BECKHOFF New Automation Technology

Documentation | EN

# EL6731

## Master/Slave terminal for PROFIBUS

2020-10-21 | Version: 2.9

# Table of contents

**1 Foreword ..... 5**
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 Beckhoff Identification Code (BIC) ..... 12

**2 Product overview ..... 14**
2.1 Introduction ..... 14
2.2 Technical data ..... 15

**3 Mounting and wiring ..... 16**
3.1 Instructions for ESD protection ..... 16
3.2 Recommended mounting rails ..... 16
3.3 Installation positions ..... 17
3.4 Mounting and demounting - terminals with traction lever unlocking ..... 18
3.5 Mounting and demounting - terminals with front unlocking ..... 20
3.6 Positioning of passive Terminals ..... 22
3.7 PROFIBUS cabling ..... 22
3.8 PROFIBUS Connection ..... 24
3.9 ATEX - Special conditions (extended temperature range) ..... 27
3.10 Continuative documentation about explosion protection ..... 28
3.11 UL notice ..... 28

**4 Basics communication ..... 29**
4.1 EtherCAT basics ..... 29
4.2 General notes for setting the watchdog ..... 29
4.3 EtherCAT State Machine ..... 31
4.4 CoE Interface ..... 33

**5 Parameterization and commissioning ..... 38**
5.1 EL6731 - PROFIBUS master terminal ..... 38
5.1.1 PROFIBUS protocols ..... 38
5.1.2 PROFIBUS DP ..... 39
5.1.3 PROFIBUS MC ..... 41
5.1.4 Synchronization ..... 42
5.1.5 ADS (acyclic services) ..... 43
5.1.6 TwinCAT (2.1x) System Manager ..... 54
5.1.7 Operation of PROFIdrive MC drives ..... 102
5.1.8 Diagnostics and error description ..... 115
5.1.9 Slave Prioritization/Multiple DP Cycles ..... 130
5.2 EL6731-0010 - PROFIBUS slave terminal ..... 131

**6 EtherCAT communication EL6731-00x0 ..... 135**
6.1 PROFIBUS master ..... 135
6.1.1 State machine ..... 135
6.1.2 Synchronization ..... 139
6.1.3 Object description and parameterization ..... 143

6.2 PROFIBUS slave ... 161
6.2.1 State machine ... 161
6.2.2 Object description and parameterization ... 165

**7 Appendix ... 173**
7.1 EtherCAT AL Status Codes ... 173
7.2 Firmware compatibility ... 173
7.3 Firmware Update EL/ES/EM/ELM/EPxxxx ... 174
7.3.1 Device description ESI file/XML ... 175
7.3.2 Firmware explanation ... 178
7.3.3 Updating controller firmware *.efw ... 179
7.3.4 FPGA firmware *.rbf ... 180
7.3.5 Simultaneous updating of several EtherCAT devices ... 184
7.4 Support and Service ... 185

# 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®, 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.

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 |
|---|---|
| 2.9 | - Update Technical data<br>- Update revision status<br>- Update structure |
| 2.8 | - Update structure |
| 2.7 | - Update chapter “ADS Error Codes”<br>- Update structure<br>- Update revision status |
| 2.6 | - Update chapter "Technical data" |
| 2.5 | - Update chapter “DP State of slaves”<br>- Update structure<br>- Update revision status |
| 2.4 | - Update chapter “Master diagnostic data”<br>- Update structure<br>- Update revision status |
| 2.3 | - Update chapter “Introduction”<br>- Update chapter “LED description”<br>- Update chapter “PROFIBUS Connection” |
| 2.2 | - Update chapter "PROFIBUS protocols"<br>- Update chapter "ADS interface”<br>- "S5-FDL communication" chapter removed |
| 2.1 | - Update chapter "Technical data"<br>- Update chapter "ADS interface"<br>- Addenda chapter “ESD protection”-<br>- Update revision status<br>- Update structure |
| 2.0 | - Migration |
| 1.7 | - Update chapter "Technical data"<br>- Update chapter "Firmware compatibility"<br>- Update structure |
| 1.6 | - Update chapter "Technical data"<br>- Update chapter "Object description and parameterization"<br>- Update structure |
| 1.5 | - "Operation of PROFIDrive MC drives" chapter added |
| 1.4 | - "FDL interface" chapter added |
| 1.3 | - Technical data added, object description amended |
| 1.2 | - Technical data corrected |
| 1.1 | - Technical data corrected |
| 1.0 | - Revision, technical data amended |
| 0.1 | - Preliminary version for internal use |

## 1.4 Version identification of EtherCAT devices

### Designation

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

- family key
- type
- version
- revision

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

### Notes

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

### Identification number

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

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

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

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

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

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

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

Syntax: D ww yy x y z u

D - prefix designation
ww - calendar week
yy - year
x - firmware version of the bus PCB
y - hardware version of the bus PCB
z - firmware version of the I/O PCB
u - hardware version of the I/O PCB

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

### Unique serial number/ID, ID number

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

See also the further documentation in the area

- IP67: EtherCAT Box
- Safety: TwinSafe
- Terminals with factory calibration certificate and other measuring terminals

### Examples of markings

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

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

Fig. 3: CU2016 switch with serial/ batch number

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

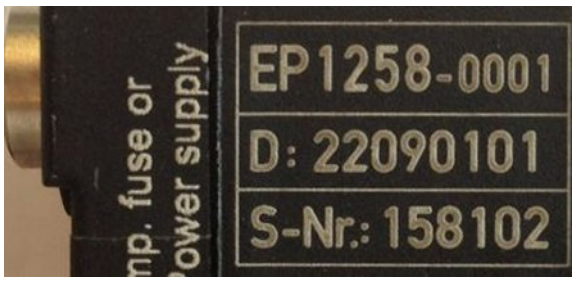



Fig. 5: EP1258-00001 IP67 EtherCAT Box with batch number/ date code 22090101 and unique serial number 158102

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

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

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

### 1.4.1 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.

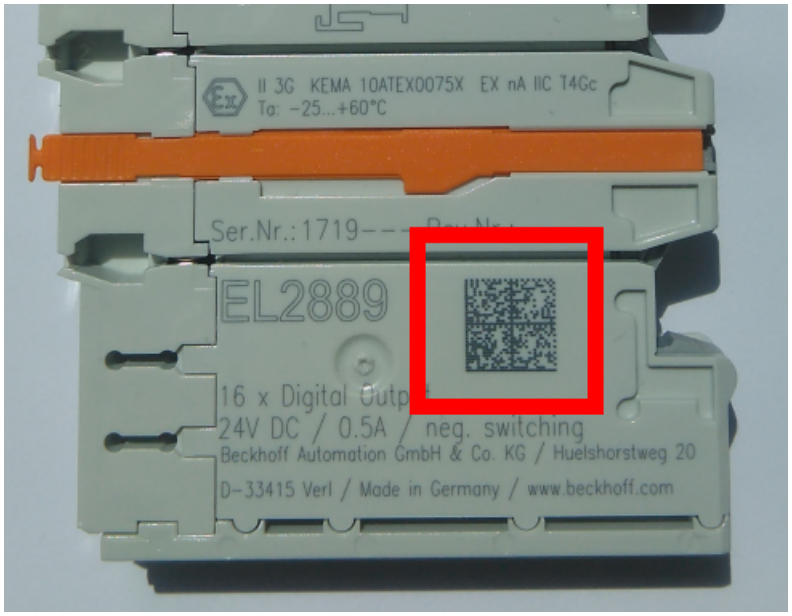



Fig. 9: BIC as data matrix code (DMC, code scheme ECC200)

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. The data under positions 1 to 4 are always available.

The following information is contained:

| Item no. | Type of information | Explanation | Data identifier | Number of digits incl. data identifier | Example |
|---|---|---|---|---|---|
| 1 | Beckhoff order number | **Beckhoff order number** | 1P | 8 | 1P072222 |
| 2 | Beckhoff Traceability Number (BTN) | **Unique serial number, see note below** | S | 12 | SBTNk4p562d7 |
| 3 | Article description | **Beckhoff article description, e.g. EL1008** | 1K | 32 | 1KEL1809 |
| 4 | Quantity | **Quantity in packaging unit, e.g. 1, 10, etc.** | Q | 6 | Q1 |
| 5 | Batch number | Optional: Year and week of production | 2P | 14 | 2P401503180016 |
| 6 | ID/serial number | Optional: Present-day serial number system, e.g. with safety products | 51S | 12 | 51S678294104 |
| 7 | Variant number | Optional: Product variant number on the basis of standard products | 30P | 32 | 30PF971, 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 item 1 to 4 and 6. The data identifiers are marked in red for better display:

### BTN

An important component of the BIC is the Beckhoff Traceability Number (BTN, item no. 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.

# 2 Product overview

## 2.1 Introduction

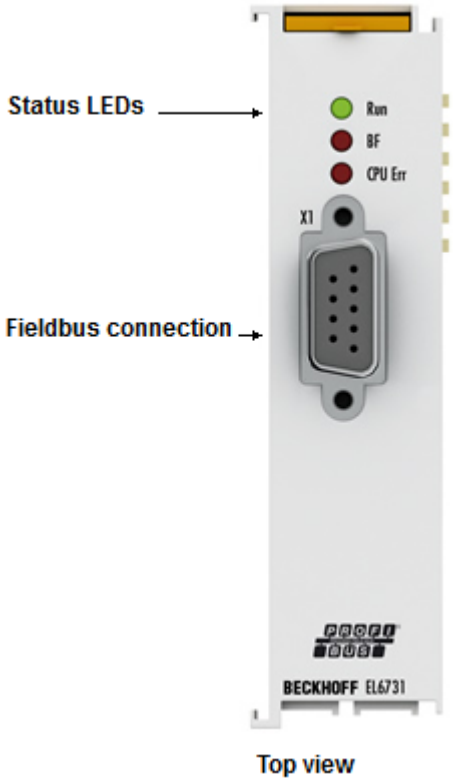



Fig. 10: EL6731

**Master and slave terminals for PROFIBUS**

The master and slave terminals for PROFIBUS correspond to the FC3101 PCI card from Beckhoff. Thanks to the connection via Ethernet, no PCI slots are required in the PC.

The EL6731 can run the PROFIBUS protocol with all features; in the EtherCAT Terminal network it enables integration of any Profibus devices. Thanks to the in-house development of the PROFIBUS chip, the terminals are equipped with the latest version of the PROFIBUS technology – including high-precision isochronous mode for axis control and extended diagnostic options. This masters are the only ones that support different poll rates for the slaves.

Features:

- Cycle times from 200 µs are possible
- PROFIBUS DP, PROFIBUS DP-V1, PROFIBUS DP-V2
- Master, slave and Profibus monitor up to 12 Mbit/s
- Powerful parameter and diagnostics interfaces
- Error management freely configurable for each bus device
- It is possible to read the bus configuration and automatically assign the "GSD" files

## 2.2 Technical data

| Technical data | EL6731-0000 | EL6731-0010 |
|---|---|---|
| Bus system | PROFIBUS DP (standard),<br>PROFIBUS DP-V1 (Cl. 1+2: acyclic services, alarms),<br>PROFIBUS DP-V2, PROFIBUS MC (equidistance) | |
| Version | Master | Slave |
| Number of fieldbus channels | 1 | |
| Data transfer rate | 9.6 kBit/s to 12 MBit/s [1)] | |
| Bus interface | 1 x D-Sub socket, 9-pin, galvanically isolated | |
| Bus nodes | maximum 125 slaves with up to 244 bytes input, output, parameter, configuration or diagnostic data per slave | |
| Cycle time | Differing DP cycle times per slave are possible using the CDL concept | |
| Process image | Sum max.: 1.4 kbyte input and 1.4 kbyte output data | |
| Diagnostics | Status LEDs | |
| Power supply | via the E-bus | |
| Current consumption via E-bus | typ. 350 mA | |
| Electrical isolation | 500 V (E-bus/Profibus) | |
| Configuration | with TwinCAT System Manager | |
| Weight | approx. 70 g | |
| Permissible ambient temperature range during operation | -25 °C ... +60 °C (extended Temperature range) (for EL6731-0010 from Hardware 17) | |
| 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 [▶ 18] | 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 | |
| Approval | CE<br>ATEX [▶ 27]<br>cULus [▶ 28] | |

[1)] 6 MBit/s are not supported by the EL6731 and EL6731-0010 from HW 29 [▶ 173] and higher.

# 3 Mounting and wiring

## 3.1 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 by 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.

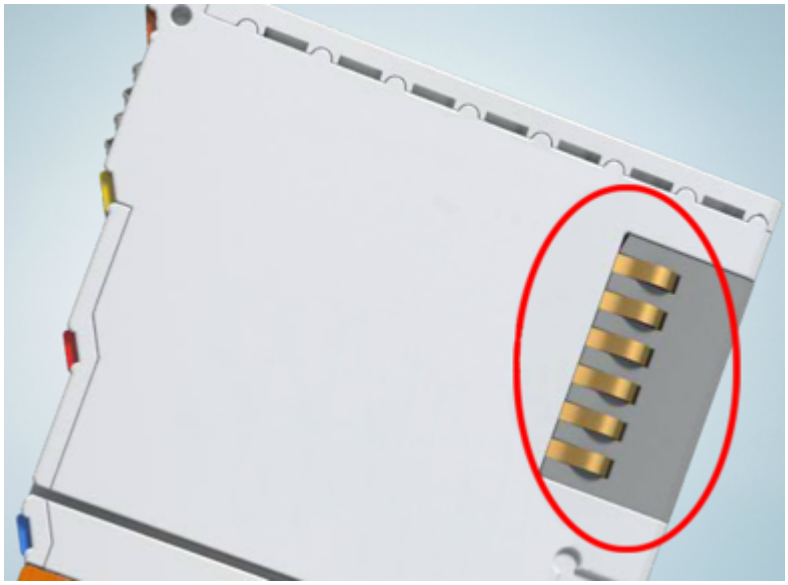


Fig. 11: Spring contacts of the Beckhoff I/O components

## 3.2 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**

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)!

## 3.3 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.

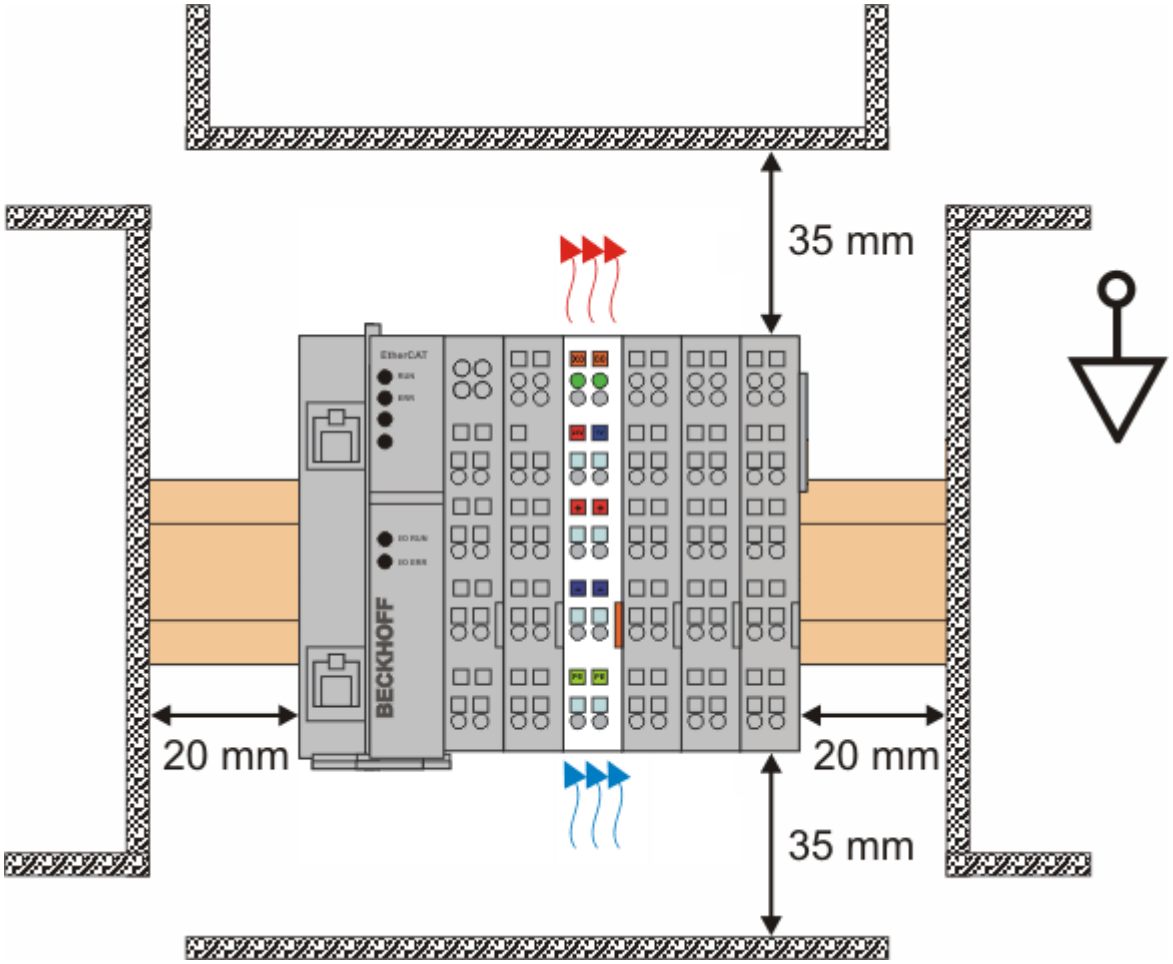



Fig. 12: Recommended distances for standard installation position

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.

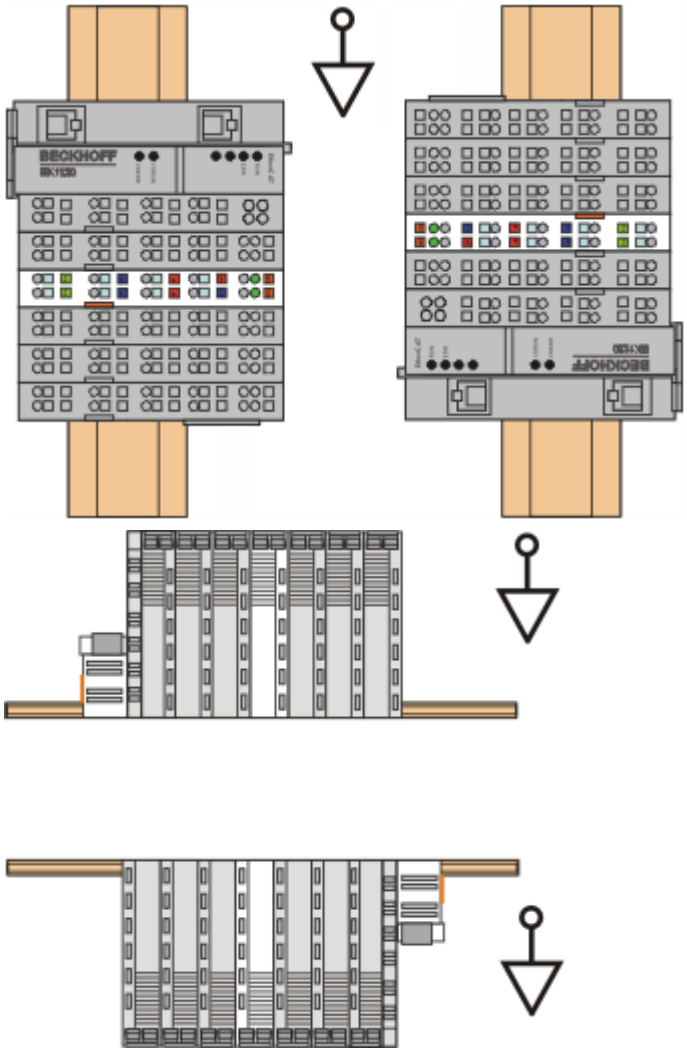


Fig. 13: Other installation positions

## 3.4 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**

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).

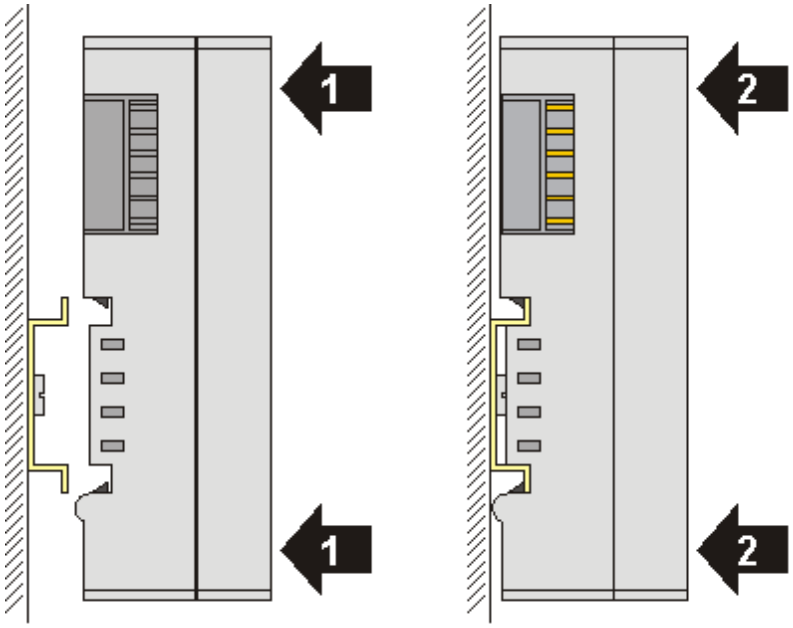
**⚠ WARNING**

**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.

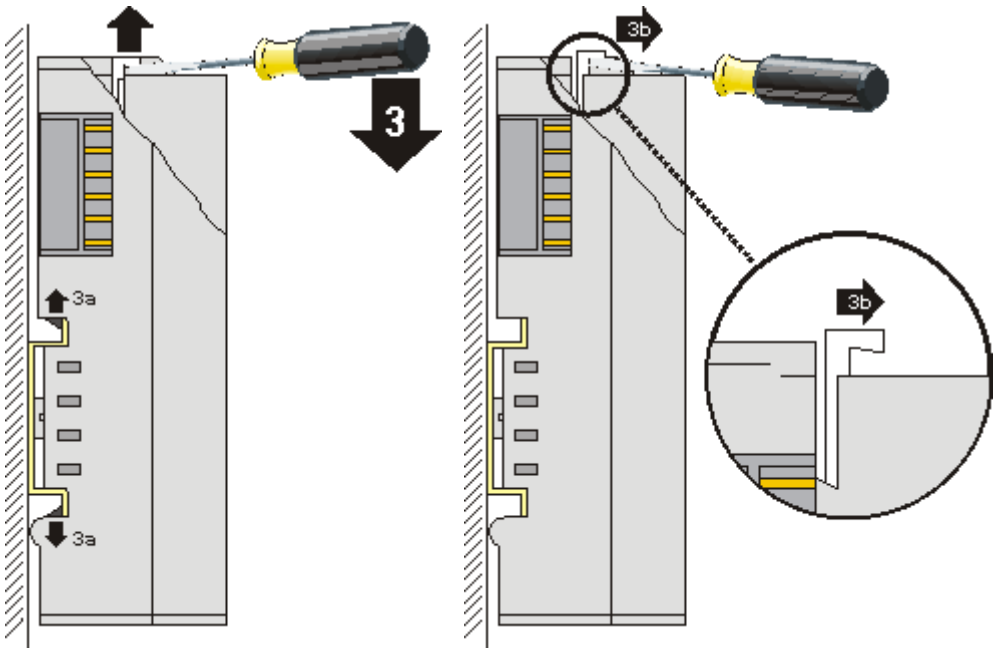



and press (1) the terminal module against the mounting rail until it latches in place on the mounting rail (2).

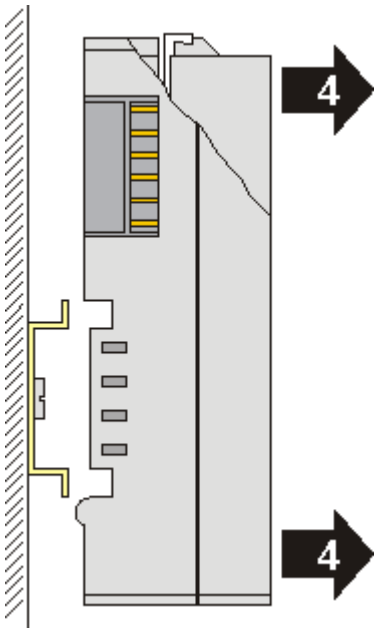
- Attach the cables.

**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.




## 3.5 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**

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).

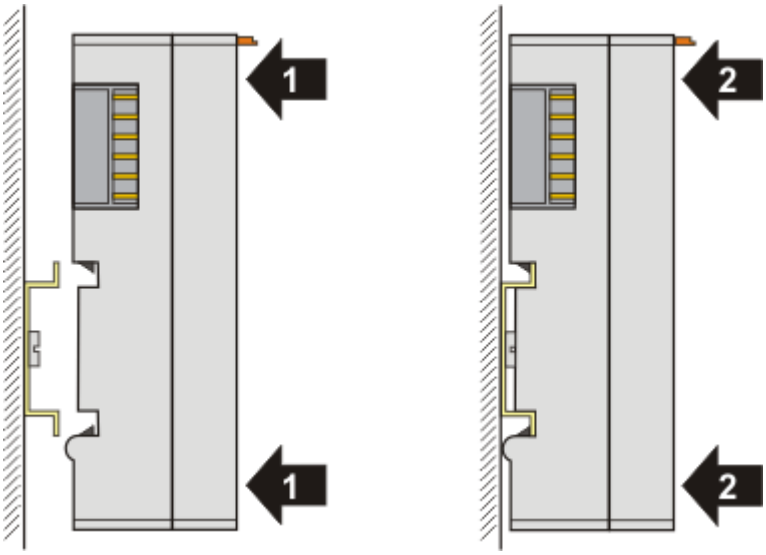
**⚠ WARNING**

**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.

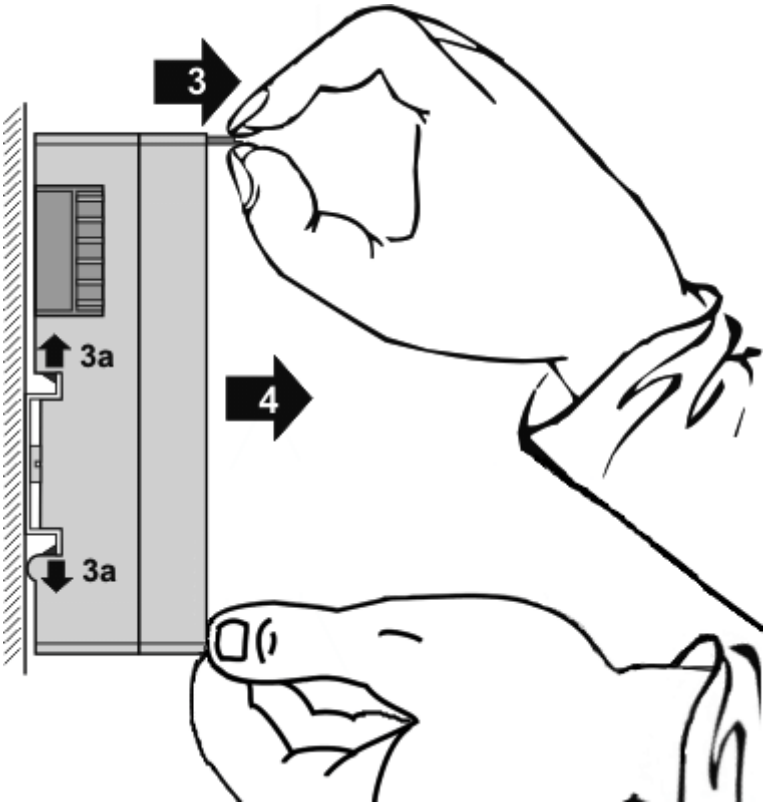



and press (1) the terminal module against the mounting rail until it latches in place on the mounting rail (2).

- 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.

## 3.6 Positioning of passive Terminals

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

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

**Examples for positioning of passive terminals (highlighted)**

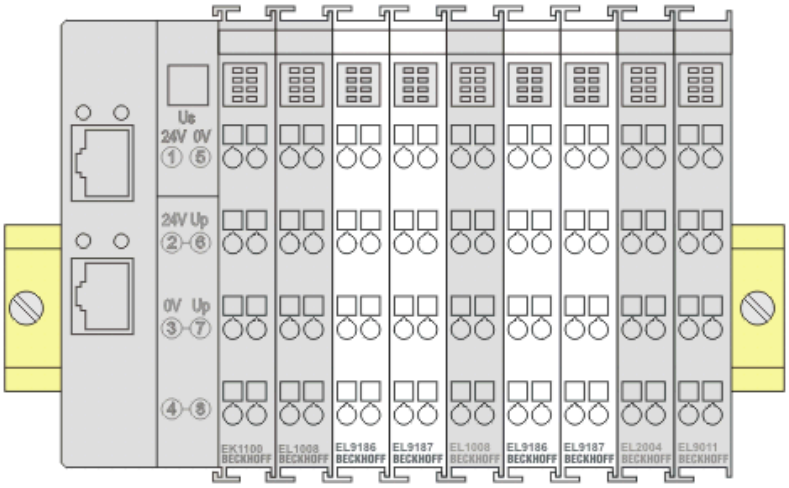



Fig. 14: Correct positioning

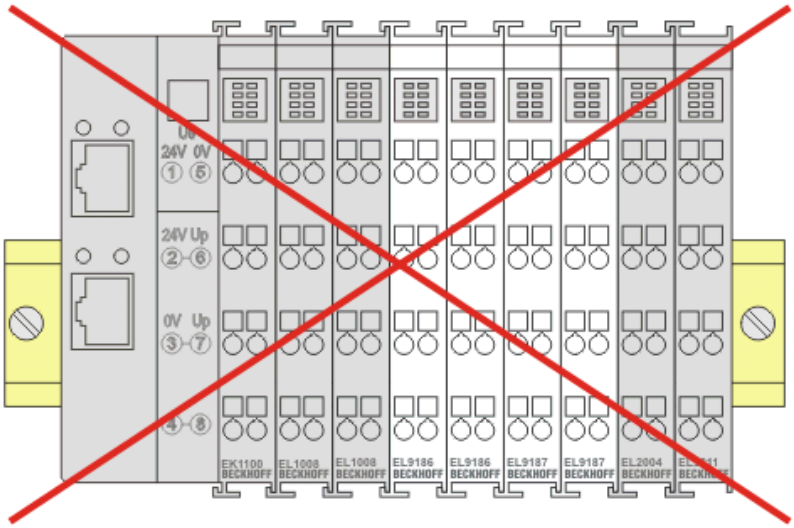



Fig. 15: Incorrect positioning

## 3.7 PROFIBUS cabling

Physical aspects of the data transmission are defined in the PROFIBUS standard (see PROFIBUS layer 1: Physical Layer).

The types of area where a fieldbus system can be used is largely determined by the choice of the transmission medium and the physical bus interface. In addition to the requirements for transmission security, the expense and work involved in acquiring and installing the bus cable is of crucial significance. The PROFIBUS standard therefore allows for a variety of implementations of the transmission technology while retaining a uniform bus protocol.

### Cable-based transmission

This version, which accords with the American EIA RS-485 standard, was specified as a basic version for applications in production engineering, building management and drive technology. A twisted copper cable with one pair of conductors is used. Depending on the intended application area (EMC aspects should be considered) the screening may be omitted.

Two types of conductor are available, with differing maximum conductor lengths (see the RS-485 table).

### RS485 - Fundamental properties

| RS-485 transmission according to the PROFIBUS standard | |
|---|---|
| Network topology | Linear bus, active bus terminator at both ends, stubs are possible. |
| Medium | Screened twisted cable, screening may be omitted, depending upon the environmental conditions (EMC). |
| Number of stations | 32 stations in each segment with no repeater. Can be extended to 127 stations with repeater |
| Max. bus length without repeater | 100 m at 12 Mbit/s<br>200 m at 1500 Kbit/s, up to 1.2 km at 93.75 kbit/s |
| Max. bus length with repeater | Line amplifiers, or repeaters, can increase the bus length up to 10 km. The number of repeaters possible is at least 3, and, depending on the manufacturer, may be up to 10. |
| Transmission speed (adjustable in steps) | 9.6 kbit/s; 19.2 kbit/s; 93.75 kbit/s; 187.5 kbit/s; 500 kbit/s; 1500 kbit/s; 12 Mbit/s |
| Connector | 9-pin D-Sub connector for IP20<br>M12 round connector for IP65/67 |

### Cabling for PROFIBUS DP and PROFIBUS FMS

Note the special requirements on the data cable for baud rates greater than 1.5 Mbaud. The correct cable is a basic requirement for correct operation of the bus system. If a simple 1.5 Mbaud cable is used, reflections and excessive attenuation can lead to some surprising phenomena. It is possible, for instance, for a connected PROFIBUS station not to achieve a connection, but for it to be included again when the neighboring station is disconnected. Or there may be transmission errors when a specific bit pattern is transmitted. The result of this can be that when the equipment is not operating, PROFIBUS works without faults, but that there are apparently random bus errors after start-up. Reducing the baud rate (< 93.75 kbaud) corrects this faulty behavior.

If reducing the baud rate does not correct the error, then in many cases this can indicate a wiring fault. The two data lines may be crossed over at one or more connectors, or the termination resistors may not be active, or they may be active at the wrong locations.

**Recommended cables**

Installation is made a great deal more straightforward if preassembled cables from Beckhoff are used! Wiring errors are avoided, and commissioning is more rapidly completed. The Beckhoff range includes fieldbus cables, power supply cables, sensor cables and accessories such as termination resistors and T-pieces. Connectors and cables for field assembly are nevertheless also available.

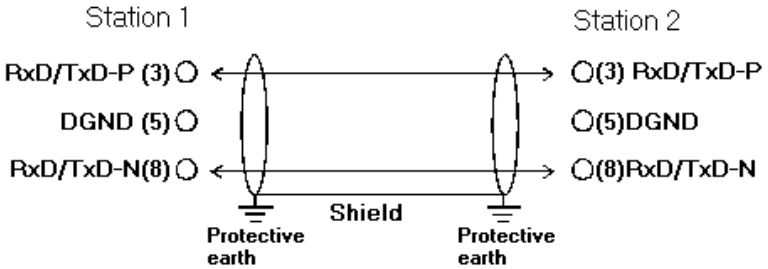



Fig. 16: PROFIBUS cable assignment

**Termination resistors**

In systems with more than two stations all devices are wired in parallel. The PROFIBUS cable must be terminated with resistances at both ends, in order to avoid reflections and associated transfer problems.

### Distances

The bus cable is specified in EN 50170. This yields the following lengths for a bus segment.

| Baud rate in kbits/sec | 9,6 | 19,2 | 93,75 | 187,5 | 500 | 1500 | 12000 |
|---|---|---|---|---|---|---|---|
| Cable length in m | 1200 | 1200 | 1200 | 1000 | 400 | 200 | 100 |

Stubs up to 1500 kbaud <6.6 m; at 12 Mbaud stub segments should not be used.

### Bus segments

A bus segment consists of at most 32 devices. 126 devices are permitted in a PROFIBUS network. Repeaters are required to refresh the signal in order to achieve this number. Each repeater is counted as one device.

IP-Link is the subsidiary bus system for Fieldbus Boxes, whose topology is a ring structure. There is an IP master in the coupler modules (IP230x-Bxxx or IP230x-Cxxx) to which up to 120 extension modules (IExxxx) may be connected. The distance between two modules may not exceed 5 m. When planning and installing the modules, remember that because of the ring structure the IP-Link master must be connected again to the last module.

### Installation guidelines

When assembling the modules and laying the cables, observe the technical guidelines provided by the PROFIBUS User Organization (PROFIBUS Nutzerorganisation e.V.) for PROFIBUS DP/FMS (see www.profibus.de).

### Checking the PROFIBUS wiring

A PROFIBUS cable (or a cable segment when using repeaters) can be checked with a few simple resistance measurements. The cable should meanwhile be removed from all stations:

1. Resistance between A and B at the start of the lead: approx. 110 Ohm
2. Resistance between A and B at the end of the lead: approx. 110 Ohm
3. Resistance between A at the start and A at the end of the lead: approx. 0 Ohm
4. Resistance between B at the start and B at the end of the lead: approx. 0 Ohm
5. Resistance between screen at the start and screen at the end of the lead: approx. 0 Ohm

If these measurements are successful, the cable is okay. If, in spite of this, bus malfunctions still occur, this is usually a result of EMC interference. Observe the installation notes from the PROFIBUS User Organization (www.profibus.com).

## 3.8 PROFIBUS Connection

### M12 round connector

The M12 socket is inverse-coded and has five pins. Pin 1 transfers 5 $V_{DC,}$ pin 3 transfers GND for the active termination resistor. These must never be misused for other functions, as this can lead to destruction of the device.

Pins 2 and 4 transfer the PROFIBUS signals. These must never be swapped over, as this will prevent communication. Pin 5 transfers the shield, which is capacitively connected to the base of the Fieldbus Box.

### Pin assignment M12 socket (-B310)

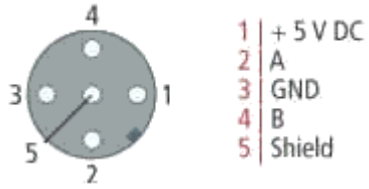



Fig. 17: Pin assignment M12 socket (-B310)

### Pin assignment M12 socket/plug connector (-B318)

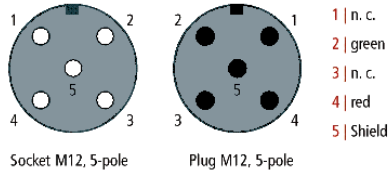



Fig. 18: Pin assignment M12 socket/plug connector (-B318)

### Nine-pin D-Sub

Pin 6 transfers 5 $V_{DC,}$ pin 5 transfers GND for the active termination resistor. These must never be misused for other functions, as this can lead to destruction of the device.

Pins 3 and 8 transfer the PROFIBUS signals. These must never be swapped over, as this will prevent communication.

### Pin assignment of the PROFIBUS D-sub socket

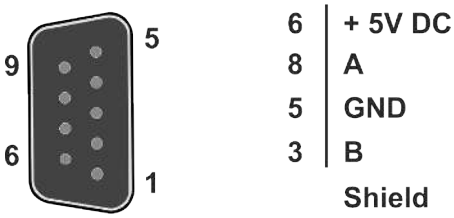



Fig. 19: Pin assignment of the PROFIBUS D-sub socket

### PROFIBUS wire colors

| PROFIBUS line | M12 | D-Sub |
|---|---|---|
| B red | Pin 4 | Pin 3 |
| A green | Pin 2 | Pin 8 |

### Connection of the Fieldbus Box modules

The Fieldbus Box modules are connected either directly or via a T-piece (or Y-piece).

The B318 series features a socket and a plug connector, i.e. this is where the PROFIBUS is routed in the module. The supply voltage (+5 $V_{DC}$) for the termination resistor is only present at the socket. The termination resistor ZS1000-1610 is only available as a plug connector.

The incoming PROFIBUS line should always end with a socket.

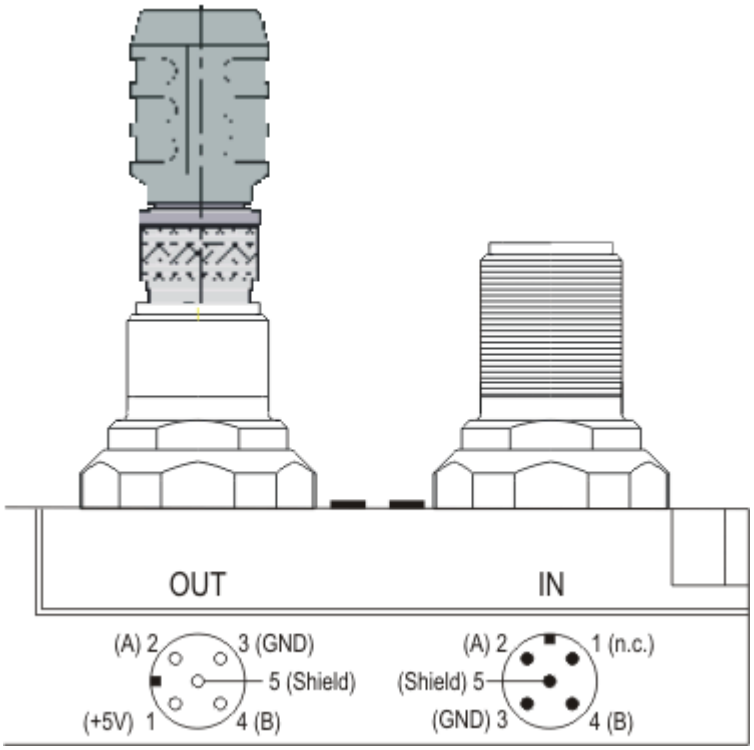



Fig. 20: Pin assignment socket/plug connector Fieldbus Box modules

Two T-pieces are available:

- ZS1031-2600 with +5 $V_{DC}$ transfer for supplying the termination resistor
- ZS1031-2610 without +5 $V_{DC}$ transfer

# 3.9 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 60079-0 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 IIC 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 nC IIC T4 Gc Ta: -25 … +60°C**

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

## 3.10 Continuative documentation about explosion protection

**Explosion protection for terminal systems**

Pay also attention to the continuative documentation

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

that is available for download on the Beckhoff homepage https:\\www.beckhoff.com!

## 3.11 UL notice

| | |
|---|---|
| c UL us | **Application**<br>Beckhoff EtherCAT modules are intended for use with Beckhoff's UL Listed EtherCAT System only. |
| c UL us | **Examination**<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). |
| c UL us | **For devices with Ethernet connectors**<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:

c UL us

# 4 Basics communication

## 4.1 EtherCAT basics

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

## 4.2 General notes for setting the watchdog

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

The EtherCAT slave controller (ESC) in the EL2xxx terminals features 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.

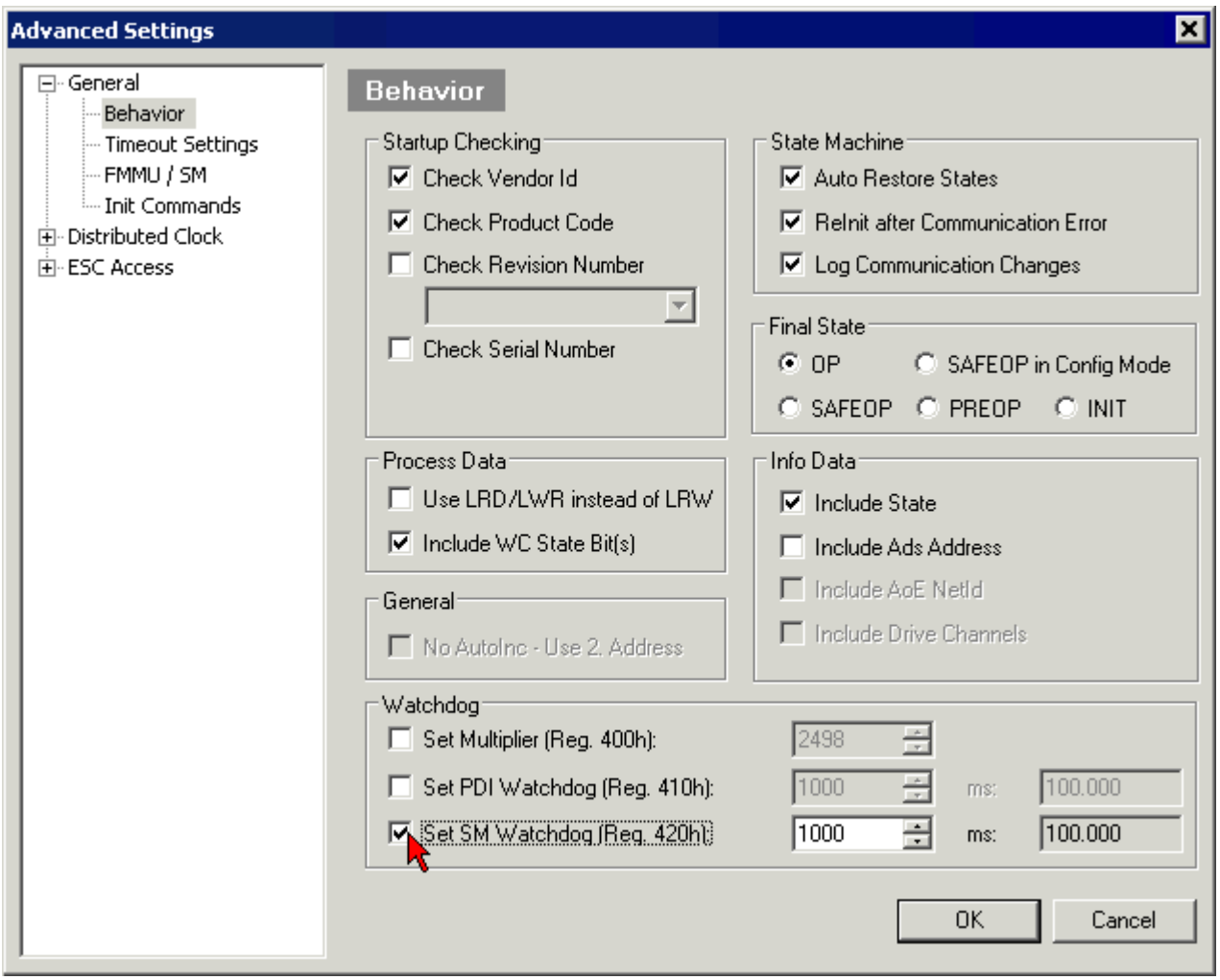



Fig. 21: EtherCAT tab -> Advanced Settings -> Behavior -> Watchdog

Notes:

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

**Multiplier**

Multiplier

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

1/25 MHz * (watchdog multiplier + 2) = 100 µs (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.

## 4.3 EtherCAT State Machine

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

A distinction is made between the following states:

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

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

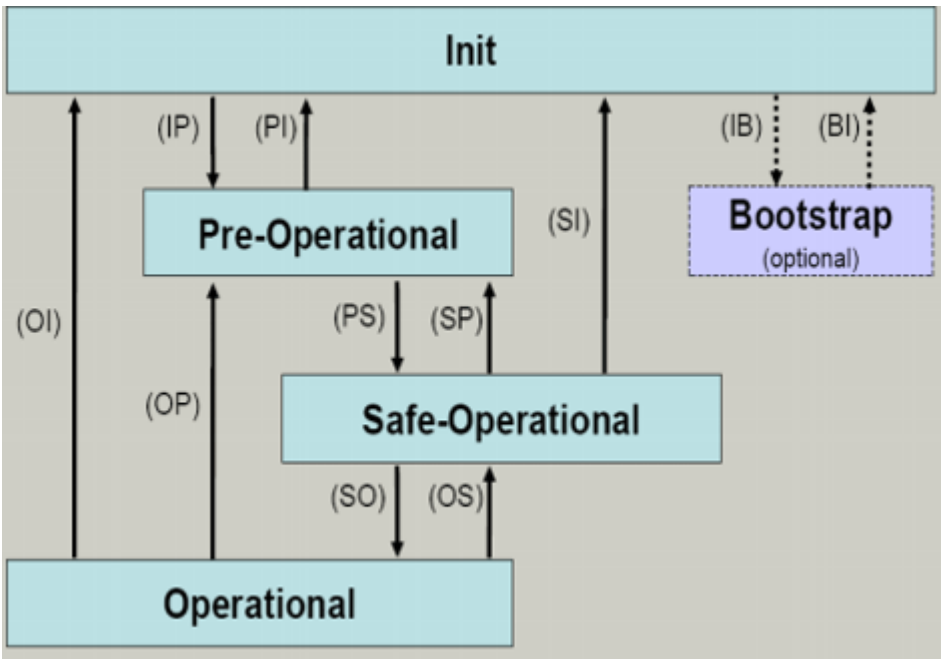



Fig. 22: States of the EtherCAT State Machine

### 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**

The default set watchdog [▶ 29] 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.

## 4.4 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 ($0...65535_{dez}$)
- SubIndex: 0x00…0xFF ($0...255_{dez}$)

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:

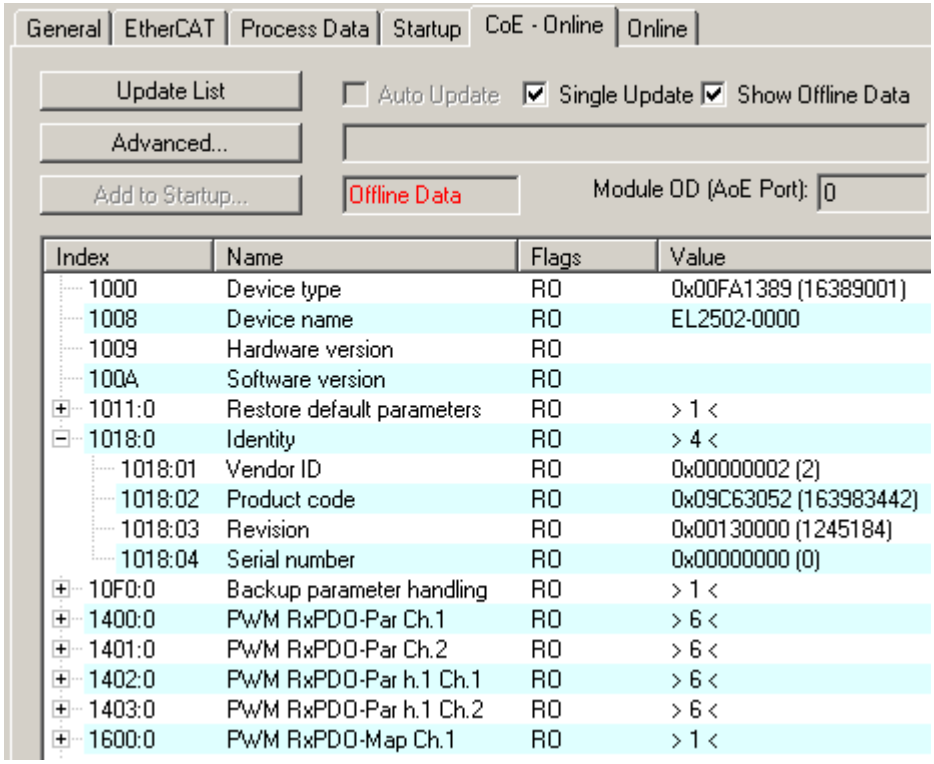



Fig. 23: “CoE Online” tab

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**

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

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.

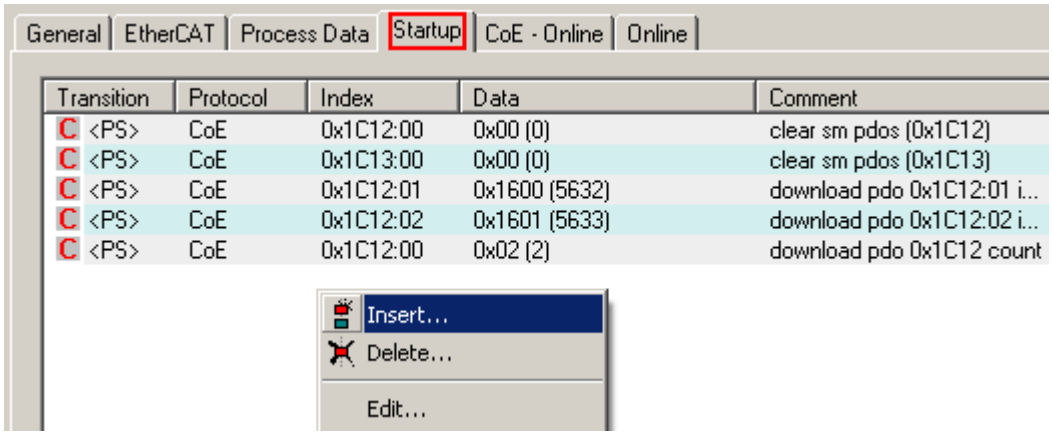



Fig. 24: Startup list in the TwinCAT System Manager

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.

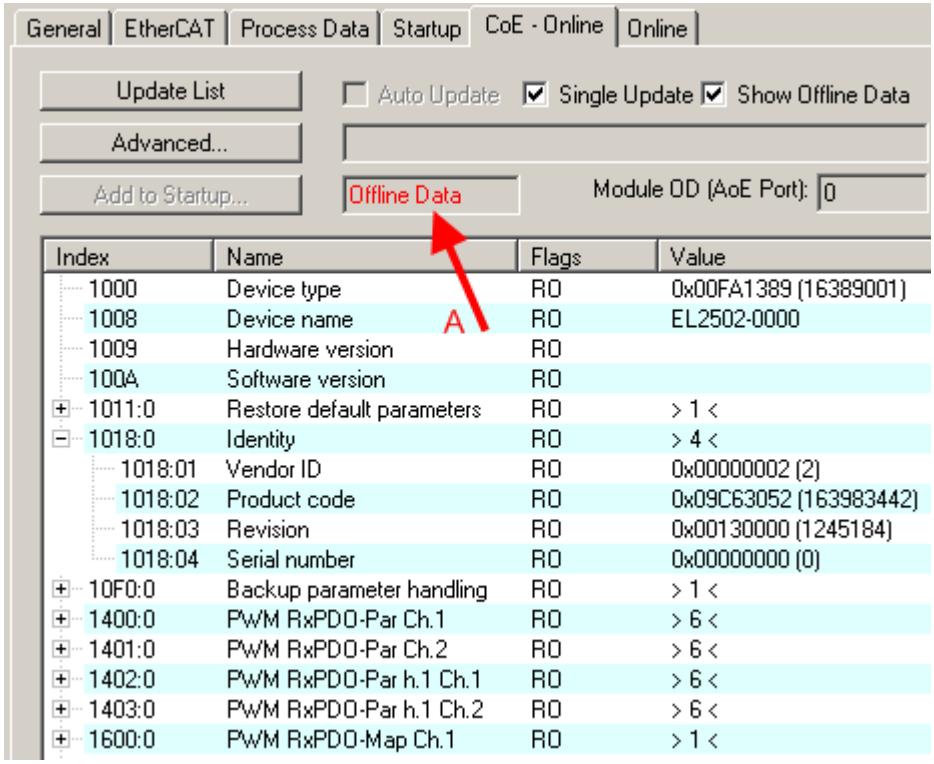



Fig. 25: Offline list

- 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.

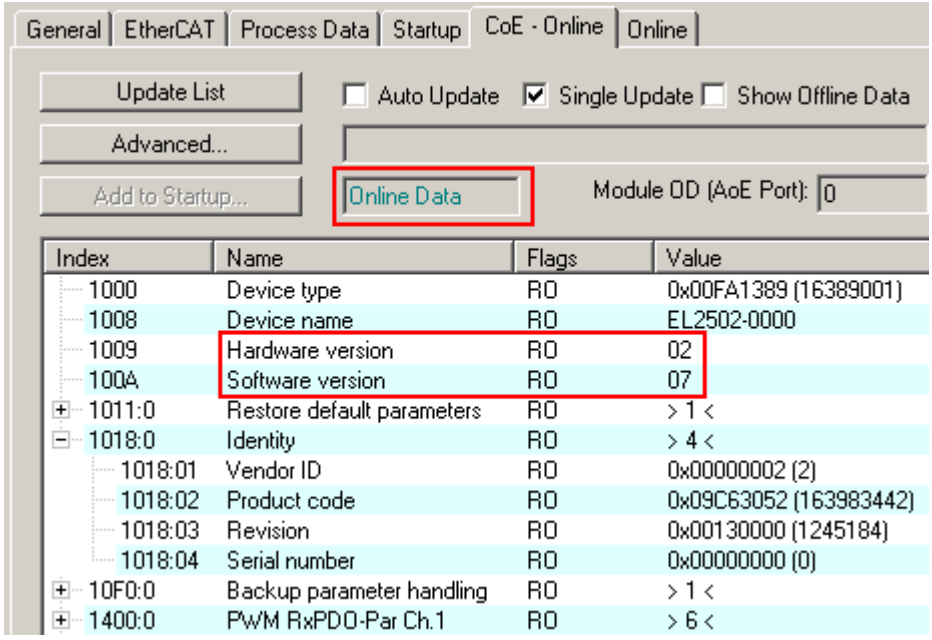



Fig. 26: Online list

### 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.

# 5 Parameterization and commissioning

## 5.1 EL6731 - PROFIBUS master terminal

### 5.1.1 PROFIBUS protocols

As master, the PROFIBUS DP, PROFIBUS DPV1, PROFIBUS DPV2, S5-FDL-AGAG communication (only FC310x) and the PROFIDRIVE-PKW interface protocols are supported.

#### PROFIBUS DP

A summary of the PROFIBUS-DP master functions follows:

| Function | Description |
|---|---|
| **Standard DP** | The PROFIBUS DP [▶ 39] section describes the necessary steps for establishing a DP connection (Set_Prm - parameter, Chk_Cfg - configuration) and for the exchange of user data (Data_Exchange). |
| **Task synchronization** | The Synchronization [▶ 42] section describes how the TwinCAT task is synchronized with the PROFIBUS cycle. |
| **Slave priorities (only FC310x, FC3151, CX1500-M310, EL6731)** | The slaves can receive telegrams with differing cycle times. The necessary settings are described in the Slave Prioritization/ Multiple DP Cycles [▶ 130] section. |
| **Several DP cycles (only FC310x, FC3151, CX1500-M310, EL6731)** | In order to receive the most recent possible inputs when the task cycles are long, a number of DP cycles can be carried out for each task cycle, as described in the section on Slave Prioritization/Multiple DP Cycles [▶ 130]. |
| **Diagnostics** | The diagnostic [▶ 119] facilities are described in this section. |
| **Error Reactions** | It is possible for different error reactions [▶ 116] to be set in the event of a fault (a slave fails or the task is stopped). |
| **Sync/Freeze** | Activation of the sync and freeze commands is described in the Sync/Freeze [▶ 43] section. |
| **Upload Configuration** | The slaves connected to the PROFIBUS can be read via Upload Configuration [▶ 50]. |
| **Master redundancy (only FC310x)** | The Master redundancy section describes the settings required to have a second master with the same configuration configured as a standby master (as from TwinCAT 2.9). |

#### PROFIBUS DPV1

A summary of the PROFIBUS-DPV1 master functions follows:

| Function | Description |
|---|---|
| **MSAC_C1** | The MSAC_C1 [▶ 47] connection is established along with the cyclic connection. The Read, Write and Data_Transport services are supported. |
| **MSAC_C2** | The MSAC_C1 [▶ 47] connection is established independently of the cyclic connection, and can also be used by a second master (while the first one is communicating with the slave over the cyclic MSAC_C1 connection). The Initiate, Abort, Read, Write and Data_Transport services are supported. |

#### PROFIBUS DPV2

A summary of the PROFIBUS-DPV2 master functions follows:

| Function | Description |
|---|---|
| **Equidistance** | The DPV2 equidistance functionality is described in the PROFIBUS MC [▶ 41] section. |

#### S5-FDL-AGAG Communication (only FC310x)

S5-FDL-AGAG communication is described in the documentation of FC310x in S5-FDL section.

#### PROFIDRIVE-PKW Interface

The PROFIDRIVE PKW protocol [▶ 50] is implemented in the PROFIBUS master, and can be used by means of acyclic ADS calls.

## 5.1.2 PROFIBUS DP

#### Standard DP Operation

In order to configure standard DP operation, proceed as follows in the TwinCAT system manager:

#### Configure the DP Master

#### FC310x

It is first necessary to configure a "Profibus Master FC310x, PCI" I/O device (selecting "I/O devices" with the right hand mouse button, and then selecting "Insert device"). The appropriate channel is on the "FC310x" tab and searched for ("Search" button) and the baud rate, which is set to 12 Mbaud as standard, adjusted if necessary.

#### EL6731

Procedure as above; configure "Profibus master EL6731, EtherCAT" (right-click on "I/O devices", then select "Append Device").

#### Add DP slaves

The Beckhoff slaves or devices from other manufacturers are to be configured. All slaves whose GSD files are stored in the system manager's Profibus subdirectory are displayed automatically, sorted according to manufacturer; in order to link other GSD files, "General Profibus box (GSD)" is to be selected under Miscellaneous.

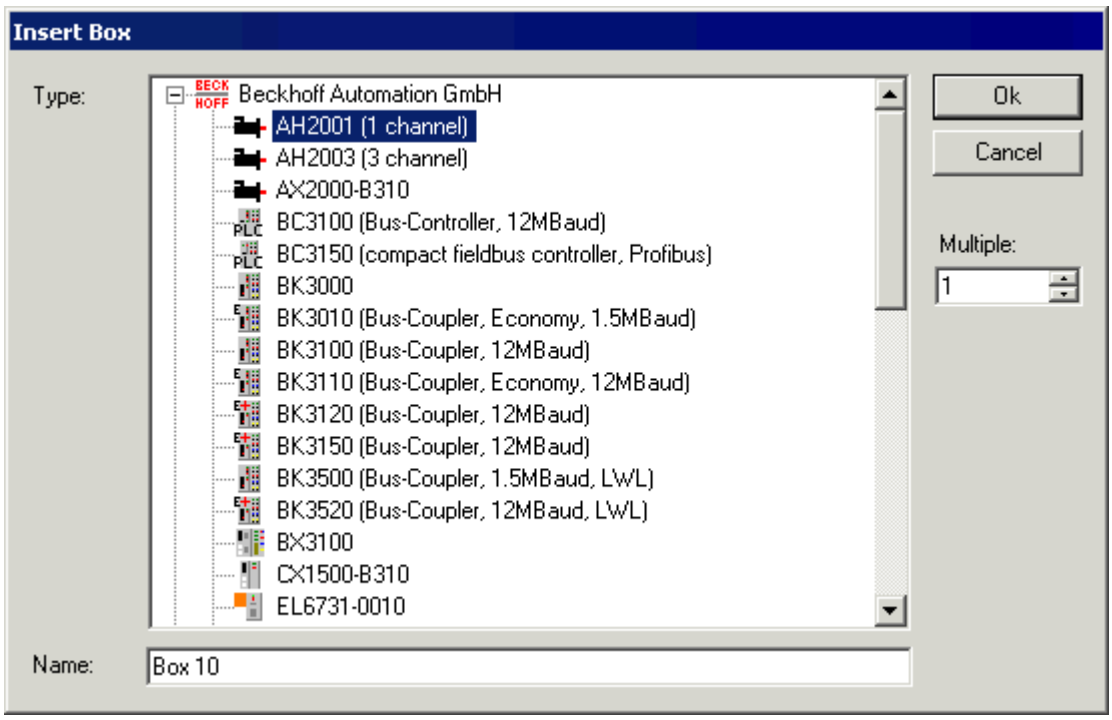



Fig. 27: Adding a DP slave

In the case of modular slaves, it is then still necessary to add the terminal/IL modules (for Beckhoff slaves) or the DP module (for devices from other manufacturers).

### System Start

### TwinCAT Config mode

For the TwinCAT configuration mode, it is only necessary to exchange data with the configured slaves. To do this, TwinCAT configuration mode is started, and configuration of the DP master is activated using the "Reload devices" button on the toolbar. After this, the data relating to the configured slaves can be read and written on the associated Variables tab from the System Manager.

### TwinCAT Run-Mode

For the TwinCAT Run mode, it is now necessary for at least one variable of the PROFIBUS master or of the configured slave to be linked to a task. The project is then to be stored in the registry, and the TwinCAT system then started in Run mode. Data exchange with the slaves is not carried out until the associated task is started. If a number of tasks are linked with the PROFIBUS master or with the configured slave, then whichever task has the highest priority must be started in order for data to be exchanged with the slave.

### Bus parameters

The PROFIBUS DP-bus parameters can be found in the Bus parameters [▶ 88] dialog, which can be selected via the **"FC310x"** or **"EL6731"** tab (Bus parameters (DP) button). They should only be modified by experienced users.

## 5.1.3 PROFIBUS MC

The difference between PROFIBUS MC and PROFIBUS DP is that the PROFIBUS cycle is constant, with a jitter of a few microseconds (for PROFIBUS DP, the jitter is greater than 100 µs), and at the start of the cycle a broadcast global control telegram is sent, which can be used by the MC slaves for synchronization. This enables precise synchronization of drive control loops with the NC.

However, this precise synchronization means that bus disturbances, switching off of slaves, pulling of bus plugs etc. will usually lead to a loss of synchronicity between master and slave, since the bus timing is changed.

### DP/MC Equidistant Mode

In order to operate the EL6731 with PROFIBUS MC, the **Operation Mode** "DP/MC (equidistant)" must be set on the tab "**EL6731**" (for TwinCAT) of the master. Whichever task uses the equidistant functionality of the EL6731 (most often this is the NC task) should have the highest priority, as otherwise the synchronicity can be disturbed.

The EL6731 generally runs in Sync Master mode, EtherCAT synchronized via the distributed clocks, using DC Optimized mode. The shift time in the distributed clock settings of the EtherCAT master must be equal or greater than the CalcAndCopy time of the EL6731. The CalcAndCopy time of the EL6731 depends on the number of configured DP slaves, although it can be measured in OPERATIONAL state (set entry 0x1C32:08 to 1, then read entry 0x1C32:06). The following diagram shows the cycle for a task with IO at the start of the task:

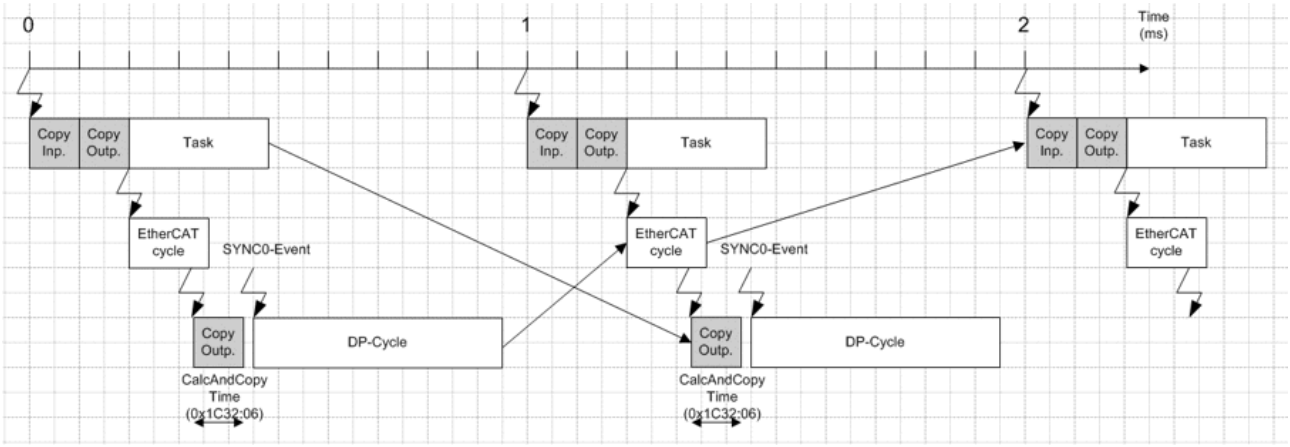



Fig. 28: EL6731 task cycle

### Setting of Equidistant Times

The button **Calculate MC-Times** (TwinCAT; see tab MC [▶ 91]) can be used to set all equidistant parameters automatically.

### Diagnosis of Equidistant Times

The MC-Diag [▶ 95] tab in the System Manager or ADS in the control program is available for diagnosing the equidistance times (see section Master diagnostics [▶ 119]).

## 5.1.4 Synchronization

### 5.1.4.1 Overview

In TwinCAT RunMode, the EL6731 is always synchronized with the highest priority task with which the variables are linked. A separate EtherCAT telegram is defined for each EL6731. The cycle time for the corresponding task is displayed under Cycle time on the master's "EL6731" tab, as soon as the mapping has once been created. It is possible to set for the task whether the "I/O at task start" should be updated or not.

#### I/O at Task Start

If the setting "I/O at Task Start" (check box) is selected (default setting for NC task), the EtherCAT telegram is transferred before the task starts.

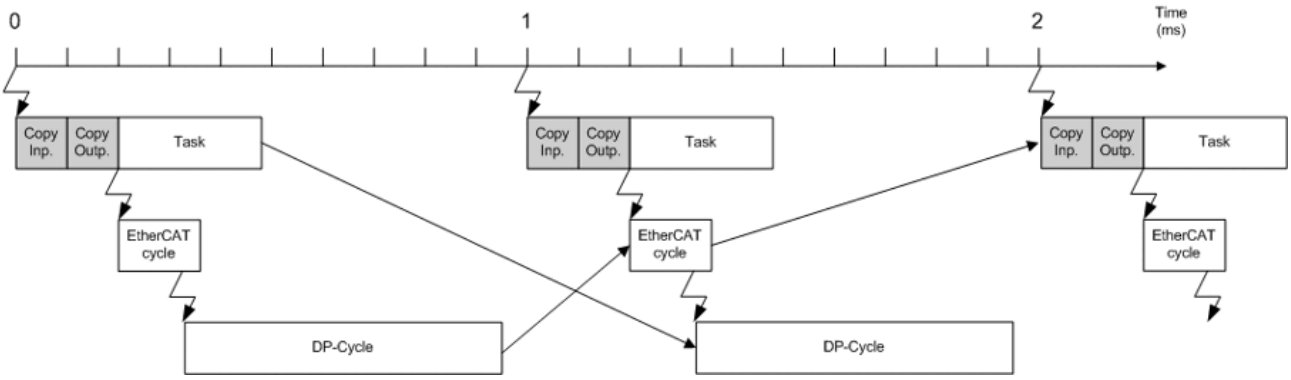



Fig. 29: Cycle with IO at task start

#### I/O not at Task Start

If the setting "I/O at Task Start" (check box) is not selected (default setting for PLC task), the EtherCAT telegram is transferred after the task is complete. Compared with "I/O at Task Start" the outputs are therefore newer by one cycle, although the EtherCAT telegram jitters with the runtime of the task.

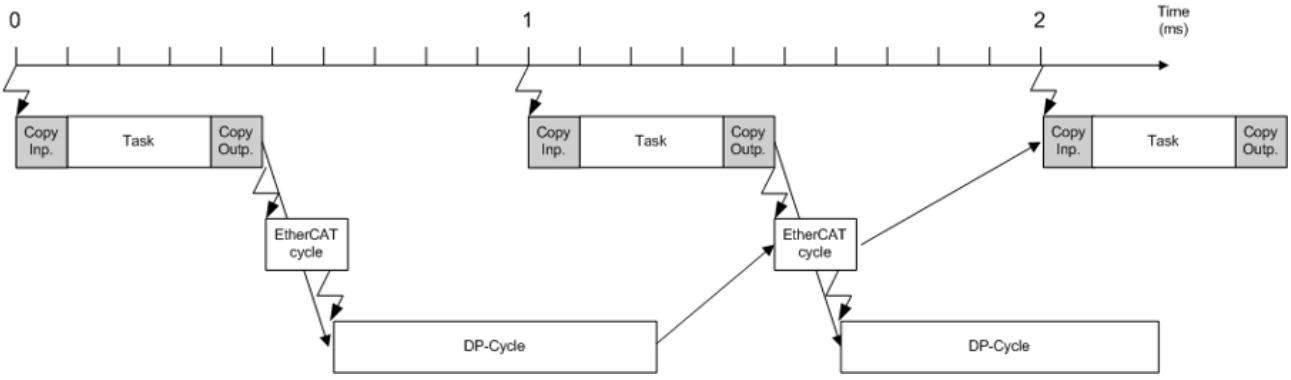



Fig. 30: Cycle with IO not at task start

### 5.1.4.2 Sync/Freeze functionality

Sync is used for the simultaneous outputting of outputs for several slaves, Freeze is used for reading in inputs from several slaves simultaneously.

The sequence in TwinCAT with FC310x / EL6731 and Bus Couplers (in K-bus synchronous mode) would therefore be as follows (see section Synchronization [▶ 42]):

- The outputs are written at the beginning (I/O at the start of the task) or the end (I/O not at the start of the task) of the task cycle
- This will start the PROFIBUS cycle
- A Sync/Freeze telegram is sent at the start of the PROFIBUS cycle
- This will cause the Bus Couplers to start a K-Bus cycle with the outputs from the last task cycle and transfer the inputs from the last K-Bus cycle
- The master will then send the current outputs to each slave and pick up the transferred inputs
- The inputs are read at the start of the next task cycle
- etc.

Outputs and inputs are therefore always one cycle old.

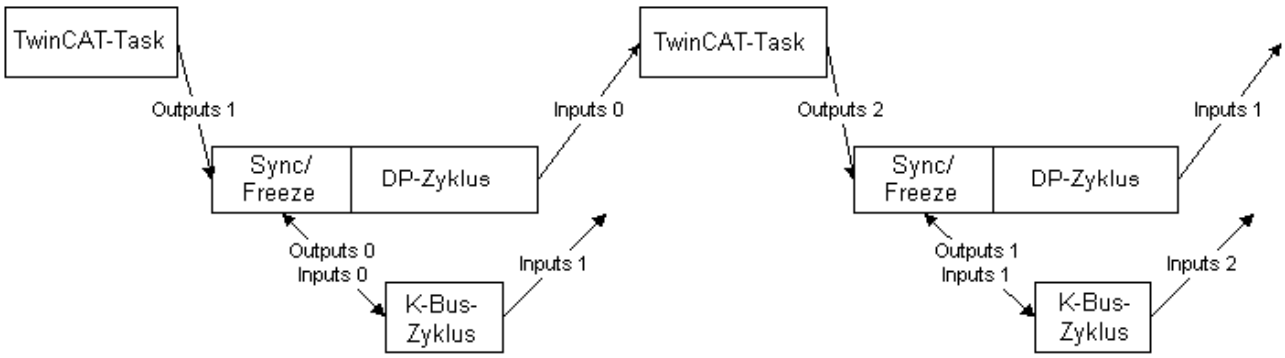



Fig. 31: TwinCAT task, DP cycle and K-bus cycle sequence

On the FC310x/EL6731, set the **Operation Mode** on the **"FC310x"** or **"EL6731"** (for TwinCAT) tab of the master to "DP/MC (equidistant)". At the boxes to be operated via Sync/Freeze, click the checkbox **Sync/ Freeze enable** on the slave's Profibus [▶ 97] tab. The master always uses group 1 for the Sync/Freeze synchronization.

## 5.1.5 ADS (acyclic services)

### 5.1.5.1 ADS Interface

All acyclic data are transmitted to or from the FC310x/EL6731 via ADS-Read, ADS-Write or ADS-Write-Control. The FC310x/EL6731 has its own Net ID and supports the following ports:

| Port | Description |
|---|---|
| 200 | Addresses the FC310x/EL6731 itself, i.e. data stored locally on the FC310x/EL6731, for which usually no additional bus access is required. |
| 0x1000 - 0x107E | This addresses a connected PROFIBUS device, with the address calculated from port-0x1000; this always involves a bus access |

#### ADS-Read

An overview of the IndexGroups/IndexOffsets supported by the FC310x/EL6731 during ADS-Read is provided below.

**IndexGroup for local addressing of the FC310x/EL6731 (port 200)**

| IndexGroup (Lo-Word) | IndexGroup (Hi-Word) | IndexOffset | Description | |
|---|---|---|---|---|
| | | | FC310x | EL6731 |
| 0xF100 | 0x00 | BYTE offset within the data | This is used to read the diagnostic data of the FC310x/EL6731. If the ADS-Read is answered without error (error code = 0), the data will contain the diagnostic data of the FC310x/EL6731 described in section Master Diagnostics [▶ 119]. The FC310x/EL6731 resets its DiagFlag, which is set again when the diagnostic data of the FC310x/EL6731 change again. | |
| 0xF181 | 0x00-0x7E | BYTE offset within the data | This will read the diagnostic data of a configured DP slave. The station address is calculated from the IndexGroup(Hi-Word). If the ADS Read is answered without error (error code = 0), the data will contain the diagnostic data of a configured DP slave described in section Slave-Diagnose [▶ 120]. | |
| 0xF830 | 0x8000-0x807E | always 0 | This enables detection of the DP slaves present at the PROFIBUS, independent of whether they were configured or not. The station address is calculated from IndexGroup(Hi-Word)-0x8000. If the ADS-Read is answered without error (error code = 0), the corresponding DP slave has answered correctly. The data contain the Ident no. of the slave (BYTE offset 0-1) and the read CfgData (from BYTE offset 2) (see section Upload Configuration [▶ 50]). | |
| 0xF840 | 0 | 0 | This is used to read the firmware version and station address of the FC310x/EL6731. If the ADS-Read is answered without error (error code = 0), the data will contain the firmware version (BYTE offset 0-1) and the station address of the FC310x/EL6731 (BYTE offset 2). | This is used to read the firmware version and station address of the FC310x/EL6731. If the ADS-Read is answered without error (error code = 0), the data will contain a contant (BYTE-Offset 0-1) and the station address of the EL6731 (BYTE-Offset 2).<br>Data[0] 0x45<br>Data[1] 0x23<br>Data[2] Adress<br>Data[3] 0x00 |

**Change of the PROFIBUS station address via ADSWRITE (only EL6731 and EL6731-0010)**

If the station address is set with this command, it is saved in the flash ROM of the terminal.
In this case, a designated station address via the Init commands (index 0x8000) is ignored.
After restoring the default parameters with index 0x1010, the address in index 0x8000:01 (for PROFIBUS Master: 0xF800:01) is accepted again.
Note: to receive a new address, the terminal has set through the INIT state to OP state again.
NetId of the EL6731 or EL6731-0010
Port: 200

| Index-Group (Lo-Word) | Index-Group (Hi-Word) | Index-Offset | Länge (in Byte) | Description |
|---|---|---|---|---|
| 0xF480 | 0x00 | 0 -2 | 4 | Data[0] 0x45<br>Data[1] 0x23<br>Data[2] PROFIBUS address < 127<br>Data[3] 0x00 |

**IndexGroup for addressing of a configured PROFIBUS device (port 0x1000-0x107E)**

| IndexGroup (Lo-Word) | IndexGroup (Hi-Word) | IndexOffset | Description |
|---|---|---|---|
| 0x00-0xFF | 0x00 | 0x00-0xFF | This will send a DPV1-Read to the appropriate, configured DPV1 slave via a Class 1 connection; the DPV1 slot number corresponds to the IndexGroup, the DPV1 index corresponds to the IndexOffset. If the ADS-Read is answered without error (error code = 0), the data will contain the read DPV1 data (see section DPV1 [▶ 47]) |
| 0x100-0x1FF | 0x00 | 0x00-0xFF | This will send a DPV1-Read to the appropriate, configured DPV1 slave via a Class 2 connection; the DPV1 slot number corresponds to the IndexGroup - 0x100, the DPV1 index corresponds to the IndexOffset. If the ADS-Read is answered without error (error code = 0), the data will contain the read DPV1 data (see section DPV1 [▶ 47]) |
| 0x0000 - 0xFFF | 0x10000000 - 0xF0000000 | 0x00-0xFF | This will transmit a PKW-Read to the appropriate, configured PROFIDRIVE slave; the parameter number (PNU) is contained in the Low WORD of the IndexGroup, the sub-index for access to an array is in the IndexOffset, the addressed axis is in bits 28-31 of the IndexGroup (for a 1-axis unit, this must be 1), the PKW compatibility can be adjusted in bits 26, 27 (unfortunately, not all PROFIDRIVE slaves are compatible, see section PKW-Interface [▶ 50]). |
| 0 | 0x01000000 | 0 | This will send FDL-Read for Siemens AG interfacing to the appropriate configured FDL station (only FC310x, see documentation for FC310x section S5-FDL). |

**ADS Write**

An overview of the IndexGroups/IndexOffsets supported by the FC310x/EL6731 during ADS-Write is provided below.

**IndexGroup for local addressing of the FC310x/EL6731 (port 200)**

| IndexGroup (Lo-Word) | IndexGroup (Hi-Word) | IndexOffset | Description |
|---|---|---|---|
| 0xF100 | 0x00 | 0 -2 | This will reset the equidistant diagnostic data (IndexOffset = 0), the Repeat counters (IndexOffset = 1) or the NoAnswer counters (IndexOffset = 2) of the FC310x/EL6731. |

**IndexGroup for addressing of a configured PROFIBUS device (port 0x1000-0x107E)**

| IndexGroup (Lo-Word) | IndexGroup (Hi-Word) | IndexOffset | Description |
|---|---|---|---|
| 0x00-0xFF | 0x00 | 0x00-0xFF | This will send a DPV1-Write to the appropriate, configured DPV1 slave via a Class 1 connection; the DPV1 slot number corresponds to the IndexGroup, the DPV1 index corresponds to the IndexOffset (see section DPV1 [▶ 47]). |
| 0x100-0x1FF | 0x00 | 0x00-0xFF | This will send a DPV1-Write to the appropriate, configured DPV1 slave via a Class 2 connection; the DPV1 slot number corresponds to the 0x100 IndexGroup, the DPV1 index corresponds to the IndexOffset (see section DPV1 [▶ 47]). |
| 0x400 | 0x00 | 0x00 | With this, a DPV1 Abort is sent to the appropriate configured DPV1 slave via a class 2 connection; the abort parameters are included in the data (-> section DPV1 [▶ 47]). |
| 0x0000 - 0xFFF | 0x10000000 - 0xF0000000 | 0x00-0xFF | This will transmit a PKW-Write to the appropriate, configured PROFIDRIVE slave; the parameter number (PNU) is contained in the Low WORD of the IndexGroup, the sub-index for access to an array is in the IndexOffset, the addressed axis is in bits 28-31 of the IndexGroup (for a 1-axis unit, this must be 1), the PKW compatibility can be adjusted in bits 26, 27 (unfortunately, not all PROFIDRIVE slaves are compatible, see section PKW-Interface [▶ 50]). |
| 0 | 0x01000000 | 0 | This will send FDL-Write for Siemens AG interfacing to the appropriate configured FDL station (only FC310x, see documentation for FC310x section S5-FDL). |
| 0 | 0x02000000 | 0 | This will send a SetSlaveAddress command to a configured DP slave, whereby the DP slave must be configured with the new station address; the old station address must be entered at BYTE offset 0 of the ADS-Write data. Furthermore, the Ident no. of the slave must be contained under BYTE offset 1 and 2, and BYTE offset 3 must contain information as to whether the slave may be modified later (0) or not (not equal 0). Altogether, 4 bytes of ADS-Write data will therefore have to be sent. |

### ADS-ReadWrite

An overview of the IndexGroups/IndexOffsets supported by the FC310x/EL6731 during ADS-ReadWrite is provided below.

**IndexGroup for addressing of a configured PROFIBUS device (port 0x1000-0x107E)**

| IndexGroup (Lo-Word) | IndexGroup (Hi-Word) | IndexOffset | Description |
|---|---|---|---|
| 0x100-0x1FF | 0x00 | 0x00-0xFF | This will send a DPV1-Data_Transport to the appropriate, configured DPV1 slave via a Class 2 connection; the DPV1 slot number corresponds to the 0x100 IndexGroup, the DPV1 index corresponds to the IndexOffset (see section DPV1 [▶ 47]). |
| 0x200 | 0x00 | 0x00 | With this, a DPV1 Initiate is sent to the appropriate configured DPV1 slave via a class 2 connection; the initiate parameters are included in the data (-> section DPV1 [▶ 47]). |

### ADS-WriteControl

An overview of the ADS-Write-Control commands supported by the FC310x/EL6731 is provided below.

**ADS-WriteControl for local FC310x/EL6731 addressing (port 200)**

| AdsState | DeviceState | State of the FC310x/ EL6731 | Description |
|---|---|---|---|
| STOP (6) | 0x00 | RUN (5) | This will stop the FC310x/EL6731, i.e. the process data connections to all DP slaves (Data_Exchange) are removed (with SetPrm,Unlock). |
| RUN (5) | 0x00 | STOP (6) | This will restart the FC310x/EL6731 after a stop, i.e. the process data connections to all DP slaves (Data_Exchange) are re-established (normal DP start-up). |

**ADS-WriteControl for addressing a configured PROFIBUS device (port 0x1000-0x107E)**

| AdsState | DeviceState | State of the FC310x/ EL6731 | Description |
|---|---|---|---|
| STOP (6) | 0x00 | RUN (5) | This will stop the slave, i.e. the process data connection to the relevant DP slave (Data_Exchange) is removed (with SetPrm,Unlock). |
| RUN (5) | 0x00 | STOP (6) | This will restart the slave after a stop, i.e. the process data connection to the relevant DP slave (Data_Exchange) is re-established (normal DP start-up). |

**ADS Error Codes**

The 32 bit ADS error code always consists of a general ADS error code (Low Word, see ADS documentation) and a FC310x/EL6731-specific, unique error code (High Word, -> chapter ADS error codes [▶ 128]). The appropriate text message will also be displayed in the TwinCAT System Manager Logger.

### 5.1.5.2 PROFIBUS DPV1

On a C1 connection, the master supports the Read and Write services, and on the C2 connection it supports the Read, Write, Data_Transport, Initiate and Abort services.

**C1 Connection (MSAC-C1)**

The C1 connection is reserved for the master that cyclically exchanges data with the slave (C1 master). In order for a slave to be able to use the C1 connection, the slave must support DPV1 (this means that the line "DPV1_Slave = 1" and the key word "C1_Max_Data_Len" with an appropriate length must be in the GSD file). If it is also generally necessary to activate the C1 functionality by setting bit 7 in the **PrmData** byte 0 (see the slave's Profibus [▶ 97] tab) for the corresponding slave (this is done automatically for those Beckhoff devices that support DPV1).

MSAC-C1-Read is shown in ADS-Read, and MSAC-C1-Write is represented in ADS-Write:

**MSAC-C1 Read**

| ADS-Read parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 0x1000 + station address of the slave |
| IndexGroup | Slot number (DPV1 parameter) |
| IndexOffset | Index (DPV1 parameter) |
| Length | Length of the data that is to be read |
| Data | In response: data that has been read |

### MSAC-C1 Write

| ADS-Write parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 0x1000 + station address of the slave |
| IndexGroup | Slot number (DPV1 parameter) |
| IndexOffset | Index (DPV1 parameter) |
| Length | Length of the data that is to be written |
| Data | In request: data that is to be written |

### C2 Connection (MSAC-C2)

The C2 connection is as a rule intended for a second master (C2 master) that does not communicate with the slave cyclically, but it is also possible for the C1 master to make use of the C2 connection. In order for a slave to be able to use the C2 connection, the slave must support DPV1 (this means that the line "DPV1_Slave = 1" and the key word "C2_Max_Data_Len" with an appropriate length must be in the GSD file).

The connection is automatically established by the master as soon as a Read, Write or Data_Transport access is requested; it can, however, also be explicitly established through an Initiate. When the connection is being established automatically, the master sends the Initiate parameters that have most recently been passed (see the description of Initiate), and initializes the Initiate parameters with 0 after a TwinCAT start (or restart); monitoring of the connection is an exception - this is initialized in accordance with the value set in the System Manager (**Watchdog** under **DPV1 Class 2** on the slave's Profibus [▶ 97] tab).

It is also necessary for the C2 functionality to be activated for each slave that is to be addressed using C2 services by selecting the **Enable** check box under **DPV1 Class 2** (see the slave's Profibus [▶ 97] tab).

If a different master performs cyclic data exchange with the slave, then the "No cyclic connection" setting must be chosen under **DP Class 2** (see the slave's Profibus [▶ 97] tab). This could, for instance, be useful in order to be able to debug a BC3100/IL23xx-C310 over PROFIBUS, even though it is being operated by an external controller.

MSAC-C2-Read is represented in ADS-Read, MSAC-C2-Write in ADS-Write, MSAC-C2-Data_Transport in ADS-ReadWrite, MSAC-C2-Initiate in ADS-ReadWrite, and MSAC-C2-Abort in ADS-Write:

### MSAC-C2 Read

| ADS-Read parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 0x1000 + station address of the slave |
| IndexGroup | 0x100 + slot number (DPV1 parameter) |
| IndexOffset | Index (DPV1 parameter) |
| Length | Length of the data that is to be read |
| Data | In response: data that has been read |

### MSAC-C2 Write

| ADS-Write parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 0x1000 + station address of the slave |
| IndexGroup | 0x100 + slot number (DPV1 parameter) |
| IndexOffset | Index (DPV1 parameter) |
| Length | Length of the data that is to be written |
| Data | In request: data that is to be written |

### MSAC-C2 Data_Transport

| ADS-ReadWrite parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 0x1000 + station address of the slave |
| IndexGroup | 0x100 + slot number (DPV1 parameter) |
| IndexOffset | Index (DPV1 parameter) |
| Write-Length | Length of the data that is to be written |
| Read-Length | Length of the data that is to be read |
| Data | In request: data that is to be written; in Response: data that has been read |

### MSAC-C2 Initiate

The MSAC-C2-Initiate service allows the C2 connection to the slave to be established or, if it already exists, for new Initiate parameters to be passed.

| ADS-ReadWrite parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 0x1000 + station address of the slave |
| IndexGroup | 0x200 + slot number (DPV1 parameter) |
| IndexOffset | 0 |
| Read-Length | Length of the Initiate Response parameter (6) |
| Write-Length | Length of the Initiate Request parameter (10 - 42) |
| Data | Initiate Request parameter or Initiate Response parameter |

### Initiate Request parameter

| | |
|---|---|
| 0x00 - 0x01 | Feature_Supported |
| 0x02 - 0x03 | Profile_Feature_Supported |
| 0x04 - 0x05 | Profile_Ident_number |
| 0x06 | sType |
| 0x07 | sLen: Length of sAddr (0 - 16) |
| 0x08 | dType |
| 0x09 | dLen: Length of dAddr (0 - 16) |
| 0x0A - 0x19 | sAddr |
| 0x1A - 0x29 | dAddr |

### Initiate Response parameter

| | |
|---|---|
| 0x00 - 0x01 | Feature_Supported (value received from slave) |
| 0x02 - 0x03 | Profile_Feature_Supported (value received from slave) |
| 0x04 - 0x05 | Profile_Ident_number (value received from slave) |

### MSAC-C2 Abort

The MSAC-C2 Abort service allows the C2 connection to the slave to be removed again.

| ADS-Write parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 0x1000 + station address of the slave |
| IndexGroup | 0x400 + slot number (DPV1 parameter) |
| IndexOffset | 0 |
| Length | Length of the Abort parameter (3) |
| Data | In request: Abort parameter |

**Abort parameter**

| | |
|---|---|
| 0x00 | Reason_Code |
| 0x01 - 0x02 | Additional_Detail |

### 5.1.5.3 Uploading the Configuration

The PROFIBUS can be scanned by ADS [▶ 43] Read for new devices during operation:

| ADS-Read parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 200 |
| IndexGroup | 0xzzyyF830 (yy = station address, zz = 0: in Beckhoff devices, tables 0, 1 and 9 are read, zz = 0x80: Beckhoff devices return the same information as devices from other manufacturers) |
| IndexOffset | 0 |
| Length | 1538 |
| Data | Configuration data of the slave |

If the IndexGroup indicates that, for Beckhoff devices, tables 0, 1 and 9 are to be read, then the following data is supplied, provided the device is a Beckhoff device:

| Offset | Description |
|---|---|
| 0 -1 | 0 |
| 2 -513 | Table 0. Amongst the information contained here is the precise coupler type and the firmware version. |
| 514 - 1025 | Table 9 (includes the coupler number and the terminal numbers) |
| 1026 - 1537 | Table 1 (only relevant for bus controllers. The assignment of the terminals is part of the information contained here) |

If the device is from another manufacturer, or if the IndexGroup indicates that Beckhoff devices are to behave in exactly the same way as devices from other manufacturers, then the following information is returned in the ADS read response:

| Offset | Description |
|---|---|
| 0 -1 | 1 |
| 2 - 7 | DP diagnostic data bytes 0-5 (see Slave Diagnostics [▶ 120]) |
| 8 - 251 | DP configuration data (CfgData [▶ 123]) |

### 5.1.5.4 PKW Interface of PROFIDRIVE Slaves

The PKW interface is integrated into the FC310x/EL6731; it can then be accessed via ADS [▶ 43] from the controller program. PKW Read is then represented by ADS Read, PKW Write by ADS Write and PKW-Read No Of Array Elements by ADS Read:

### PKW Read

| ADS-Read parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 0x1000 + station address of the slave |
| IndexGroup | Bit 0-11: parameter number (PNU) |
| | Bit 12-25: 0 |
| | Bit 26: 1 = Sub-index in octet 3 (standard), 0 = sub-index in octet 4 (Simodrive 611U) |
| | Bit 27: 1 = ARRAY codes are not supported by PROFIDRIVE slave |
| | Bit 28-31: axis number (for single axis modules always 1) |
| IndexOffset | Sub-index (for ARRAY access) |
| Length | Parameter length: 2 or 4 |
| Data | In response: Parameter value |

### PKW Write

| ADS-Write parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 0x1000 + station address of the slave |
| IndexGroup | Bit 0-11: parameter number (PNU) |
| | Bit 12-25: 0 |
| | Bit 26: 1 = Sub-index in octet 3 (standard), 0 = sub-index in octet 4 (Simodrive 611U) |
| | Bit 27: 1 = ARRAY codes are not supported by PROFIDRIVE slave |
| | Bit 28-31: axis number (for single axis modules always 1) |
| IndexOffset | Sub-index (for ARRAY access) |
| Length | Parameter length: 2 or 4 |
| Data | In request: Parameter value |

### PKW ReadNoOfArrayElements

| ADS-Read parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 0x1000 + station address of the slave |
| IndexGroup | Bit 0-11: parameter number (PNU) |
| | Bit 12-15: 0 |
| | Bit 16: 1 |
| | Bit 17-25: 0 |
| | Bit 26: 1 = Sub-index in octet 3 (standard), 0 = sub-index in octet 4 (Simodrive 611U) |
| | Bit 27: 1 = ARRAY codes are not supported by PROFIDRIVE slave |
| | Bit 28-31: axis number (for single axis modules always 1) |
| IndexOffset | 0 |
| Length | Parameter length: 1 |
| Data | In response: Number of the parameter 's array elements |

### 5.1.5.5 FDL interface

The FDL interface can be used to send FDL telegrams and pick up received FDL telegrams via ADS [▶ 43]. This functionality is supported by the FC31xx from V02.66 and the EL6731 from V00.75.

#### FDL send

ADS Write can be used to transfer one or several FDL requests or FDL responses. The FDL requests were sent in the specified order, the FDL responses are sent when they are retrieved by another master.

| ADS-Write parameters | Meaning | | |
|---|---|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) | | |
| Port | 200 | | |
| IndexGroup | 0xF400 | | |
| IndexOffset | 0 | | |
| Length | Data length | | |
| Data | Byte Offset | Meaning | |
| | 0 | Cmd: 1=request follows, 3=response follows, 255=enable reception for FDL service on all SAPs (no further data (length=1)) | |
| | 1 | Request | 0 |
| | | Response | 0=send single, 1=send multiple |
| | 2 | DA byte (bit 7 indicates whether a DSAP exists) | |
| | 3 | SA byte (bit 7 indicates whether an SSAP exists) | |
| | 4 | Request | Request: 0x03=SDA low, 0x05=SDA high, 0x04=SDN low, 0x06=SDN high, 0x0C=SRD low, 0x0D=SRD high |
| | | Response | Response: 0x08=DL, 0x0A=DH |
| | 5 | Length of the following bytes of the request or response | |
| | 6 | if DA.7=1: DSAP otherwise if SA.7=1: SSAP otherwise first data byte | |
| | 7 | if DA.7=1 and SA.7=1: SSAP otherwise if DA.7=1 or SA.7=1 first data byte otherwise second data byte | |
| | 8 | if DA.7=1 and SA.7=1: first data byte otherwise if DA.7=1 or SA.7=1 second data byte otherwise third data byte | |
| | 9-n | further data of the request or response | |
| | n+1 | perhaps next Cmd: 1=request follows, 3=response follows | |
| | ... | ... | |

### FDL receive

Cmd=255 has to be sent once (see FDL send) to ensure that FDL telegrams can be received. Depending on the length, ADS-Read is then used to fetch FDL indications received by another master or locally generated FDL confirmations. An FDL confirmation is returned for each FDL request that is sent.

| ADS-Read parameters | Meaning | | |
|---|---|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) | | |
| Port | 200 | | |
| IndexGroup | 0xF400 | | |
| IndexOffset | 0 | | |
| Length | Length of the read buffer | | |
| Data | Byte Offset | Meaning | |
| | 0 | Cmd: 2=indication follows, 4=confirmation follows | |
| | 1 | Indication: | 1=response sent (only for SRD indications) |
| | | Confirmation: | 0 |
| | 2 | DA byte (bit 7 indicates whether a DSAP exists) | |
| | 3 | SA byte (bit 7 indicates whether an SSAP exists) | |
| | 4 | Indication: | 0x03=SDA low, 0x05=SDA high, 0x04=SDN low, 0x06=SDN high, 0x0C=SRD low, 0x0D=SRD high |
| | | Confirmation: | 0x00=OK, 0x02=RR, 0x03=RS, 0x08=DL, 0x0A=DH, 0x80=NR, 0x9F,0xAF,0xBF=NA |
| | 5 | n: Length of the following bytes of the indication or confirmation | |
| | 6 | if DA.7=1: DSAP otherwise if SA.7=1: SSAP otherwise first data byte | |
| | 7 | if DA.7=1 and SA.7=1: SSAP otherwise if DA.7=1 or SA.7=1 first data byte otherwise second data byte | |
| | 8 | if DA.7=1 and SA.7=1: first data byte otherwise if DA.7=1 or SA.7=1 second data byte otherwise third data byte | |
| | 9-n | Further indication or confirmation data; if n is even, an Align byte is used for filling; (n+1) is always even | |
| | n+1 | perhaps next Cmd: 2=indication follows, 4=confirmation follows | |
| | | | |

## 5.1.6 TwinCAT (2.1x) System Manager

### 5.1.6.1 General notes

#### 5.1.6.1.1 TwinCAT Development Environment

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

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

**Details:**

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

**Additional features:**

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

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

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

### 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.

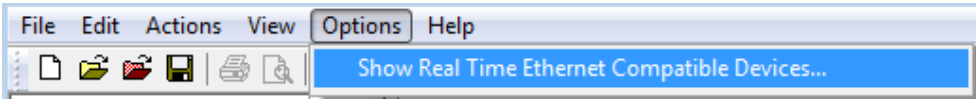



Fig. 32: System Manager “Options” (TwinCAT 2)

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

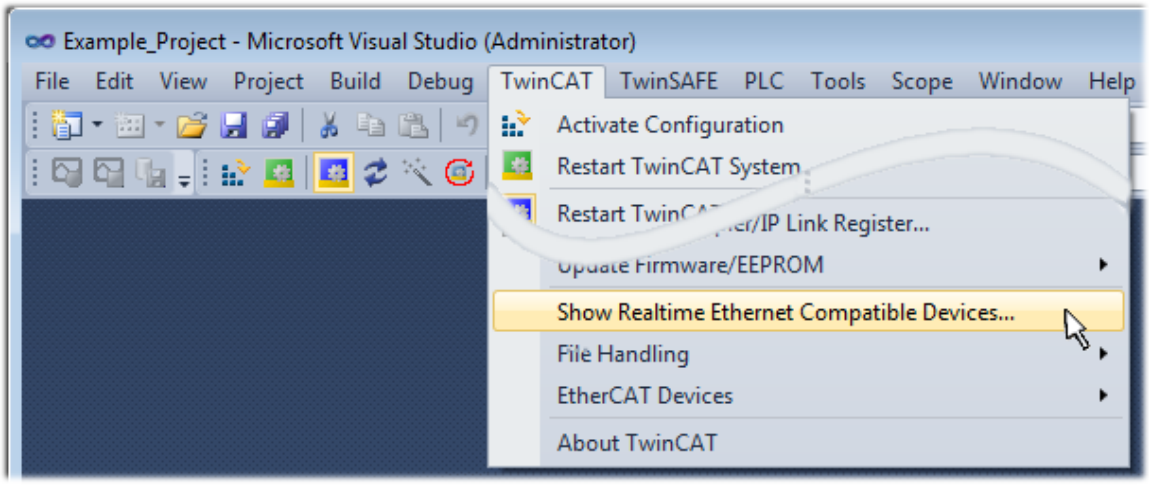



Fig. 33: Call up under VS Shell (TwinCAT 3)

The following dialog appears:

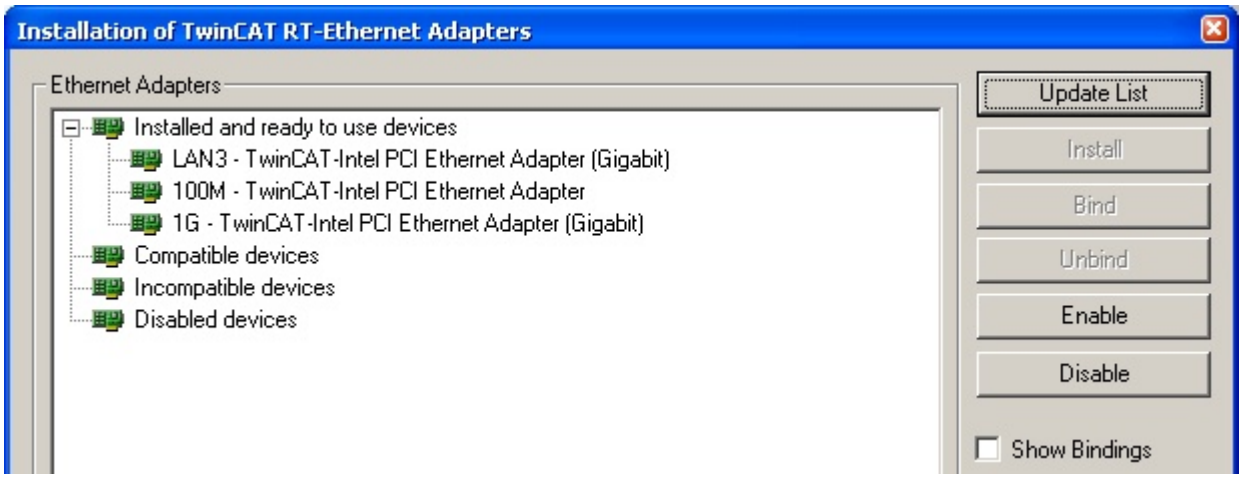



Fig. 34: Overview of network interfaces

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

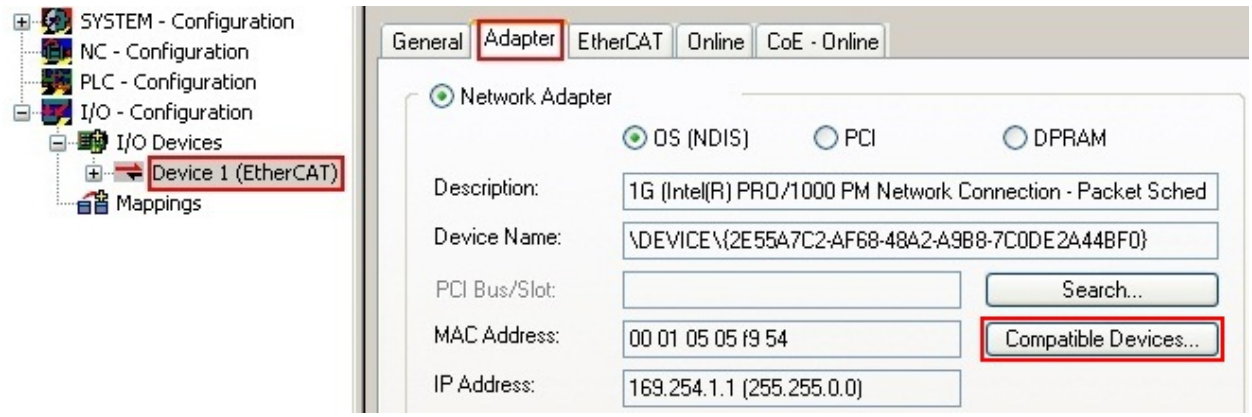
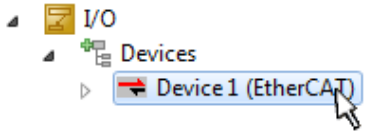



Fig. 35: EtherCAT device properties(TwinCAT 2): click on “Compatible Devices…” of tab “Adapte””

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)

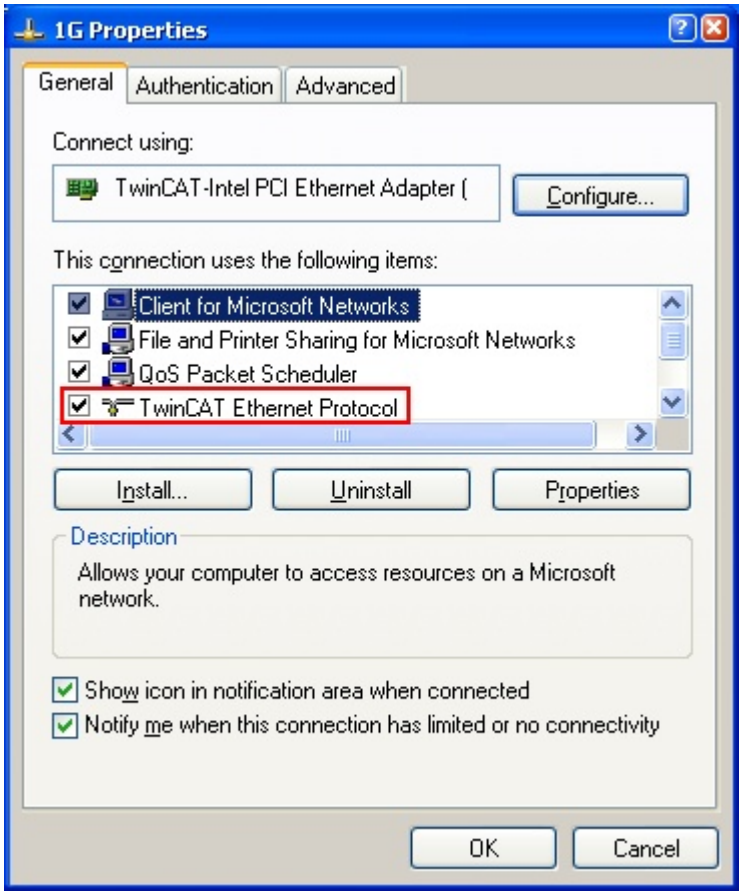



Fig. 36: Windows properties of the network interface

A correct setting of the driver could be:

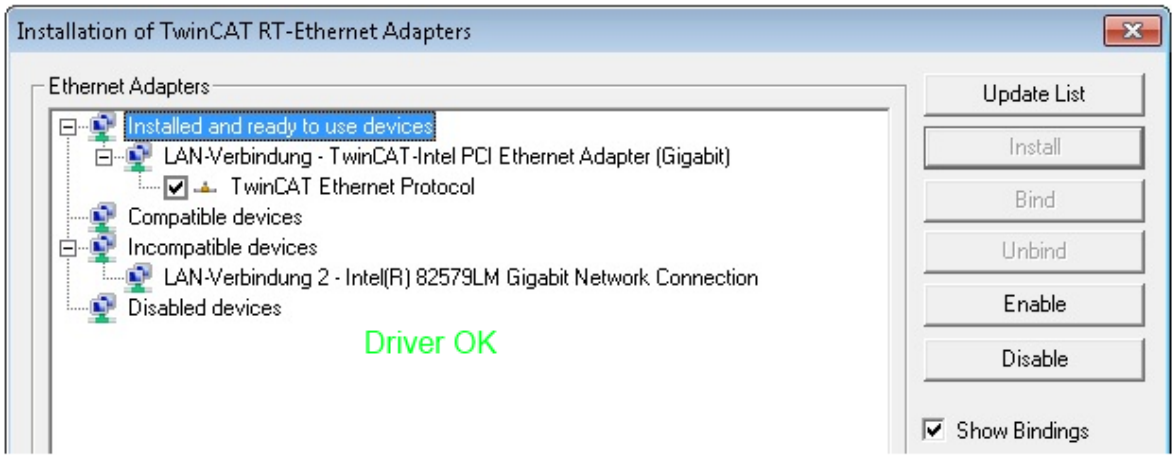



Fig. 37: Exemplary correct driver setting for the Ethernet port

Other possible settings have to be avoided:

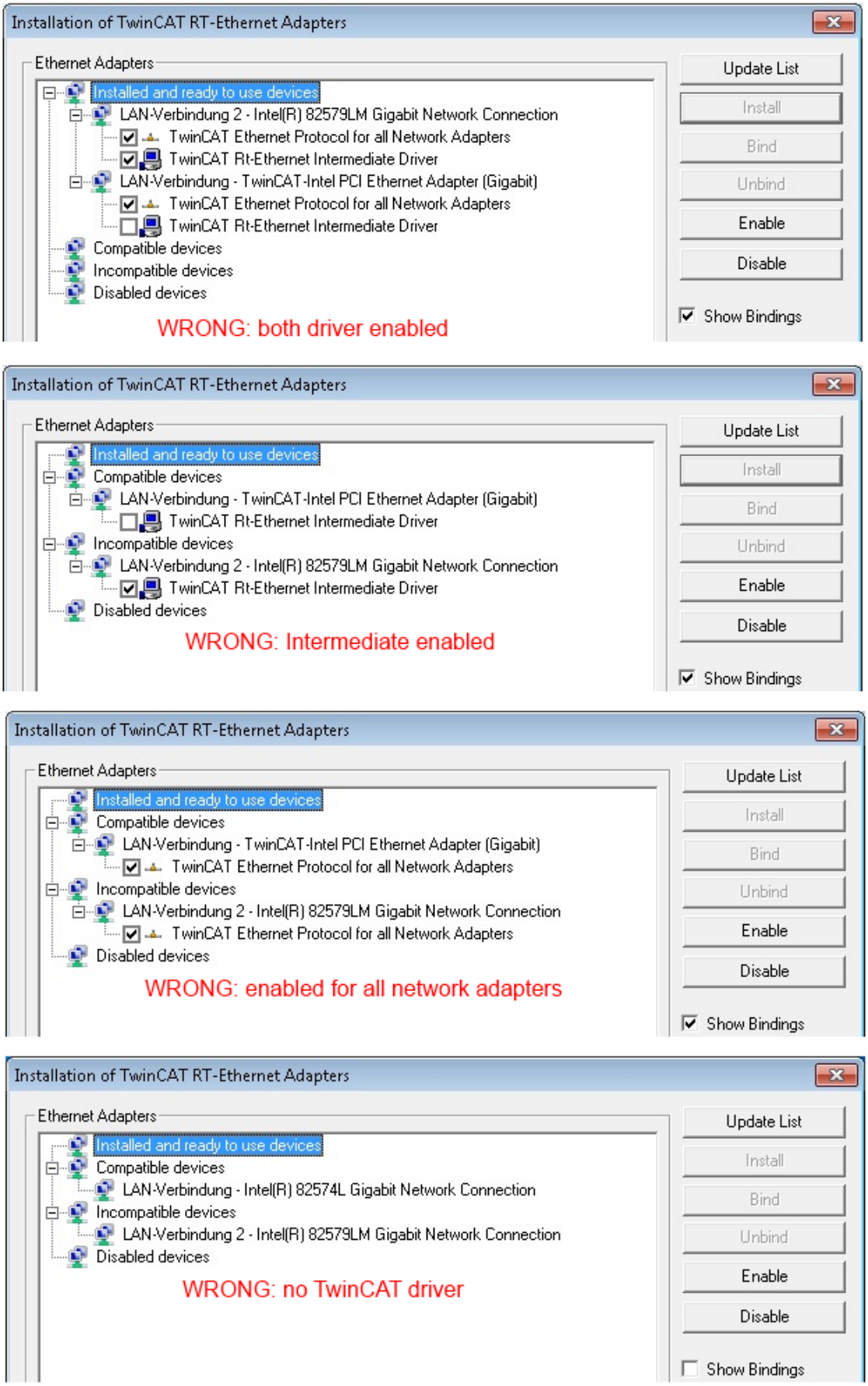



Fig. 38: Incorrect driver settings for the Ethernet port

### IP address of the port used

**IP address/DHCP**

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

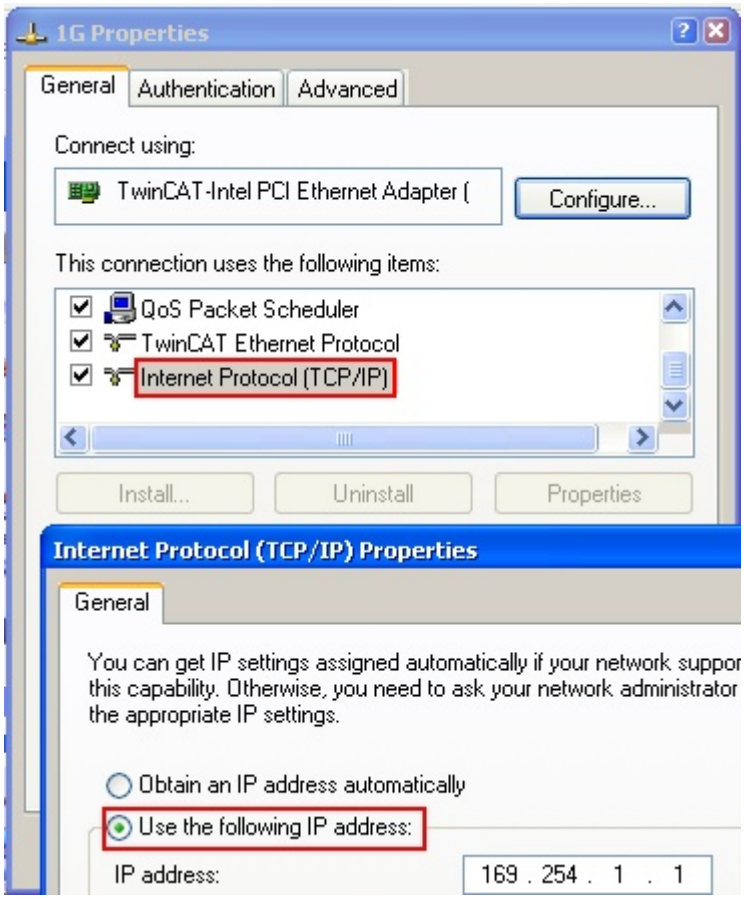



Fig. 39: TCP/IP setting for the Ethernet port

## 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 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”

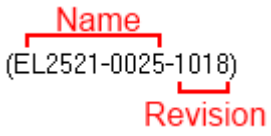



Fig. 40: Identifier structure

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 [▶ 8].

### 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.

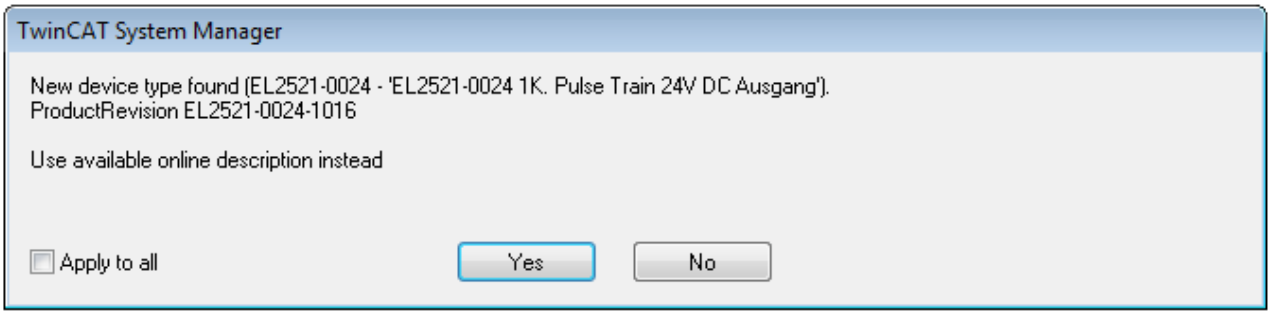



Fig. 41: OnlineDescription information window (TwinCAT 2)

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

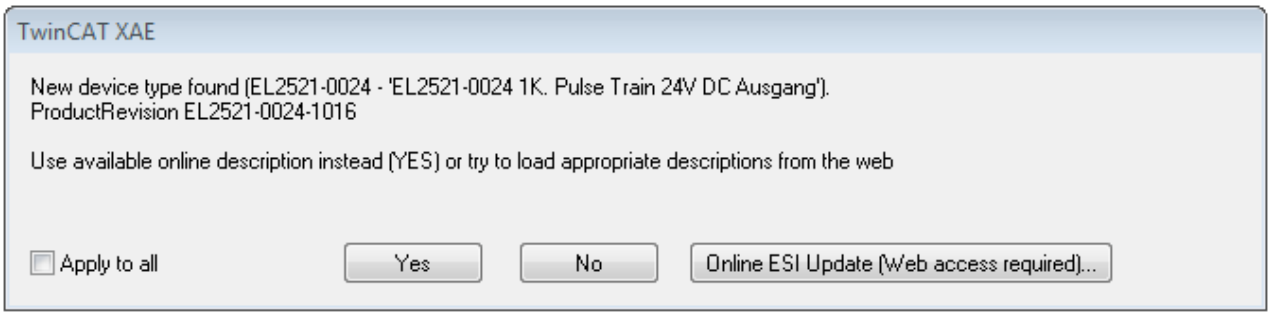



Fig. 42: Information window OnlineDescription (TwinCAT 3)

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 [▶ 64]".

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

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

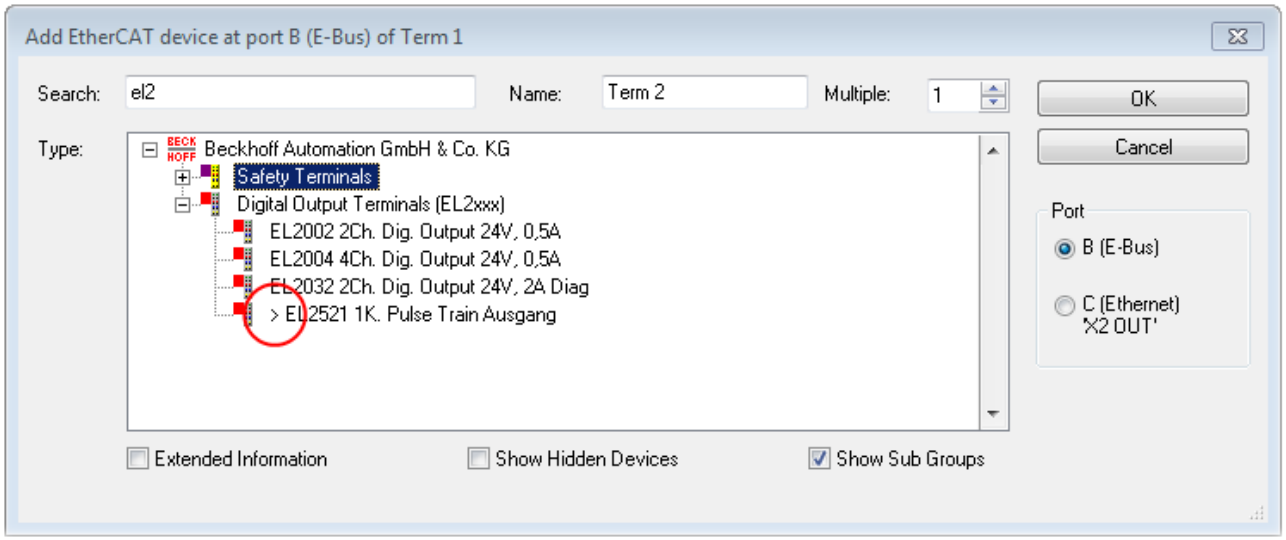



Fig. 44: Indication of an online recorded ESI of EL2521 as an example

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

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

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

**OnlineDescription for TwinCAT 3.x**

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

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

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

### Faulty ESI file

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

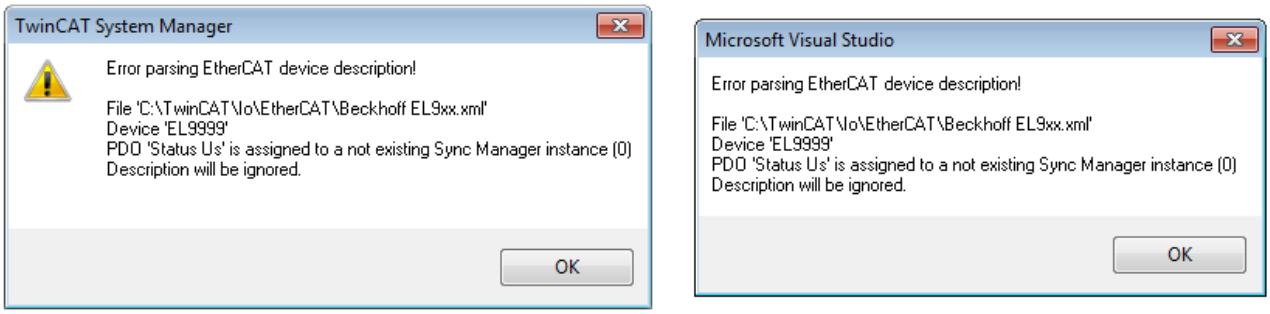



Fig. 45: Information window for faulty ESI file (left: TwinCAT 2; right: TwinCAT 3)

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

## OFFLINE configuration creation

### Creating the EtherCAT device

Create an EtherCAT device in an empty System Manager window.

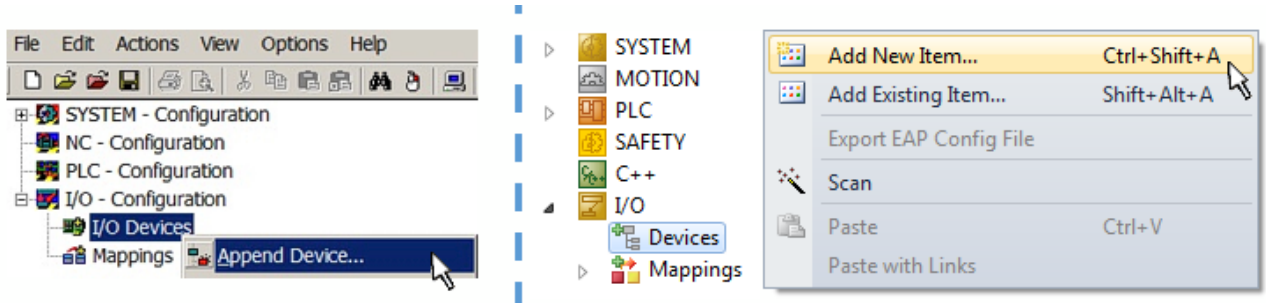



Fig. 46: Append EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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”.

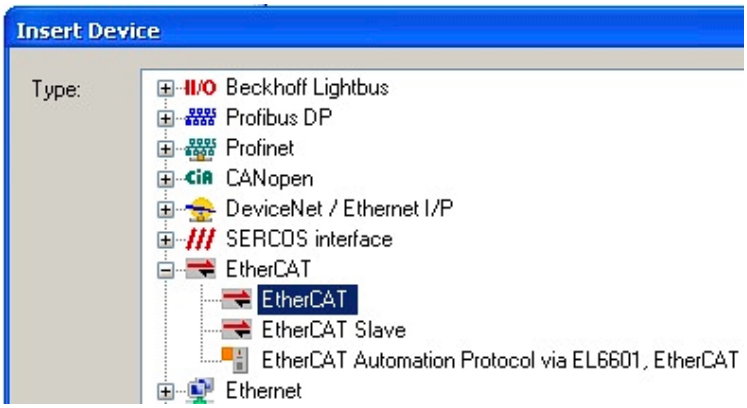



Fig. 47: Selecting the EtherCAT connection (TwinCAT 2.11, TwinCAT 3)

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

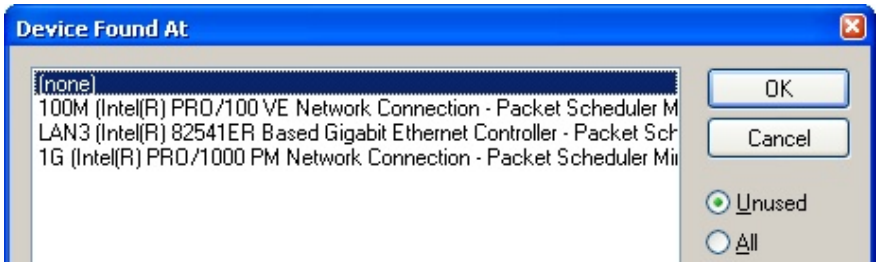



Fig. 48: Selecting the Ethernet port

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)”.

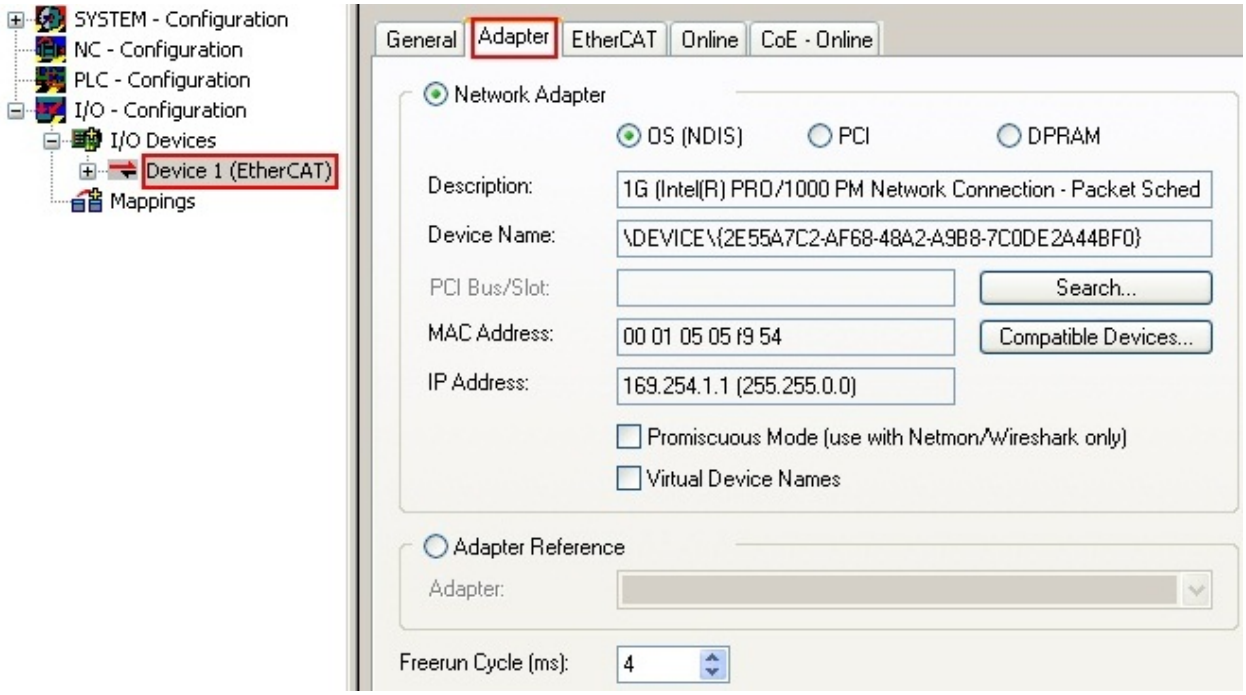
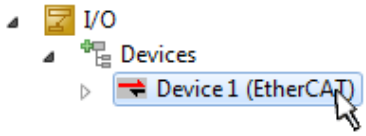



Fig. 49: EtherCAT device properties (TwinCAT 2)

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




### Selecting the Ethernet port

Ethernet ports can only be selected for EtherCAT devices for which the TwinCAT real-time driver is installed. This has to be done separately for each port. Please refer to the respective installation page [▶ 54].

### Defining EtherCAT slaves

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

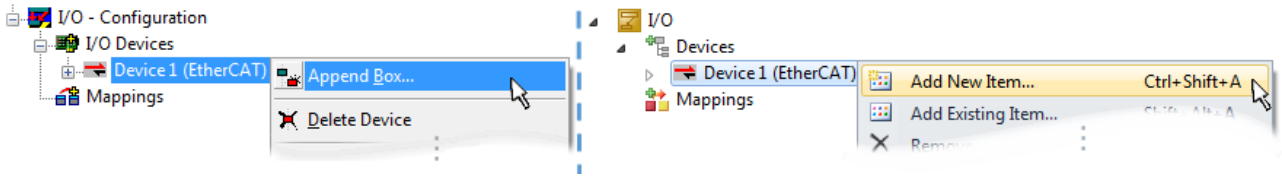



Fig. 50: Appending EtherCAT devices (left: TwinCAT 2; right: TwinCAT 3)

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).

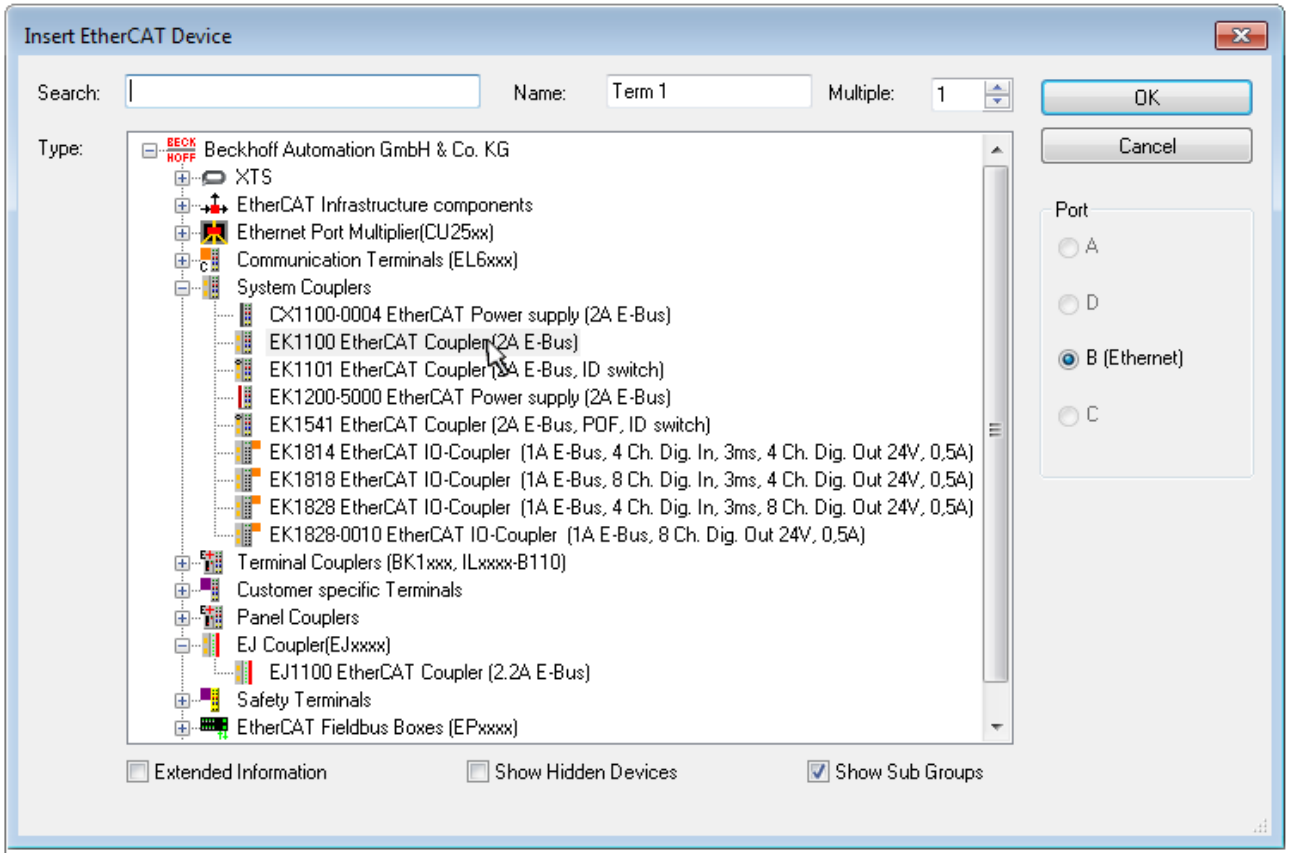



Fig. 51: Selection dialog for new EtherCAT device

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”.

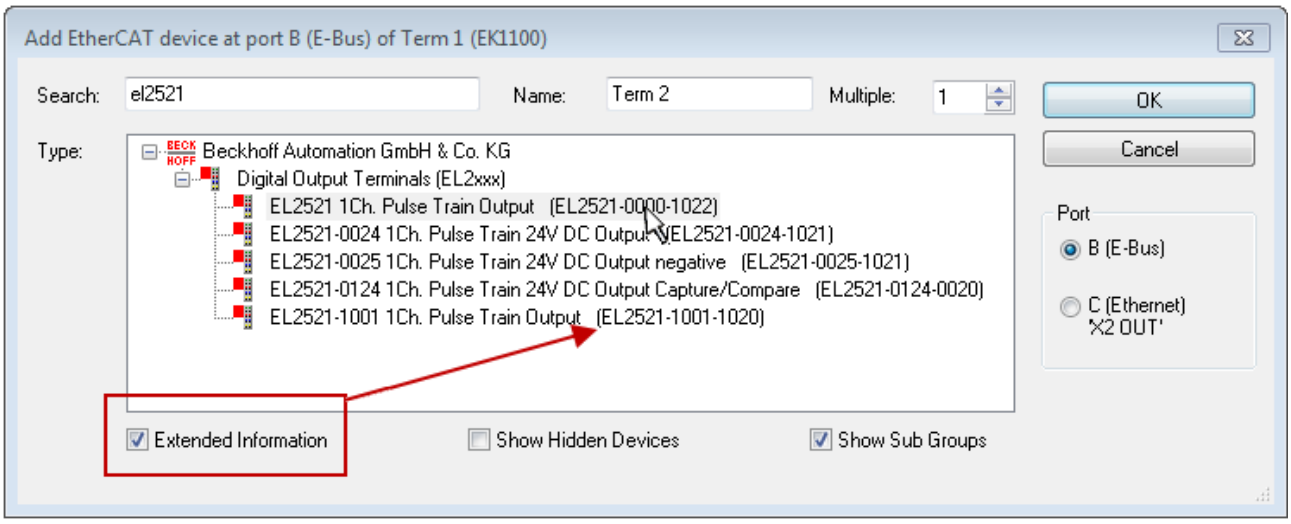



Fig. 52: Display of device revision

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”.

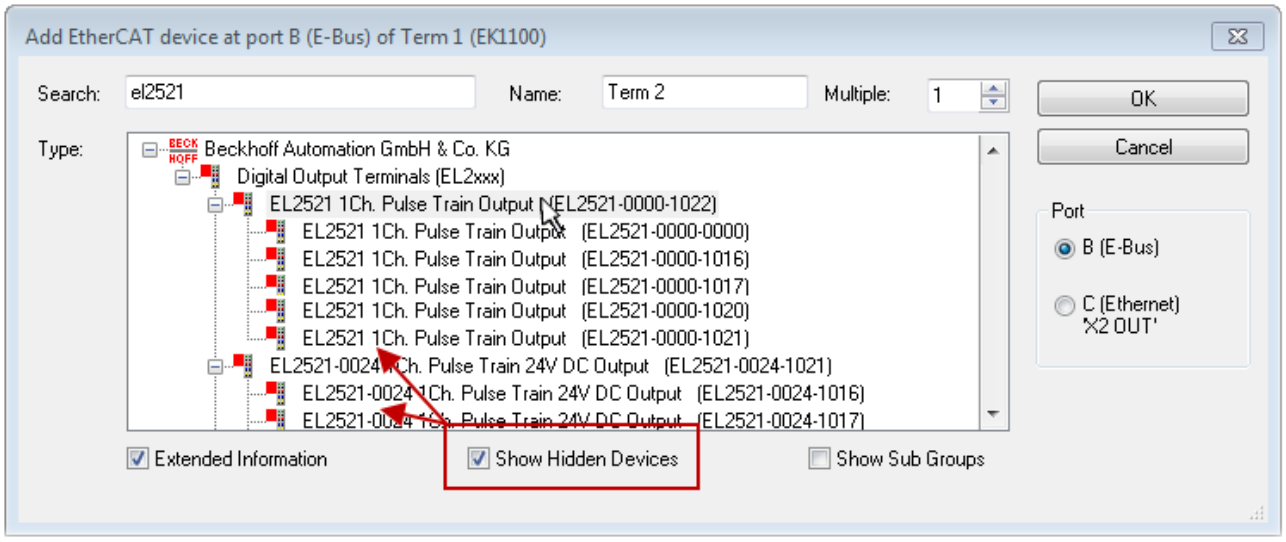



Fig. 53: Display of previous revisions

### Device selection based on revision, compatibility

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

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

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

**Example**

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

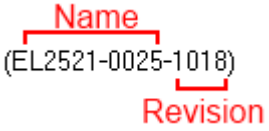



Fig. 54: Name/revision of the terminal

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, ...

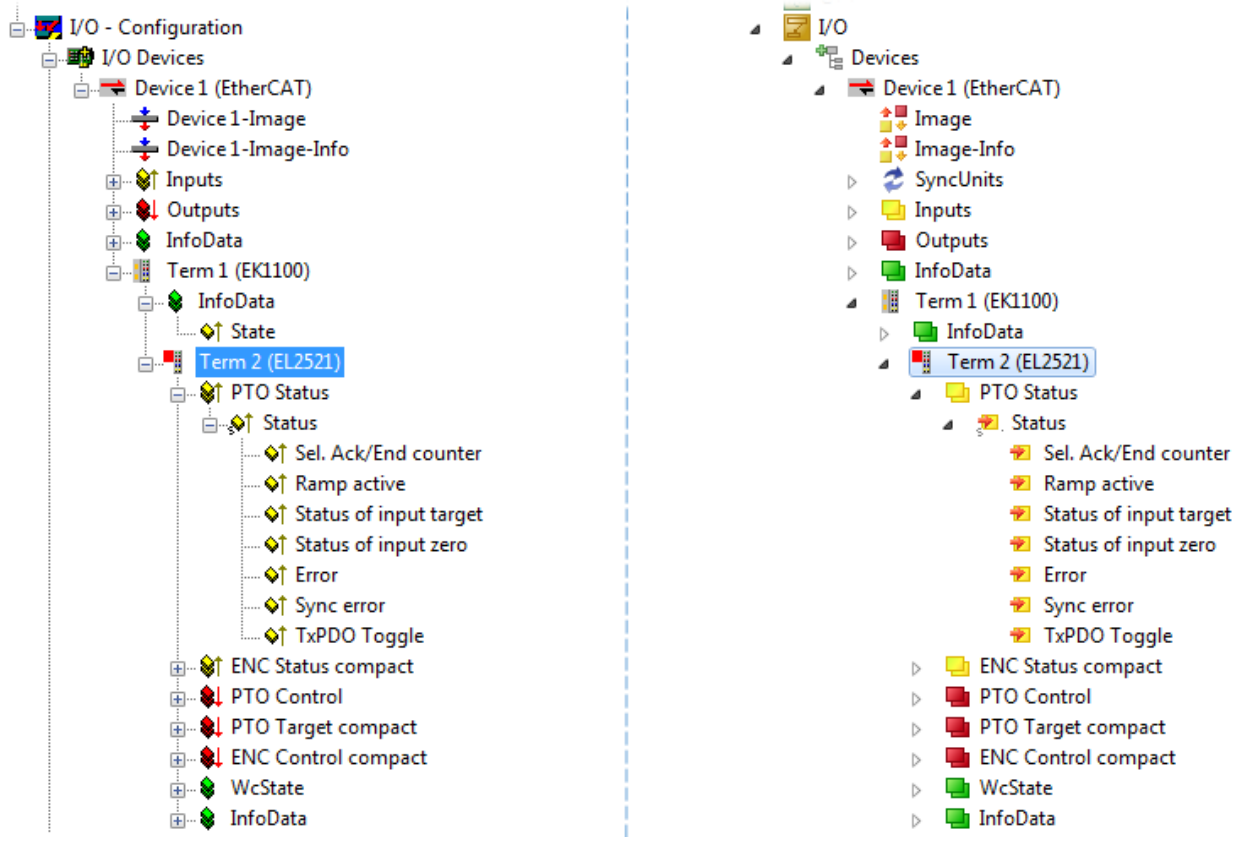



Fig. 55: EtherCAT terminal in the TwinCAT tree (left: TwinCAT 2; right: TwinCAT 3)

## ONLINE configuration creation

### Detecting/scanning of the EtherCAT device

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

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

TwinCAT can be set into this mode:

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

**Online scanning in Config mode**

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

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

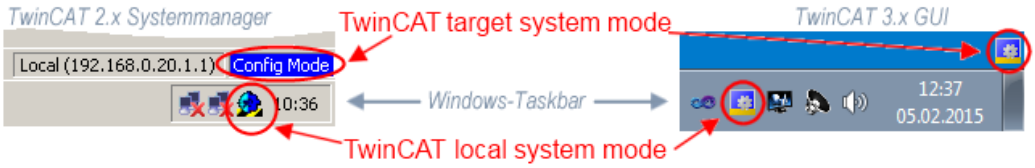



Fig. 56: Differentiation local/target system (left: TwinCAT 2; right: TwinCAT 3)

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

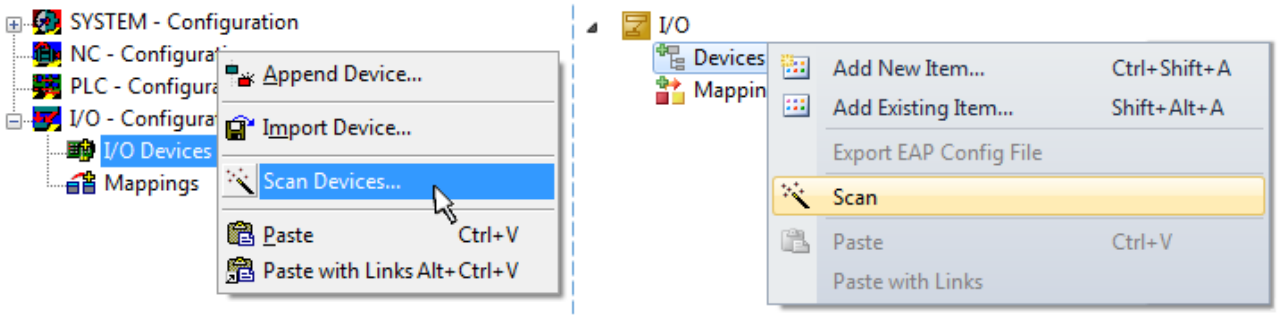



Fig. 57: Scan Devices (left: TwinCAT 2; right: TwinCAT 3)

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

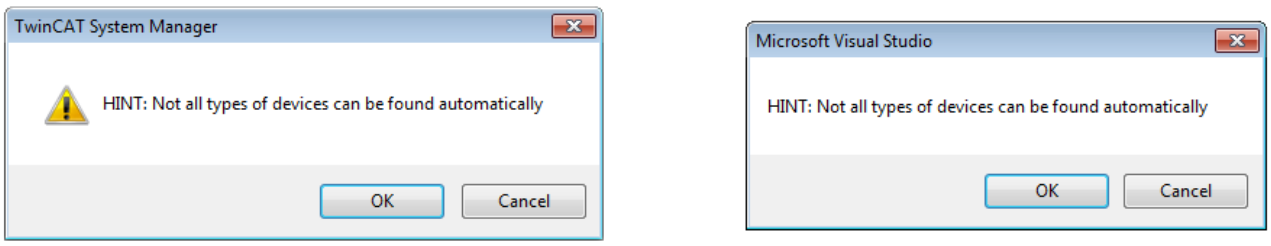



Fig. 58: Note for automatic device scan (left: TwinCAT 2; right: TwinCAT 3)

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” .

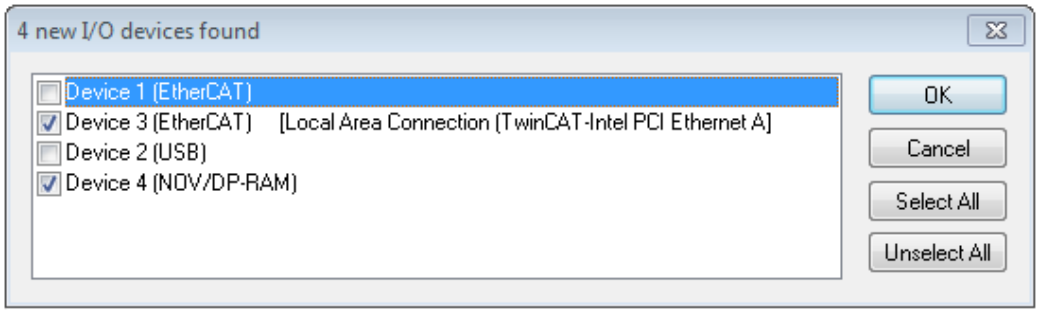



Fig. 59: Detected Ethernet devices

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 [▶ 54].

#### Detecting/Scanning the EtherCAT devices

### Online scan functionality

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

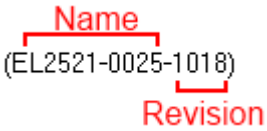



Fig. 60: Example default state

**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 [▶ 74] 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:

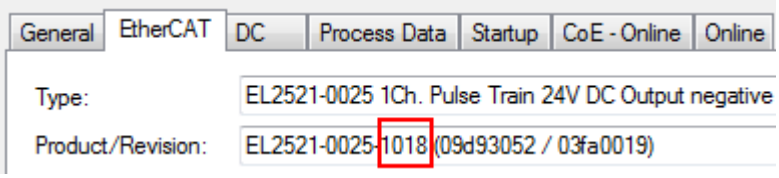



Fig. 61: Installing EthetCAT terminal with revision -1018

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 [▶ 74] 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:

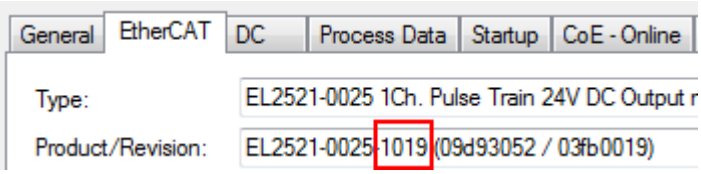



Fig. 62: Detection of EtherCAT terminal with revision -1019

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. Þ if series machine production is established, the scan should only be performed for informative purposes for comparison with a defined initial configuration. Changes are to be made with care!

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

Fig. 63: Scan query after automatic creation of an EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

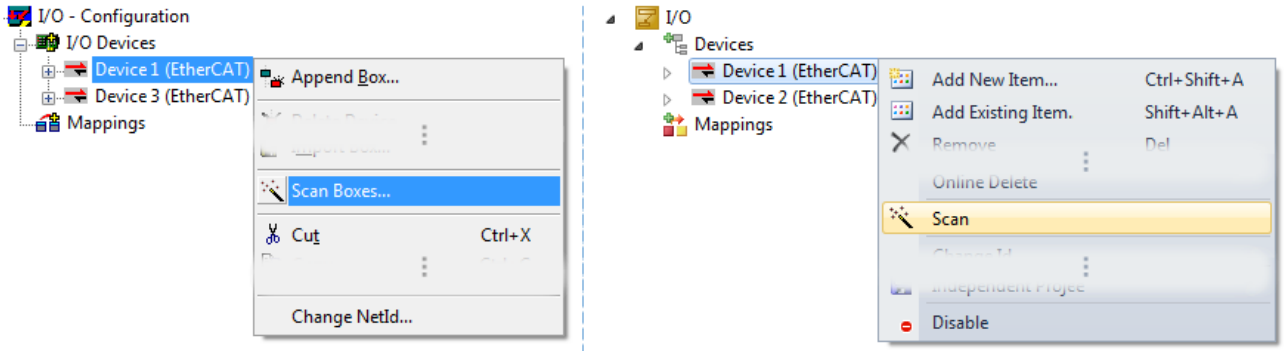



Fig. 64: Manual triggering of a device scan on a specified EtherCAT device (left: TwinCAT 2; right: TwinCAT 3)

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.

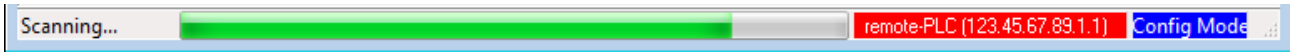



Fig. 65: *Scan progressexemplary by TwinCAT 2*

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

Fig. 66: 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).

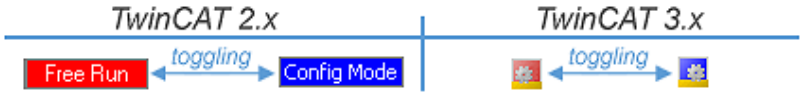



Fig. 67: Displaying of “Free Run” and “Config Mode” toggling right below in the status bar

Fig. 68: 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*.

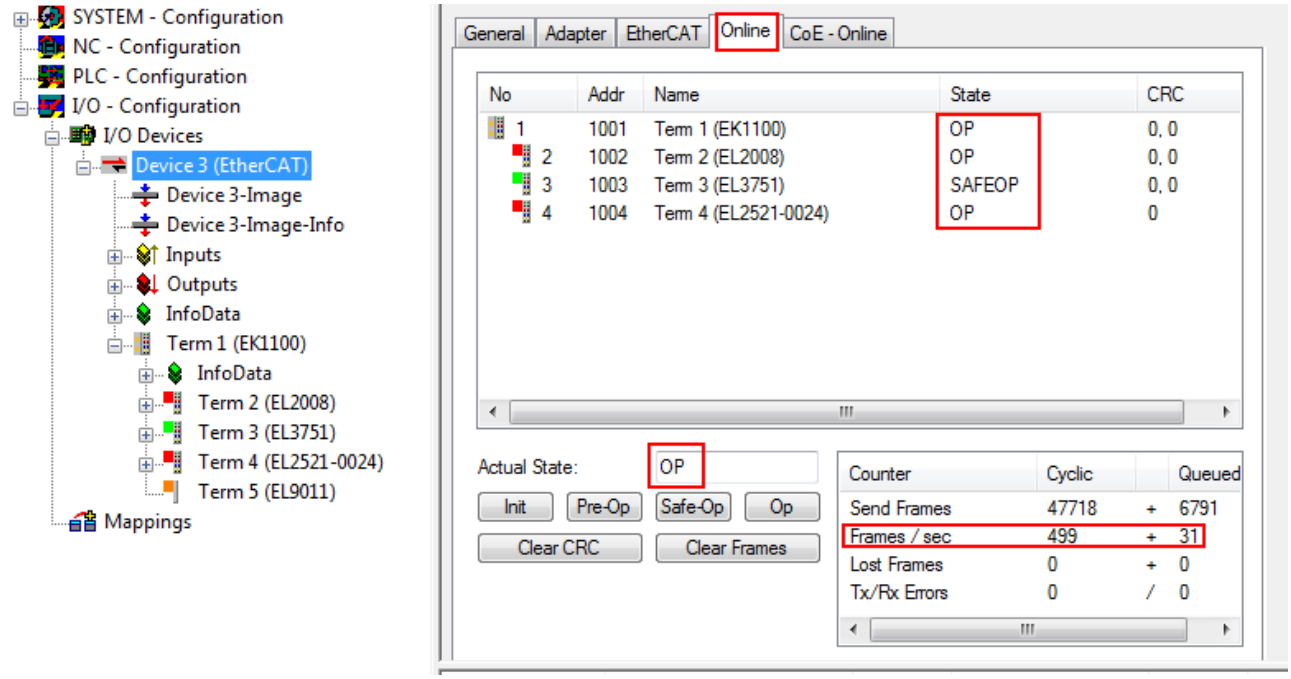



Fig. 69: Online display example

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 [▶ 64].

### 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.

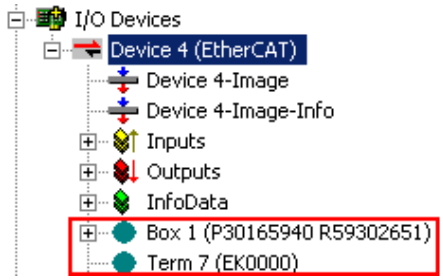



Fig. 70: Faulty identification

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. 71: 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.

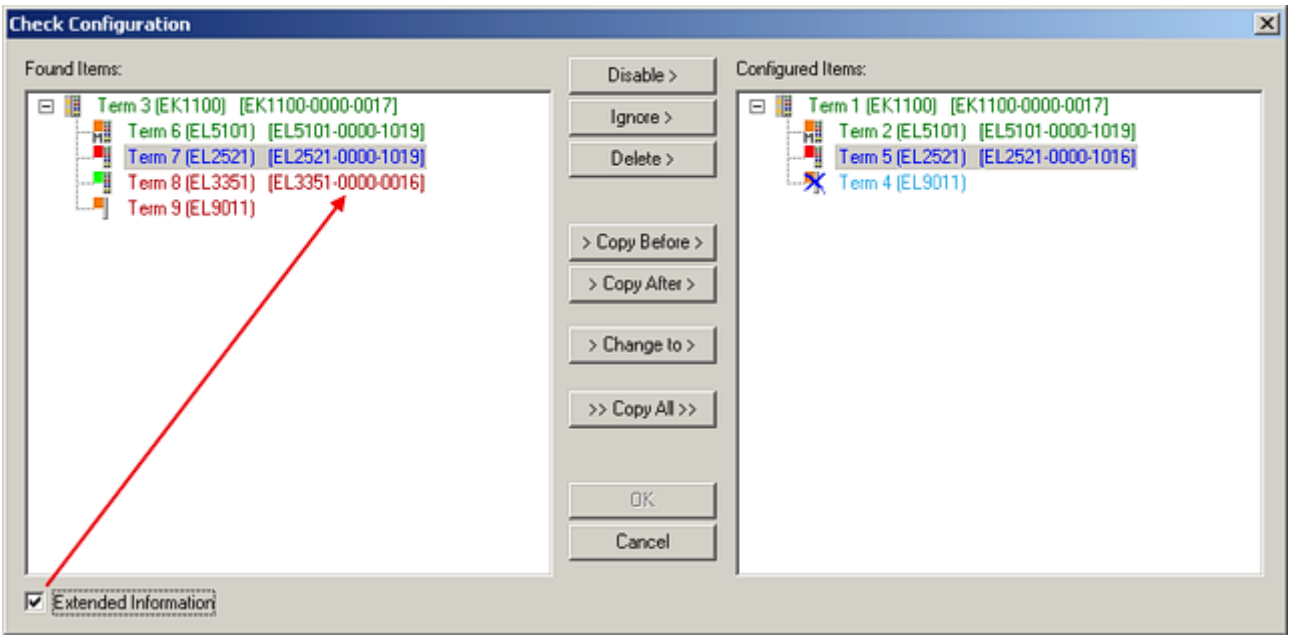



Fig. 72: Correction dialog

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>If the found revision is lower than the configured revision, it is likely that the slave cannot be used. The found device may not support all functions that the master expects based on the higher revision number. |
| light blue | This EtherCAT slave is ignored (“Ignore” button) |
| red | • This EtherCAT slave is not present on the other side.<br>• 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.<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. |

**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.

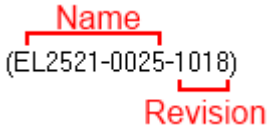



Fig. 73: Name/revision of the terminal

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, ...

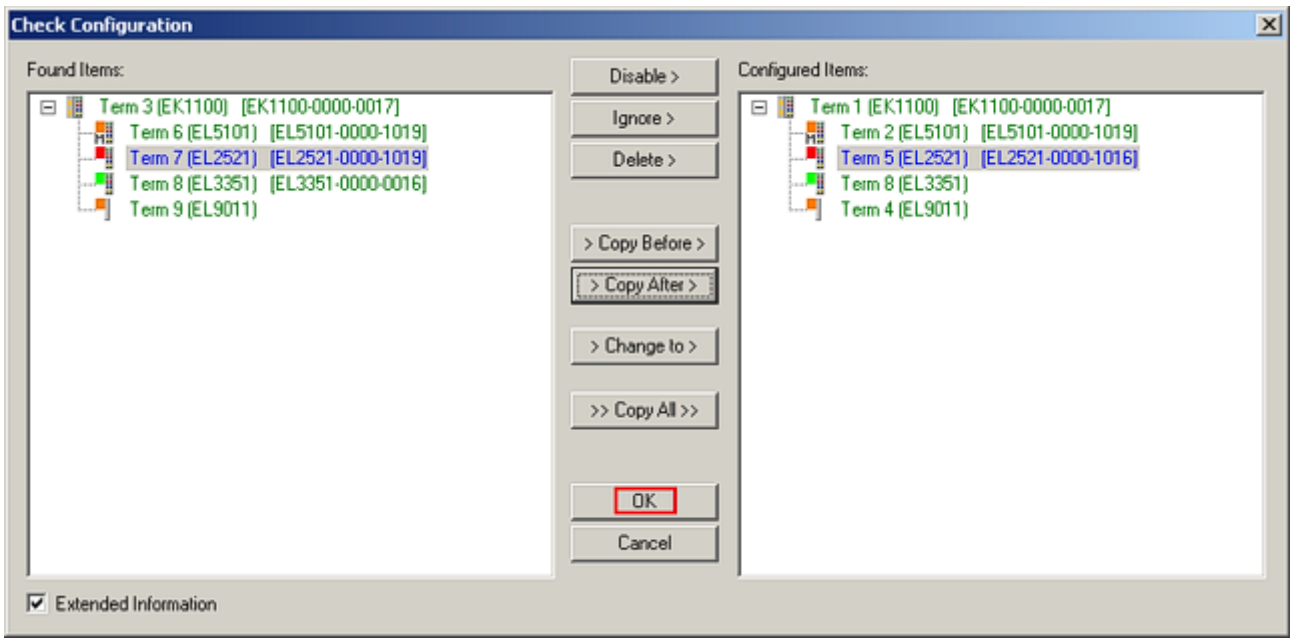



Fig. 74: Correction dialog with modifications

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.

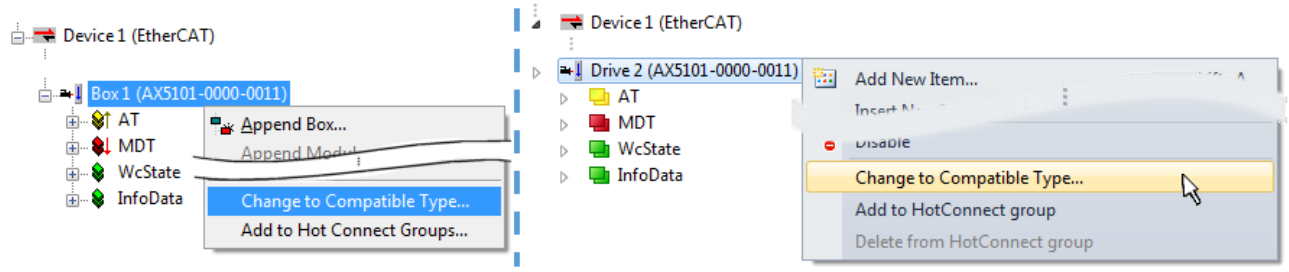



Fig. 75: Dialog “Change to Compatible Type…” (left: TwinCAT 2; right: TwinCAT 3)

This function is preferably to be used on AX5000 devices.

### Change to Alternative Type

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

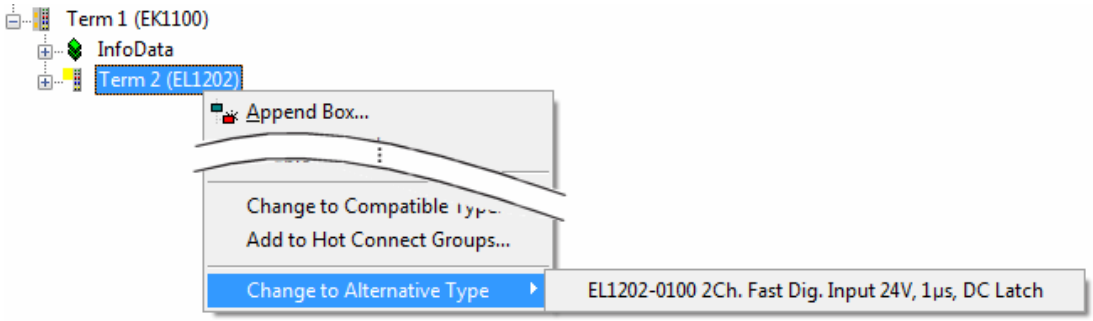



Fig. 76: TwinCAT 2 Dialog Change to Alternative Type

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).

## EtherCAT slave process data settings

The process data transferred by an EtherCAT slave during each cycle (**P**rocess **D**ata **O**bjects, PDOs) 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 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”).

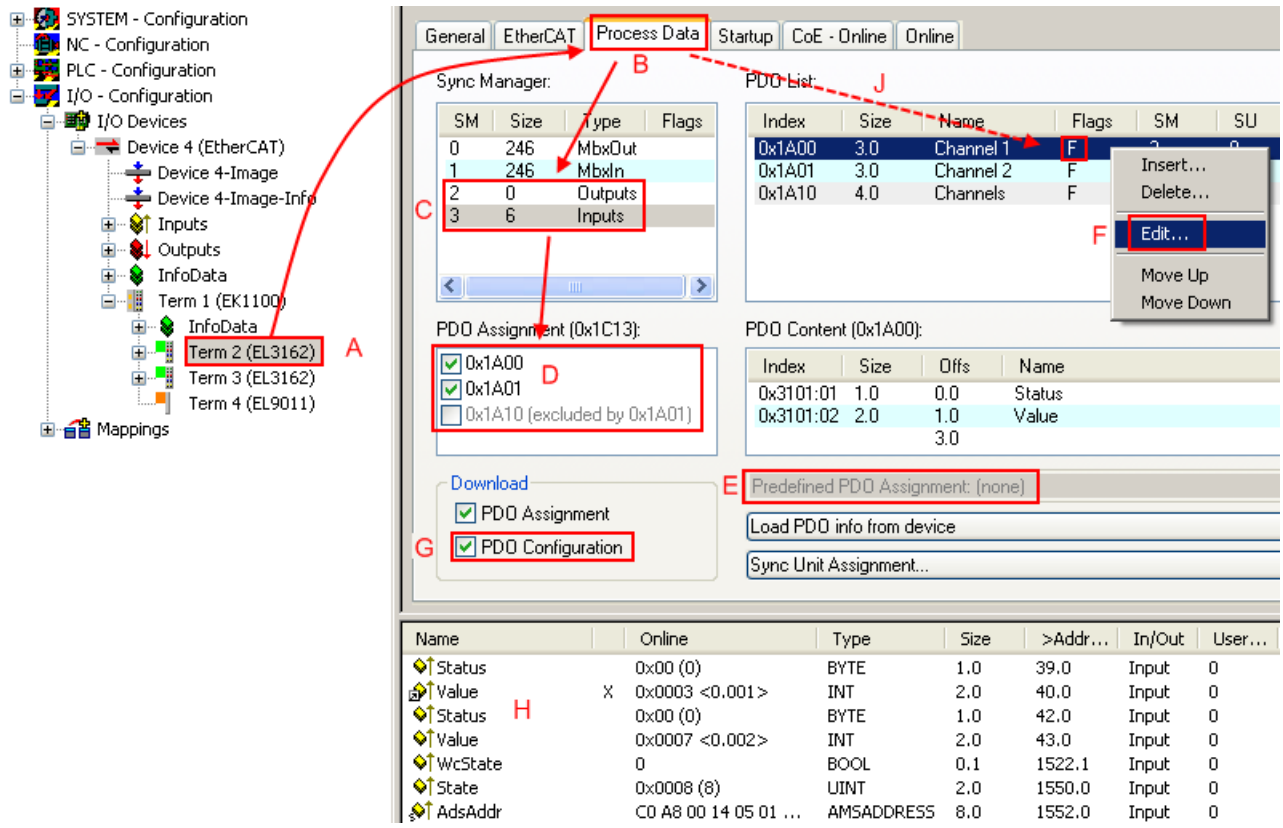



Fig. 77: *Configuring the process data*

### Manual modification of the process data

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

## 5.1.6.1.2 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.

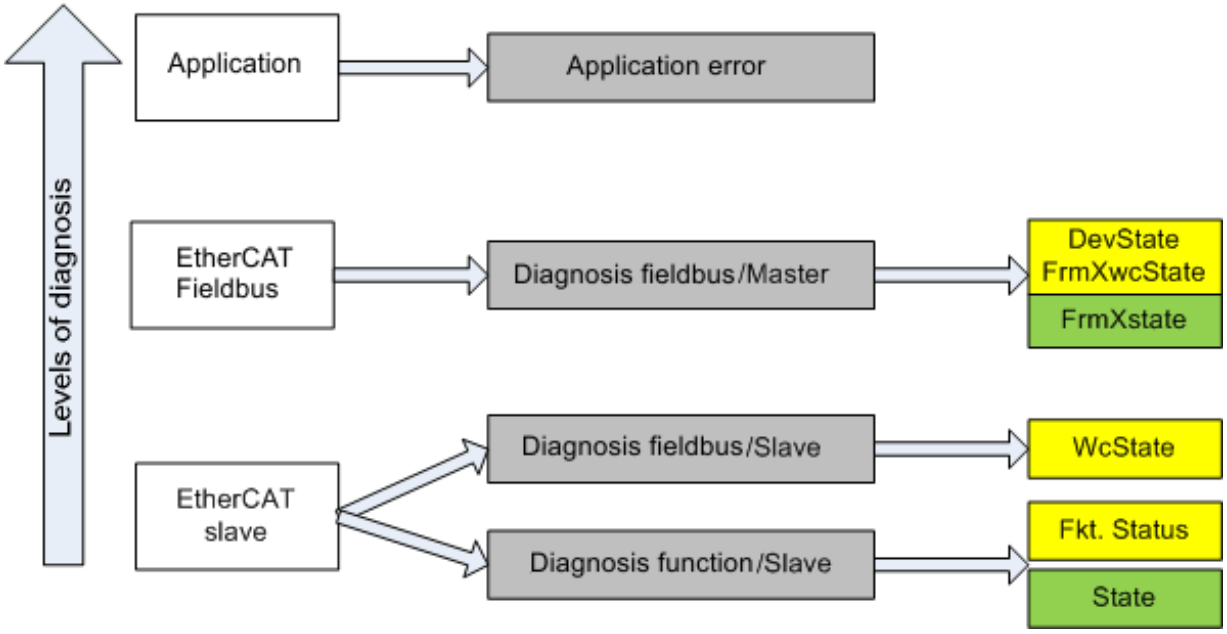



Fig. 78: Selection of the diagnostic information of an EtherCAT Slave

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.

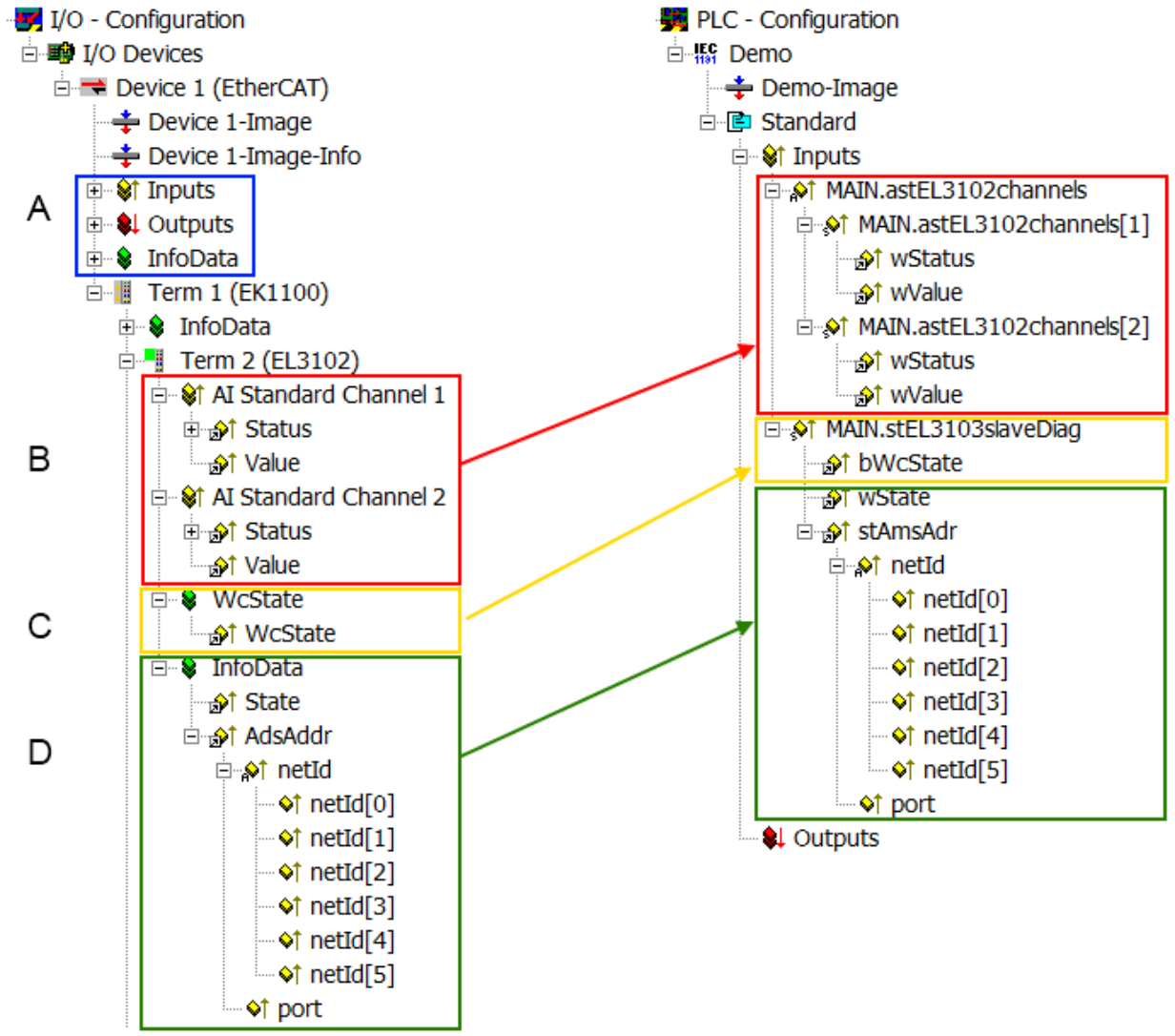



Fig. 79: Basic EtherCAT Slave Diagnosis in the PLC

The following aspects are covered here:

| Code | Function | Implementation | Application/evaluation |
|---|---|---|---|
| A | The EtherCAT Master's diagnostic information<br>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:<br>• CoE in the Master for communication with/through the Slaves<br>• Functions from *TcEtherCAT.lib*<br>• Perform an OnlineScan |
| B | In the example chosen (EL3102) the EL3102 comprises two analogue input channels that transmit a single function status for the most recent cycle. | Status<br>• the bit significations may be found in the device documentation<br>• other devices may supply more information, or none that is typical of a slave | In order for the higher-level PLC task (or corresponding control applications) to be able to rely on correct data, the function status must be evaluated there. Such information is therefore provided with the process data for the most recent cycle. |
| C | For every EtherCAT Slave that has cyclic process data, the Master displays, using what is known as a WorkingCounter, whether the slave is participating successfully and without error in the cyclic exchange of process data. This important, elementary information is therefore provided for the most recent cycle in the System Manager<br>1. at the EtherCAT Slave, and, with identical contents<br>2. as a collective variable at the EtherCAT Master (see Point A)<br>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<br>• is only rarely/never changed, except when the system starts up<br>• is itself determined acyclically (e.g. EtherCAT Status) | State<br>current Status (INIT..OP) of the Slave. The Slave must be in OP (=8) when operating normally.<br>*AdsAddr*<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 *port* (= EtherCAT address). | Information variables for the EtherCAT Master that are updated acyclically. This means that it is possible that in any particular cycle they do not represent the latest possible status. It is therefore possible to read such variables through ADS. |

**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*:

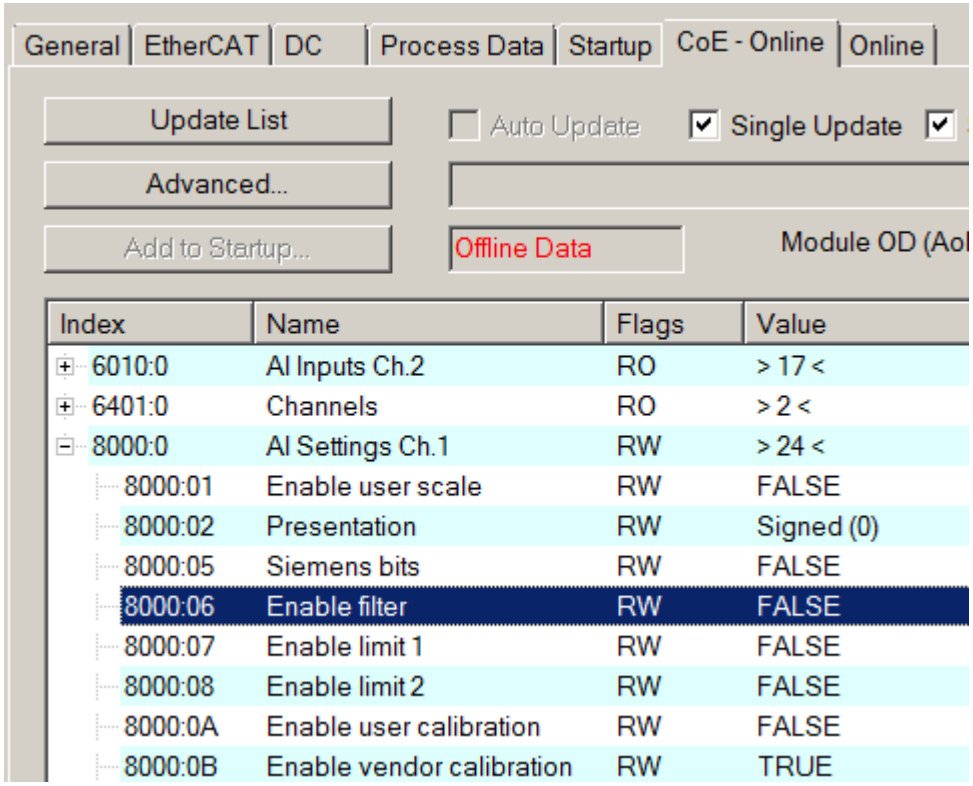



Fig. 80: EL3102, CoE directory

**EtherCAT System Documentation**

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

A few brief extracts:

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

### Commissioning aid in the TwinCAT System Manager

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

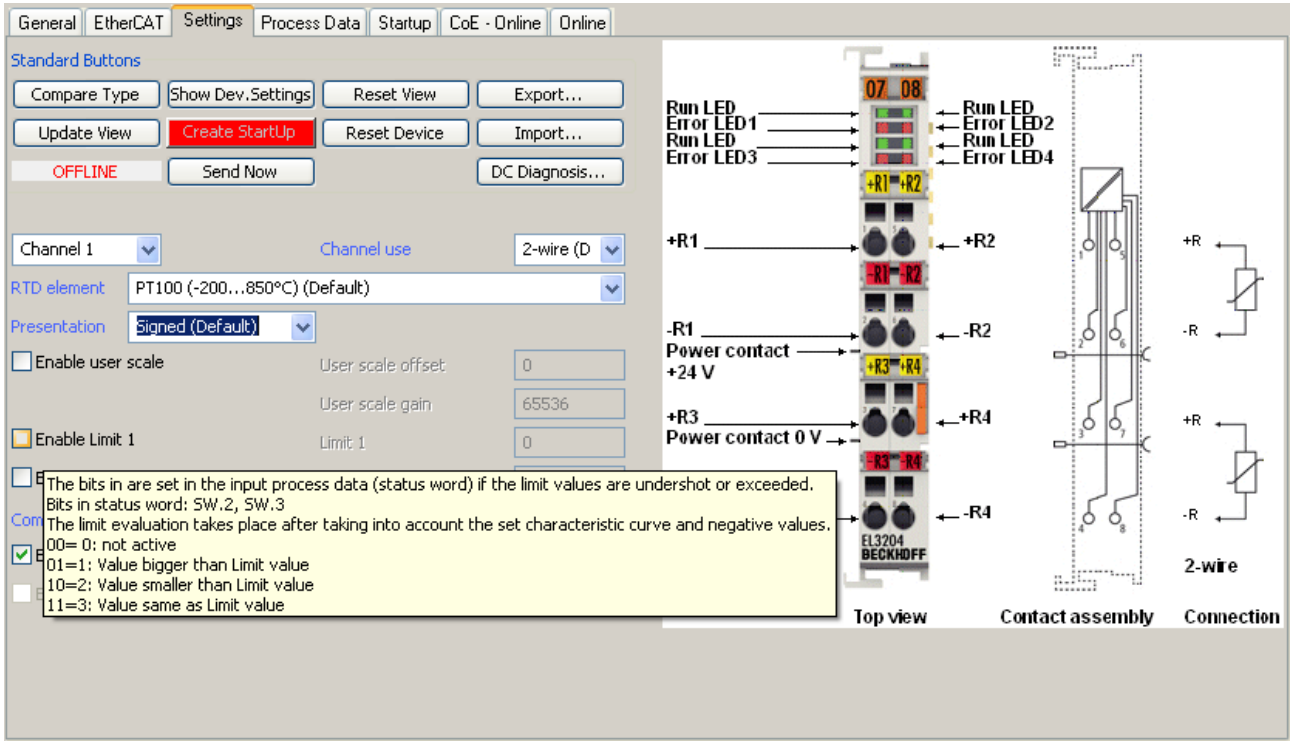



Fig. 81: Example of commissioning aid for a EL3204

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 [▶ 31]" 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.

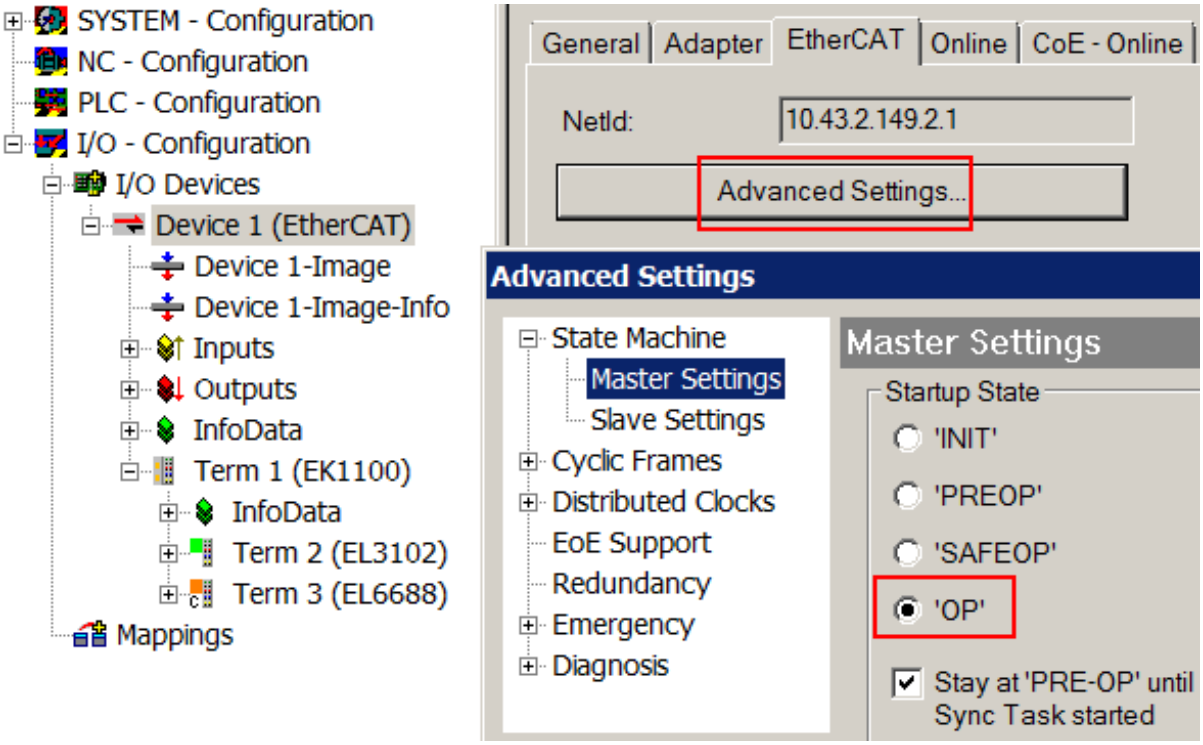



Fig. 82: Default behaviour of the System Manager

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

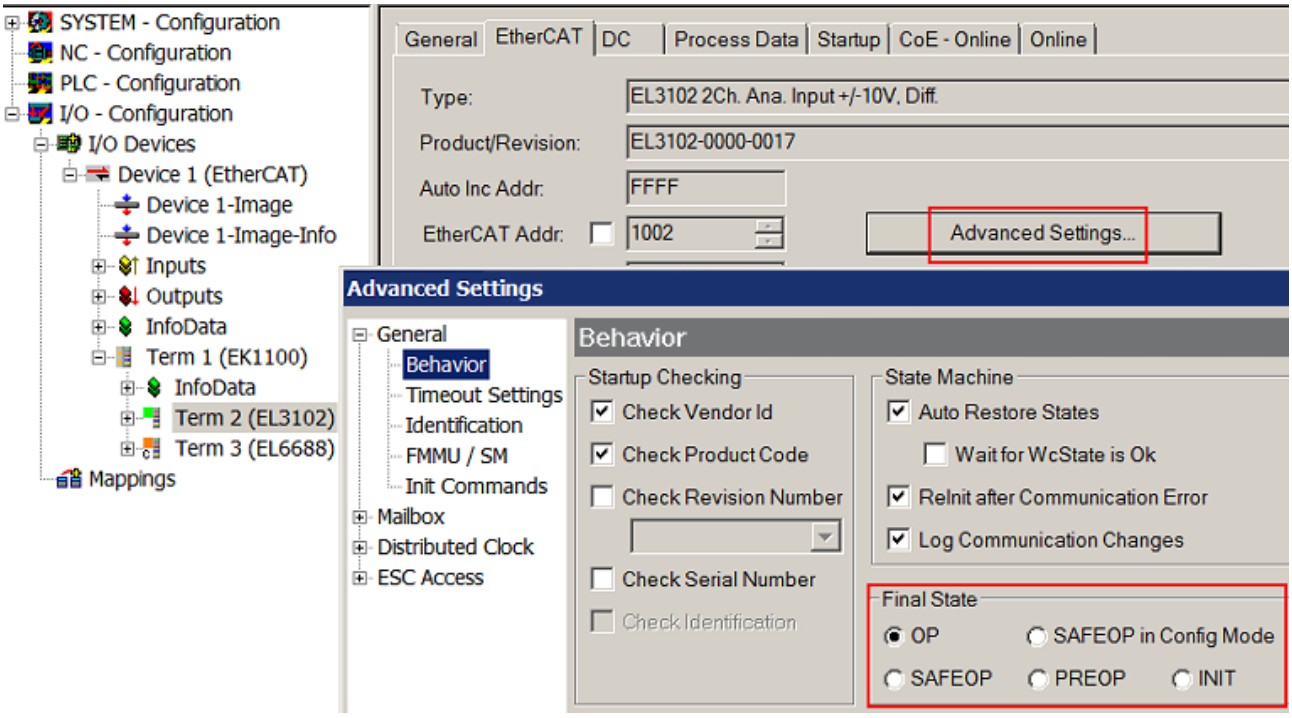



Fig. 83: Default target state in the Slave

### 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.

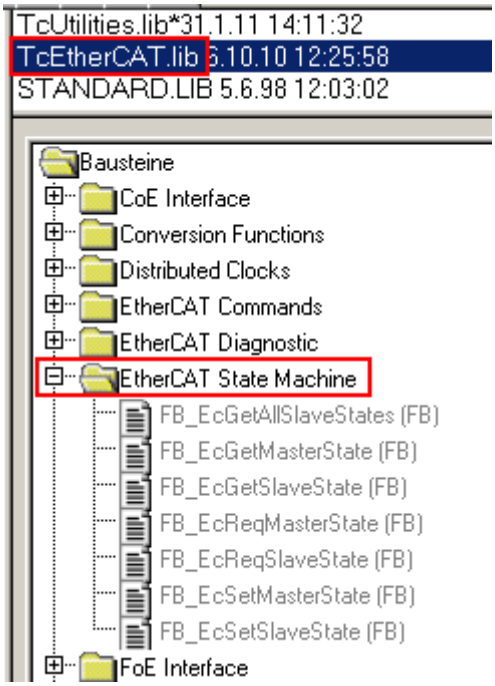
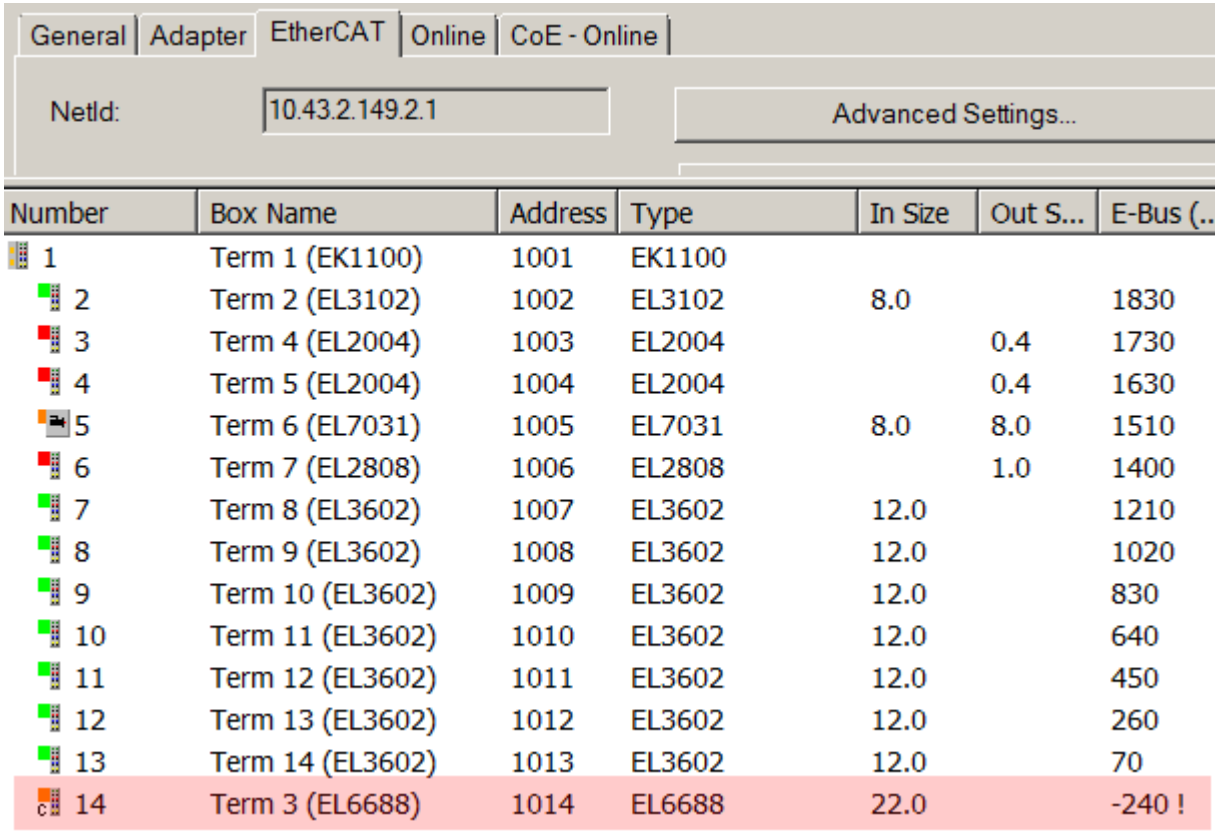



Fig. 84: PLC function blocks

### Note regarding E-Bus current

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

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




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

Fig. 85: 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:

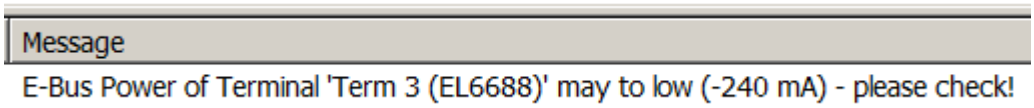



Fig. 86: Warning message for exceeding E-Bus current

**NOTE**

**Caution! Malfunction possible!**

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

### 5.1.6.2 TwinCAT tree "EL6731 device"

#### 5.1.6.2.1 EL6731 tab

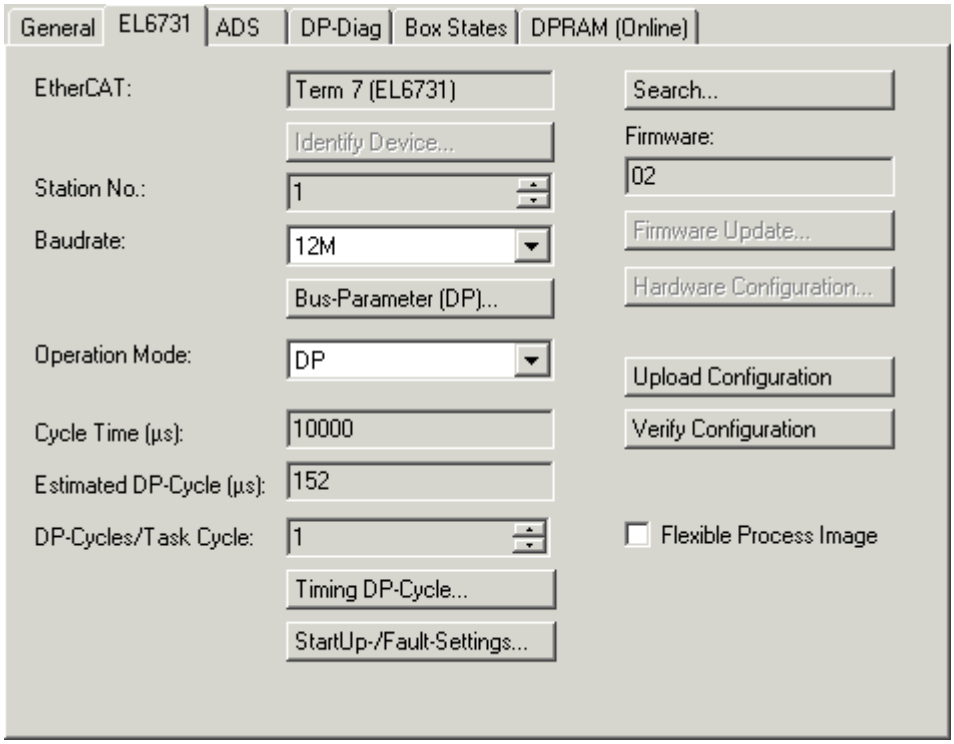



Fig. 87: EL6731 tab

**EtherCAT**
Terminal ID in the terminal network.

**Search**...
This is used to search all EL6731 channels; the required channel can be selected.

**Station no.**
Each Profibus device requires a unique station number - including the master.

**Baud rate**
Set the Profibus baud rate.

**Bus parameters (DP)**...:
The Bus parameters dialog [▶ 88] is selected here.

**Operation mode**
In all three operation modes, the highest-priority task linked to the appropriate device will take control of the PROFIBUS cycle and is therefore synchronized with the DP cycle (see the Synchronization [▶ 42] section). If this task is stopped or reaches a breakpoint, the EL6731 switches to CLEAR mode (slave outputs will assume 0 or safe values) (see the Error reactions [▶ 116] section). All other tasks are served asynchronously via corresponding buffers. If one of these tasks is stopped or reaches a breakpoint, the System Manager will generally display a message saying that the watchdog of the appropriate asynchronous mapping has been activated, and the appropriate outputs are set to 0. For all operation modes, one polling rate per slave can be set (in the Features [▶ 98] tab for the Box). The sequence of the slaves in the PROFIBUS cycle corresponds to the sequence in which they are located in the FC310x/EL6731 device tree. "DP" mode is used for standard DP operation; the operation modes "DP/MC (equidistant)" and "Equidistant (no GC)" are described in section PROFIBUS-MC [▶ 41].

**Cycle time**
Shows the cycle time of the corresponding task with the highest priority.

**Expected DP cycle time**
Shows the expected PROFIBUS cycle time.

**DP cycles/task cycle [only FC310x]**
This can be used to set several DP cycles in a task cycle, in order to obtain as many new inputs as possible (see section Slave prioritization/several DP cycles [▶ 130]).

**Timing DP cycle**...
Shows the timing of the DP cycle; this is particularly advantageous when slave prioritization [▶ 130] is used.

**Startup/fault settings**...

This is used to select the Fault settings dialog [▶ 90]

**Firmware**
Shows the current firmware version of the EL6731.

**Scan Devices**...
This is used to scan the PROFIBUS. All devices that are found are added to the EL6731. In the case of Beckhoff boxes, the configuration is read precisely. In the case of external devices, the corresponding GSD file will be searched.

**Check configuration**... [only FC310x]
This is used to scan the PROFIBUS and compare it with the currently attached boxes; changes are displayed.

### 5.1.6.2.2 Bus parameters dialog

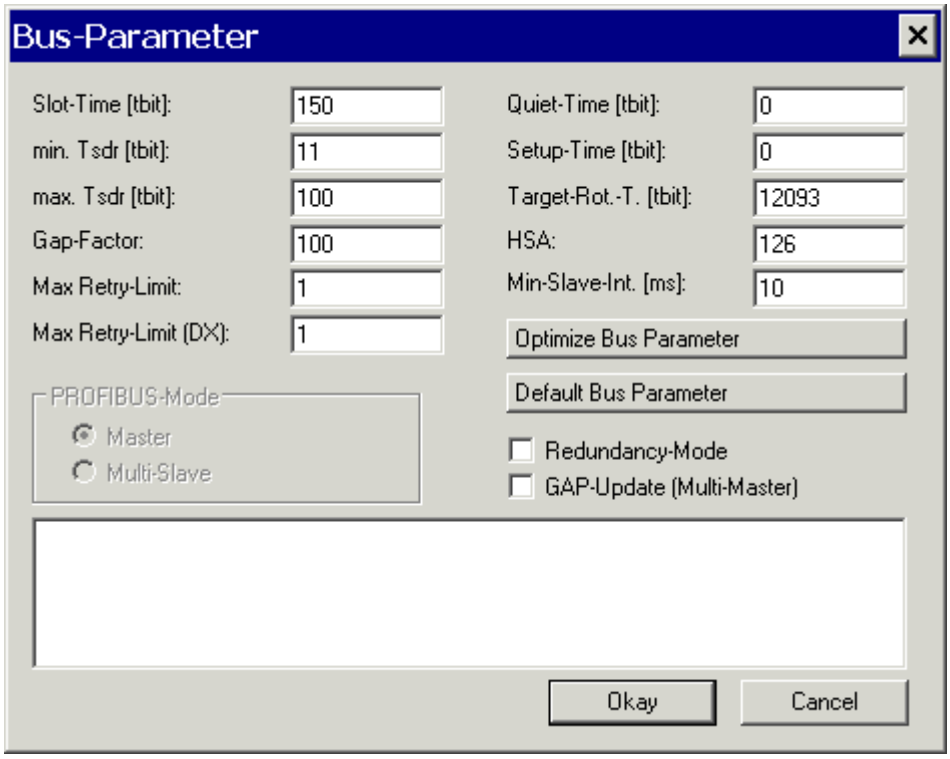



Fig. 88: Bus parameters dialog

**Slot Time**
The Slot Time indicates how long the DP master will wait for a response from the DP slave before it sends either a repetition or the next telegram.

**min. Tsdr**
The minimum Tsdr indicates the minimum length of time for which the DP slave will wait with a response. This time is set for all the DP slaves during the DP start-up (the value range is 11-255 bit periods). The minimum Tsdr must be smaller than the maximum Tsdr.

**max. Tsdr**
The maximum Tsdr indicates the maximum length of time for which the DP slave may wait with a response. This time is set according to the DP slave's GSD file entries. The maximum Tsdr must be smaller than the slot time.

**GAP factor**
The GAP factor determines how often the GAP update will be carried out (assuming it is activated). The time between two GAP updates cycles is **Gap factor x Target rot. t.**

**Max Retry Limit**
The Max Retry Limit specifies how often a telegram should be repeated, if the device addressed does not answer. The minimum value should be 1, so that, in case of an error, there will be at least one repeat for acyclic telegrams (see the Error Reactions [▶ 116] section).

**Max Retry Limit (DX)**
Since the Data_Exchange telegram is repeated cyclically, a value of 0 could be used for the repetition of the Data_Exchange telegram here, in order to keep the cycle relatively constant in equidistant mode, even if there is no response from a device. However, in this case it would make sense to set the Features [▶ 98] tab for the box such that lack of response of the slave would not lead to DATA EXCH being exited. The fact that a device has not responded is apparent from DpState [▶ 122], which would not be equal 0 for one cycle (see the Error Reactions [▶ 116] section).

**HSA**
The HSA specifies the highest active address up to which the GAP update is carried out (assuming it is active).

**Min. Slave-Int.**
The MinSlaveInterval indicates the minimum cycle time with which the DP StartUp telegrams are sent the DP slaves (it is determined from the settings found in the GSD file).

**Redundancy mode**
Redundancy mode can be set here for the DP master. In that case all that it does is to listen to the bus (see the Master Redundancy section).

**GAP update**
The GAP update asks all stations up to HSA at intervals to confirm their presence. It can be en/disabled. The GAP update is relevant only for multi-master operation. In single master operation it increases PROFIBUS cycle jitter and is therefore switched off by default.

**Optimize bus parameters**
This is used to set the optimized bus parameters.

**Default bus parameters**
This is used to set the default bus parameters.

### 5.1.6.2.3 Startup/fault settings dialog

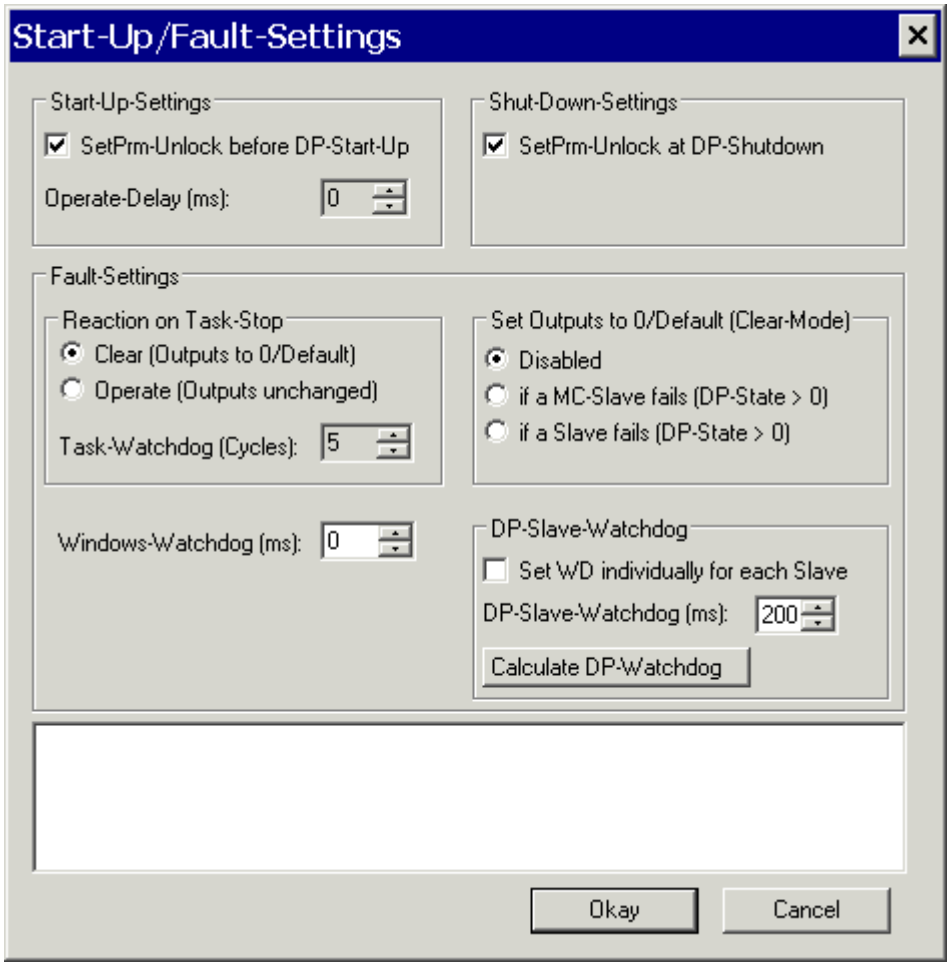



Fig. 89: Startup/fault settings dialog

**SetPrm unlock before DP start-up**
Normally, during DP start-up, the DP master removes the cyclic connections, so that the DP slave can always recognize that the DP master has restarted. In redundancy mode, however, it may be specifically desirable for the DP slave to remain unaware of this, because the switch-over from the primary master to the redundant master should not have any interactions for the DP slave (see the Master Redundancy section).

**SetPrm unlock at DP shutdown**
Normally, during DP shut-down, the DP master removes the cyclic connections, so that the DP slave can always recognize that the DP master has stopped. In redundancy mode, however, it may be specifically desirable for the DP slave to remain unaware of this, because the switch-over from the primary master to the redundant master should not have any interactions for the DP slave (see the Master Redundancy section).

**Operate delay**
The DP master changes automatically, observing the Auto-Clear-Mode, into the operate state when the task is started. The transition from Clear to Operate can be delayed or with the Operate delay time. In the Clear state, all the outputs are set to 0 (if the DP slave does not support Fail_Safe values) or to the Fail_Safe value (if the DP slave supports Fail_Safe), whereas in the Operate state the outputs have the values specified by the task.

**Response to task STOP**
It is possible to specify here whether the DP master should set the outputs to 0 when reaching a PLC stop or breakpoint, or should leave them unchanged (see the Fault Reactions [▶ 116] section).

**Task watchdog (EtherCAT watchdog)**
The DP master changes automatically into the clear mode (the outputs of the slaves are set either to 0 or to the fail-safe value) when it ceases to receive an interrupt from the associated task (e.g. a PLC breakpoint

has been reached, or the system has crashed). It is possible to specify here how many missing tasks cycles can be tolerated before the master switches into the clear mode. This setting is independent of the setting in the **Clear mode**.

**Clear mode**
Here you can specify that the master should switch to or remain in "Clear" state, as long as at least one MC slave (setting: "if an MC slave fails") or any slave (setting: "if a slave fails") does not respond correctly (i.e. DpState [▶ 122] is not equal 0) (see section Error Reactions [▶ 116]).

**Windows watchdog (only FC310x)**
Not relevant for EL6731.

**Set WD individually for each slave**
Here you can select whether the WD should be set individually for each slave (on the Profibus [▶ 97] tab of the box).

**DP watchdog time**
If the checkbox "Set WD individually for each slave" is not ticked, the DP watchdog can be set here to a uniform value for all slaves.

**Calculate DP slave watchdog time**
This is used to set the DP watchdog time for all DP slaves to a suitable value.

### 5.1.6.2.4 MC tab

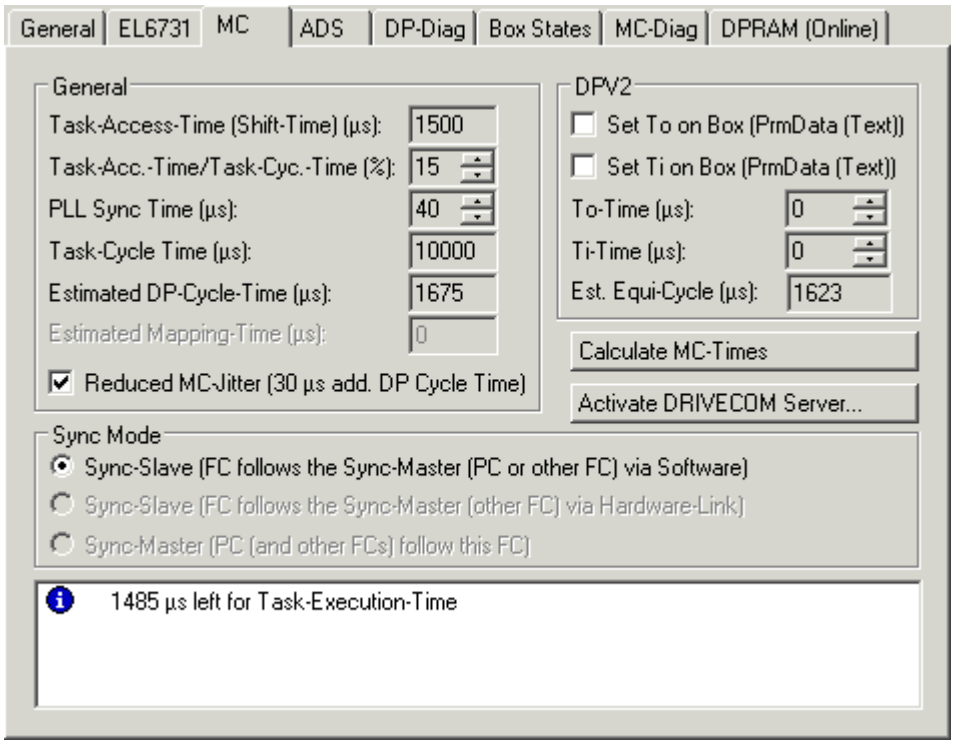



Fig. 90: MC tab

**Task access time (shift time)**
Not relevant for EL6731.

**Task access time/task cycle time**
Not relevant for EL6731.

**PLL-Sync-Time**
Not relevant for EL6731.

**Task cycle time**
Displays the cycle time of the corresponding highest priority task.

**Estimated DP cycle time**
Displays the expected PROFIBUS cycle time.

**Set To on box**
The To time can be set individually for each slave on the Prm data (text) [▶ 101] tab of the box.

**Set Ti on box**
The Ti time can be set individually for each slave on the Prm data (text) [▶ 101] tab of the box.

**To time**
If the checkbox **Set To on box** is not ticked, the To time can be set to the same value for all slaves.

**Ti time**
If the checkbox **Set Ti on box** is not ticked, the Ti time can be set to the same value for all slaves.

**Estimated equi cycle time**
Shows the DPV2 part of the expected PROFIBUS cycle time.

**Reduced MC jitter (optional for FC/CX, selected by default for EL)**
The FC/CX cycle is controlled through a local timer, which is synchronized with TwinCAT. In order to compensate the interrupt latency time (jitter of the DP cycle approx. 960 ns without compensation, approx. 320 ns with compensation), the interrupt routine always waits for a certain timer value, before the DP cycle is started. This extends the cycle by approx. 30 µs.

**Calculate MC times**
This button can be used to calculate all DPV2 times automatically.

**Sync mode**
Not relevant for EL6731; synchronization is controlled via the distributed clocks and therefore corresponds to the Sync master setting in the FC310x.

**Activate DRIVECOM server**
An XML file is generated for the DRIVECOM server (OPC server), through which the SimoCon U tool can access the connected 611U drives via PROFIBUS.

### 5.1.6.2.5 ADS tab

The EL6731 is an ADS device with its own net ID, which can be changed here. All ADS services (diagnosis, non-cyclical communication) going to the EL6731 must address this net ID.

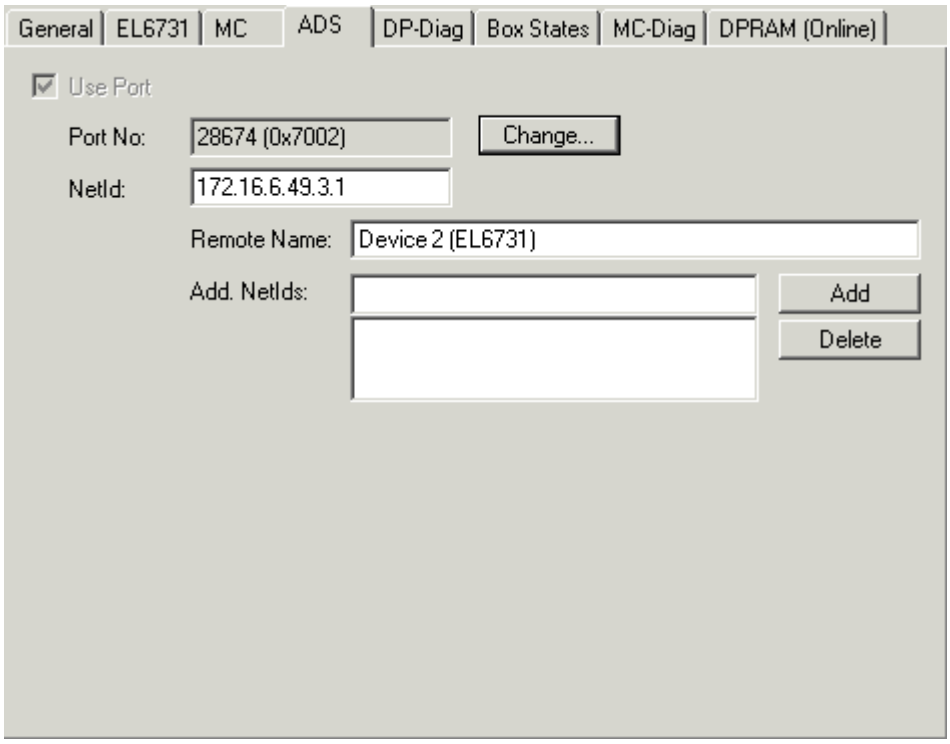



Fig. 91: ADS tab

### 5.1.6.2.6 DP Diag tab

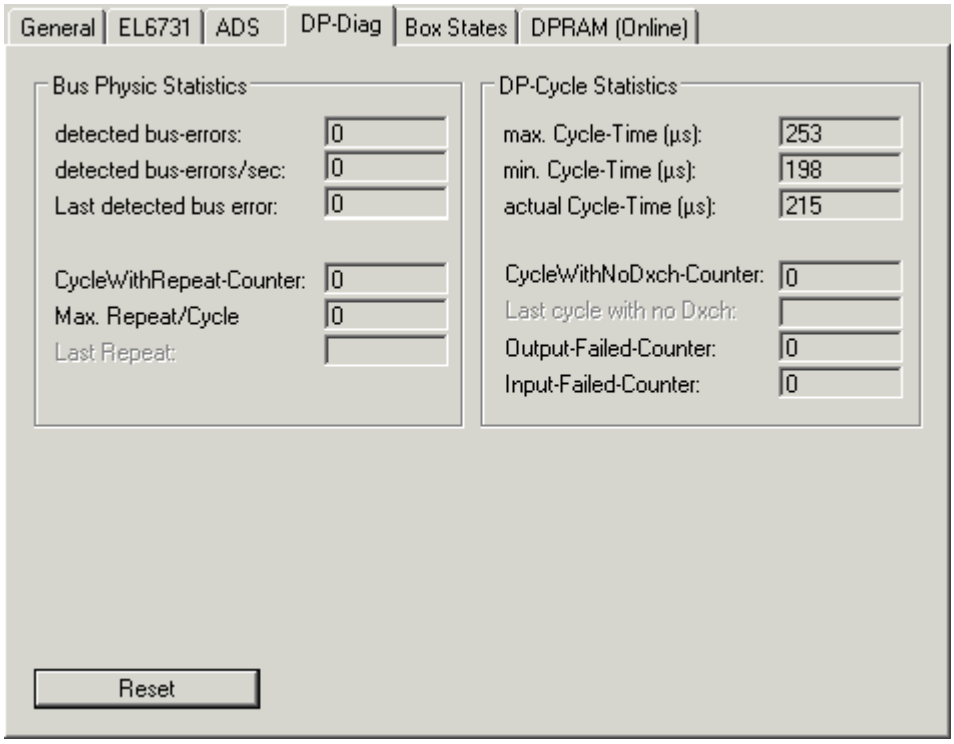



Fig. 92: DP Diag tab

Here, bus cabling problems and DP cycle times are displayed:

**Detected bus errors**
Here, the number of detected bus errors is displayed. If this counter is not equal 0, the cabling should be checked (provided that no PROFIBUS connectors were pulled or inserted - usually there are short bus disturbances during pulling or inserting of PROFIBUS connectors).

**CycleWithRepeatCounter**
Here, the number of PROFIBUS cycles is displayed, in which a telegram was repeated at least once. Repetitions indicate that there is something wrong with the bus physics.

**Max. repeat/cycle**
Here, the maximum number of repetitions within a cycle is displayed.

**min./max./actual cycle time**
Here, the minimum, maximum and current DP cycle time is displayed. Only those cycles are considered, during which all slaves participated in the data exchange and no repetitions occurred.

**CycleWithNoDxch counter**
Increments if not all slaves participate in the data exchange (i.e. have a DpState not equal 0).

**Output-Failed-Cycle-Counter**
Increments if the DP cycle was not complete before the next EtherCAT cycle (EL) started and all slaves are in data exchange (i.e. they have a DpState of 0).

### 5.1.6.2.7 Box states tab

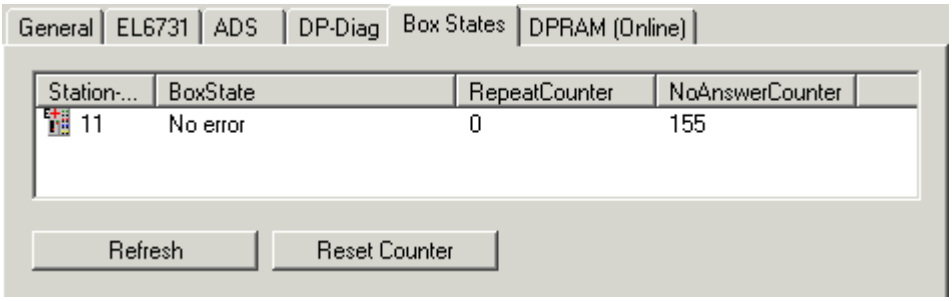



Fig. 93: Box states tab

Here, an overview of all current box states, the Repeat counter (increments for each telegram repeat to the slave) and the NoAnswer counter (increments every time the slave fails to answer) is displayed.

### 5.1.6.2.8 MC Diag tab

In online mode, the "MC Diag" tab shows various equidistant monitoring parameters:

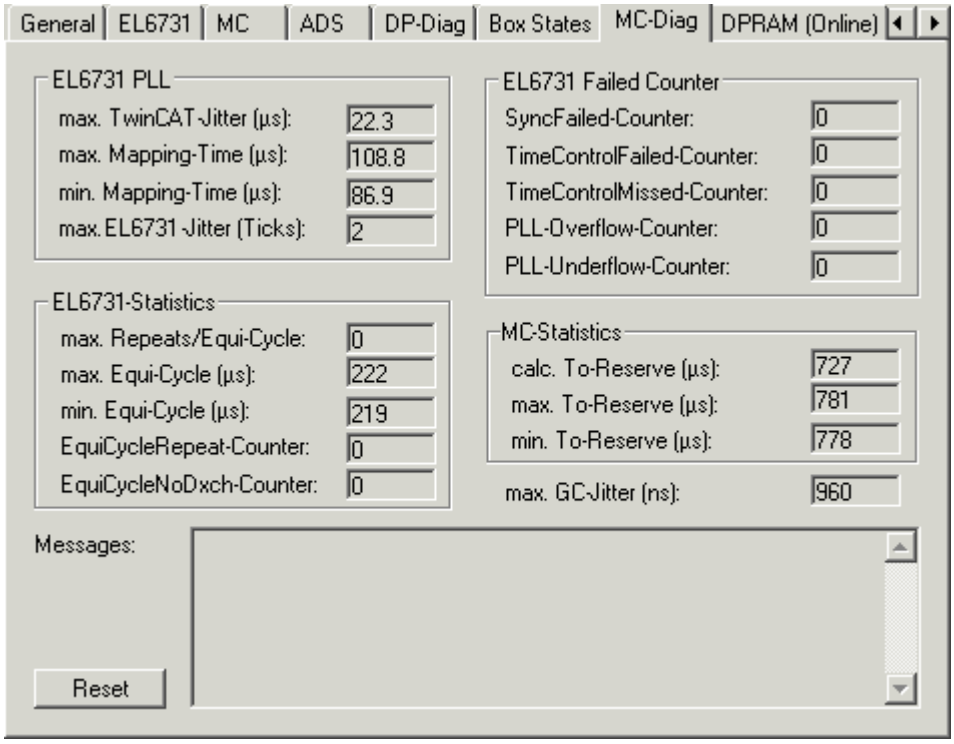



Fig. 94: MC Diag tab

#### EL6731 PLL

No longer relevant for EL6731.

#### EL6731 Statistics

Since the PROFIBUS MC slaves should always be added to the EL6731 first, they are always addressed before the DP slaves in the DP cycle. This part of the DP cycle is referred to as Equi cycle below. If the Equi cycle exceeds the To time of the MC slaves, the last MC slaves at the EL6731 will usually get a synchronization error (error 597 or 598 for Simodrive 611U).

**max. repeats/equi cycle**

This will show the maximum number of repeated Data_Exchange telegrams during an Equi-Cycle. This will extend the Equi-Cycle, usually no repetitions should occur (unless a bus plug was pulled or an MC slave was switched off).

**min./max. equi cycle**
Here, the minimum or maximum Equi-Cycle Time is displayed.

**Equi cycle repeat counter**
Here, the number of telegram repetitions within the Equi-Cycle is displayed.

**Equi Cycle NoDxch Counter**

Here, the number of occurrences is displayed, for which not all MC slaves were in data exchange during an Equi-Cycle.

### EL6731 Failed Counter

**Sync-Failed-Counter**
This counter will increment if TwinCAT task and DP cycle are not synchronized with each other. This may happen during start-up of the TwinCAT system, after which this counter should no longer increment. If the associated (NC) task does not have the highest priority, this counter can also increment. This should be avoided.

**Time-Control-Failed-Counter**
This counter will increment if the PROFIBUS was not free at the time of the DP cycle start. Possible causes are bus faults, non-existent device, a second master or a safety time that is too small.

**PLL-Overflow/Underflow Counter**
Not relevant for EL6731.

### MC Statistics

For each MC slave the To-time specifies when, relative to the DP cycle start, the slave should accept the outputs received from the master. The MC slaves can be synchronized with each other, if the same To value is set for all MC slaves. However, this value must be equal or greater than the Equi-Cycle Time plus a safety margin of approx. 200 µs. The To-time is calculated for all MC slaves via the button "Calc. Equi-Times" (see above).

**calc. To-Reserve:** This contains the calculated To-reserve (To-time - Equi-Cycle Time)

**min./max. To-Reserve**: Here, the min. or max. To-reserve is measured.

### max. GC jitter

Here, the maximum jitter of the DP cycle is measured (GC for global control telegram, which is always sent at the start of a cycle). During start-up, the jitter may be somewhat greater, in the steady state it should not exceed 1 µs (for Sync mode "Sync Master") or 2 µs (for Sync mode "Disabled").

### 5.1.6.3 TwinCAT "Box" tree

#### 5.1.6.3.1 Profibus tab

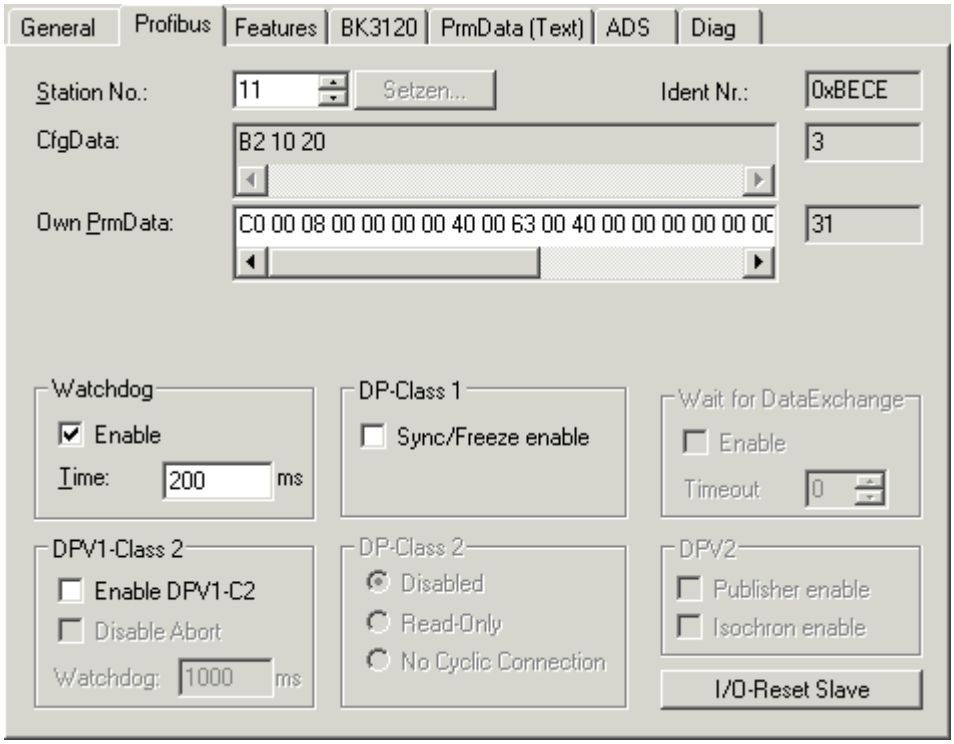



Fig. 95: Profibus tab

**Station no.**
Here, the PROFIBUS station address must be set for each slave. For some slaves, the station address cannot be set in the hardware, but only via the SetSlaveAddress service. In this case, the button "Set.." should be pressed. This will open a dialog, through which transmission of a SetSlaveAddress telegram can be triggered.

**Watchdog**
Activates the DP watchdog. If the slave does not receive a DP telegram for the duration of the watchdog time with the watchdog switched on, it will automatically exit the data exchange. The minimum watchdog time to be set depends on the DP cycle time, and should be larger than the value calculated by the following formula: **Estimated cycle time x 10**

For particularly critical outputs it is possible to set a DP watchdog time down to as little as 2 ms for DP slaves that support a watchdog base time of 1 ms (namely all Beckhoff slaves with the exception of the BK3000 and BK3100, and any devices from other manufacturers whose GSD file contains the entry "WD_Base_1ms_supp = 1"). The DP watchdog time should, however, be at least twice as long as the greater of the **Cycle time** and the **Estimated cycle time** (see the master's tab).

**Ident no.**
Here, the Ident number from the GSD file is displayed.

**Own PrmData**
Allows editing of the Profibus-specific parameter data. The values of the current parameter data are also displayed. The PrmData can usually be set as text (-> PrmData (text)) or for Beckhoff DP slaves partly via the ”Beckhoff" tab.

**CfgData**
The current configuration data (resulting from the attached modules or terminals) as well as their length is displayed.

**Sync/Freeze**
In operation mode DP/MC (equidistant) of the master, slaves can be operated with Sync and Freeze [▶ 43].

**DPV1-Class 2**
With FC310x/EL6731, a DPV1 class 2 connection to a DPV1 slave can be activated. The Timeout parameter is used to set the connection monitoring time for the class 2 connection (see section DPV1 [▶ 47]).

**DP Class 2**
If the DP slave is in data exchange with another master but should nevertheless be addressed acyclically by TwinCAT, select "No cycl. connection" under DP class 2. The "ReadOnly" setting is not supported at present.

**I/O-ResetSlave**
With this button, provided TwinCAT has been started, cyclic data exchange with the DP slave can be disabled and re-established immediately (corresponds to a IO reset but only for the one slave).

### 5.1.6.3.2 Features tab

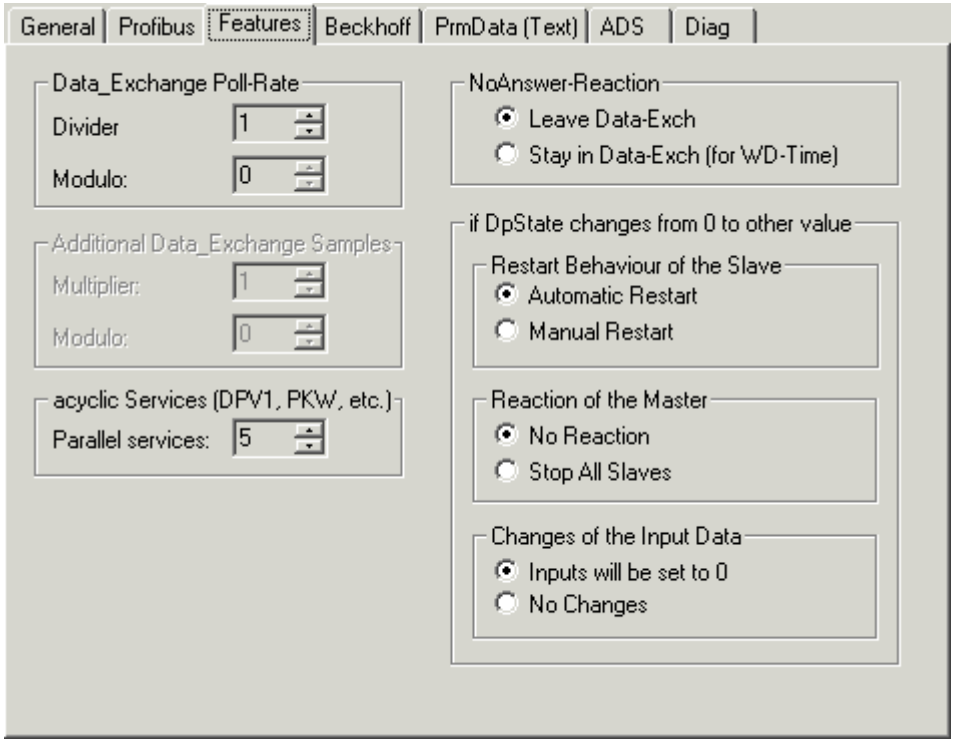



Fig. 96: Features tab

**Data_Exchange poll rate [only FC310x]**
A different polling rate (Divider) can be set for each slave. Divider 1 means that the slave is polled during each cycle, Divider 2 means every 2nd cycle, etc. The modulo will distribute slaves with dividers greater than 1 across different cycles in order to minimize the maximum cycle time (divider 2 and modulo 0 means that the slave is polled every even-numbered cycle, divider 2 and modulo 1 means that the slave is polled every oddly-numbered cycle). (see the section on Slave Prioritization/Multiple DP Cycles [▶ 130]).

**Additional Data_Exchange samples**
It is possible to run a number of DP cycles within one task cycle. It is then optionally possible for each slave to be supplied with different output data in each DP cycle, and for the input data from each of those DP cycles to be transferred to the controller. In this case, there is an individual set of variables for each DP cycle (see the section on Slave Prioritization/Multiple DP Cycles [▶ 130]).

**NoAnswer reaction**
You can specify, for each slave, whether it should remain in the Data Exch, despite responding incorrectly or not at all. In this case (Stay in Data-Exch), data exchange is only exited if the slave has never responded correctly within the address monitoring time (provided the **watchdog** (see the box's Profibus [▶ 97] tab) is activated, otherwise data exchange is only terminated once the slave has not responded correctly 65,535 times) (see the Error Reactions [▶ 116] section).

**Restart behavior**
It is possible to specify for each slave whether, after leaving Data-Exch, it should automatically start up again, or should remain in the Wait-Prm state (see the Error Reactions [▶ 116] section).

**Reaction of the master**
For each slave you can specify whether its exit of Data Exch results in stopping of PROFIBUS cycle (FC310x: all slaves exit the data exchange and switch to Wait-Prm, restart only through IO reset or restarting of the TwinCAT system; EL6731: EL switches to PREOP state, restart through switching to OP state) (see section Error Reactions [▶ 116]).

**Changes of the input data**
For each slave it can be specified whether, on exiting of Data Exch (DpState not equal 0), its input data should be set to 0 or remain unchanged (see the Error Reactions [▶ 116] section).

**acyclic services**
The number of parallel ADS services to one box can be set here.

### 5.1.6.3.3 <Beckhoff> tab

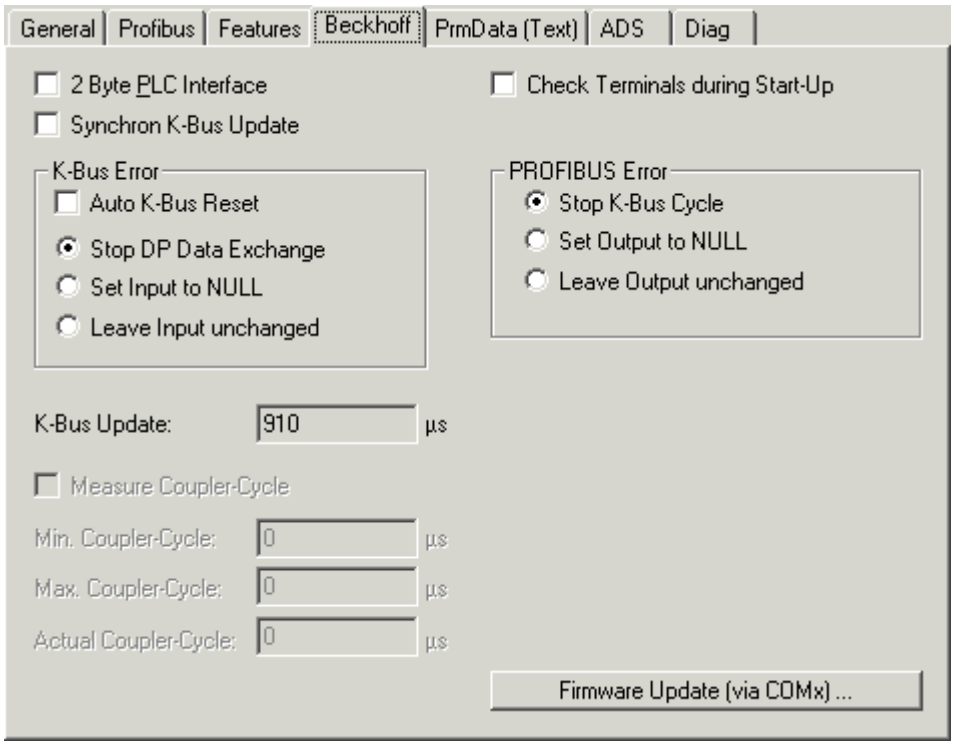



Fig. 97: <Beckhoff> tab

**Firmware Update**
This button enables updating of the firmware of a Beckhoff DP slave over a KS2000 cable via the serial port.

**2 byte PLC interface**
Switches on the 2 byte PLC interface of the Beckhoff DP slave.

**Synchron K-bus update**
For Bus Couplers, the expected internal cycle time (K-Bus cycle + DP buffer transfer) is specified here.

**Check terminals during start-up**
If this checkbox is activated, table 9 is transferred to the coupler by means of DPV1 write, and the coupler only enters data exchange (DpState = 0) if the entries agree. This makes it possible to check the terminals more precisely when starting up than is done with PROFIBUS CfgData.

**K-bus error**
It is possible to specify the reaction to a K-bus error here (automatic or manual K-bus reset, reaction in the input data in the coupler)

**PROFIBUS error**
It is possible to specify the reaction to a PROFIBUS error here (reaction in the output data in the coupler)

**Measure coupler cycle**
The cycle time on the coupler (DP + K-bus) can be measured here.

### 5.1.6.3.4 Process data tab

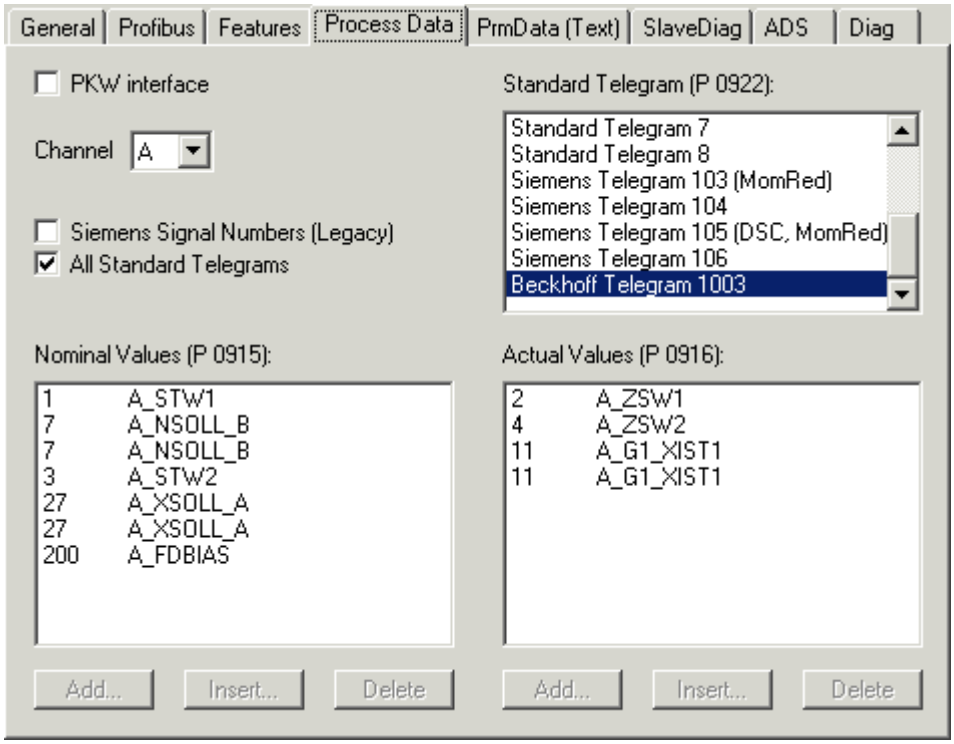



Fig. 98: Process data tab

Under Siemens AG or Profidrive MC, the boxes Profidrive MC and Profidrive MC (twice) may be selected. By default, the box will have the settings for a Simodrive 611U with standard telegram 3. For other PROFIBUS MC slaves, the identification number must be appropriately changed on the slave's Profibus [▶ 97] tab. The required telegram type can be set on the **ProcessData** tab and must usually correspond with parameter 922 of the PROFIBUS MC slaves, which can be set via a manufacturer-specific configuration tool (SimoCon U forSimodrive 611U).

This tab also has a check-box "PKW interface", which will activate the PKW interface. This enables online display of the box parameters on the Online tab (hitherto this only worked with the Simodrive 611U, since a parameter file generated by the SimoCon U is required). In any case it is possible to read and write the parameters via the PKW interface per ADS (see the PKW Protocol [▶ 50] section).

### 5.1.6.3.5 The PrmData (text) tab

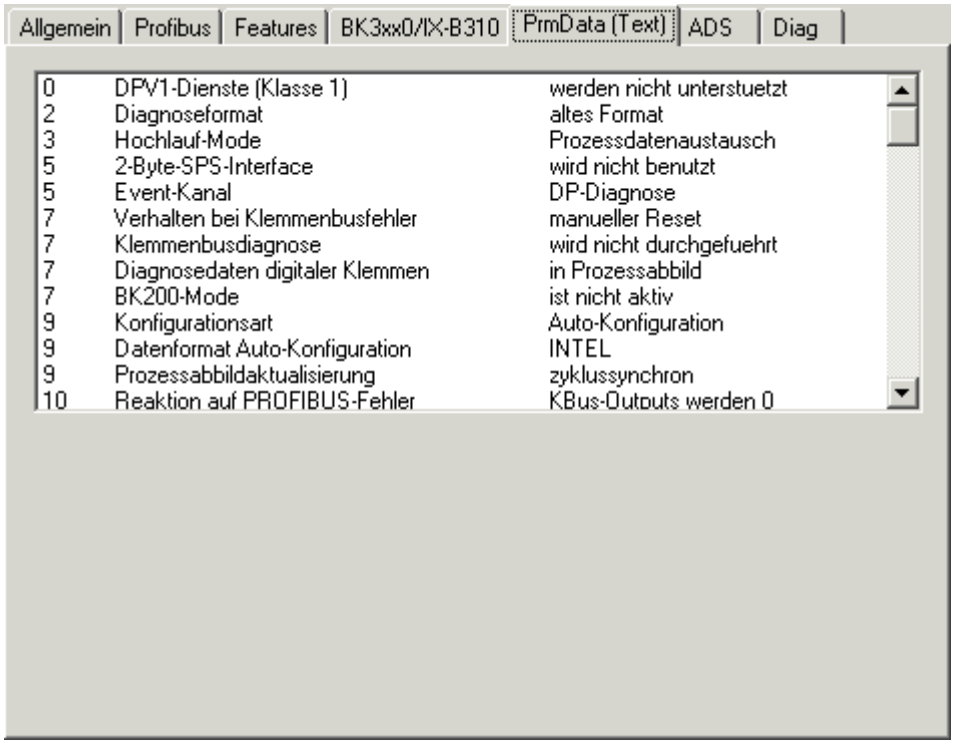



Fig. 99: The PrmData (text) tab

Click on a line to change the current value. The description of the respective settings can be found in the documentation of the relevant manufacturer.

### 5.1.6.3.6 Diag tab

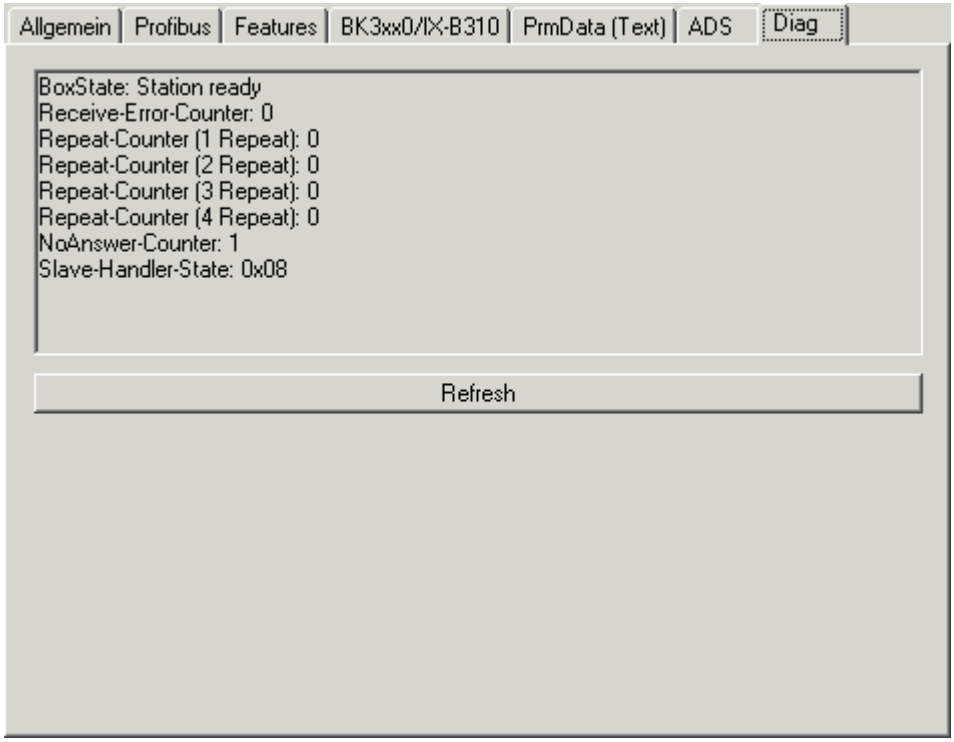



Fig. 100: Diag tab

The following information can be displayed here:

**BoxState**
The current DpState [▶ 122] is displayed here.

**Receive-Error-Counter**
Number of disturbed telegrams from the slave.

**Repeat-Counter**
Number of required repetitions due to missing or disturbed response from the slave.

**NoAnswer Counter**
Number of telegrams that remained unanswered by the slave.

**Last DPV1 error**
Error-Decode, Error-Class, Error-Code and Error-Code 2 (see description of the DPV1 Error Codes [▶ 127])

For Beckhoff DP slaves, further diagnostic information will be displayed.

## 5.1.7 Operation of PROFIdrive MC drives

### 5.1.7.1 I/O configuration

The following section describes the integration of PROFIdrive MC drives (in this case a Sinamics S120 servo drive controller) in the Beckhoff TwinCAT NC/PTP.

The quick reference guide focuses on the operation of synchronous servo axes. It is based on TwinCAT NC/PTP version 2.10 (TwinCAT NC/PTP 2.10 build 1340 or higher is recommended).
Familiarity with Beckhoff TwinCAT NC/PTP is a prerequisite.

#### 5.1.7.1.1 Configuring a fieldbus master

**Selecting a Profibus fieldbus master card**

It is assumed that the user is familiar with the procedure for inserting a Profibus device (an EL6731 fieldbus master card/fieldbus master terminal) in the TwinCAT System Manager. Once installation is complete and all bus devices are operational, use Scan Devices in Config mode or the option Insert Device -> Profibus DP via the context menu in the I/O devices tree. If the configuration doesn't take on the target device (or without the fieldbus master being connected to your PC), you can acknowledge the dialog "Device found at address" with "Cancel". The card address can then be determined later during commissioning.

Possible devices are offered, depending on selected target system (only PC, CX, BX, All). The sample refers to an FC31xx PCI fieldbus master card; if an EL6731 is used the description applies accordingly.

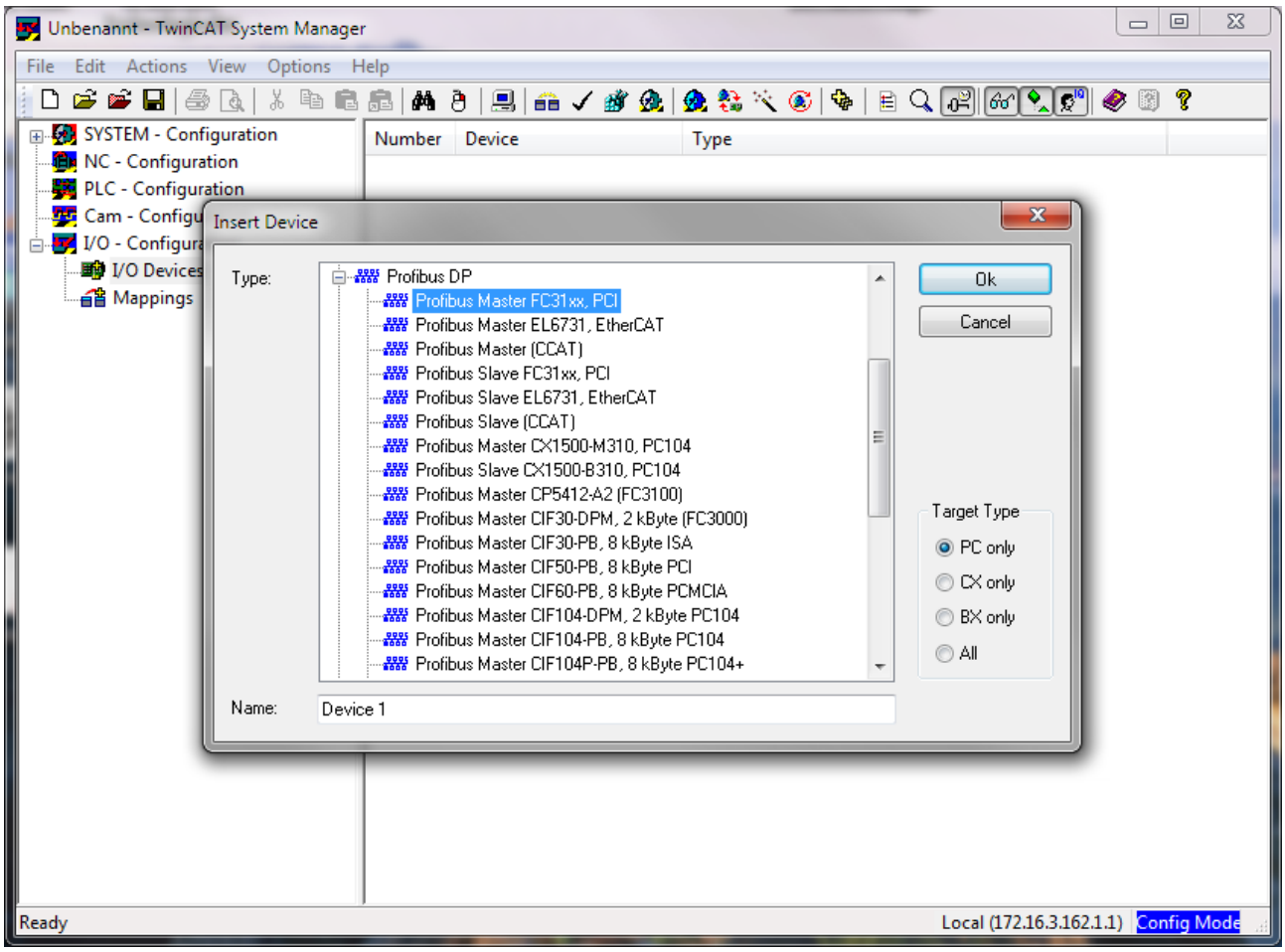



Fig. 101: Inserting an FC31xx PCI I/O device

Select the suitable I/O device (in this case Profibus master FC31xx PCI). Operation with a Profibus master EL6731 via EtherCAT is tested with 2 CU.

### Configuring Profibus DP

The Profibus master card is configured via the tab FC31xx. This can only be done during commissioning when the fieldbus is running and operational and all elements are configured:

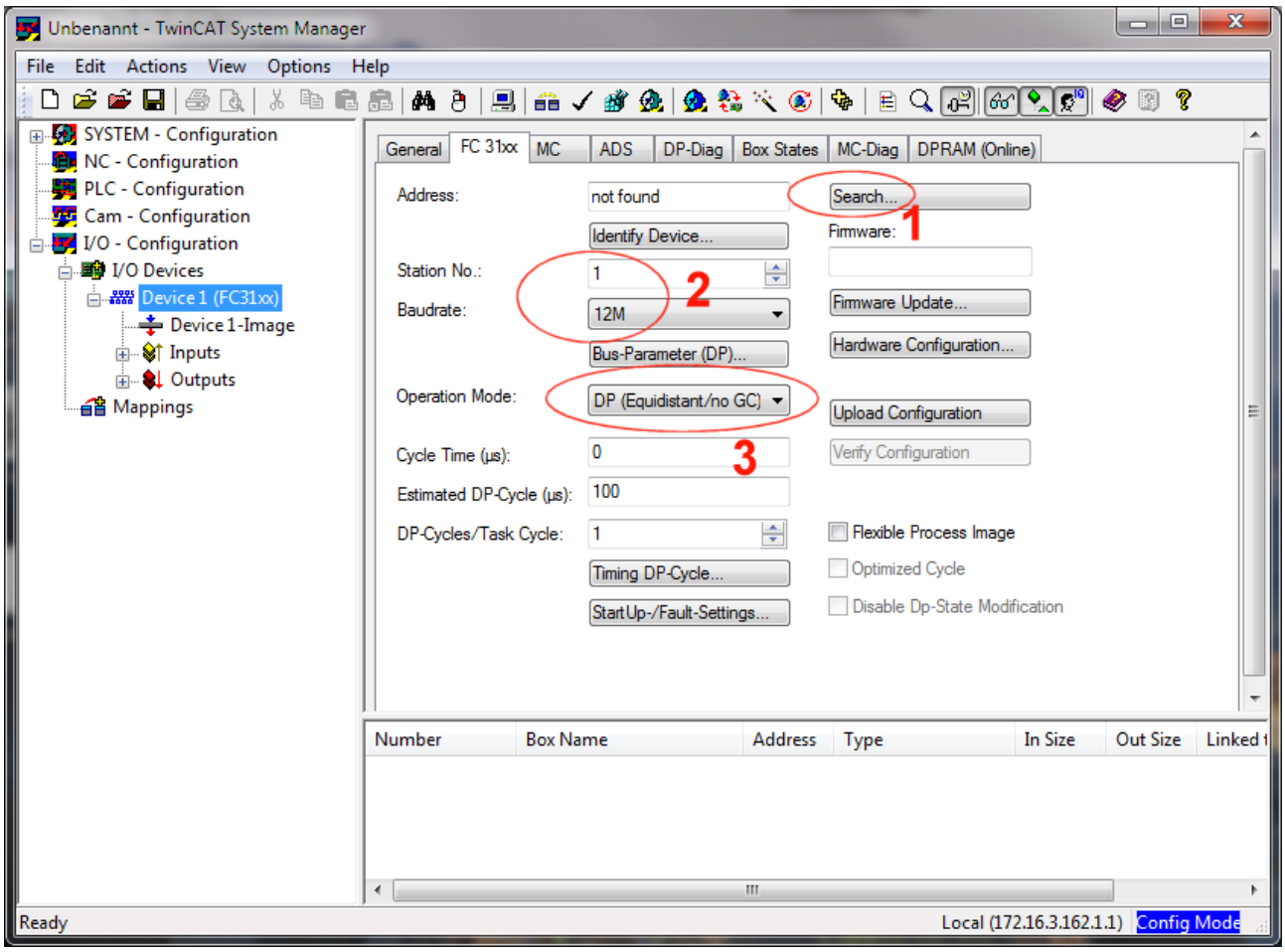



Fig. 102: Configuration 1

(1) The card address must be determined later during commissioning. The firmware version of the FC310x Profibus card should be 2.58 or higher, for the FC3151 2.5b or higher.
(2) The station address of the fieldbus master is preset to 1. Avoid address conflicts with other devices (e.g. a programming device with the STARTER software) and check the baud rate setting (12 M).
(3) The mode must be set to DP/MC (Equidistant).

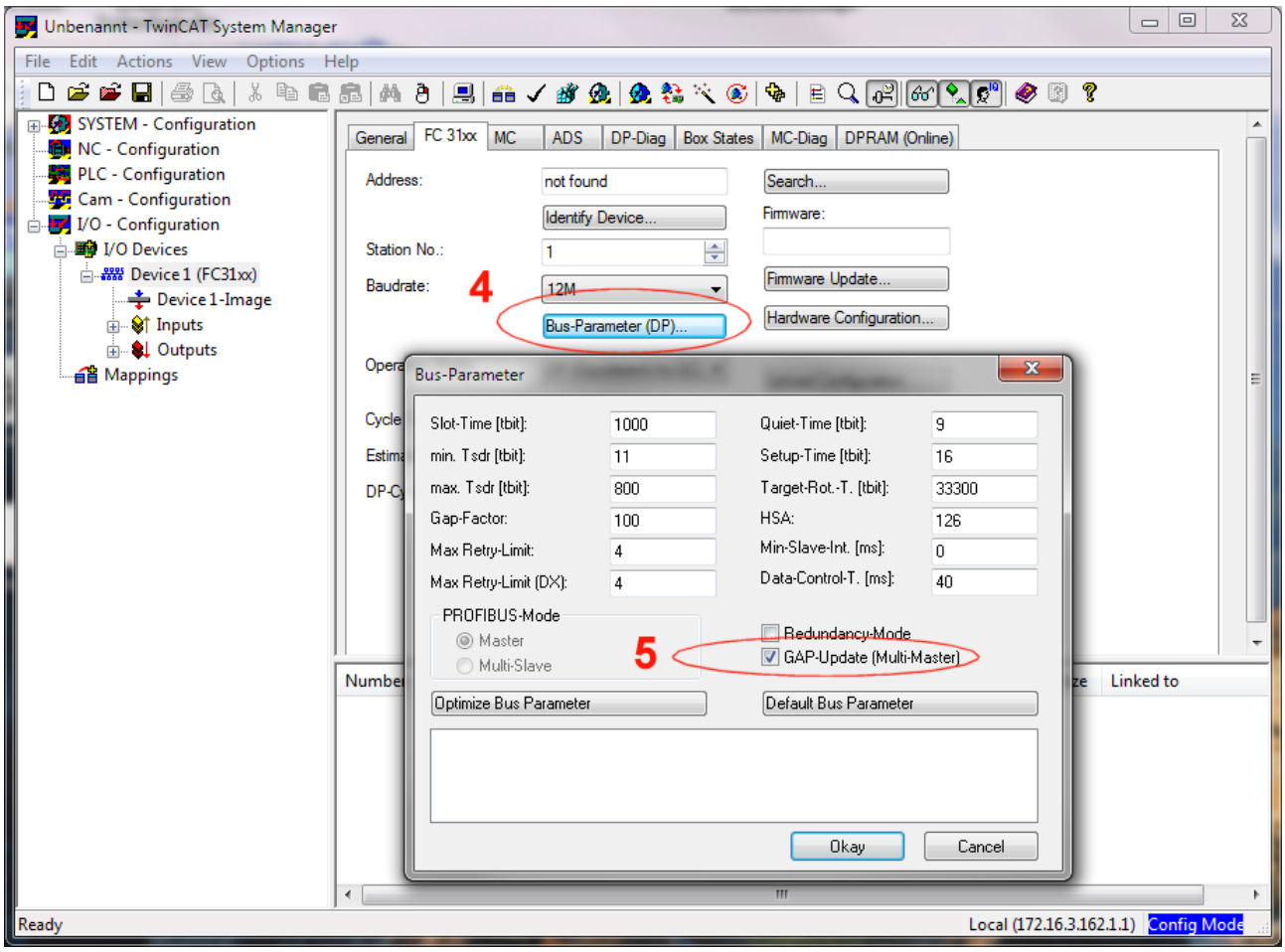



Fig. 103: Configuration 2

(4) Use the Bus Parameter (DP) button to activate GAP update (multimaster) via the dialog
(5) in order to enable operation of a programming devices at the Profibus for parallel configuration of the STARTER software

### 5.1.7.1.2 Configuring fieldbus devices

#### Append box

In the context menu of the Profibus master card Device1 select "Append Box". Select from folder PROFIdrive MC (DPV2) (with the respective multiple). Please note that numbers 1-8 are available for selection, representing up to 6 drive controllers, a CU and an Active Line Module. In the example (sevenfold) communication via the Profibus takes place in a system of 6 axes and a CU.

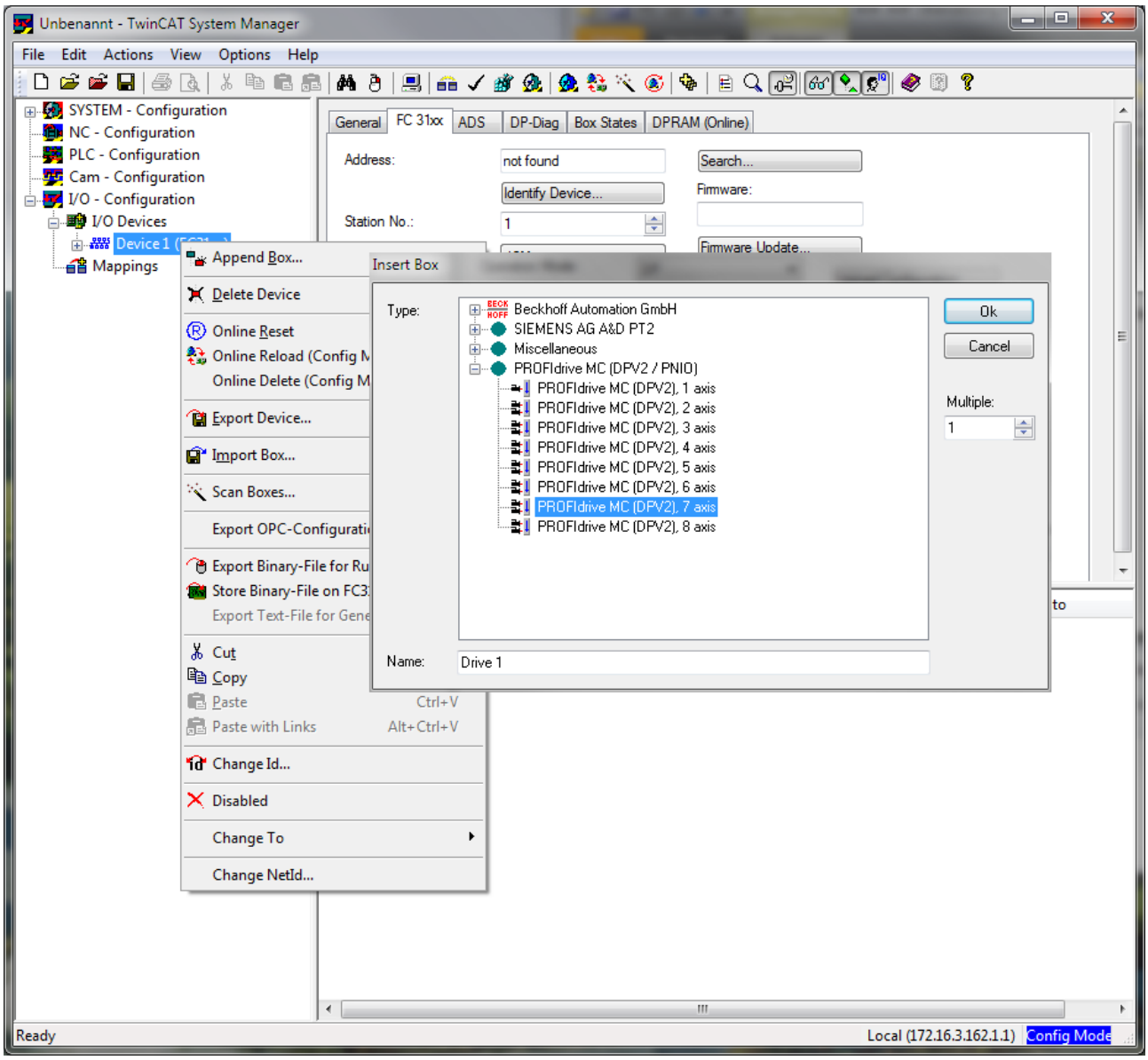



Fig. 104: Inserting a box

### GSG file

Please select "Sinamics S" from the "Select Box" dialog:
Ensure that the file Profidrive2.dat and the GSG file (editor) referred to in this file are in directory \TwinCAT \IO\Profibus\

Fig. 105: Selection dialog

Example Profidrive2.dat:

```
Simodrive 611U=
Sinamics S=si0480e5.gsg
FC310x-MC-Slave=FC310xMC.gsd
```

The corresponding GSG file is read, and the data structures are preset. To use a new, modified GSG file, create a new box (axis combination) and links to the software structures (see section NC configuration [▶ 111]).

### Profibus slave configuration

The fieldbus configuration for the axis system is set in the Profibus tab:
Set the Profibus address (station no. 2 in the screenshot below) according to the CU address specified in the drive configuration (STARTER program).

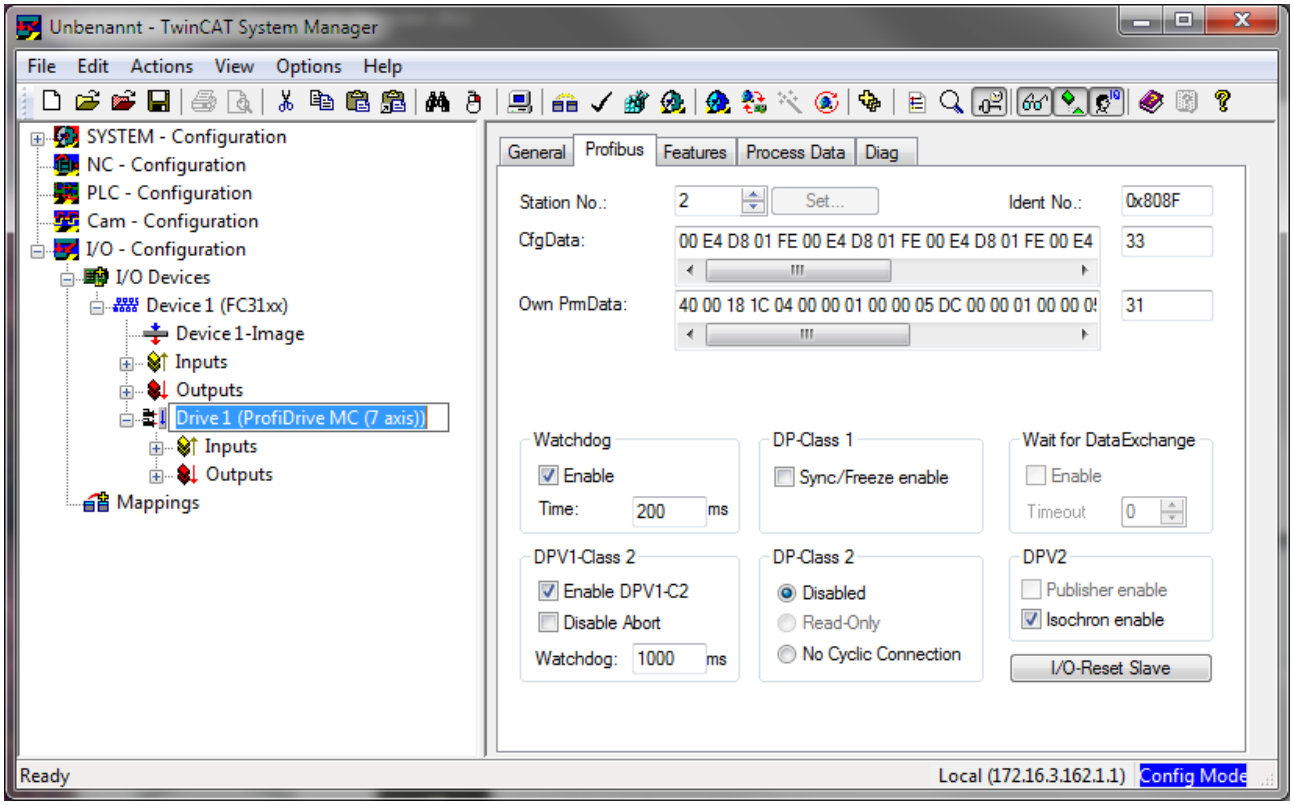



Fig. 106: Profibus tab, address selection

### Configuration of the PROFIdrive axis system in TwinCAT

The object number (object ID) 1-8 set in the STARTER software matches the channel A-H. Each channel must be assigned a suitable telegram. In the Process Data tab the individual drives of the axis (the axis system) are set to telegram 3 by default. Tests indicated that DSC mode (telegram 5) is not necessary in conjunction with TwinCAT. Telegram 390 is assigned to the CU.

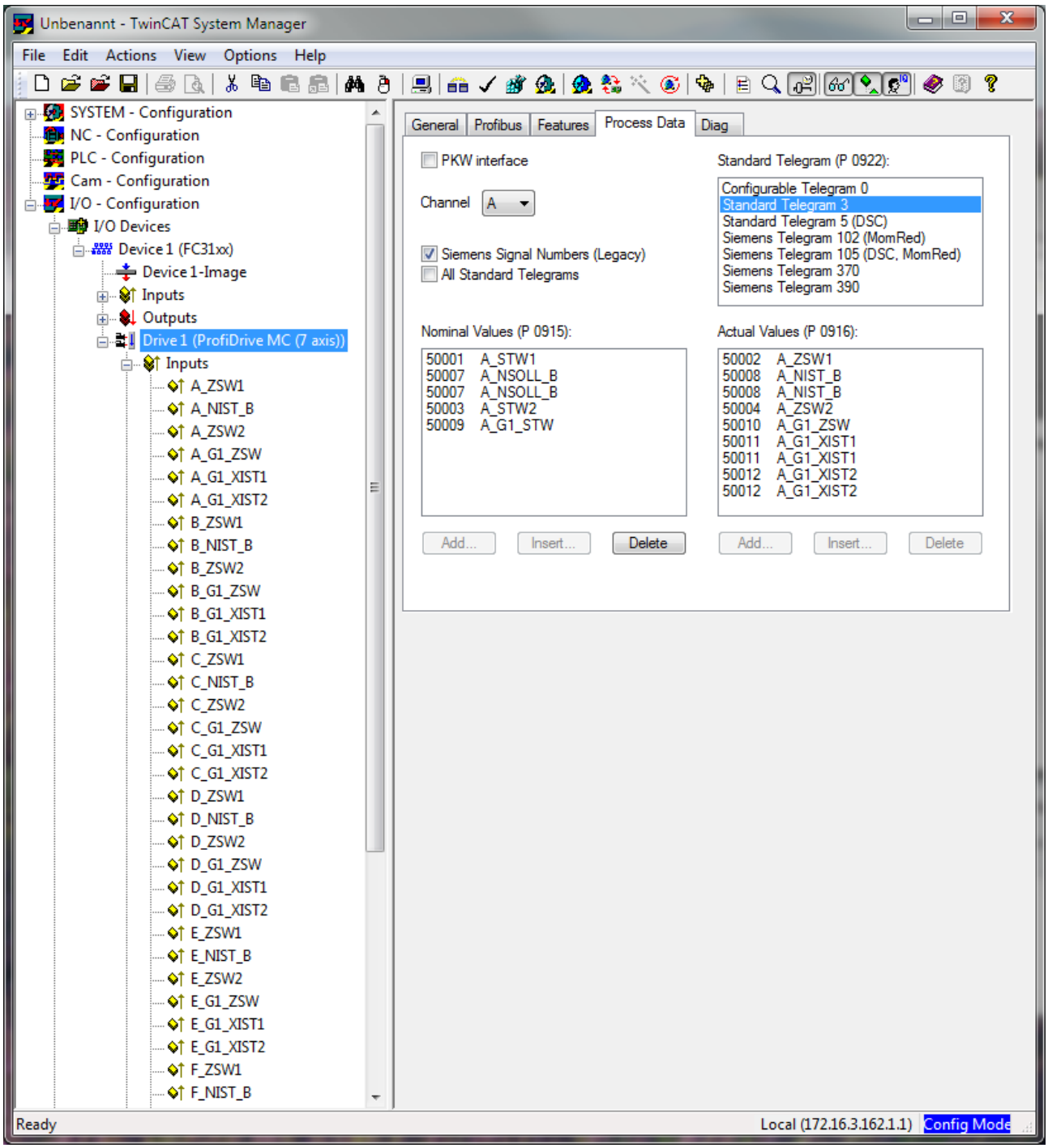



Fig. 107: Process Data tab, telegram assignment

Up to TwinCAT 2.10, B1319, this telegram 370 is not predefined for the Active Line Module ALM, but it can be configured via the Configurable Telegram 0.

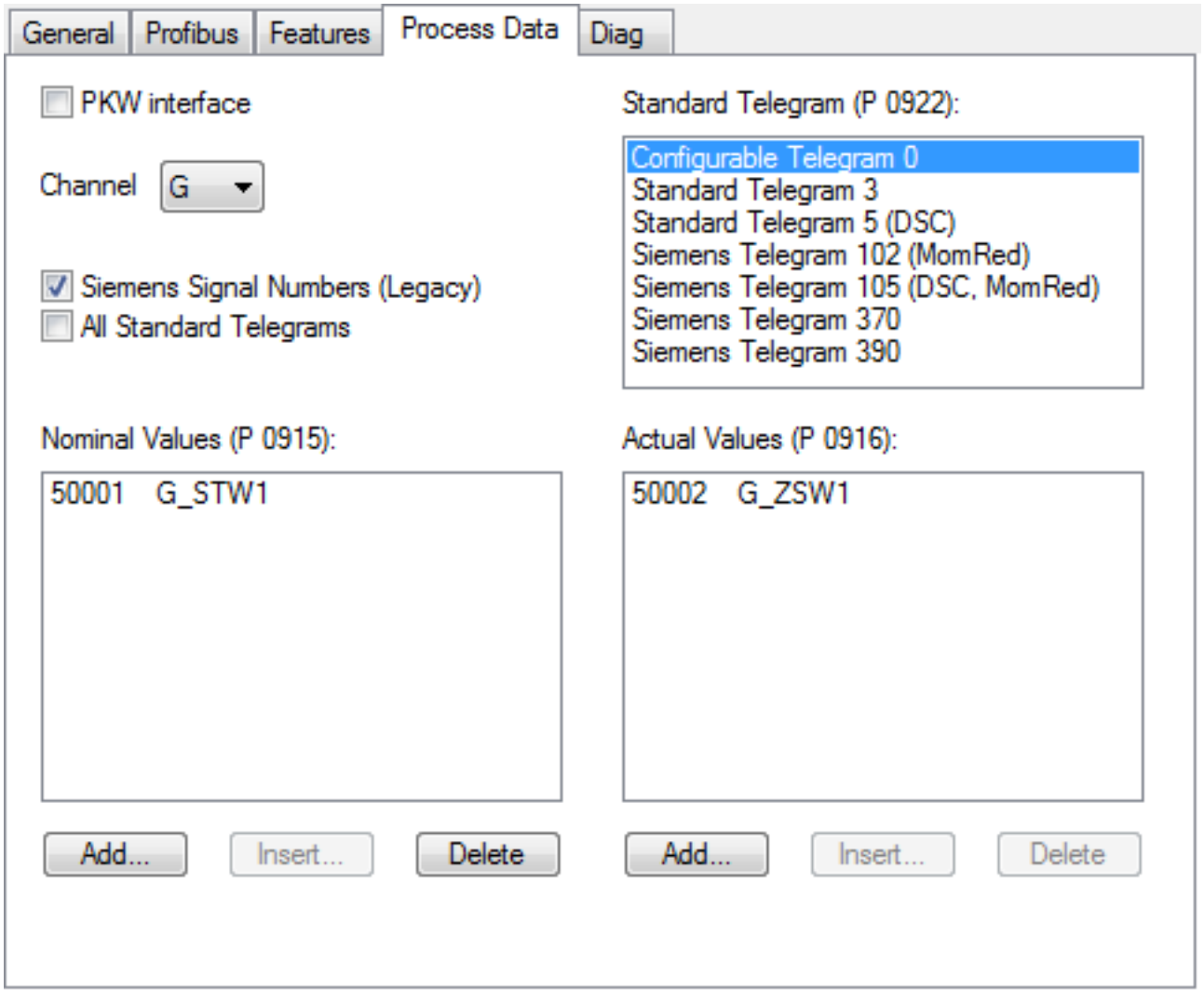



Fig. 108: Process Data tab, determine telegram

In the sample the parameter "PKW Interface" has to remain disabled, since Sinamics S (unlike Simodrive 611U) no longer supports the PKW interface.

**Determining the Profibus DP cycle**

Once the complete fieldbus has been configured, press the button "Calculate MC Times" in the MC tab of the FC31xx.
The "Estimated DP Cycle Time" (in this case 374 µs) for all axes must be smaller than the NC task time referred to in section NC Configuration [▶ 110] (e.g. 2 ms). The actual value should be checked later online.

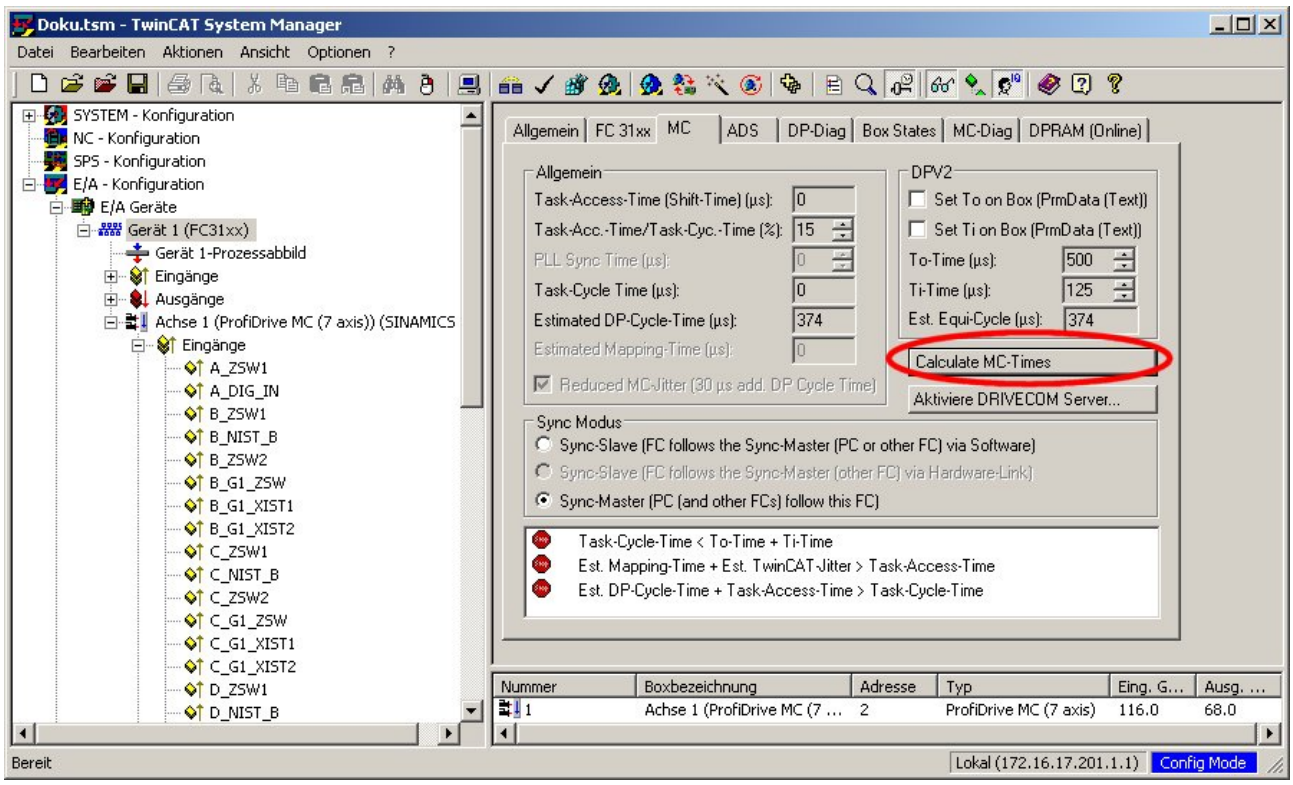



Fig. 109: MC tab, determine PB - DP cycle

## 5.1.7.2 NC - Configuration

### Appending and parameterizing an NC task

After appending an NC task the cycle time of the block execution task (NC task x SAF) has to be parameterized. The cycle ticks define the cycle time in multiples of the set basic TwinCAT time (standard: 1 ms) for the task. The cycle time takes effect if/when auto start is selected. The task cycle time must be greater than the expected fieldbus cycle time (Estimated DP Cycle Time - see section I/O Configuration [▶ 102]). Shorter times than 2 ms have not yet been tested with Sinamics S120.

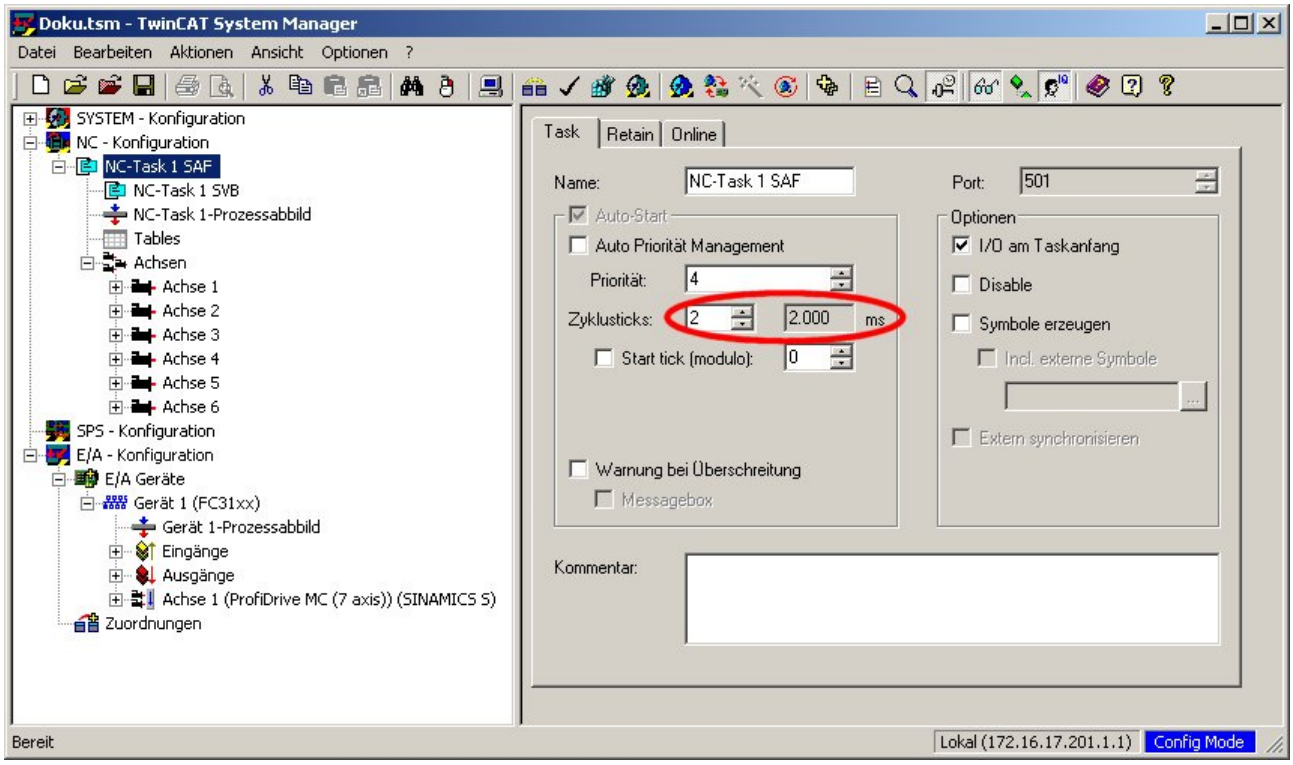



Fig. 110: NC task, cycle time

### Appending and parameterizing axes

Enter the required number of axes in the context menu for the Axes symbol. The names Axis 1, Axis 2,… are only placeholders. To facilitate identification the names of the function are specified in the software, or the structures from a previous project are adopted. Set the **axis type** to Profidrive MC (DPV2) and the unit for actual position logging (e.g. mm for a linear axis) in the Settings tab for the respective individual axis. Use the button "Linked with…" to specify which of these logical axes (software structures) are linked to which individual PROFIdrive MC axis (axis system x, #A…#H).

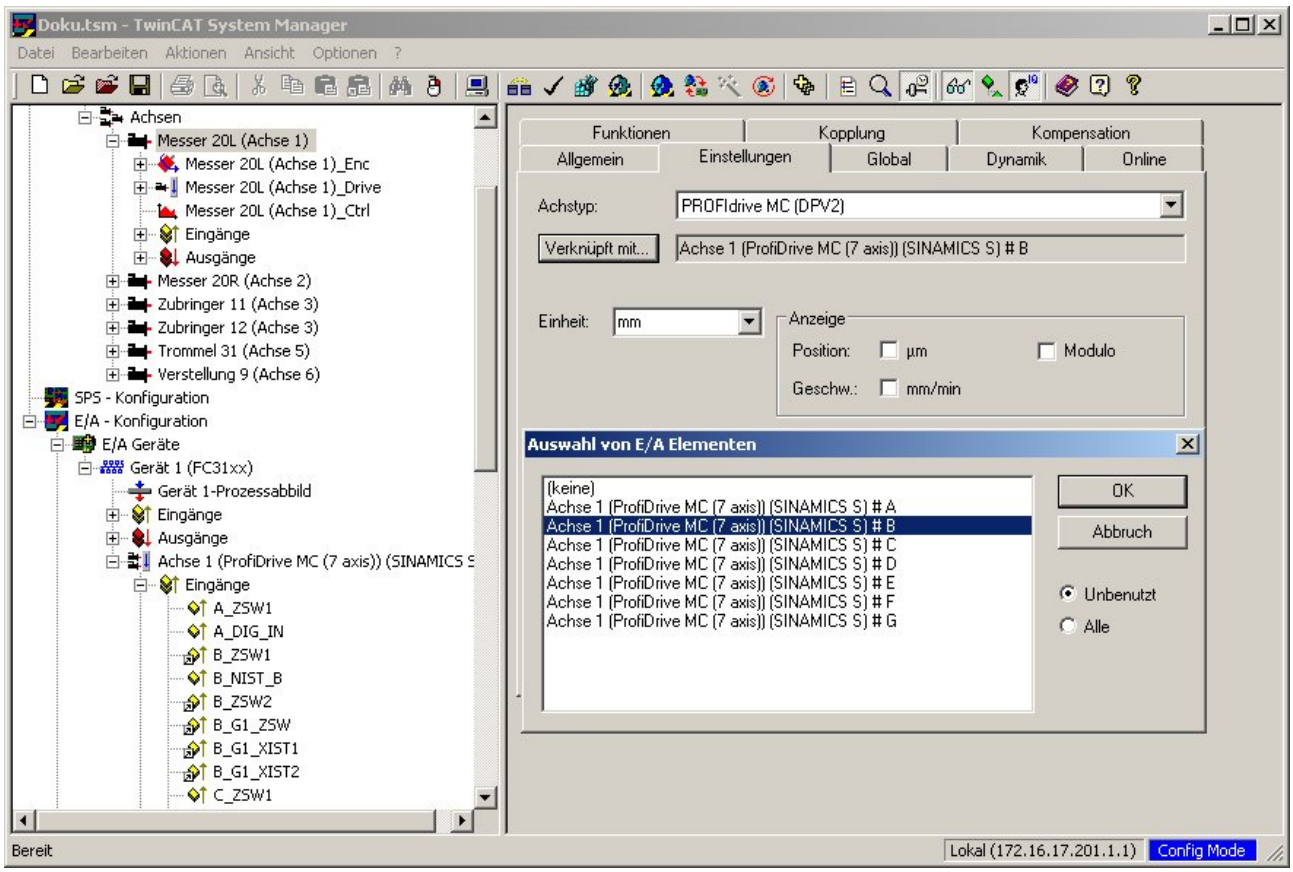



Fig. 111: Configuration, linking of axes

If required for control or diagnostics, CU and Active Line Module are linked with variables (structures) of a PLC project (which have to be created).

### Verifying Profibus MC times

Once the NC and the fieldbus have been configured, in online mode press the button "Calculate MC Times" in the MC tab of the master card FC31xx (I/O configuration) and note any warnings. The "Estimated DP Cycle Time" for all axes must be smaller than the NC task time referred to above. A safety margin of approx. 10% should prevail.

### Reading parameters from PROFIdrive MC

The following step takes place during commissioning and requires the existence of parameterized axes and a functional fieldbus for the PROFIdrive MC axis system.

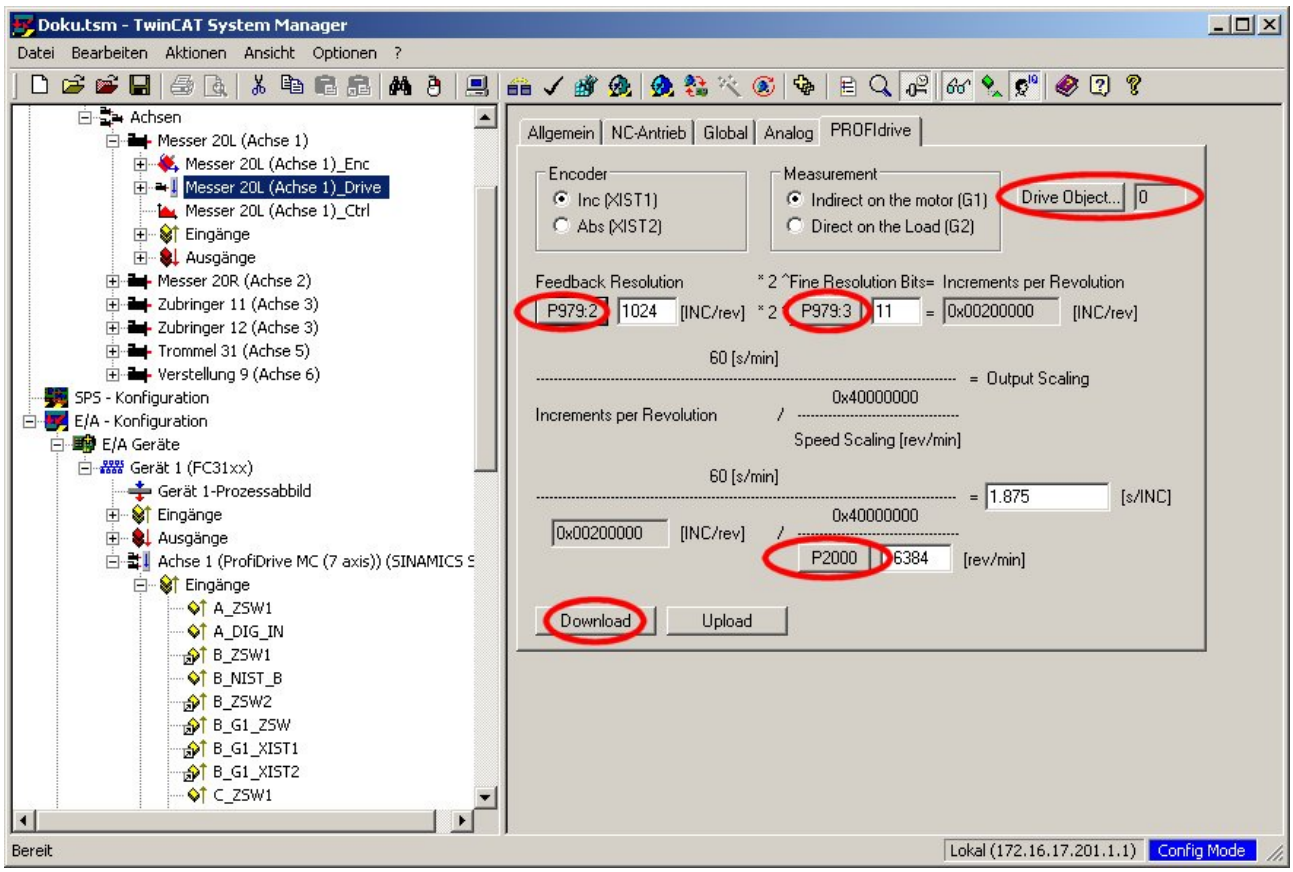



Fig. 112: PROFIdrive tab, reading motor parameters

The "PROFIdrive" tab of the axis drive (in the current example Messer 20L(Axis 1)_Drive) features a dialog for reading motor parameters from the drive. To this end the drive object number from the STARTER software has to be specified. The respective buttons can then be used to read the encoder resolution via parameters P979.2, P979.3 and P2000. Alternatively, these parameters can be read with the STARTER software and manual written into the input fields. In both cases the values are transferred to TwinCAT via the Download button.

Together with the mechanical data (e.g. transmission ratio) of the axis, the value INC/rev (increments per revolution) can be used to calculate the scaling factor (in the linear axis example in mm/INC).

**Formula**

Scaling factor = path per revolution of the drive shaft / (gear ratio x increments per revolution) Gear ratio = motor shaft revolutions / drive shaft revolutions

**Example conveyor belt:**

300 mm feed per shaft revolution, gear ratio 10, encoder resolution 20000H = 2097152DEC INC/rev scaling factor = 300 mm/rev / (10 x 2097152 INC/rev ) = 0.000014305 mm/INC

**Example drum rotation:**

Angle= 360° per shaft revolution, gear ratio 17.08, transmission via toothed belt 64/50, encoder resolution 20000H = 2097152DEC INC/rev scaling factor = 360°/rev / (17.08 x 64/50 x 2097152 INC/rev ) = 0.0000078519 mm/INC

**Setting the encoder parameter scaling factor**

Enter the determined value [▶ 112] as the scaling factor for the encoder in the Global tab and accept the values through Download

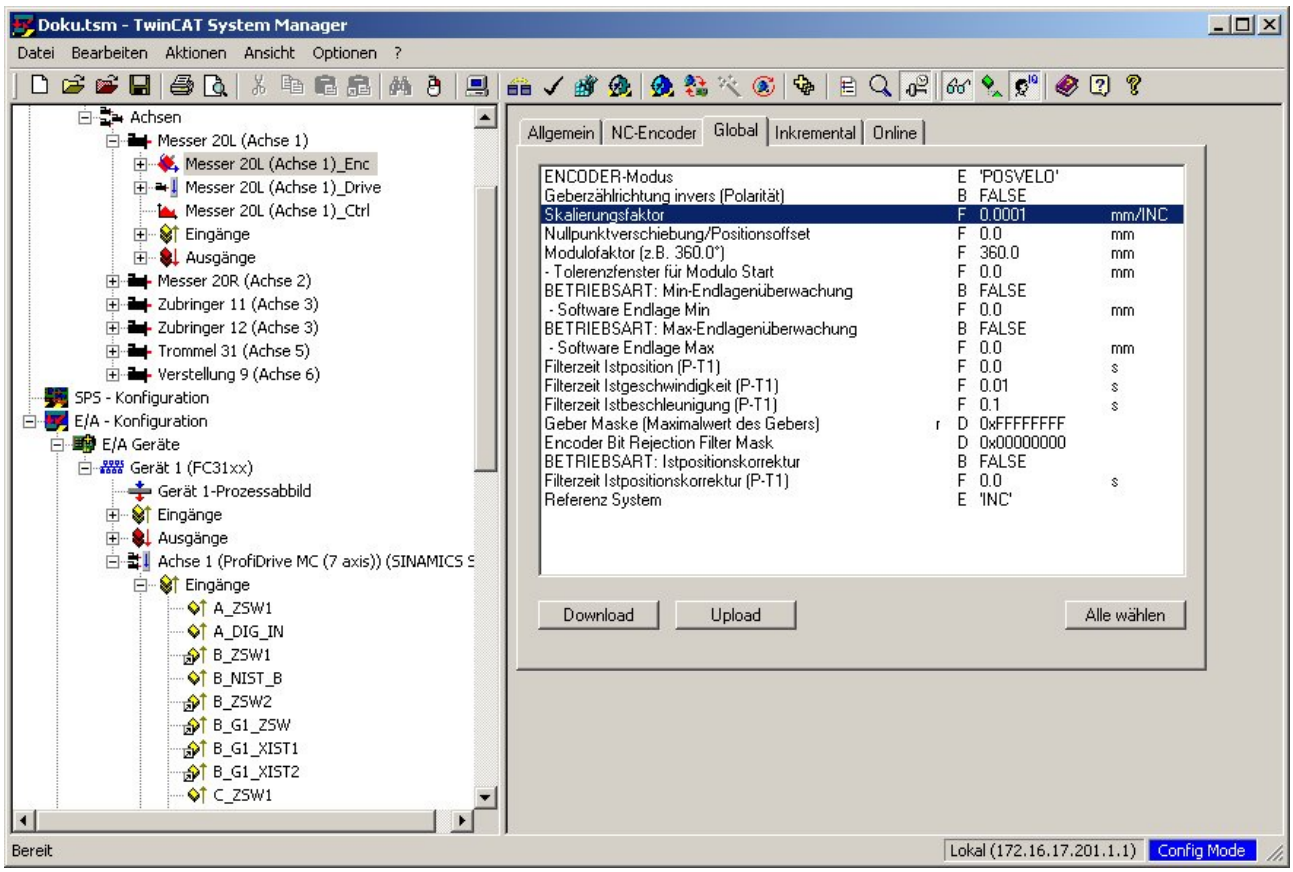



Fig. 113: “Global” tab, scaling factor entry

All other axis settings and commissioning steps correspond to the usual procedure.

## 5.1.8 Diagnostics and error description

### 5.1.8.1 LED description

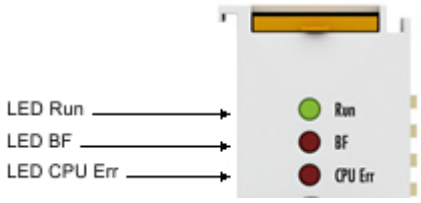



Fig. 114: LEDs

#### LED behavior

The LEDs facilitate diagnosing of the main terminal states:

| LED | Color | Meaning | |
|---|---|---|---|
| RUN | green | This LED indicates the terminal's operating state: | |
| | | off | State of the EtherCAT State Machine:<br>**INIT** = initialization of the terminal;<br>BOOTSTRAP = function for firmware updates of the terminal |
| | | single flash | State of the EtherCAT State Machine:<br>**PREOP** = function for mailbox communication and different standard-settings set |
| | | Flashing | State of the EtherCAT State Machine:<br>**SAFEOP** = verification of the sync manager channels and the distributed clocks.<br>Outputs remain in safe state |
| | | on | State of the EtherCAT State Machine: **OP** = normal operating state; mailbox and process data communication is possible |
| BF | red | off | DP master is in CLEAR/OPERATE mode, all DP slaves are in data exchange |
| | | Flashing | DP master is in CLEAR/OPERATE mode, at least one DP slave is in data exchange |
| | | on | DP master is in STOP mode |
| CPU Err | red | on | EL6731 processor error |
| | | single flash | The EL6731 processor starts |

### 5.1.8.2 Error Reactions

**Failure of a slave**

If a slave does not respond or the response is faulty, the master repeats the telegram several times, according to the **Max Retry Limit** (see Bus parameters [▶ 88] dialog). If a faulty telegram is received, the master repeats immediately, in the event of a timeout the master waits for a response from the slave until the **Slot Time** has elapsed (see Bus parameters [▶ 88] dialog). At 12 Mbaud, a slot time of 1000 bit-periods and a max retry limit of 4 (the default values) then a Data_Exchange telegram will delay sending the following telegram by

TDelay = (4 x ((15 + number of outputs) x 11 + 1000) - (15 + number of inputs) x 11)/12 µs

The DpState [▶ 122] of the slave is set to 0x02 (timeout) or 0x0B (faulty telegram). The effect on the DP connection can be set (see below).

**Normal DP cycle (12 Mbaud, 5 slaves, an average of 20 bytes I, 20 bytes O for each slave)**

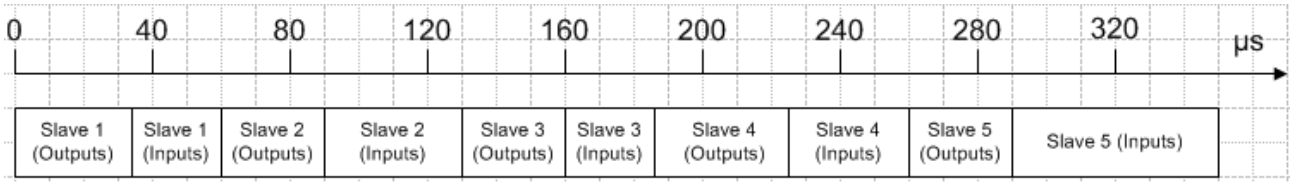



Fig. 115: Diagram showing a normal DP cycle

**First occurrence of a faulty DP cycle (slave 3 does not answer)**

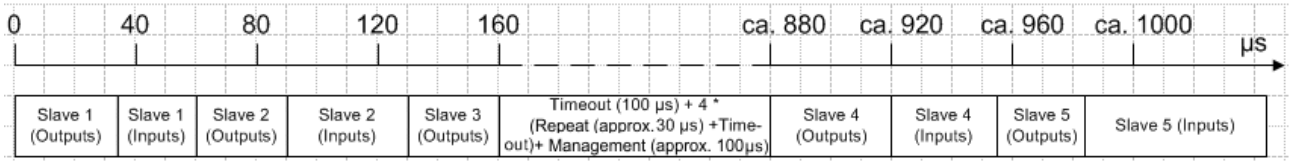



Fig. 116: Diagram showing the first occurrence of a faulty DP cycle

**Subsequent DP cycles (slave 3 no longer in the polling list)**

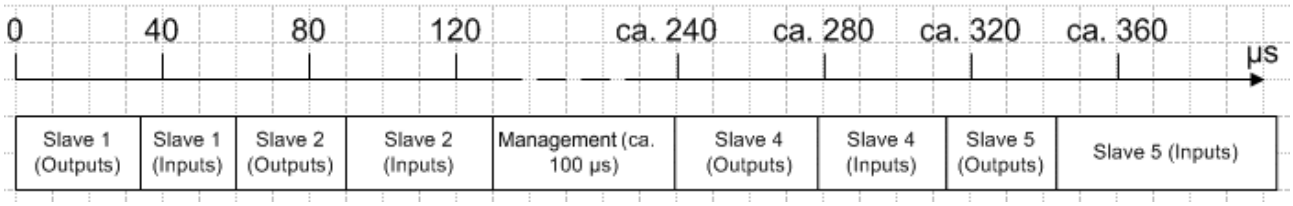



Fig. 117: Diagram showing subsequent DP cycles

It can still happen that the slave answers incorrectly (e.g. because, as a result of a local event on the slave, the DP connection has been removed). In this case, the telegram is not repeated, but the system continues by sending the next telegram. The DpState [▶ 122] is set to a value other than 0, the slave is removed from the polling list and is no longer addressed in the following DP cycle (which means that the time at which the following telegram is sent changes), until the DP connection can be established again.

**Reactions in the master**

The master's reactions can be set differently for each slave (see the tab for the slave's Features [▶ 98]).

**Effect on the DP connection (NoAnswer reaction) if the slave either does not answer or does not answer correctly**

This specifies whether the DP connection to the slave should be removed immediately in the absence of a correct reception telegram, or only after the DP watchdog time has elapsed (see the slave's Profibus [▶ 97] tab).

1. If the DP connection is to be removed immediately (Leave Data Exch, default setting) the slave is removed from the polling list and is no longer addressed in the following DP cycles until the DP connection is established once again. In order to re-establish the DP connection to the slave, at least 7 telegrams are sent, and the process generally requires at least 10-20 ms.
2. If the DP connection is only to be removed when the slave has not answered (or not answered correctly) within the DP watchdog time (Stay in Data-Exch (for WD-Time)), a further attempt is made in the next polling cycle to address the slave, but if the slave does not answer, a repeat is not sent.

The "Stay in Data-Exch (for WD-Time))" (2.) setting makes sense if the PROFIBUS cycle is to continue to operate at the most regular possible period even if a slave fails, and if the failure of a slave for one or more cycles can be tolerated (e.g. in the DP/MC (Equidistant) [▶ 41] operation mode). In this case the DP watchdog time for the slave should be set according to the tolerable outage time of the slave, and the **Max Retry Limit (DX)** (see Bus parameters [▶ 88] dialog) should be set to 0.

**Normal DP cycle (12 Mbaud, 5 slaves, an average of 20 bytes I, 20 bytes O for each slave) in the "Stay in Data-Exch (for WD-Time)" mode**

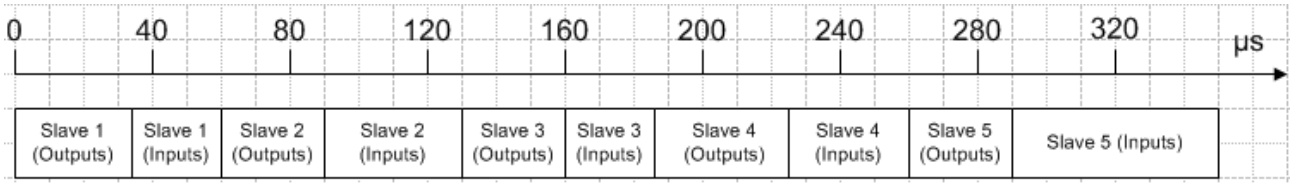



Fig. 118: Diagram showing a normal DP cycle

**First faulty and subsequent DP cycles in the "Stay in Data-Exch (for WD-Time)" mode (slave 3 does not respond)**

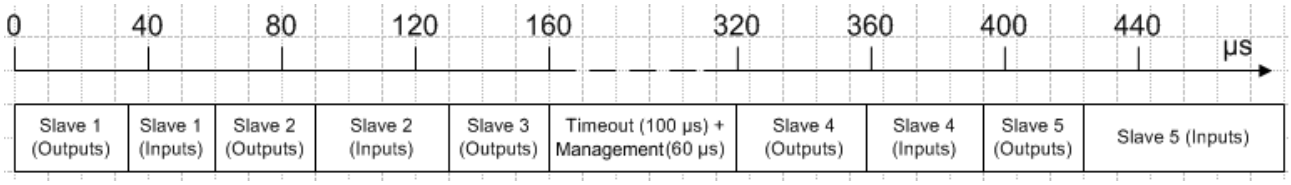



Fig. 119: Diagram: First faulty and subsequent DP cycles in the "Stay in Data-Exch (for WD-Time)" mode

**Changes of the slave's input data if the slave does not respond correctly**

This specifies whether the slave's input data is set to 0 when it fails ("Inputs will be set to 0", which is the default setting) or whether the existing values are retained ("No changes"). In either case the DpState [▶ 122] of the slave is set to value other than 0, so that the task can always recognize whether or not the data is valid. If a slave gives a faulty answer, the input data is always set to 0, independently of the setting of **Changes of the Input Data**.

**Setting the slave's restart behavior if the DP connection to the slave is removed**

This specifies whether the DP connection to slave whose DP connection has been removed is automatically re-established, or whether this should be done manually as a result of a call to ADS-WriteControl (see ADS-Interface [▶ 43]).

### The reaction of the master if the DP connection to the slave is removed

This specifies whether removing the DP connection to a slave has no other effects (No Reaction, the default setting), or whether the master should enter the STOP state, thus removing the DP connections to all the slaves.

### Effect on the state of the master (Clear mode), if the DP connection to the slave is removed

**Clear mode** (see Fault Settings [▶ 90] dialog) can be used to specify that the master should switch to or remain in "Clear" state, as long as at least one MC slave (setting: "Only MC slaves") or any slave (setting: "All slaves") does not respond correctly (i.e. has a DpState [▶ 122] not equal 0).

The **Reaction of the Master** setting (see the slave's Features [▶ 98] tab), which was described in the previous section, has priority over the **Auto-Clear mode**, so that when an appropriately set slave fails, the Master enters the STOP state.

### Failure of the master

### Monitoring in the PLC/IO task

In the presence of a persistent bus fault, the DP cycle can be extended up to 100 ms, even at 12 Mbaud. In order to monitor the DP master, there is a status variable CycleCounter, and this can be linked in the PLC (see the Master Diagnosis [▶ 119] section). This variable is incremented by the master after each DP cycle, so that failure of the master can be detected in by monitoring this variable in the PLC.

### Monitoring in the slave

In order to monitor failure of the master and data transmission on the PROFIBUS, a **watchdog** (see the box's Profibus [▶ 97] tab) can be activated (default setting: watchdog activated with 200 ms). The **Watchdog** must be set to at least twice the maximum **Estimated Cycle Time** and **Cycle Time** (see **"FC310x** or **"EL6731"** tab (for TwinCAT of the master)).

### Failure of the PLC/IO task (FC310x) or EtherCAT interruption (EL6731)

A distinction is made between the cases PLC stop, reaching of a breakpoint and task stop [EL6731: EtherCAT interruption] (IO task, NC task is only stopped on system stop). In the case of a PLC stop, the output data is set to 0 by the PLC, whereas when a breakpoint is reached the data initially remains unchanged.

In the master, the task is monitored with a monitoring time (according to the **Task Watchdog** setting times the task cycle time, see Fault Settings [▶ 90] dialog). If no new data transfer takes place within this monitoring time, the master switches to "Clear" state according to the setting **Reaction on PLC Stop** or **Reaction on Task Stop** (see Fault Settings [▶ 90] dialog; outputs are set to 0 or safe state (Fail_Safe = 1 in the GSD file, default setting) or remains in "Operate" state (outputs retain the last value). The "Operate" setting is valuable when the outputs should not be cleared when a breakpoint is reached in the PLC. However, if the PLC stops, the outputs will still be set to 0 (by the PLC), even if the master remains in the "Operate" state. It should, however, be noted that the outputs will only be zeroed if the previous DP cycle is completed in time (see the Synchronization [▶ 42] section). It should therefore only be set during the commissioning phase.

### Host failure [only FC310x]

In order to monitor the host for crashes (e.g. Blue Screen in the case of a PC), a **watchdog time** can be set (see Fault Settings [▶ 90] dialog). If this watchdog timer elapses, the master enters the OFFLINE state, so that the DP connections to all the slaves are removed, and the master logs off from the PROFIBUS, ceasing to carry out bus accesses.

### Start-up behavior

The DP connections to all the slaves are established when the TwinCAT system starts up. Until the highest priority task that is involved has not been started, the master still does not send any Data_Exchange telegrams even after the DP connection has been established, and sends only diagnostic telegrams. As

soon as the highest priority task has transferred data once, and the DP connection for the corresponding DP slave has been established, the master cyclically (with the highest priority assigned task) sends one Data_Exchange telegram to each of the corresponding slaves.

In addition, the **Operate Delay** and **Clear Mode** settings (see Fault Settings [▶ 90] dialog) can be used to specify when the master switches from "Clear" state (outputs are set to 0 or safe state (Fail_Safe = 1 in the GSD file)) to "Operate" state (outputs correspond to the outputs transferred by the task). The **Operate Delay** specifies the minimum length of time for which the master should remain in the "Clear" state following the first transfer of data. As has been described above, the **Clear mode** specifies whether the master changes into or remains in the "Clear" state if a slave in general or an MC slave in particular fails.

#### Shut-down behavior

The reaction to the stopping of the TwinCAT system is exactly the same as has been described above in the "Failure of the Host" section; the DP connections to all slaves are removed, and the master logs itself off from the bus.

### 5.1.8.3 Master Diagnostics

#### Diagnostic Inputs

The EL6731 features various diagnostic variables, which describe the state of the terminal and the Profibus and can be linked in the PLC:

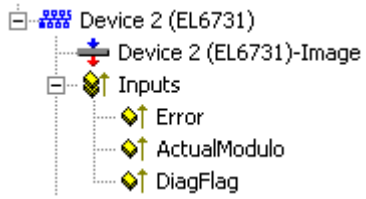



Fig. 120: Diagnostic inputs for EL6731 in the TwinCAT tree

**Error**
Shows the number of slaves with which data exchange could not take place in the last cycle. The box status (BoxState) of the slaves should only be checked if this value is not equal to 0.

**ActualModulo**
Indicates the current modulo. This variable is only of significance if the slaves are prioritized (see the Slave Prioritization/Multiple DP Cycles [▶ 130] section)

**DiagFlag**
Indicates whether the card's master diagnostic information has changed. It can then be read via ADS [▶ 43] by the controller program, after that the "DiagFlag" variable is reset.

#### Master Diagnostics Data

The master diagnostic data can be read by ADS [▶ 43]:

| ADS-Read parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 200 |
| IndexGroup | 0x0000F100 |
| IndexOffset | offset within the diagnosis data |
| Length | Length of the diagnostic data that is to be read |
| Data | Diagnostic data |

The master diagnostic data has the following structure:

| Offset | Description |
|---|---|
| 0 -125 | BusStatus List, one byte per station address 0-125 which contains the station status (see BoxState for PROFIBUS boxes and for non-selected stations: 0x80 - not present, 0x81 - slave, 0x82 - master not ready for token ring, 0x83 - master ready for token ring, 0x84 - master in token ring). |
| 126 - 127 | reserved |
| 128 - 135 | State of the EL6731 |
| 130 - 131 | detected bus errors |
| 136 - 137 | Send error counter for all sent telegrams. |
| 138 - 139 | Receive error counter for all received telegrams. |
| 140 - 255 | reserved for extensions |
| 256 - 257 | Sync Failed Counter (see MC-Diag [▶ 95] tab) |
| 258 - 259 | Cycle Start Error Counter, counts up one when the PROFIBUS cycle is restarted before the old cycle is complete (intercepted by the TwinCAT-IO driver, only possible with customized drivers). |
| 260 - 261 | Time Control Failed Counter (see MC-Diag [▶ 95] tab) |
| 262 - 263 | reserved for extensions |
| 264 - 265 | Minimum reload value of real time timer. |
| 266 - 267 | Maximum reload value of the real-time timer (max. jitter, see MC-Diag [▶ 95] tab, = max. reload value - min. reload value) |
| 268 - 269 | PLL Overflow Counter (see MC-Diag [▶ 95] tab) |
| 270 - 271 | PLL Underflow Counter (see MC-Diag [▶ 95] tab) |
| 296 - 297 | Output-Failed-Counter |
| 298 - 299 | Input-Failed-Counter |
| 300 - 301 | actual Cycle-Time (µs) |
| 302 – 303 | min. Cycle-Time (µs) |
| 304 - 305 | max. Cycle-Time (µs) |
| 306 - 307 | CycleWithNoDxch-Counter |
| 308 - 309 | CycleWithRepeat-Counter |
| 310- 311 | max. Repeat/Cycle |

**Further information on the diagnostic data**

Further information on the diagnostic data can be found in the chapter DP-Diag tab [▶ 93].

### 5.1.8.4 Slave diagnosis

#### DP-State

Each DP slave has a status variable that indicates the current state of that DP slave. This status is maintained in real time, so that it is always adapted to the current DP slave data, and can be linked to a PLC variable (-> DpState [▶ 122] of the slave):

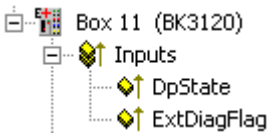



Fig. 121: Input DpState in the TwinCAT tree

### Diagnostic data

Any DP slave can acyclically report DP diagnostic data during data exchange operation. The slave here sets the Diag_Flag in the response to the cyclic Data_Exchange telegram, as a result of which the DP master automatically reads the DP diagnostic data from the slave. This does not affect the Data-Exchange cycle in the Beckhoff DP master, because the DP diagnostic telegram is sent at the end of the cyclic Data-Exchange cycle, and before the beginning of the next cycle. If the DP diagnostic data read from the slave has changed from its previous state, the DP master sets the "ExtDiagFlag" variable, which can be linked to a variable in the controller program.

The DP slave's current diagnostic data is displayed in the system manager on the slave's Diag [▶ 101] tab. It can also be read by the controller program via ADS [▶ 43], which will cause the "ExtDiagFlag" flag to be reset once more:

| ADS-Read parameters | Meaning |
|---|---|
| Net-ID | Net-ID of the master (see the device's ADS [▶ 93] tab) |
| Port | 200 |
| IndexGroup | 0x00yyF181 (yy = station address of the slave) |
| IndexOffset | offset within the diagnosis data |
| Length | Length of the diagnostic data that is to be read |
| Data | Diagnostic data |

The diagnostic data contains the slave statistics (32 bytes) and the DP diagnostic data sent by the slave (up to 244 bytes), and is constructed as follows:

| Offset | Meaning |
|---|---|
| **Slave statistics** | |
| 0 | Receive Error Counter (WORD): The number of faulty telegrams occurring while communicating with this slave. |
| 2 | Repeat Counter[8] (WORD): The repeat counters indicate how many repeats have had to be made, and how often. Repeat Counter[0] indicates how often it has been necessary to repeat a telegram for this slave once, Repeat Counter[1] shows how often a telegram for this slave has had to be repeated twice, and so on. The maximum number of retries is set with the parameter **Max Retry Limit** (see Bus parameters [▶ 88] dialog). The value range is from 0 to 8, therefore there are 8 repeat counters (for 1 to 8 retries) |
| 18 | reserved for extensions |
| 20 | NoAnswer Counter (DWORD): The number of telegrams in communication with this slave that have not received an answer. The first time that a slave fails to answer, the telegram is repeated up to **MaxRetryLimit** times, but if it does not answer even then, further telegrams are not repeated. |
| 24-27 | Last-DPV1-Error[4] (BYTE): The most recent faulty DPV1 response is entered here (byte 0: DPV1 service (bit 7 is set, thus indicating an error), byte 1: Error_Decode, byte 2: Error_Code_1 (Error_Class/Error_Code), byte 3: Error_Code_2), see description of the DPV1 error codes [▶ 127] |
| 27-31 | reserved for extensions |
| from 32 | DP diagnostic data [▶ 124] |

### 5.1.8.5 DP State of the Slaves

| Value | Description |
|---|---|
| 0 | No Error - station is exchanging data |
| 1 | Station deactivated - slave has been deactivated, temporary state during StartUp |
| 2 | Station not exists - slave does not reply on the bus -> check whether slave is switched on, whether PROFIBUS plug is in, correct station address or bus cables |
| 3 | Master lock - slave is exchanging data with another master -> remove other master from bus or release slave again by other master |
| 4 | Invalid slave response - incorrect answer from slave, occurs temporarily if slave has ceased data exchange as a result of a local event |
| 5 | Parameter fault - check whether Bus Coupler / GSD file is correct, that station address is correct or that UserPrmData settings are correct |
| 6 | Not supported - DP function is not supported -> check whether GSD file is correct or whether station address is correct |
| 7 | Config fault – configuration fault -> check whether the added terminals / modules are correct |
| 8 | Station not ready -> station starting up, temporarily displayed during StartUp |
| 9 | Static diagnosis - slave signaling static diagnosis and cannot deliver valid data at present -> check operating state at the slave |
| 10 | Diagnosis overflow - slave signaling a diagnosis overflow -> check diagnostic data (using ADS-Read, see below) and operating state at the slave |
| 11 | Physical fault - physical fault interfering with slave response -> check cables |
| 13 | Severe bus fault ->check cabling |
| 14 | Telegram fault - slave responding with an invalid telegram -> must not occur |
| 15 | Station has no resources -> slave has insufficient resources for the telegram -> check that GSD file is correct |
| 16 | Service not activated -> temporary fault when slave ceases data exchange due to a local event, otherwise check whether DP functions are disabled at the slave |
| 17 | Unexpected telegram -> can occur temporarily if two PROFIBUS networks are connected together or check whether bus times for the second master are set identically |
| 18 | Station ready -> can occur temporarily during StartUp and until the task is started |
| 19 | DPV1 StartUp -> occurs temporarily after the DP has started up if there is still data to be sent by DPV1 Write |
| 31 | only for EtherCAT gateways: WC-State of cyclic EtherCAT frame is 1 |
| 128 | FC310x/EL6731 in slave mode, waiting for data transfer -> slave was parameterized and configured but has not yet received a Data_Exchange telegram |
| 129 | FC310x/EL6731 in slave mode, waiting for configuration -> slave was parameterized, but has not yet received a Chk_Cfg telegram |
| 130 | FC310x/EL6731 in slave mode, waiting for parameters -> slave was not yet parameterized, waiting for Set_Prm (Lock) telegram |

### 5.1.8.6 Configuration Data - CfgData

The CfgData describes the structure and length of the input and output data that is to be cyclically exchanged via Data_Exchange. There follows a description of the DP configuration data bytes

| Bits 4-7 | Meaning |
|---|---|
| 0000B | Module without data. Bits 0-3 indicate how many bytes of manufacturer-specific data are still to follow. |
| 0001B | Inputs of type byte, with no consistency. Bits 0-3 contain the length of the input data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one byte, while bits 0-3 = 1111B corresponds to a length of 16 bytes) |
| 0010B | Outputs of type byte, with no consistency. Bits 0-3 contain the length of the output data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one byte, while bits 0-3 = 1111B corresponds to a length of 16 bytes) |
| 0011B | Inputs and outputs of type byte, with no consistency. Bits 0-3 contain the length of the input or output data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one byte, while bits 0-3 = 1111B corresponds to a length of 16 bytes) |
| 0100B | A special identification format for inputs. A byte follows that describes the associated input data (see below). Bits 0-3 indicate how many bytes of manufacturer-specific data are still to follow. |
| 0101B | Inputs of type word, with no consistency. Bits 0-3 contain the length of the input data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one word, while bits 0-3 = 1111B corresponds to a length of 16 words) |
| 0110B | Outputs of type word, with no consistency. Bits 0-3 contain the length of the output data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one word, while bits 0-3 = 1111B corresponds to a length of 16 words) |
| 0111B | Inputs and outputs of type word, with no consistency. Bits 0-3 contain the length of the input or output data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one word, while bits 0-3 = 1111B corresponds to a length of 16 words) |
| 1000B | A special identification format for outputs. A byte follows that describes the associated output data (see below). Bits 0-3 indicate how many bytes of manufacturer-specific data are still to follow. |
| 1001B | Inputs of type byte, with consistency. Bits 0-3 contain the length of the input data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one byte, while bits 0-3 = 1111B corresponds to a length of 16 bytes) |
| 1010B | Outputs of type byte, with consistency. Bits 0-3 contain the length of the output data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one byte, while bits 0-3 = 1111B corresponds to a length of 16 bytes) |
| 1011B | Inputs and outputs of type byte, with consistency. Bits 0-3 contain the length of the input or output data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one byte, while bits 0-3 = 1111B corresponds to a length of 16 bytes) |
| 1100B | A special identification format for inputs and outputs. A byte first follows that describes the associated output data, and then one describing the associated input data (see below). Bits 0-3 indicate how many bytes of manufacturer-specific data are still to follow. |
| 1101B | Inputs of type word, with consistency. Bits 0-3 contain the length of the input data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one word, while bits 0-3 = 1111B corresponds to a length of 16 words) |
| 1110B | Outputs of type word, with consistency. Bits 0-3 contain the length of the output data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one word, while bits 0-3 = 1111B corresponds to a length of 16 words) |
| 1111B | Inputs and outputs of type word, with consistency. Bits 0-3 contain the length of the input or output data minus 1 (i.e. bits 0-3 = 0000B corresponds to a length of one word, while bits 0-3 = 1111B corresponds to a length of 16 words) |

If the first byte has the type "special identification format", then the second or third bytes have the following meaning:

| Bits 6-7 | Meaning |
|---|---|
| 00B | Type byte, with no consistency. Bits 0-5 contain the length of the data minus 1 (i.e. bits 0-5 = 000000B corresponds to a length of one byte, while bits 0-5 = 111111B corresponds to a length of 64 bytes) |
| 01B | Type word, with no consistency. Bits 0-5 contain the length of the data minus 1 (i.e. bits 0-5 = 000000B corresponds to a length of one word, while bits 0-5 = 111111B corresponds to a length of 64 words) |
| 10B | Type byte, with consistency. Bits 0-5 contain the length of the data minus 1 (i.e. bits 0-5 = 000000B corresponds to a length of one byte, while bits 0-5 = 111111B corresponds to a length of 64 bytes) |
| 11B | Type word, with consistency. Bits 0-5 contain the length of the data minus 1 (i.e. bits 0-5 = 000000B corresponds to a length of one word, while bits 0-5 = 111111B corresponds to a length of 64 words) |

### 5.1.8.7 Diagnostic Data - DiagData

There follows a description of the DP diagnostic data

| Offset | Meaning |
|---|---|
| 0x00.0 | StationNonExistent: slave did not reply to the last telegram |
| 0x00.1 | StationNotReady: slave still processing the Set_Prm / Chk_Cfg telegram |
| 0x00.2 | CfgFault: slave signaling a configuration error |
| 0x00.3 | ExtDiag: extended DiagData available and valid |
| 0x00.4 | NotSupported: slave does not support a feature requested via Set_Prm or Global_Control |
| 0x00.5 | InvalidSlaveResponse: slave response not DP-compatible |
| 0x00.6 | PrmFault: slave reports a parameterization error |
| 0x00.7 | MasterLock: slave currently exchanging data with another master |
| 0x01.0 | PrmReq: Slave must be parameterized and configured again |
| 0x01.1 | StatDiag: slave signaling static diagnosis / DPV1 slave application not yet ready for data exchange |
| 0x01.2 | PROFIBUS DP slave |
| 0x01.3 | WdOn: DP watchdog on |
| 0x01.4 | FreezeMode: DP slave in freeze mode |
| 0x01.5 | SyncMode: DP slave in sync mode |
| 0x01.6 | reserved |
| 0x01.7 | Deactivated: DP slave has been deactivated |
| 0x02.0 | reserved |
| 0x02.1 | reserved |
| 0x02.2 | reserved |
| 0x02.3 | reserved |
| 0x02.4 | reserved |
| 0x02.5 | reserved |
| 0x02.6 | reserved |
| 0x02.7 | ExtDiagOverflow: too much extended data present |
| 0x03 | MasterAdd: station address of master with which slave is exchanging data |
| 0x04, 0x05 | IdentNumber |
| from 0x06 | Extended DiagData |

#### Extended DiagData

A distinction is made in the Extended DiagData between identification diagnosis, channel diagnosis and manufacturer-specific diagnosis. The first byte indicates the type of the diagnosis and the length of the associated data. Several diagnostic types can also follow one another in the Extended DiagData.

### Header Byte

| Bit | Meaning |
|---|---|
| 0-5 | Length of the associated diagnostic data, including header byte |
| 6-7 | 0 = manufacturer-specific diagnosis (DPV1 is not supported) or DPV1 diagnosis (DPV1 is supported (DPV1_Enable = 1) in associated GSD file) |
| | Module diagnosis |
| | Channel diagnosis |
| | Revision number |

### Manufacturer-specific diagnosis

The structure of the manufacturer-specific diagnosis may be found in the documentation for the DP slave.

### DPV1 Diagnosis

In the case of DP slaves that also support DPV1, the DPV1 diagnosis, in which a distinction is made between status messages and alarms, is sent instead of the manufacturer-specific diagnosis.

| Byte | Meaning |
|---|---|
| 0 | Header-Byte (bits 6,7 = 0, bits 0-5 = 4..63) |
| 1 | Bit 0-6: Alarm type |
| | Bit 7:<br>0: Alarm<br>1: Status |
| 2 | Slot number (0-254) |
| 3 | Bit 0-1: Alarm specifier |
| | Bit 2: Additional acknowledge |
| | Bit 3-7: Sequence number |
| 4-63 | Manufacturer specific (see the documentation for the DP slave) |

### Alarm type

| Value | Meaning |
|---|---|
| 0 | reserved |
| 1 | Diagnostic alarm |
| 2 | Process alarm |
| 3 | Withdrawal alarm |
| 4 | Insertion alarm |
| 5 | Status alarm |
| 6 | Update alarm |
| 7-31 | reserved |
| 20-126 | Manufacturer specific (see the documentation for the DP slave) |
| 127 | reserved |

### Module diagnosis

The module diagnosis contain one bit for each DP module. The bit indicates whether a diagnosis for the corresponding DP module is present.

| Byte | Meaning |
|---|---|
| 0 | Header-Byte (bits 6,7 = 1, bits 0-5 = 2.0.32) |
| 1 | Bit 0: 1st DP module has diagnosis |
| | Bit 1: 2nd DP module has diagnosis |
| | ... |
| | Bit 7: 8th DP module has diagnosis |
| ... | ... |
| 31 | Bit 0: 241st DP module has diagnosis |
| | Bit 1: 242nd DP module has diagnosis |
| | Bit 2: 243rd DP module has diagnosis |
| | Bit 3: 244th DP module has diagnosis (a maximum of 244 DP modules as possible) |

### Channel diagnosis

The channel diagnosis provides a closer description of the cause of the diagnosis of a DP module.

| Byte | Meaning |
|---|---|
| 0 | Header byte = 0x83 (3 bytes including header, bits 6, 7 = 2) |
| 1 | Bit 0-5: Channel number |
| | Bit 6-7: 0 = reserved, 1 = input, 2 = output, 3 = input/output |
| 2 | Bit 0-4: Error type |
| | Bit 5-7: Channel type |

### Error type

| Value | Meaning |
|---|---|
| 0 | reserved |
| 1 | Short circuit |
| 2 | Undervoltage |
| 3 | Overvoltage |
| 4 | Overload |
| 5 | Overtemperature |
| 6 | Wire breakage |
| 7 | Upper limit value exceeded |
| 8 | Value below lower limit |
| 9 | Error |
| 10-15 | reserved |
| 16-31 | Manufacturer specific (see the documentation for the DP slave) |

### Channel type

| Value | Meaning |
|---|---|
| 0 | Any type |
| 1 | Bit |
| 2 | 2 bit |
| 3 | 4 bit |
| 4 | Byte |
| 5 | Word |
| 6 | 2 words |
| 7 | reserved |

**Revision number**

The structure of the revision number may be found in the documentation for the DP slave.

### 5.1.8.8 DPV1 Error Codes

In the event of an incorrect DPV1 access, the slave replies with 4 bytes of data (any values that are not described here are not defined in the DPV1 standard, and are therefore to be found in the slave's manual).

| byte 0 | DPV1 service |
|---|---|
| 0xD1 | Data_Transport |
| 0xD7 | Initiate |
| 0xDE | Read |
| 0xDF | Write |

| byte 1 | Error_Decode |
|---|---|
| 0x80 | DPV1 |
| 0xFE | FMS |
| 0xFF | HART |

| byte 2 | | Error_Code_1 |
|---|---|---|
| Error-Class (bits 4-7) | Error-Code (bits 0-3) | |
| 0x0A | 0x00 | Application, Read Error |
| | 0x01 | Application, Write Error |
| | 0x02 | Application, Module Failure |
| | 0x08 | Application, Version Conflict |
| | 0x09 | Application, Feature Not Supported |
| 0x0B | 0x00 | Access, Invalid Index |
| | 0x01 | Access, Write Length Error |
| | 0x02 | Access, Invalid Slot |
| | 0x03 | Access, Type Conflict |
| | 0x04 | Access, Invalid Area |
| | 0x05 | Access, State Conflict |
| | 0x06 | Access, Access Denied |
| | 0x07 | Access, Invalid Range |
| | 0x08 | Access, Invalid Parameter |
| | 0x09 | Access, Invalid Type |
| 0x0C | 0x00 | Resource, Read Constrain Conflict |
| | 0x01 | Resource, Write Constrain Conflict |
| | 0x02 | Resource, Busy |
| | 0x03 | Resource, Unavailable |

### 5.1.8.9 ADS Error Codes

| Error code | Meaning |
|---|---|
| 0x1129 | IndexOffset too large during reading of the FC310x/EL6731 diagnostic data |
| 0x112B | IndexOffset too large during reading of the slave diagnostic data |
| 0x112D | Invalid station address during reading of the slave diagnostic data |
| 0x2023 | Invalid IndexOffset during resetting of the FC310x/EL6731 diagnostic data |
| 0x2024 | Invalid data during resetting of the FC310x/EL6731 diagnostic data |
| 0x2025 | Invalid data length during resetting of the FC310x/EL6731 diagnostic data |
| 0x2101 | DPV1-C1-Read: cyclic connection to slave not yet established |
| 0x2102 | PKW-Read: only data lengths 2 and 4 are permitted |
| 0x2103 | PKW-Read: slave not in data exchange |
| 0x2105 | PKW-Read: slave does not support PKW |
| 0x2106 | PKW-Read: Incorrect IndexOffset |
| 0x2107 | PKW-Read: Incorrect IndexGroup |
| 0x2109 | DPV1-C1-Read: FDL fault (no response etc.) |
| 0x210A | DPV1-C1-Read: syntax error (DPV1 syntax not correct) |
| 0x210B | DPV1-C1-Read: DPV1 fault (4 bytes error code in the diagnostic data of the slave) |
| 0x210C | PKW-Read: syntax error |
| 0x210D | PKW-Read: PKW error |
| 0x210E | PKW-Read: incorrect data type |
| 0x210F | DPV1-C1-Write: cyclic connection to slave not yet established |
| 0x2110 | PKW-Write: only data lengths 2 and 4 are permitted |
| 0x2111 | PKW-Write: Incorrect IndexOffset |
| 0x2112 | PKW-Write: slave does not support PKW |
| 0x2113 | PKW-Write: Incorrect IndexGroup |
| 0x2114 | Read general: Incorrect IndexGroup |
| 0x2115 | DPV1-C1-Write: FDL fault (no response etc.) |
| 0x2116 | DPV1-C1-Write: syntax error (DPV1 syntax not correct) |
| 0x2117 | DPV1-C1-Write: DPV1 fault (4 bytes error code in the diagnostic data of the slave) |
| 0x211C | Read general: Incorrect IndexGroup |
| 0x211D | SetSlaveAdress: Incorrect IndexOffset |
| 0x211E | FDL-AGAG-Write: Incorrect IndexOffset |
| 0x211F | FDL-AGAG-Read: Incorrect IndexOffset |
| 0x2120 | FDL-AGAG-Write: Incorrect length |
| 0x2121 | SetSlaveAddress: Incorrect length |
| 0x2122 | FDL-AGAG-Read: Incorrect length |
| 0x2131 | Write general: wrong IndexGroup |
| 0x2132 | Write general: wrong IndexGroup |
| 0x2137 | PKW-Read: WORD received, but read data length does not equal 2 |
| 0x2138 | PKW-Read: DWORD received, but read data length does not equal 4 |
| 0x2139 | PKW-Read: unknown AK received (1.2 or 7 expected) |
| 0x213A | PKW-Read-Array: WORD received, but read data length does not equal 2 |
| 0x213B | PKW-Read-Array: DWORD received, but read data length does not equal 4 |
| 0x213C | PKW-Read-Array: unknown AK received (4.5 or 7 expected) |
| 0x213D | PKW-Write-Array: unknown AK received (2 or 7 expected) |
| 0x213E | PKW-Write: unknown AK received (1 or 7 expected) |
| 0x213F | PKW-Write: unknown AK received (2 or 7 expected) |
| 0x2140 | PKW-Write-Array: unknown AK received (1 or 7 expected) |
| 0x2142 | SetSlaveAddress: wrong parameter during setting of address in slave mode |
| 0x2144 | Incorrect IndexGroup in ReadWrite |
| 0x2147 | DPV1-C2-Initiate: MSAC_C2 is not activated |
| 0x2148 | Incorrect IndexGroup in Read |
| 0x2149 | Incorrect IndexGroup in Write |

| Error code | Meaning |
|---|---|
| 0x214E | DPV1-C2-Read: MSAC_C2 is not activated |
| 0x214F | DPV1-C2-Write: MSAC_C2 is not activated |
| 0x2150 | DPV1-C2-DataTransport: MSAC_C2 is not activated |
| 0x2151 | DPV1-C2-Read: FDL fault (no response etc.) |
| 0x2152 | DPV1-C2-Read: connection aborted |
| 0x2153 | DPV1-C2-Read: DPV1 fault (4 bytes error code in the diagnostic data of the slave) |
| 0x2154 | PKW-ReadNoOfElements: length must equal 1 |
| 0x2155 | PKW-ReadNoOfElements: PKW is not activated |
| 0x2156 | PKW-ReadNoOfElements: axis number is too great |
| 0x2157 | PKW-ReadNoOfElements: slave not in data exchange |
| 0x2158 | PKW-ReadNoOfElements: unknown AK received (6 or 7 expected) |
| 0x215A | DPV1-C2-Write: FDL fault (no response etc.) |
| 0x215B | DPV1-C2-Write: connection aborted |
| 0x215C | DPV1-C2-Write: DPV1 fault (4 bytes error code in the diagnostic data of the slave) |
| 0x215D | DPV1-C2-DataTransport: FDL fault (no response etc.) |
| 0x215E | DPV1-C2-DataTransport: connection aborted |
| 0x215F | DPV1-C2-DataTransport: DPV1 fault (4 bytes error code in the diagnostic data of the slave) |
| 0x2163 | DPV1-C2-DataTransport: Incorrect IndexOffset |
| 0x2600-0x26FF | AK 7 (error) during PKW processing, error code in low byte |
| 0x2700-0x27FF | Fault during DPV1 processing, 4 bytes error code in the slave diagnostic data, byte 3 of the error code (error class) is in Low byte |

### "Not expected" error codes from the PROFIBUS slave in 0x28xx

| Error code | Meaning |
|---|---|
| 0x0 | OK (is usually not passed on by the terminal) |
| 0x1 | UE = User Error |
| 0x2 | RR = No resources |
| 0x3 | RS = SAP not enabled |
| 0x8*** | DL = Data Low (normal case with DP) |
| 0x9*** | NR = No response data ready |
| 0xA | DH = Data High (DP diagnosis pending) |
| 0xC | RDL = Data not received and Data Low |
| 0xD | RDH = Data not received and Data High |

***) DL and NR are handled by the EL6731

## 5.1.9 Slave Prioritization/Multiple DP Cycles

### Distribution of the DP slaves over a number of DP cycles (prioritization of the slaves)

Slaves can be prioritized so that the DP cycle time can be kept as short as possible in systems in which a few slaves must be polled very rapidly, whereas a larger cycle time would be adequate for other slaves. It is possible to specify for each slave in what multiple of cycles (Divider under **Data-Exch Poll-Rate** on the slave's Features [▶ 98] tab) it will be polled. Distributing the polling is then helpful, as, for instance, in the case illustrated below where there are four slaves, each of which is only to be addressed in every second cycle. It is possible to make settings so that two slaves will be polled in one cycle and the other two slaves in the other cycle, so that the overall DP cycle time is kept as constant as possible. For this purpose, the Features [▶ 98] tab for the slave offers the **Modulo** setting under Data-Exch Poll-Rate. In the illustrated case, slaves 3 and 5 have **Modulo** 0 while slaves 4 and 6 are given **Modulo** 1. The current modulo value can be read in the **ActualModulo** variable which is described in the Master Diagnostics [▶ 119] section.

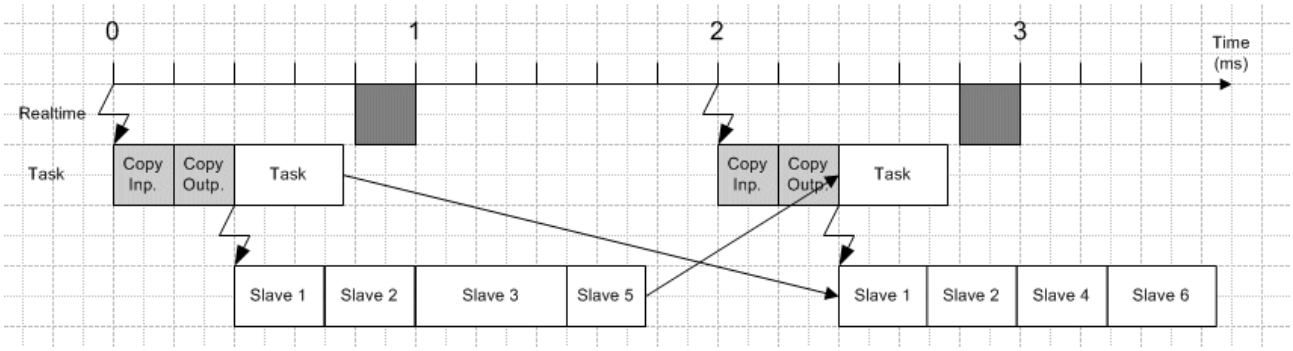



Fig. 122: Diagram showing allocation of DP slaves to several DP cycles

### Multiple DP Cycles in one Task Cycle

If the task cycle time is more than twice as long as the DP cycle time, it is possible for a number of DP cycles to be carried out within a single task cycle, in order to acquire the most up-to-date input data possible. Based on the **Number of DP cycles per task cycle** factor, which can be set on the **"FC310x"** tab of the master (for TwinCAT 2.8/2.9), a timer with the cycle time (task cycle time/number DP cycles per task cycle) is started when the first DP cycle starts, through which further DP cycles can then be started. It is, however, necessary to check that the last DP cycle is completed in good time (before the next task start), since otherwise one DP cycle will fail (or possibly more than one, depending on the ratio expressed in the number of DP cycles per task cycle), as described in the Synchronization [▶ 42] section.

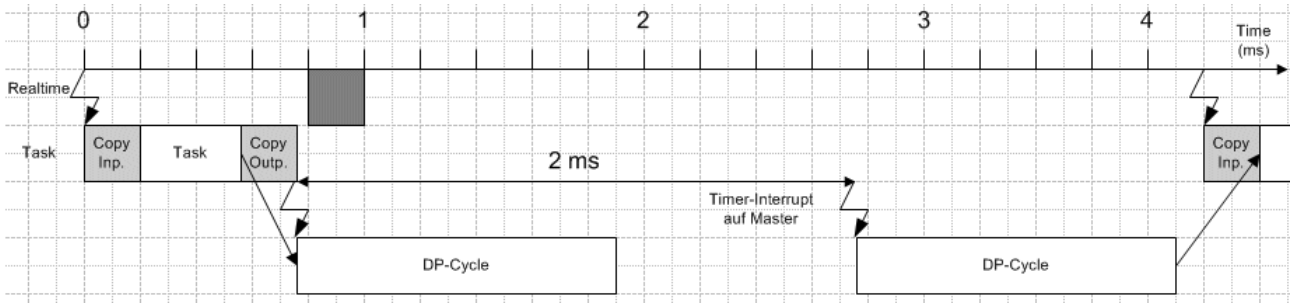



Fig. 123: Diagram showing several DP cycles in a task cycle

### Multiple data samples within one task cycle

The two functionalities just described can now be combined in order, for instance, to give one or more slaves in a 2 ms cycle new data every 1 ms, or to obtain new data from the slave in order to achieve better regulation quality. In this case, settings are made under **Additional Data_Exchange Samples** on the slave's Features [▶ 98] tab instead of under **Data-Exch Poll-Rate** (as described above).

In the sample shown below, first set the factor **Number of DP cycles per task cycle** in the **"FC310x"** tab (for TwinCAT 2.8/2.9) of the master to 2. So that the task can send 2 different values to the slave, or is able to receive 2 different values from the slave, the appropriate slave is to be entered into the System Manager

twice. All settings, with the exception of **Modulo** under **Additional Data_Exchange Samples** on the slave's Features [▶ 98] tab, must be the same. A 0 is entered here for the corresponding slave in one of the boxes and a 1 in the other box in the System Manager. The **Multiplier** under **Additional Data_Exchange Samples** is to be set for this slave in both boxes to 2, so that each of the two boxes that have been entered is only polled in every second DP cycle (the slave is in fact polled in each DP cycle, as it is entered twice). For all other slaves that are only to be polled once within the task cycle (and which of course are only therefore entered once in the System Manager), the **Multiplier** under **Additional Data_Exchange Samples** is also set to 2. **Modulo** under **Additional Data_Exchange Samples** can now be used to distribute these slaves over the two cycles. A slave that is polled twice but which is only to have one variable image in the task is only to be inserted once; the **Multiplier** would be set to 1, and **Modulo** to 0.

In the present example, slaves 1 and 2 would each be entered into the System Manager twice, with these settings:

- Additional Data_Exchange Samples/Multiplier = 2
- Additional Data_Exchange Samples/Modulo = 0 or 1

Slaves 3 and 5 would only be entered into the System Manager once, and would have these settings:

- Additional Data_Exchange Samples/Multiplier = 2
- Additional Data_Exchange Samples/Modulo = 0

Slaves 4 and 6 would also only be entered into the System Manager once, with these settings:

- Additional Data_Exchange Samples/Multiplier = 2
- Additional Data_Exchange Samples/Modulo = 1

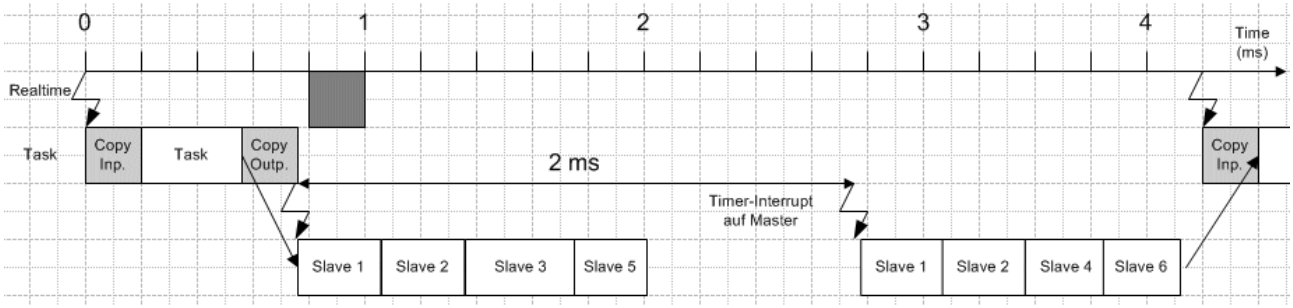



Fig. 124: Diagram showing several data samples within one task cycle

For slaves 1 and 2, the variables are present twice (in each case there 2 boxes in the System Manager). The variables associated with the box where Additional Data_Exchange Samples/Modulo is set to 0 are sent or received first.

## 5.2 EL6731-0010 - PROFIBUS slave terminal

The EL6731-0010 supports the PROFIBUS DP and PROFIBUS DPV1 protocols.

### PROFIBUS DP

In order to configure the slave for cyclic DP operation, proceed as follows in the TwinCAT system manager:

### Configure DP slave

It is first necessary to configure a "Profibus Slave EL6731, EtherCAT" I/O device (selecting "I/O devices" with the right hand mouse button, and then selecting "Insert device"). The device and a box are added (to this end the GSD file **"EL31095F.GSD"** must be in directory "TwinCAT\Io\Profibus"):

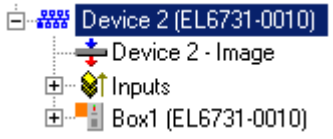



Fig. 125: Adding a „Profibus Slave EL6731, EtherCAT“ in the TwinCAT tree

Find the corresponding channel on the tab **"EL6731-0010"** of the device ("Search" button), adjust the station address and the baud rate, if necessary (the default setting is 12 Mbaud).

### Add modules

Modules are to be added to the box corresponding to the data that is to be transferred cyclically. This is done by clicking with the right mouse button on the box, and then selecting "Add modules":

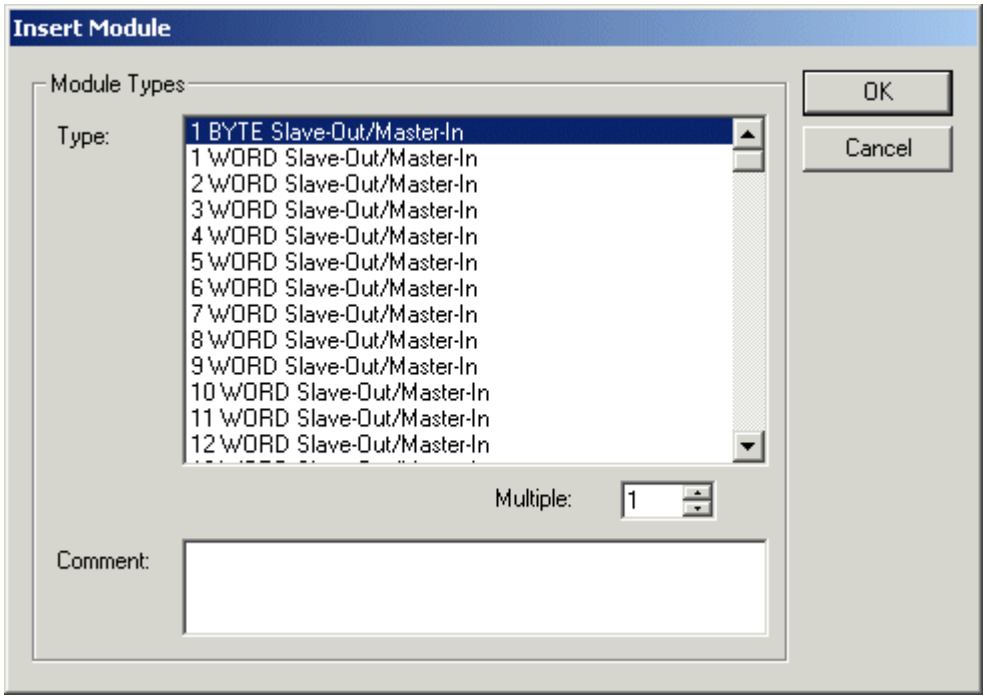



Fig. 126: Append modules

### Configuring the Master

The file "**EL31095F.GSE**" is to be taken out of the "TwinCAT\Io\Profibus" directory to configure the master. The modules must be inserted in the master configuration **in the same sequence** as they are in the configuration of the slave in the System Manager.

### PROFIBUS DPV1

The DP slave supports a DPV1-MSAC_C1 server connection that is established along with the cyclic connection. This can be used so that larger quantities of acyclic data can be transferred alongside the cyclic data. A DPV1 read telegram received by the master is reported to the PLC as an ADS read indication, while a DPV1 write telegram is reported to the PLC as an ADS write indication. The PLC program is then responsible for the read or write response. To do this, the ADS read response or ADS write response functions are to be called.

### MSAC-C1 Read

A DPV1-MSAC_C1 read indication is represented in an ADS read indication as follows:

| ADS read indication parameter | Meaning |
|---|---|
| Source-Net-ID (NETID) | Net-ID of the slave (see the device's ADS [▶ 93] tab) |
| Source-Port (PORT) | 0x200 |
| Invoke-ID (INVOKEID) | A unique number that must reappear in the response |
| IndexGroup (IDXGRP) | Slot number (DPV1 parameter) |
| IndexOffset (IDXOFFS) | Index (DPV1 parameter) |
| Length (LENGTH) | Length of the data that is to be read |

An ADS read response is represented in a DPV1-MSAC_C1 read response as follows:

| ADS read response parameter | Meaning |
|---|---|
| Destination-Net-ID (NETID) | Net-ID of the slave (see the device's ADS [▶ 93] tab) |
| Destination-Port (PORT) | 0x200 |
| Invoke-ID (INVOKEID) | A unique number, as under indication |
| Result (RESULT) | Result of the read: 0 = no error, otherwise: bits 0-15 = standard ADS error codes, bits 16-23 = Error_Code_1, bits 24-31 = Error_Code_2, see description of the DPV1 error codes [▶ 127] |
| Length (LENGTH) | Length of the data that has been read |
| Data (DATAADDR) | read data |

### MSAC-C1 Write

A DPV1-MSAC_C1 write indication is represented in an ADS write indication as follows:

| ADS write indication parameter | Meaning |
|---|---|
| Source-Net-ID (NETID) | Net-ID of the slave (see the device's ADS [▶ 93] tab) |
| Source-Port (PORT) | 0x200 |
| Invoke-ID (INVOKEID) | A unique number that must reappear in the response |
| IndexGroup (IDXGRP) | Slot number (DPV1 parameter) |
| IndexOffset (IDXOFFS) | Index (DPV1 parameter) |
| Length (LENGTH) | Length of the data that is to be written |
| Data (DATAADDR) | data that is to be written |

An ADS read response is represented in a DPV1-MSAC_C1 read response as follows:

| ADS read response parameter | Meaning |
|---|---|
| Destination-Net-ID (NETID) | Net-ID of the slave (see the device's ADS [▶ 93] tab) |
| Destination-Port (PORT) | 0x200 |
| Invoke-ID (INVOKEID) | A unique number, as under indication |
| Result (RESULT) | Result of the read: 0 = no error, otherwise: bits 0-15 = standard ADS error code, bits 16-23 = Error_Code_1, bits 24-31 = Error_Code_2, see description of the DPV1 error codes [▶ 127] |
| Length (LENGTH) | Length of the data that has been read |

### ADS Interface

Communication can also take place via ADS in TwinCAT systems. The functionality is extremely similar to that of an ADS connection between two PCs over Ethernet, although transmission takes place over PROFIBUS, with the exception that the requester that initiates the ADS job is always the DP master PC. An EL6731 DP master is then linked to an EL6731-DP slave.

In the DP master, the **ADS Interface** should be activated on the "ADS" tab of the box, and the **Net-ID** of the DP slave PC is to be entered:

Fig. 127: DP master, ADS tab: Activating the ADS interface and entering the Net ID

At the DP slave the Net-ID of the DP master PC is to be added under **Add. NetIds** on the device's "ADS" tab:

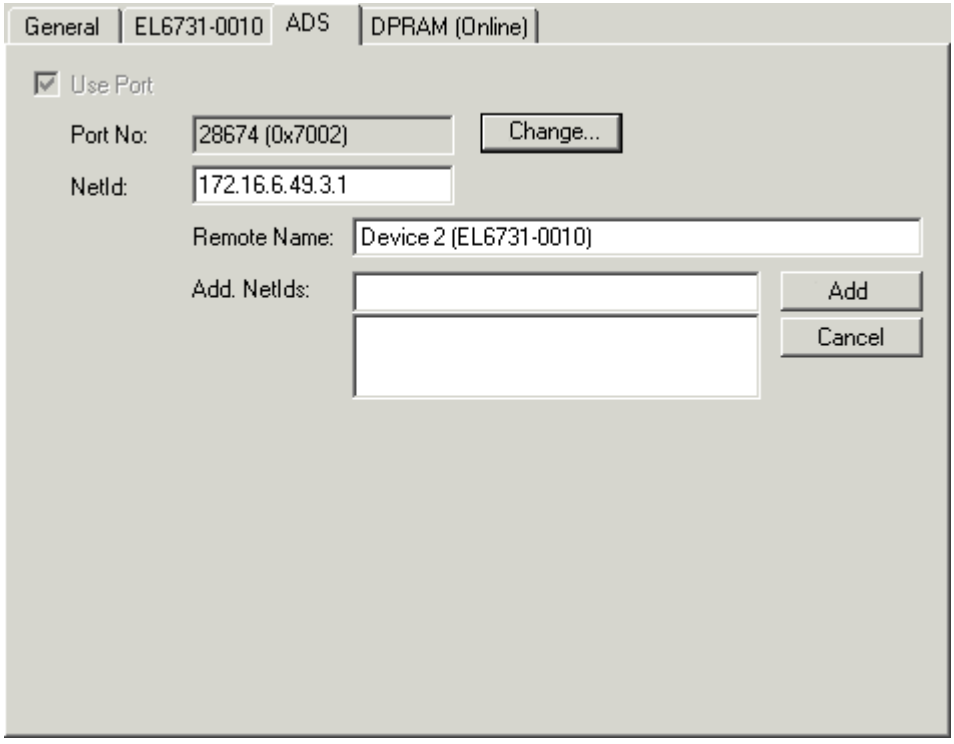



Fig. 128: DP slave, ADS tab: Entering the DP master Net ID with “Add. Net IDs”

# 6 EtherCAT communication EL6731-00x0

## 6.1 PROFIBUS master

### 6.1.1 State machine

The EL6731 can be configured in several ways:

1. Configuration of the EL6731 with StartUp SDOs [▶ 135]: Here, the StartUp SDOs are calculated in the EtherCAT configurator and transferred to the EtherCAT master, in the same way as is carried out, for example, in the TwinCAT System Manager.
2. Configuration of the EL6731 with Backup Parameter Storage [▶ 137]: Here, the configuration of the DP slave is stored in the flash memory of the EL6751 and need only be transmitted once.

**Configuration of the EL6731 with StartUp SDOs:**

The following flow chart shows the sequence of the configuration of the EL6731 with StartUp SDOs:

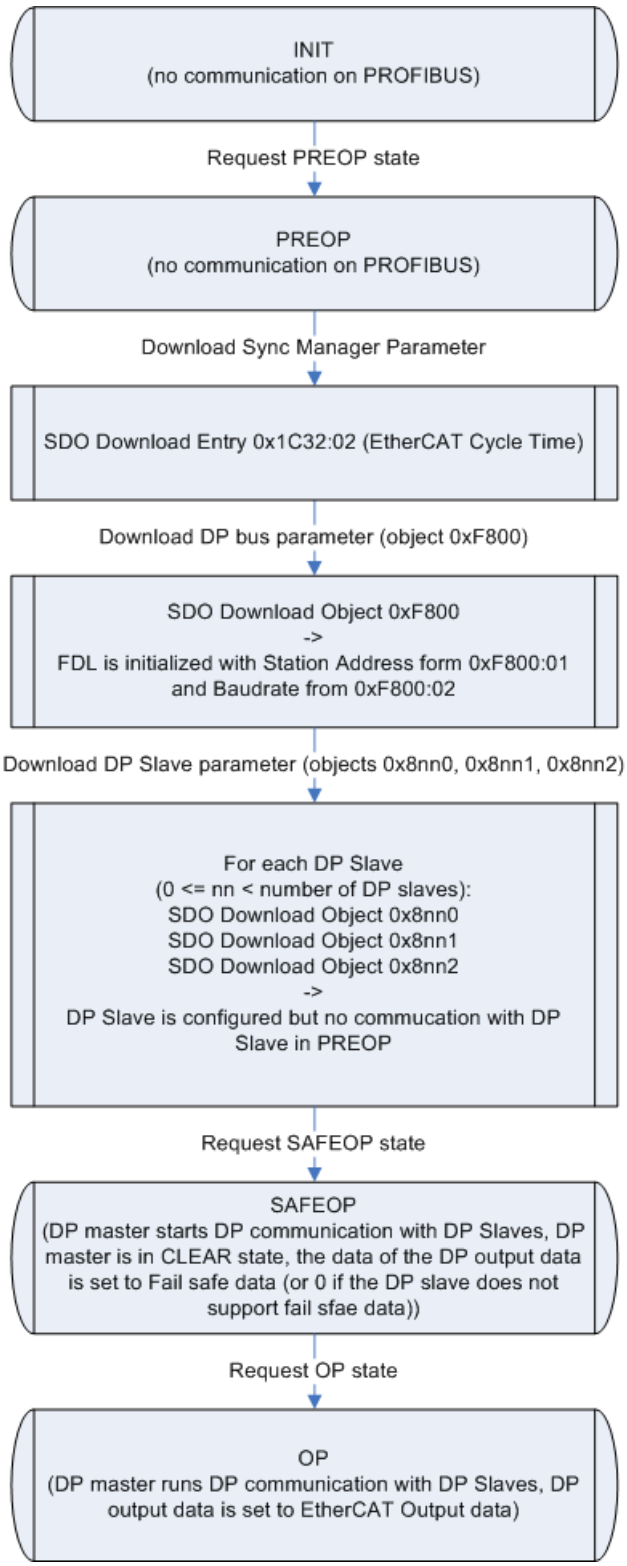



Fig. 129: Flow chart: Sequence of the configuration of the EL6731 with start SDOs

Following a power-on, the EL6731 is in the INIT state and has no DP configuration.  The EL6731 is not active on the PROFIBUS.

### DP bus parameters

The DP configuration is carried out via SDO download in the PREOP state. The objects to be loaded must be transmitted either with Complete Access or with consistency nesting (first set SubIndex 0 to 0, then write SubIndex 1-n, then set SubIndex 0 to n). Care should thereby be taken to always start with object 0xF800 [▶ 159]. After the receipt of object 0xF800, the EL6731 is active on the PROFIBUS with the corresponding baud rate from 0xF800:02 (FDL only, no DP).

### DP slave configuration

After object 0xF800, the objects 0x8yy0 [▶ 155], 0x8yy1 [▶ 156] and 0x8yy2 [▶ 156] must be transferred in this order, depending on the DP slave to be configured. yy is to be incremented (starting from 0) for each DP slave to be configured.

### PDO Mapping

For each configured DP slave there is an EtherCAT RxPDO (if the DP slave has DP outputs) and an EtherCAT TxPDO (if the DP slave has DP inputs). The PDO mapping of the EtherCAT PDOs is automatically calculated by the EL6731 after the download of the respective 0x8yyz objects and can be read. The PDO mapping objects 0x16yy and 0x1Ayy thereby belong to the configuration objects 0x8yyz. The PDO mapping objects can only be written with the values that the EL6731 has calculated itself. The writing of the PDO mapping thus serves only to check the PDO mapping calculated by the EtherCAT configurator and can therefore be omitted.

### PDO Assign

In addition, there are a few other EtherCAT PDOs that contain control, status and diagnostic information. These PDOs are selected via the PDO Assign. It should thereby be ensured that all EtherCAT PDOs that are assigned to the configured DP slaves (PDO number <= 125) always appear in the PDO Assign. With regard to the order of the PDOs in the PDO Assign, it is important to ensure that the index of the assigned EtherCAT PDO increases with each entry in the corresponding PDO Assign object. If the EtherCAT master does not transmit any PDO Assign in the StartUp SDOs, then the PDOs 0x1A81 [▶ 147], 0x1A82 [▶ 148], 0x1A83 [▶ 148], 0x1A84 [▶ 148] and 0x1A85 [▶ 148] are assigned for status and diagnosis.

### Cyclic DP communication

During the transition to SAFEOP, the EL6731 checks the length configured in the Sync Manager channels 2 and 3 against the length calculated from PDO Mapping and PDO Assign. The SAFEOP state is only adopted if these lengths match. The EL6731 starts communication with the configured DP slaves in the SAFEOP state. Fail Safe data are sent to the DP slaves as long as the EL6731 is in SAFEOP. As soon as the EL6731 has been switched to OP, the data from the EtherCAT outputs are transmitted to the DP slaves.

### Configuration of the EL6731 with Backup Parameter Storage

The following flow chart shows the sequence of the configuration of the EL6731 with Backup Parameter Storage:

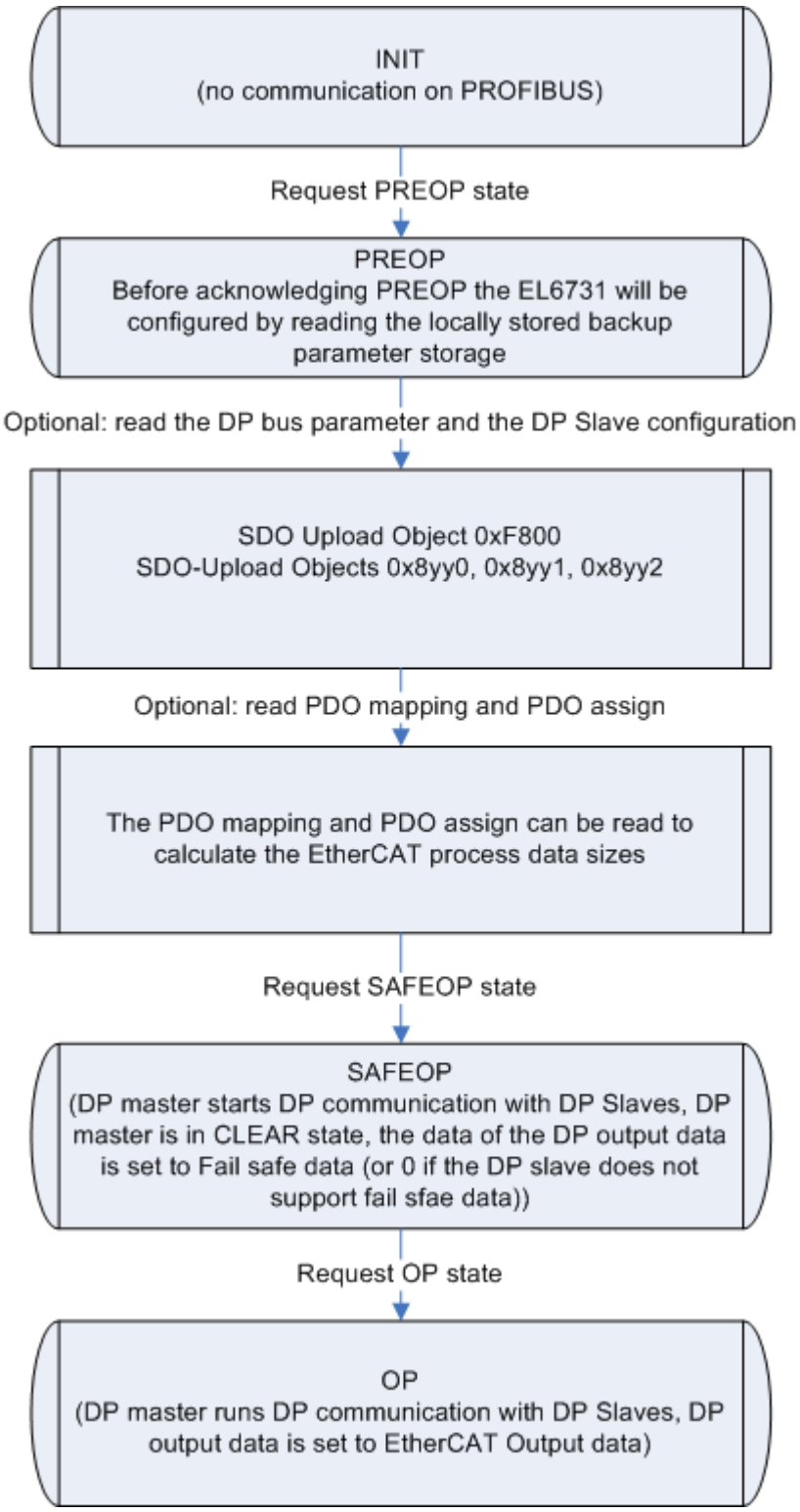



Fig. 130: Flow chart: Sequence of the configuration of the EL6731 with backup parameter storage

Following a power-on, the EL6731 is in the INIT state and has no DP configuration. The EL6731 is not active on the PROFIBUS.

**DP bus parameters / DP slave configuration**

The configuration stored in the Backup Parameter Storage object 0x10F2 is loaded during the transition from INIT to PREOP. Since the StartUp SDOs from the Configuration of the EL6731 with StartUp SDOs [▶ 135] are stored in the Backup Parameter Storage object, the sequence is the same as the one described there. Hence, the stored data is initially written to object 0xF800 and the EL6731 is active on the PROFIBUS with the corresponding baud rate from 0xF800:02. Subsequently, the DP slaves are generated according to the

stored DP slave configuration. If the PREOP state is quit, the current DP configuration in objects 0xF800, 0x8yy0, 0x8yy1 and 0x8yy2 can be read. The cycle time of the EtherCAT master (0x1C32:02) must still be transmitted before the EL6731 is switched to SAFEOP.

#### PDO Mapping / PDO Assign

In addition, the EtherCAT master can also read the PDO Mapping and PDO Assign in the PREOP state in order to determine the lengths of the EtherCAT process data.

#### Creating the Backup Parameter Storage

The Backup Parameter Storage can be created as follows:

1. Download object 0x10F2 (in PREOP without having sent StartUp SDOs beforehand): in this case the data received will be saved as Backup Parameter Storage in the flash memory

#### Deleting the Backup Parameter Storage

In order to load a new Backup Parameter Storage or to simply delete the existing one, the value 0x64616F6C has to be written to entry 0x1011:01.

### 6.1.2 Synchronization

In the EL 6731, the DP cycle is synchronized with the EtherCAT cycle. Synchronization takes place by default via the Sync Manager 2 event or, if there is no EtherCAT output process data, via the Sync Manager 3 event. Alternatively, the EL6731 can also be operated in the Distributed Clocks mode, in which case synchronization takes place via the SNYC0 or SYNC1 events.

#### SM-synchronous

The illustration below shows the sequence of the DP cycle if no Distributed Clocks are used.

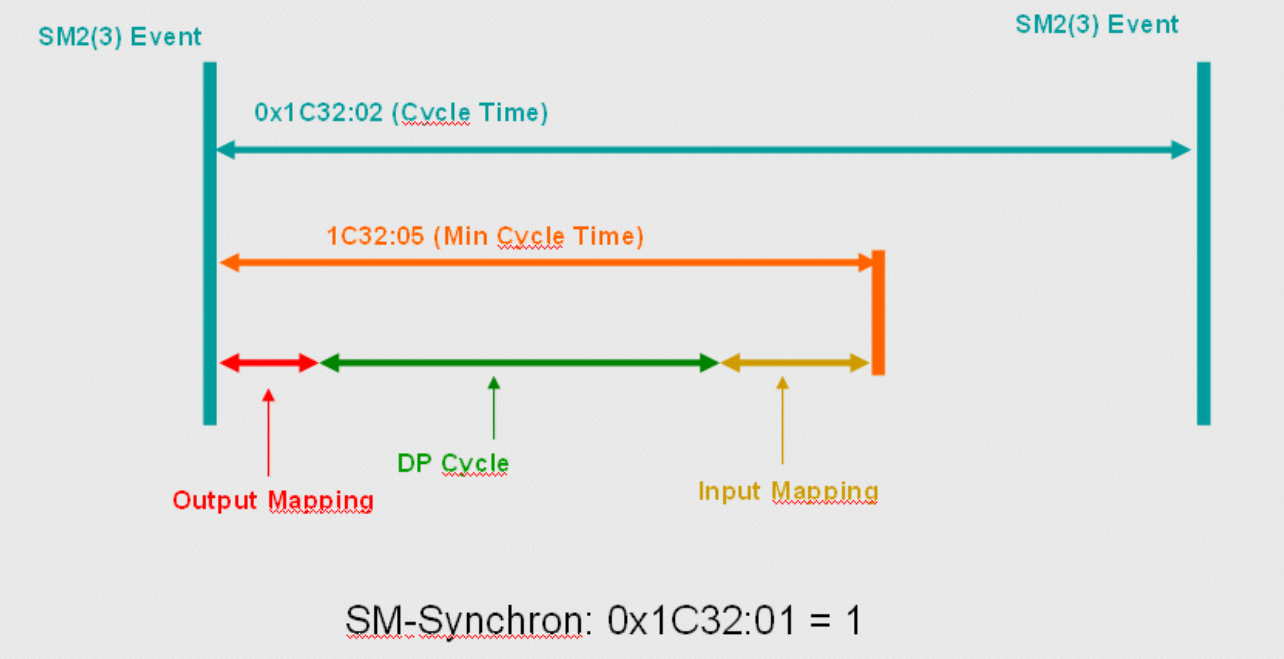



Fig. 131: Sequence of the DP cycle without distributed clocks

When receiving the EtherCAT process data telegram, the SM2 event (SM3 if no EtherCAT output data are configured, i.e. only DP slaves without DP outputs) is generated by the EtherCAT slave controller, thus starting the DP cycle. A data exchange telegram with the output process data received over EtherCAT is thereby sent for each configured DP Slave (that is in the data exchange state). The sequence corresponds to the sequence of the configuration, i.e. it starts with the DP slave configured in the objects 0x800z. Once all

configured DP slaves have been dealt with, the EtherCAT input data is updated and the DP cycle is finished. If the next SM2 (SM3) event is received before the DP cycle is completed, the Cycle Exceed counter (0x1C32:0B or 0x1C33:0B) is incremented and one DP cycle is skipped.

### SYNC0-synchronous

The DP cycle is started by the SM2 (SM3) event. The sending of the first telegram is delayed until the SYNC0 event occurs, so that the sending of the Global Control telegram takes place with a jitter of maximum 500 ns. The remainder of the DP cycle sequence corresponds to that in the case of synchronization without Distributed Clocks.

### Transmission of the process data with LRW telegram (Separate Input Update = FALSE)

The illustration below shows the sequence of the DP cycle and the meaning of the Sync Manager parameters if the DP cycle is controlled with Distributed Clocks via SM and SYNC0 event.

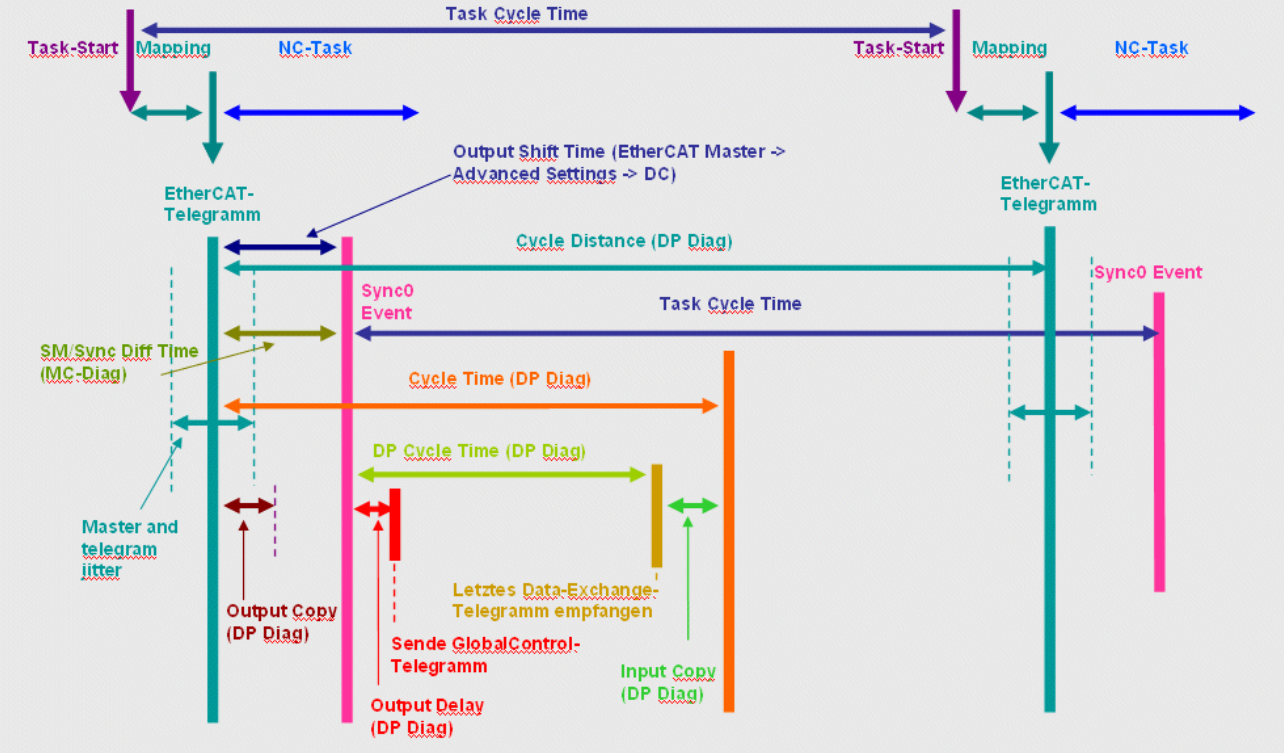



Fig. 132: DP cycle with distributed clocks and control via SM and SYNC0 event

### Transmission of the output data with LWR telegram and the input data with LRD telegram (Separate Input Update = TRUE, Task Cycle Time = Base Time)

If EtherCAT outputs and inputs are transmitted with separate telegrams, so that the inputs are as up to date as possible (click on Separate Input Update for the associated TwinCAT task in the TwinCAT System Manager), there is less room for the DP cycle. If the Task Cycle Time (= EtherCAT Master Cycle Time) is equal to the TwinCAT Realtime Base Time, the LRD telegram, with which the inputs are read in as late as possible, is sent in accordance with the set CPU limit.

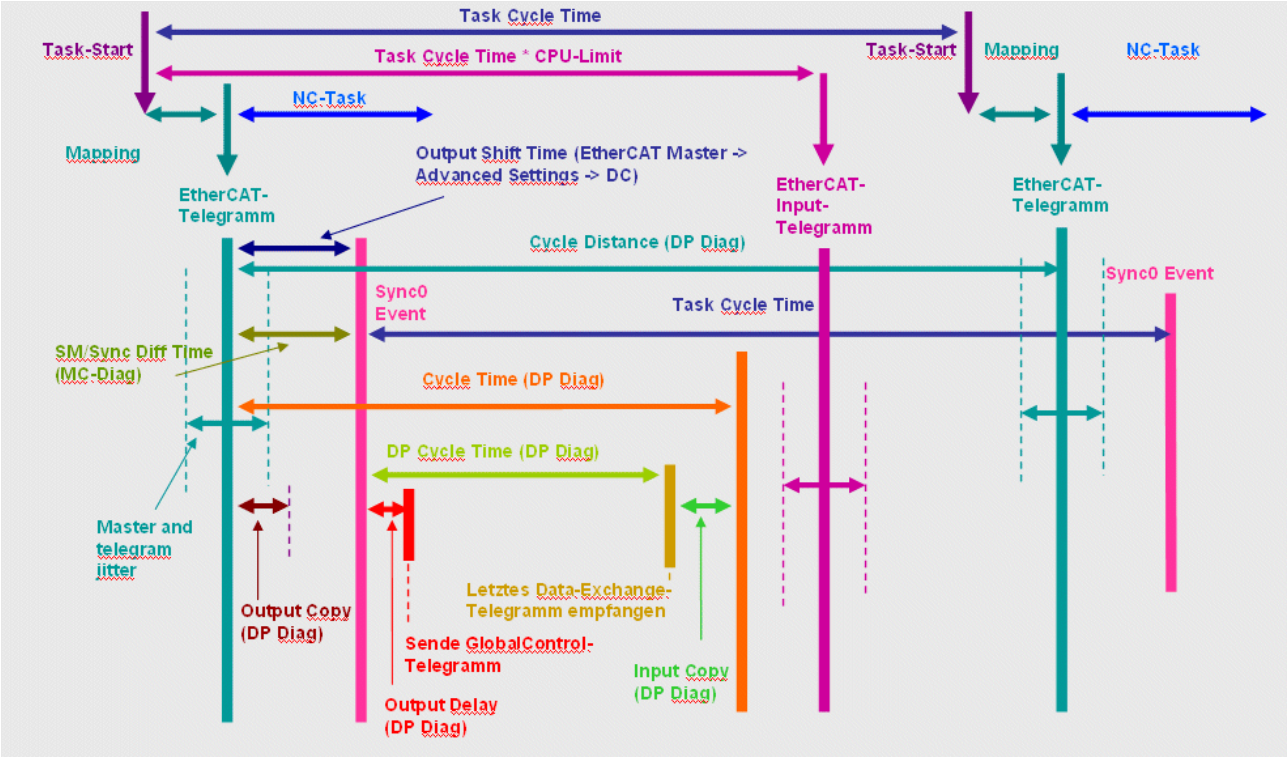



Fig. 133: Transfer of output data with LWR, input data with LRD telegram (Separate Input Update = TRUE, Task Cycle Time = Base Time)

**Transmission of the output data with LWR telegram and the input data with LRD telegram (Separate Input Update = TRUE, Task Cycle Time = 4*Base Time, pre-ticks = 1)**

If the Task Cycle Time (= EtherCAT Master Cycle Time) is larger than the TwinCAT Realtime Base Time, then the LRD telegram is sent at a previous Base Time in accordance with the set pre-tick.

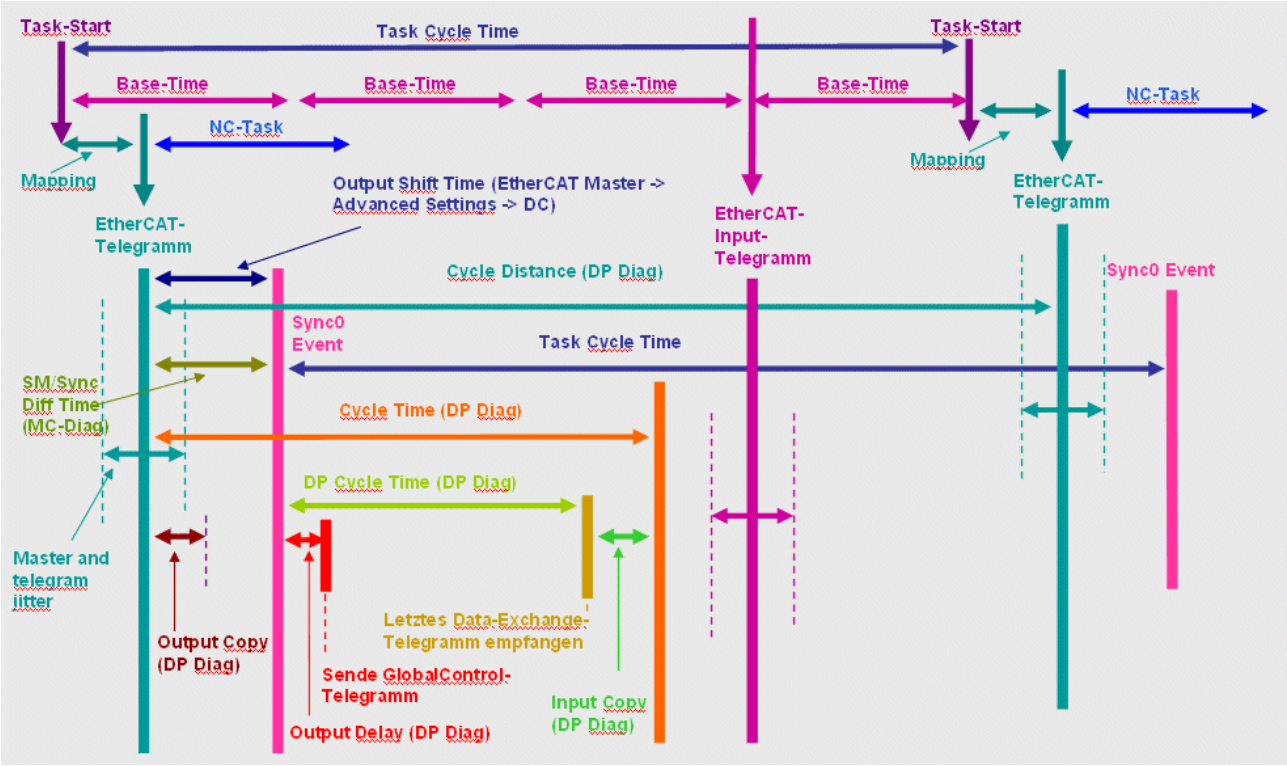



Fig. 134: Transfer of output data with LWR, input data with LRD telegram (Separate Input Update = TRUE, Task Cycle Time = 4*Base Time, Pre ticks = 1)

### SYNC1-synchronous

The DP cycle is started through the SYNC0 event. Before the first telegram is sent, the system waits for the SYNC1 event, so that the global control telegram is sent with a maximum jitter of 500 ns. The remainder of the DP cycle sequence corresponds to that in the case of synchronization without Distributed Clocks.

The same settings for Separate Input Update apply as in the case of synchronization with SYNC0, therefore only the case without Separate Input Update is illustrated here:

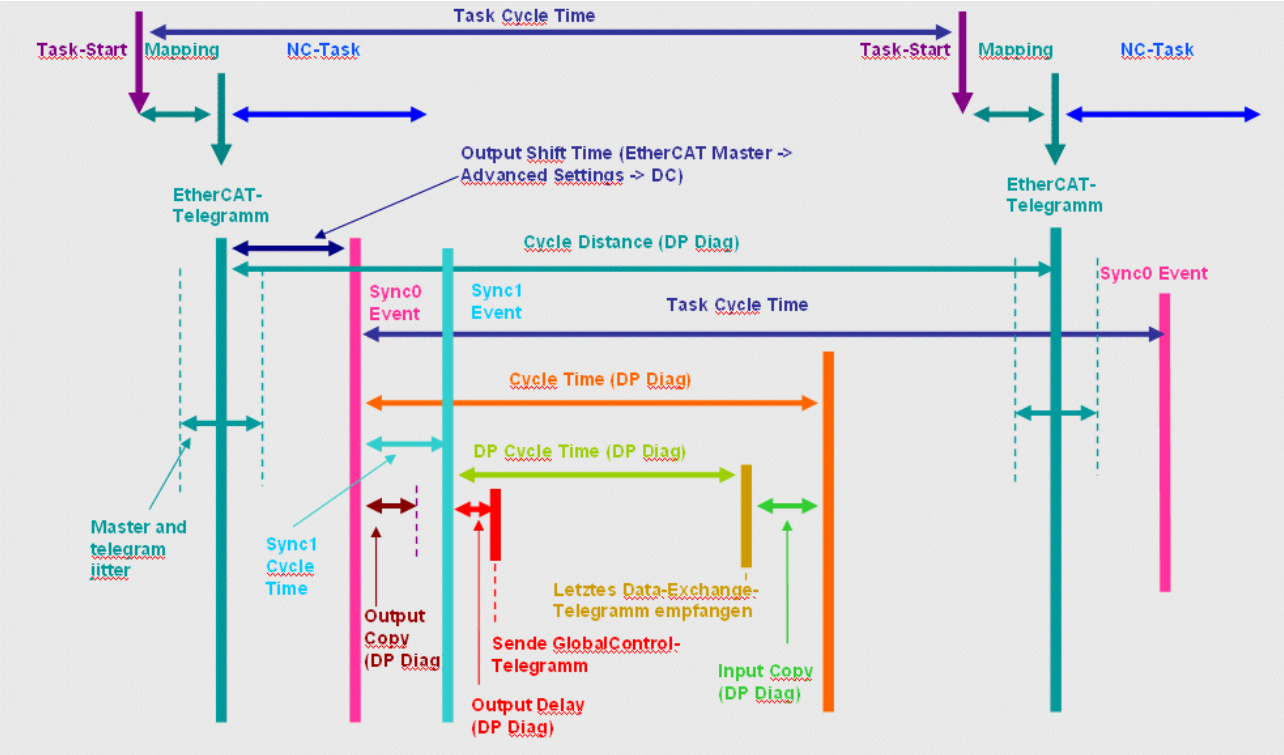



Fig. 135: DP cycle start via SYNC0, first telegram is sent after the SYNC1 event

## 6.1.3 Object description and parameterization

**EtherCAT XML Device Description**

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

**Parameterization via the CoE list (CAN over EtherCAT)**

The EtherCAT device is parameterized via the CoE-Online tab (double-click on the respective object) or via the Process Data tab (allocation of PDOs). Please note the following general CoE notes [▶ 33] 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

### 6.1.3.1 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 | 0x0C1C1389 ($203166601_{dec}$) |

#### Index 1008 Device name

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1008:0 | Device name | Device name of the EtherCAT slave | STRING | RO | EL6731 |

#### Index 1009 Hardware version

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1009:0 | Hardware version | Hardware version of the EtherCAT slave | STRING | RO | 06 |

#### Index 100A Software version

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 100A:0 | Software version | Firmware version of the EtherCAT slave | STRING | RO | 10 |

#### Index 1011 Restore default parameters

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

### Index 1018 Identity

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1018:0 | Identity | Information for identifying the slave | UINT8 | RO | 0x04 ($4_{dec}$) |
| 1018:01 | Vendor ID | Vendor ID of the EtherCAT slave | UINT32 | RO | 0x00000002 ($2_{dec}$) |
| 1018:02 | Product code | Product code of the EtherCAT slave | UINT32 | RO | 0x1A4B3052 ($441135186_{dec}$) |
| 1018:03 | Revision | Revision number of the EtherCAT slave; the low word (bit 0-15) indicates the special terminal number, the high word (bit 16-31) refers to the device description | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1018:04 | Serial number | Serial number of the EtherCAT slave; the low byte (bit 0-7) of the low word contains the year of production, the high byte (bit 8-15) of the low word contains the week of production, the high word (bit 16-31) is 0 | UINT32 | RO | 0x00000000 ($0_{dec}$) |

### Index 10F0 Backup parameter handling

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 10F0:0 | Backup parameter handling | Information for standardized loading and saving of backup entries | UINT8 | RO | 0x01 ($1_{dec}$) |
| 10F0:01 | Checksum | Checksum of the Backup Parameter Storage (object 0x10F2 [▶ 144], word 2-3) | UINT32 | RO | 0x00000000 ($0_{dec}$) |

### Index 10F2 Backup parameter storage

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 10F2:0 | Backup parameter storage | If this object is used, no StartUp SDOs may be transmitted in the PREOP state, since the Backup Parameter Storage contains the complete StartUp SDOs (see Configuration of the EL6731 with Backup Parameter Storage [▶ 137]). The EL6731 is rebooted 5s after the flashing of the Backup Parameter Storage (switches to INIT with AL status code = 0x60). The data have the following meaning: | OCTET-STRING[n] | RW | |

| Word-Offset | Description |
|---|---|
| 0 | Command: with 0xC0DE, the received data are stored in the flash memory |
| 1 | Length of the data from word offset 4 in bytes |
| 2-3 | Checksum, which is calculated locally |
| 4 | Index of the object of the 1st StartUp SDO |
| 5 | len1: Length of the object of the 1st StartUp SDO as CompleteAccess (from SubIndex 0) in bytes |
| 6-n1 | Data of the object of the 1st StartUp SDO as CompleteAccess (n1 = 2*((len1+1)/2)+5) |
| n1+1 | Index of the object of the 2nd StartUp SDO |
| n1+2 | len2: Length of the object of the 2nd StartUp SDO as CompleteAccess (from SubIndex 0) in bytes |
| (n1+3)-n2 | Data of the object of the 2nd StartUp SDO as CompleteAccess (n2 = 2*((len2+1)/2)+n1+2) |
| ... | |
| m | Index of the object of the 3rd StartUp SDO |
| m+1 | len3: Length of the object of the 3rd StartUp SDO as CompleteAccess (from SubIndex 0) in bytes |
| (m+2)-n3 | Data of the object of the 3rd StartUp SDO as CompleteAccess (n3 = 2*((len3+1)/2)+m+1) |

### Index 10F3 Diagnosis History

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 10F3:0 | Diagnosis History | Maximum subindex | UINT8 | RO | 0x45 ($69_{dec}$) |
| 10F3:01 | Maximum Messages | Maximum number of stored messages<br>A maximum of 50 messages can be stored | UINT8 | RO | 0x00 ($0_{dec}$) |
| 10F3:02 | Newest Message | Subindex of the latest message | UINT8 | RO | 0x00 ($0_{dec}$) |
| 10F3:03 | Newest Acknowledged Message | Subindex of the last confirmed message | UINT8 | RW | 0x00 ($0_{dec}$) |
| 10F3:04 | New Messages Available | Indicates that a new message is available | BOOLEAN | RO | 0x00 ($0_{dec}$) |
| 10F3:05 | Flags | not used | UINT16 | RW | 0x0000 ($0_{dec}$) |
| 10F3:06 | Diagnosis Message 001 | Message 1 | OCTET-STRING[20] | RO | {0} |
| ... | ... | ... | ... | ... | ... |
| 10F3:45 | Diagnosis Message 064 | Message 64 | OCTET-STRING[20] | RO | {0} |

### Index 10F8 Actual Time Stamp

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 10F8:0 | Actual Time Stamp | Time stamp | UINT64 | RO | |

### Index 1600-167C RxPDO-Map DP-Slave yyy

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1600+n:0 | RxPDO-Map DP-Slave yyy | There is an RxPDO containing the DP output data of the DP slave for each configured DP slave. There is an entry in the RxPDO mapping object 0x1600+n for each DP module with DP output data configured in object 0x8002 [▶ 156]+(n*16). If a DP slave does not contain any DP output data, then the PDO mapping object 0x1600+n does not exist. These PDOs are mandatory and must always be contained in the PDO Assign object 0x1C12 [▶ 150], depending on the configured DP slaves. SubIndex 0 contains the number of DP modules with DP output data of the $(n+1)^{th}$ configured DP slave. | UINT8 | RO | |
| (1600+n):01 | | first DP module with DP output data of the $(n+1)^{th}$ configured DP slave | UINT32 | RO | |
| ... | | .. | | | |
| (1600+n):FF | | last DP module with DP output data of the $(n+1)^{th}$ configured DP slave | UINT32 | RO | |

### Index 167F DPM RxPDO-Map Control

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 167F:0 | DPM RxPDO-Map Control | PDO Mapping RxPDO 128 | UINT8 | RO | 0x02 ($2_{dec}$) |
| 167F:01 | SubIndex 001 | $1^{st}$ PDO Mapping entry (object 0xF200, entry 0x01) | UINT32 | RO | 0xF200:01, 1 |
| 167F:02 | SubIndex 002 | $2^{nd}$ PDO Mapping entry (15 bits align) | UINT32 | RO | 0x0000:00, 15 |

### Index 187F DPM TxPDO-Par Status PDO

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 187F:0 | DPM TxPDO-Par Status PDO | PDO parameter TxPDO 128 | UINT8 | RO | 0x06 ($6_{dec}$) |
| 187F:06 | Exclude TxPDOs | Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 128 | OCTET-STRING[10] | RO | 80 1A 00 00 00 00 00 00 00 00 |

### Index 1880 DPM TxPDO-Par Status PDO

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1880:0 | DPM TxPDO-Par Status PDO | PDO parameter TxPDO 129 | UINT8 | RO | 0x06 ($6_{dec}$) |
| 1880:06 | Exclude TxPDOs | Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 129 | OCTET-STRING[10] | RO | 81 1A 82 1A 83 1A 84 1A 85 1A |

### Index 1881 DPM TxPDO-Par PDO State

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1881:0 | DPM TxPDO-Par PDO State | PDO parameter TxPDO 130 | UINT8 | RO | 0x06 ($6_{dec}$) |
| 1881:06 | Exclude TxPDOs | Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 130 | OCTET-STRING[10] | RO | 80 1A 00 00 00 00 00 00 00 00 |

### Index 1882 DPM TxPDO-Par PDO Toggle

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1882:0 | DPM TxPDO-Par PDO Toggle | PDO parameter TxPDO 131 | UINT8 | RO | 0x06 ($6_{dec}$) |
| 1882:06 | Exclude TxPDOs | Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 131 | OCTET-STRING[10] | RO | 80 1A 00 00 00 00 00 00 00 00 |

### Index 1883 DPM TxPDO-Par DP Diag

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1883:0 | DPM TxPDO-Par DP Diag | PDO parameter TxPDO 132 | UINT8 | RO | 0x06 ($6_{dec}$) |
| 1883:06 | Exclude TxPDOs | Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 132 | OCTET-STRING[10] | RO | 80 1A 00 00 00 00 00 00 00 00 |

### Index 1884 DPM TxPDO-Par DP Status

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1884:0 | DPM TxPDO-Par DP Status | PDO parameter TxPDO 133 | UINT8 | RO | 0x06 ($6_{dec}$) |
| 1884:06 | Exclude TxPDOs | Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 133 | OCTET-STRING[10] | RO | 80 1A 00 00 00 00 00 00 00 00 |

### Index 1885 DPM TxPDO-Par DP-Master Diag

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1885:0 | DPM TxPDO-Par DP-Master Diag | PDO parameter TxPDO 134 | UINT8 | RO | 0x06 ($6_{dec}$) |
| 1885:06 | Exclude TxPDOs | Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 134 | OCTET-STRING[10] | RO | 80 1A 00 00 00 00 00 00 00 00 |

### Index 1886 DPM TxPDO-Par Cycle Statistics

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1886:0 | DPM TxPDO-Par Cycle Statistics | PDO parameter TxPDO 135 | UINT8 | RO | 0x06 ($6_{dec}$) |
| 1886:06 | Exclude TxPDOs | Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 135 | OCTET-STRING[10] | RO | 80 1A 87 1A 00 00 00 00 00 00 |

### Index 1887 DPM TxPDO-Par DC Cyclic Statistics

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1887:0 | DPM TxPDO-Par DC Cyclic Statistics | PDO parameter TxPDO 136 | UINT8 | RO | 0x06 ($6_{dec}$) |
| 1887:06 | Exclude TxPDOs | Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 136 | OCTET-STRING[10] | RO | 80 1A 86 1A 00 00 00 00 00 00 |

### Index 1A00-1A7C DPS TxPDO-Map Slave yyy

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A00+n:0 | DPS TxPDO-Map Slave yyy | There is one TxPDO containing the DP input data of the DP slave for each configured DP slave. There is an entry in the TxPDO mapping object 0x1A00+n for each DP module with DP input data configured in object 0x8002 [▶ 156]+(n*16). If a DP slave does not contain any DP input data, then the PDO mapping object 0x1A00+n does not exist. These PDOs are mandatory and must always be contained in the PDO Assign object 0x1C13 [▶ 150], depending on the configured DP slaves. SubIndex 0 contains the number of DP modules with DP input data of the (n+1)$^{th}$ configured DP slave. | UINT8 | RO | |
| (1A00+n):01 | | first DP module with DP input data of the (n+1)$^{th}$ configured DP slave | UINT32 | RO | |
| ... | | .. | | | |
| (1A00+n):FF | | last DP module with DP input data of the (n+1)$^{th}$ configured DP slave | UINT32 | RO | |

### Index 1A7F DPM TxPDO-Map Status PDO

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A7F:0 | DPM TxPDO-Map Status PDO | PDO Mapping TxPDO 128 | UINT8 | RO | 0x03 ($3_{dec}$) |
| 1A7F:01 | SubIndex 001 | 1$^{st}$ PDO Mapping entry (object 0xF100, entry 0x01) | UINT32 | RO | 0xF100:01, 8 |
| 1A7F:02 | SubIndex 002 | 2$^{nd}$ PDO Mapping entry (7 bits align) | UINT32 | RO | 0x0000:00, 7 |
| 1A7F:03 | SubIndex 003 | 3$^{rd}$ PDO Mapping entry (object 0xF100, entry 0x0F) | UINT32 | RO | 0xF100:0F, 1 |

### Index 1A80 DPM TxPDO-Map PDO Status

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A82:0 | DPM TxPDO-Map PDO Status | This PDO is assigned if Legacy Mapping (no MDP) is active | UINT8 | RO | |
| 1A82:01 | | first PDO mapping entry in the case of Legacy Mapping | UINT32 | RO | |
| ... | | | | | |
| 1A82:m | | last PDO mapping entry in the case of Legacy Mapping | UINT32 | RO | |

### Index 1A81 DPM TxPDO-Map PDO State

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A81:0 | DPM TxPDO-Map PDO State | In this PDO there is a bit for each configured DP slave that is set if the latter is not in exchanging data (a more detailed error cause can be found in 0xF102 [▶ 158]:m for the m$^{th}$ configured DP slave). If the bit is set, the data of the associated TxPDO m is to be ignored. This PDO is optional. | UINT8 | RO | |
| 1A81:01 | | PDO state of the first configured DP slave (configured via the objects 0x800z) | UINT32 | RO | |
| ... | | | | | |
| 1A81:m | | PDO state of the last (m$^{th}$) configured DP slave (configured via the objects 0x800z+(m-1)*16 (1 <= m <= 127)) | UINT32 | RO | |

### Index 1A82 DPM TxPDO-Map PDO Toggle

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A82:0 | DPM TxPDO-Map PDO Toggle | For each configured DP slave there is a bit in this PDO that toggles every time new DP input data has been received from the DP slave and updated in the EtherCAT input data. This PDO is optional. | UINT8 | RO | |
| 1A82:01 | | PDO toggle of the first configured DP slave (configured via the objects 0x800z) | UINT32 | RO | |
| ... | | | | | |
| 1A82:m | | PDO toggle of the last ($m^{th}$) configured DP slave (configured via the objects 0x800z+(m-1)*16 (1 <= m <= 127)) | UINT32 | RO | |

### Index 1A83 DPM TxPDO-Map DP Status

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A83:0 | DPM TxPDO-Map DP Status | In this PDO there is a byte for each configured DP slave that contains the communication state (object 0xF102 [▶ 158]) to the DP slave. This PDO is optional. | UINT8 | RO | Number of configured DP slaves |
| 1A83:01 | | Node state of the first configured DP slave (configured via the objects 0x800z) | UINT32 | RO | 0xF102:01,8 |
| ... | | | | | |
| 1A83:m | | Node state of the last ($m^{th}$) configured DP slave (configured via the objects 0x800z+(m-1)*16 (1 <= m <= 127)) | UINT32 | RO | 0xF102:m,8 |

### Index 1A84 DPM TxPDO-Map DP Diag

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A84:0 | DPM TxPDO-Map DP Diag | In this PDO there is a bit for each configured DP slave that is set if the diagnostic information (object 0xF103) has changed. This PDO is optional. | UINT8 | RO | Number of configured DP slaves |
| 1A84:01 | | Diag Flag of the first configured DP slave (configured via the objects 0x800z) | UINT32 | RO | 0xF103:01,1 |
| ... | | | | | |
| 1A84:FF | | Diag Flag of the last ($m^{th}$) configured DP slave (configured via the objects 0x800z+(m-1)*16 (1 <= m <= 127)) | UINT32 | RO | 0xF103:m,1 |

### Index 1A85 DPM TxPDO-Map DP-Master Diag

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A85:0 | DPM TxPDO-Map DP-Master Diag | PDO Mapping TxPDO 134 contains the DP master status (object 0xF101 [▶ 157]) | UINT8 | RW | 0x0D ($13_{dec}$) |
| 1A85:01 | SubIndex 001 | $1^{st}$ PDO Mapping entry (object 0xF101 (DP Master Status data), entry 0x01 (Bus Error Counter)) | UINT32 | RW | 0xF101:01, 16 |
| 1A85:02 | SubIndex 002 | $2^{nd}$ PDO Mapping entry (object 0xF101 (DP Master Status data), entry 0x02 (Cycle Counter)) | UINT32 | RW | 0xF101:02, 16 |
| 1A85:03 | SubIndex 003 | $3^{rd}$ PDO Mapping entry (object 0xF101 (DP Master Status data), entry 0x03 (Slave Status Counter)) | UINT32 | RW | 0xF101:03, 16 |
| 1A85:04 | SubIndex 004 | $4^{th}$ PDO Mapping entry (object 0xF101 (DP Master Status data), entry 0x04 (Cycle Time)) | UINT32 | RW | 0xF101:04, 16 |
| 1A85:05 | SubIndex 005 | $5^{th}$ PDO Mapping entry (object 0xF101 (DP Master Status data), entry 0x05 (Repeat Counter)) | UINT32 | RW | 0xF101:05, 16 |
| 1A85:06 | SubIndex 006 | $6^{th}$ PDO Mapping entry (16 bits align) | UINT32 | RW | 0x0000:00, 16 |
| 1A85:07 | SubIndex 007 | $7^{th}$ PDO Mapping entry (16 bits align) | UINT32 | RW | 0x0000:00, 16 |
| 1A85:08 | SubIndex 008 | $8^{th}$ PDO Mapping entry (8 bits align) | UINT32 | RW | 0x0000:00, 8 |
| 1A85:09 | SubIndex 009 | $9^{th}$ PDO Mapping entry (4 bits align) | UINT32 | RW | 0x0000:00, 4 |
| 1A85:0A | SubIndex 010 | $10^{th}$ PDO Mapping entry (object 0xF101 (DP Master Status data), entry 0x14 (Device Diag)) | UINT32 | RW | 0xF101:14, 1 |
| 1A85:0B | SubIndex 011 | $11^{th}$ PDO Mapping entry (object 0xF101 (DP Master Status data), entry 0x15 (Sync Error)) | UINT32 | RW | 0xF101:15, 1 |
| 1A85:0C | SubIndex 012 | $12^{th}$ PDO Mapping entry (object 0xF101 (DP Master Status data), entry 0x16 (Cycle Toggle)) | UINT32 | RW | 0xF101:16, 1 |
| 1A85:0D | SubIndex 013 | $13^{th}$ PDO Mapping entry (object 0xF101 (DP Master Status data), entry 0x17 (Cycle State)) | UINT32 | RW | 0xF101:17, 1 |

### Index 1A86 DPM TxPDO-Map Cycle Statistics

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A86:0 | DPM TxPDO-Map Cycle Statistics | PDO Mapping TxPDO 135 contains the DP master cycle statistics (object 0xF10F [▶ 158]) for the SM synchronous mode | UINT8 | RW | 0x05 ($5_{dec}$) |
| 1A86:01 | SubIndex 001 | 1st PDO Mapping entry (object 0xF10F (Cycle Statistic data), entry 0x03 (Cycle Time)) | UINT32 | RW | 0xF10F:03, 16 |
| 1A86:02 | SubIndex 002 | 2nd PDO Mapping entry (object 0xF10F (Cycle Statistic data), entry 0x04 (Cycle Distance Time)) | UINT32 | RW | 0xF10F:04, 16 |
| 1A86:03 | SubIndex 003 | 3rd PDO Mapping entry (object 0xF10F (Cycle Statistic data), entry 0x05 (Output Calc And Copy Time)) | UINT32 | RW | 0xF10F:05, 16 |
| 1A86:04 | SubIndex 004 | 4th PDO Mapping entry (object 0xF10F (Cycle Statistic data), entry 0x06 (Input Calc And Copy Time)) | UINT32 | RW | 0xF10F:04, 16 |
| 1A86:05 | SubIndex 005 | 5th PDO Mapping entry (object 0xF10F (Cycle Statistic data), entry 0x07 (Output Delay Time)) | UINT32 | RW | 0xF10F:07, 16 |

### Index 1A87 DPM TxPDO-Map DC Cyclic Statistics

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A87:0 | DPM TxPDO-Map DC Cyclic Statistics | PDO Mapping TxPDO 136 contains the DP master cycle statistics (object 0xF10F [▶ 158]) for the DC mode | UINT8 | RW | 0x06 ($6_{dec}$) |
| 1A87:01 | SubIndex 001 | 1st PDO Mapping entry (object 0xF10F (Cycle Statistic data), entry 0x03 (Cycle Time)) | UINT32 | RW | 0xF10F:03, 16 |
| 1A87:02 | SubIndex 002 | 2nd PDO Mapping entry (object 0xF10F (Cycle Statistic data), entry 0x04 (Cycle Distance Time)) | UINT32 | RW | 0xF10F:04, 16 |
| 1A87:03 | SubIndex 003 | 3rd PDO Mapping entry (object 0xF10F (Cycle Statistic data), entry 0x05 (Output Calc And Copy Time)) | UINT32 | RW | 0xF10F:05, 16 |
| 1A87:04 | SubIndex 004 | 4th PDO Mapping entry (object 0xF10F (Cycle Statistic data), entry 0x06 (Input Calc And Copy Time)) | UINT32 | RW | 0xF10F:04, 16 |
| 1A87:05 | SubIndex 005 | 5th PDO Mapping entry (object 0xF10F (Cycle Statistic data), entry 0x07 (Output Delay Time)) | UINT32 | RW | 0xF10F:07, 32 |
| 1A87:06 | SubIndex 006 | 6th PDO Mapping entry (object 0xF10F (Cycle Statistic data), entry 0x08 (SM/SYNC Event Distance Time)) | UINT32 | RW | 0xF10F:08, 32 |

### Index 1A88 DPM TxPDO-Map Redundancy Status

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A88:0 | DPM TxPDO-Map Redundancy Status | PDO Mapping TxPDO 137 | UINT8 | RO | 0x05 ($5_{dec}$) |
| 1A88:01 | SubIndex 001 | 1st PDO Mapping entry (object 0xF118, entry 0x01) | UINT32 | RO | 0xF118:01, 1 |
| 1A88:02 | SubIndex 002 | 2nd PDO Mapping entry (15 bits align) | UINT32 | RO | 0x0000:00, 15 |
| 1A88:03 | SubIndex 003 | 3rd PDO Mapping entry (object 0xF118, entry 0x11) | UINT32 | RO | 0xF118:11, 16 |
| 1A88:04 | SubIndex 004 | 4th PDO Mapping entry (object 0xF118, entry 0x12) | UINT32 | RO | 0xF118:12, 16 |
| 1A88:05 | SubIndex 005 | 5th PDO Mapping entry (object 0xF118, entry 0x11) | UINT32 | RO | 0xF118:13, 16 |

### Index 1C00 Sync manager type

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

### Index 1C12 RxPDO assign

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1C12:0 | RxPDO assign | PDO Assign Outputs: the RxPDOs must be assigned in the order of their indexes. The RxPDOs of the configured DP slaves (0x1600 [▶ 145]-0x167E) must be assigned if object 0x1C12 is transmitted in the StartUp SDOs. | UINT8 | RW | 0x00 ($0_{dec}$) |
| 1C12:01 | | 1st allocated RxPDO (contains the index of the associated RxPDO mapping object) | | | |
| ... | | | | | |
| 1C12:7E | | 126th allocated RxPDO (contains the index of the associated RxPDO mapping object) | | | |

### Index 1C13 TxPDO assign

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1C13:0 | TxPDO assign | PDO Assign Inputs: the TxPDOs must be assigned in the order of their indexes. The TxPDOs of the configured DP slaves (0x1A00 [▶ 147]-0x1A7C) must be assigned if object 0x1C13 is transmitted in the StartUp SDOs. It can then still be decided via TxPDO Assign whether or not the TxPDOs PDO State (index 0x1A81 [▶ 147]), PDO Toggle (index 0x1A82 [▶ 148]), DP Diag (index 0x1A83 [▶ 148]), DP Status (index 0x1A84 [▶ 148]), DP Master Diag (index 0x1A85 [▶ 148]), Cycle Statistics (index 0x1A86 [▶ 149]) and DC Cycle Statistics (index 0x1A87 [▶ 149]) are transferred to the EtherCAT input data. | UINT8 | RW | |
| 1C13:01 | | 1st allocated TxPDO (contains the index of the associated TxPDO mapping object) | UINT16 | RW | |
| ... | | | | | |
| 1C13:86 | | 134th allocated TxPDO (contains the index of the associated TxPDO mapping object) | UINT16 | RW | |

### Index 1C32 SM output parameter

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1C32:0 | SM output parameter | Synchronization parameters for the outputs | UINT8 | RO | 0x20 ($32_{dec}$) |
| 1C32:01 | Sync mode | Current synchronization mode:<br>• 0: Free run<br>• 1: synchronous with SM 2 event (SM synchronous)<br>• 2: DC-Mode - Synchronous with SYNC1 event (DC-SYNC0-synchronous)<br>• 3: DC-Mode - Synchronous with SYNC1 event (DC-SYNC1-synchronous) | UINT16 | RW | 0x0001 ($1_{dec}$) |
| 1C32:02 | Cycle time | Cycle time (in ns):<br>• Free Run: Cycle time of the local timer<br>• Synchronous with SM 2 event: Master cycle time<br>• DC mode: SYNC0 cycle time (normally also the cycle time of the master) | UINT32 | RW | 0x000F4240 ($1000000_{dec}$) |
| 1C32:03 | Shift time | DC mode only (as for 0x1C32:09 [▶ 151]) | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C32:04 | Sync modes supported | Supported synchronization modes:<br>• Bit 0 = 1: free run is supported<br>• Bit 1 = 1: Synchron with SM 2 event is supported<br>• Bit 2-3 = 01: DC mode is supported<br>• Bit 4-5 = 10: Output shift with SYNC1 event (only DC mode)<br>• Bit 14 = 1: dynamic times (measurement through writing of 1C32:08 [▶ 151]) | UINT16 | RO | 0x440B ($17419_{dec}$) |
| 1C32:05 | Minimum cycle time | This entry contains the measured cycle time (in ns) of the DP cycle, including the update of the EtherCAT process data; the maximum measured value (sum of 0x1C32:06+0x1C33:09+0x1C33:06) is located here. If the CycleExceeded counter (0x1C32:0C or 0x1C33:0C) increments, then the cycle time set in 0x1C32:02 is too low for the configured DP bus structure. | UINT32 | RO | 0x0003CFB1 ($249777_{dec}$) |
| 1C32:06 | Calc and copy time | This entry contains the measured time after the first event (SM2 (SM-synchronous or DC SYNC0-synchronous) or SYNC0 (DC SYNC1-synchronous)) until the start of the DP cycle (SM-synchronous) or after which the second event (SYNC0 (DC SYNC0-synchronous) or SYNC1 (DC SYNC1-synchronous)) may occur at the earliest. When setting DC SYNC0-synchronous, this time may also to be considered for the SYNC shift time for the EtherCAT master outputs. When setting DC SYNC1-synchronous, this time may also to be considered for the SYNC1 shift time of the EL6731. If the ShiftTooShort counter (0x1C32:0D) increments, then the corresponding shift time is set too low. | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C32:07 | Minimum delay time | | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C32:08 | Command | • 0: Measurement of the local cycle time is stopped<br>• 1: Measurement of the local cycle time is started<br>The entries 0x1C32:03 [▶ 151], 0x1C32:05 [▶ 151], 0x1C32:06 [▶ 151], 0x1C32:09 [▶ 151], 0x1C33:03 [▶ 153], 0x1C33:06 [▶ 151], 0x1C33:09 [▶ 153] are updated with the maximum measured values.<br>For a subsequent measurement the measured values are reset | UINT16 | RW | 0x0000 ($0_{dec}$) |
| 1C32:09 | Delay time | This entry contains the time after the second event (SYNC0 event (DC SYNC0-synchronous) or SYNC1 event (DC SYNC1 synchronous)) and the time and output of the outputs (in ns, DC mode only) | UINT32 | RO | 0x00000000 ($0_{dec}$) |

### Index 1C32 SM output parameter

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1C32:0B | SM event missed counter | This entry contains the number of missed SM events in OPERATIONAL (DC mode only). The counter is automatically reset at the transition from SAFEOP to OP. If it increments, then the SYNC shift time for outputs in the case of the EtherCAT master is too short. | UINT16 | RO | 0x0000 ($0_{dec}$) |
| 1C32:0C | Cycle exceeded counter | This entry contains the number of cycle time infringements during OPERATIONAL (cycle was not completed in time or the next cycle began too early). The counter is automatically reset at the transition from SAFEOP to OP. If it increments, then the cycle time of the EtherCAT master is too short. | UINT16 | RO | 0x0000 ($0_{dec}$) |
| 1C32:0D | Shift too short counter | This entry contains the number of times the distance between SM2 and SYNC 0 event (DC SYNC0) or between SYNC0 and SYNC1 event (DC SYNC1) was too short. The counter is automatically reset at the transition from SAFEOP to OP. If it increments, then the SYNC shift time for outputs in the case of the EtherCAT master (DC SYNC0) or the SYNC1 shift time in the case of the EL6731 (DC SYNC1) is too short. | UINT16 | RO | 0x0000 ($0_{dec}$) |
| 1C32:0F | Minimum Cycle Distance | | | | 0x00000000 ($0_{dec}$) |
| 1C32:10 | Maximum Cycle Distance | | | | 0x00000000 ($0_{dec}$) |
| 1C32:11 | Minimum SM SYNC Distance | | | | 0x00000000 ($0_{dec}$) |
| 1C32:12 | Maximum SM SYNC Distance | | | | 0x00000000 ($0_{dec}$) |
| 1C32:13 | Application Cycle Exceeded Counter | | | | 0x0000 ($0_{dec}$) |
| 1C32:14 | Frame repeat time | | | | 0x00000000 ($0_{dec}$) |
| 1C32:20 | Sync error | The synchronization was not correct in the last cycle (outputs were output too late; DC mode only) | BOOLEAN | RO | 0x00 ($0_{dec}$) |

### Index 1C33 SM input parameter

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1C33:0 | SM input parameter | Synchronization parameters for the inputs | UINT8 | RO | 0x20 ($32_{dec}$) |
| 1C33:01 | Sync mode | Current synchronization mode:<br>• 0: Free Run<br>• 1: Synchron with SM 3 Event (no outputs available)<br>• 2: DC - Synchron with SYNC0 Event<br>• 3: DC - Synchron with SYNC1 Event<br>• 34: Synchron with SM 2 Event (outputs available) | UINT16 | RW | 0x0022 ($34_{dec}$) |
| 1C33:02 | Cycle time | as 0x1C32:02 [▶ 151] | UINT32 | RW | 0x000F4240 ($1000000_{dec}$) |
| 1C33:03 | Shift time | Time between SYNC0 event and reading of the inputs (in ns, only DC mode) | UINT32 | RW | 0x00000000 ($0_{dec}$) |
| 1C33:04 | Sync modes supported | Supported synchronization modes:<br>• Bit 0: free run is supported<br>• Bit 1: Synchronous with SM 2 Event is supported (outputs available)<br>• Bit 1: Synchronous with SM 3 Event is supported (no outputs available)<br>• Bit 2-3 = 01: DC mode is supported<br>• Bit 4-5 = 01: input shift through local event (outputs available)<br>• Bit 4-5 = 10: input shift with SYNC1 event (no outputs available)<br>• Bit 14 = 1: dynamic times (measurement through writing of 0x1C32:08 [▶ 151] or 0x1C33:08 [▶ 153]) | UINT16 | RO | 0x440B ($17419_{dec}$) |
| 1C33:05 | Minimum cycle time | as 0x1C32:05 [▶ 151] | UINT32 | RO | 0x0003CFB1 ($249777_{dec}$) |
| 1C33:06 | Calc and copy time | Time between reading of the inputs and availability of the inputs for the master (in ns, only DC mode) | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C33:07 | Minimum delay time | | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C33:08 | Command | as 0x1C32:08 [▶ 151] | UINT16 | RW | 0x0000 ($0_{dec}$) |
| 1C33:09 | Maximum delay time | Time between SYNC1 event and reading of the inputs (in ns, only DC mode) | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C33:0B | SM event missed counter | as 0x1C32:11 [▶ 151] | UINT16 | RO | 0x0000 ($0_{dec}$) |
| 1C33:0C | Cycle exceeded counter | as 0x1C32:12 [▶ 151] | UINT16 | RO | 0x0000 ($0_{dec}$) |
| 1C33:0D | Shift too short counter | as 0x1C32:13 [▶ 151] | UINT16 | RO | 0x0000 ($0_{dec}$) |
| 1C32:0F | Minimum Cycle Distance | | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C32:10 | Maximum Cycle Distance | | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C32:11 | Minimum SM SYNC Distance | | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C32:12 | Maximum SM SYNC Distance | | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C32:13 | Application Cycle Exceeded Counter | | UINT16 | RO | 0x0000 ($0_{dec}$) |
| 1C32:14 | Frame repeat time | | UINT32 | RW | 0x00000000 ($0_{dec}$) |
| 1C33:20 | Sync error | as 0x1C32:32 [▶ 151] | BOOLEAN | RO | 0x00 ($0_{dec}$) |

### 6.1.3.2 Profile-specific objects (0x6000-0xFFFF)

The profile-specific objects have the same meaning for all EtherCAT slaves that support the profile 5001.

#### Index 6000-67C0 DP Inputs Slave yyy

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 6000+n*16: 0 | DP Inputs Slave yyy | This object exists for each configured DP slave and contains the DP input data of the DP slave. There is an entry in the input data object 0x6000+(n*16) for each DP module with DP input data configured in object 0x8002 [▶ 156]+(n*16). DP module m thereby corresponds to SubIndex m; if a DP module does not contain any DP input data, then this SubIndex does not exist. If a DP slave does not contain any DP input data, then object 0x6000+(n*16) does not exist. SubIndex 0 contains the number of DP modules of the (n+1)th configured DP slave. | UINT8 | RO | |
| (6000+n*16) :01 | | if the first DP module has DP input data: DP input data of the first DP module | OCTET-STRING | RO | |
| ... | | | | | |
| (6000+n*16) :m | | if the mth DP module has DP input data: DP input data of the mth DP module | OCTET-STRING | RO | |

#### Index 7000-77C0 DP Outputs Slave yyy

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 7000+n*16: 0 | DP Outputs Slave yyy | This object exists for each configured DP slave and contains the DP output data of the DP slave. There is an entry in the input data object 0x7000+(n*16) for each DP module with DP output data configured in object 0x8002 [▶ 156]+(n*16). DP module m thereby corresponds to SubIndex m; if a DP module does not contain any DP output data, then this SubIndex does not exist. If a DP slave does not contain any DP output data, then object 0x7000+(n*16) does not exist. SubIndex 0 contains the number of DP modules of the (n+1)th configured DP slave. | UINT8 | RO | |
| (7000+n*16) :01 | | if the first DP module has DP output data: DP output data of the first DP module | OCTET-STRING | RO | |
| ... | | | | | |
| (7000+n*16) :F4 | | if the mth DP module has DP output data: DP output data of the mth DP module | OCTET-STRING | RO | |

### Index 8000-87C0 DP Slave Parameter Set Slave yyy

| Index (hex) | Name | Meaning | | Data type | Flags | Default |
|---|---|---|---|---|---|---|
| 8000+n*16:0 | DP Slave Parameter Set Slave yyy | This object contains the DP configuration of the (n+1)th configured DP slave (0 <= n <125). The object is to be transmitted with Complete Access, or SubIndex 0 must first be set to 0, then the individual SubIndexes transmitted (non-existent SubIndexes or gaps are thereby to be omitted) and finally SubIndex 0 set to the correct value. | | UINT8 | RW | 0x33 ($51_{dec}$) |
| (8000+n*16):01 | Station Address | DP station address of the DP slave, permitted values: 0-125, the entry 0xF020:(n+1) is automatically updated with this | | UINT16 | RW | 0x0000 ($0_{dec}$) |
| (8000+n*16):04 | Device type | DP Ident Number of the DP slave | | UINT32 | RW | 0x00000000 ($0_{dec}$) |
| (8000+n*16):1D | Network Flags | reserved for AMS via DP | | UINT16 | RW | 0x0000 ($0_{dec}$) |
| (8000+n*16):1E | Network Port | reserved for AMS via DP | | UINT16 | RW | 0x0000 ($0_{dec}$) |
| (8000+n*16):1F | Network Segment Address | reserved for AMS via DP | | OCTET-STRING[6] | RW | {0} |
| (8000+n*16):20 | SI_Flag | SI_flag | | UINT8 | RW | 0x80 ($128_{dec}$) |
| | | Bits 0-2 | Reserve, must be 0 | | | |
| | | Bit 3 | Activate DPV1 communication for DP slave | | | |
| | | Bit 4 | Data exchange telegram is sent as a DXB Broadcast | | | |
| | | Bit 5 | FailSafe function (send data exchange without data in the CLEAR (SAFE-OP) state) is active | | | |
| | | Bit 6 | Reserve, must be 0 | | | |
| | | Bit 7 | Reserve, must be 1 | | | |
| (8000+n*16):21 | Slave_Type | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| (8000+n*16):22 | Max_Diag_Data_Len | maximum length of the DP DiagData of the DP slave (6 <= Max_Diag_Data_Len <= 244) | | UINT8 | RW | |
| (8000+n*16):23 | Max_Alarm_Len | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| (8000+n*16):24 | Max_Channel_Data_Length | maximum length of the DPV1 telegram of the DP slave (4 <= Max_Channel_Data_Length <= 244) | | UINT8 | RW | |
| (8000+n*16):25 | Diag_Upd_Delay | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| (8000+n*16):26 | Alarm_Mode | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| (8000+n*16):27 | Add_SI_Flag | Reserve, must be 1 | | UINT8 | RW | 0x01 ($1_{dec}$) |
| (8000+n*16):28 | C1_Timeout | Timeout during waiting for a DPV1 response (in 10 ms) | | UINT16 | RW | |
| (8000+n*16):29 | Number of tolerated Data Exchange with no answer | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| (8000+n*16):2A | Parallel AoE Services | Number of parallel AoE services (e.g. for DPV1); the value 0 corresponds to 5 parallel services | | UINT8 | RW | 0x00 ($0_{dec}$) |
| (8000+n*16):2B | Reaction on no answer | 0 | Data exchange is quit | BIT1 | RW | 0x00 ($0_{dec}$) |
| | | 1 | DP slave remains in Data Exchange if a valid Data Exchange response is received within the DP watchdog time | | | |
| (8000+n*16):2C | Restart behavior after DP fault | 0 | DP slave is automatically restarted (Set_Prm-Unlock, then Slave_Diag, Set_Prm-Lock, Chk_Cfg, Slave_Diag) | BIT1 | RW | 0x00 ($0_{dec}$) |
| | | 1 | The DP slave must be manually restarted via AoE | | | |

### Index 8000-87C0 DP Slave Parameter Set Slave yyy

| Index (hex) | Name | Meaning | | Data type | Flags | Default |
|---|---|---|---|---|---|---|
| (8000+n*16):2D | Master reaction after DP fault | 0 | no response | BIT1 | RW | 0x00 ($0_{dec}$) |
| | | 1 | Data exchange with all DP slaves is stopped (Set_Prm-Unlock to all slaves); DP communication must be restarted via AoE | | | |
| (8000+n*16):2E | Changes of DP Inputs after DP fault | 0 | Data of TxPDO n are set to 0 | BIT1 | RW | 0x00 ($0_{dec}$) |
| | | 1 | Data of TxPDO n remain unchanged | | | |
| (8000+n*16):2F | PKW supported | Reserve, must be 0 | | BOOLEAN | RW | 0x00 ($0_{dec}$) |
| (8000+n*16):30 | FDL only | Reserve, must be 0 | | BOOLEAN | RW | 0x00 ($0_{dec}$) |
| (8000+n*16):31 | Watchdog Base 1 ms | FALSE | The basis for the DP watchdog is 10 ms | BOOLEAN | RW | 0x00 ($0_{dec}$) |
| | | TRUE | The basis for the DP watchdog is 1 ms | | | |
| (8000+n*16):33 | Cycle Access Divider | Reserve, must be 1 | | UINT8 | RW | 0x01 ($1_{dec}$) |
| (8000+n*16):34 | Cycle Access Modulo | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| (8000+n*16):35 | Vendor specific reserved | Reserve, must be 0 | | OCTET-STRING[25] | RW | {0} |

### Index 8001-87C1 DP PrmData Slave yyy

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 8001+n*16:0 | DP PrmData Slave yyy | This object contains the DP Prm Data of the (n+1)$^{th}$ configured DP slave. | OCTET-STRING[244] | RW | {0} |

### Index 8002-87C2 DP CfgData Slave yyy

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 8002+n*16:0 | DP CfgData Slave yyy | This object contains the DP Cfg Data of the (n+1)$^{th}$ configured DP slave. | OCTET-STRING[244] | RW | {0} |

### Index A000-A7C0DP Status data Slave yyy

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| A000+n*16:0 | DP Status data Slave.000 | This object contains the status data of the (n+1)$^{th}$ configured DP slave. | UINT8 | RO | 0x04 ($4_{dec}$) |
| (A000+n*16):01 | DP state | State of the DP connection (identical to entry 0xF102 [▶ 158]:(n+1)) | UINT8 | RO | 0x00 ($0_{dec}$) |
| (A000+n*16):02 | Ext Diag | This entry indicates whether the diagnostic information in object 0xA002+(n*16) has changed | BOOLEAN | RO | 0x00 ($0_{dec}$) |
| (A000+n*16):03 | Repeat Counter | This entry increments on each repeat telegram to the DP slave | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| (A000+n*16):04 | No Answer Counter | This entry increments on each telegram to the DP slave to which no response is received | UINT32 | RO | 0x00000000 ($0_{dec}$) |

### Index A002-A07E DP Diag data Slave.000

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| A002+n*16:0 | DP Diag data Slave.000 | This object contains the DP DiagData of the (n+1)$^{th}$ configured DP slave. | OCTET-STRING[244] | RO | {0} |

### Index F000 Modular device profile

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| F000:0 | Modular device profile | General information for the modular device profile | UINT8 | RO | 0x02 ($2_{dec}$) |
| F000:01 | Module index distance | Index distance of the objects of the individual channels | UINT16 | RO | 0x0010 ($16_{dec}$) |
| F000:02 | Maximum number of modules | Number of channels | UINT16 | RO | 0x007D ($125_{dec}$) |
| F000:03 | General Configuration Entries | General configuration entries | UINT32 | RO | 0x70000009 ($1879048201_{dec}$) |

### Index F008 Code word

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

### Index F010 Module list

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| F010:0 | Module list | Max. subindex | UINT8 | RW | 0x00 ($0_{dec}$) |
| F010:01 | | reserved | | | |
| ... | | reserved | | | |
| F010:7D | | reserved | | | |

### Index F101 DP Master Status data

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| F101:0 | DP Master Status data | This object contains the DP master status data, which are mapped into TxPDO 134 (Index 0x1A85 [▶ 148]) | UINT8 | RO | 0x17 ($23_{dec}$) |
| F101:01 | Bus Error Counter | increments on each bus error | UINT16 | RO | 0x0000 ($0_{dec}$) |
| F101:02 | Cycle Counter | increments on each DP cycle | UINT16 | RO | 0x0000 ($0_{dec}$) |
| F101:03 | Slave Status Counter | Number of DP slaves that were not exchanging data in the preceding cycle | UINT16 | RO | 0x0000 ($0_{dec}$) |
| F101:04 | Cycle Time | Cycle time of the preceding DP cycle in 1/9 µs | UINT16 | RO | 0x0000 ($0_{dec}$) |
| F101:05 | Repeat Counter | Number of repeats in the preceding DP cycle | UINT16 | RO | 0x0000 ($0_{dec}$) |
| F101:14 | Device Diag | indicates whether diagnostic information is present in the master that can be collected via AoE | BOOLEAN | RO | 0x00 ($0_{dec}$) |
| F101:15 | Sync Error | this is set if a synchronization error occurred in the preceding cycle in Distributed Clocks mode | BOOLEAN | RO | 0x00 ($0_{dec}$) |
| F101:16 | Cycle Toggle | toggles on each DP cycle | BOOLEAN | RO | 0x00 ($0_{dec}$) |
| F101:17 | Cycle State | this is set if at least one DP slave is not exchanging data | BOOLEAN | RO | 0x00 ($0_{dec}$) |

### Index F102 DP Slave Status data

| Index | Name | Meaning | | Data type | Flags | Default |
|---|---|---|---|---|---|---|
| F102:0 | DP Slave Status data | This object contains the status of communication to the DP slaves that is mapped to TxPDO 133 (index 0x1A84 [▶ 148]) | | UINT8 | RO | |
| F102:01 | | Communication status of the first configured DP slave (in the objects 0x8000 [▶ 155], 0x8001 [▶ 156], 0x8002 [▶ 156]) | | UINT8 | RO | |
| | | 0 | Slave is exchanging data; DP inputs are up to date | | | |
| | | 1 | Slave is deactivated (via AoE) | | | |
| | | 2 | Slave does not respond | | | |
| | | 3 | slave currently exchanging data with another master | | | |
| | | 4 | incorrect slave response (e.g. service not activated in the case of data exchange) | | | |
| | | 5 | Slave reports parameterization error (error in the SetPrm telegram (usually Ident-Number or UserPrmData)) | | | |
| | | 6 | Slave reports that a DP function is not supported | | | |
| | | 7 | Slave reports configuration error 8 - error in the ChkCfg telegram (usually incorrect DP module configured)) | | | |
| | | 8 | Slave is in DP bootup but is not yet ready for data exchange | | | |
| | | 9 | Slave reports static diagnosis | | | |
| | | 11 | Response with bus error (e.g. parity or checksum error) | | | |
| | | 14 | Response with telegram error (e.g. request bit set) | | | |
| | | 15 | Slave reports no resources (e.g. PrmData or CfgData is too long) | | | |
| | | 16 | Slave reports that the DP service is not activated | | | |
| | | 17 | Unexpected telegram received (e.g. token when waiting for a response from a slave) | | | |
| | | 18 | Slave is ready for data exchange; however, no process data are exchanged over EtherCAT as yet | | | |
| ... | | | | | | |
| F102:7D | | Communication status of the mth configured DP slave | | UINT8 | RO | |

### Index F10F Cycle Statistic data

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| F10F:0 | Cycle Statistic data | This object contains the measured times of the respective last cycle. They can be transferred cyclically with TxPDO 135 (0x1A86 [▶ 149]) or TxPDO 136 (0x1A87 [▶ 149]) and they then always contain the values for the preceding cycle. | UINT8 | RO | 0x08 ($8_{dec}$) |
| F10F:01 | Multiplier | The entries 0xF10F:03, 0xF10F:04, 0xF10F:05 and 0xF10F:06 must be multiplied by this value in order to convert these entries into the unit 100 ns | UINT16 | RO | 0x0000 ($0_{dec}$) |
| F10F:02 | Divider | The entries 0xF10F:03, 0xF10F:04, 0xF10F:05 and 0xF10F:06 must be divided by this value in order to convert these entries into the unit 100 ns | UINT16 | RO | 0x0000 ($0_{dec}$) |
| F10F:03 | Cycle Time | Cycle time (0x1C32:05 [▶ 151]) of the preceding DP cycle in local ticks | UINT16 | RO | 0x0000 ($0_{dec}$) |
| F10F:04 | Cycle Distance Time | Distance between the last two SM2 events (only if 0x1C32:01 [▶ 151] = 1 or 0x1C32:01 = 2) | UINT16 | RO | 0x0000 ($0_{dec}$) |
| F10F:05 | Output Calc And Copy Time | Output Calc And Copy Time (0x1C32:06 [▶ 151]) of the preceding DP cycle in local ticks | UINT16 | RO | 0x0000 ($0_{dec}$) |
| F10F:06 | Input Calc And Copy Time | Input Calc And Copy Time (0x1C33:06) of the preceding DP cycle in local ticks | UINT16 | RO | 0x0000 ($0_{dec}$) |
| F10F:07 | Output Delay Time | Output Delay Time (0x1C32:09 [▶ 151]) of the preceding cycle in ns | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| F10F:08 | SM/SYNC Event Distance Time | SM/SYNC0 Event Distance Time (distance between SM2 and SYNC0 event) of the preceding cycle in ns (must be activated via register 0x816 in the ET1100). | INT32 | RO | 0x00000000 ($0_{dec}$) |

### Index F800 DP Bus Parameter Set

| Index (hex) | Name | Meaning | | Data type | Flags | Default |
|---|---|---|---|---|---|---|
| F800:0 | DP Bus Parameter Set | This object contains the DP bus parameters. The object is to be transmitted with Complete Access, or SubIndex 0 must first be set to 0, then the individual SubIndexes transmitted (non-existent SubIndexes or gaps are thereby to be omitted) and finally SubIndex 0 set to the correct value. | | UINT8 | RW | 0x18 ($24_{dec}$) |
| F800:01 | DL_Add | DP-Master station address | | UINT8 | RW | 0x00 ($0_{dec}$) |
| F800:02 | Data_rate | Baud rate | | UINT8 | RW | 0x00 ($0_{dec}$) |
| | | 0 | 9.6 kbaud | | | |
| | | 1 | 19.2 kbaud | | | |
| | | 2 | 93.75 kbaud | | | |
| | | 3 | 187.5 kbaud | | | |
| | | 4 | 500 kbaud | | | |
| | | 6 | 1.5 Mbaud | | | |
| | | 7 | 3 Mbaud | | | |
| | | 8 | 6 Mbaud | | | |
| | | 9 | 12 Mbaud | | | |
| F800:03 | Tsl | Slot time in bit times | | UINT16 | RW | * |
| F800:04 | min Tsdr | Min. TSDR in bit times | | UINT16 | RW | 0x0B ($11_{dec}$) |
| F800:05 | max Tsdr | Max. TSDR in bit times | | UINT16 | RW | * |
| F800:06 | Tqui | Quiet Time in bit times | | UINT8 | RW | * |
| F800:07 | Tset | Setup Time in bit times | | UINT8 | RW | * |
| F800:08 | Ttr | Target Token Rotation Time in bit times | | UINT32 | RW | * |
| F800:09 | G | GAP Update Factor (1-100) | | UINT8 | RW | 0x64 ($100_{dec}$) |
| F800:0A | HSA | Highest master address (0-126) | | UINT8 | RW | 0x7E ($126_{dec}$) |
| F800:0B | max_retry_limit | Maximum number of repeats (1-8) | | UINT8 | RW | * |
| F800:0C | Bp_Flag | Bit 0 | Single master operation (no GAP update is sent) | UINT8 | RW | 0x00 ($0_{dec}$) |
| | | Bits 1-6 | Reserve, must be 0 | | | |
| | | Bit 8 | Auto-Clear mode is on (DP master goes to CLEAR if a DP slave is not exchanging data) | | | |

### Index F800 DP bus parameter set

| Index (hex) | Name | Meaning | | Data type | Flags | Default |
|---|---|---|---|---|---|---|
| F800:0D | Min_Slave_Interval | Cycle time for DP telegrams (except data exchange (runs synchronously to the EtherCAT master)) | | UINT16 | RW | * |
| F800:0E | Poll_Timeout | Reserve, must be 0 | | UINT16 | RW | 0x0000 ($0_{dec}$) |
| F800:0F | Data_Control_Time | Cycle time of the Global_Control telegram | | UINT16 | RW | * |
| F800:10 | Alarm_Max | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| F800:11 | Max_User_Global_control | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| F800:12 | Max Retry Limit (Data-Exchange) | Maximum number of data exchange repeats | | BIT4 | RW | * |
| F800:13 | Send Set-Prm-Unlock | Reserve, must be 0 | | BIT2 | RW | 0x00 ($0_{dec}$) |
| F800:14 | Auto Clear Mode | Reserve, must be 0 | | BIT2 | RW | 0x00 ($0_{dec}$) |
| F800:15 | Operate Delay (in 100 ms) | Automatic switching to OPERATE in the OP state is delayed according to this setting | | UINT8 | RW | 0x00 ($0_{dec}$) |
| F800:16 | Safety Time (in us) | Reserve, must be 0 | | UINT16 | RW | 0x0000 ($0_{dec}$) |
| F800:17 | Flags | Reserve, must be 0 | | UINT16 | RW | 0x0000 ($0_{dec}$) |
| F800:18 | Watchdog Reaction | 0 | DP Master in CLEAR Mode | OCTET-STRING[26] | RW | {0} |
| | | 1 | DP Master in OPERATE Mode | | | |
| | | 2 | DP Master in STOP Mode | | | |
| F800:1A | Vendor specific | Reserve, must be 0 | | OCTET-STRING[26] | RW | |

### Index F920 AoE Settings

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| F920:0 | AoE Settings | Max. subindex | UINT8 | RO | 0x01 ($1_{dec}$) |
| F920:01 | Local AoE Net ID | Local AoE Net ID | OCTET-STRING[6] | RO | {0} |

* These parameters depend on the baud rate; the following table shows the default values for the respective baud rate:

| Data_rate | Tsl | max. Tsdr | Tqui | Tset | Ttr | max_retry_limit | Min_Slave_Interval | Data_Con-trol_Time | Max Retry (DX) |
|---|---|---|---|---|---|---|---|---|---|
| 9 (12 Mbaud) | 1000 | 800 | 9 | 16 | 12000 | 4 | 10 | 1 | 4 |
| 8 (6 Mbaud) | 600 | 450 | 6 | 8 | 12000 | 3 | 20 | 2 | 3 |
| 7 (3 Mbaud) | 400 | 250 | 3 | 4 | 12000 | 2 | 40 | 4 | 2 |
| 6 (1.5 Mbaud) | 300 | 150 | 0 | 1 | 12000 | 1 | 80 | 8 | 1 |
| 4 (500 kbaud) | 200 | 100 | 0 | 1 | 12000 | 1 | 200 | 20 | 1 |
| 3 (187.5 kbaud) | 150 | 100 | 0 | 1 | 12000 | 1 | 400 | 40 | 1 |
| 2 (93.75 kbaud) | 150 | 100 | 0 | 1 | 12000 | 1 | 800 | 80 | 1 |
| 1 (19.2 kbaud) | 150 | 100 | 0 | 1 | 12000 | 1 | 4000 | 400 | 1 |
| 0 (9.6 kbaud) | 150 | 100 | 0 | 1 | 12000 | 1 | 8000 | 800 | 1 |

# 6.2 PROFIBUS slave

## 6.2.1 State machine

The EL6731-0010 can be configured in several ways:

1. Configuration of the EL6731-0010 with StartUp SDOs [▶ 161]: Here, the StartUp SDOs are calculated in the EtherCAT configurator and transferred to the EtherCAT master, in the same way as is carried out, for example, in the TwinCAT System Manager.
2. Configuration of the EL6731-0010 with Backup Parameter Storage [▶ 163]: Here, the configuration of the CANopen slave is stored in the flash memory of the EL6731-0010 and need only be transmitted once.

**Configuration of the EL6731-0010 with StartUp SDOs:**

The following flow chart shows the sequence of the configuration of the EL6731-0010 with StartUp SDOs:

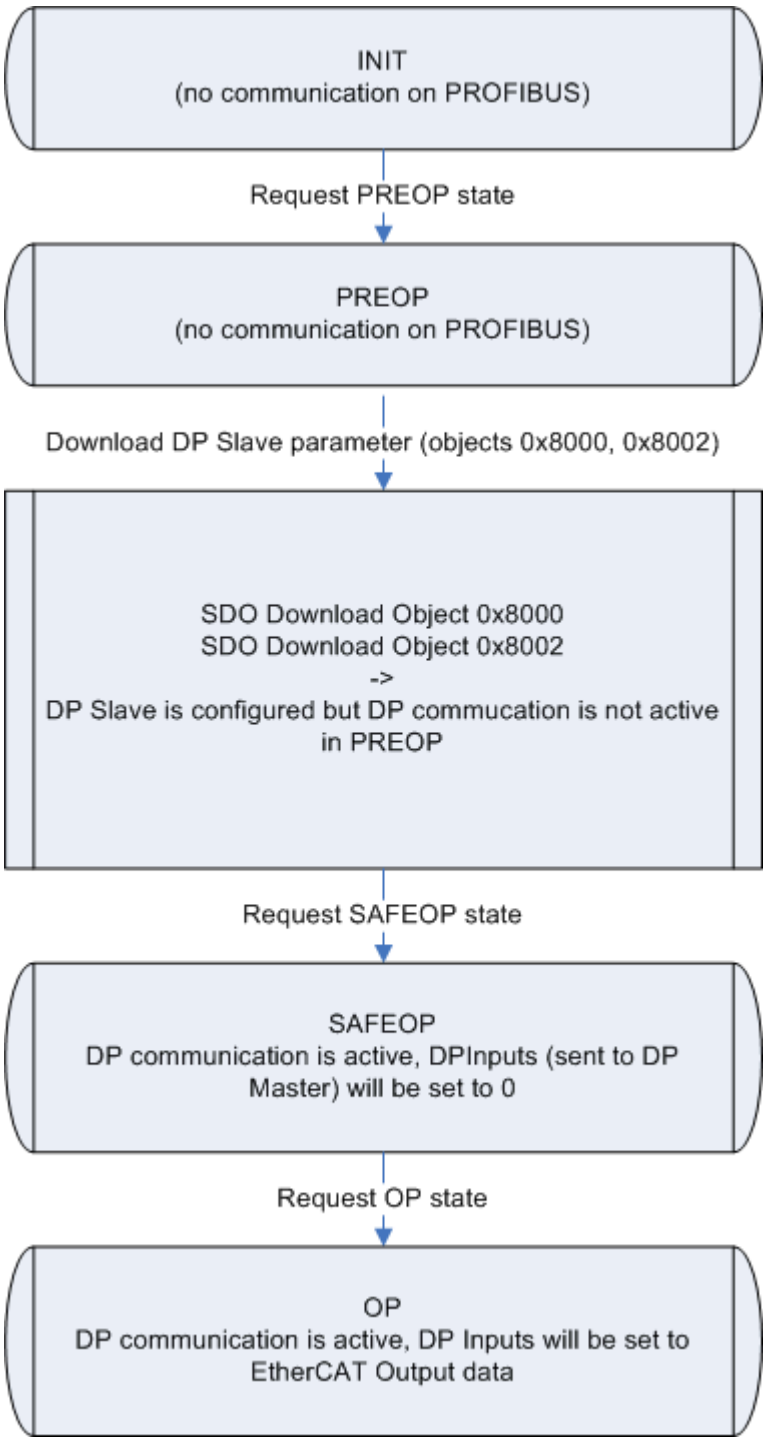



Fig. 136: Flow chart: Sequence of the configuration of the EL6731-0010 with start SDOs

Following a power-on, the EL6731-0010 is in the INIT state and has no DP configuration. The EL6731-0010 is not active on the PROFIBUS.

### DP slave configuration

The DP configuration is carried out via SDO download in the PREOP state. The objects to be loaded must be transmitted either with Complete Access or with consistency nesting (first set SubIndex 0 to 0, then write SubIndex 1-n, then set SubIndex 0 to n). Care should thereby be taken to always start with object 0xF800 [▶ 171]. The DP CfgData are to be transferred with object 0x8002 [▶ 171]. After the receipt of object 0x8000, the EL6731-0010 is active on the PROFIBUS (FDL only, no DP). The EL6731-0010 supports automatic baud rate search, therefore no baud rate need be configured.

### PDO Mapping

There is an EtherCAT RxPDO (if the DP slave has DP inputs (is sent to the DP master)) and an EtherCAT TxPDO (if the DP slave has DP outputs (is received from the DP master)). The PDO mapping of the EtherCAT PDOs is automatically calculated by the EL6731-0010 after the download of the respective 0x8yyz objects and can be read. The PDO mapping objects can only be written with the values that the EL6731-0010 has calculated itself. The writing of the PDO mapping thus serves only to check the PDO mapping calculated by the EtherCAT configurator and can therefore be omitted.

### PDO Assign

Additionally there is an EtherCAT PDO that contains status information. This PDO is selected via the PDO Assign. With regard to the order of the PDOs in the PDO Assign, it is important to ensure that the index of the assigned EtherCAT PDO increases with each entry in the corresponding PDO Assign object. If the EtherCAT master does not transfer a PDO Assign to the StartUp SDOs, then the PDO 0x1A7F [▶ 168] is assigned for the status.

### Cyclic DP communication

During the transition to SAFEOP, the EL6731 checks the length configured in the Sync Manager channels 2 and 3 against the length calculated from PDO Mapping and PDO Assign. The SAFEOP state is only adopted if these lengths match. The DP function of the EL6731-0010 is active in the SAFEOP state. As long as the EL6731-0010 is in SAFEOP, the DP input data that are sent to the DP-master are set to 0. As soon as the EL6731-0010 has been switched to OP, the data from the EtherCAT outputs are transmitted to the DP master.

The EL6731-0010 runs independently of the EtherCAT cycle.

### Configuration of the EL6731-0010 with Backup Parameter Storage

The following flow chart shows the sequence of the configuration of the EL6731-0010 with Backup Parameter Storage:

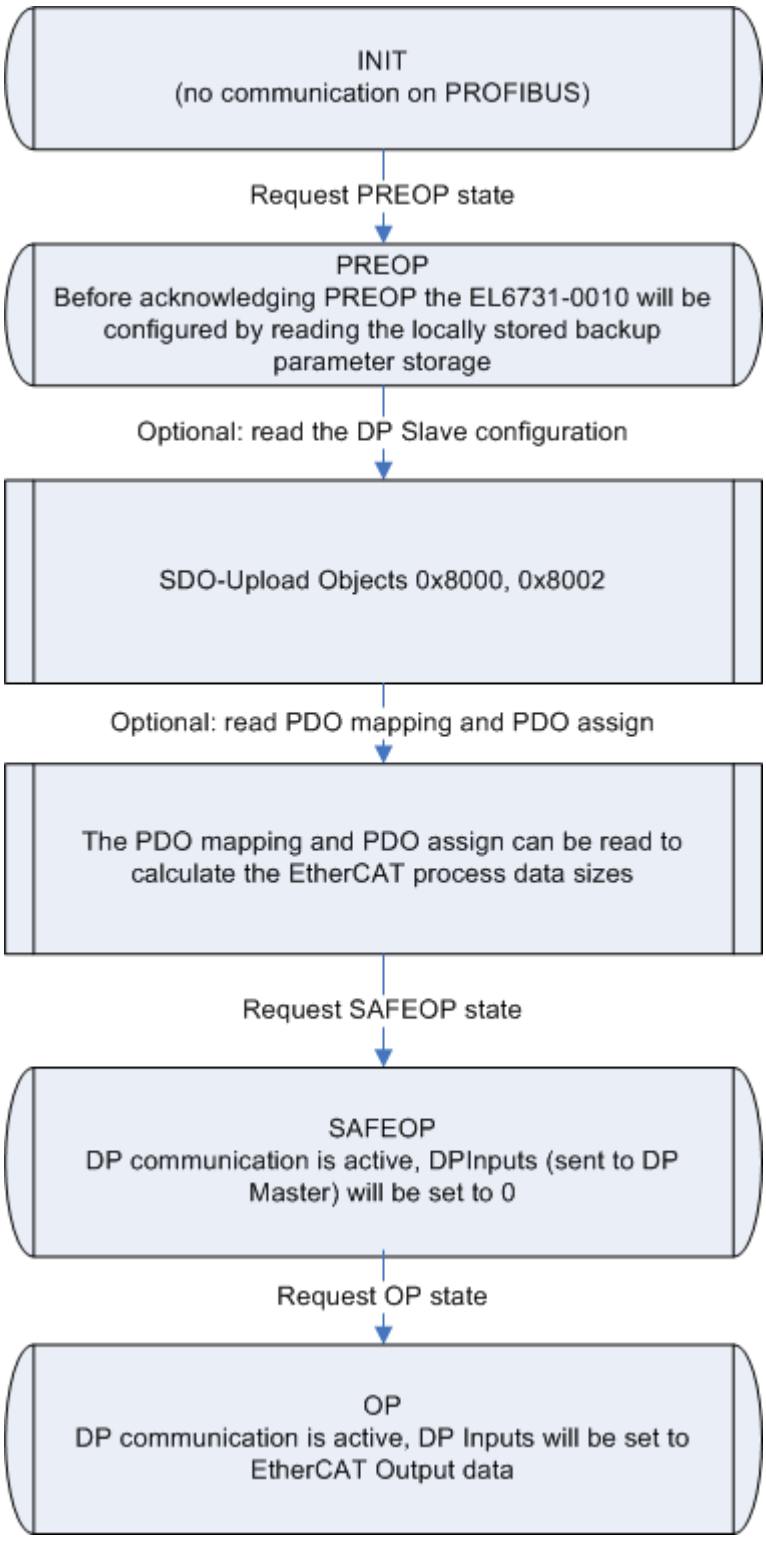



Fig. 137: Flow chart: Sequence of the configuration of the EL6731-0010 with backup parameter storage

Following a power-on, the EL6731-0010 is in the INIT state and has no DP configuration. The EL6731-0010 is not active on the PROFIBUS.

**DP bus parameters / DP slave configuration**

The configuration stored in the Backup Parameter Storage object 0x10F2 is loaded during the transition from INIT to PREOP. Since the StartUp SDOs from the Configuration of the EL6731-0010 with StartUp SDOs [▶ 161] are stored in the Backup Parameter Storage object, the sequence is the same as the one described there. Hence, the stored data is initially written to object 0x8000 and the EL6731-0010 is active on the

PROFIBUS with automatic baud rate search. Subsequently, DP slave configuration is generated in object 0x8002. If the PREOP state is quit, the current DP configuration can be read from objects 0x8000 and 0x8002.

#### PDO Mapping / PDO Assign

In addition, the EtherCAT master can also read the PDO Mapping and PDO Assign in the PREOP state in order to determine the lengths of the EtherCAT process data.

#### Creating the Backup Parameter Storage

The Backup Parameter Storage can be created as follows:

1. Download object 0x10F2 (in PREOP without having sent StartUp SDOs beforehand): in this case the data received will be saved as Backup Parameter Storage in the flash memory

#### Deleting the Backup Parameter Storage

In order to load a new Backup Parameter Storage or to simply delete the existing one, the value 0x64616F6C has to be written to entry 0x1011:01.

## 6.2.2 Object description and parameterization

**EtherCAT XML Device Description**

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

**Parameterization via the CoE list (CAN over EtherCAT)**

The EtherCAT device is parameterized via the CoE-Online tab (double-click on the respective object) or via the Process Data tab (allocation of PDOs). Please note the following general CoE notes [▶ 33] 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

### 6.2.2.1 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 | 0x0C1D1389 ($203232137_{dec}$) |

#### Index 1008 Device name

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1008:0 | Device name | Device name of the EtherCAT slave | STRING | RO | EL6731-0010 |

#### Index 1009 Hardware version

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

### Index 100A Software version

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

### Index 1011 Restore default parameters

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

### Index 1018 Identity

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1018:0 | Identity | Information for identifying the slave | UINT8 | RO | 0x04 ($4_{dec}$) |
| 1018:01 | Vendor ID | Vendor ID of the EtherCAT slave | UINT32 | RO | 0x00000002 ($2_{dec}$) |
| 1018:02 | Product code | Product code of the EtherCAT slave | UINT32 | RO | 0x1A4B3052 ($441135186_{dec}$) |
| 1018:03 | Revision | Revision number of the EtherCAT slave; the low word (bit 0-15) indicates the special terminal number, the high word (bit 16-31) refers to the device description | UINT32 | RO | |
| 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 | |

### Index 10F0 Backup parameter handling

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 10F0:0 | Backup parameter handling | Information for standardized loading and saving of backup entries | UINT8 | RO | |
| 10F0:01 | Checksum | Checksum of the Backup Parameter Storage (object 0x10F2 [▶ 167], word 2-3) | UINT32 | RO | |

### Index 10F2 Backup parameter storage

| Index (hex) | Name | Meaning | | Data type | Flags | Default |
|---|---|---|---|---|---|---|
| 10F2:0 | Backup parameter storage | If this object is used, no StartUp SDOs may be transmitted in the PREOP state, since the Backup Parameter Storage contains the complete StartUp SDOs (see Configuration of the EL6731-0010 with Backup Parameter Storage). The EL6731-0010 is rebooted 5s after the flashing of the Backup Parameter Storage (switches to INIT with AL status code = 0x60). The data have the following meaning: | | OCTET-STRING[n] | RW | |
| | | Word-Offset | Description | | | |
| | | 0 | Command: with 0xC0DE, the received data are stored in the flash memory | | | |
| | | 1 | Length of the data from word offset 4 in bytes | | | |
| | | 2-3 | Checksum, which is calculated locally | | | |
| | | 4 | Index of the object of the 1st StartUp SDO | | | |
| | | 5 | len1: Length of the object of the 1st StartUp SDO as CompleteAccess (from SubIndex 0) in bytes | | | |
| | | 6-n1 | Data of the object of the 1st StartUp SDO as CompleteAccess (n1 = 2*((len1+1)/2)+5) | | | |
| | | n1+1 | Index of the object of the 2nd StartUp SDO | | | |
| | | n1+2 | len2: Length of the object of the 2nd StartUp SDO as CompleteAccess (from SubIndex 0) in bytes | | | |
| | | (n1+3)-n2 | Data of the object of the 2nd StartUp SDO as CompleteAccess (n2 = 2*((len2+1)/2)+n1+2) | | | |
| | | ... | | | | |
| | | m | Index of the object of the 3rd StartUp SDO | | | |
| | | m+1 | len3: Length of the object of the 3rd StartUp SDO as CompleteAccess (from SubIndex 0) in bytes | | | |
| | | (m+2)-n3 | Data of the object of the 3rd StartUp SDO as CompleteAccess (n3 = 2*((len3+1)/2)+m+1) | | | |

### Index 1600 DPS RxPDO-Map Slave

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1600:0 | DPS RxPDO-Map Slave | This RxPDO contains the DP input data of the DP slave, which are sent to the DP master. There is an entry in the RxPDO mapping object 0x1600 for each DP module with DP input data configured in object 0x8002 [▶ 171]. If the DP slave does not contain any DP input data, then the PDO mapping object 0x1600 does not exist. This PDO is mandatory and must always be contained in the PDO Assign object 0x1C12 [▶ 168] . SubIndex 0 contains the number of DP modules with DP input data of the DP slave. | UINT8 | RO | |
| 1600:01 | | first DP module with DP input data of the DP slave | UINT32 | RO | |
| ... | | .. | | | |
| 1600:m | | last DP module with DP input data of the DP slave | UINT32 | RO | |

### Index 187F TxPDO-Par Status PDO

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 187F:0 | TxPDO-Par Status PDO | PDO parameter TxPDO 129 | UINT8 | RO | 0x06 ($6_{dec}$) |
| 187F:06 | Exclude TxPDOs | Specifies the TxPDOs (index of TxPDO mapping objects) that must not be transferred together with TxPDO 128 | OCTET-STRING[10] | RO | 80 1A 00 00 00 00 00 00 00 00 |
| 187F:07 | TxPDO State | The TxPDO state is set if the DP slave is not exchanging data | BOOLEAN | RO | |
| 187F:09 | TxPDO Toggle | The TxPDO toggle becomes | BOOLEAN | RO | |

### Index 1A00 DPS TxPDO-Map Slave

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A00:0 | DPS TxPDO-Map Slave | This TxPDO contains the DP output data of the DP slave that were received by the DP master. There is an entry in the TxPDO mapping object 0x1A00 for each DP module with DP output data configured in object 0x8002 [▶ 171]. If the DP slave does not contain any DP output data, then the PDO mapping object 0x1A00 does not exist. This PDO is mandatory and must always be contained in the PDO Assign object 0x1C13 [▶ 168] . SubIndex 0 contains the number of DP modules with DP output data of the DP slave. | UINT8 | RO | |
| 1A00:01 | | first DP module with DP output data of the DP slave | UINT32 | RO | |
| ... | | .. | | | |
| 1A00:m | | last DP module with DP output data of the DP slave | UINT32 | RO | |

### Index 1A7F TxPDO-Map Status PDO

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1A7F:0 | TxPDO-Map Status PDO | This PDO contains the status information of the DP slave (object 0xF100 [▶ 172]) | UINT8 | RO | 4 |
| 1A7F:01 | | 1st PDO Mapping entry (object 0xF100 (DP Status), entry 0x01 (DP Communication Status)) | UINT32 | RO | 0xF100:01, 08 |
| 1A7F:02 | | 2nd PDO Mapping entry (6 Bit Align) | UINT32 | RO | 0x0000:00, 06 |
| 1A7F:03 | | 3rd PDO Mapping entry (object 0xF100 (DP Status), entry 0x0F (TxPDO State)) | UINT32 | RO | 0xF100:0F, 01 |
| 1A7F:04 | | 4th PDO Mapping entry (object 0x1800 (TxPDO-Par Status PDO), entry 0x09 (TxPDO Toggle)) | UINT32 | RO | 0x1800:09, 01 |

### Index 1C00 Sync manager type

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

### Index 1C12 RxPDO assign

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1C12:0 | RxPDO assign | PDO Assign Outputs: if the DP slave has DP inputs that are sent to the DP master, then the RxPDO Assign contains the RxPDO 0x1600, otherwise it is empty (SubIndex 0 = 0) | UINT8 | RW | 0x01 ($1_{dec}$) |
| 1C12:01 | | 1st assigned RxPDO (index 0x1600) | UINT16 | RW | 0x1600 ($5632_{dec}$) |

### Index 1C13 TxPDO assign

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1C13:0 | TxPDO assign | PDO Assign Inputs: if the DP slave has DP outputs that are received by the DP master, then the TxPDO Assign contains the TxPDOs 0x1A00 and 0x1A7F, otherwise only the TxPDO 0x1A7F | UINT8 | RW | 0x02 ($2_{dec}$) |
| 1C13:01 | | 1st assigned TxPDO (index 0x1A00) | UINT16 | RW | 0x1A00 ($6656_{dec}$) |
| 1C13:02 | | 2nd assigned TxPDO (index 0x1A7F) | UINT16 | RW | 0x1A7F ($6783_{dec}$) |

### Index 1C32 SM output parameter

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1C32:0 | SM output parameter | Synchronization parameters for the outputs | UINT8 | RO | 0x20 ($32_{dec}$) |
| 1C32:01 | Sync mode | Current synchronization mode:<br>• 0: Free Run | UINT16 | RW | 0x0000 ($0_{dec}$) |
| 1C32:02 | Cycle time | Cycle time (in ns):<br>• Cycle time of the EtherCAT master | UINT32 | RW | 0x000F4240 ($1000000_{dec}$) |
| 1C32:03 | Shift time | not supported | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C32:04 | Sync modes supported | Supported synchronization modes:<br>• Bit 1 = 1: Synchron with SM 2 event is supported<br>• Bit 2-3 = 01: DC mode is supported<br>• Bit 14 = 1: dynamic times (measurement through writing of 0x1C32:08 [▶ 169]) | UINT16 | RO | 0x0001 ($1_{dec}$) |
| 1C32:05 | Minimum cycle time | not supported | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C32:06 | Calc and copy time | not supported | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C32:08 | Command | not supported | UINT16 | RW | 0x0000 ($0_{dec}$) |
| 1C32:09 | Delay time | not supported | UINT32 | RO | 0x00000000 ($0_{dec}$) |
| 1C32:0B | SM event missed counter | not supported | UINT16 | RO | 0x0000 ($0_{dec}$) |
| 1C32:0C | Cycle exceeded counter | not supported | UINT16 | RO | 0x0000 ($0_{dec}$) |
| 1C32:0D | Shift too short counter | not supported | UINT16 | RO | 0x0000 ($0_{dec}$) |
| 1C32:20 | Sync error | not supported | BOOLEAN | RO | FALSE |

### Index 1C33 SM input parameter

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 1C33:0 | SM input parameter | Synchronization parameters for the inputs (as for 0x1C32 [▶ 169]) | UINT8 | RO | 0x20 ($32_{dec}$) |
| 1C33:01 | Sync mode | as 0x1C32:01 [▶ 169] | UINT16 | RO | 0x0000 ($0_{dec}$) |
| ... | | | | | |
| 1C33:20 | Sync error | as 0x1C32:32 [▶ 169] | BOOLEAN | RO | 0x00 ($0_{dec}$) |

### 6.2.2.2 Profile-specific objects (0x6000-0xFFFF)

The profile-specific objects have the same meaning for all EtherCAT slaves that support the profile 5001.

#### Index 6000 Input Data

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 6000:0 | Input Data | This object contains the DP output data of the DP slave (data are received by the DP master ). There is an entry in the object 0x6000 for each DP module with DP output data configured in object 0x8002 [▶ 171]. DP module m thereby corresponds to SubIndex m; if a DP module does not contain any DP output data, then this SubIndex does not exist. If the DP slave does not contain any DP output data, then object 0x6000 does not exist. SubIndex 0 contains the number of DP modules of the DP slave. | UINT8 | RO | |
| 6000:01 | | if the first DP module has DP output data: DP output data of the first DP module | OCTET-STRING | RO | |
| ... | | | | | |
| 6000:m | | if the m$^{th}$ DP module has DP output data: DP output data of the m$^{th}$ DP module | OCTET-STRING | RO | |

#### Index 7000 Output Data

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 7000:0 | Output Data | This object contains the DP input data of the DP slave (data are sent to the DP master). There is an entry in object 0x7000 for each DP module with DP input data configured in object 0x8002 [▶ 171]. DP module m thereby corresponds to SubIndex m; if a DP module does not contain any DP input data, then this SubIndex does not exist. If the DP slave does not contain any DP input data, then object 0x7000 does not exist. SubIndex 0 contains the number of DP modules of the DP slave. | UINT8 | RO | |
| 7000:01 | | if the first DP module has DP input data: DP input data of the first DP module | OCTET-STRING | RO | |
| ... | | | | | |
| 7000:m | | if the m$^{th}$ DP module has DP input data: DP input data of the m$^{th}$ DP module | OCTET-STRING | RO | |

### Index 8000 DP Slave Parameter Set Slave

| Index (hex) | Name | Meaning | | Data type | Flags | Default |
|---|---|---|---|---|---|---|
| 8000:0 | DP Slave Parameter Set Slave yyy | This object contains the DP configuration of the (n+1)th configured DP slave (0 <= n <125). The object is to be transmitted with Complete Access, or SubIndex 0 must first be set to 0, then the individual SubIndexes transmitted (non-existent SubIndexes or gaps are thereby to be omitted) and finally SubIndex 0 set to the correct value. | | UINT8 | RW | 0x33 ($51_{dec}$) |
| 8000:01 | Station Address | DP station address of the DP slave, permitted values: 0-125 | | UINT16 | RW | |
| 8000:04 | Device type | DP Ident Number of the DP slave | | UINT32 | RW | |
| 8000:1D | Network Flags | reserved for AMS via DP | | UINT16 | RW | 0x0000 ($0_{dec}$) |
| 8000:1E | Network Port | reserved for AMS via DP | | UINT16 | RW | 0x0000 ($0_{dec}$) |
| 8000:1F | Network Segment Address | reserved for AMS via DP | | OCTET-STRING[6] | RW | {0} |
| 8000:20 | SI_Flag | SI_flag | | UINT8 | RW | 0xA8 ($168_{dec}$) |
| | | Bits 0-2 | Reserve, must be 0 | | | |
| | | Bit 3 | Reserve, must be 1 | | | |
| | | Bit 4 | Reserve, must be 0 | | | |
| | | Bit 5 | Reserve, must be 1 | | | |
| | | Bit 6 | Reserve, must be 0 | | | |
| | | Bit 7 | Reserve, must be 1 | | | |
| 8000:21 | Slave_Type | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| 8000:22 | Max_Diag_Data_Len | Reserve, must be 244 | | UINT8 | RW | 0xF4($244_{dec}$) |
| 8000:23 | Max_Alarm_Len | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| 8000:24 | Max_Channel_Data_Length | Reserve, must be 240 | | UINT8 | RW | 0xF0($240_{dec}$) |
| 8000:25 | Diag_Upd_Delay | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| 8000:26 | Alarm_Mode | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| 8000:27 | Add_SI_Flag | Reserve, must be 1 | | UINT8 | RW | 0x01 ($1_{dec}$) |
| 8000:28 | C1_Timeout | Reserve, must be 1000 | | UINT16 | RW | 0x03E8 ($1000_{dec}$) |
| 8000:29 | Number of tolerated Data Exchange with no answer | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| 8000:2A | Parallel AoE Services | Reserve, must be 5 | | UINT8 | RW | 0x05 ($5_{dec}$) |
| 8000:2B | Reaction on no answer | Reserve, must be 0 | | BIT1 | RW | 0x00 ($0_{dec}$) |
| 8000:2C | Restart behavior after DP fault | Reserve, must be 0 | | BIT1 | RW | 0x00 ($0_{dec}$) |
| 8000:2D | Master reaction after DP fault | Reserve, must be 0 | | BIT1 | RW | 0x00 ($0_{dec}$) |
| 8000:2E | Changes of DP Inputs after DP fault | 0 | Data of TxPDO n are set to 0 | BIT1 | RW | |
| | | 1 | Data of TxPDO n remain unchanged | | | |
| 8000:2F | PKW supported | Reserve, must be 0 | | BOOLEAN | RW | FALSE |
| 8000:30 | FDL only | Reserve, must be 0 | | BOOLEAN | RW | FALSE |
| 8000:31 | Watchdog Base 1 ms | Reserve, must be 0 | | BOOLEAN | RW | FALSE |
| 8000:33 | Cycle Access Divider | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| 8000:34 | Cycle Access Modulo | Reserve, must be 0 | | UINT8 | RW | 0x00 ($0_{dec}$) |
| 8000:35 | Vendor specific reserved | Reserve, must be 0 | | OCTET-STRING[25] | RW | {0} |

### Index 8002 DP CfgData Slave

| Index (hex) | Name | Meaning | Data type | Flags | Default |
|---|---|---|---|---|---|
| 8002:0 | DP CfgData Slave | This object contains the DP Cfg Data of the DP slave. | OCTET-STRING[244] | RW | {0} |

### Index F000 Modular device profile

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

### Index F100 DP Status

| Index (hex) | Name | Meaning | | Data type | Flags | Default |
|---|---|---|---|---|---|---|
| F100:0 | DP Status | This object contains the DP status data, which are mapped into TxPDO 128 (Index 0x1A7F [▶ 168]) | | UINT8 | RO | |
| F100:01 | DP Communication Status | 0 | Slave is exchanging data | UINT8 | RO | |
| | | 128 | Slave is ready for data exchange | | | |
| | | 129 | Slave is waiting for the ChkCfg telegram | | | |
| | | 130 | Slave is waiting for the SetPrm telegram | | | |
| F100:0E | Sync Error | this is set if a synchronization error occurred in the preceding cycle in Distributed Clocks mode | | BOOLEAN | RO | |
| F100:10 | TxPDO State | this is set if the DP slave is not exchanging data | | BOOLEAN | RO | |
| F100:0F | TxPDO Toggle | this toggles if new DP outputs received by the DP master have been entered in the EtherCAT input buffer for the first time after an EtherCAT input update. | | BOOLEAN | RO | |

# 7 Appendix

## 7.1 EtherCAT AL Status Codes

For detailed information please refer to the EtherCAT system description.

## 7.2 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 [▶ 174]. 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!

| EL6731-0000 | | | |
|---|---|---|---|
| **Hardware (HW)** | **Firmware** | **Revision no.** | **Date of release** |
| 06 - 07 | 07 | EL6731-0000-0016 | 2008/03 |
| | 08 | EL6731-0000-0017 | 2008/04 |
| | 09 | EL6731-0000-0018 | 2008/11 |
| 08 - 13 | 10 | EL6731-0000-0020 | 2009/04 |
| | | EL6731-0000-0021 | 2012/01 |
| 14 - 22 | 11 | | 2012/02 |
| | 12 | | 2012/07 |
| | | EL6731-0000-0022 | 2012/10 |
| | | EL6731-0000-0023 | 2013/11 |
| | 13 | EL6731-0000-0024 | 2014/07 |
| 23- 32* | 14 | EL6731-0000-0025 | 2014/06 |
| | 15 | | 2014/10 |
| | 17 | | 2016/03 |
| | 18 | | 2018/07 |
| | 19 | | 2020/02 |
| | 20 | | 2020/07 |
| | 21* | | 2020/07 |

| EL6731-0010 | | | |
|---|---|---|---|
| **Hardware (HW)** | **Firmware** | **Revision no.** | **Date of release** |
| 05 - 07 | 04 | EL6731-0010-0016 | 2008/05 |
| | 05 | | 2008/09 |
| | 06 | EL6731-0010-0017 | 2008/11 |
| 07 | 07 | | 2008/12 |
| 08 - 20 | 08 | | 2009/09 |
| | 09 | | 2010/10 |
| | | EL6731-0010-0018 | 2012/10 |
| | | EL6731-0010-0019 | 2013/11 |
| | 10 | EL6731-0010-0020 | 2014/07 |
| 21 - 30* | 11 | EL6731-0010-0021 | 2014/06 |
| | 12 | | 2014/10 |
| | 13 | | 2018/07 |
| | 14 | | 2020/07 |
| | 15* | | 2020/07 |

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

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

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

### Storage locations

An EtherCAT slave stores operating data in up to 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.de). All ESI files are accessible there as zip files.

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

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

### Simplified update by bundle firmware

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

- for the firmware to be in a packed format: recognizable by the file name, which also contains the revision number, e.g. ELxxxx-xxxx_REV0016_SW01.efw
- for password=1 to be entered in the download dialog. If password=0 (default setting) only the firmware update is carried out, without an ESI update.

- for the device to support this function. The function usually cannot be retrofitted; it is a component of many new developments from year of manufacture 2016.

Following the update, its success should be verified

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

**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.

### 7.3.1 Device description ESI file/XML

**NOTE**

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

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

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

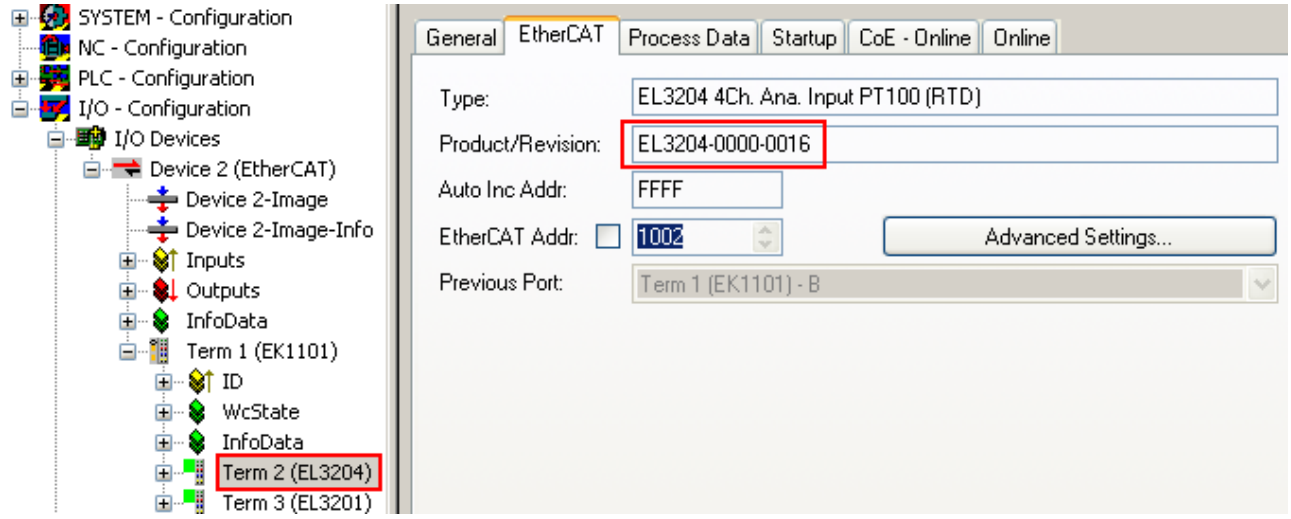



Fig. 138: Device identifier consisting of name EL3204-0000 and revision -0016

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

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

**Update of XML/ESI description**

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

### Display of ESI slave identifier

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

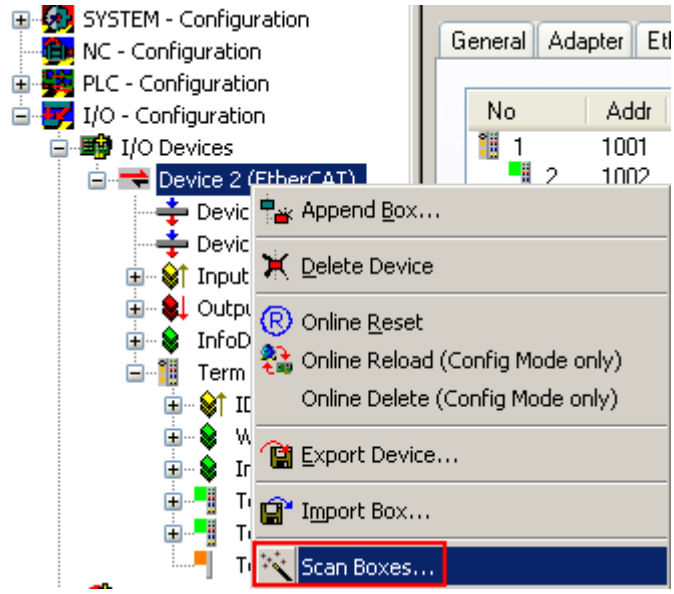



Fig. 139: Scan the subordinate field by right-clicking on the EtherCAT device

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

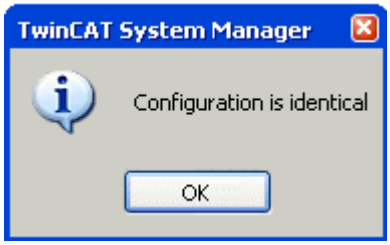



Fig. 140: Configuration is identical

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

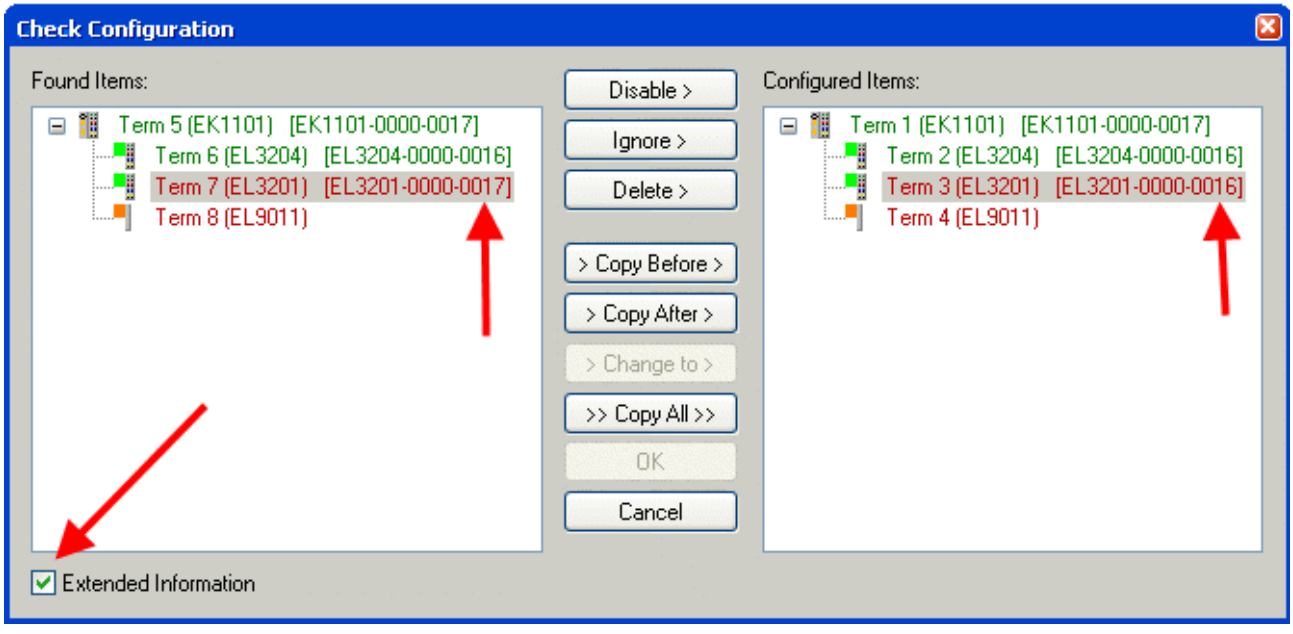



Fig. 141: Change dialog

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*

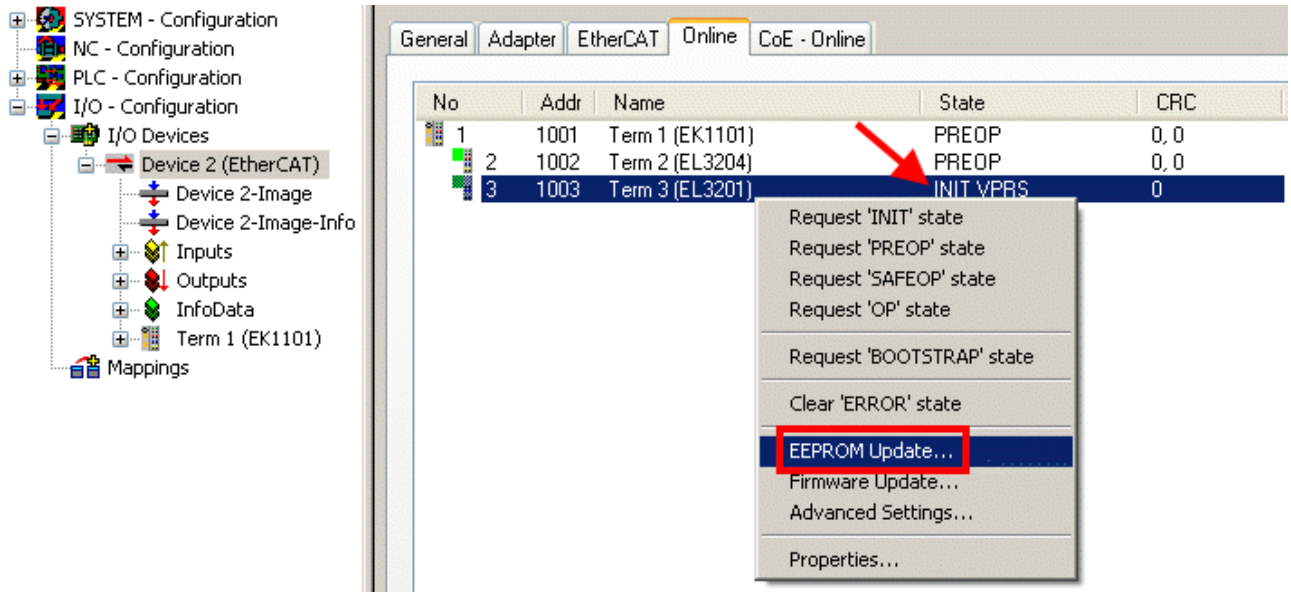



Fig. 142: EEPROM Update

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.

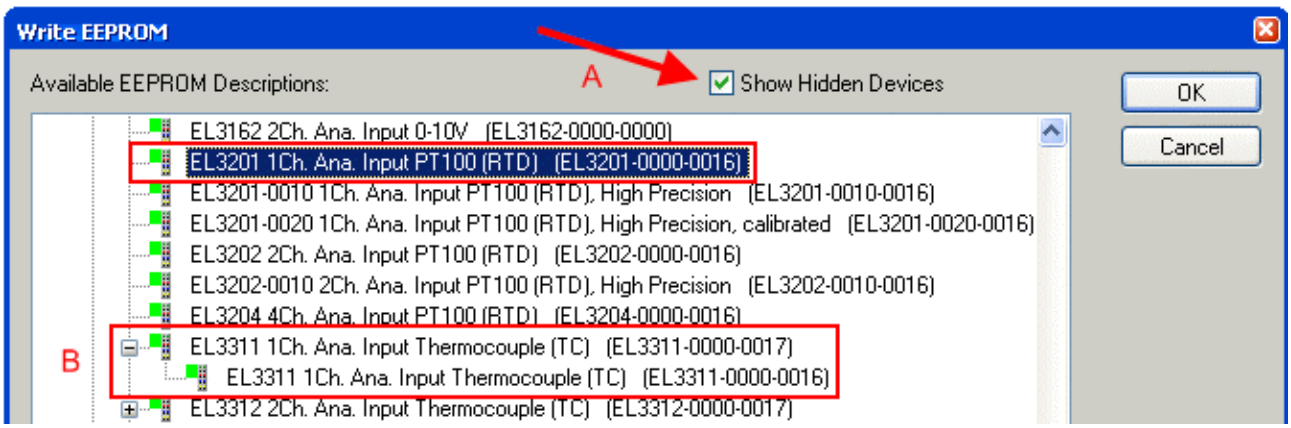



Fig. 143: Selecting the new ESI

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

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

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

## 7.3.2 Firmware explanation

### Determining the firmware version

### Determining the version on laser inscription

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

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

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

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

### Determining the version via the System Manager

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

**CoE Online and Offline CoE**

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.

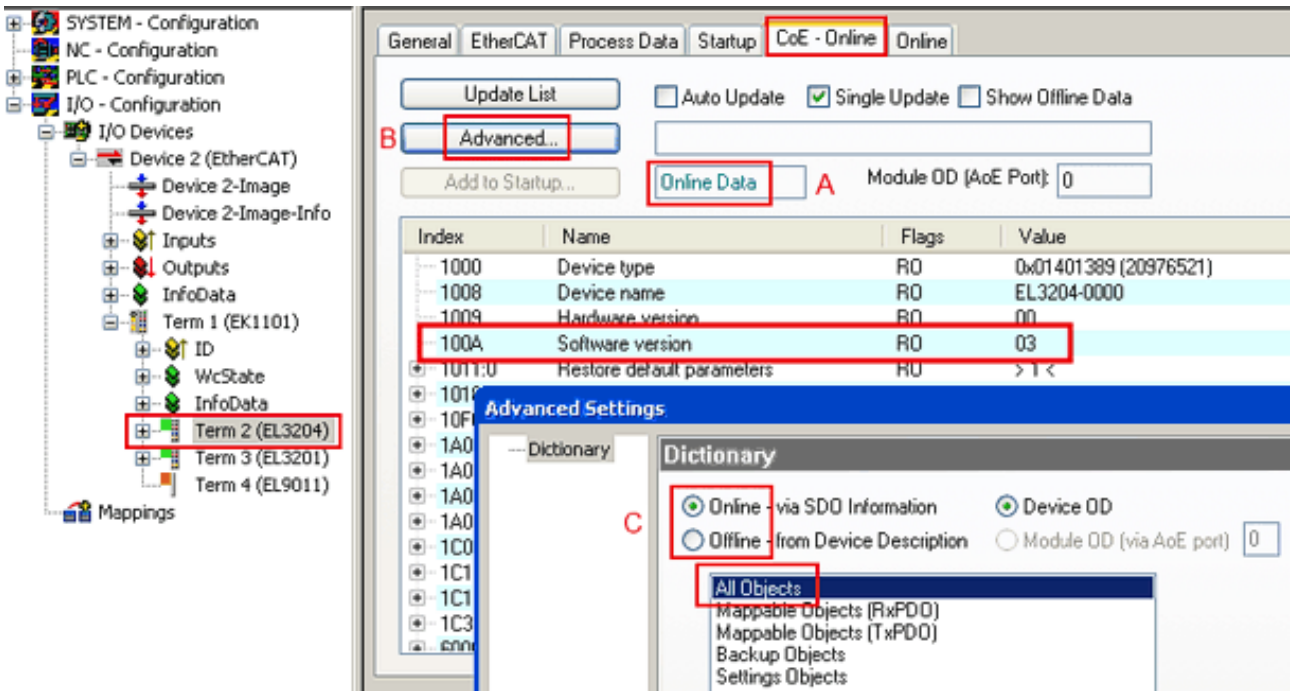



Fig. 144: Display of EL3204 firmware version

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

## 7.3.3 Updating controller firmware *.efw

**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.*

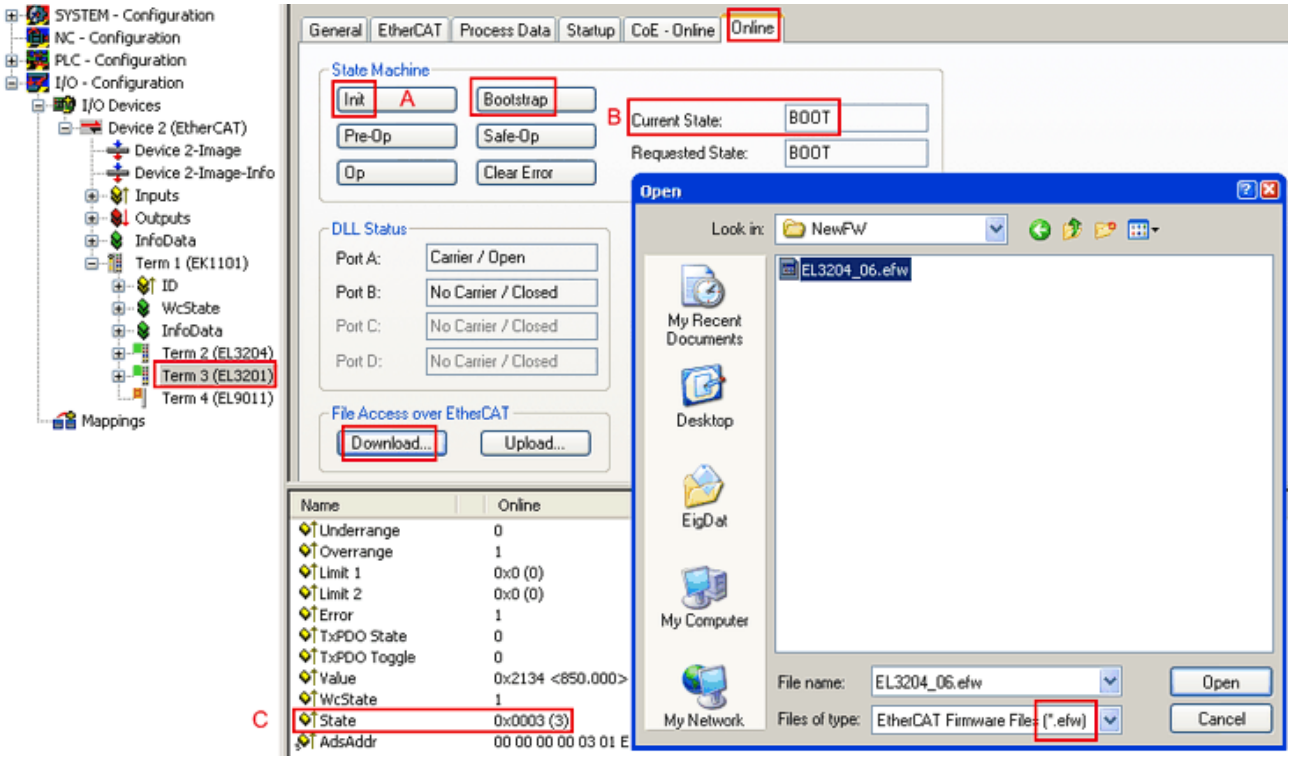
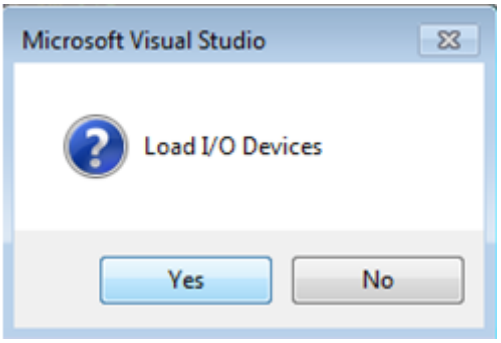
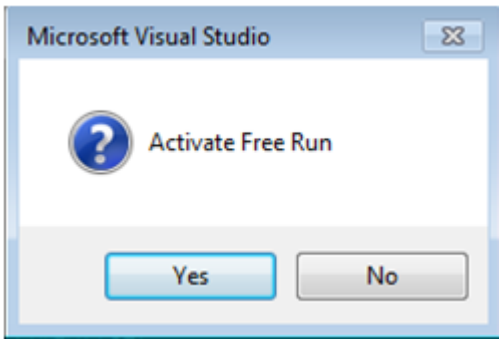



Fig. 145: Firmware Update

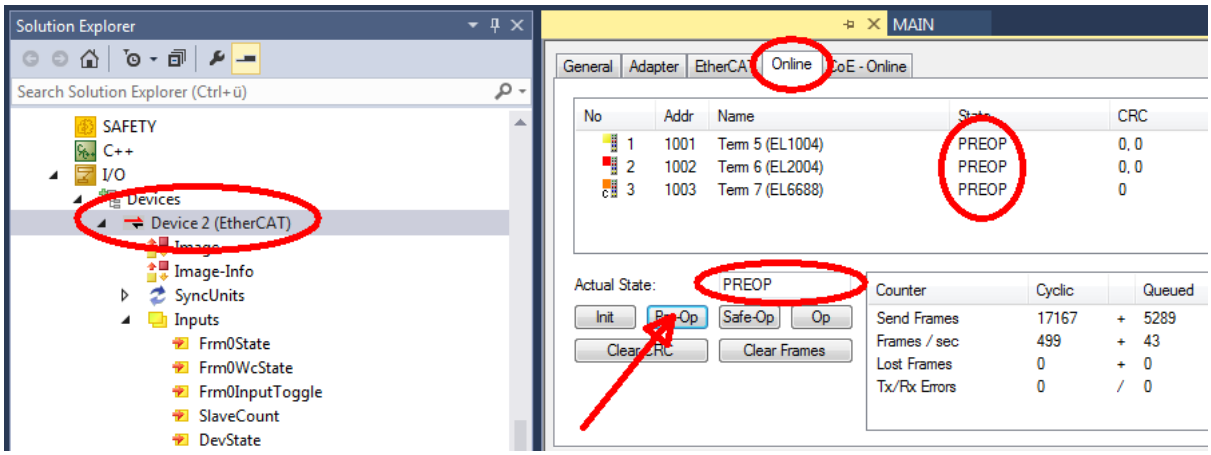
Proceed as follows, unless instructed otherwise by Beckhoff support. Valid for TwinCAT 2 and 3 as EtherCAT master.

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

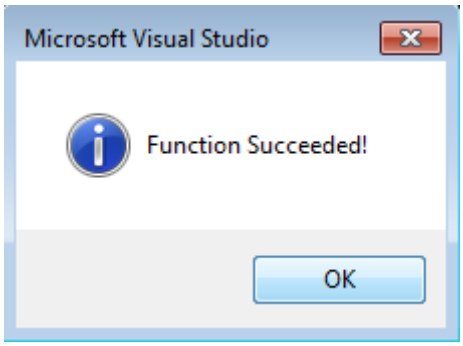


- Switch EtherCAT Master to PreOP




- Switch slave to INIT (A)
- Switch slave to BOOTSTRAP
- Check the current status (B, C)
- Download the new *efw file (wait until it ends). A pass word will not be neccessary usually.




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

## 7.3.4 FPGA firmware *.rbf

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

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

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

### Determining the version via the System Manager

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

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

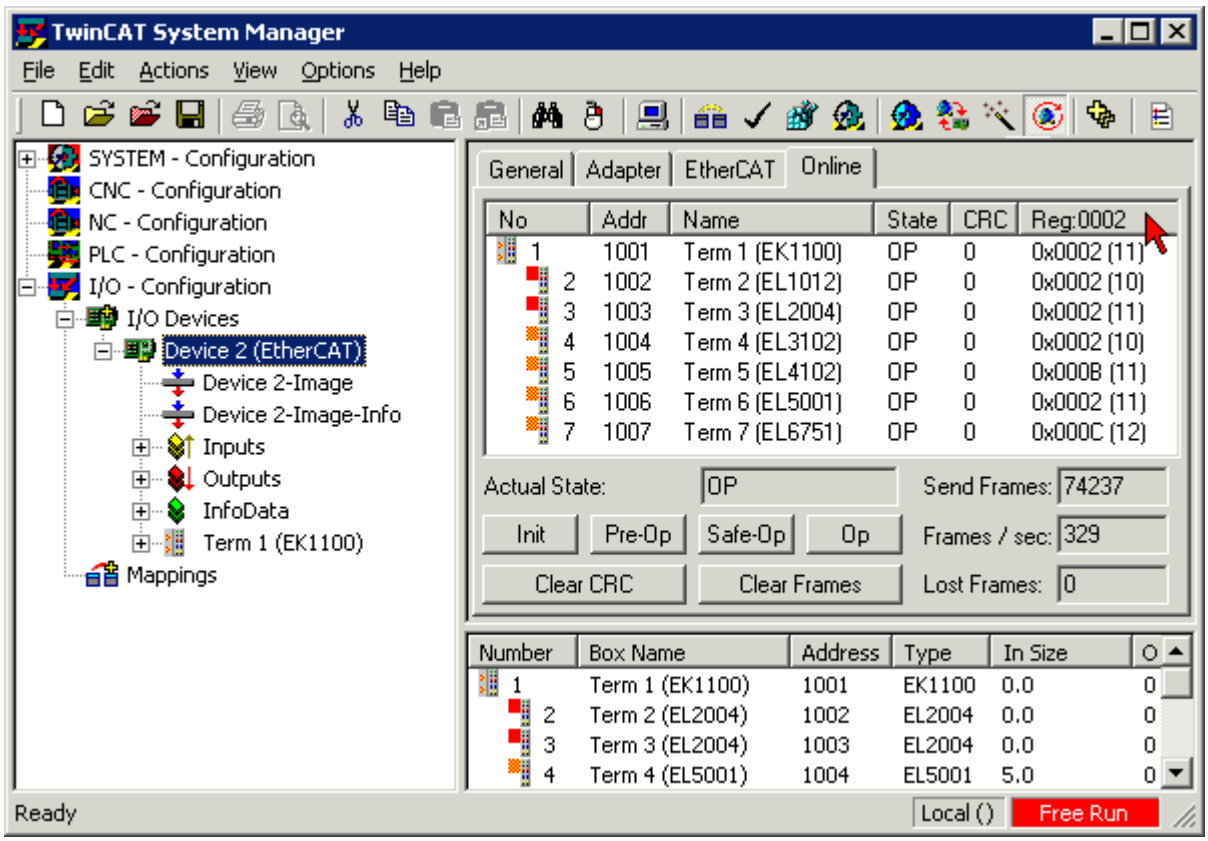



Fig. 146: FPGA firmware version definition

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

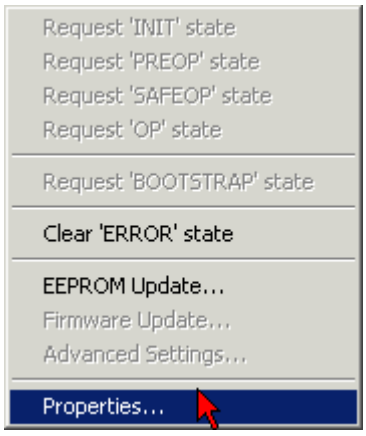



Fig. 147: Context menu *Properties*

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.

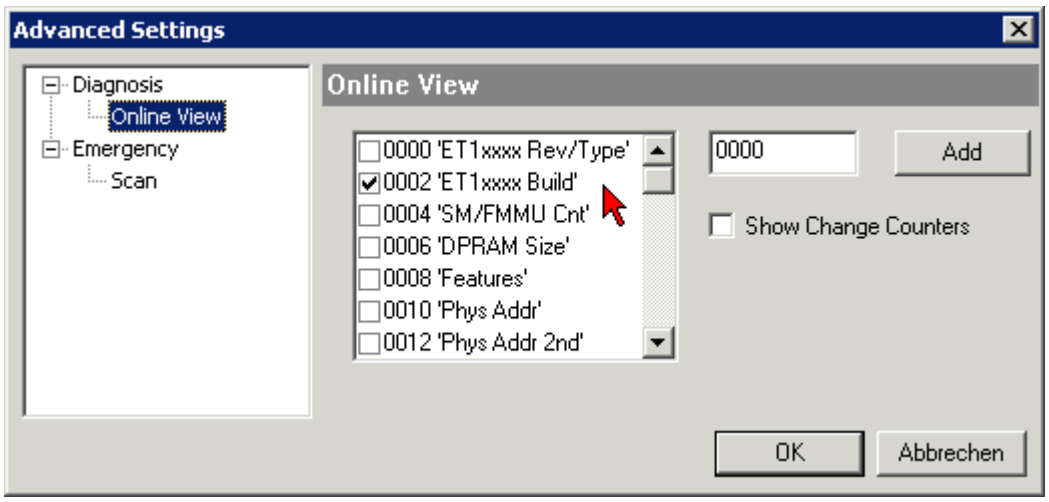



Fig. 148: Dialog *Advanced Settings*

### Update

For updating the FPGA firmware

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

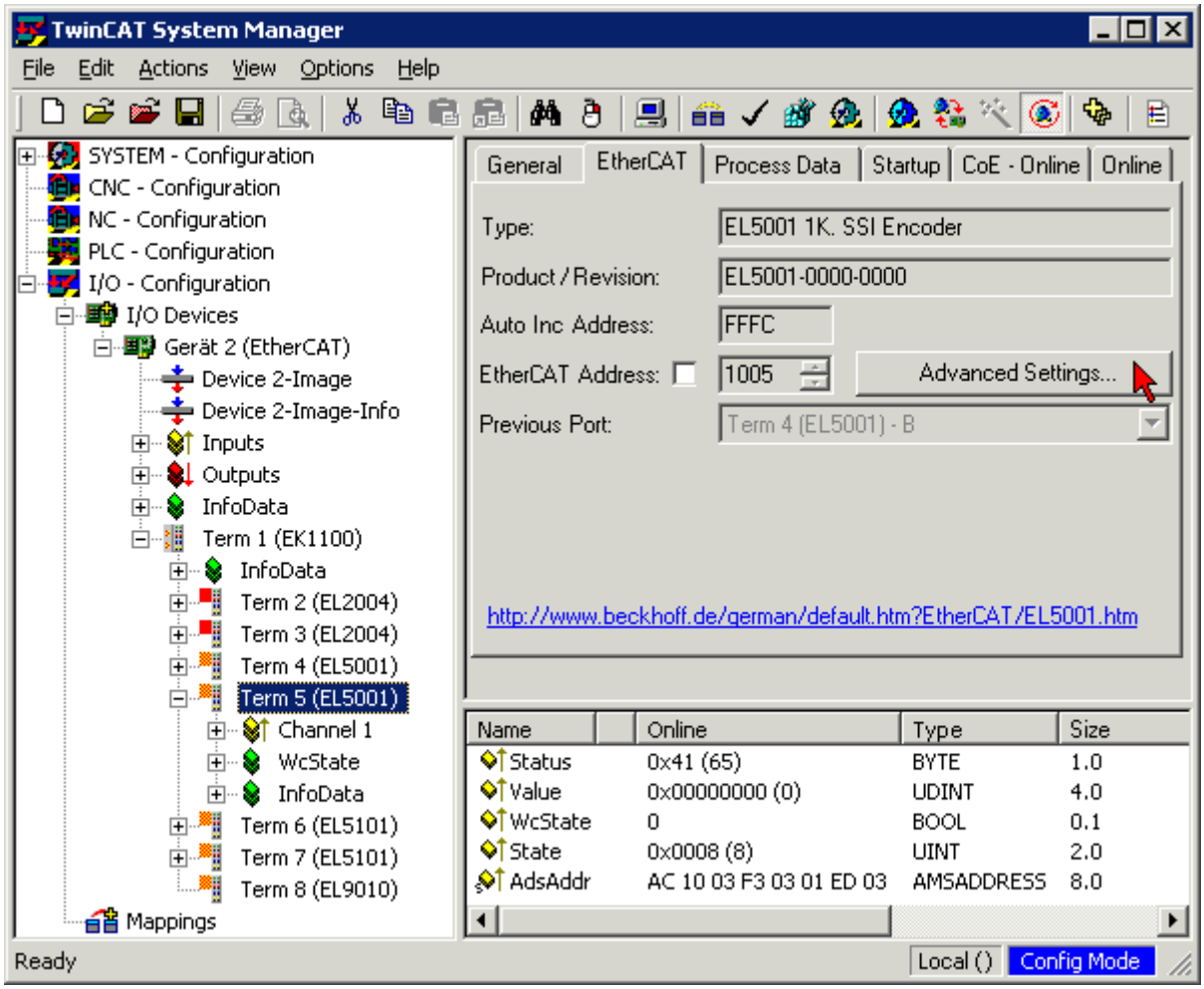
Older firmware versions can only be updated by the manufacturer!

### Updating an EtherCAT device

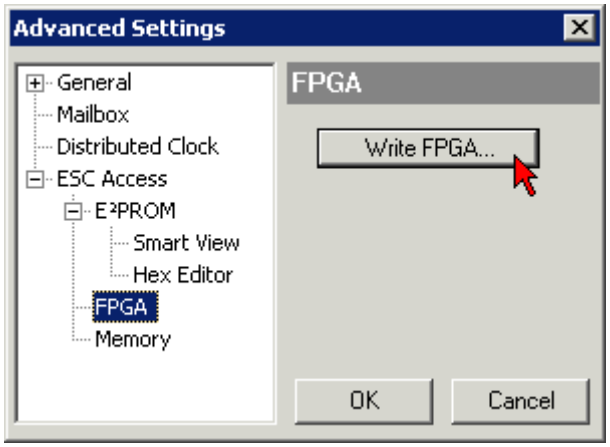
The following sequence order have to be met if no other specifications are given (e.g. by the Beckhoff support):

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

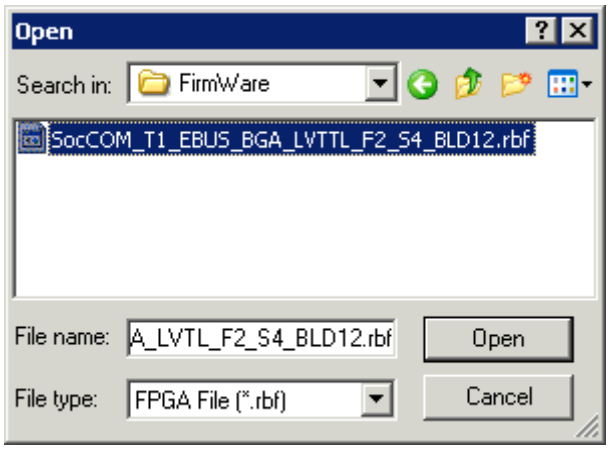
- In the TwinCAT System Manager select the terminal for which the FPGA firmware is to be updated (in the example: Terminal 5: EL5001) and
  click the *Advanced Settings* button in the *EtherCAT* tab:




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



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




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

**NOTE**

**Risk of damage to the device!**

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

### 7.3.5 Simultaneous updating of several EtherCAT devices

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

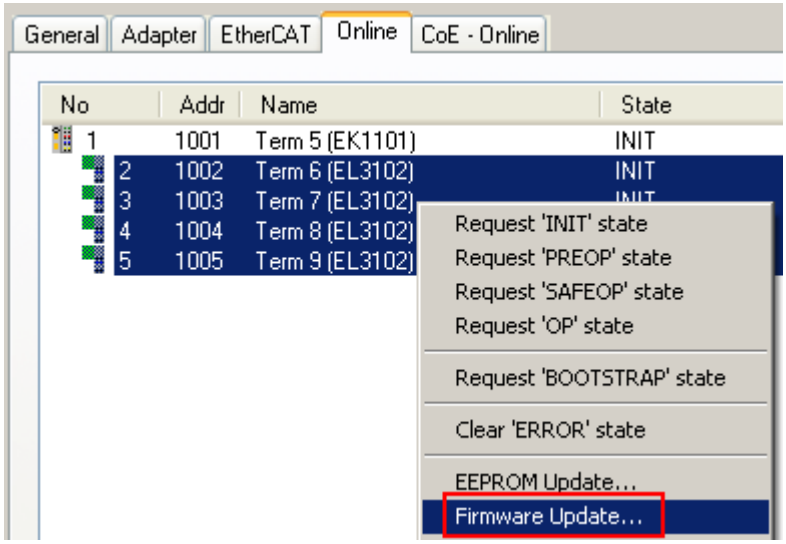



Fig. 149: Multiple selection and firmware update

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

## 7.4 Support and Service

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

**Beckhoff's branch offices and representatives**

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

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

You will also find further documentation for Beckhoff components there.

**Beckhoff Headquarters**

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

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

**Beckhoff Support**

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

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

| | |
|---|---|
| Hotline: | +49 5246 963 157 |
| Fax: | +49 5246 963 9157 |
| e-mail: | support@beckhoff.com |

**Beckhoff Service**

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

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

| | |
|---|---|
| Hotline: | +49 5246 963 460 |
| Fax: | +49 5246 963 479 |
| e-mail: | service@beckhoff.com |

# Table of figures


Fig. 1 EL5021 EL terminal, standard IP20 IO device with serial/ batch number and revision ID (since 2014/01) ... 9
Fig. 2 EK1100 EtherCAT coupler, standard IP20 IO device with serial/ batch number ... 10
Fig. 3 CU2016 switch with serial/ batch number ... 10
Fig. 4 EL3202-0020 with serial/ batch number 26131006 and unique ID-number 204418 ... 10
Fig. 5 EP1258-00001 IP67 EtherCAT Box with batch number/ date code 22090101 and unique serial number 158102 ... 11
Fig. 6 EP1908-0002 IP67 EtherCAT Safety Box with batch number/ date code 071201FF and unique serial number 00346070 ... 11
Fig. 7 EL2904 IP20 safety terminal with batch number/ date code 50110302 and unique serial number 00331701 ... 11
Fig. 8 ELM3604-0002 terminal with unique ID number (QR code) 100001051 and serial/ batch number 44160201 ... 11
Fig. 9 BIC as data matrix code (DMC, code scheme ECC200) ... 12
Fig. 10 EL6731 ... 14
Fig. 11 Spring contacts of the Beckhoff I/O components ... 16
Fig. 12 Recommended distances for standard installation position ... 17
Fig. 13 Other installation positions ... 18
Fig. 14 Correct positioning ... 22
Fig. 15 Incorrect positioning ... 22
Fig. 16 PROFIBUS cable assignment ... 23
Fig. 17 Pin assignment M12 socket (-B310) ... 25
Fig. 18 Pin assignment M12 socket/plug connector (-B318) ... 25
Fig. 19 Pin assignment of the PROFIBUS D-sub socket ... 25
Fig. 20 Pin assignment socket/plug connector Fieldbus Box modules ... 26
Fig. 21 EtherCAT tab -> Advanced Settings -> Behavior -> Watchdog ... 30
Fig. 22 States of the EtherCAT State Machine ... 32
Fig. 23 “CoE Online” tab ... 34
Fig. 24 Startup list in the TwinCAT System Manager ... 35
Fig. 25 Offline list ... 36
Fig. 26 Online list ... 36
Fig. 27 Adding a DP slave ... 40
Fig. 28 EL6731 task cycle ... 41
Fig. 29 Cycle with IO at task start ... 42
Fig. 30 Cycle with IO not at task start ... 42
Fig. 31 TwinCAT task, DP cycle and K-bus cycle sequence ... 43
Fig. 32 System Manager “Options” (TwinCAT 2) ... 55
Fig. 33 Call up under VS Shell (TwinCAT 3) ... 55
Fig. 34 Overview of network interfaces ... 55
Fig. 35 EtherCAT device properties(TwinCAT 2): click on “Compatible Devices…” of tab “Adapte”” ... 56
Fig. 36 Windows properties of the network interface ... 56
Fig. 37 Exemplary correct driver setting for the Ethernet port ... 57
Fig. 38 Incorrect driver settings for the Ethernet port ... 58
Fig. 39 TCP/IP setting for the Ethernet port ... 59
Fig. 40 Identifier structure ... 60
Fig. 41 OnlineDescription information window (TwinCAT 2) ... 61

Fig. 42 Information window OnlineDescription (TwinCAT 3) ..... 61
Fig. 43 File OnlineDescription.xml created by the System Manager ..... 62
Fig. 44 Indication of an online recorded ESI of EL2521 as an example ..... 62
Fig. 45 Information window for faulty ESI file (left: TwinCAT 2; right: TwinCAT 3) ..... 62
Fig. 46 Append EtherCAT device (left: TwinCAT 2; right: TwinCAT 3) ..... 64
Fig. 47 Selecting the EtherCAT connection (TwinCAT 2.11, TwinCAT 3) ..... 64
Fig. 48 Selecting the Ethernet port ..... 64
Fig. 49 EtherCAT device properties (TwinCAT 2) ..... 65
Fig. 50 Appending EtherCAT devices (left: TwinCAT 2; right: TwinCAT 3) ..... 65
Fig. 51 Selection dialog for new EtherCAT device ..... 66
Fig. 52 Display of device revision ..... 66
Fig. 53 Display of previous revisions ..... 67
Fig. 54 Name/revision of the terminal ..... 67
Fig. 55 EtherCAT terminal in the TwinCAT tree (left: TwinCAT 2; right: TwinCAT 3) ..... 68
Fig. 56 Differentiation local/target system (left: TwinCAT 2; right: TwinCAT 3) ..... 69
Fig. 57 Scan Devices (left: TwinCAT 2; right: TwinCAT 3) ..... 69
Fig. 58 Note for automatic device scan (left: TwinCAT 2; right: TwinCAT 3) ..... 69
Fig. 59 Detected Ethernet devices ..... 70
Fig. 60 Example default state ..... 70
Fig. 61 Installing EthetCAT terminal with revision -1018 ..... 71
Fig. 62 Detection of EtherCAT terminal with revision -1019 ..... 71
Fig. 63 Scan query after automatic creation of an EtherCAT device (left: TwinCAT 2; right: TwinCAT 3) ..... 71
Fig. 64 Manual triggering of a device scan on a specified EtherCAT device (left: TwinCAT 2; right: TwinCAT 3) ..... 72
Fig. 65 Scan progressexemplary by TwinCAT 2 ..... 72
Fig. 66 Config/FreeRun query (left: TwinCAT 2; right: TwinCAT 3) ..... 72
Fig. 67 Displaying of “Free Run” and “Config Mode” toggling right below in the status bar ..... 72
Fig. 68 TwinCAT can also be switched to this state by using a button (left: TwinCAT 2; right: TwinCAT 3) ..... 72
Fig. 69 Online display example ..... 73
Fig. 70 Faulty identification ..... 73
Fig. 71 Identical configuration (left: TwinCAT 2; right: TwinCAT 3) ..... 74
Fig. 72 Correction dialog ..... 74
Fig. 73 Name/revision of the terminal ..... 75
Fig. 74 Correction dialog with modifications ..... 76
Fig. 75 Dialog “Change to Compatible Type…” (left: TwinCAT 2; right: TwinCAT 3) ..... 76
Fig. 76 TwinCAT 2 Dialog Change to Alternative Type ..... 76
Fig. 77 Configuring the process data ..... 78
Fig. 78 Selection of the diagnostic information of an EtherCAT Slave ..... 79
Fig. 79 Basic EtherCAT Slave Diagnosis in the PLC ..... 80
Fig. 80 EL3102, CoE directory ..... 82
Fig. 81 Example of commissioning aid for a EL3204 ..... 83
Fig. 82 Default behaviour of the System Manager ..... 84
Fig. 83 Default target state in the Slave ..... 84
Fig. 84 PLC function blocks ..... 85
Fig. 85 Illegally exceeding the E-Bus current ..... 86

Fig. 86 Warning message for exceeding E-Bus current ..... 86
Fig. 87 EL6731 tab ..... 87
Fig. 88 Bus parameters dialog ..... 88
Fig. 89 Startup/fault settings dialog ..... 90
Fig. 90 MC tab ..... 91
Fig. 91 ADS tab ..... 93
Fig. 92 DP Diag tab ..... 93
Fig. 93 Box states tab ..... 94
Fig. 94 MC Diag tab ..... 95
Fig. 95 Profibus tab ..... 97
Fig. 96 Features tab ..... 98
Fig. 97 <Beckhoff> tab ..... 99
Fig. 98 Process data tab ..... 100
Fig. 99 The PrmData (text) tab ..... 101
Fig. 100 Diag tab ..... 101
Fig. 101 Inserting an FC31xx PCI I/O device ..... 103
Fig. 102 Configuration 1 ..... 104
Fig. 103 Configuration 2 ..... 105
Fig. 104 Inserting a box ..... 106
Fig. 105 Selection dialog ..... 106
Fig. 106 Profibus tab, address selection ..... 107
Fig. 107 Process Data tab, telegram assignment ..... 108
Fig. 108 Process Data tab, determine telegram ..... 109
Fig. 109 MC tab, determine PB - DP cycle ..... 110
Fig. 110 NC task, cycle time ..... 111
Fig. 111 Configuration, linking of axes ..... 112
Fig. 112 PROFIdrive tab, reading motor parameters ..... 113
Fig. 113 “Global” tab, scaling factor entry ..... 114
Fig. 114 LEDs ..... 115
Fig. 115 Diagram showing a normal DP cycle ..... 116
Fig. 116 Diagram showing the first occurrence of a faulty DP cycle ..... 116
Fig. 117 Diagram showing subsequent DP cycles ..... 116
Fig. 118 Diagram showing a normal DP cycle ..... 117
Fig. 119 Diagram: First faulty and subsequent DP cycles in the "Stay in Data-Exch (for WD-Time)" mode ..... 117
Fig. 120 Diagnostic inputs for EL6731 in the TwinCAT tree ..... 119
Fig. 121 Input DpState in the TwinCAT tree ..... 120
Fig. 122 Diagram showing allocation of DP slaves to several DP cycles ..... 130
Fig. 123 Diagram showing several DP cycles in a task cycle ..... 130
Fig. 124 Diagram showing several data samples within one task cycle ..... 131
Fig. 125 Adding a „Profibus Slave EL6731, EtherCAT“ in the TwinCAT tree ..... 131
Fig. 126 Append modules ..... 132
Fig. 127 DP master, ADS tab: Activating the ADS interface and entering the Net ID ..... 134
Fig. 128 DP slave, ADS tab: Entering the DP master Net ID with “Add. Net IDs” ..... 134
Fig. 129 Flow chart: Sequence of the configuration of the EL6731 with start SDOs ..... 136
Fig. 130 Flow chart: Sequence of the configuration of the EL6731 with backup parameter storage ..... 138

Fig. 131 Sequence of the DP cycle without distributed clocks ..... 139
Fig. 132 DP cycle with distributed clocks and control via SM and SYNC0 event ..... 140
Fig. 133 Transfer of output data with LWR, input data with LRD telegram (Separate Input Update = TRUE, Task Cycle Time = Base Time) ..... 141
Fig. 134 Transfer of output data with LWR, input data with LRD telegram (Separate Input Update = TRUE, Task Cycle Time = 4*Base Time, Pre ticks = 1) ..... 141
Fig. 135 DP cycle start via SYNC0, first telegram is sent after the SYNC1 event ..... 142
Fig. 136 Flow chart: Sequence of the configuration of the EL6731-0010 with start SDOs ..... 162
Fig. 137 Flow chart: Sequence of the configuration of the EL6731-0010 with backup parameter storage 164
Fig. 138 Device identifier consisting of name EL3204-0000 and revision -0016 ..... 175
Fig. 139 Scan the subordinate field by right-clicking on the EtherCAT device ..... 176
Fig. 140 Configuration is identical ..... 176
Fig. 141 Change dialog ..... 176
Fig. 142 EEPROM Update ..... 177
Fig. 143 Selecting the new ESI ..... 177
Fig. 144 Display of EL3204 firmware version ..... 178
Fig. 145 Firmware Update ..... 179
Fig. 146 FPGA firmware version definition ..... 181
Fig. 147 Context menu Properties ..... 181
Fig. 148 Dialog Advanced Settings ..... 182
Fig. 149 Multiple selection and firmware update ..... 184

More Information:
**beckhoff.de/EL6731/**

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