
- **Control Value update toggle:** Bit toggles when a new control value is available.
- **Timestamp update toggle:** Bit toggles when new DC data were supplied.
- **External device not connected:** 1 = other side is not connected to its EtherCAT fieldbus.
- **Internal time stamp:** Distributed clocks time on the current side (primary or secondary)
- **External time stamp:** Distributed clocks time on the other side (primary or secondary).
- **Control Value for DC Master Clock:** Offset for correction of the lower priority reference clock.

### Process data declaration on the secondary side

Exactly as on the primary side (see the fig. *Appending process data variables*), the process data can be created manually on the secondary side also. The order and bit size on the secondary side must match those defined on the primary side.

Alternatively, from TwinCAT 2.10 build 1329 the configuration for the primary side can be loaded into the EL6692 and read on the secondary side. Corresponding support is necessary in the TwinCAT system manager for this and both EtherCAT sides must be accessible online. This avoids the manual configuration of the secondary side; see the fig. *"Create configuration" on the primary side* and *"Get configuration" on the secondary side*.

### ● Process data declaration in older TwinCAT 2.10 versions

#### **i** Build 1325 to Build 1329:

The functionality of the additional "EL6692" tab in the TwinCAT System Manager can be ensured by downloading an EtherCAT System Manager extension. This can be downloaded if necessary from the download area of the [Beckhoff Website](http://www.beckhoff.com) at [www.beckhoff.com](http://www.beckhoff.com). Later TwinCAT versions already have this functionality integrated.

#### **before Build 1325:**

The process data declaration on the secondary side described below is not supported in this version.

The procedure is as follows:

- Define the desired process data on the primary side, see the fig. *Appending process data variables*.  
Notes:
  - max. 480 bytes in each data direction.
  - no structures, only standard data types are allowed.
- Execute "Create Configuration" on the primary side – the table is created; see the fig. *"Create configuration" on the primary side*.
- Carry out a reload on the primary side or restart the configuration in Config mode. In order to apply the defined process data to the secondary side, both sides have to pass from the INIT state to the PREOP state.

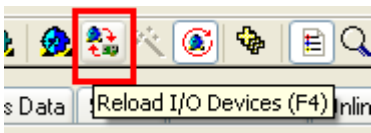


Fig. 138: Device Reload

The error message "Invalid SM In cfg" or "Invalid SM Out cfg" appears in the logger window. The secondary side can be read, if both EtherCAT sides of the EL6692 are at least in PRE-OP state.

- Execute "Get configuration" on the secondary side; see the fig. "Get configuration" on the secondary side. The process data configured on the primary side are generated accordingly in the configuration tree.

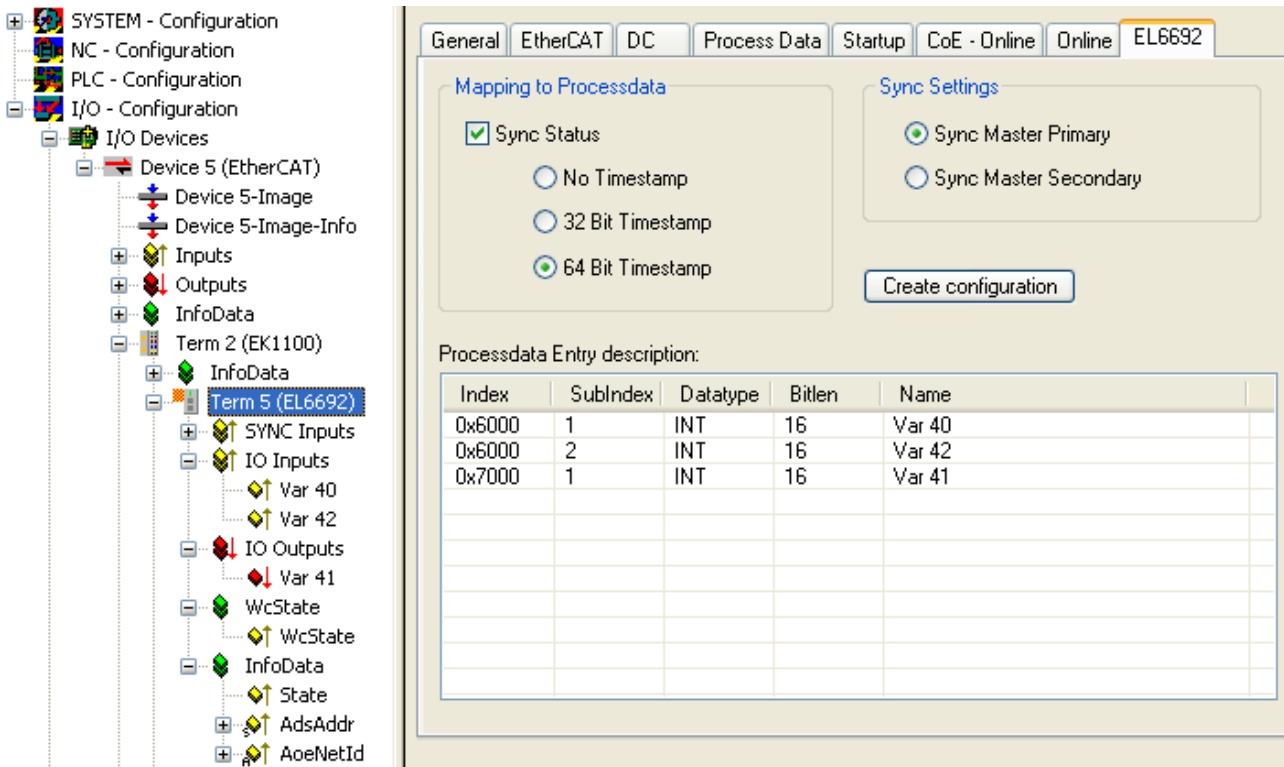


Fig. 139: "Create configuration" on the primary side

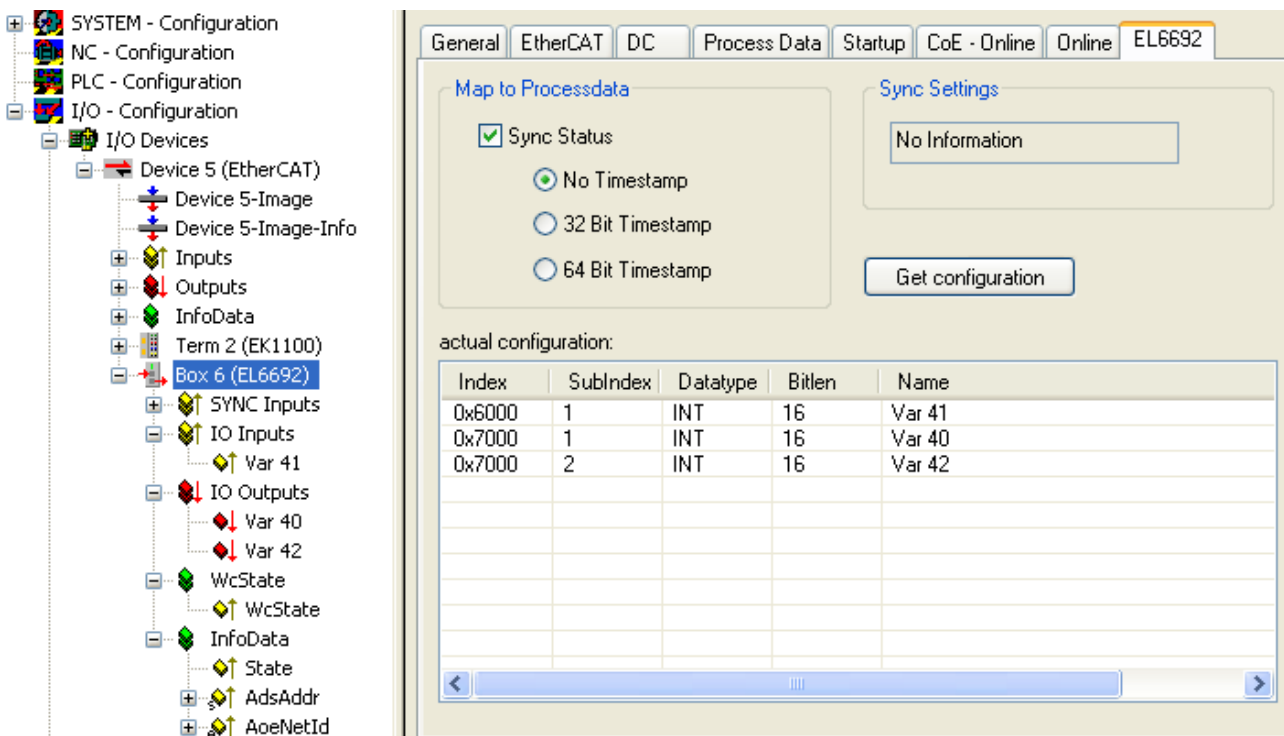


Fig. 140: "Get configuration" on the secondary side

The CoE folders of both halves of the EL6692 are used to save the process data configuration in the EL6692. The types and complexity of the definable process data are therefore limited. If the notes from the fig. "Get configuration" on the secondary side are displayed during the "Create Configuration" procedure, the

secondary side must be checked for correct transmission of the configuration. The primary configuration may need to be changed if the note shown in the fig. *Warning messages for complex data types* appears during "Get Configuration".

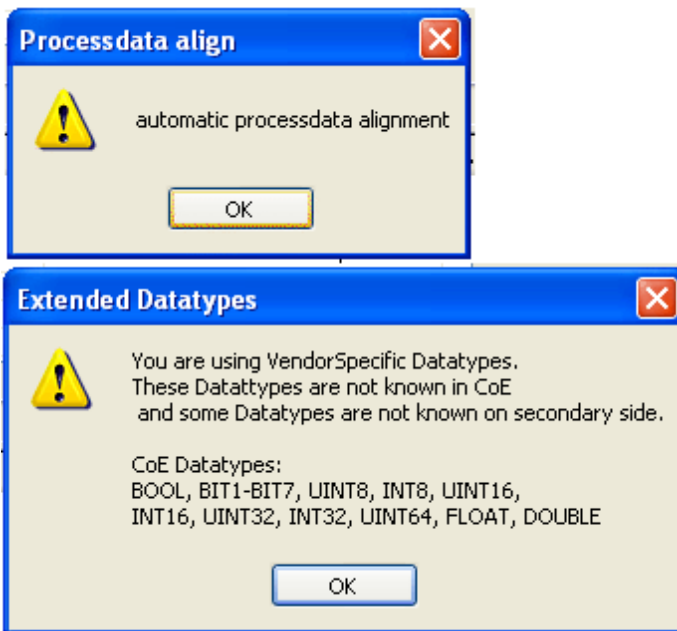


Fig. 141: Warning messages for complex data types

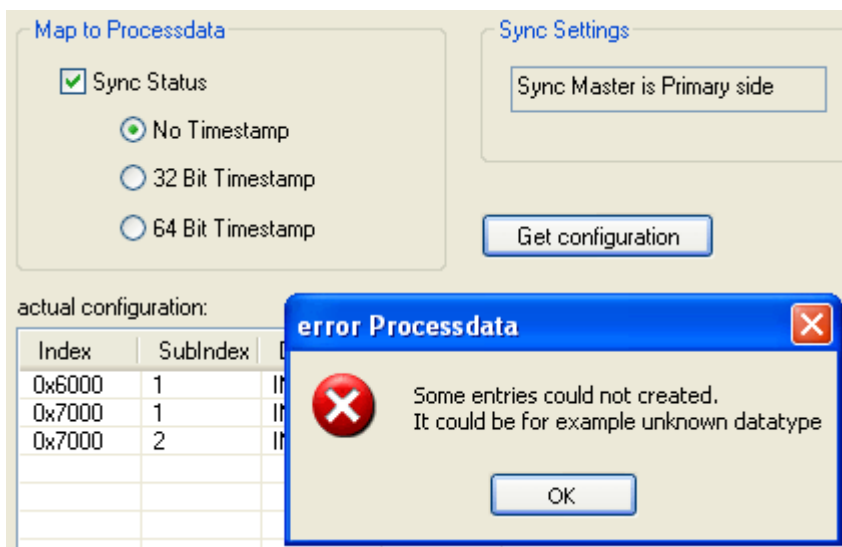


Fig. 142: Warning message for complex data types

## Application notes

### Process data

The following must be observed in relation to the EL6692 process data:

- **Data quantity**  
a maximum of 480 bytes can be transmitted in each direction.
- **Number of variables**  
no more than 59 variables should be configured per side (primary/secondary) and data direction (input/output). In total, 236 variables (4 \* 59) can be cyclically transmitted.
- **Alignment**  
The order of the variables of both data sides must be the same. In addition, 2-byte alignment must be complied with. This is automatically created with *Create/GetConfiguration*.

| PDO List |      |            |
|----------|------|------------|
| Index    | Size | Name       |
| 0x1A00   | 6.0  | IO Inputs  |
| 0x1600   | 2.0  | IO Outputs |

Fig. 143: Correct 2-byte alignment of the PDO list

**Diagnostics**

**Message**

'Term 3 (EL6692)' (1003) 'PS': CoE ('InitDown' 0x1600:00) - SDO Abort ('General parameter incompatibility reason.', 0x06040043): 'download pdo 0x1600 entries'.

Fig. 144: Error message during bootup – General parameter incompatibility reason

**Solution**

Variable alignment not complied with, secondary side doesn't match primary side  
 → transmit the variable configuration online to the secondary side using Create/GetConfiguration

**Message**

(EL6692)' (1002) 'PS': CoE ('InitDown' 0x1a00:00) - SDO Abort ('Data type does not match, length of service parameter does not match', 0x06070010): 'download pdo 0x1A00 entries'.

Fig. 145: Error message during bootup – Data type does not match, length of service parameter does not match, download pdo entries

**Solution**

more than 59 variables created, ADS logger message during bootup, data transmission in OP state may be possible  
 → use fewer variables OR  
 → do not download the PDO configuration (then up to 255 variables are possible without error message, but this should not be used)

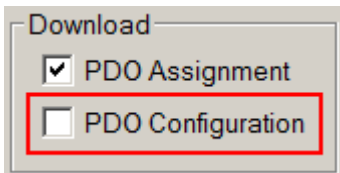


Fig. 146: Deactivate PDO configuration download

**Operation without primary side**

The central controller of the EL669x is supplied with power from the primary side. If the EL669x is started from the secondary side without primary voltage, the secondary side is therefore unable to change from the INIT state. Similarly, if both sides were in operation, after a power failure on the primary side a regular status change is no longer possible on the secondary side. The secondary side responds to associated requests with ERROR.

For an automatic restart of the secondary bridge side, function blocks from the TcEtherCAT.lib should be used in the secondary-side PLC in this case. The default settings of the system manager "Relnit after ComError" and "Auto Restore States" are insufficient in this case.

The LED RUN SEC responds accordingly.

## Operation mode

The EL6692-0002 can be operated with or without distributed clocks support. If no DC synchronization is required DC support can be switched off and the terminal can be operated in FreeRun mode. In this case the corresponding diagnostic process data are irrelevant. If DC support is desired, it must be activated on the primary and secondary side; see the fig. *Mode switching*. The operating mode "DC synchronous" only affects the operation of the EL6692 with distributed clocks synchronization.

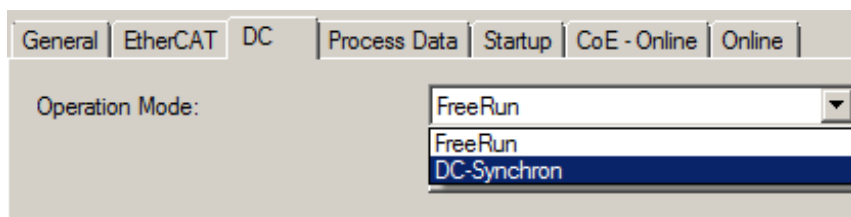


Fig. 147: Mode switching

### ● Support for distributed clocks with EL6692 in Beckhoff TwinCAT

**i** Full support for distributed clocks synchronization with the EL6692 is provided from TwinCAT version 2.10 build 1340 or higher. Previous versions are limited to process data exchange.

## Data transfer through the EL6692

Transporting the configured process data from one EtherCAT side to the other takes a certain amount of time, depending on the number of bytes. The data transfer is free-running and therefore not synchronized with one of the two EtherCAT sides. As a typical sample, measurement of an EL6692 with FW01 shows the following values:

| EL6692 Throughput, FW01 |          | Task cycle time |             |
|-------------------------|----------|-----------------|-------------|
| process data            |          | 100 $\mu$ s     | 500 $\mu$ s |
|                         | 50 byte  | max. 1,2 ms     | max. 1,5 ms |
|                         |          | ø: 0,7 ms       | ø: 1 ms     |
|                         | 100 byte | max. 1,3 ms     |             |
|                         |          | ø: 0,8 ms       |             |
|                         | 200 byte | max. 1,5 ms     |             |
| ø: 1,0 ms               |          |                 |             |

Fig. 148: Typical throughput time measurement for a EL6692 (FW01) - Beckhoff reserves the right to modifications without notification.

The user is responsible for ensuring (via the diagnostic variables available for the EL6692 or a higher-level handshake) that data overload of the EL6692 is avoided with small EtherCAT cycle times (i.e. the quantity of data supplied to the device must not exceed the quantity the device can transport to the other side).

## Link with process data SyncMode

The "SyncMode" status process data has 2 bits. IEC61131-PLC contains no data type that can be linked with this process data directly. This data was nevertheless chosen deliberately for reasons of data transfer efficiency.

An input variable of the type *Byte* must be defined in the task/program for linking; see the fig. *Definition of byte process data*.

```
VAR
    Var109 AT %I*:BYTE;
END_VAR
```

Fig. 149: Byte process data definition



After double-clicking on *SyncMode* in the system manager, the dialog box from the fig. *Definition of a variable of type Byte for linking* opens.

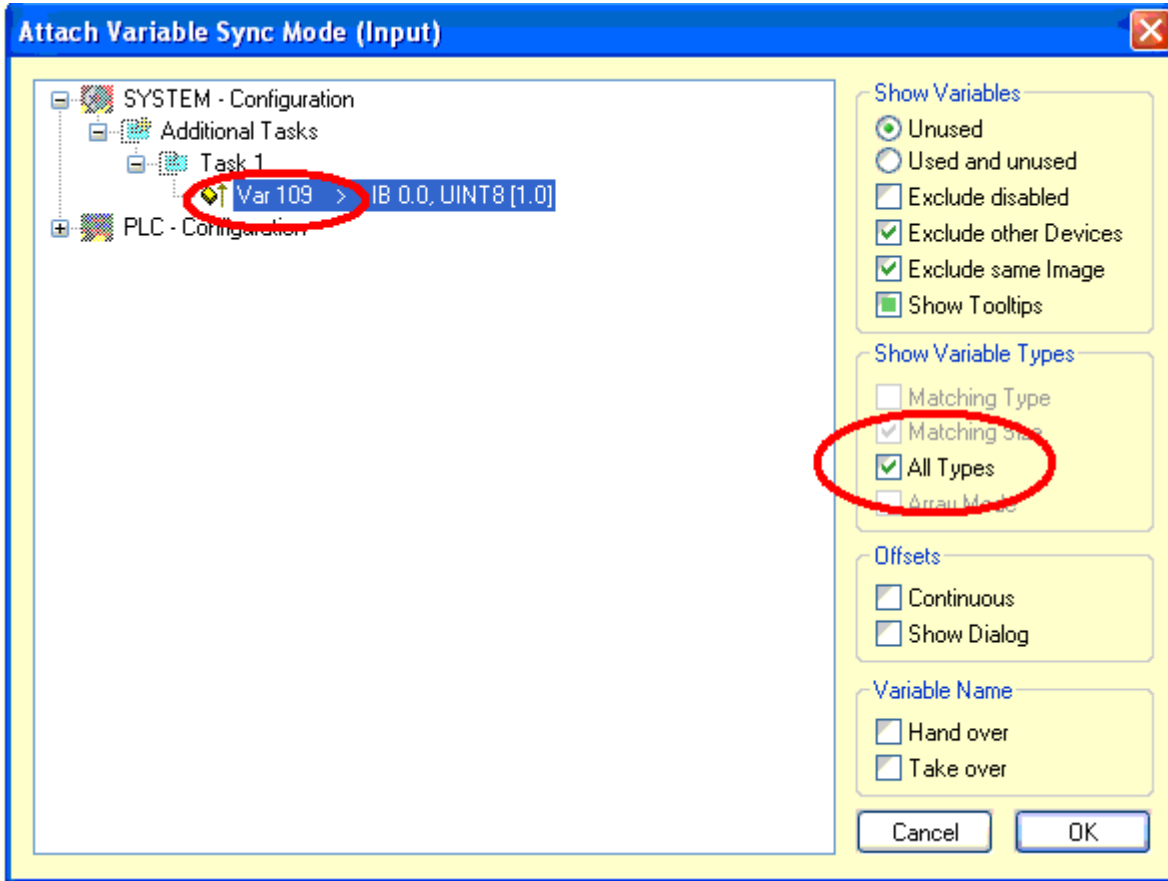


Fig. 150: Definition of a variable of type Byte for linking

Select "All Types" to make Var109 available for selection. Otherwise only exactly matching 2 bit process data would be displayed. Now a dialog box opens for the specification of the offset; see the fig. *Offset specification*. Confirm the suggestion.

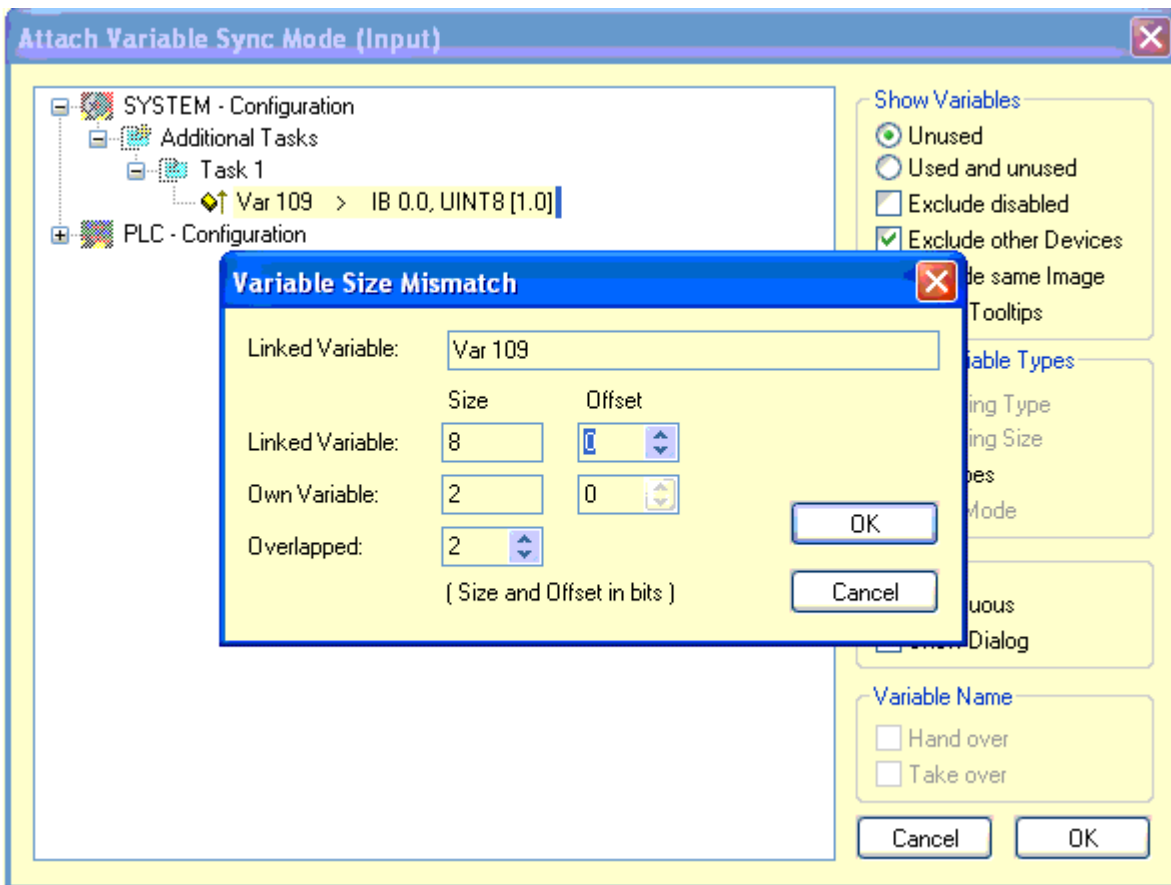


Fig. 151: Offset specification

### Extent of process data

The EL6692 can transport a maximum of 480 bytes in each direction. In each transport direction the data may consist of a single large process data such as a structure or an array.

### Declaration of bit process data

In the IEC61131-PLC a bit process data is only created on a *bit* address if it was defined "alone".

```
PROGRAM MAIN
VAR
  bBit1 AT %I*:BOOL;
  bBit2 AT %IX0.2:BOOL;
  abBitArray AT %I*:ARRAY[0..9] OF BOOL;
END_VAR
```

Fig. 152: Bit value declaration

In Fig. *Bit values declaration* "bBit 1" and "bBit2" are created as individual bit process data and interpreted as such in the link dialog of the System Manager. The abBitArray array consisting of 10 bits is created as 10 bytes, since the IEC61131-PLC is unable to handle an array consisting of individual bits.

The same applies to bit variables in structures (in IEC61131-PLC).

## 5.5 Handling acyclic data

In addition to cyclic PDO, the EL6692 can also transport so-called acyclic data (mailbox traffic) to the respective opposite side. These data streams can be of different types, different protocols are defined, which are transported in this way.


Note: the EL6692 as predecessor of the EL6695 offers here only limited functions (see following explanations), with extensive requirements concerning the protocol transport please check the EL6695 for usability.

### EoE (Ethernet-over-EtherCAT)

Explanation of the EoE function in the EL6692: When the operating system (Windows) is instructed to send an IP frame (e.g. to 192.168.2.1), it uses a broadcast to all available network adapters to determine through which adapter the destination address can be reached. The EL6692 transports this broadcast, which reaches it through acyclic mailbox communication it to the other side, which responds based on its network mask.

### AoE (ADS-over-EtherCAT)

ADS (Automation Device Specification) is a Beckhoff-specific protocol for data exchange between hardware- or software-based devices. The configuration of these ADS telegrams is described in the Beckhoff Information System. AoE is a VoE (vendor-specific protocol over EtherCAT). The EL6692 can transport ADS telegrams to the other side directly (AoE), i.e. without underlying IP channel.

To this end the EL6692 must be entered in the System Manager as an ADS gateway; see the following sample (<https://infosys.beckhoff.com/content/1033/el6692/Resources/zip/2460148491.zip>) 

Two PCs with TwinCAT 2.10, b1340 and an EL6692 with Firmware 05 are used.

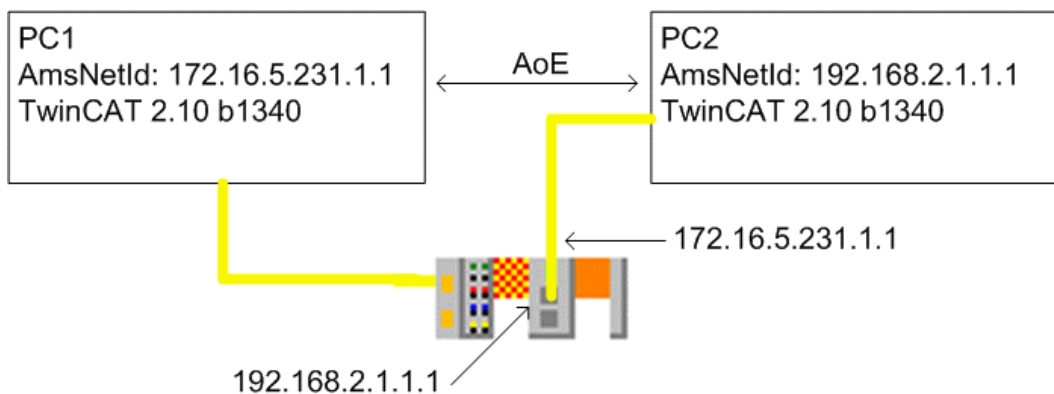


Fig. 153: AoE demo configuration

Both PCs send 10,000 bytes of data each to the other side. An Ads-Write is used on the sender side and the Indication/Response method on the receiver side. To this end, the target AmsNetId must be entered in both System Manager configurations so that, for sample, the ADS router in PC1 can reach the ADS device '192.168.2.1.1.1' via the 'channel' EL6692.

The procedure is as follows on the primary side; the secondary side (PC2) follows suit.

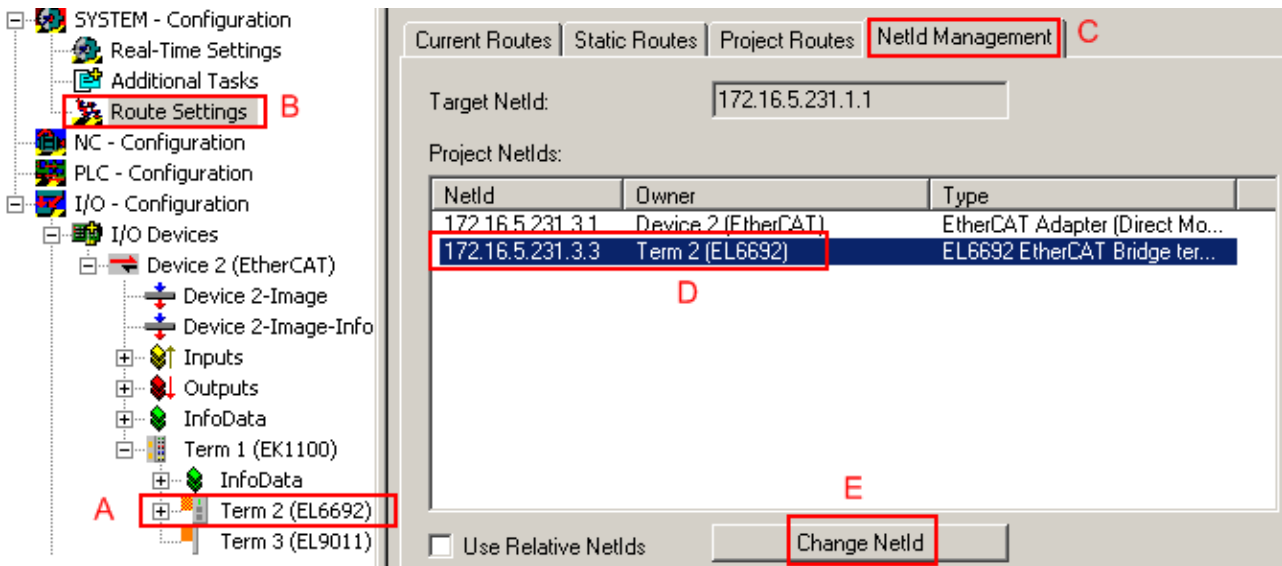


Fig. 154: Entering the EL6692 as an ADS gateway

If an EL6692 (A) exists, the NetId of the EL6692 (D) can be changed (E) under *RouteSettings* (B), *NetIdManagement* (C).

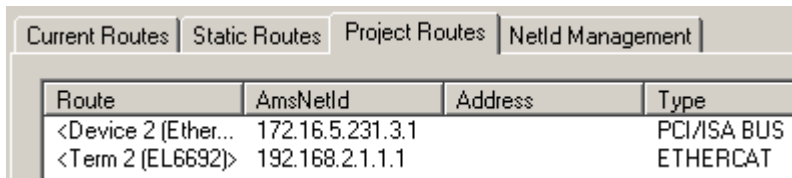


Fig. 155: New NetId entered

After activating the configuration and restarting TwinCAT, the ADS router is now instructed to send queries to the ADS device 192.168.2.1.1.1 via the AoE interface of the EL6692. AoE must be activated in the advanced settings of the EL6692.

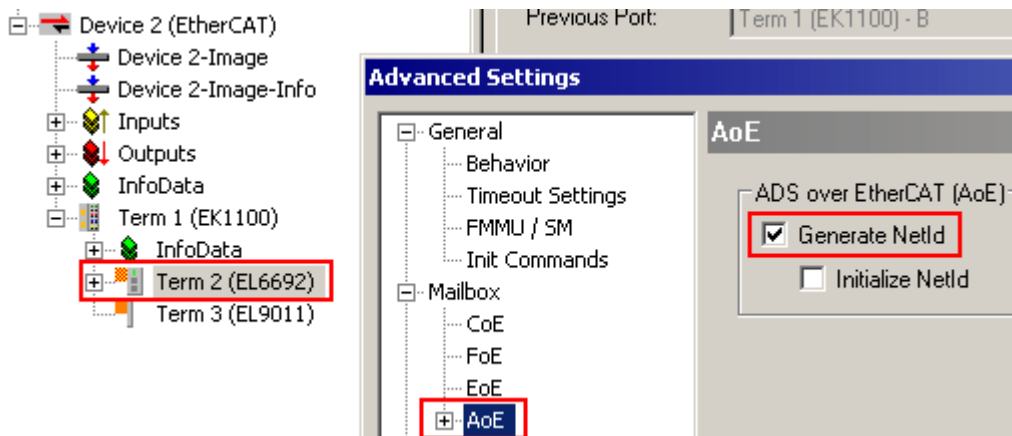


Fig. 156: New NetId entered

**Forwarding blocking for acyclic data (mailbox traffic)**

As of firmware 12, the forwarding of some protocols can be specifically blocked. To do this, the following bits must be set in the CoE index 0xF800:01:

- AoE: 0x8000
- EoE: 0x4000
- FoE: 0x2000
- VoE: 0x0800

Blocking protocols can be helpful if EtherCAT devices communicate frequently, e.g. during the start-up phase of the system, and this is transmitted by the bridge terminal to the opposite EtherCAT system and causes unwanted reactions there. E.g. the lock can be entered as StartUp command in the transition INIT->PreOP:

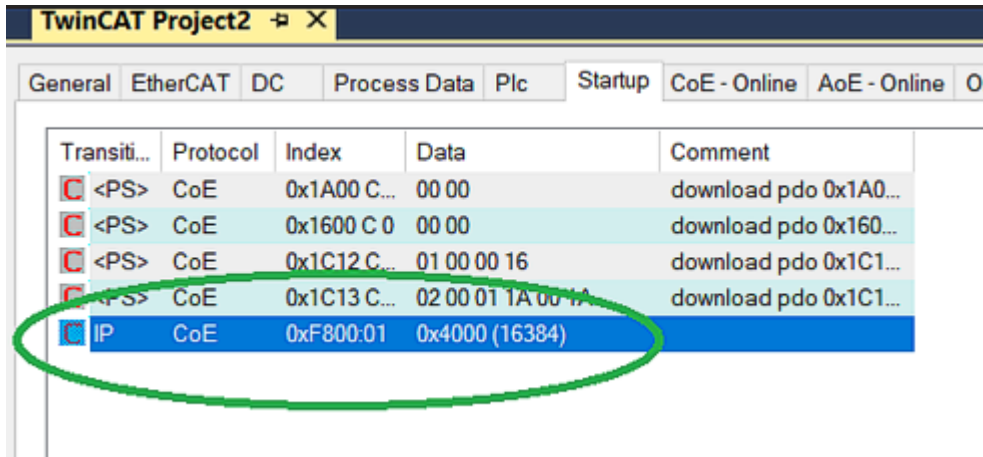


Fig. 157: Insert forwarding blocking as StartUp command

## 5.6 External TwinCAT synchronization

### Sample: EL6692 bridge terminal

#### Using the sample programs

This document contains sample applications of our products for certain areas of application. The application notices provided here are based on typical features of our products and only serve as samples. The notices contained in this document explicitly do not refer to specific applications. The customer is therefore responsible for assessing and deciding whether the product is suitable for a particular application. We accept no responsibility for the completeness and correctness of the source code contained in this document. We reserve the right to modify the content of this document at any time and accept no responsibility for errors and missing information.

In this example, two Beckhoff IPCs with TwinCAT 2.11, b1539, will be synchronized with each other. One PC is the master clock, the second (slave clock) synchronizes its 'time' to the first. As the fieldbus, EtherCAT makes the necessary operating resources available, in particular EtherCAT's own synchronization mechanism, distributed clocks.

The procedure is as explained in the previous chapter.

The following must be observed:

- The master PC works autonomously on the basis of its DC time
- Following the start of TwinCAT, the slave PC readjusts its distributed clocks time to the master IPC:
  - At the start of EtherCAT, the initial offset between the two times is determined.
  - The subsequent adjustment keeps this offset constant and makes it known.
  - The readjustment takes place continuously.
- In the case of failure of the synchronization (interruption of the connection, restart of one of the systems), the behavior is as follows:
  - If the adjustment restarts in the slave PC, a new offset is calculated there and made known.
  - The application must therefore continuously observe this offset.
- The local DC time must still be used for tasks related to the respective station hardware (EtherCAT slaves, terminals).
- The EtherCAT cycle time must be identical in both systems.
- If different configurations are used in the two systems, i.e. the number, type and/or order of the slaves is different, the automatically calculated shift times will also differ.

Sample: Both systems use an EL2202-0100, which are both supposed to switch their output at the same time. A constant difference is measured, since different output shift times were calculated.


In the system with the smaller output shift time the shift time of the other system should be entered.

#### NOTE

##### Effect on devices when changing the shift times

Side effects relating to the function of the other slaves when the shift times are changed should be taken into account!

- In a controlled system the time offset between the systems is subject to certain fluctuations.

 Sample program (<https://infosys.beckhoff.com/content/1033/el6692/Resources/zip/2460148491.zip>), TwinCAT 2.11

Pay attention in the program to the use of 'signed' and 'unsigned' 64-bit variables as required.

## Topology

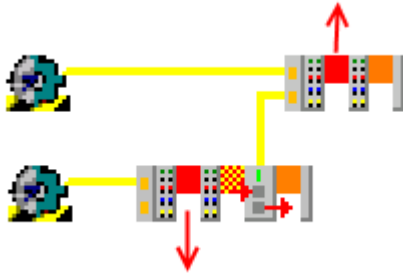


Fig. 158: Topology of the example program

Station Master: EK1100, EL2202, EL6692

Station Slave: EK1100, EL2202

In this example, the EL6692 is synchronized in the direction *PrimarySide* --> *SecondarySide* (RJ45 connection). Synchronization in the other direction is also possible.



### EL6692 documentation

Please note the information in the EL6692 documentation regarding the system behavior of this terminal.

## Demo program

In this demo program, the slave's own local DC time from the ReferenceClock in the EtherCAT strand is offset by the time difference to the external synchronization device. This offset therefore only makes sense on a platform that is a synchronization slave to one master.

The synchronization route can be

- another EtherCAT system; means: Beckhoff EL6692 bridge terminal (this example)
- an IEEE1588 system; means: Beckhoff EL6688 PTP terminal
- an arbitrary timer with time information (GPS, radio clock); means: TwinCAT 'External Synchronization' supplement

The principle:

TwinCAT cyclically receives (e.g. every second) a pair (64-bit, unit 1 ns) made up of an internal (DC) and an external time stamp. These two time stamps are originally obtained simultaneously. The offset between the two timebases is calculated from the initial difference and made known in the System Manager | EtherCAT device | InfoData. Furthermore, the slave TwinCAT readjusts its own local DC time from the course of the two time stamps with respect to each other.

Calculations:

- $\text{current control deviation} = \text{DcToExtOffset} - (\text{external time stamp} - \text{internal time stamp})$   
This value ('signed', 64-bit) is compared to an application-specific limit; the validity of the time is output if within the limit
- $\text{local synchronized time} = \text{local DC time} + \text{DcToExtOffset}$   
This 'nuLocalTime' ('unsigned', 64-bit) can now be used for data logging and events with a time reference to the master PC clock.

## Setting up TwinCAT 2.11

In the following procedure the complete system is set up as follows:

- EL6692 primary side (E-bus): Sync Master (i.e. reference clock)
- EL6692 secondary side (RJ45 sockets): Sync Slave (i.e. synchronized side)

The synchronization direction of the time can be set up the other way around; the instructions must then be followed analogously.

**Sync Master side**

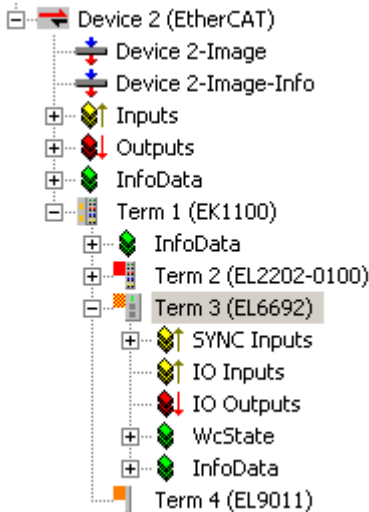


Fig. 159: Device on the master side

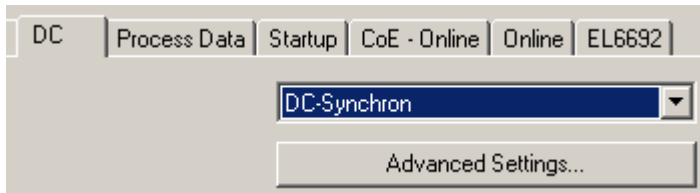


Fig. 160: Set the EL6692 PrimarySide to DC

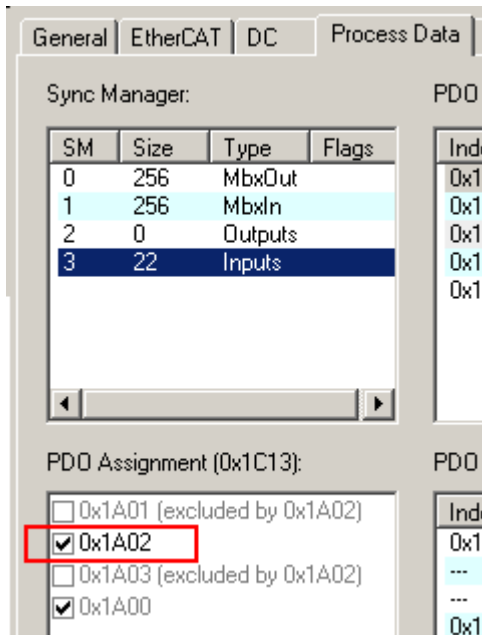


Fig. 161: Activate PDO 0x1A02 to display the time stamps

**Timestamp PDO**

**i** The activation of the timestamp PDO indicates to the TwinCAT software on the respective side that this side is to be synchronized, i.e. that it is the Sync Slave. It is not necessary to activate the timestamp PDO on the Sync Master side (i.e. the side that represents the reference clock).



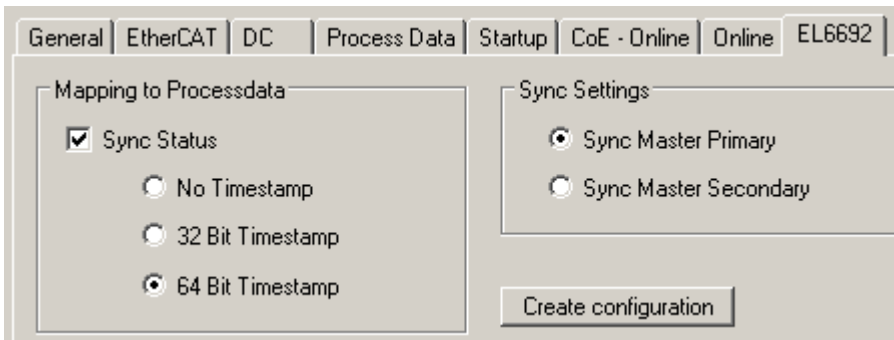


Fig. 162: Set the synchronization direction on the PrimarySide; in this case SyncSettings: Primary -> Secondary

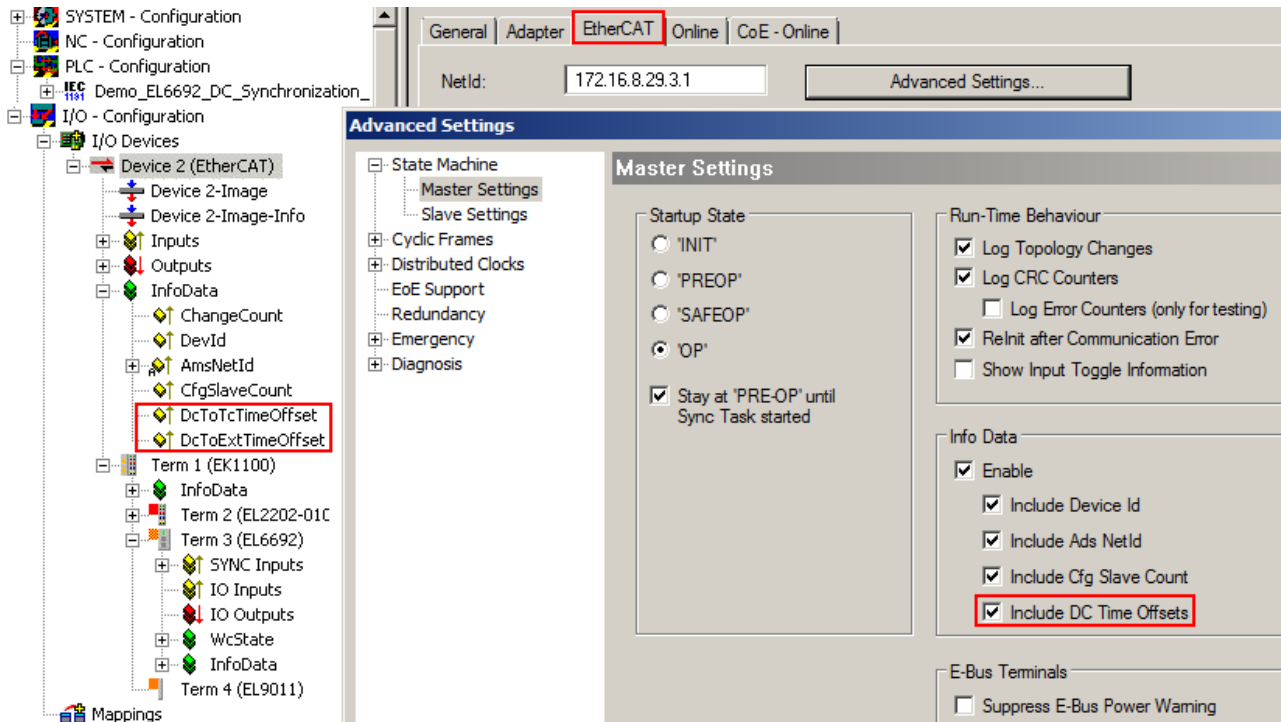


Fig. 163: Activate the display of the DC offsets in the EtherCAT master; can be evaluated on the master side

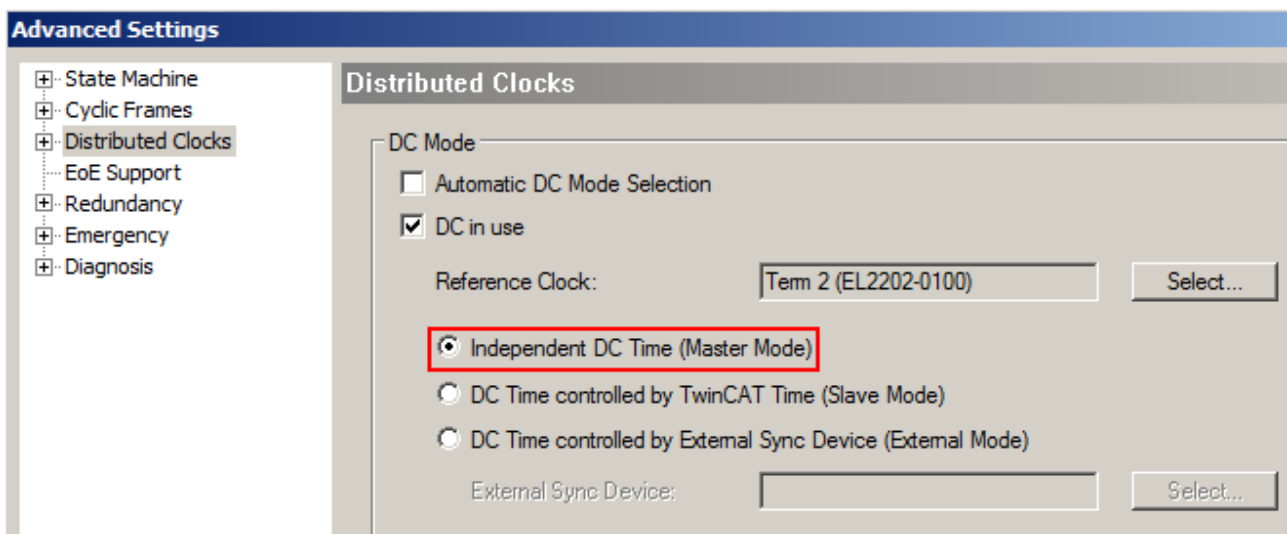


Fig. 164: Master PC works with its own ReferenceClock as a basis

TwinCAT can now be activated and started on this side. All devices must be in OP, WorkingCounter = 0, no LostFrames. The EL6692 time stamps on the PrimarySide remain at 0, because the SecondarySide has not yet been configured.

**Sync Slave side**

The EL6692, *SecondarySide* is set to DC and 0x1A02 according to the fig. *Set the EL6692 PrimarySide to DC and Activate PDO 0x1A02 to display the time stamp.*

After reloading the configuration (or restarting in *ConfigMode*, *FreeRun*), the synchronization direction can be read out by means of *GetConfiguration* on the *SecondarySide*, see the fig. *SecondarySide of the EL6692.*

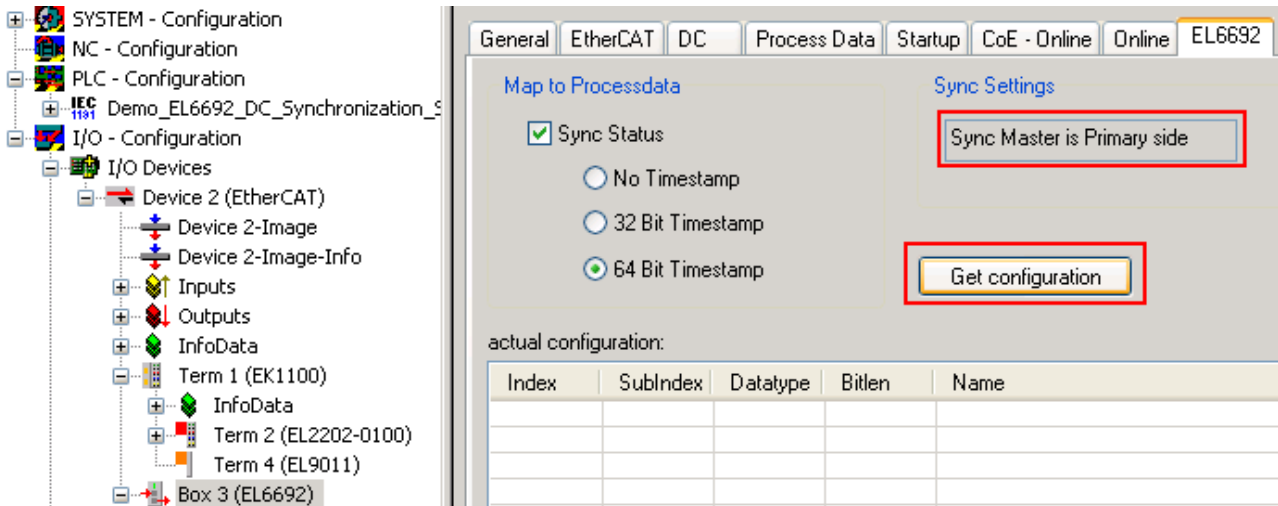


Fig. 165: SecondarySide of the EL6692

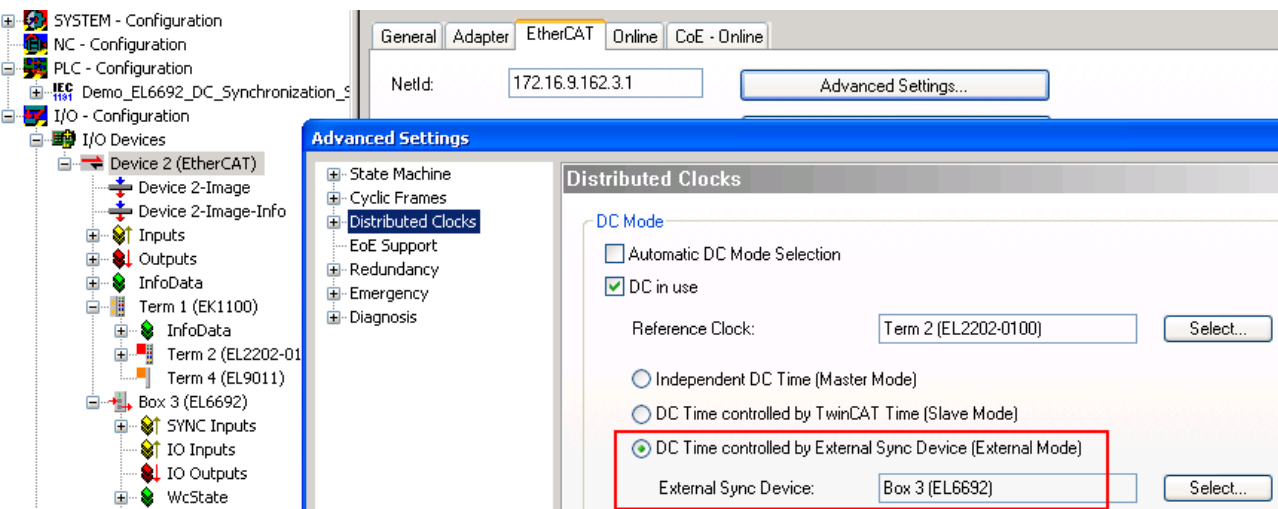


Fig. 166: EtherCAT master settings, slave side

After the restart, the DC function of the EL6692 is known to the EtherCAT master; therefore, it now offers this EL6692 as an *ExternalSyncDevice* in the DC dialogue.

The linking of the following variables is necessary for the evaluations; see the fig. *Slave side.*

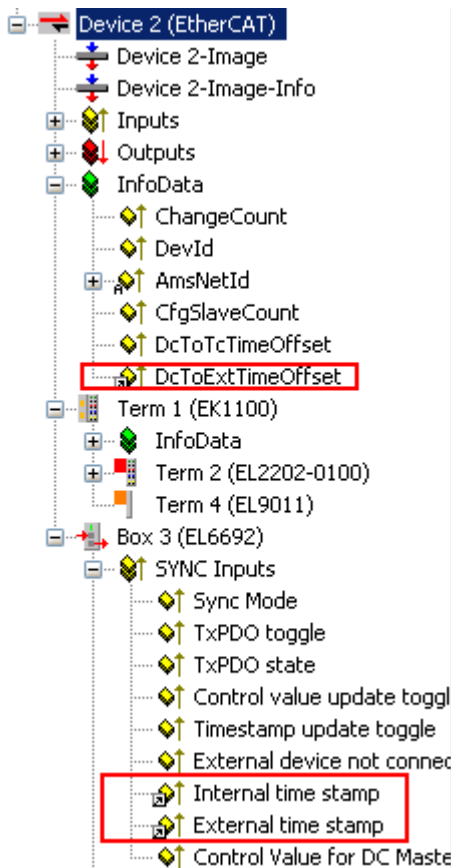


Fig. 167: Slave side

**NOTE**

**Demo program**

The following screenshots and information refer exclusively to the PLC demo program discussed here and the example code it contains, and not to the analysis functions of the system manager. See also the [note \[p. 134\]](#).

On the slave side, the start of the synchronization can be observed with the incorporated visualization.

Synchronization started

|  |                               |
|--|-------------------------------|
| local DC time (world time format):   | 2009-11-05-13:53:20.220000000 |
| local synchronized time (world time format): 2009-11-05-13:53:20.220000000 |                               |

Synchronization in Window  
window: 10000 ns

Synchronization  
in Progress

Fig. 168: Start slave side

Only the local DC time is available on the slave side immediately after the start.

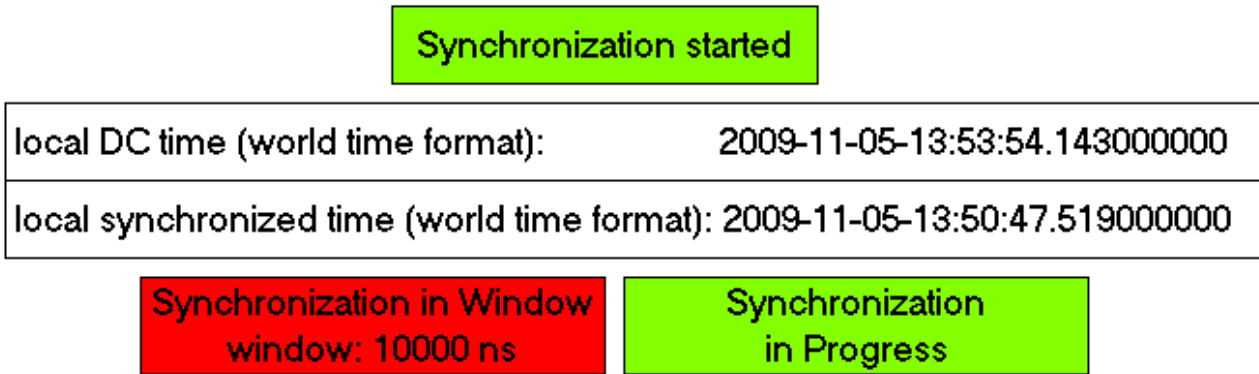


Fig. 169: Time stamp known

Following receipt of the first time stamps via the EL6692, the offset is known; in this case it is around 3 minutes different to the time of the IPC used. Synchronization has begun; in this sample a window of  $\pm 10 \mu\text{s}$  is to be achieved.

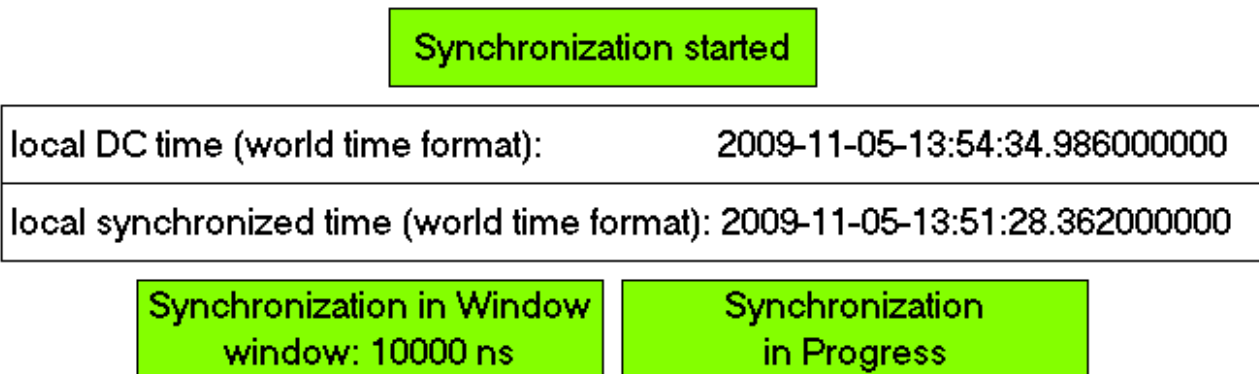


Fig. 170: Synchronization successful

## 5.7 Object description and parameterization

### ● EtherCAT XML Device Description

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

### ● Parameterization via the CoE list (CAN over EtherCAT)

**i** The EtherCAT device is parameterized via the CoE-Online tab [▶ 101] (double-click on the respective object) or via the Process Data tab [▶ 98](allocation of PDOs). Please note the following general CoE notes [▶ 25] when using/manipulating the CoE parameters:

- Keep a startup list if components have to be replaced
- Differentiation between online/offline dictionary, existence of current XML description
- use “CoE reload” for resetting changes

### Introduction

The CoE overview contains objects for different intended applications:

- Objects required for parameterization during commissioning:
  - Restore object index 0x1011
  - Configuration data index 0x80n0
- Objects intended for regular operation, e.g. through ADS access.
- Profile-specific objects:
  - Configuration data (vendor-specific) index 0x80nF
  - Input data index 0x60n0
  - Information and diagnostic data index 0x80nE, 0xF000, 0xF008, 0xF010
- Standard objects

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

### 5.7.1 Objects for commissioning

#### Index 10F4 External synchronization status

| Index (hex) | Name                              | Meaning  | Data type | Flags | Default                        |
|-------------|-----------------------------------|--|-----------|-------|--------------------------------|
| 10F4:0      | External synchronization status   | Information about the synchronization status   | UINT8     | RO    | 0x13 (19 <sub>dec</sub> )      |
| 10F4:01     | Sync Mode                         | Synchronization mode<br>0 = no synchronization<br>1 = secondary side is Sync Master<br>2 = primary side is Sync Master | BIT2      | RO    | 0x00 (0 <sub>dec</sub> )       |
| 10F4:0E     | Control value update toggle       | Bit toggles when a new control value is available  | BOOLEAN   | RO    | 0x00 (0 <sub>dec</sub> )       |
| 10F4:0F     | Time stamp update toggle          | Bit toggles when new DC data were supplied   | BOOLEAN   | RO    | 0x00 (0 <sub>dec</sub> )       |
| 10F4:10     | External device not connected     | 0 = other side is connected to its EtherCAT fieldbus<br>1 = other side is not connected to its EtherCAT fieldbus       | BOOLEAN   | RO    | 0x00 (0 <sub>dec</sub> )       |
| 10F4:11     | Internal time stamp               | Distributed clocks time on the current side (primary or secondary)   | UINT64    | RO    | -                              |
| 10F4:12     | External time stamp               | Distributed clocks time on the other side (primary or secondary)   | UINT64    | RO    | -                              |
| 10F4:13     | Control Value for DC Master Clock | Offset for correction of the lower priority reference clock  | INT32     | RO    | 0x00000000 (0 <sub>dec</sub> ) |

## Index 10F5 External synchronization settings

| Index (hex) | Name                              | Meaning  | Data type | Flags         | Default                                       |
|-------------|-----------------------------------|--|-----------|---------------|---|
| 10F5:0      | External synchronization settings | Setting for synchronizing the EtherCAT bridge                                    | UINT8     | RO            | 0x12 (18 <sub>dec</sub> )                     |
| 10F5:01     | Sync master                       | 0: Sync Master is on the primary side<br>1: Sync Master is on the secondary side | BOOLEAN   | RW<br>(PREOP) | 0x00 (0 <sub>dec</sub> )                      |
| 10F5:02     | 32 Bit time stamps                | 0: 64-bit Timestamps<br>1: 32-bit Timestamps                                     | BOOLEAN   | RW            | 0x00 (0 <sub>dec</sub> )                      |
| 10F5:11     | Control Interval (ms)             | Interval in ms for calculating the control value                                 | UINT16    | RW            | 0x0000 (0 <sub>dec</sub> )                    |
| 10F5:12     | Additional System Time            | Additional DC time for calculating the control value                             | UINT64    | RW            | 0x000000000<br>0000000<br>(0 <sub>dec</sub> ) |

## 5.7.2 Objects for regular operation

The EL6692 has no such objects.

## 5.7.3 Input data

### Index 6000 Input Data

| Index (hex) | Name       | Meaning  | Data type | Flags | Default                  |
|-------------|------------|--|-----------|-------|--------------------------|
| 6000:0      | Input Data | Declared input process data [► 124](dynamically created) | UINT8     | RO    | 0x00 (0 <sub>dec</sub> ) |
| 6000:01     | -          | -  | -         | -     | -                        |
| ...         |            |  |           |       |                          |
| 6000:FF     | -          | -  | -         | -     | -                        |

## 5.7.4 Output data

### Index 7000 Output Data

| Index (hex) | Name        | Meaning   | Data type | Flags | Default                  |
|-------------|-------------|---|-----------|-------|--------------------------|
| 7000:0      | Output Data | Declared output process data [► 124](dynamically created) | UINT8     | RO    | 0x00 (0 <sub>dec</sub> ) |
| 7000:01     | -           | -   | -         | -     | -                        |
| ...         |             |   |           |       |                          |
| 7000:FF     | -           | -   | -         | -     | -                        |

## 5.7.5 Information and diagnostic data

### Index F000 Modular device profile

| Index (hex) | Name                      | Meaning  | Data type | Flags | Default                     |
|-------------|---------------------------|--|-----------|-------|-----------------------------|
| F000:0      | Modular device profile    | General information for the modular device profile       | UINT8     | RO    | 0x02 (2 <sub>dec</sub> )    |
| F000:01     | Module index distance     | Index distance of the objects of the individual channels | UINT16    | RO    | 0x0010 (16 <sub>dec</sub> ) |
| F000:02     | Maximum number of modules | Number of channels                                       | UINT16    | RO    | 0x0001 (1 <sub>dec</sub> )  |

### Index F008 Code word

| Index (hex) | Name      | Meaning            | Data type | Flags | Default                           |
|-------------|-----------|--------------------|-----------|-------|-----------------------------------|
| F008:0      | Code word | currently reserved | UINT32    | RW    | 0x00000000<br>(0 <sub>dec</sub> ) |

**Index F010 Module list**

| Index (hex) | Name         | Meaning       | Data type | Flags | Default                        |
|-------------|--------------|---------------|-----------|-------|--------------------------------|
| F010:0      | Module list  | Max. subindex | UINT8     | RW    | 0x01 (1 <sub>dec</sub> )       |
| F010:01     | SubIndex 001 | -             | UINT32    | RW    | 0x00000000 (0 <sub>dec</sub> ) |

**5.7.6 Standard objects (0x1000-0x1FFF)**

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

**Index 1000 Device type**

| Index (hex) | Name        | Meaning   | Data type | Flags | Default                           |
|-------------|-------------|---|-----------|-------|-----------------------------------|
| 1000:0      | Device type | Device type of the EtherCAT slave: The Lo-Word contains the CoE profile used (5001). The Hi-Word contains the module profile according to the modular device profile. | UINT32    | RO    | 0x00001389 (5001 <sub>dec</sub> ) |

**Index 1008 Device name**

| Index (hex) | Name        | Meaning                           | Data type | Flags | Default     |
|-------------|-------------|-----------------------------------|-----------|-------|-------------|
| 1008:0      | Device name | Device name of the EtherCAT slave | STRING    | RO    | EL6692-0000 |

**Index 1009 Hardware version**

| Index (hex) | Name             | Meaning                                | Data type | Flags | Default |
|-------------|------------------|--|-----------|-------|---------|
| 1009:0      | Hardware version | Hardware version of the EtherCAT slave | STRING    | RO    | 00      |

**Index 100A Software version**

| Index (hex) | Name             | Meaning                                | Data type | Flags | Default |
|-------------|------------------|--|-----------|-------|---------|
| 100A:0      | Software version | Firmware version of the EtherCAT slave | STRING    | RO    | 02      |

**Index 1018 Identity**

| Index (hex) | Name          | Meaning   | Data type | Flags | Default                                |
|-------------|---------------|---|-----------|-------|--|
| 1018:0      | Identity      | Information for identifying the slave   | UINT8     | RO    | 0x04 (4 <sub>dec</sub> )               |
| 1018:01     | Vendor ID     | Vendor ID of the EtherCAT slave   | UINT32    | RO    | 0x00000002 (2 <sub>dec</sub> )         |
| 1018:02     | Product code  | Product code of the EtherCAT slave  | UINT32    | RO    | 0x1A243052 (438579282 <sub>dec</sub> ) |
| 1018:03     | Revision      | Revision number of the EtherCAT slave; the low word (bit 0-15) indicates the special terminal number, the high word (bit 16-31) refers to the device description  | UINT32    | RO    | 0x00100000 (1048576 <sub>dec</sub> )   |
| 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 <sub>dec</sub> )         |

**Index 1600 RxPDO-Map**

| Index (hex) | Name      | Meaning   | Data type | Flags | Default                  |
|-------------|-----------|---|-----------|-------|--------------------------|
| 1600:0      | RxPDO-Map | PDO mapping RxPDO 1 (PDO mapping of the declared output process data) | UINT8     | RW    | 0x00 (0 <sub>dec</sub> ) |
| 1600:01     | -         | -   | -         | -     | -                        |
| ...         |           |   |           |       |                          |
| 1600:FF     | -         | -   | -         | -     | -                        |

**Index 1801 TxPDO-Par External Sync Compact**

| Index (hex) | Name                            | Meaning  | Data type       | Flags | Default                  |
|-------------|---------------------------------|--|-----------------|-------|--------------------------|
| 1801:0      | TxPDO-Par External Sync Compact | PDO parameter TxPDO 2  | UINT8           | RO    | 0x09 (9 <sub>dec</sub> ) |
| 1801:06     | Exclude TxPDOs                  | Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 2 | OCTET-STRING[4] | RO    | 02 1A 03 1A              |
| 1801:07     | TxPDO State                     | The TxPDO state is set if it was not possible to correctly read in the associated input data             | BOOLEAN         | RO    | 0x00 (0 <sub>dec</sub> ) |
| 1801:09     | TxPDO Toggle                    | The TxPDO toggle is toggled with each update the corresponding input data                                | BOOLEAN         | RO    | 0x00 (0 <sub>dec</sub> ) |

**Index 1802 TxPDO-Par External Sync**

| Index (hex) | Name                    | Meaning  | Data type       | Flags | Default                  |
|-------------|-------------------------|--|-----------------|-------|--------------------------|
| 1802:0      | TxPDO-Par External Sync | PDO parameter TxPDO 3  | UINT8           | RO    | 0x09 (9 <sub>dec</sub> ) |
| 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    | 01 1A 03 1A              |
| 1802:07     | TxPDO State             | The TxPDO state is set if it was not possible to correctly read in the associated input data             | BOOLEAN         | RO    | 0x00 (0 <sub>dec</sub> ) |
| 1802:09     | TxPDO Toggle            | The TxPDO toggle is toggled with each update the corresponding input data                                | BOOLEAN         | RO    | 0x00 (0 <sub>dec</sub> ) |

**Index 1803 TxPDO-Par External Sync (32 Bit)**

| Index (hex) | Name                             | Meaning  | Data type       | Flags | Default                  |
|-------------|----------------------------------|--|-----------------|-------|--------------------------|
| 1803:0      | TxPDO-Par External Sync (32 Bit) | PDO parameter TxPDO 4  | UINT8           | RO    | 0x09 (9 <sub>dec</sub> ) |
| 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    | 01 1A 02 1A              |
| 1803:07     | TxPDO State                      | The TxPDO state is set if it was not possible to correctly read in the associated input data             | BOOLEAN         | RO    | 0x00 (0 <sub>dec</sub> ) |
| 1803:09     | TxPDO Toggle                     | The TxPDO toggle is toggled with each update the corresponding input data                                | BOOLEAN         | RO    | 0x00 (0 <sub>dec</sub> ) |

**Index 1A00 TxPDO-Map**

| Index (hex) | Name      | Meaning  | Data type | Flags | Default                  |
|-------------|-----------|--|-----------|-------|--------------------------|
| 1A00:0      | TxPDO Map | PDO mapping TxPDO 1 (PDO mapping of the declared input process data) | UINT8     | RW    | 0x00 (0 <sub>dec</sub> ) |
| 1A00:01     | -         | -  | -         | -     | -                        |
| ...         |           |  |           |       |                          |
| 1A00:FF     | -         | -  | -         | -     | -                        |

**Index 1A02 TxPDO-Map External Sync Compact**

| Index (hex) | Name                            | Meaning  | Data type | Flags | Default                  |
|-------------|---------------------------------|--|-----------|-------|--------------------------|
| 1A01:0      | TxPDO-Map External Sync Compact | PDO Mapping TxPDO 2  | UINT8     | RW    | 0x08 (8 <sub>dec</sub> ) |
| 1A01:01     | SubIndex 001                    | 1. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x02)                                 | UINT32    | RW    | 0x10F4:02, 2             |
| 1A01:02     | SubIndex 002                    | 2. PDO Mapping entry (6 bits align)  | UINT32    | RW    | 0x0000:00, 6             |
| 1A01:03     | SubIndex 003                    | 3. PDO Mapping entry (3 bits align)  | UINT32    | RW    | 0x0000:00, 3             |
| 1A01:04     | SubIndex 004                    | 4. PDO Mapping entry (object 0x1800, entry 0x09)   | UINT32    | RW    | 0x1800:09, 1             |
| 1A01:05     | SubIndex 005                    | 5. PDO Mapping entry (object 0x1800, entry 0x07)   | UINT32    | RW    | 0x1800:07, 1             |
| 1A01:06     | SubIndex 006                    | 6. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x0E (Control value update toggle))   | UINT32    | RW    | 0x10F4:0E, 1             |
| 1A01:07     | SubIndex 007                    | 7. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x0F (Time stamp update toggle))      | UINT32    | RW    | 0x10F4:0F, 1             |
| 1A01:08     | SubIndex 008                    | 8. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x10 (External device not connected)) | UINT32    | RW    | 0x10F4:10, 1             |



**Index 1A02 TxPDO-Map External Sync**

| Index (hex) | Name                    | Meaning   | Data type | Flags | Default                   |
|-------------|-------------------------|---|-----------|-------|---------------------------|
| 1A02:0      | TxPDO-Map External Sync | PDO Mapping TxPDO 3   | UINT8     | RW    | 0x0B (11 <sub>dec</sub> ) |
| 1A02:01     | SubIndex 001            | 1. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x02)                                      | UINT32    | RW    | 0x10F4:02, 2              |
| 1A02:02     | SubIndex 002            | 2. PDO Mapping entry (6 bits align)   | UINT32    | RW    | 0x0000:00, 6              |
| 1A02:03     | SubIndex 003            | 3. PDO Mapping entry (3 bits align)   | UINT32    | RW    | 0x0000:00, 3              |
| 1A02:04     | SubIndex 004            | 4. PDO Mapping entry (object 0x1800, entry 0x09)  | UINT32    | RW    | 0x1800:09, 1              |
| 1A02:05     | SubIndex 005            | 5. PDO Mapping entry (object 0x1800, entry 0x07)  | UINT32    | RW    | 0x1800:07, 1              |
| 1A02:06     | SubIndex 006            | 6. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x0E (Control value update toggle))        | UINT32    | RW    | 0x10F4:0E, 1              |
| 1A02:07     | SubIndex 007            | 7. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x0F (Time stamp update toggle))           | UINT32    | RW    | 0x10F4:0F, 1              |
| 1A02:08     | SubIndex 008            | 8. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x10 (External device not connected))      | UINT32    | RW    | 0x10F4:10, 1              |
| 1A02:09     | SubIndex 009            | 9. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x11 (Internal time stamp))                | UINT32    | RW    | 0x10F4:11, 64             |
| 1A02:0A     | SubIndex 010            | 10. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x12 (External time stamp))               | UINT32    | RW    | 0x10F4:12, 64             |
| 1A02:0B     | SubIndex 011            | 11. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x13 (Control Value for DC Master Clock)) | UINT32    | RW    | 0x10F4:13, 32             |

**Index 1A03 TxPDO-Map External Sync (32 Bit)**

| Index (hex) | Name                             | Meaning   | Data type | Flags | Default                   |
|-------------|----------------------------------|---|-----------|-------|---------------------------|
| 1A03:0      | TxPDO-Map External Sync (32 Bit) | PDO Mapping TxPDO 4   | UINT8     | RW    | 0x0B (11 <sub>dec</sub> ) |
| 1A03:01     | SubIndex 001                     | 1. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x02)                                      | UINT32    | RW    | 0x10F4:02, 2              |
| 1A03:02     | SubIndex 002                     | 2. PDO Mapping entry (6 bits align)   | UINT32    | RW    | 0x0000:00, 6              |
| 1A03:03     | SubIndex 003                     | 3. PDO Mapping entry (3 bits align)   | UINT32    | RW    | 0x0000:00, 3              |
| 1A03:04     | SubIndex 004                     | 4. PDO Mapping entry (object 0x1800, entry 0x09)  | UINT32    | RW    | 0x1800:09, 1              |
| 1A03:05     | SubIndex 005                     | 5. PDO Mapping entry (object 0x1800, entry 0x07)  | UINT32    | RW    | 0x1800:07, 1              |
| 1A03:06     | SubIndex 006                     | 6. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x0E (Control value update toggle))        | UINT32    | RW    | 0x10F4:0E, 1              |
| 1A03:07     | SubIndex 007                     | 7. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x0F (Time stamp update toggle))           | UINT32    | RW    | 0x10F4:0F, 1              |
| 1A03:08     | SubIndex 008                     | 8. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x10 (External device not connected))      | UINT32    | RW    | 0x10F4:10, 1              |
| 1A03:09     | SubIndex 009                     | 9. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x11 (Internal time stamp))                | UINT32    | RW    | 0x10F4:11, 32             |
| 1A03:0A     | SubIndex 010                     | 10. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x12 (External time stamp))               | UINT32    | RW    | 0x10F4:12, 32             |
| 1A03:0B     | SubIndex 011                     | 11. PDO Mapping entry (object 0x10F4 (External synchronization status), entry 0x13 (Control Value for DC Master Clock)) | UINT32    | RW    | 0x10F4:13, 32             |

**Index 1C00 Sync manager type**

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

## Index 1C12 RxPDO assign

| Index (hex) | Name         | Meaning  | Data type | Flags | Default                    |
|-------------|--------------|--|-----------|-------|----------------------------|
| 1C12:0      | RxPDO assign | PDO Assign Outputs   | UINT8     | RW    | 0x00 (0 <sub>dec</sub> )   |
| 1C12:01     | SubIndex 001 | 1. allocated RxPDO (contains the index of the associated RxPDO mapping object) | UINT16    | RW    | 0x0000 (0 <sub>dec</sub> ) |

## Index 1C13 TxPDO assign

| Index (hex) | Name         | Meaning  | Data type | Flags | Default                       |
|-------------|--------------|--|-----------|-------|-------------------------------|
| 1C13:0      | TxPDO assign | PDO Assign Inputs  | UINT8     | RW    | 0x01 (1 <sub>dec</sub> )      |
| 1C13:01     | SubIndex 001 | 1. allocated TxPDO (contains the index of the associated TxPDO mapping object) | UINT16    | RW    | 0x1A01 (6657 <sub>dec</sub> ) |
| 1C13:02     | SubIndex 002 | 2. allocated TxPDO (contains the index of the associated TxPDO mapping object) | UINT16    | RW    | 0x0000 (0 <sub>dec</sub> )    |

## 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 <sub>dec</sub> )      |
| 1C32:01     | Sync mode               | Current synchronization mode: <ul style="list-style-type: none"> <li>0: Free Run</li> <li>1: Synchronous with SM 2 event</li> <li>2: DC-Mode - Synchronous with SYNC0 Event</li> <li>3: DC-Mode - Synchronous with SYNC1 event</li> </ul>  | UINT16    | RW    | 0x0000 (0 <sub>dec</sub> )     |
| 1C32:02     | Cycle time              | Cycle time (in ns): <ul style="list-style-type: none"> <li>Free Run: Cycle time of the local timer</li> <li>Synchronous with SM 2 event: Master cycle time</li> <li>DC mode: SYNC0/SYNC1 Cycle Time</li> </ul>   | UINT32    | RW    | 0x00000000 (0 <sub>dec</sub> ) |
| 1C32:03     | Shift time              | Time between SYNC0 event and output of the outputs (in ns, DC mode only)   | UINT32    | RW    | 0x00000000 (0 <sub>dec</sub> ) |
| 1C32:04     | Sync modes supported    | Supported synchronization modes: <ul style="list-style-type: none"> <li>Bit 0 = 1: free run is supported</li> <li>Bit 1 = 1: Synchron with SM 2 event is supported</li> <li>Bit 2-3 = 01: DC mode is supported</li> <li>Bit 4-5 = 10: Output shift with SYNC1 event (only DC mode)</li> <li>Bit 14 = 1: dynamic times (measurement through writing of 0x1C32:08 [▶ 146])</li> </ul>  | UINT16    | RO    | 0xC007 (49159 <sub>dec</sub> ) |
| 1C32:05     | Minimum cycle time      | Minimum cycle time (in ns)   | UINT32    | RO    | 0x00000000 (0 <sub>dec</sub> ) |
| 1C32:06     | Calc and copy time      | Minimum time between SYNC0 and SYNC1 event (in ns, DC mode only)   | UINT32    | RO    | 0x00000000 (0 <sub>dec</sub> ) |
| 1C32:08     | Command                 | <ul style="list-style-type: none"> <li>0: Measurement of the local cycle time is stopped</li> <li>1: Measurement of the local cycle time is started</li> </ul> <p>The entries 0x1C32:03 [▶ 146], 0x1C32:05 [▶ 146], 0x1C32:06 [▶ 146], 0x1C32:09 [▶ 146], 0x1C33:03 [▶ 147], 0x1C33:06 [▶ 146], 0x1C33:09 [▶ 147] are updated with the maximum measured values.<br/>For a subsequent measurement the measured values are reset</p> | UINT16    | RW    | 0x0000 (0 <sub>dec</sub> )     |
| 1C32:09     | Delay time              | Time between SYNC1 event and output of the outputs (in ns, DC mode only)   | UINT32    | RO    | 0x00000000 (0 <sub>dec</sub> ) |
| 1C32:0B     | SM event missed counter | Number of missed SM events in OPERATIONAL (DC mode only)   | UINT16    | RO    | 0x0000 (0 <sub>dec</sub> )     |
| 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 <sub>dec</sub> )     |
| 1C32:0D     | Shift too short counter | Number of occasions that the interval between SYNC0 and SYNC1 event was too short (DC mode only)   | UINT16    | RO    | 0x0000 (0 <sub>dec</sub> )     |
| 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 <sub>dec</sub> )       |

**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 <sub>dec</sub> )      |
| 1C33:01     | Sync mode               | Current synchronization mode: <ul style="list-style-type: none"> <li>• 0: Free Run</li> <li>• 1: Synchron with SM 3 Event (no outputs available)</li> <li>• 2: DC - Synchron with SYNC0 Event</li> <li>• 3: DC - Synchron with SYNC1 Event</li> <li>• 34: Synchron with SM 2 Event (outputs available)</li> </ul>   | UINT16    | RW    | 0x0000 (0 <sub>dec</sub> )     |
| 1C33:02     | Cycle time              | as <a href="#">0x1C32:02</a> [ <a href="#">▶ 146</a> ]  | UINT32    | RW    | 0x00000000 (0 <sub>dec</sub> ) |
| 1C33:03     | Shift time              | Time between SYNC0 event and reading of the inputs (in ns, only DC mode)  | UINT32    | RW    | 0x00000000 (0 <sub>dec</sub> ) |
| 1C33:04     | Sync modes supported    | Supported synchronization modes: <ul style="list-style-type: none"> <li>• Bit 0: free run is supported</li> <li>• Bit 1: Synchronous with SM 2 Event is supported (outputs available)</li> <li>• Bit 1: Synchronous with SM 3 Event is supported (no outputs available)</li> <li>• Bit 2-3 = 01: DC mode is supported</li> <li>• Bit 4-5 = 01: input shift through local event (outputs available)</li> <li>• Bit 4-5 = 10: input shift with SYNC1 event (no outputs available)</li> <li>• Bit 14 = 1: dynamic times (measurement through writing of <a href="#">0x1C32:08</a> [<a href="#">▶ 146</a>] or <a href="#">0x1C33:08</a> [<a href="#">▶ 147</a>])</li> </ul> | UINT16    | RO    | 0xC007 (49159 <sub>dec</sub> ) |
| 1C33:05     | Minimum cycle time      | as <a href="#">0x1C32:05</a> [ <a href="#">▶ 146</a> ]  | UINT32    | RO    | 0x00000000 (0 <sub>dec</sub> ) |
| 1C33:06     | Calc and copy time      | Time between reading of the inputs and availability of the inputs for the master (in ns, only DC mode)  | UINT32    | RO    | 0x00000000 (0 <sub>dec</sub> ) |
| 1C33:08     | Command                 | as <a href="#">0x1C32:08</a> [ <a href="#">▶ 146</a> ]  | UINT16    | RW    | 0x0000 (0 <sub>dec</sub> )     |
| 1C33:09     | Delay time              | Time between SYNC1 event and reading of the inputs (in ns, only DC mode)  | UINT32    | RO    | 0x00000000 (0 <sub>dec</sub> ) |
| 1C33:0B     | SM event missed counter | as <a href="#">0x1C32:11</a> [ <a href="#">▶ 146</a> ]  | UINT16    | RO    | 0x0000 (0 <sub>dec</sub> )     |
| 1C33:0C     | Cycle exceeded counter  | as <a href="#">0x1C32:12</a> [ <a href="#">▶ 146</a> ]  | UINT16    | RO    | 0x0000 (0 <sub>dec</sub> )     |
| 1C33:0D     | Shift too short counter | as <a href="#">0x1C32:13</a> [ <a href="#">▶ 146</a> ]  | UINT16    | RO    | 0x0000 (0 <sub>dec</sub> )     |
| 1C33:20     | Sync error              | as <a href="#">0x1C32:32</a> [ <a href="#">▶ 146</a> ]  | BOOLEAN   | RO    | 0x00 (0 <sub>dec</sub> )       |

## **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.

**NOTE**

**Risk of damage to the device!**

Pay attention to the instructions for firmware updates on the [separate page \[▶ 151\]](#). 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!

| EL6690        |               |  |              |
|---------------|---------------|--|--------------|
| Hardware (HW) | Firmware (FW) | Revision no.   | Release date |
| 05            | 01            | Prim.:<br>EL6690-0001-0016<br>Sec.: EL6690-0002-0016 | 01/2008      |
|               | 02            |  | 03/2008      |
|               | 03            |  | 07/2008      |
|               | 04            |  | 09/2008      |
| 06 - 11*      | 05            | Prim.:<br>EL6690-0001-0016<br>Sec.: EL6690-0002-0018 | 09/2009      |
|               | 06*           |  | 09/2009      |
|               |               | Prim.:<br>EL6690-0001-0017<br>Sec.: EL6690-0002-0019 | 10/2012      |

| <b>EL6692</b>        |                      |  |                     |
|----------------------|----------------------|--|---------------------|
| <b>Hardware (HW)</b> | <b>Firmware (FW)</b> | <b>Revision no.</b>                                  | <b>Release date</b> |
| 00                   | 01                   |  | 01/2008             |
| 00 - 01              | 02                   |  | 07/2008             |
| 01                   | 03                   | Prim.:<br>EL6692-0000-0016<br>Sec.: EL6692-0002-0017 | 02/2009             |
|                      | 04                   |  | 07/2009             |
| 02 - 19*             | 05                   | Prim.:<br>EL6692-0000-0016<br>Sec.: EL6692-0002-0018 | 09/2009             |
|                      | 06                   |  | 11/2009             |
|                      | 07                   |  | 05/2010             |
|                      | 08                   | Prim.:<br>EL6692-0000-0017                           | 10/2010             |
|                      | 09                   |  | 05/2012             |
|                      |                      | Prim.:<br>EL6692-0000-0017<br>Sec.: EL6692-0002-0018 | 10/2012             |
|                      | 10                   | Prim.:<br>EL6692-0000-0018<br>Sec.: EL6692-0002-0020 | 08/2013             |
|                      | 11*                  |  | 01/2014             |

\*) 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/EPxxxx

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

### NOTE

#### 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 <https://www.beckhoff.com/en-us/>.

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:

- Depending on functionality and performance EtherCAT slaves have one or several local controllers for processing I/O data. The corresponding program is the so-called **firmware** in \*.efw format.
- In some EtherCAT slaves the EtherCAT communication may also be integrated in these controllers. In this case the controller is usually a so-called **FPGA** chip with \*.rbf firmware.
- In addition, each EtherCAT slave has a memory chip, a so-called **ESI-EEPROM**, for storing its own device description (ESI: EtherCAT Slave Information). On power-up this description is loaded and the EtherCAT communication is set up accordingly. The device description is available from the download area of the Beckhoff website at (<https://www.beckhoff.com>). All ESI files are accessible there as zip files.

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

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

### Simplified update by bundle firmware

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

- for the firmware to be in a packed format: recognizable by the file name, which also contains the revision number, e.g. ELxxx-xxx\_REV0016\_SW01.efw
- for password=1 to be entered in the download dialog. If password=0 (default setting) only the firmware update is carried out, without an ESI update.
- for the device to support this function. The function usually cannot be retrofitted; it is a component of many new developments from year of manufacture 2016.

Following the update, its success should be verified

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

**NOTE****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****NOTE****Attention regarding update of the ESI description/EEPROM**

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

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

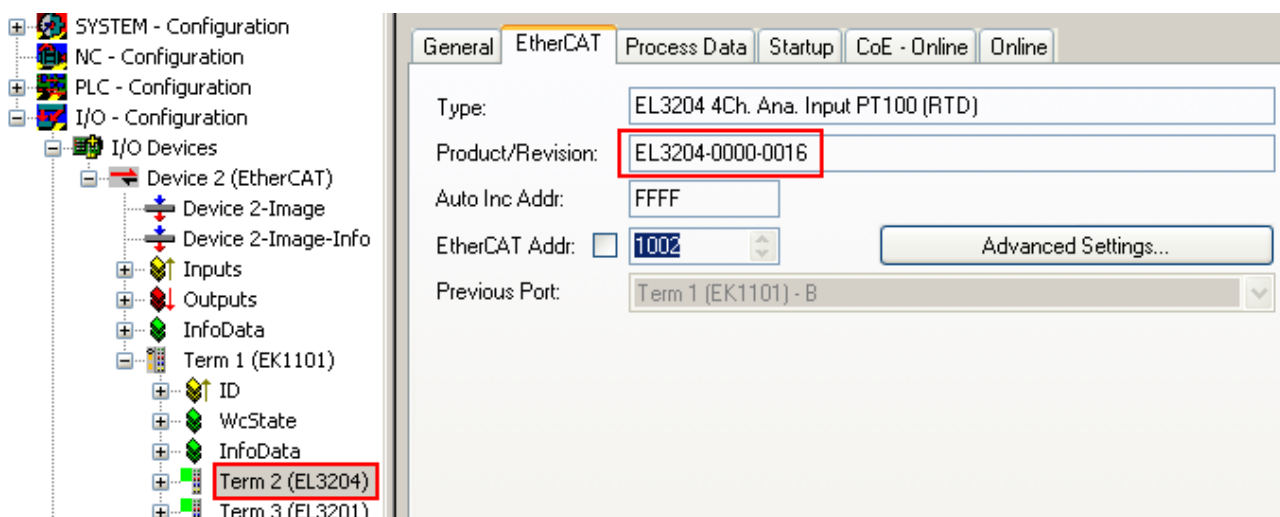


Fig. 171: Device identifier consisting of name EL3204-0000 and revision -0016

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

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

### ● Update of XML/ESI description

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

**Display of ESI slave identifier**

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



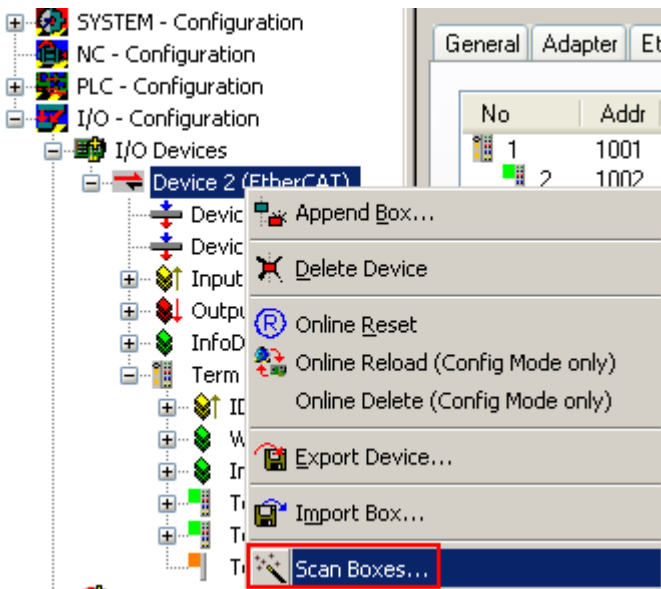


Fig. 172: Scan the subordinate field by right-clicking on the EtherCAT device

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



Fig. 173: Configuration is identical

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

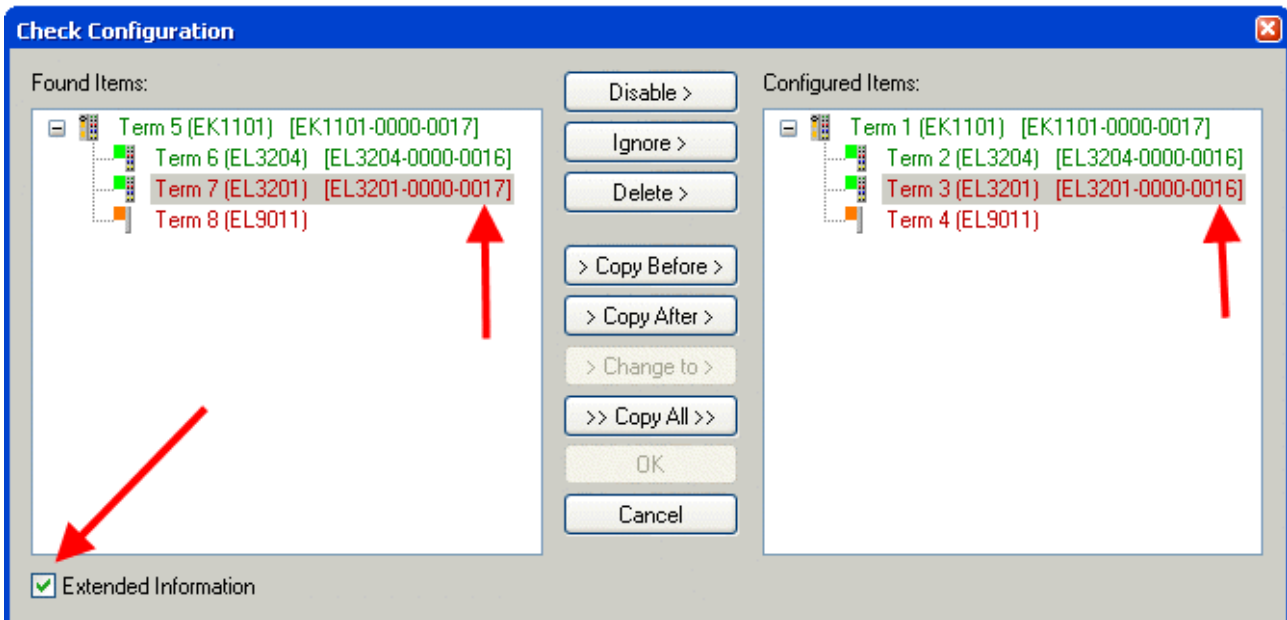


Fig. 174: Change dialog

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*

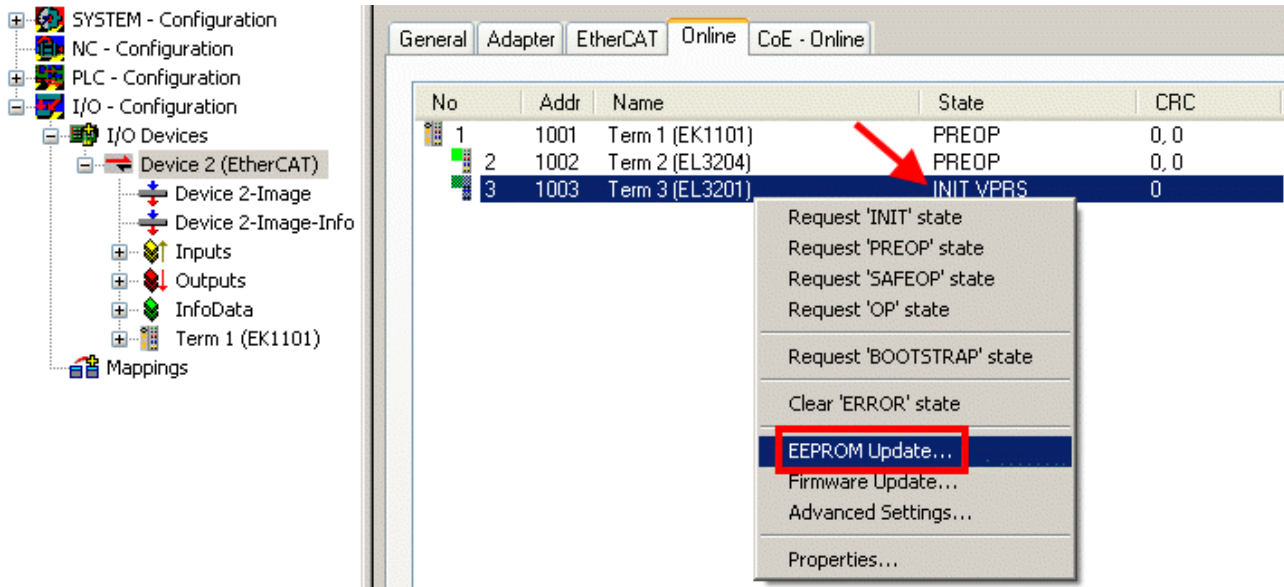


Fig. 175: EEPROM Update

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.

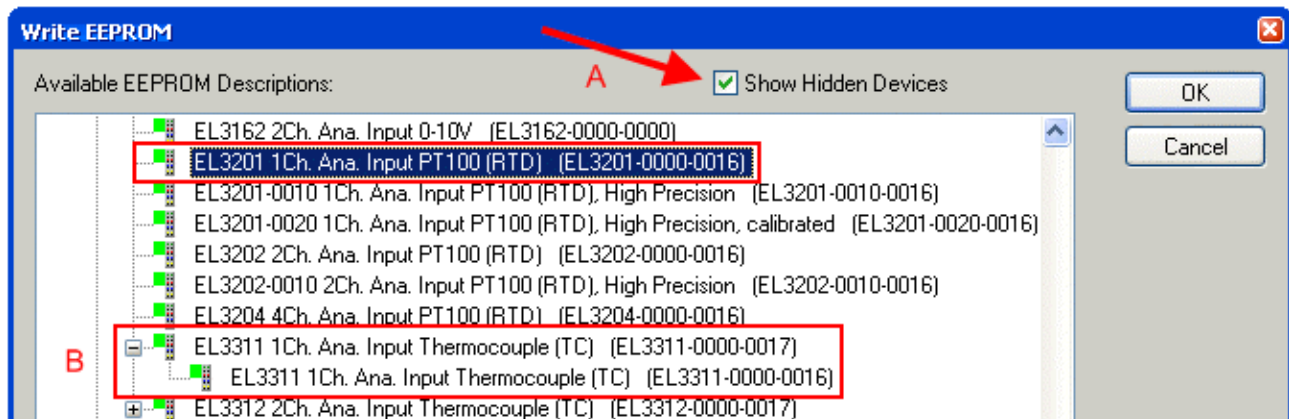


Fig. 176: Selecting the new ESI

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

### **i** The change only takes effect after a restart.

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

### 6.3.2 Firmware explanation

#### Determining the firmware version

##### Determining the version via the System Manager

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

**i** **CoE Online and Offline CoE**

Two CoE directories are available:

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

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

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

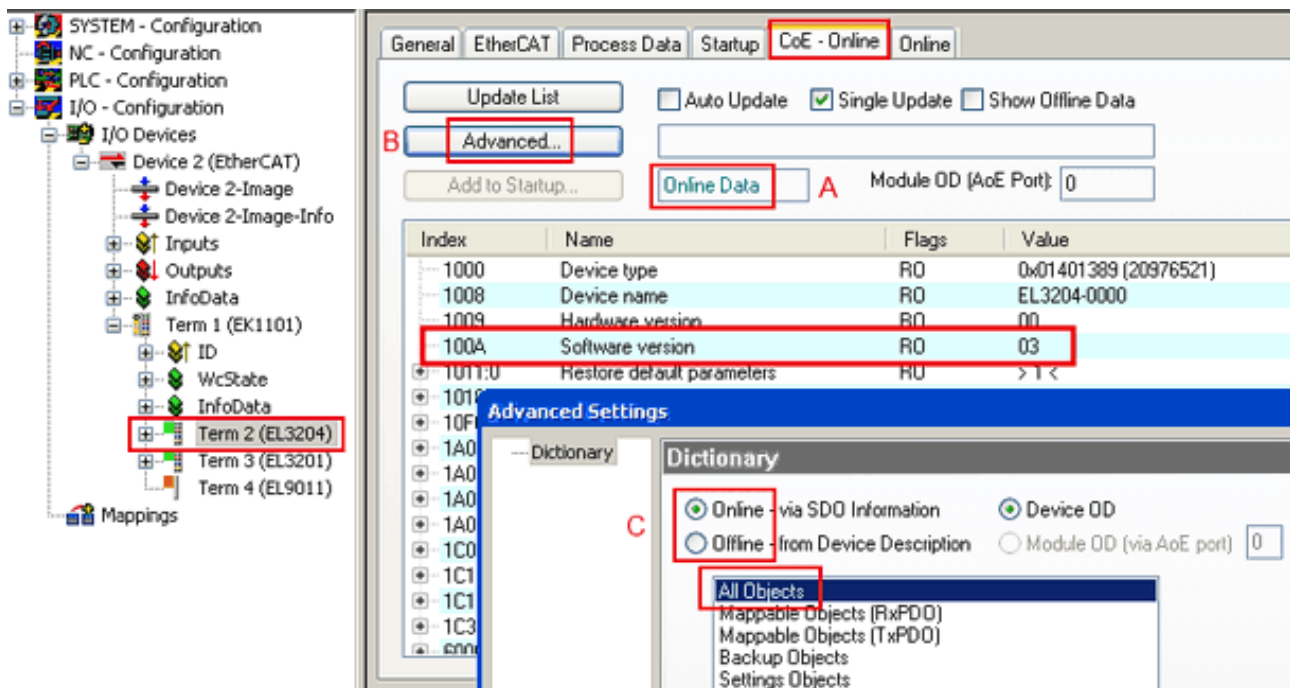


Fig. 177: Display of EL3204 firmware version

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

**i** **CoE directory**

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

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

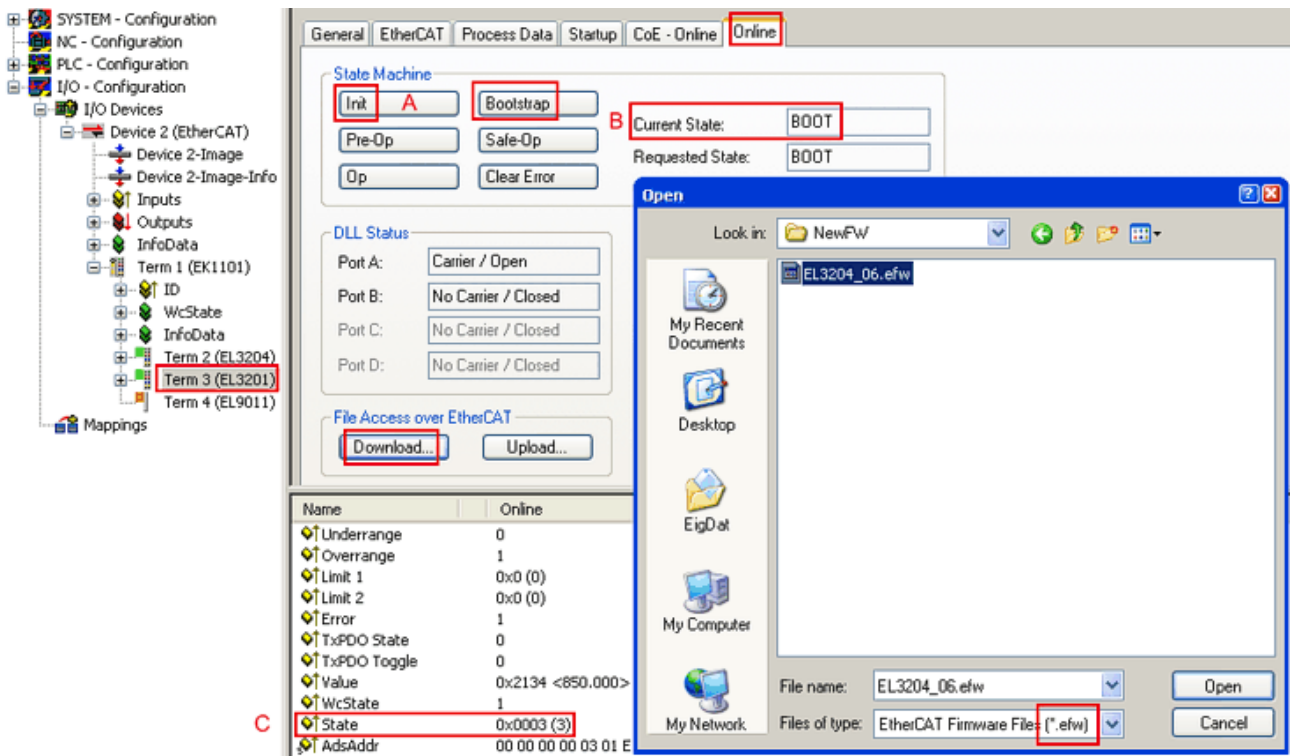
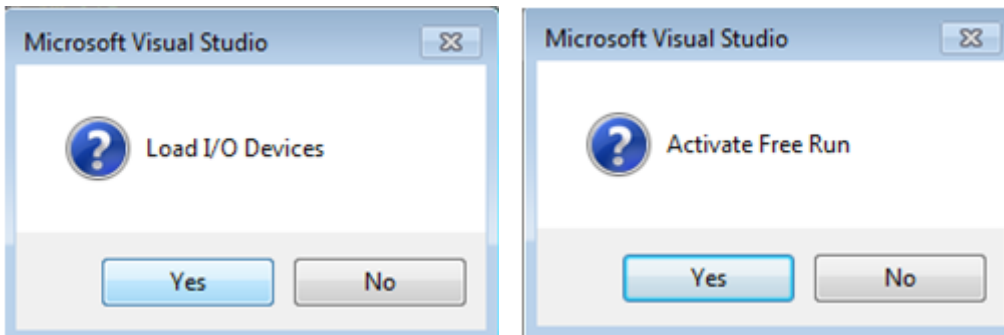


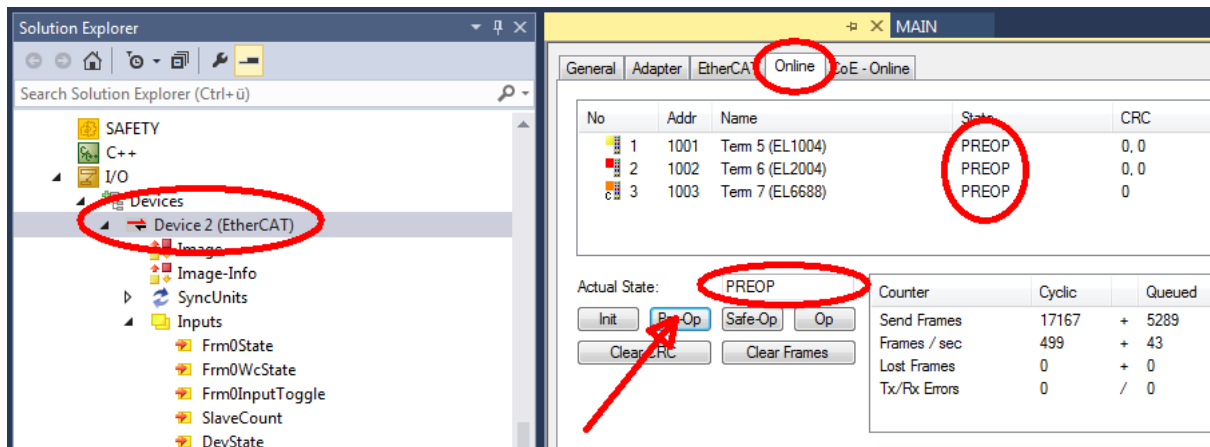
Fig. 178: Firmware Update

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  $\geq 1$  ms (default in ConfigMode is 4 ms). A FW-Update during real time operation is not recommended.

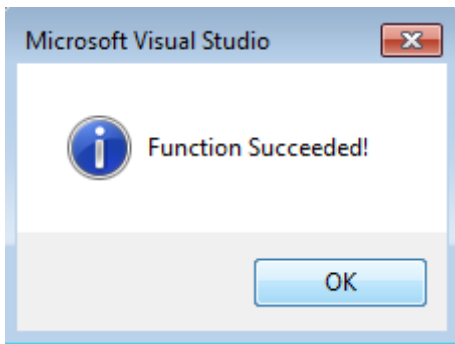


- Switch EtherCAT Master to PreOP



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

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



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

### 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 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.

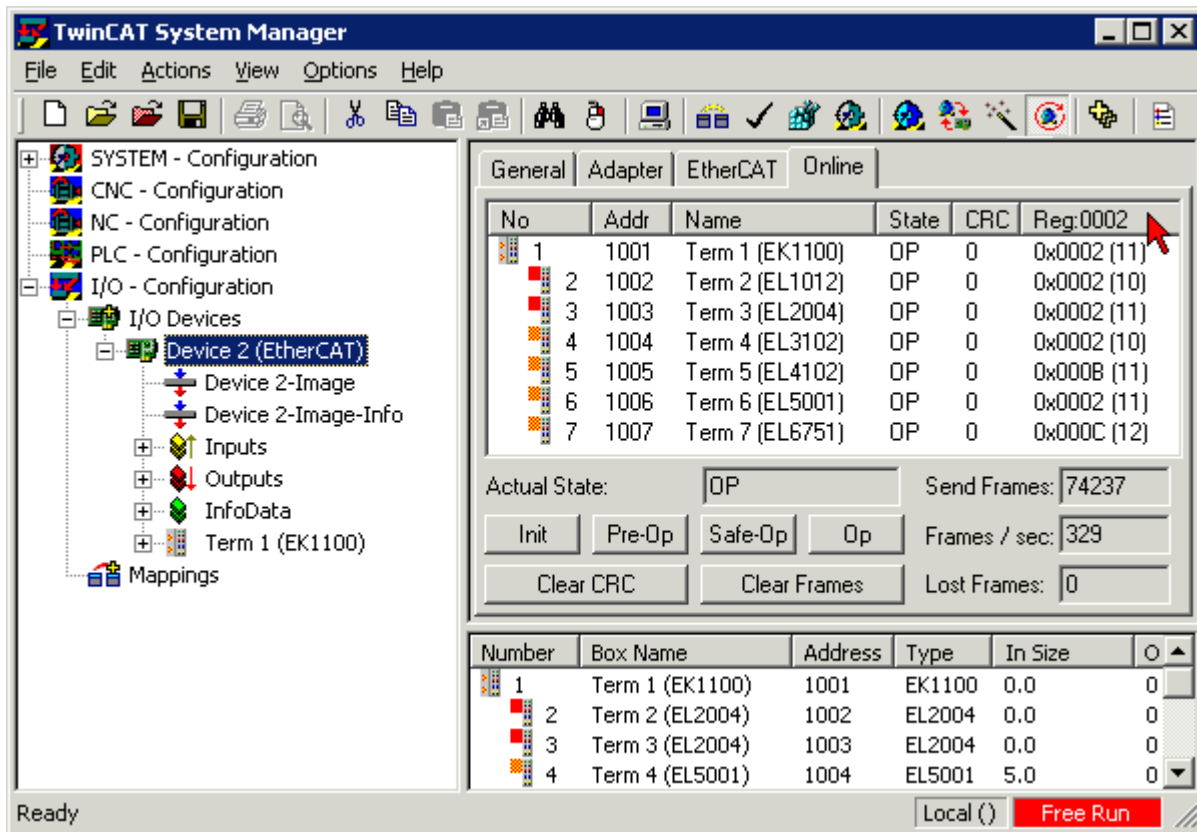
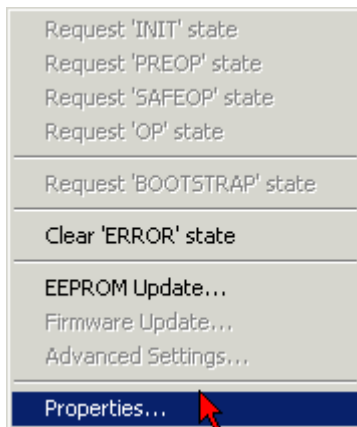


Fig. 179: FPGA firmware version definition

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

Fig. 180: 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.

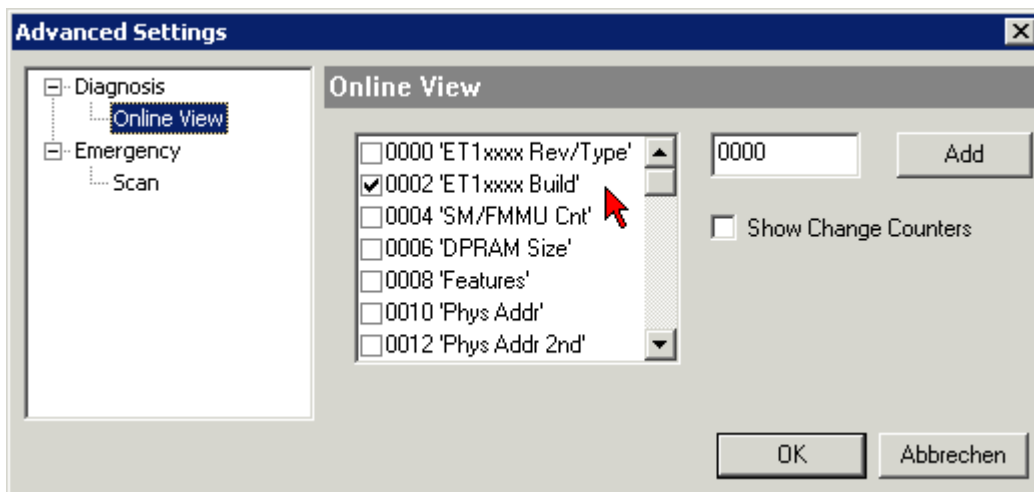


Fig. 181: Dialog *Advanced Settings*

### 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.

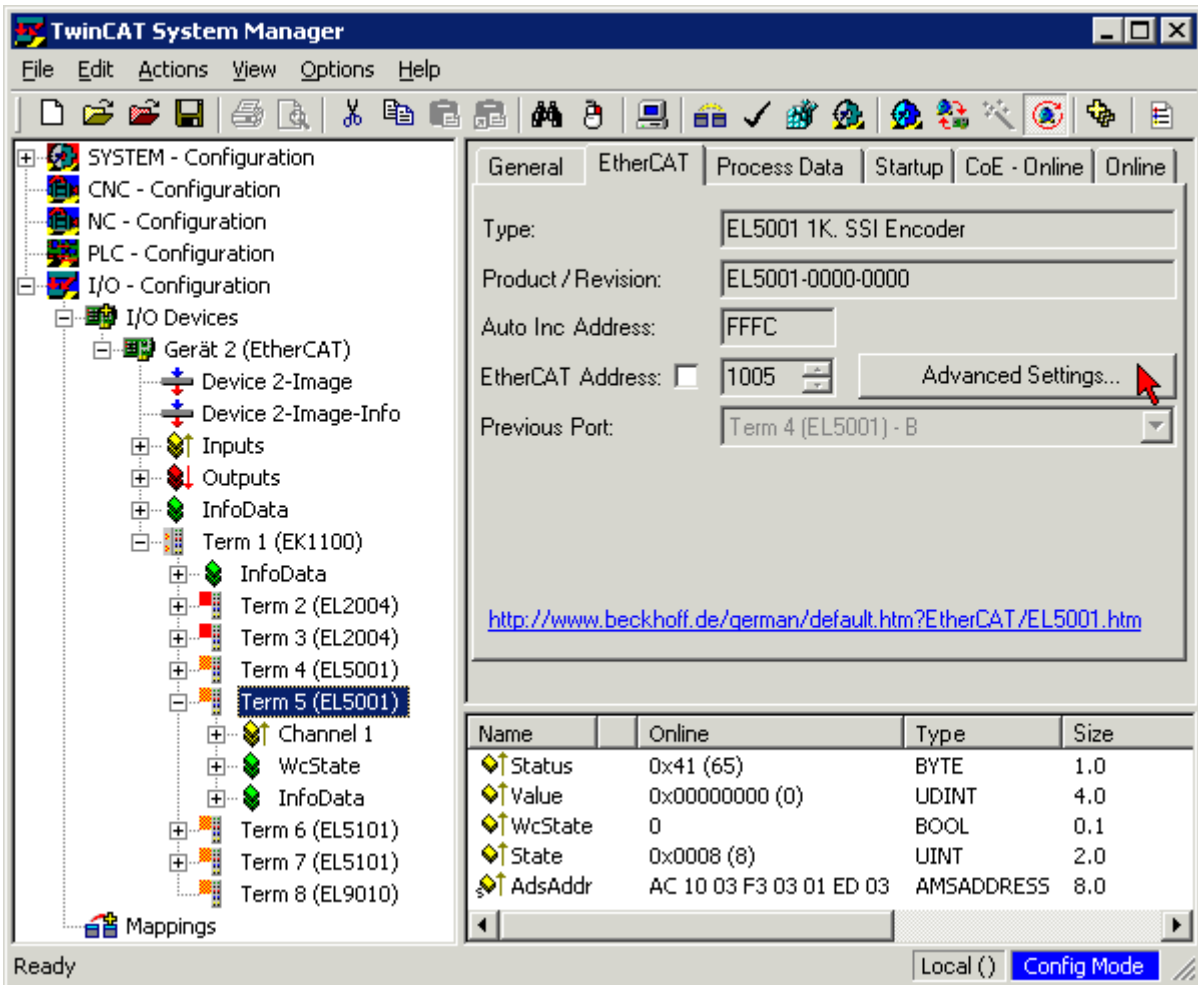
Older firmware versions can only be updated by the manufacturer!

### Updating an EtherCAT device

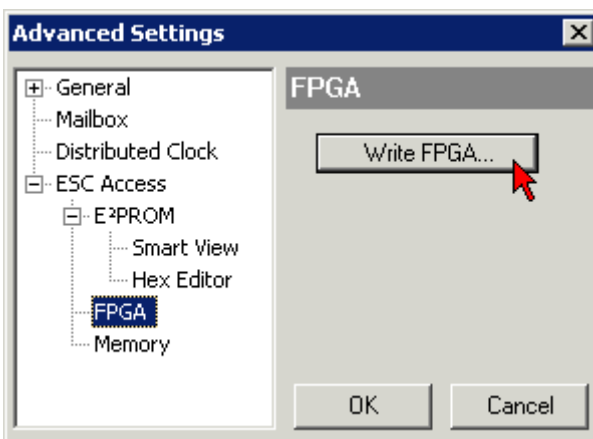
The following sequence order have to be met if no other specifications are given (e.g. by the Beckhoff support):

- Switch TwinCAT system to ConfigMode/FreeRun with cycle time  $\geq 1$  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:

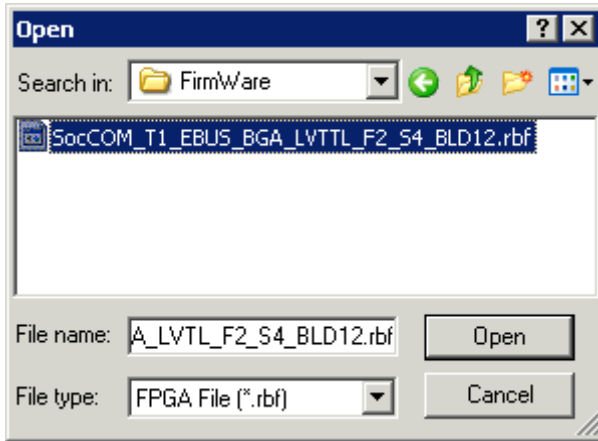


- The *Advanced Settings* dialog appears. Under *ESC Access/E<sup>2</sup>PROM/FPGA* click on *Write FPGA* button:





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



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

**NOTE**

**Risk of damage to the device!**

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

### 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.

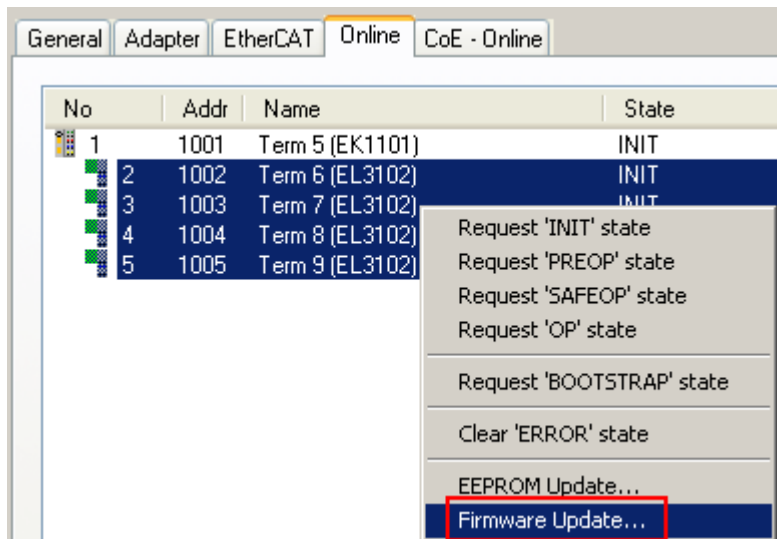


Fig. 182: Multiple selection and firmware update

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

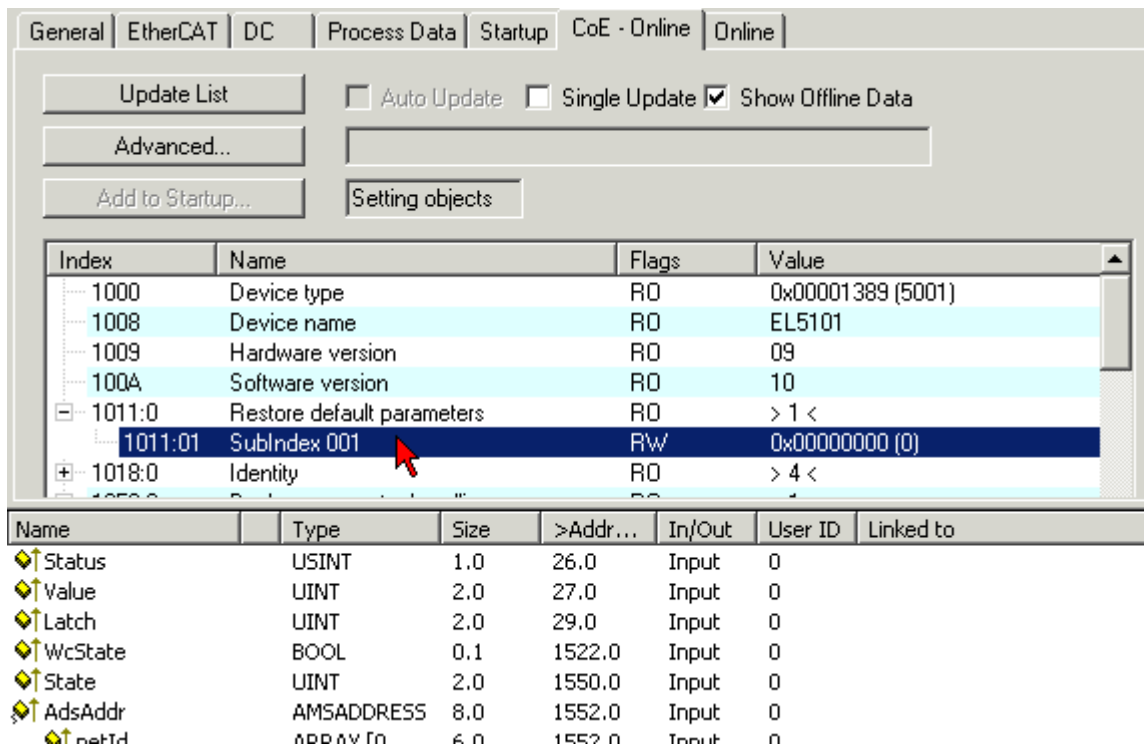


Fig. 183: Selecting the *Restore default parameters* PDO

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

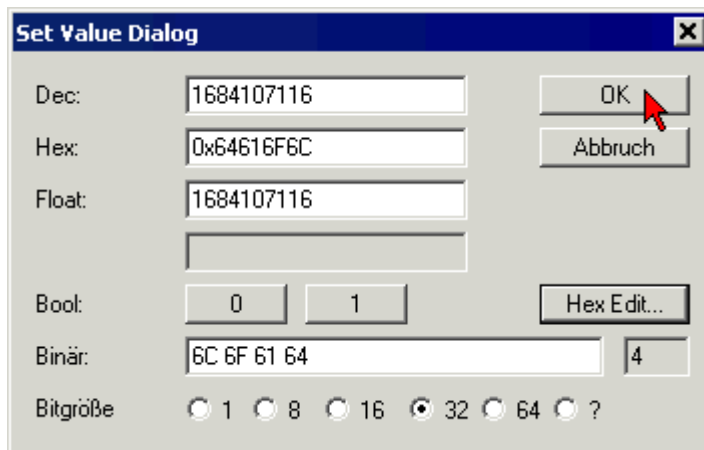


Fig. 184: Entering a restore value in the Set Value dialog

### ● Alternative restore value

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

## 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: <https://www.beckhoff.com>

You will also find further documentation for Beckhoff components there.

### Beckhoff Support

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

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

Hotline: +49 5246 963 157  
Fax: +49 5246 963 9157  
e-mail: [support@beckhoff.com](mailto:support@beckhoff.com)

### Beckhoff Service

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

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

Hotline: +49 5246 963 460  
Fax: +49 5246 963 479  
e-mail: [service@beckhoff.com](mailto:service@beckhoff.com)

### Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20  
33415 Verl  
Germany

Phone: +49 5246 963 0  
Fax: +49 5246 963 198  
e-mail: [info@beckhoff.com](mailto:info@beckhoff.com)  
web: <https://www.beckhoff.com>



More Information:  
[www.beckhoff.com/EL6692](http://www.beckhoff.com/EL6692)

Beckhoff Automation GmbH & Co. KG  
Hülshorstweg 20  
33415 Verl  
Germany  
Phone: +49 5246 9630  
[info@beckhoff.com](mailto:info@beckhoff.com)  
[www.beckhoff.com](http://www.beckhoff.com)

