Documentation

EL6752

Master/Slave Terminal for DeviceNet

Version: 2.2
Date: 2020-03-16

BECKHOFF

Table of contents

1 Foreword

1.1 Notes on the documentation

Intended audience

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

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
2.2	• Update structure
2.1	• Chapter “Explicit messages” added • Update chapter “Technical data” • Update structure • Update revision status
2.0	• Migration • Update structure
1.4	• Addendum: chapter "Configuration": changing DeviceNet address and baud rate via ADS • Update structure
1.3	• Correction to chapter "Technical data" • Addendum: chapter "Firmware status" • Update structure
1.2	• Corrections to chapter "Mounting and wiring"
1.1	• Corrections to chapter "Mounting and wiring"
1.0	• Corrections and addenda, first publication
0.2	• Corrections and addenda
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

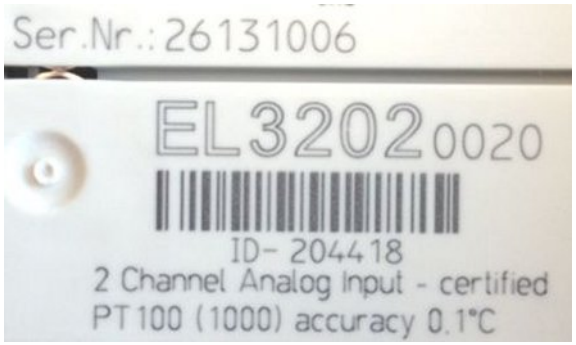


Fig. 4: EL3202-0020 with serial/ batch number 26131006 and unique ID-number 204418

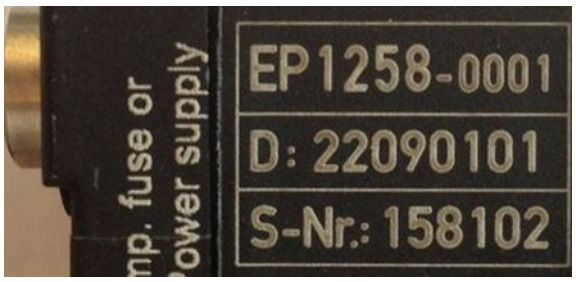


Fig. 5: EP1258-00001 IP67 EtherCAT Box with batch number/ date code 22090101 and unique serial number 158102

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.

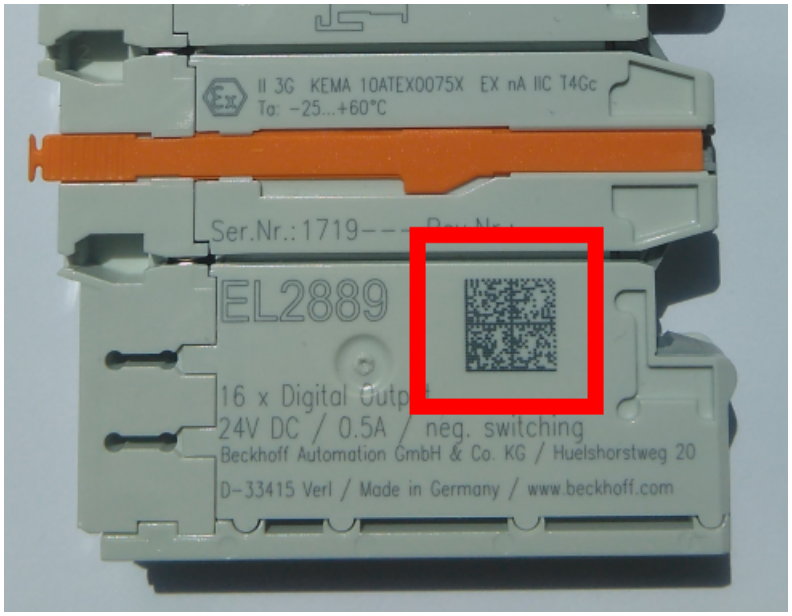


Fig. 9: BIC as data matrix code (DMC, code scheme ECC200)

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

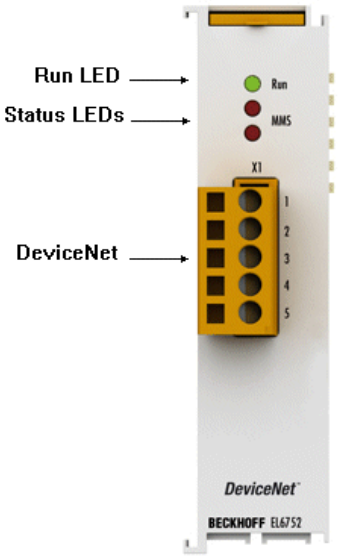


Fig. 10: EL6752

Master and slave terminals for DeviceNet

The master and slave terminals for DeviceNet correspond to the FC5201 PCI card from Beckhoff. Thanks to the connection via EtherCAT, no PCI slots are required in the PC. Within an EtherCAT terminal network, the terminal enables the integration of any DeviceNet devices.
The EL6752 is optionally available in a master or slave version and has a powerful protocol implementation with many features:

- All I/O modes of the DeviceNet are supported: polling, change of state, cyclic, strobed
- Unconnected message manager (UCMM)
- Powerful parameter and diagnostics interfaces
- Error management freely configurable for each bus device

A description of all functionalities and operating modes can be found in the chapter "Configuration [▶ 67]" and the corresponding subsections.

2.2 Technical data

Technical data	EL6752-0000	EL6752-0010
Bus system	DeviceNet	
Variante	Master	Slave
Number of fieldbus channels	1	
Data transfer rate	125, 250 or 500 kbaud	
Bus interface	Open style 5-pin connector according to DeviceNet specification, galvanically isolated; card comes with connector.	
Bus devices	maximum 63 slaves	
Communication	DeviceNet network master (scanner)	DeviceNet - slave
Diagnostics	Status LEDs	
Power supply	via the E-bus	
Current consumption via E-bus	typ. 260 mA	
Electrical isolation	500 V (E-bus/CANopen)	
Configuration	with TwinCAT System Manager	
Weight	approx. 70 g	
Permissible ambient temperature range during operation	-25 °C ... +60 °C (extended temperature range) 0 °C ... +55 °C (according to cULus [▶ 99] for Canada and the USA) 0 °C ... +55 °C (according to ATEX [▶ 29], see special conditions [▶ 29])	
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	
Mounting [▶ 17]	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 ATEX [▶ 29] cULus [▶ 99]	

3 Basic DeviceNet principles

Introduction to the system

DeviceNet is an open system based on CAN. CAN was developed some years ago by R. Bosch for data transmission in motor vehicles. Millions of CAN chips are now in use. A disadvantage for application in automation is that CAN does not contain definitions for the application layer. CAN only defines the physical and data link layer.

DeviceNet specifies a uniform application layer and this makes it possible to use the CAN protocol for industrial applications. ODVA (the Open DeviceNet Vendor Association) is an independent association which supports manufacturers and users of the DeviceNet system. ODVA ensures that all devices which conform to the specification can operate together in one system, regardless of their manufacturer. CAN's bit arbitration procedure makes it theoretically possible to operate communication networks using master/slave and multimaster access methods.
Further details can be found on the official website of the ODVA (http://www.odva.org).

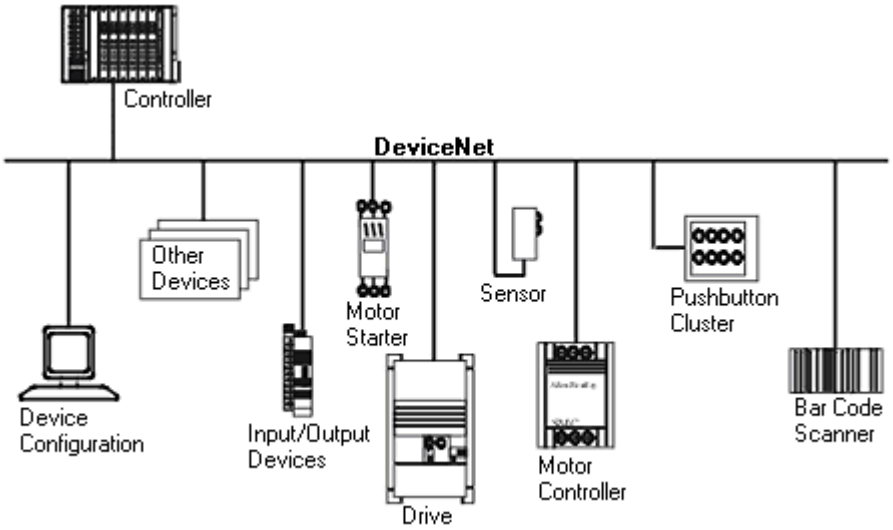


Fig. 11: Example of DeviceNet in use

Bus cable

The bus cable consists of two pairs of shielded twisted-pair wiring, one for the data transfer and one for the power supply. The latter can carry currents of up to 8 amperes. The maximum possible length of a line depends essentially on the baud rate. If you choose the highest Baud rate (500 kbaud) you are restricted to lines of at most 100 m. With the lowest Baud rate (125 kbaud) you will be able to use cable with an overall length of 500 m. Refer to the chapter "Mounting and wiring [▶ 21]" for details

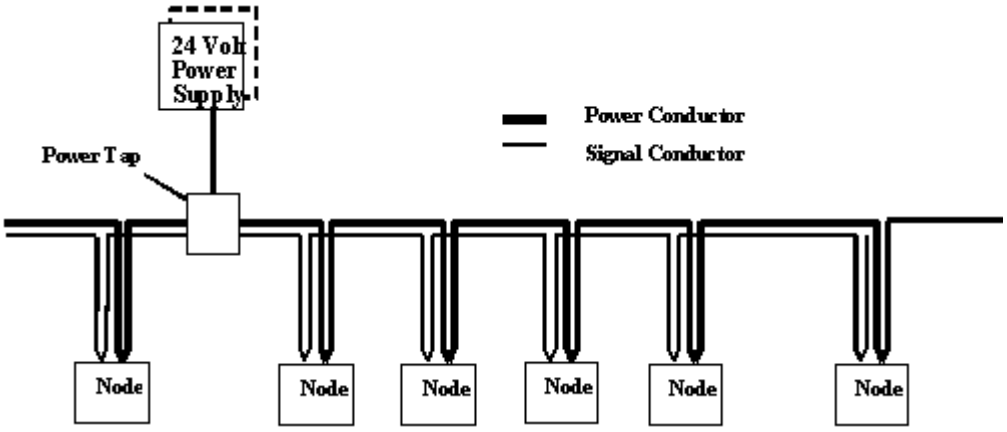


Fig. 12: Example of DeviceNet cabling

4 Mounting and cabling

4.1 Instructions for ESD protection

NOTE

Destruction of the devices by electrostatic discharge possible!

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

- Please ensure you are electrostatically discharged and avoid touching the contacts of the device directly.
- Avoid contact with highly insulating materials (synthetic fibers, plastic film etc.).
- Surroundings (working place, packaging and personnel) should 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.

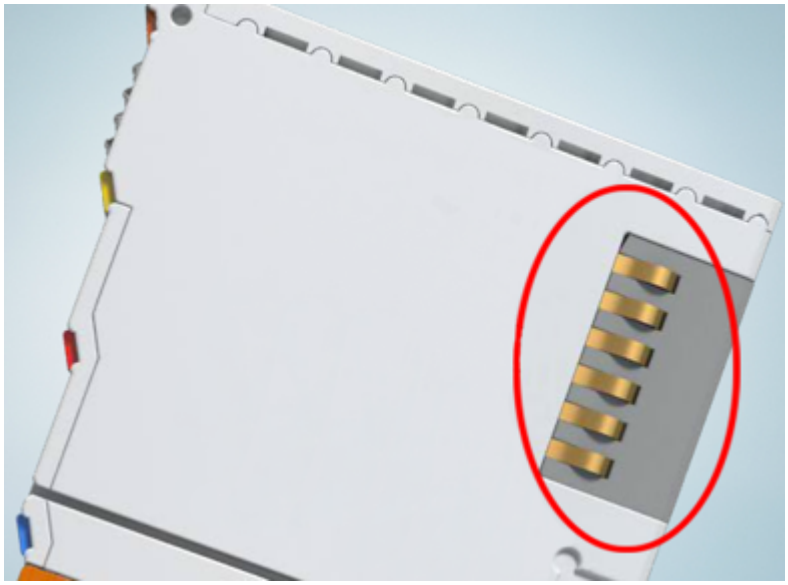


Fig. 13: Spring contacts of the Beckhoff I/O components

4.2 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)!

4.3 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).

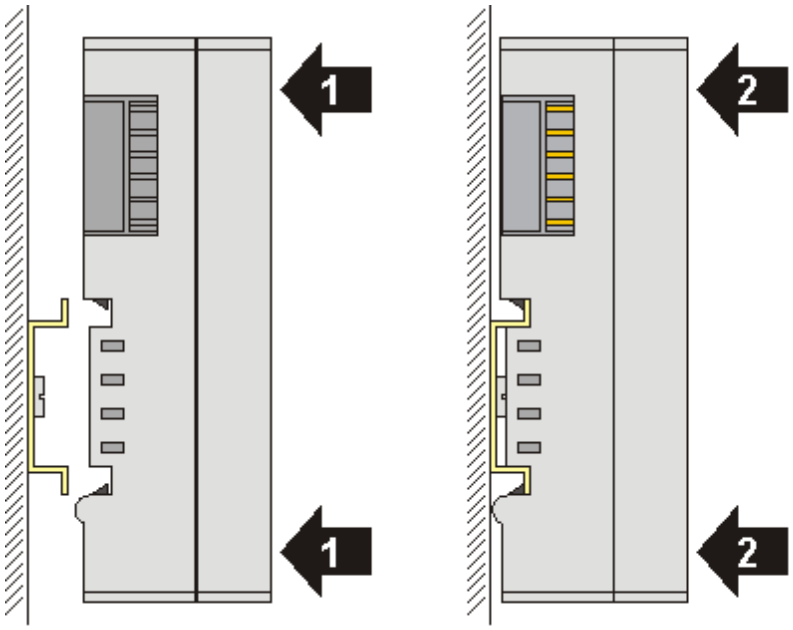
⚠ WARNING

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

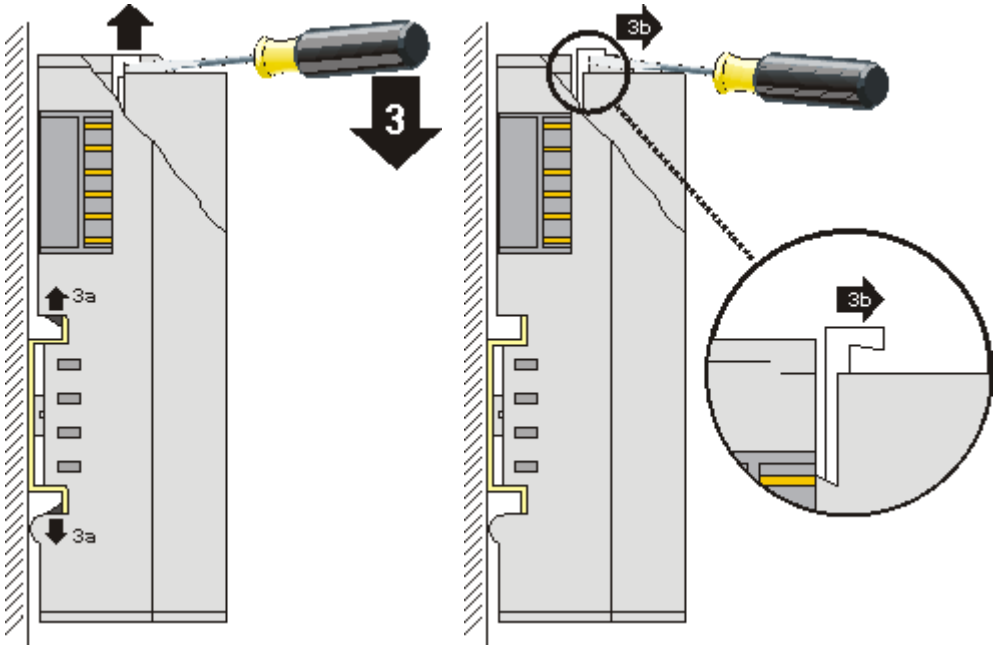
- Fit the mounting rail to the planned assembly location.



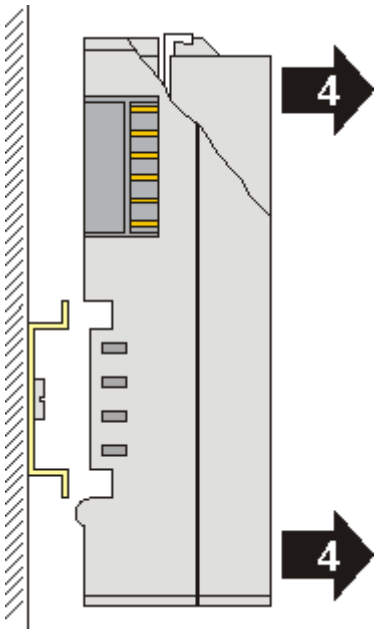
  and press (1) the terminal module against the mounting rail until it latches in place on the mounting rail (2).
- Attach the cables.

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.



4.4 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).

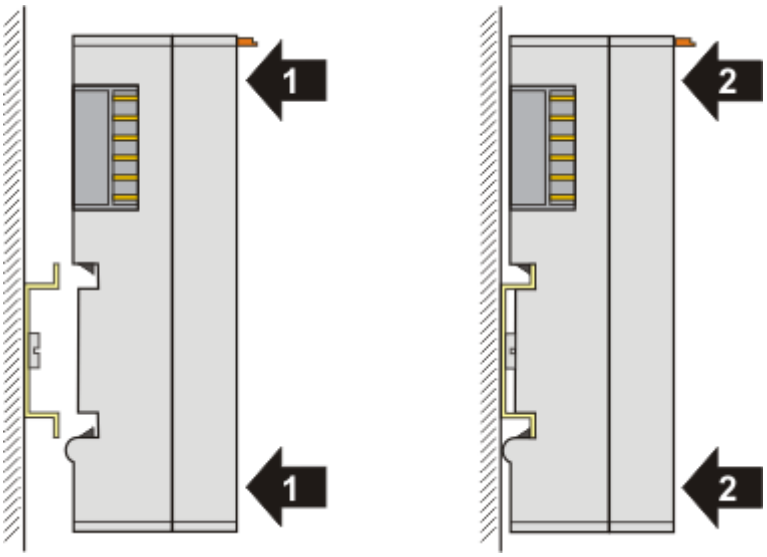
⚠ WARNING

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.

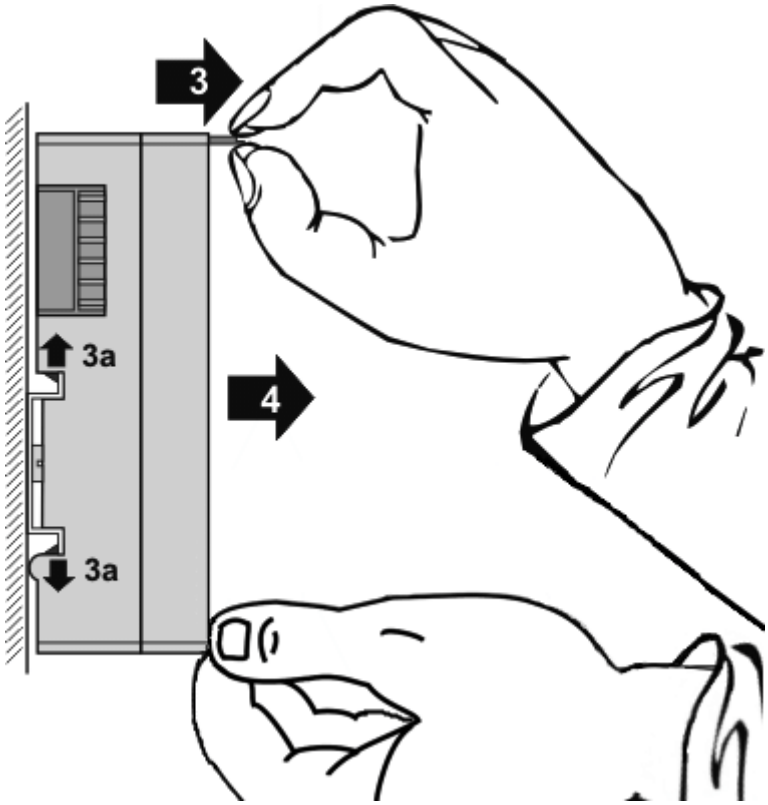


and press (1) the terminal module against the mounting rail until it latches in place on the mounting rail (2).

- 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.

4.5 DeviceNet wiring

4.5.1 CAN / DeviceNet topology

CAN/DeviceNet is a 2-wire bus system, to which all participating devices are connected in parallel (i.e. using short drop lines) (Fig. *DeviceNet Topology*). The bus must be terminated at each end with a 120 (or 121) Ohm terminating resistor to prevent reflections. This is also necessary even if the cable lengths are very short!

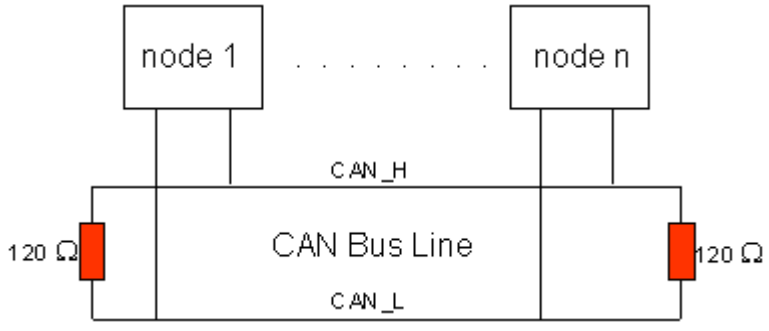


Fig. 14: DeviceNet topology

Since the CAN signals are represented on the bus as the difference between the two levels, the CAN leads are not very sensitive to incoming interference (EMI): Both leads are affected, so the interference has very little effect on the difference.

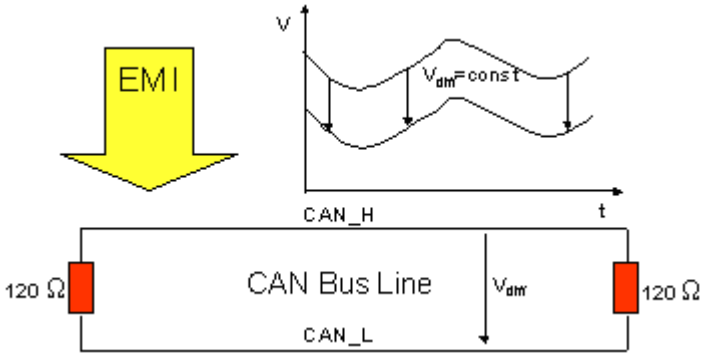


Fig. 15: Low interference through difference levels

4.5.2 Bus length

The maximum length of a CAN bus is primarily limited by the signal propagation delay. The multi-master bus access procedure (arbitration) requires signals to reach all the nodes at effectively the same time (before the sampling within a bit period). Since the signal propagation delays in the CAN connecting equipment (transceivers, opto-couplers, CAN controllers) are almost constant, the line length must be chosen in accordance with the baud rate:

Baud rate	Bus length
500 kbit/s	< 100 m
250 kbit/s	< 250 m
125 kbit/s	< 500 m

4.5.3 Drop lines

Drop lines must always be avoided as far as possible, since they inevitably cause reflections. The reflections caused by drop lines are not however usually critical, provided they have decayed fully before the sampling time. In the case of the bit timing settings selected in the Bus Couplers it can be assumed that this is the case, provided the following drop line lengths are not exceeded:

Baud rate	Drop line length	Total length of all drop lines
500 kbit/s	< 6 m	< 39 m
250 kbit/s	< 6 m	< 78 m
125 kbit/s	< 6 m	< 156 m

Drop lines must not be furnished with termination resistors (Fig. *Drop line topology*).

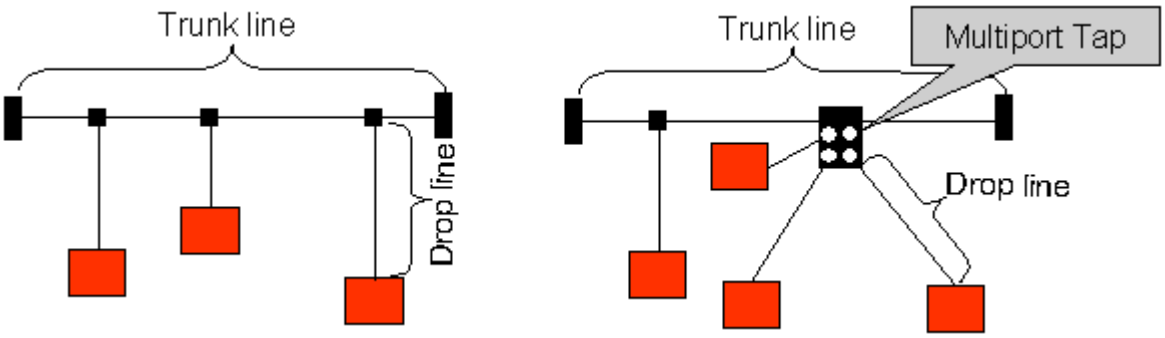


Fig. 16: Drop line topology

4.5.4 Star Hub (Multiport Tap)

Shorter drop line lengths must be maintained when passive distributors ("multiport taps"), such as the Beckhoff ZS5052-4500 Distributor Box. The following table indicates the maximum drop line lengths and the maximum length of the trunk line (without the drop lines):

Guide values

The following values are recommended by BECKHOFF.

Baud rate	Drop line length with multiport topology	Trunk line length (without drop lines)
500 kbit/s	< 1.2 m	< 66 m
250 kbit/s	< 2.4 m	< 120 m
125 kbit/s	< 4.8 m	< 310 m

4.5.5 CAN cable

Screened twisted-pair cables (2x2) with a characteristic impedance of between 108 and 132 Ohm is recommended for the CAN wiring. If the CAN transceiver's reference potential (CAN ground) is not to be connected, the second pair of conductors can be omitted. (This is only recommended for networks of small physical size with a common power supply for all the participating devices).

ZB5200 CAN/DeviceNet Cable

The ZB5200 cable material corresponds to the DeviceNet specification, and is also suitable for CANopen systems. The ready-made ZK1052-xxxx-xxxx bus cables for the Fieldbus Box modules are made from this cable material. It has the following specification:

- 2 x 2 x 0.34 mm² (AWG 22) twisted pairs
- double screened - braided screen with filler strand
- characteristic impedance (1 MHz): 126 ohm
- Conductor resistance 54 Ohm/km
- sheath: grey PVC, outside diameter 7.3 mm
- printed with "InterlinkBT DeviceNet Type 572" as well as UL and CSA ratings
- stranded wire colours correspond to the DeviceNet specification
- UL recognized AWM Type 2476 rating
- CSA AWM I/II A/B 80°C 300V FT1
- corresponds to the DeviceNet "Thin Cable" specification

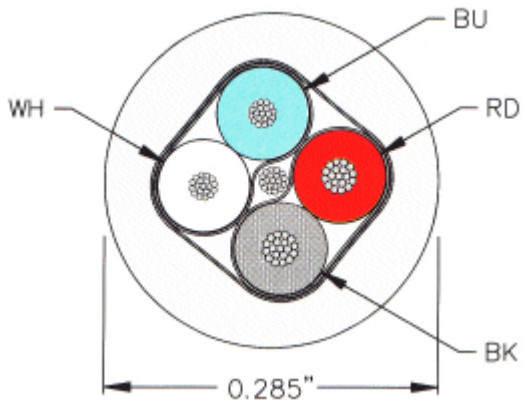


Fig. 17: DeviceNet cable configuration

4.5.6 Shielding

The screen is to be connected over the entire length of the bus cable, and only galvanically grounded at one point, in order to avoid ground loops.
The design of the screening, in which HF interference is diverted through R/C elements to the mounting rail assumes that the rail is appropriately earthed and free from interference. If this is not the case, it is possible that HF interference will be transmitted from the mounting rail to the screen of the bus cable. In that case the screen should not be attached to the couplers - it should nevertheless still be fully connected through.

4.5.7 Cable colours and pin assignment

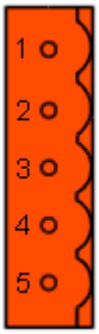


Fig. 18: Pin assignment (top view EL6752)

Suggested method of using the Beckhoff CAN cable on Bus Terminal and Fieldbus Box:

Pin	EL6752 assignment	ZB5200 cable color
1	V+ (24 V)	red
2	CAN High	white
3	Shield	Filler strand
4	CAN Low	blue
5	V-	black

4.6 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.

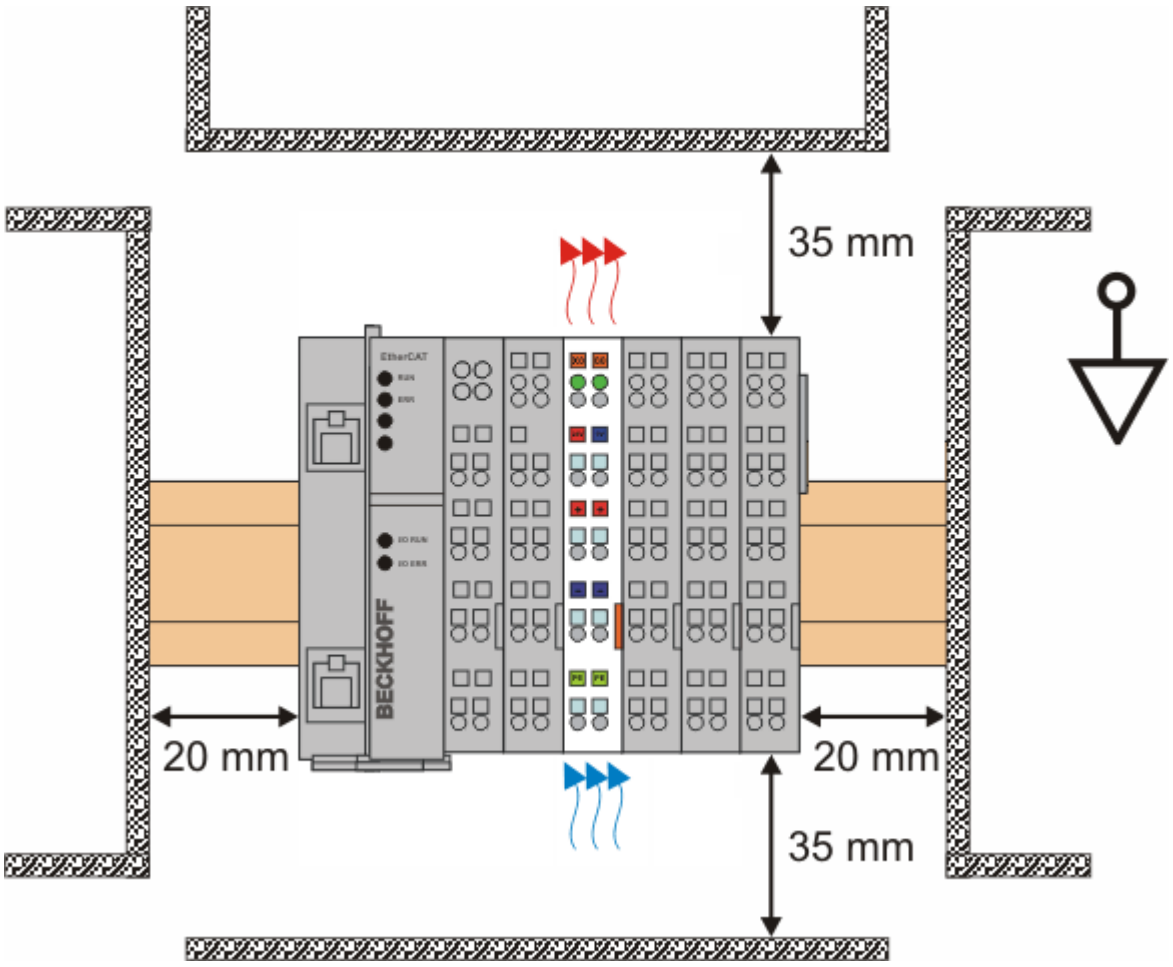


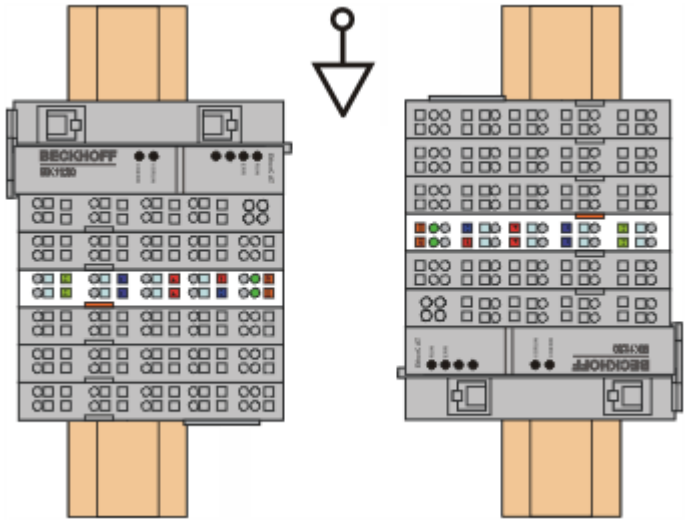
Fig. 19: Recommended distances for standard installation position

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.



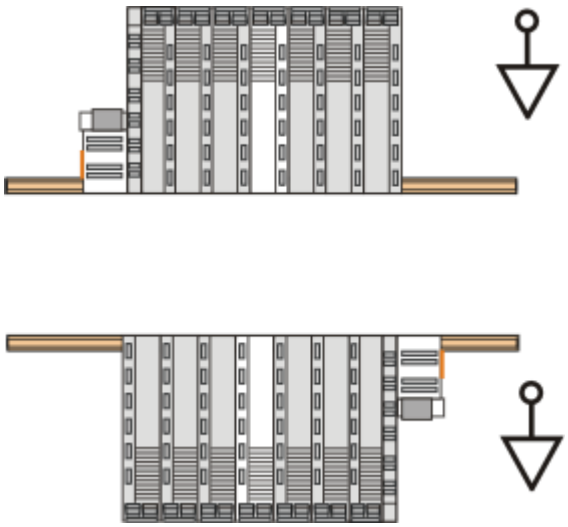


Fig. 20: Other installation positions

4.7 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 2 passive terminals!

Examples for positioning of passive terminals (highlighted)

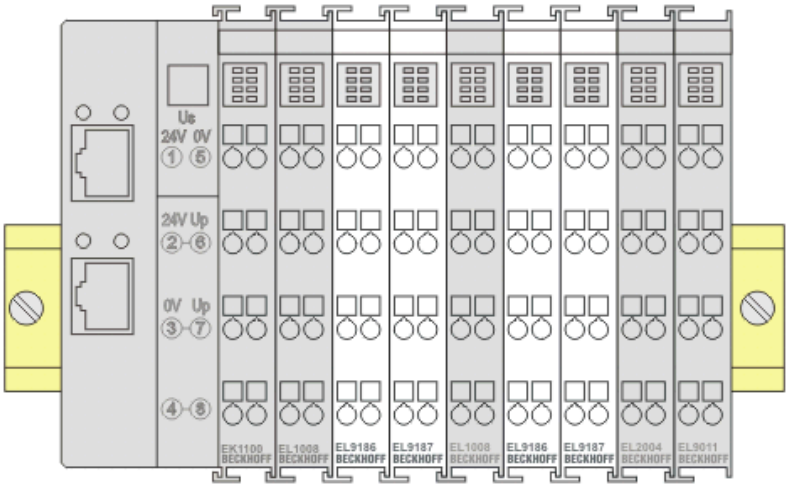


Fig. 21: Correct positioning

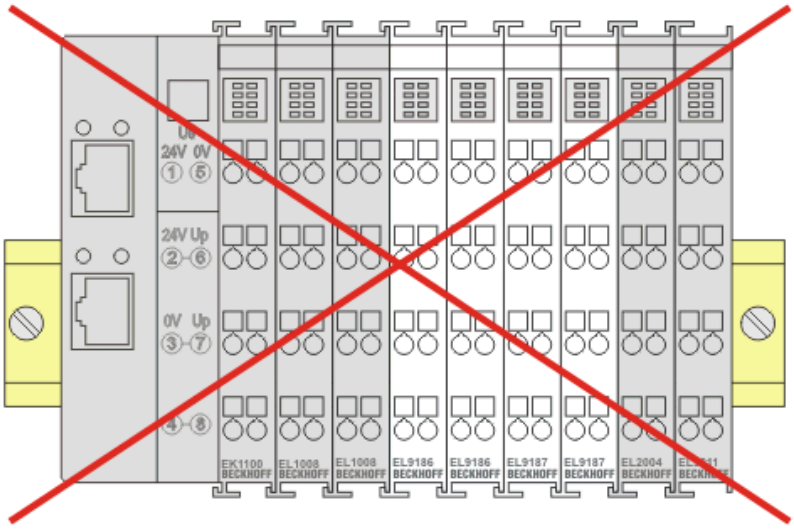


Fig. 22: Incorrect positioning

4.8 ATEX - Special conditions (standard temperature range)

⚠ WARNING

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

Standards

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

- EN 60079-0:2012+A11:2013
- EN 60079-15:2010

Marking

The Beckhoff fieldbus components with standard temperature range certified according to the ATEX directive for potentially explosive areas bear one of the following markings:

II 3G KEMA 10ATEX0075 X Ex nA IIC T4 Gc Ta: 0 … +55°C

or

II 3G KEMA 10ATEX0075 X Ex nC IIC T4 Gc Ta: 0 … +55°C

5 DeviceNet communication

5.1 DeviceNet Introduction

Fig. 23: DeviceNet

DeviceNet is an open system based on CAN. CAN was developed some years ago by R. Bosch for data transmission in motor vehicles. Millions of CAN chips are now in use. A disadvantage for application in automation is that CAN does not contain definitions for the application layer. CAN only defines the physical and data link layer.

DeviceNet specifies a uniform application layer and this makes it possible to use the CAN protocol for industrial applications. ODVA (the Open DeviceNet Vendor Association) is an independent association which supports manufacturers and users of the DeviceNet system. ODVA ensures that all devices which conform to the specification can operate together in one system, regardless of their manufacturer.

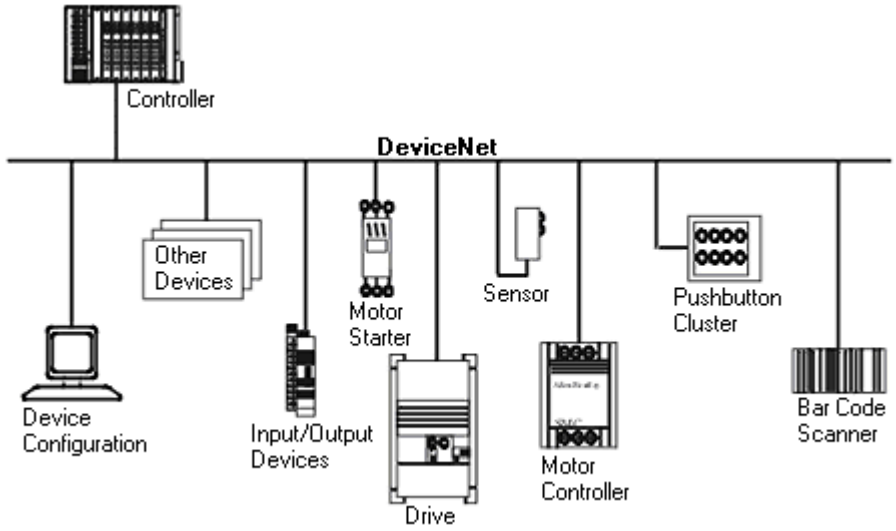


Fig. 24: Example of DeviceNet in use

DeviceNet is a sensor/actuator bus system. It is internationally standardised (EN50325) and is based on CAN (Controller Area Network). DeviceNet supports a number of communication types for the input and output data:

- Polling: The master module ("scanner") sends the output data cyclically to the assigned devices and receives the input data in an answer telegram.
- Change-of-State: Telegrams are sent as soon as their contents have changed.
- Cyclic: The modules send the data automatically after a cycle time has elapsed.
- Strobed: The scanner requests the input data using a broadcast telegram to all the devices.

The DeviceNet devices support all I/O communication types.

The DeviceNet devices are parameterized via acyclical services (explicit messaging).

The effective utilization of the bus bandwidth allows DeviceNet, particularly in Change-of-State mode, to achieve short system reaction times in spite of the relatively low data rates. The BECKHOFF DeviceNet devices have a powerful implementation of the protocol. Through active participation in the ODVA's technical committees, BECKHOFF are contributing to the further development of this bus system, and has in this way itself gathered profound DeviceNet expertise.

Configuration

The node address is set in the range from 0 to 63 using two decimally coded rotary switches. The data transfer rate set at the DeviceNet scanner is automatically recognized by the DeviceNet Box (auto baud rate). "Electronic Data Sheets" (EDS files) for DeviceNet configuration tools are available for download from the Beckhoff internet site (http://www.beckhoff.de), and on the BECKHOFF product CDs. Special I/O parameters that are not covered by the DeviceNet standard can be set via the KS2000 software (serial connection) or via acyclical explicit messages.

Diagnostics

The extensive diagnostic functions of the BECKHOFF DeviceNet devices allow rapid fault localisation. The diagnostic messages are transmitted over the bus and collated by the master. The status of the network connection, the device status, the status of the inputs and outputs and of the power supply are displayed by LEDs.

Data transfer rates

Three data transfer rates from 125 kbaud to 500 kbaud are available for different bus lengths. The effective utilization of the bus bandwidth allows DeviceNet to achieve short system reaction times at relatively low data rates.

Topology

DeviceNet is based on a linear topology. The number of devices participating in each network is logically limited by DeviceNet to 64, but physically the present generation of drivers allows up to 64 nodes in one network segment. The maximum possible size of the network for any particular data rate is limited by the signal propagation delay required on the bus medium. For 500 kbaud, for instance, the network may extend 100 m, whereas at 125 kbaud the network may reach up to 500 m. At low data rates the size of the network can be increased by repeaters, which also allow the construction of tree structures.

Bus access procedures

CAN utilizes the Carrier Sense Multiple Access (CSMA) procedure, i.e. all participating devices have the same right of access to the bus and may access it as soon as it is free (multi-master bus access). The exchange of messages is thus not device-oriented but message-oriented. This means that every message is unambiguously marked with a prioritized identifier. In order to avoid collisions on the bus when messages are sent by different devices, a bit-wise bus arbitration is carried out at the start of the data transmission. The bus arbitration assigns bus bandwidth to the messages in the sequence of their priority. At the end of the arbitration phase only one bus device occupies the bus, collisions are avoided and the bandwidth is optimally exploited.

Configuration and parameterization

The TwinCAT System Manager allows all the DeviceNet parameters to be set conveniently. An "eds" file (electronic data sheet) is available on the BECKHOFF website (http://www.beckhoff.de) for the parameterization of BECKHOFF DeviceNet devices using configuration tools from other manufacturers.

5.2 Explicit messages

Program example „ExplMessageEditor“: https://infosys.beckhoff.com/content/1033/el6752/Resources/zip/5979571979.zip

With the following ADS commands you can use EL6752 to send explicit messages.

```
GET_ATTRIBUTE_SINGLE via ADSRead Data Transfer
SET_ATTRIBUTE_SINGLE via ADSWrite Data Transfer
COMMON SERVICE via ADSReadWrite Data Transfer
```

For the ADS NetID and the port, the values from the system manager are to be used.

```
(*
GET_ATTRIBUTE_SINGLE via ADSRead Data Transfer

IDXGRP: Index GroupNumber = Object Class
IDXOFFS: Index OffsetNumber = (Object Instance *. 0x100) + Attribute Id
LEN: Read Data Lengths in Bytes
DESTADDR: Address of DataBuffer to read with the Get-Attribute Single Service
*)
```

```
fbADSRead(
NETID:= ADSNetId,
PORT:= ADSPort,
IDXGRP:= IGrp_ADSRead,
IDXOFFS:= IOff_ADSRead,
LEN:= ADSReadLen,
DESTADDR:= ADR(GetAttributeData[0]),
READ:= ADSReadCommand,
TMOUT:= T#5s,
BUSY=> ADSReadBusy,
ERR=> ADSReadErr,
ERRID=> ADSReadErrID);
(*
```

```
COMMON SERVICE via ADSReadWrite Data Transfer

IDXGRP: Index GroupNumber = Object Class
IDXOFFS: Index OffsetNumber = (Object Instance *. 0x100) + Service Id
WRITELEN: Write Data Lengths in Bytes
READLEN: Read Data Lengths in Bytes
SRCADDR: Address of DataBuffer to write
DESTADDR: Address of DataBuffer to read
*)
```

```
fbADSReadWrite(
NETID:= ADSNetId,
PORT:= ADSPort,
IDXGRP:= Grp_ADSReadWrite,
IDXOFFS:= IOff_ADSReadWrite,
WRITELEN:= ADSReadWriteWriteLen,
READLEN:= ADSReadWriteReadLen,
SRCADDR:= ADR(CommonServiceWriteData[0]),
DESTADDR:= ADR(CommonServiceReadData[0]),
WRTRD:= ADSReadWriteCommand,
TMOUT:= T#5s,
BUSY=> ADSReadWriteBusy,
ERR=> ADSReadWriteErr,
ERRID=> ADSReadWriteErrID);
```

and

```
(*
SET_ATTRIBUTE_SINGLE via ADSWrite Data Transfer
IDXGRP: Index GroupNumber = Object Class
IDXOFFS: Index OffsetNumber = (Object Instance *. 0x100) + Attribute Id
LEN: Write Data Lengths in Bytes
SRCADDR: Address of DataBuffer to write with the Set-Attribute Single Service
*)
```

```
fbADSWrite(
NETID:= ADSNetId,
PORT:= ADSPort,
IDXGRP:= IGrp_ADSWrite,
```

```
IDXOFFS:= IOff_ADSWrite,
LEN:= ADSWriteLen,
SRCADDR:= ADR(SetAttributeWriteData[0]),
WRITE:= ADSWriteCommand,
TMOUT:= T#5s,
BUSY=> ADSWriteBusy,
ERR=> ADSWriteErr,
ERRID=> ADSWriteErrID);
```

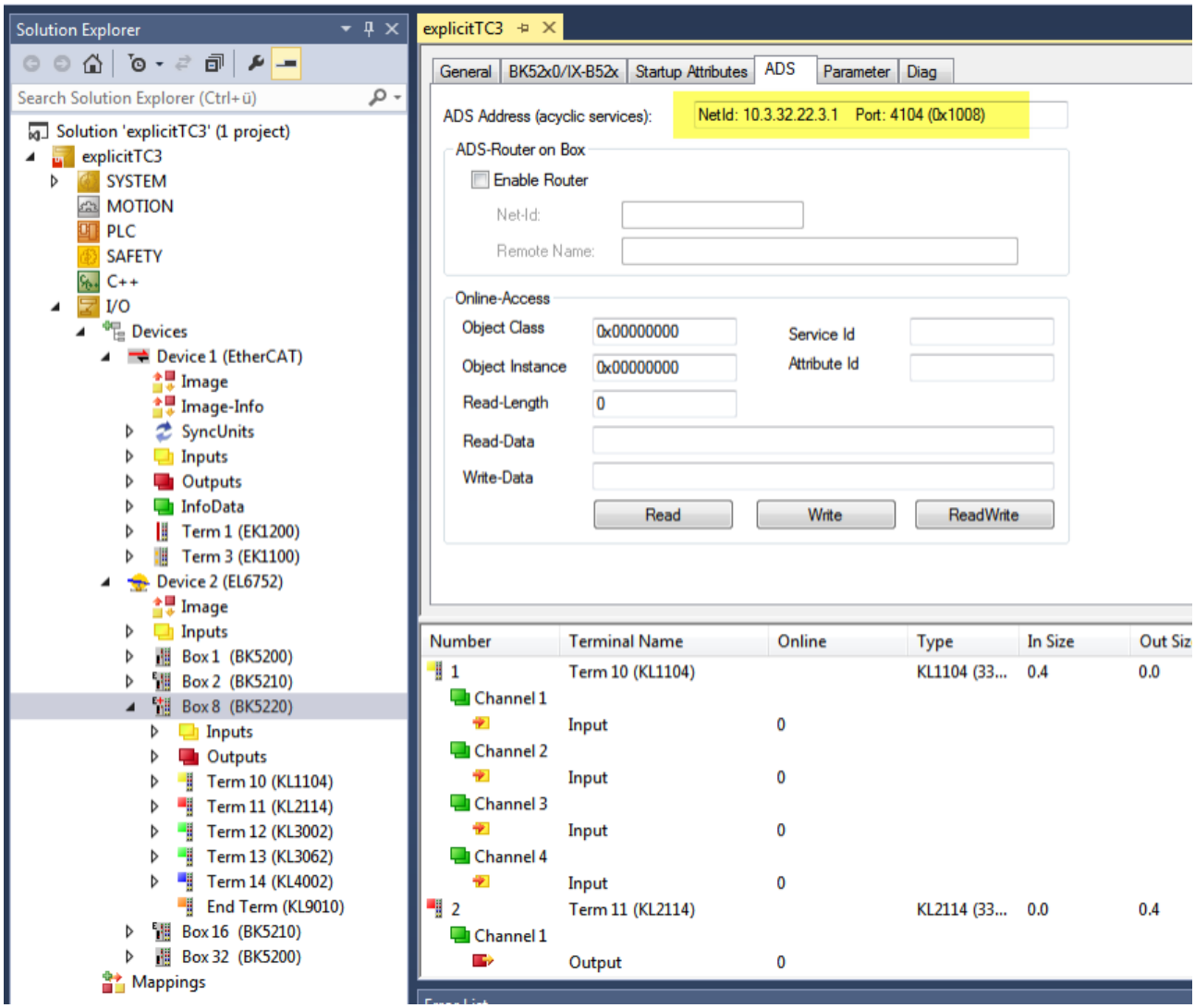


Fig. 25: Using ADS NetID and Port from System Manager

6 Parameterization and commissioning

6.1 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 2 levels via hexadecimal numbering: (main)index, followed by subindex. The value ranges are

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

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

The relevant ranges for EtherCAT fieldbus users are:

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

Other important ranges are:

- 0x4000: 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:

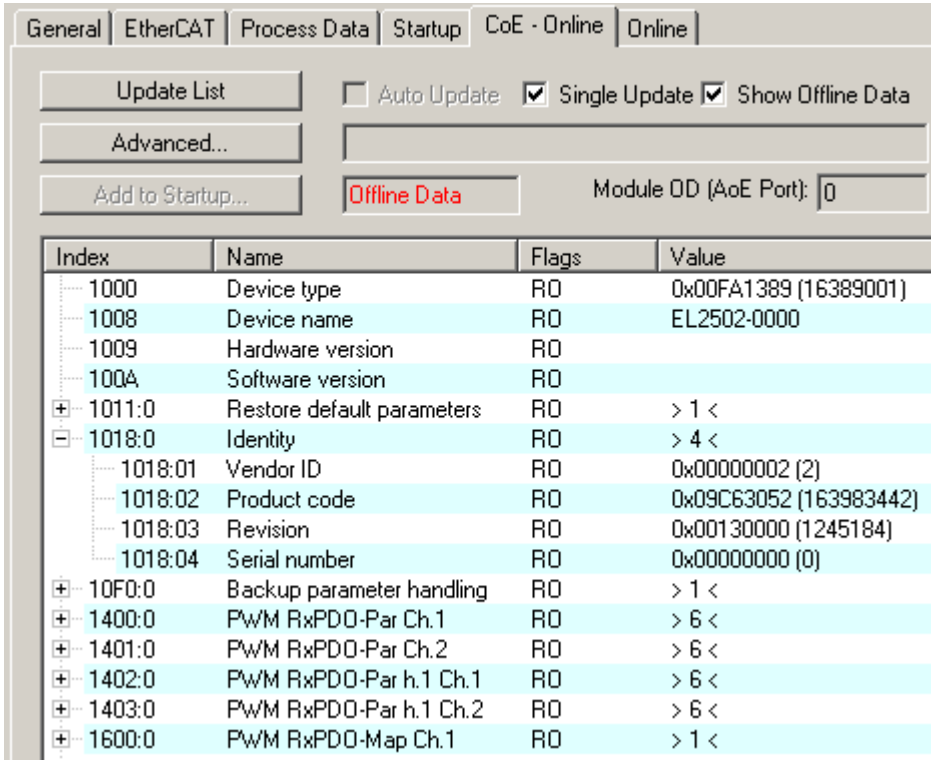


Fig. 26: "CoE Online " tab

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

Data management and function "NoCoeStorage"

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

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

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.

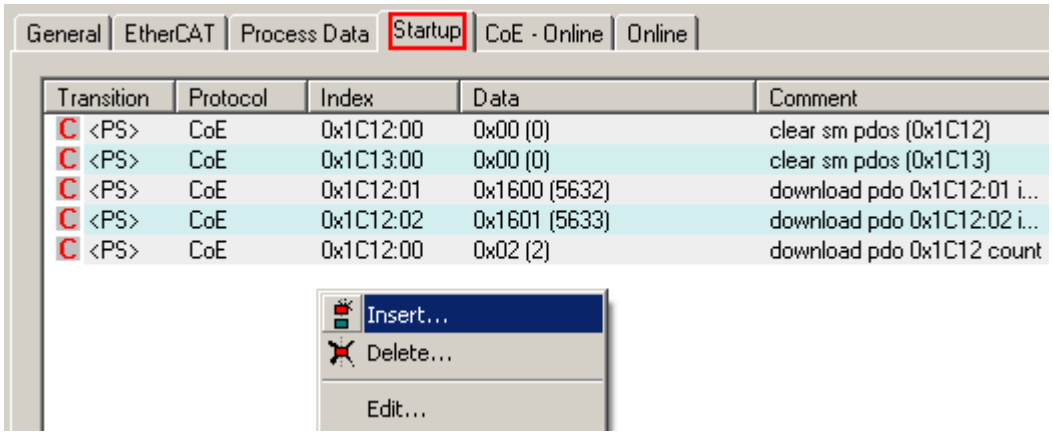


Fig. 27: Startup list in the TwinCAT System Manager

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.

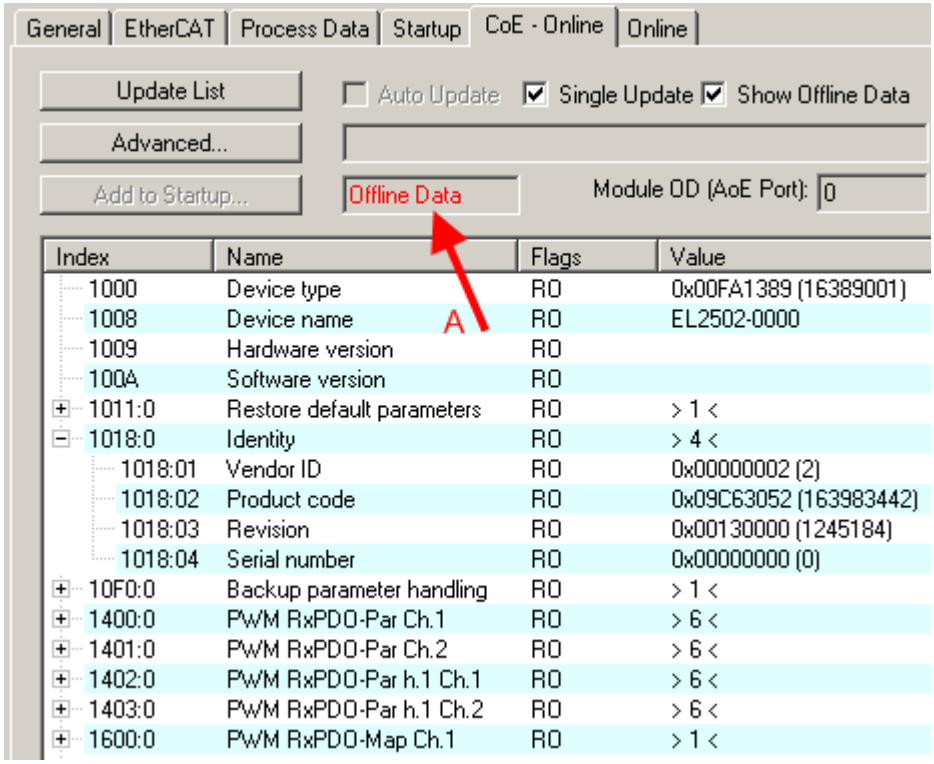


Fig. 28: Offline list

- 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.

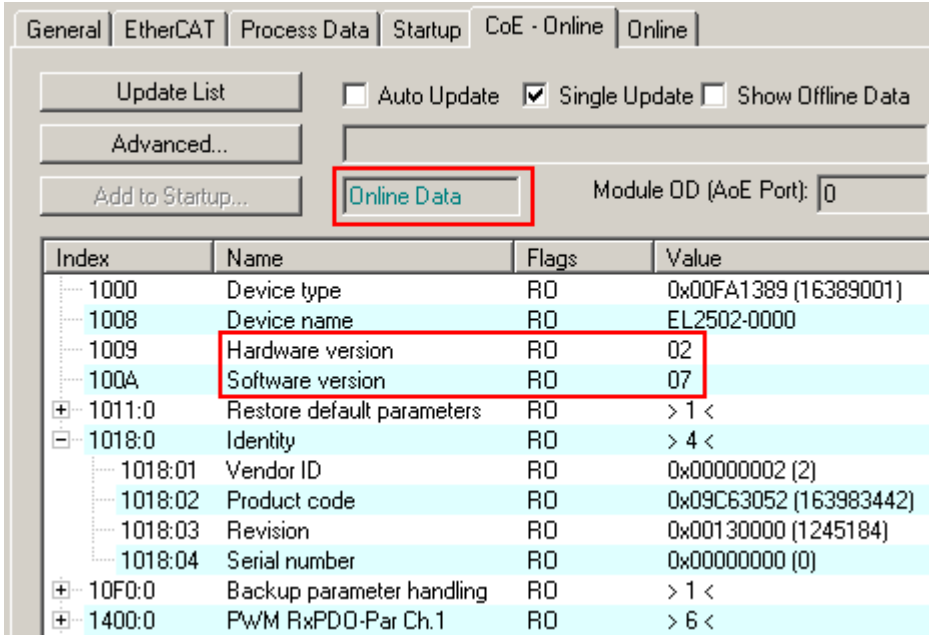


Fig. 29: Online list

Channel-based order

The CoE list is available in EtherCAT devices that usually feature several functionally equivalent channels. For example, a 4-channel analog 0...10 V input terminal also has 4 logical channels and therefore 4 identical sets of parameter data for the channels. In order to avoid having to list each channel in the documentation, the placeholder "n" tends to be used for the individual channel numbers.

In the CoE system 16 indices, each with 255 subindices, are generally sufficient for representing all channel parameters. The channel-based order is therefore arranged in $16_{dec}/10_{hex}$ 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.

6.2 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 2 watchdogs:

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

SM watchdog (SyncManager Watchdog)

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

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

PDI watchdog (Process Data Watchdog)

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

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

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

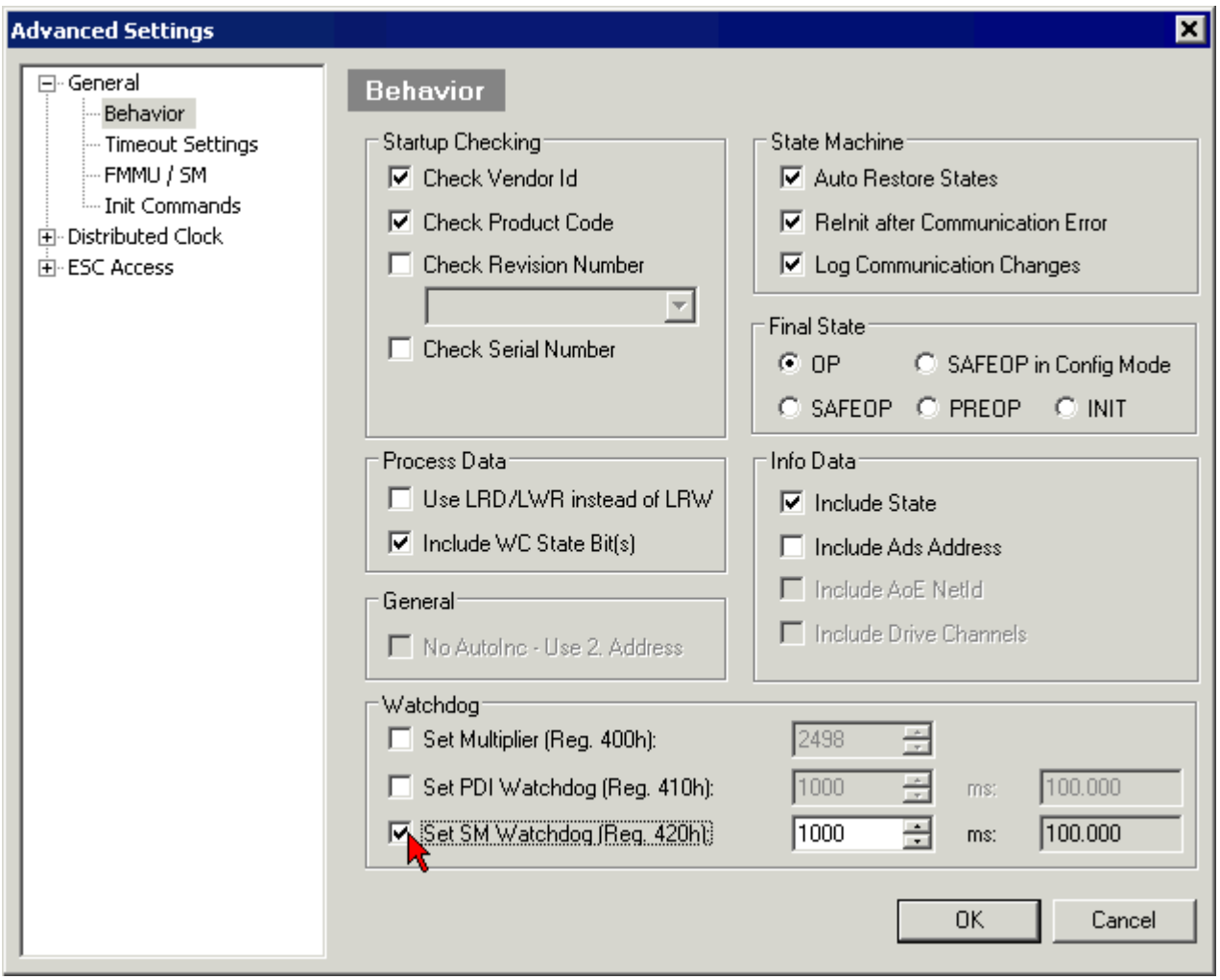


Fig. 30: EtherCAT tab -> Advanced Settings -> Behavior -> Watchdog

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.

6.3 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.

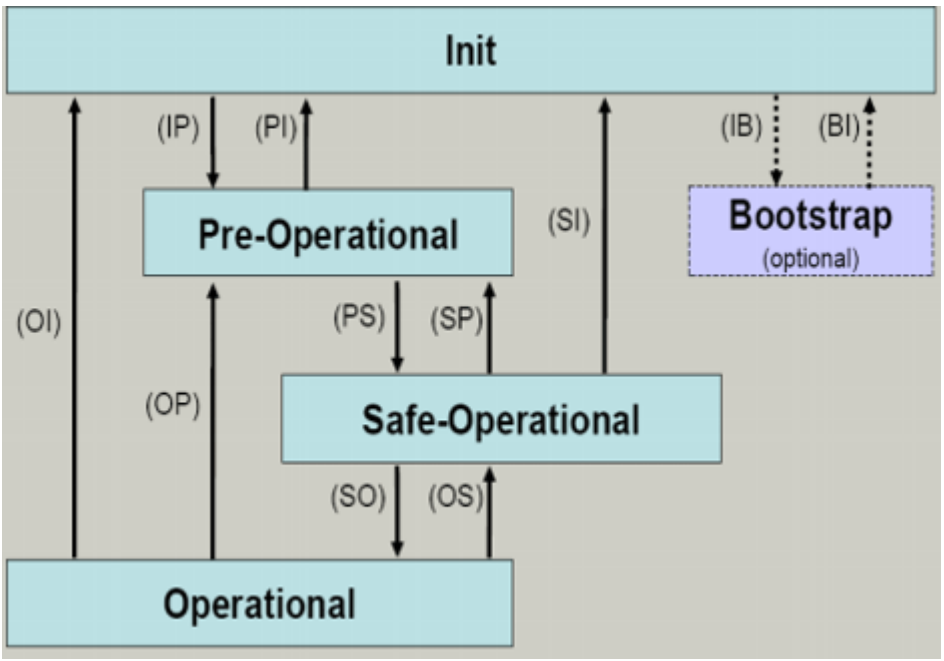


Fig. 31: States of the EtherCAT State Machine

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 [▶ 38] 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.

6.4 TwinCAT System Manager

The TwinCAT System Manager tool is used for the configuration of the EL6752 DeviceNet master/slave terminal. The System Manager provides a representation of the number of programs of the TwinCat PLC systems, the configuration of the axis control and of the connected I/O channels as a structure, and organizes the mapping of the data traffic.

Fig. 32: TwinCAT System Manager logo

For applications without TwinCAT PLC or NC, the TwinCAT System Manager Tool configures the programming interfaces for a wide range of application programs:

- ActiveX control (ADS-OCX) for e.g. Visual Basic, Visual C++, Delphi, etc.
- DLL interface (ADS-DLL) for e.g. Visual C++ projects
- Script interface (ADS script DLL) for e.g. VBScript, JScript, etc.

System Manager – Features

- Bit-wise association of server process images and I/O channels
- Standard data formats such as arrays and structures
- User defined data formats
- Continuous variable linking
- Drag and Drop
- Import and export at all levels

Configuration by means of the TwinCAT System Manager

The procedure and the configuration facilities in the System Manager are described below.

EL6752 DeviceNet master terminal [▶ 43]
EL6752-0010 - DeviceNet slave terminal [▶ 46]

EL6752 DeviceNet master terminal

Append device

The terminal can be appended to the I/O configuration either using the "Device search" routine in the TwinCAT System Manager or by manually selecting the "DeviceNet Master EL6752, EtherCAT" from the possible DeviceNet devices (Fig. *Appending the device "DeviceNet slave EL6752, EtherCAT"*). A right-click brings up the following context menu for selection:

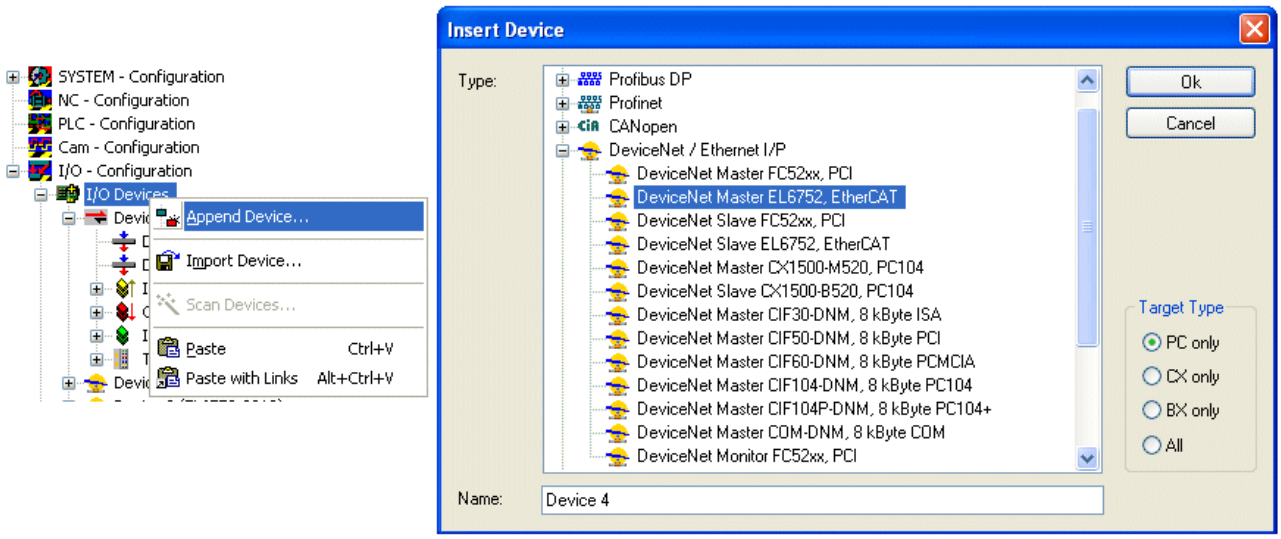


Fig. 33: Appending the device "DeviceNet master EL6752, EtherCAT"

"EL6752" tab

Click on the "Device EL6752" in the TwinCAT tree and then on the EL6752 tab:

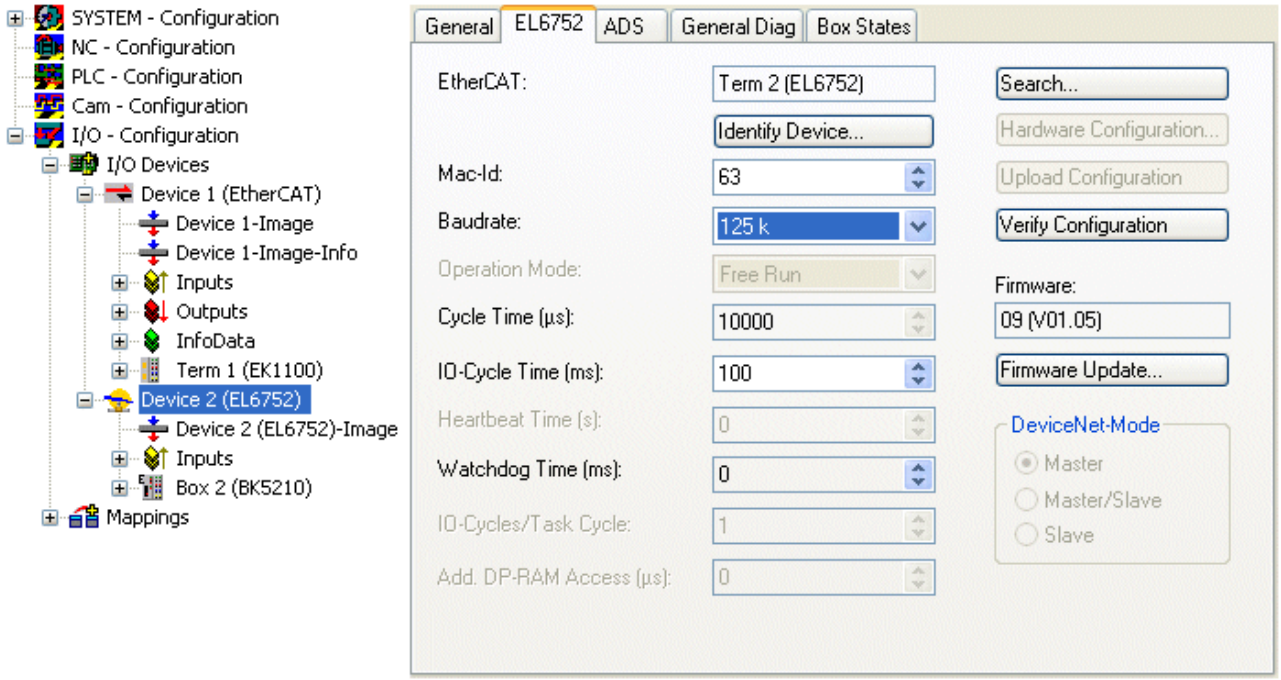


Fig. 34: "EL6752" tab

EtherCAT

Terminal ID in the terminal network.

MAC-ID

Each DeviceNet device - master included - requires a unique station number referred to as MAC ID (Medium Access Identifier) - value range: 0...63.

Baud rate

Baud rate setting: 125 kbaud, 250 kbaud or 500 kbaud

Cycle time

Displays the cycle time of the corresponding highest priority task. The display is updated when the mapping is generated.

IO-Cycle Time

Setting of the cycle time for the I/O connections. This value is the standard value for newly inserted boxes.

Watchdog time

Time until triggering of the watchdog

Search...

This function searches for all existing channels of the EL6752 and the desired one can be selected.

Check configuration

In preparation.

Firmware

Shows the current firmware version of the EL6752.

Firmware Update...

Updates the EL6752 firmware. Attention: The TwinCAT System must be stopped for this function.

"ADS" tab

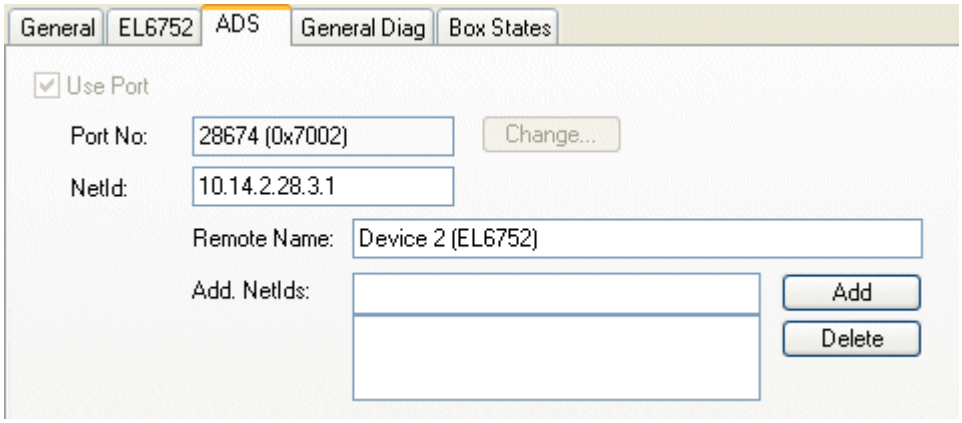


Fig. 35: "ADS" tab

The EL6752 is an ADS device with its own net ID, which can be changed here. All ADS services (diagnostics, acyclical communication) associated with the EL6752 device must address the card via this NetID.

"Box States" tab

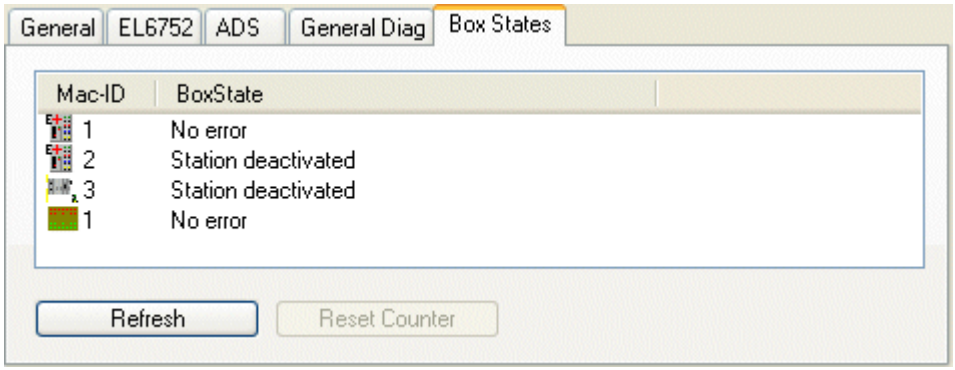


Fig. 36: "Box states" tab

Displays an overview of all current box statuses.

EL6752-0010 - DeviceNet slave terminal

In the system configuration tree structure right-click on I/O Devices and “Append device” to open the selection list of supported fieldbus cards.
Select EL6752-0010 CANopenSlave. TwinCAT searches for the terminal and displays the memory addresses and slots it finds. Select the required address and confirm.

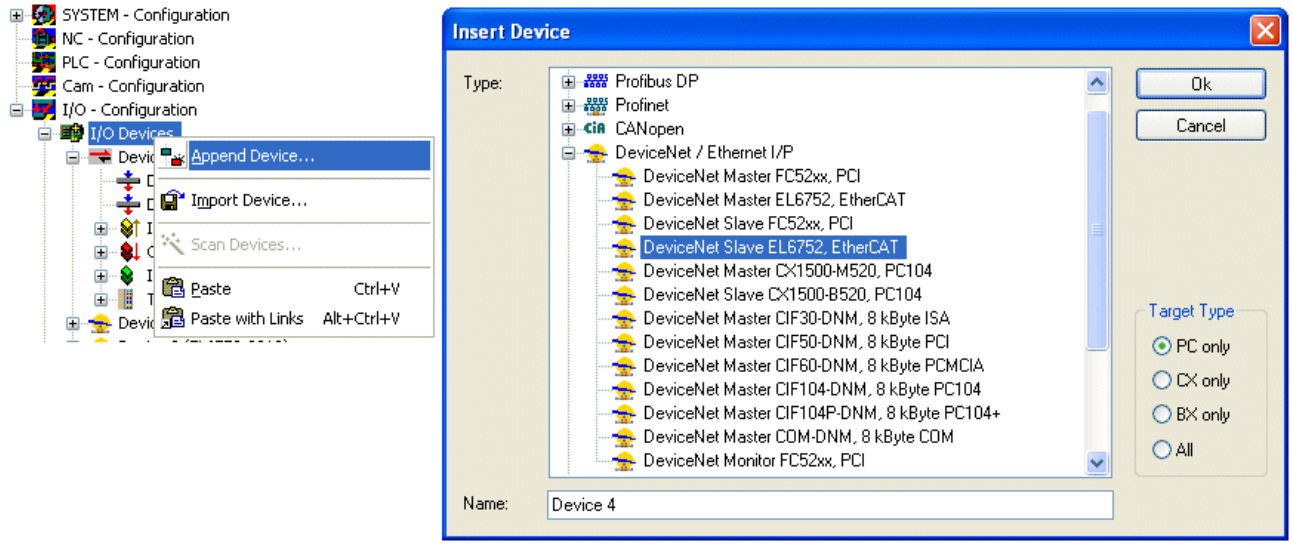


Fig. 37: Appending the device "DeviceNet slave EL6752, EtherCAT"

Right-click on "Device (EL6752-0010)" to insert the box for the EL6752-0010:

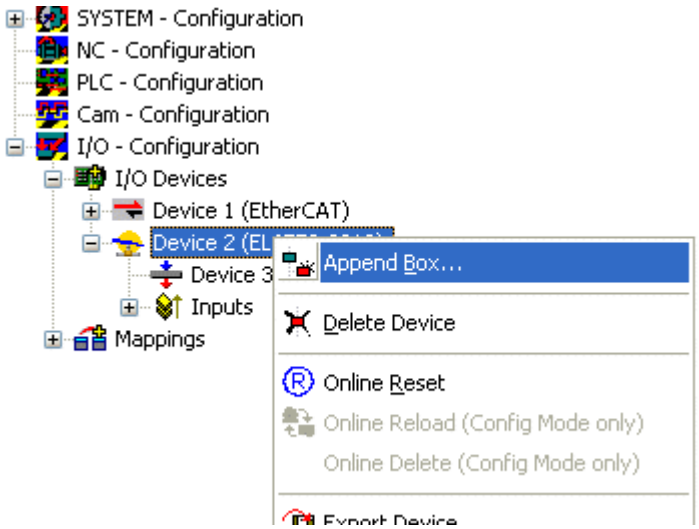


Fig. 38: Appending the box "DeviceNet slave EL6752, EtherCAT"

Selecting the I/O device for the EL6752-0010 in the tree structure opens a dialog with various configuration options:

“EL6752-0010” tab

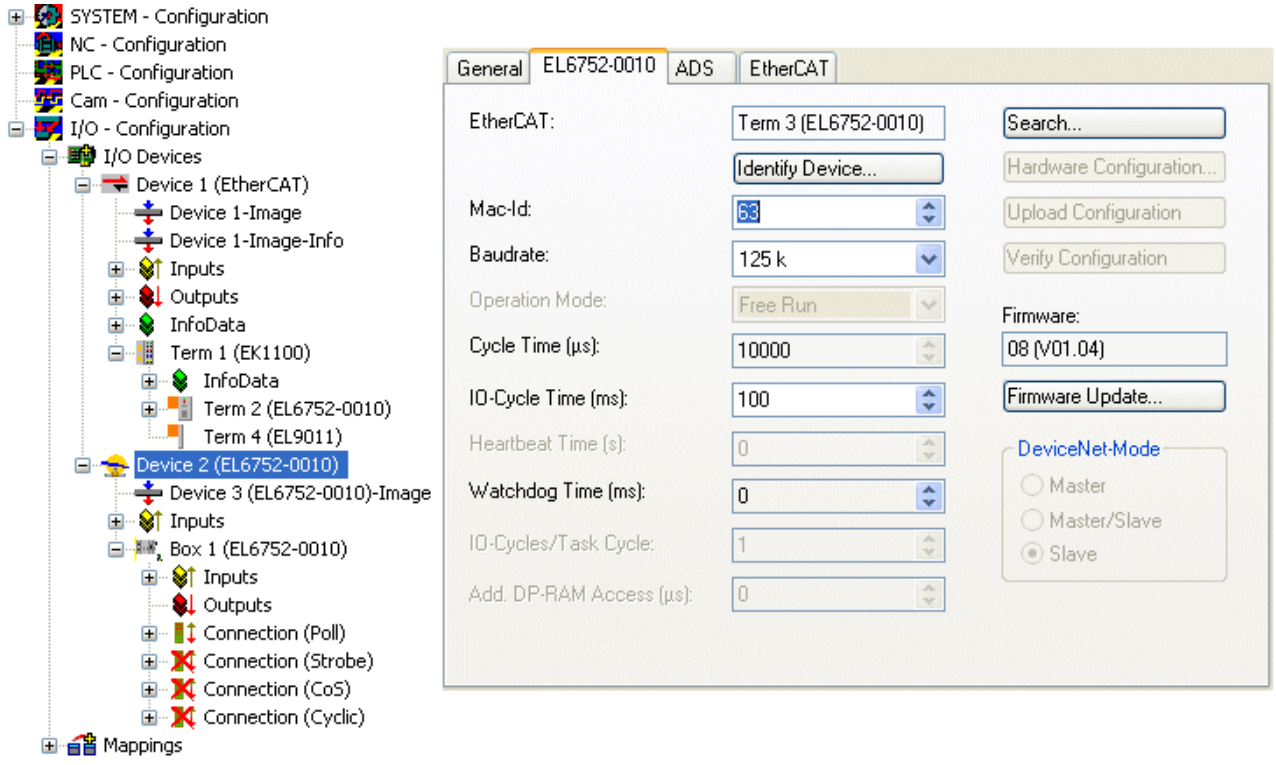


Fig. 39: “EL6752-0010” tab

EtherCAT

Terminal ID in the terminal network.

MAC-ID

Each DeviceNet device requires a unique station number referred to as MAC ID (Medium Access Identifier) - value range: 0...63.

Baud rate

The baud rate is set here.

Cycle time

Displays the cycle time of the corresponding highest priority task. The display is updated when the mapping is generated. The network variables are updated with the cycle of this task.

Watchdog time

Time until the watchdog is triggered

Search...

Searches for all available EL6752-0010 channels, from which the required channel can be selected. In the case of an FC5102 both channels A and B appear. These behave in logical terms like two FC5101 cards.

Firmware

Displays the current EL6752-0010 firmware version.

Firmware Update...

Updates the EL6752-0010 firmware. Attention: The TwinCAT System must be stopped for this function.

"ADS" tab

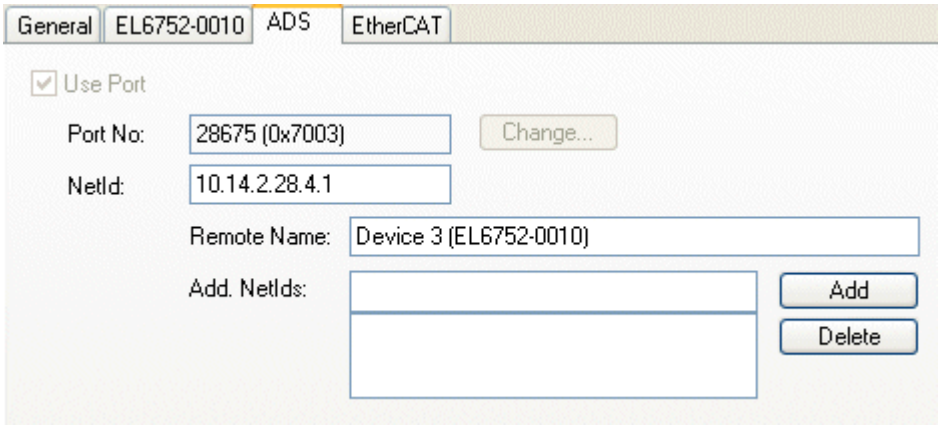


Fig. 40: "ADS" tab

The EL6752-0010 is an ADS device with its own net ID, which can be changed here. All ADS services (diagnostics, acyclic communication) associated with the EL6752-0010 device must address the card via this NetID. Additional ADS Net IDs can be entered for addressing subordinate ADS devices (e.g. an additional fieldbus card in the same PC) via the card.

"(Online) DPRAM" tab

Read access to the DPRAM of the card is provided for diagnostic purposes.

Box EL6752-0010 slave

A box "EL6752-0010 (DeviceNet slave)" is created automatically. Further parameters have to be set:

Box EL6752-0010 tab:

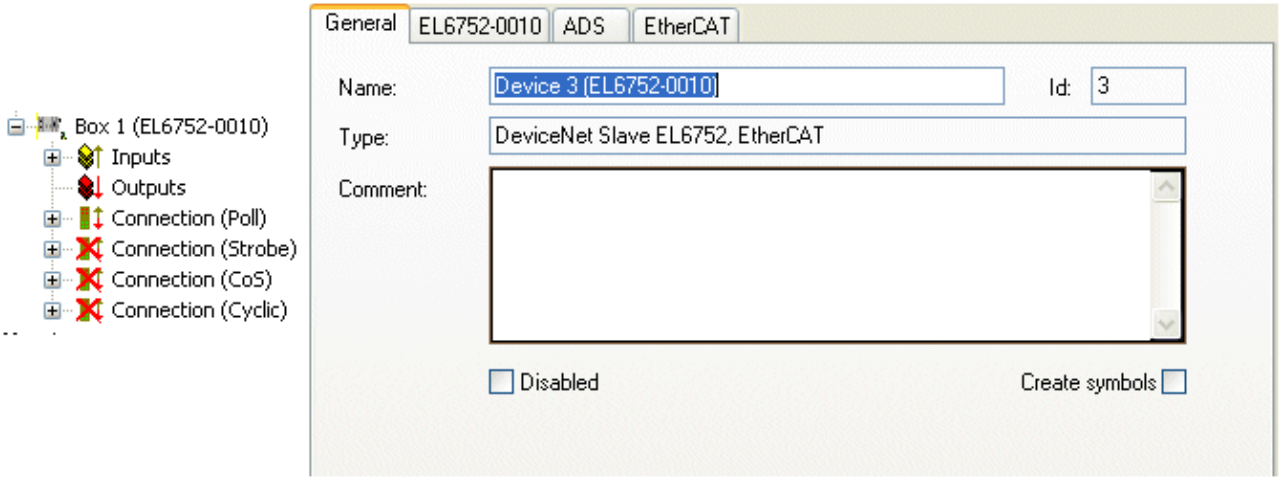


Fig. 41: "General" tab, Box EL6752-0010

DeviceNet IO modes

The EL6752-0010 supports the DeviceNet modes cyclic polling, change of state / cyclic and bit strobe. The IO modes can be selected according to the DeviceNet specification.

The DeviceNet IO mode cyclic polling is the default selection for the EL6752-0010:

IO mode	Input data length / bytes	Output data length / bytes
Polling	0 - 255	0 - 255
Change of State	0 - 255	0 - 255
Cyclic	0 - 255	0 - 255
Bit strobe	1 bit	0-8

Polling / Change of State (COS) / Cyclic

The cyclic polling mode is characterized by cyclic polling of the IO data by the master. The change of state mode is characterized by event-oriented sending of IO data. In cyclic mode the IO data are sent cyclically based on the communication parameters configured by the master. Since the communication settings are specified through the master no further settings are possible. Further information on the modes can be found in section DeviceNet Communication. The settings are identical for these modes.

The input and output data lengths are pre-initialized to 8 byte each:

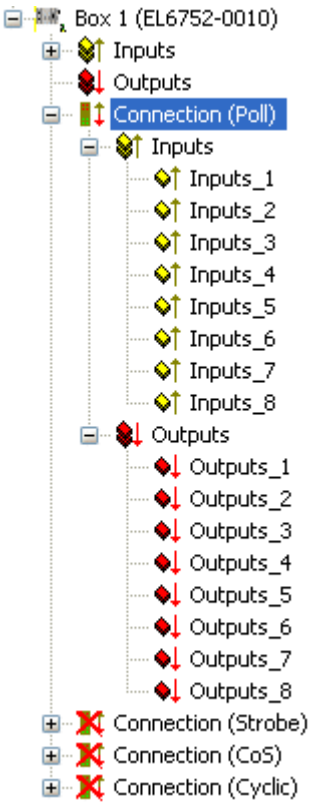


Fig. 42: Pre-initialized input and output data lengths in polling mode

According to needs and the application, further input or output data can be appended by right-clicking (Fig. Adding further variables). Any data type can be selected:

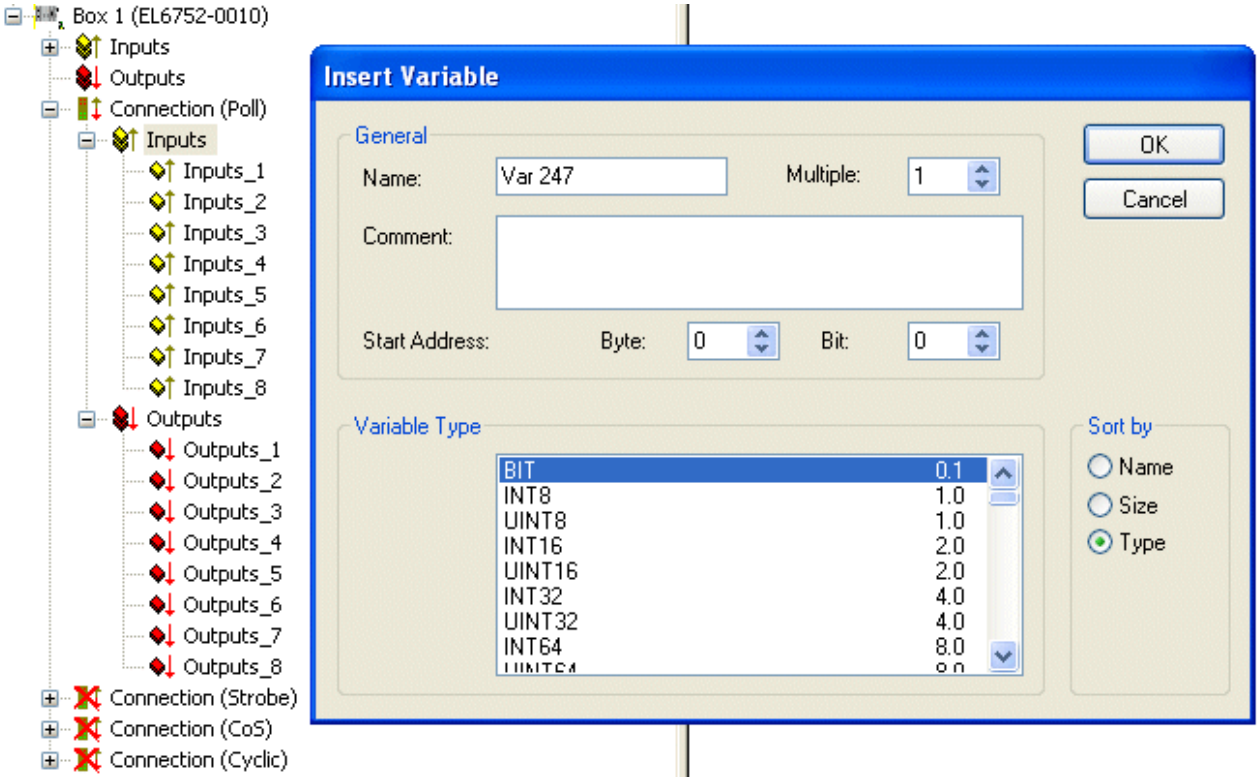


Fig. 43: Adding further variables

The data length is converted to a byte stream according to the DeviceNet specification and displayed in the tab for the corresponding connection:

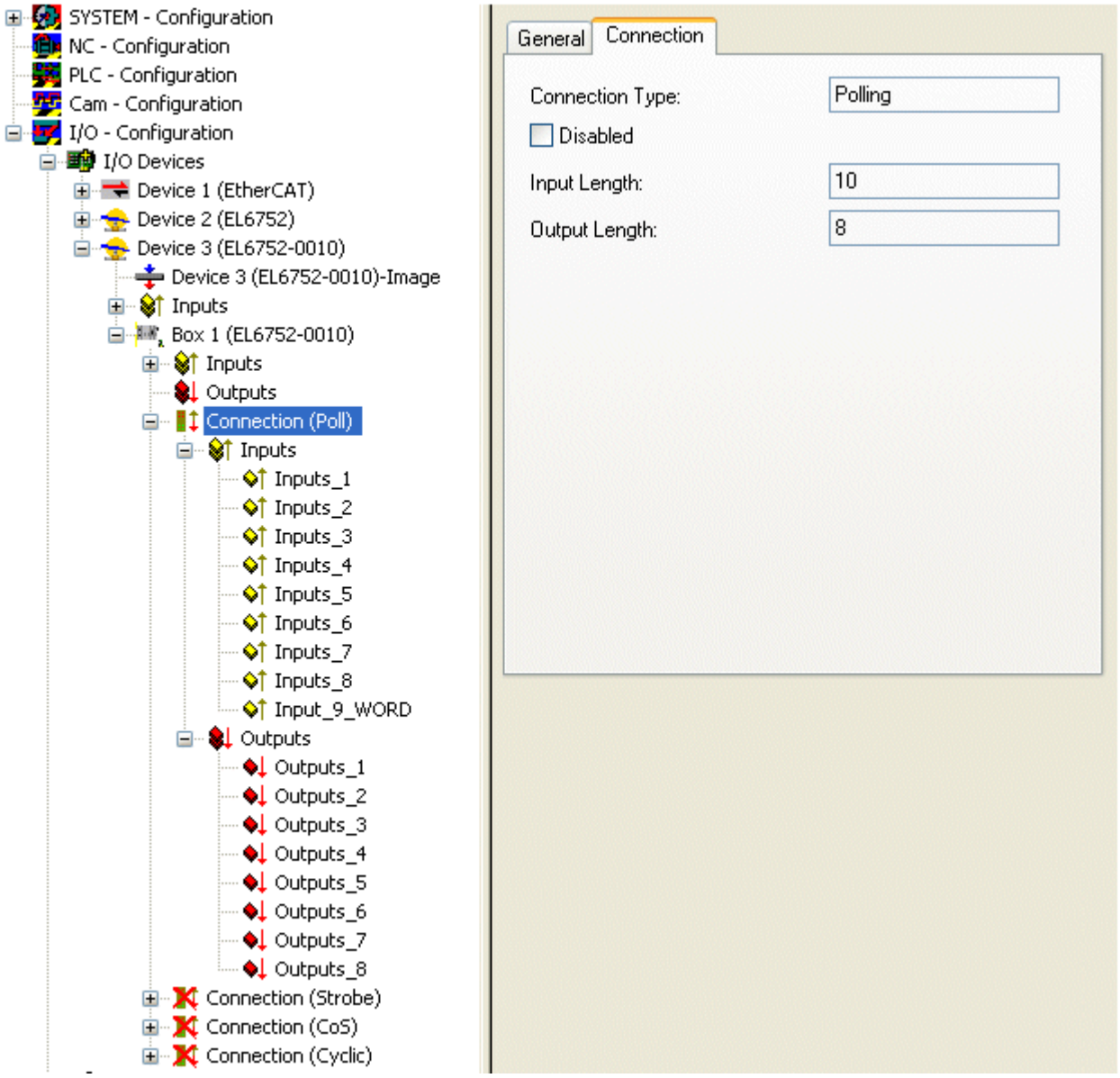


Fig. 44: "Connection" tab showing connection type "Polling" and input and output parameters

Maximum output data length

The maximum data length per data direction is 255 bytes.

The indicated input and output data lengths must be configured for the corresponding DeviceNet master.

Bit strobe

The IO mode bit strobe involves an 8-byte command from the master to the slaves. For each possible address/MAC ID (DeviceNet address space: 64) 1 bit of user data is allocated. The maximum length of the response message from the slave is 8 bytes. It is sent to the master immediately when the bit strobe command is received.

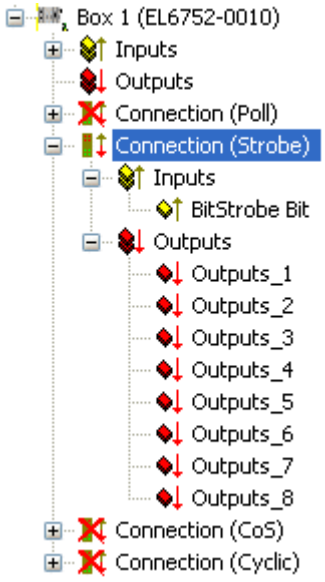


Fig. 45: Display of output parameters in the TwinCAT tree for connection type "Bit strobe"

Maximum output data length

The maximum output data length is 8 bytes. The input data length is fixed.

Since the communication settings are specified through the master no further settings are possible.

6.5 Beckhoff DeviceNet Bus Coupler

The Bus Coupler BK52xx and the IPxxx-B520 Fieldbus Box are used in the **DeviceNet** bus. The specific properties which distinguish them from other Bus Couplers and/or fieldbus box modules are then described below.

Types	Description
BK5210	Economy Bus Coupler
BK5220	Economy + Bus Coupler
LC5200	Low-Cost Bus Coupler
BK5250	Compact Bus Coupler
BC5250	Compact Bus Terminal Controller with 48 kbyte program memory
BX5200	BX Bus Terminal Controller with 256 kbyte program memory
IPxxxx-B520	Fieldbus compact box: DeviceNet input/output module in protection class IP67

The following tabs are used for parameterization:

- "BK52x0" tab [▶ 53]
- "Startup Attributes" tab [▶ 55]
- "ADS" tab [▶ 56]
- "Parameters" tab [▶ 57]
- "Diag" tab [▶ 57]

"BK52x0" tab

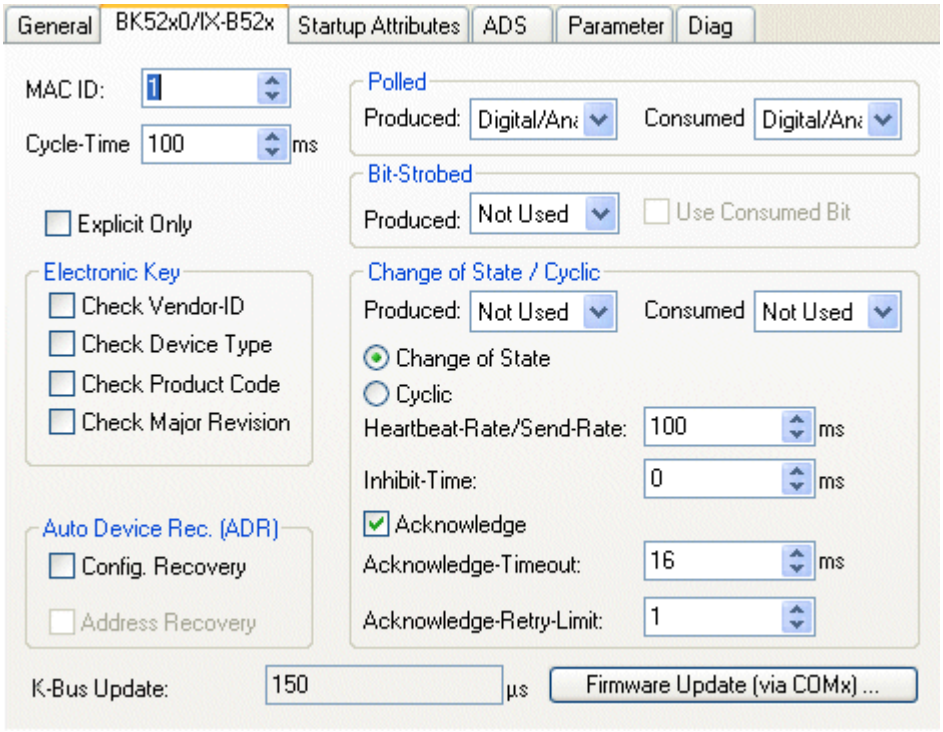


Fig. 46: "BK52x0" tab

MAC-ID

Sets the MAC ID, i.e. the device address of the DeviceNet device (between 0 and 63). This value must comply with the value set at the Bus Coupler and/or at the compact box.

Cycle time

Sets the cycle time of the IO connection polling and bit strobe. The value is used as “Expected Packet Rate” attribute of the “Connection Object” according to the DeviceNet specification.

Electronic Key

Serves to check the devices within the network at the system StartUp. The electronic key is read from the devices at every system StartUp and compared with the saved configuration.

Polled

Produced/Consumed
Activation of the “Polling” mode, cyclic writing and reading of IO data. Setting of the data content of the data transmitted via the polled IO connections. You can choose from digital data, analog data or both. The selection depends upon the BK52xx terminal arrangement.

Bit-Strobed

Produced/Consumed
Activation of the “Bit Strobe” operating mode. A broadcast message requests all nodes to send their bit strobe message (up to 7 bytes input or status data). Setting of the data content of the data transmitted via the bit-strobed IO connections. You can choose between digital data or diagnostic data.

Change of State / Cyclic

Produced/Consumed
Setting of the data content of the data transmitted via the change of state/cyclical IO connections. You can choose from digital data, analog data or both. The selection depends upon the BK52xx terminal arrangement.

Change of State / Cyclic
Selection of the required operating mode.

Heartbeat-Rate / Send-Rate
In the “Change of State” mode the heartbeat rate gives the cycle time of the cyclical send of the lower-level (i.e. in addition to the event driven) IO data. In “Cyclic” mode the send rate specifies the cycle time with which the IO data are sent.

Inhibit-Time
Delay time in “Change of State” mode; after a change of state IO data are sent after the specified time at the earliest.

Acknowledge Timeout
Time before the retransmission in the event of faulty acknowledgement of a change of state / cyclical message.

Acknowldege Retry Limit
Maximum number of retransmissions until IO connection goes into error mode.

K-Bus update

Calculates the expected time required for a full update of the terminal bus (depends on the connected terminals).

Auto Device Replacement (ADR)

Not supported.

“Startup Attributes” tab

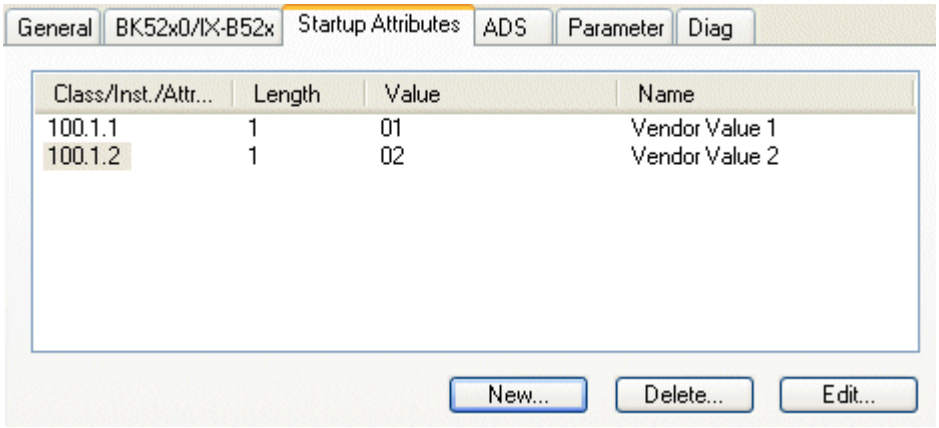


Fig. 47: “Startup Attributes” tab

The startup attributes are sent to the slave before the cyclic data exchange. The messages are sent before the actual IO data traffic.

Use the “New” or “Edit” button for configuration:

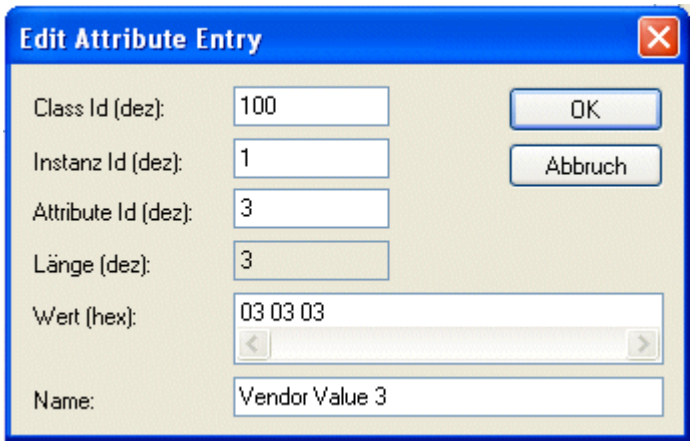


Fig. 48: Edit an attribute entry

The attributes are initialized via Class/Instance/Attributes. Note the “Value” specification in hexadecimal form.

“ADS” tab

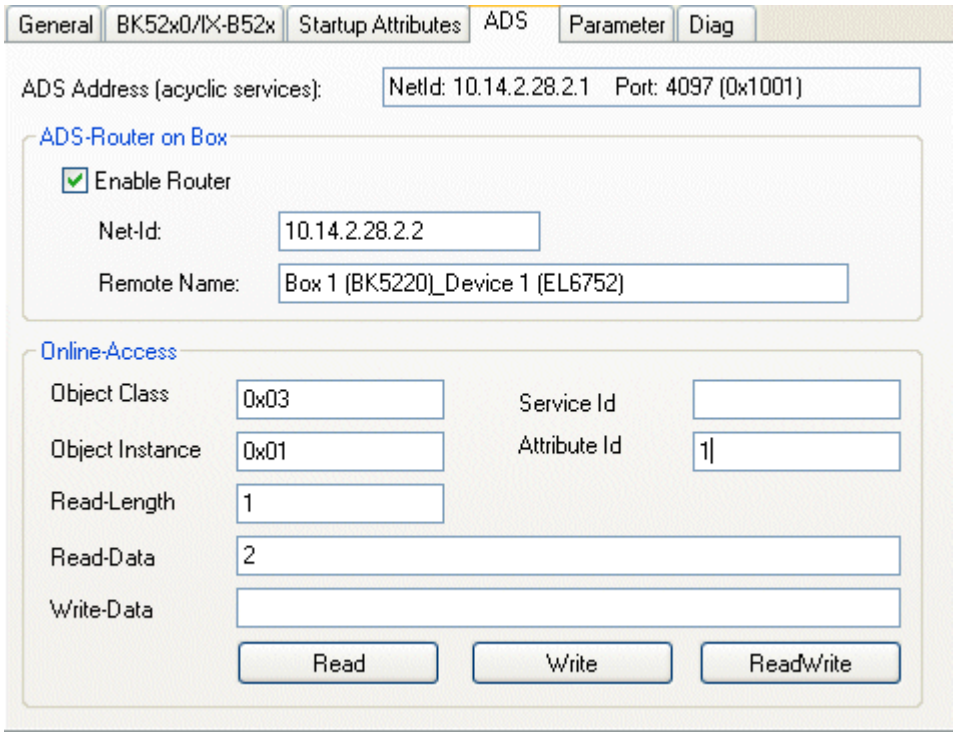


Fig. 49: “ADS” tab

The node (Bus Coupler) is assigned an ADS port to enable writing and reading of DeviceNet objects at runtime (e.g. from the PLC). It can be changed if required. A detailed description of explicit messages can be found in section “DeviceNet Communication” under “Explicit Messages”.

DeviceNet objects can be accessed via Online Access. To this end the DeviceNet-specific information such as Class/Instance/Attributes has to be entered.

Read
Reading of an object attribute via DeviceNet “Get_Attribute_Single” service. A service ID is not required.

Write
Writing of an object attribute via DeviceNet “Set_Attribute_Single” service. A service ID is not required.

Read/Write
Executing any DeviceNet service. Specification of the service ID is required.

“Parameter” tab

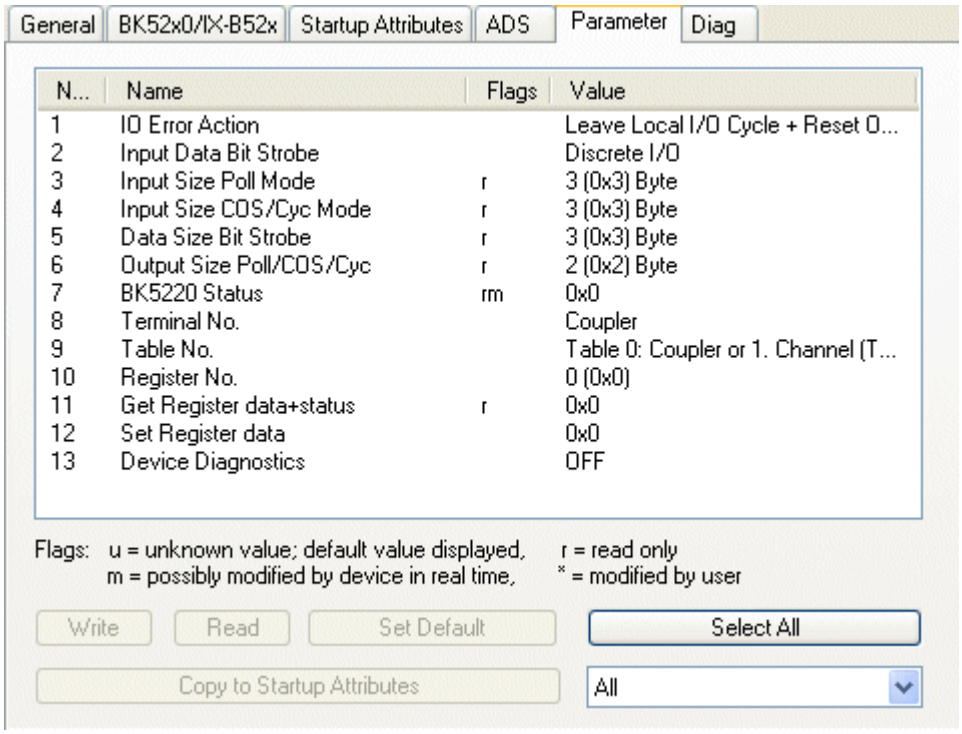


Fig. 50: “Parameter” tab

The parameters read from the EDS file are shown under the “Parameters” tab. Parameters can be read, written and entered in the list of the startup parameters.

“Diag” tab

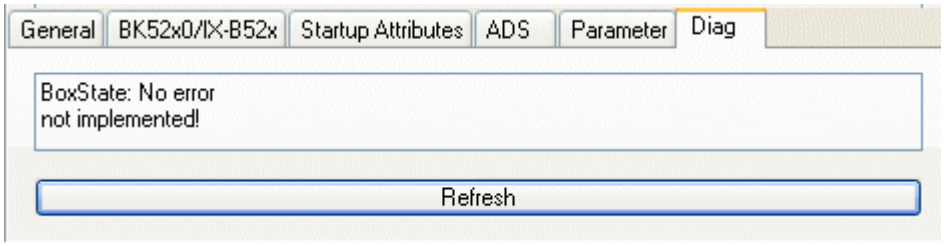


Fig. 51: "Diag" tab

The “Diag” tab indicates the state of the box. No further diagnostic options are available.

Also see about this

- Explicit messages [▶ 32]

6.6 General DeviceNet device

DeviceNet devices are integrated as general DeviceNet devices.

Fig. 52: Adding a DeviceNet device (I/O Devices-> Device n (EL6752)->right-click-> Append Box...)

6.6.1 Integrating a DeviceNet device with EDS file

If an EDS file is available for the DeviceNet to be integrated, it must be copied into the **..TwinCAT/IO/DeviceNet** directory.

Subsequently the device appears under the "Append Box" selection (see fig. *Adding a DeviceNet device (I/O Devices -> Device n (EL6752) -> right-click -> Append Box ...)* with the manufacturer ID:

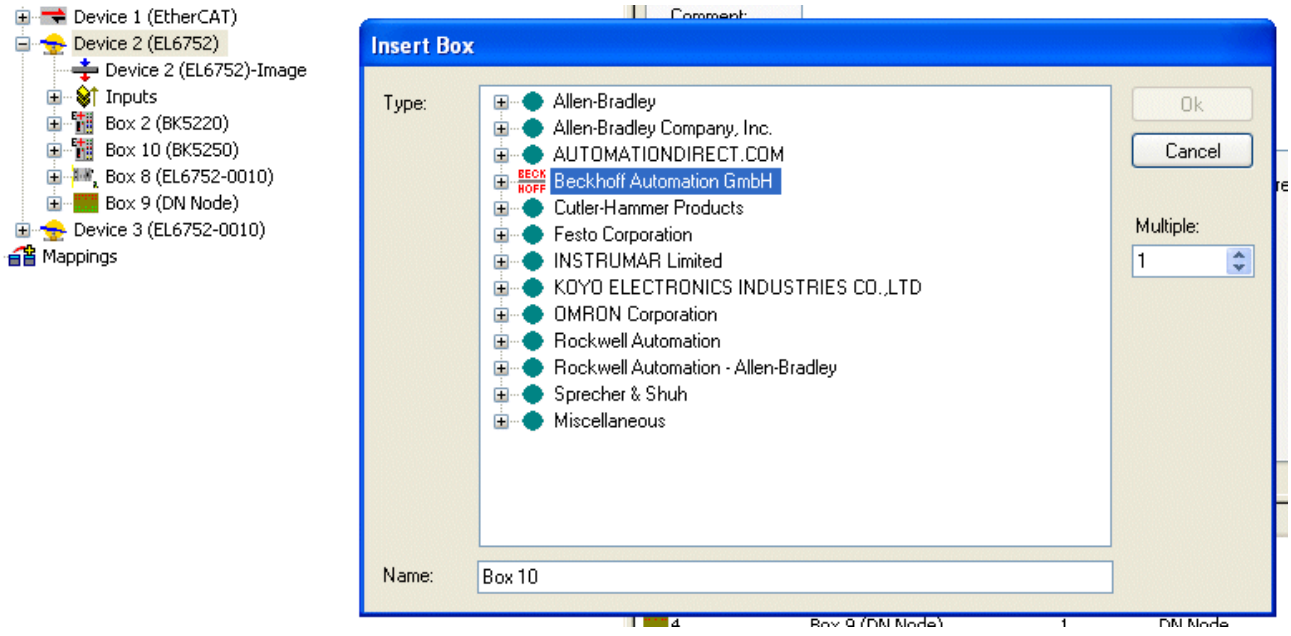


Fig. 53: Adding a box with the manufacturer ID

Alternatively a DeviceNet device with EDS file can be integrated via the "Miscellaneous" option:

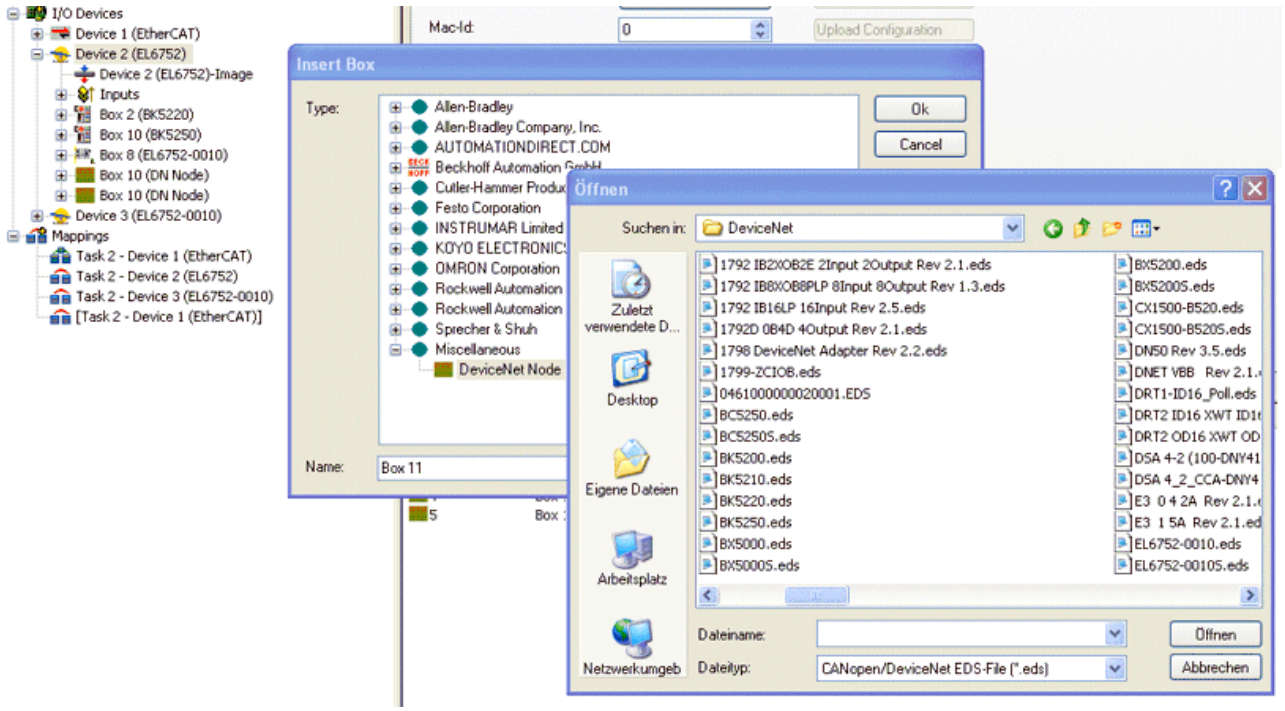


Fig. 54: Adding a box without EDS file

Depending on the information contained in the EDS file a DeviceNet node will appear with or without Parameters tab.

The IO mode and the corresponding data lengths are specified in the EDS file.

6.6.2 Integrating a DeviceNet device without EDS file

A DeviceNet device without EDS file can be integrated via the "Miscellaneous" option:

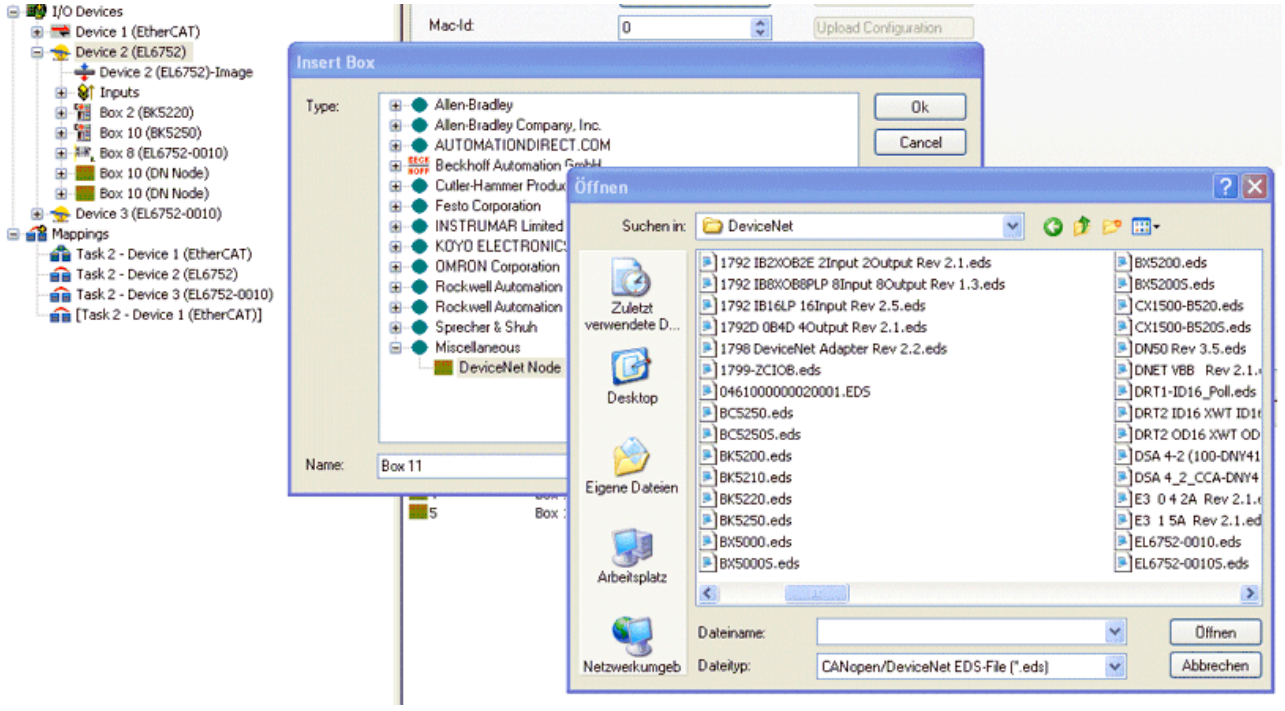


Fig. 55: Adding a box without EDS file (click "Cancel")

Terminate EDS file selection with "Cancel". A general DeviceNet device is created.

Selection of the IO mode and the general configuration must then be carried out manually.

DeviceNet IO modes

For DeviceNet devices the EL6752 supports the DeviceNet modes cyclic polling, change of state / cyclic and bit strobe. The IO modes can be selected according to the DeviceNet specification.

The DeviceNet IO mode cyclic polling is the default selection for the EL6752:

IO mode	Input data length / bytes	Output data length / bytes
Polling	0 - 255	0 - 255
Change of State	0 - 255	0 - 255
Cyclic	0 - 255	0 - 255
Bit strobe	1 bit	0-8
Total of all IO data	max. xxx bytes	max. xxx bytes

polling / change of state (COS) / cyclic

The cyclic polling mode is characterized by cyclic polling of the IO data by the master. The change of state mode is characterized by event-oriented sending of IO data. In cyclic mode the IO data are sent cyclically based on the communication parameters configured by the master. The settings are identical for these modes.

The input and output data lengths must be supplemented according to the device configuration:

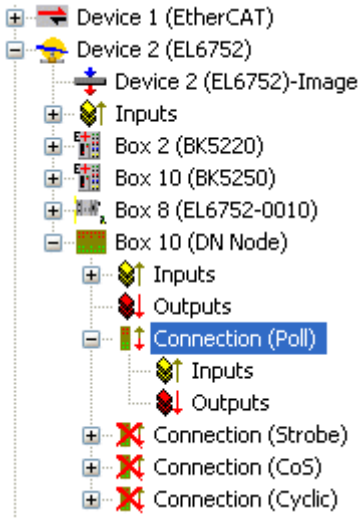


Fig. 56: Supplementing the input and output data

Input or output data must be appended depending on the device configuration. Any data type can be selected:

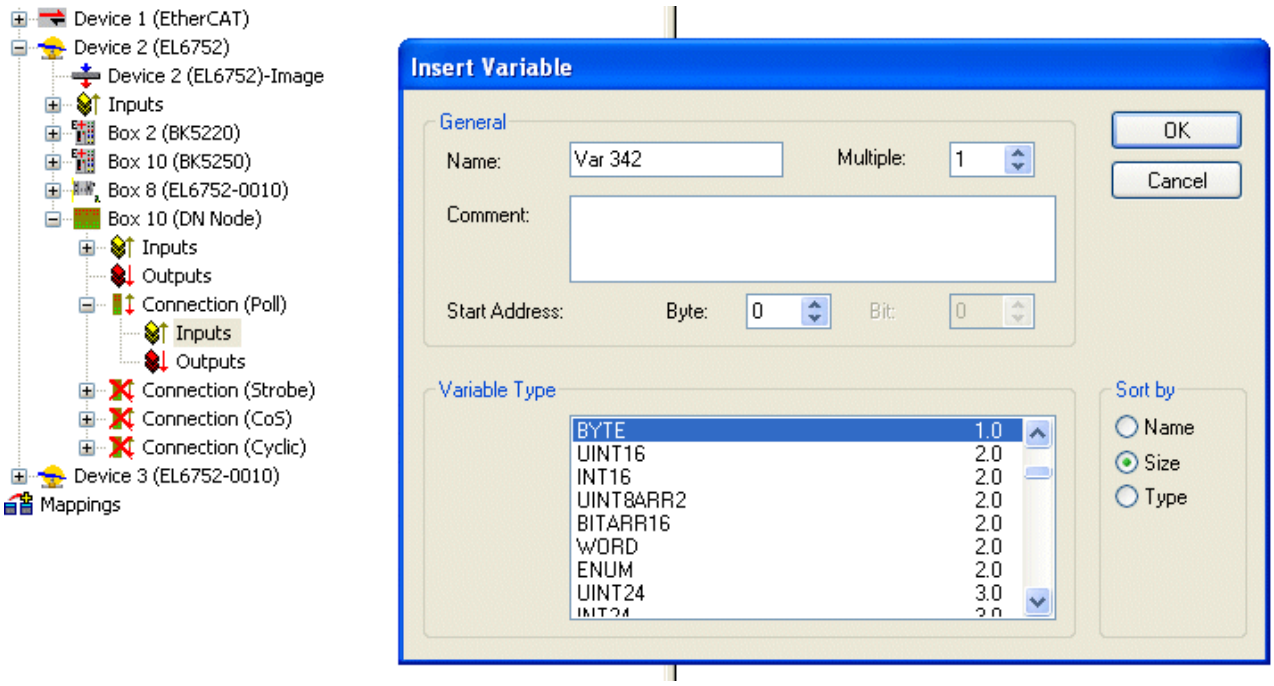


Fig. 57: Add Variables

The data length is converted to a byte stream according to the DeviceNet specification and displayed in the tab for the corresponding connection:

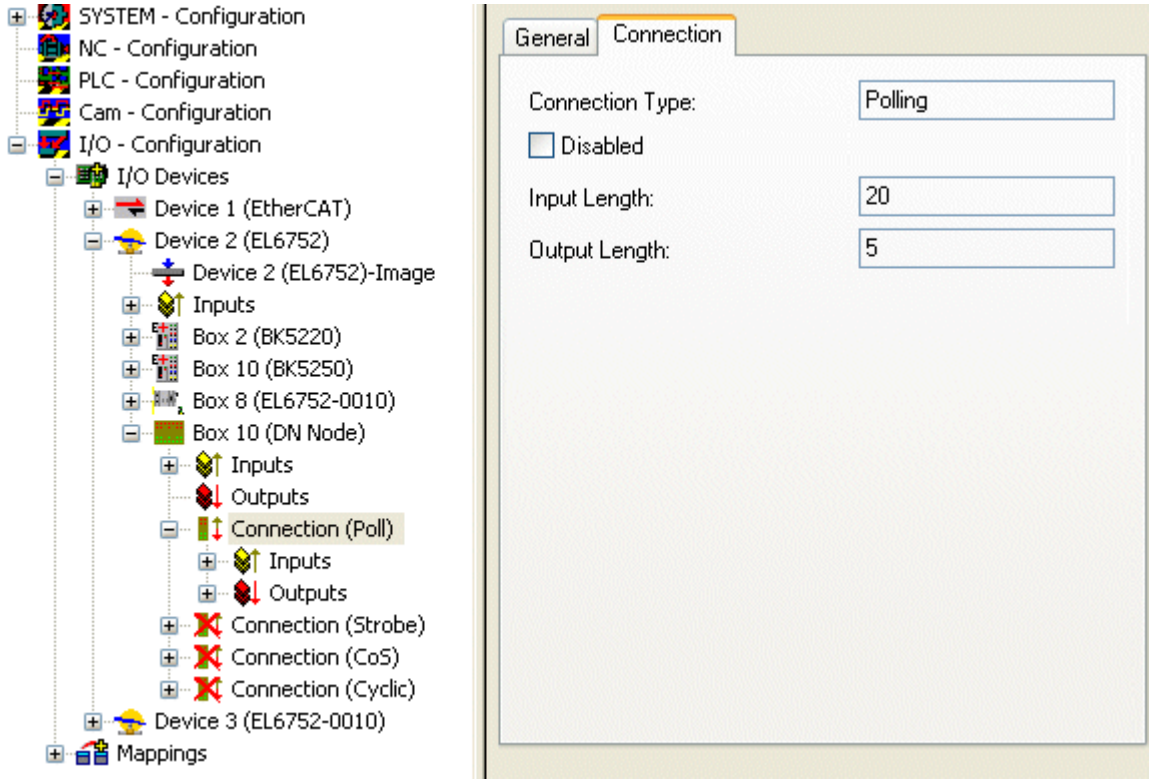


Fig. 58: "Connection" tab showing connection type "Polling" and input and output parameters

Maximum data length

The maximum data length per data direction is 255 bytes.

Bit strobe

The IO mode bit strobe involves an 8-byte command from the master to the slaves. For each possible address/MAC ID (DeviceNet address space: 64) 1 bit of user data is allocated. The maximum length of the response message from the slave is 8 bytes. It is sent to the master immediately when the bit strobe command is received.

After selection of the bit strobe mode the input data must be configured accordingly. Any data type can be selected (see polling/ COS / cyclic). The data length is converted to a byte stream according to the DeviceNet specification and displayed in the tab for the bit strobe connection:

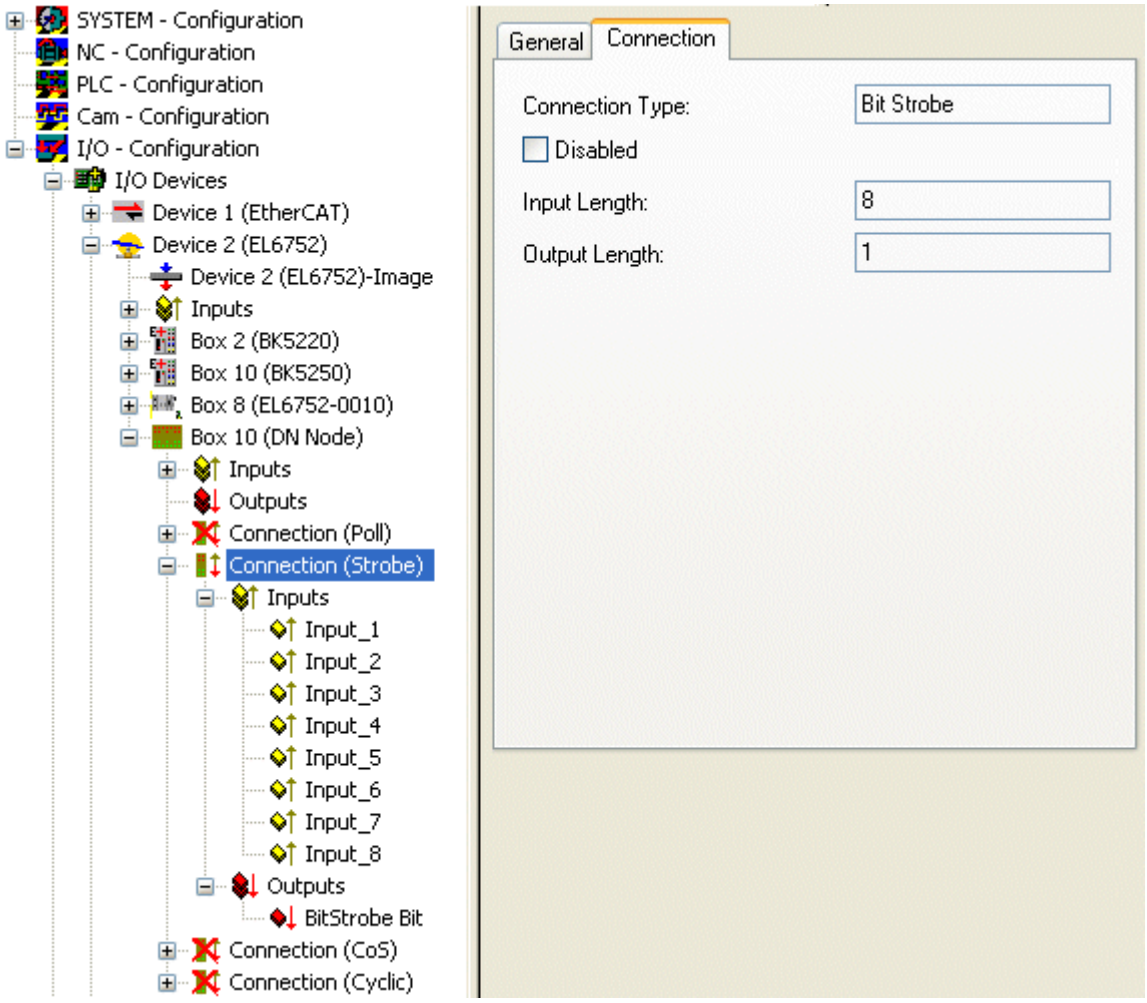


Fig. 59: "Connection" tab showing connection type "Bit Strobe" and input and output parameters

Maximum data length

The maximum input data length is 8 bytes. The output data length is fixed.

Since the communication settings are specified through the master no further settings are possible.

6.6.3 Parameterization of a DeviceNet device

The DeviceNet devices are parameterized with the following tabs:

- "DeviceNet Node" tab [▶ 63]
- "Startup Attributes" tab [▶ 64]
- "ADS" tab [▶ 65]
- "Parameter" tab [▶ 66]

- "Diag" tab [▶ 66]

"DeviceNet Node" tab

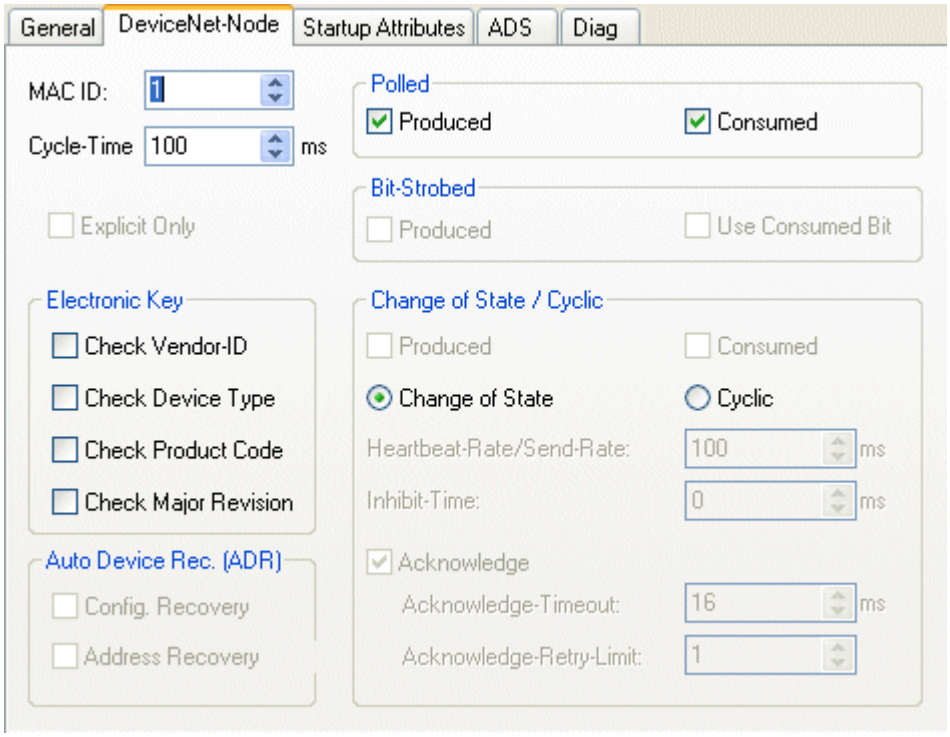


Fig. 60: "DeviceNet Node" tab

MAC-ID

Sets the MAC ID, i.e. the device address of the DeviceNet device (between 0 and 63). This value must comply with the value set at the Bus Coupler and/or at the compact box.

Cycle time

Sets the cycle time of the IO connection polling and bit strobe. The value is used as “Expected Packet Rate” attribute of the “Connection Objects” according to the DeviceNet specification.

Electronic Key

Serves to check the devices within the network at the system StartUp. The electronic key is read from the devices at every system StartUp and compared with the saved configuration.

Polled

Produced/Consumed
Activation of the “Polling” mode, cyclic writing and reading of IO data. Setting of the data content of the data transmitted via the polled IO connections. You can choose from digital data, analog data or both. The selection depends upon the BK52xx terminal arrangement.

Bit-Strobed

Produced/Consumed
Activation of the “Bit Strobe” operating mode. A broadcast message requests all nodes to send their bit strobe message (up to 7 bytes input or status data). Setting of the data content of the data transmitted via the bit-strobed IO connections. You can choose between digital data or diagnostic data.

Change of State / Cyclic

Produced/Consumed
Setting of the data content of the data transmitted via the change of state/cyclical IO connections. You can choose from digital data, analog data or both. The selection depends upon the BK52xx terminal arrangement.

Change of State / Cyclic
Selection of the required operating mode.

Heartbeat-Rate / Send-Rate
In the “Change of State” mode the heartbeat rate gives the cycle time of the cyclical send of the lower-level (i.e. in addition to the event driven) IO data. In “Cyclic” mode the send rate specifies the cycle time with which the IO data are sent.

Inhibit Time
Delay time in “Change of State” mode; after a change of state IO data are sent after the specified time at the earliest.

Acknowledge Timeout
Time before the retransmission in the event of faulty acknowledgement of a change of state / cyclical message.

Acknowledge Retry Limit
Maximum number of re-sends until IO connection goes into error mode.

K-Bus update

Calculates the expected time required for a full update of the terminal bus (depends on the connected terminals).

Auto Device Replacement (ADR)

Not supported.

“Startup Attributes” tab

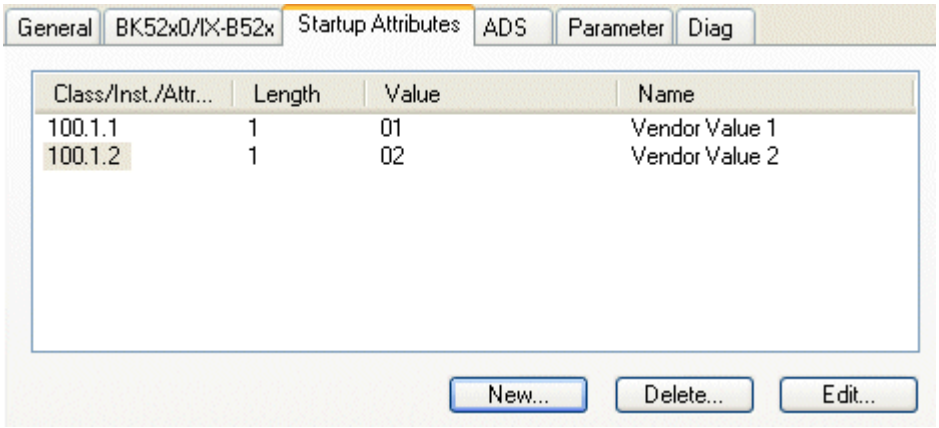


Fig. 61: “Startup Attributes” tab

The startup attributes are sent to the slave before the cyclic data exchange. The messages are sent before the actual IO data traffic.

Use the “New” or “Edit” button for configuration:

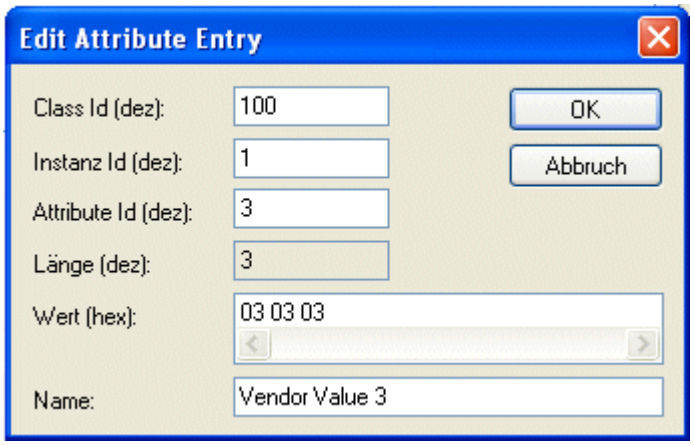


Fig. 62: Edit an attribute entry

The attributes are initialized via Class/Instance/Attributes. Note the “Value” specification in hexadecimal form.

“ADS” tab

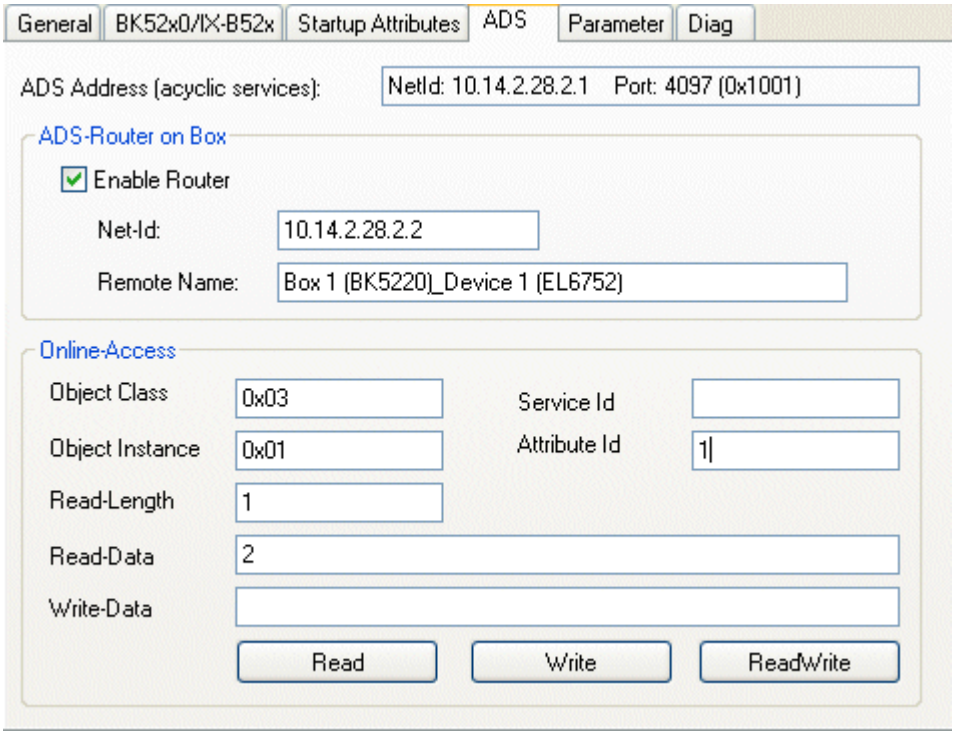


Fig. 63: “ADS” tab

The node (Bus Coupler) is assigned an ADS port to enable writing and reading of DeviceNet objects at runtime (e.g. from the PLC). It can be changed if required. A detailed description of explicit messages can be found in section “DeviceNet Communication” under “Explicit Messages”.

DeviceNet objects can be accessed via Online Access. To this end the DeviceNet-specific information such as Class/Instance/Attributes has to be entered.

Read
Reading of an object attribute via DeviceNet “Get_Attribute_Single” service. A service ID is not required.

Write
Writing of an object attribute via DeviceNet “Set_Attribute_Single” service. A service ID is not required.

Read / Write
Executing any DeviceNet service. Specification of the service ID is required.

“Parameter” tab

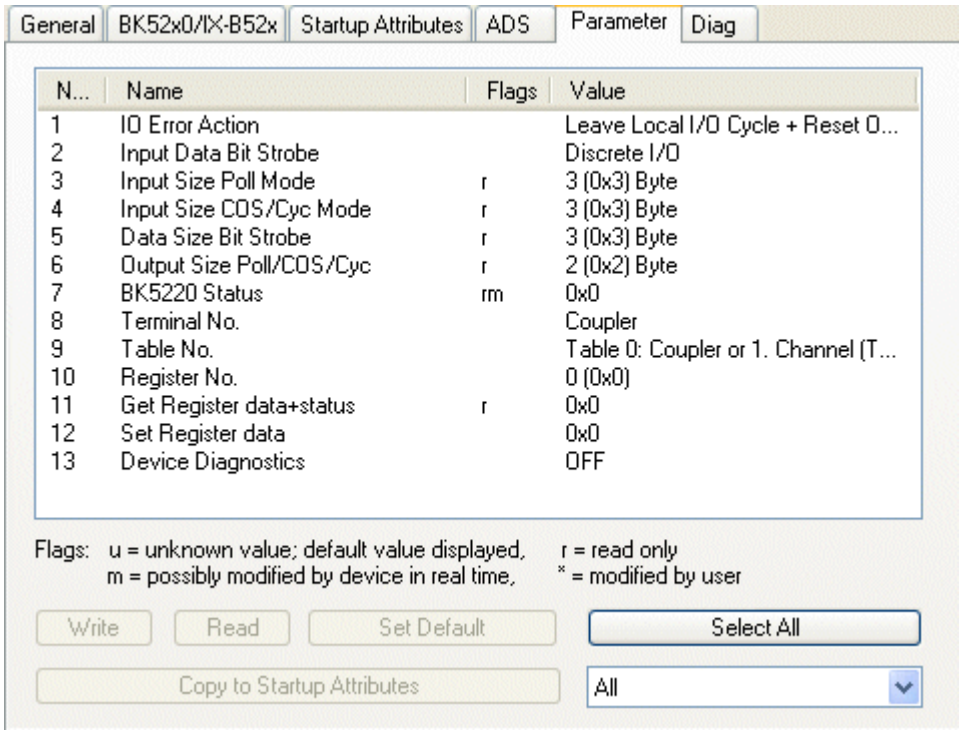


Fig. 64: “Parameter” tab

The parameters read from the EDS file are shown under the “Parameters” tab. Parameters can be read, written and entered in the list of the startup parameters.

“Diag” tab

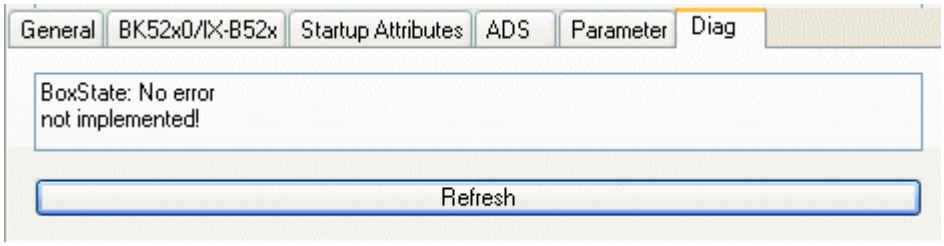


Fig. 65: "Diag" tab

The “Diag” tab indicates the state of the box. No further diagnostic options are available.

6.7 EtherCAT description

6.7.1 Introduction

The DeviceNet functionality and configuration options can be changed and parameterized depending on the different EtherCAT states.

EtherCAT states

The EtherCAT states (INIT, PREOP, SAFEOP, OP) have the following meaning according to the fieldbus-specific functions:

EtherCAT state	Meaning
INIT	Fieldbus not running
PREOP	Load fieldbus configuration
SAFEOP	Fieldbus cyclic operation, safe state. Inputs are read, outputs are not written
OP	Fieldbus cyclic operation. Inputs are read, outputs are written

The procedure and the configuration options are described below

6.7.1.1 EL6752 DeviceNet master configuration

The DeviceNet master and the associated DeviceNet slaves are configured in EtherCAT state PREOP. The DeviceNet master parameters are written via the EtherCAT object 0xF800, the slave parameters are written via the EtherCAT objects from 0x80n0 [▶ 73], see section EtherCAT Object Description.

The EtherCAT states are mapped to DeviceNet as follows:

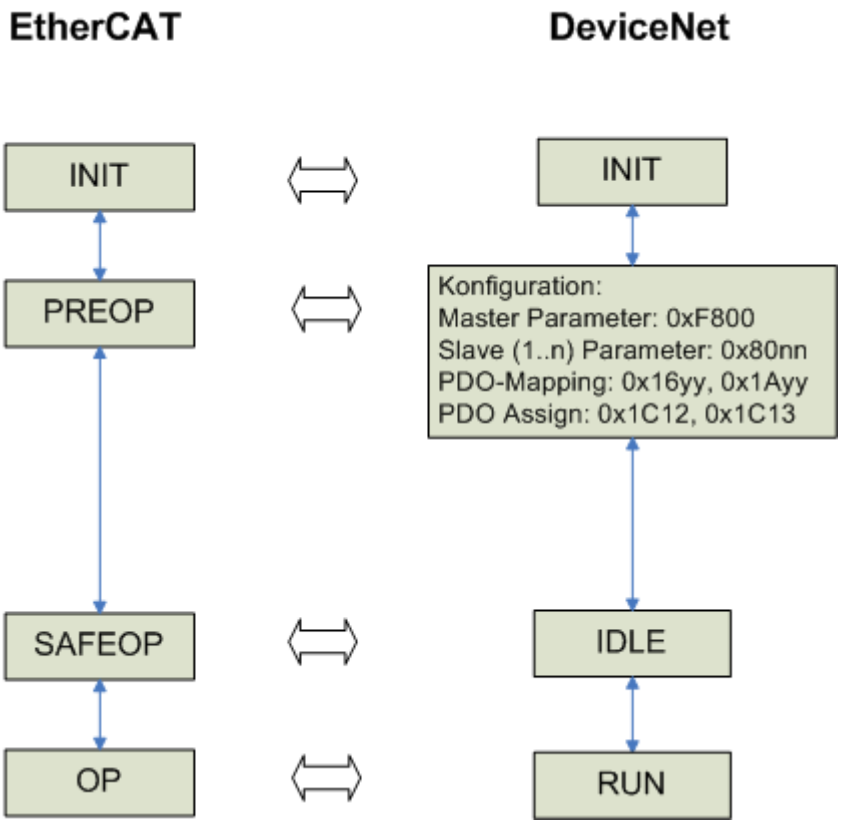


Fig. 66: EtherCAT states in mapping on EL6752-0000

The EtherCAT PDO mapping (EtherCAT objects 0x16yy, 0x1Ayy) and the PDO assignment (EtherCAT objects 0x1C12 [▶ 75], 0x1C13 [▶ 76]) can be read once the DeviceNet master parameters and the DeviceNet slave parameters have been written. The associated process image is then generated.

Once the DeviceNet master parameters have been written via the EtherCAT object 0xF800 [▶ 78], the DeviceNet master registers itself in the network and carries out the Duplicate MAC ID check.

Starting the fieldbus

During the EtherCAT state transition from PREOP to SAFEOP the DeviceNet master starts the data communication with the slaves and allocates the configured operating modes. In EtherCAT state SAFEOP the DeviceNet master is in IDLE mode. During the EtherCAT state transition from SAFEOP to OP the DeviceNet master switches to RUN mode.

Loading a new configuration

A new DeviceNet configuration can only be loaded through an EtherCAT state transition to IDLE or PREOP. The DeviceNet master parameters and DeviceNet slave parameters then have to be written again.

6.7.1.2 EL6752-0010 DeviceNet slave configuration

The DeviceNet slaves are configured in EtherCAT state PREOP. The general DeviceNet slave parameters are written via the EtherCAT object 0xF800 [▶ 83], the slave configuration data, i.e. the communication features and the IO configuration are written via the EtherCAT object 0x8000 [▶ 79], see section EtherCAT Object Description.

The EtherCAT states are mapped to DeviceNet as follows:

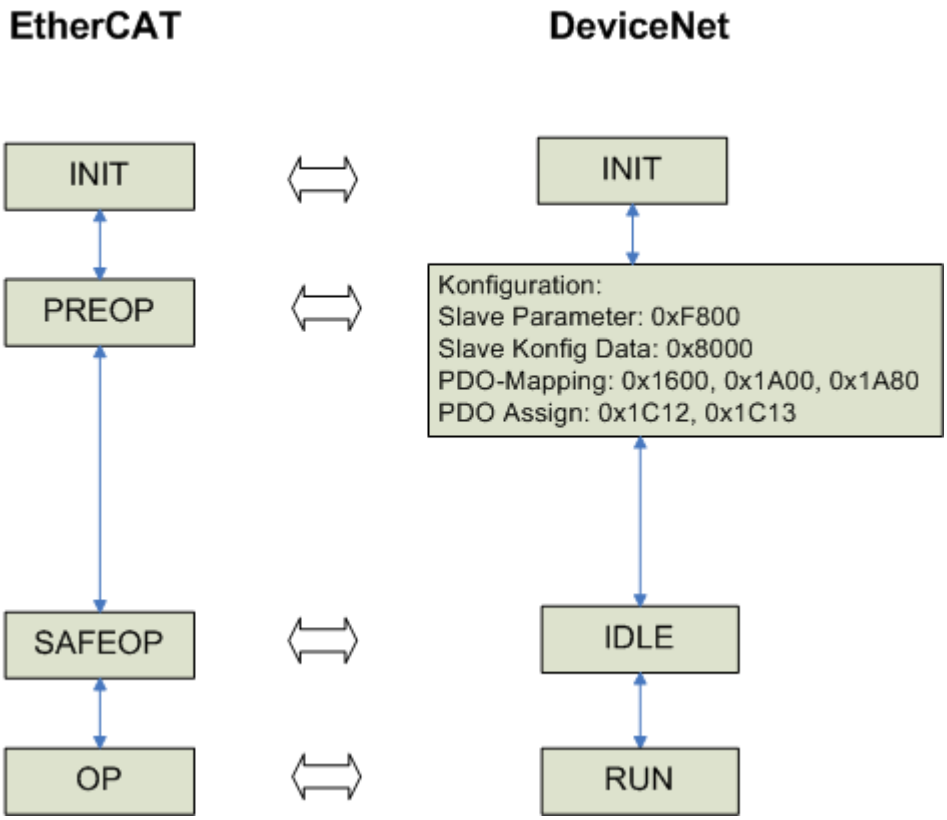


Fig. 67: EtherCAT states in mapping on EL6752-0010

The EtherCAT PDO mapping (EtherCAT objects 0x1600 [▶ 80], 0x1A00 [▶ 80], 0x1A80) and the PDO assignment (EtherCAT objects 0x1C12 [▶ 81], 0x1C13 [▶ 81]) can be read once the DeviceNet slave parameters and the DeviceNet slave configuration data have been written. The associated process image is then generated.

Once the DeviceNet slave parameters have been written via the EtherCAT object 0xF800 [▶ 83], the DeviceNet slave registers itself in the network and carries out the Duplicate MAC ID check.

Starting the fieldbus

During the EtherCAT state transition from PREOP to SAFEOP the DeviceNet slave starts the data communication, i.e. it is now ready for communication with a DeviceNet master. In EtherCAT state SAFEOP the DeviceNet slave is in IDLE mode. During the EtherCAT state transition from SAFEOP to OP the DeviceNet slave switches to RUN mode.

Loading a new configuration

A new DeviceNet configuration can only be loaded through an EtherCAT state transition to IDLE or PREOP. The DeviceNet slave parameters and DeviceNet slave configuration data then have to be written again.

6.7.1.3 EL6752-0010 - Changing the DeviceNet address and baud rate using ADS

The DeviceNet address (MACId) and the baud rate of the EL6752-0010 DeviceNet Slave terminal can be set using an ADS command in addition to the familiar functions as already described in the chapter "Configuration with the TwinCAT System Manager [▶ 47]"

ADS command

Setting the MAC-ID and the baud rate using ADS

```
IDXGRP=0x1F480
Index Offset 0x00

LEN=6

DATA[0]=0x45
DATA[1]=0x23
DATA[2]=MACId (0 _ 63)
DATA[3]=0
DATA[4]=Baudrate (1=500k, 2=250k, 3=125k)
DATA[5]=0

Ams Net Id: die der EL6752
Ams Port: 200
```

After writing the command the terminal must be switched once to INIT and then back to OP. The set data can be read in the object 0xF800 Index 1 (MAC ID) and Index 2 (baud rate).

Command taking the example of the TwinCAT AMS ADS Viewer

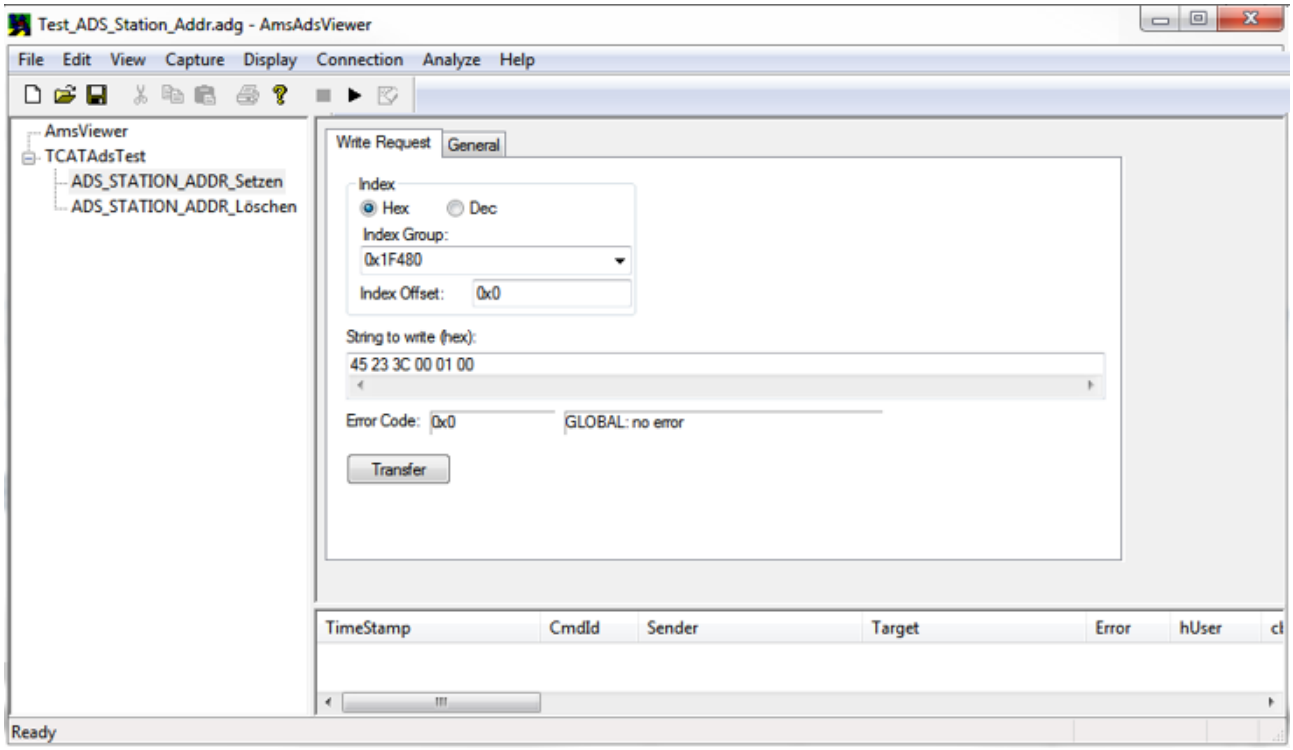


Fig. 68: ADS command with the data 3C - MACId (60dec) and 01 - baud rate (500k)

Reset

Once the MAC ID and the baud rate have been set using the ADS command, the terminal stores the information persistently. Once these data have been written, the entries in the objects 0x8000:01, 0xF800:01 and 0xF800:02 are ignored! ! This concerns the start-up commands, which are then ignored by the terminal.

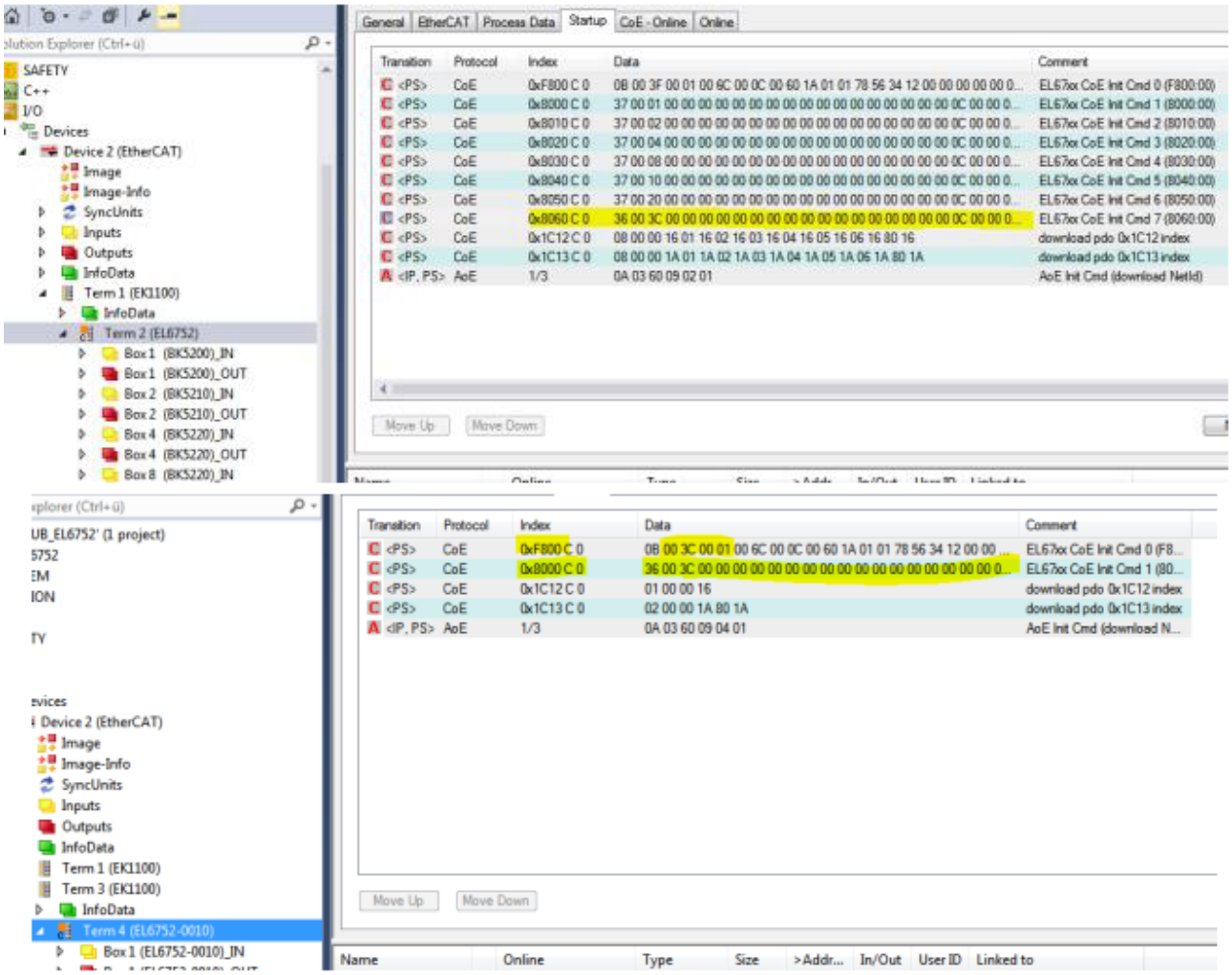


Fig. 69: Example of start-up CMD (0x8000:01; 0xF800:01 and 0xF800:02) that are ignored by the slave terminal after successfully setting the MACId and baud rate using ADS

ADS command (reset)

```
IDXGRP=0x1F480
Index Offset 0x00

LEN=6

DATA[0]=0
DATA[1]=0
DATA[2]=0
DATA[3]=0
DATA[4]=0
DATA[5]=0


Ams Net Id: die der EL6752
Ams Port: 200
```

In this way the data can be permanently deleted again and the terminal behaves as in the delivery condition.

Reset command taking the example of the TwinCAT AMS ADS Viewer

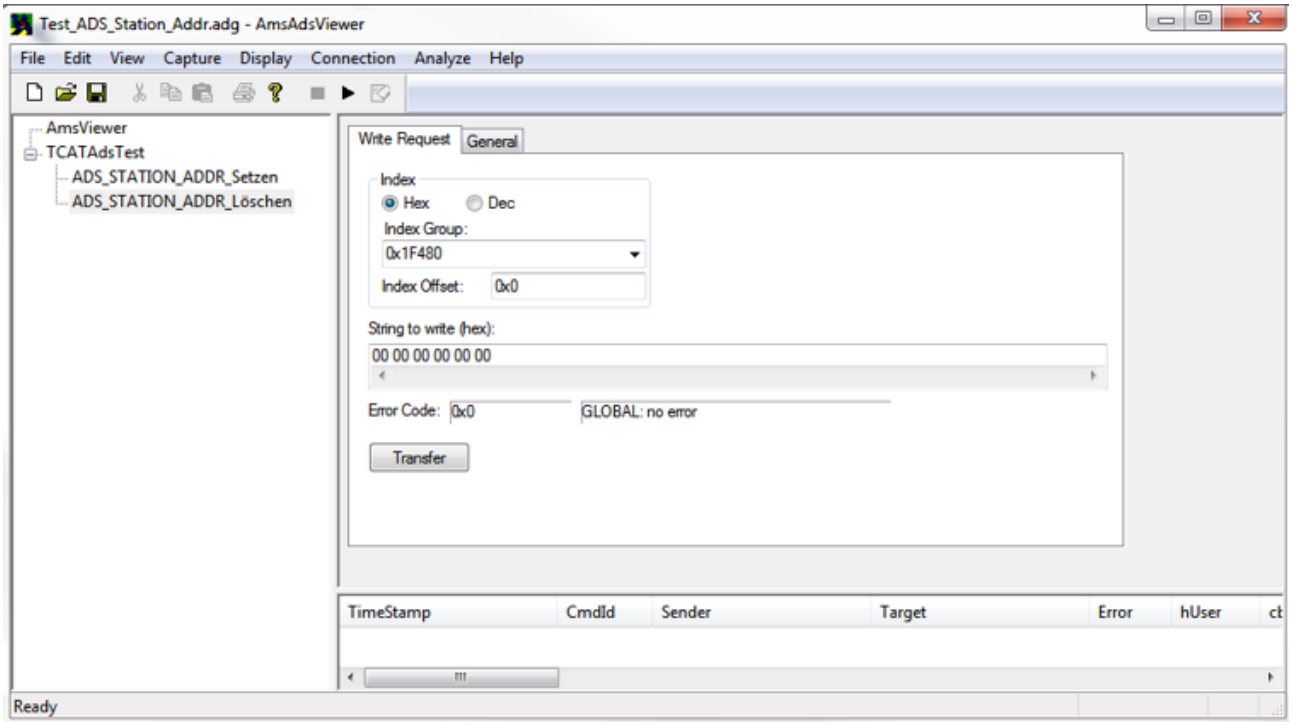


Fig. 70: Resetting the persistent data for MAC ID and baud rate

6.7.2 Object description and parameterization

6.7.2.1 DeviceNet master - EL6752

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 [▶ 34] when using/manipulating the CoE parameters:

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

Introduction

The CoE overview contains objects for different intended applications:

- Objects required for parameterization during commissioning
- Objects for indicating internal settings (may be fixed)

The parameterization and the objects required for normal operation will be presented first of all below. All further objects that are not needed for the normal application case can be found in the lower section of the table.

6.7.2.1.1 Objects for the parameterization

Index 8000-803E configuration data

Index (hex)	Name	Meaning	Data type	Flags	Default
8000+n*16:0	Configuration data	(for each module one object is defined (0 ≤ n < maximum number of modules))	UINT8	RW	0x33 (51_{dez})
(8000+n*16):01	MAC ID	DeviceNet device address (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):03	ProductName	Product name	OCTET-STRING[32]	RW	{0}
(8000+n*16):04	Device Type	Device type (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dez})
(8000+n*16):05	Vendor ID	Manufacturer ID (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):06	Product Code	Product code (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):07	Revision Number	Version number (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dez})
(8000+n*16):08	Serial Number	Serial number (see DeviceNet specification)	UINT32	RW	0x00000000 (0_{dec})
(8000+n*16):1D	Network Flags	Reserved for AMS via DeviceNet	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):1E	Network Port	Reserved for AMS via DeviceNet	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):1F	Network Segment Address	Reserved for AMS via DeviceNet	OCTET-STRING[6]	RW	{0}
(8000+n*16):20	Allocation Choice	DeviceNet mode selection (see DeviceNet specification) Bit 0: reserved (0) Bit1: Polled Bit2: Bit-Strobed Bit3: reserved (0) Bit4: Change of State Bit5: Cyclic Bit6: Acknowledge Suppression Bit7: reserved(0)	UINT16	RW	0x0100 (256_{dec})
(8000+n*16):21	Expected Packet Rate - Poll	Timing parameter for the poll connection (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):22	Expected Packet Rate - Bit Strobe	Timing parameter for the bit strobe connection (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):23	Expected Packet Rate - COS/Cyclic	Timing parameter for the COS/cyclic connection (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):24	Produced Data Size - Poll	Data length in poll mode	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):25	Produced Data Size - Bit Strobe	Data length in bit strobe mode	UINT16	RW	0x0000 (0_{dez})
(8000+n*16):26	Produced Data Size - COS/Cyclic	Data length in change of state / cyclic mode	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):27	Consumed Data Size - Poll	Data length in poll mode	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):28	Consumed data size - Bit strobe	Data length in bit strobe mode	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):29	Consumed Data Size - COS/Cyclic	Data length in change of state / cyclic mode	UINT16	RW	0x0000 (0_{dec})

Index (hex)	Name	Meaning	Data type	Flags	Default
(8000+n*16):2A	Electronic Key	Electronic key bit mask: Bit 0: Check Vendor Id Bit 1: Check DeviceType Bit 2: Check Product Code Bit 3: Check Revision Bit 4: reserved(0) Bit 5: reserved(0) Bit 6: reserved(0) Bit 7: reserved(0)	UINT16	RW	0x0100 (256_{dec})
(8000+n*16):2B	Acknowledge Timer	Timing parameter for the COS/cyclic connection (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):2C	Acknowledge Retry Limit	Timing parameter for the COS/cyclic connection (see DeviceNet specification)	UINT16	RW	0x0100 (256_{dec})
(8000+n*16):2D	Inhibit Time	Timing parameter for the COS/cyclic connection (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):2E	Produced data type - poll	reserved	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):2F	Produced data type - Bit strobe	reserved	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):30	Produced data type - COS/cyclic	reserved	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):31	Consumed data type - poll	reserved	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):32	Consumed data type - Bit strobe	reserved	UINT16	RW	0x0000 (0_{dec})
(8000+n*16):33	Consumed data type - COS/cyclic	reserved	UINT16	RW	0x0000 (0_{dec})

6.7.2.1.2 Objects for internal settings

Standard objects (0x1000-0x1FFF)

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	0x14501389 (340792201_{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	EL6752

Index 1009 Hardware version

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

Index 100A Software version

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

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	0x1A603052 (442511442_{dec})
1018:03	Revision	Revision number of the EtherCAT slave; the low word (bit 0-15) indicates the special terminal number, the high word (bit 16-31) refers to the device description	UINT32	RO	0x00100000 (1048576_{dec})
1018:04	Serial number	Serial number of the EtherCAT slave; the low byte (bit 0-7) of the low word contains the year of production, the high byte (bit 8-15) of the low word contains the week of production, the high word (bit 16-31) is 0	UINT32	RO	0x00000000 (0_{dec})

Index 1A85 DNM TxPDO-Map Device

Index (hex)	Name	Meaning	Data type	Flags	Default
1A85:0	DNM TxPDO-Map Device	PDO Mapping TxPDO 134	UINT8	RW	0x09 (9_{dec})
1A85:01	SubIndex 001	1. PDO Mapping entry (object 0xF100 (DeviceNet status), entry 0x01 (Communication status))	UINT32	RW	0xF100:01, 8
1A85:02	SubIndex 002	2. PDO Mapping entry (7 bits align)	UINT32	RW	0x0000:00, 7
1A85:03	SubIndex 003	3. PDO Mapping entry (object 0xF100 (DeviceNet status), entry 0x10 (TxPdoState))	UINT32	RW	0xF100:10, 1
1A85:04	SubIndex 004	4. PDO Mapping entry (object 0xF101 (Network status), entry 0x01 (Device status))	UINT32	RW	0xF101:01, 8
1A85:05	SubIndex 005	5. PDO Mapping entry (object 0xF101 (Network status), entry 0x09 (CAN BUS-OFF))	UINT32	RW	0xF101:09, 1
1A85:06	SubIndex 006	6. PDO Mapping entry (object 0xF101 (Network status), entry 0x0A (CAN warning limit))	UINT32	RW	0xF101:0A, 1
1A85:07	SubIndex 007	7. PDO Mapping entry (object 0xF101 (Network status), entry 0x0B (CAN Overrun))	UINT32	RW	0xF101:0B, 1
1A85:08	SubIndex 008	8. PDO Mapping entry (5 bits align)	UINT32	RW	0x0000:00, 5
1A85:09	SubIndex 009	9. PDO Mapping entry (object 0xF101 (Network status), entry 0x11 (CAN BUS load))	UINT32	RW	0xF101:11, 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	UINT8	RW	0x00 (0_{dec})
1C12:01		1st allocated RxPDO (contains the index of the associated RxPDO mapping object)			
...					
1C12:FF		255th 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	UINT8	RW	0x01 (1_{dec})
1C13:01		1st allocated TxPDO (contains the index of the associated TxPDO mapping object)			
...					
1C13:FF		255th allocated TxPDO (contains the index of the associated TxPDO mapping object)			

Profile-specific objects (0x6000-0xFFFF)

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

Index 6000-603E Poll Produced Data

Index (hex)	Name	Meaning	Data type	Flags	Default
6000+n*16:0	Poll Produced Data	Output data of the polling connection	UINT8	RO	0x01 (1_{dec})
(6000+n*16):01					
...					
(6000+n*16):01					

Index 6001-603F COS Produced Data

Index (hex)	Name	Meaning	Data type	Flags	Default
6001+n*16:0	COS Produced Data	Output data of the change of state connection	UINT8	RO	0x01 (1_{dec})
(6001+n*16):01					
...					
(6001+n*16):01					

Index 7000-703E Poll Consumed Data

Index (hex)	Name	Meaning	Data type	Flags	Default
7000+n*16:0	Poll Consumed Data	Input data of the polling connection	UINT8	RO	0x01 (1_{dec})
(7000+n*16):01					
...					
(7000+n*16):01					

Index 7001-703F COS Consumed Data

Index (hex)	Name	Meaning	Data type	Flags	Default
7001+n*16:0	COS Consumed Data	Input data of the change of state connection	UINT8	RO	0x01 (1_{dec})
(7001+n*16):01					
...					
(7001+n*16):01					

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	0x00FF (255_{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	List of the DeviceNet slaves connected to the EL6752.	UINT8	RW	0x00 (0_{dec})
F010:01		Product code of the first DeviceNet slave	UINT16	RO	0x00 (0_{dec})
...					
F010:FF					

Index F100 DeviceNet status

Index (hex)	Name	Meaning	Data type	Flags	Default
F100:0	DeviceNet status	DeviceNet status of the EL6752	UINT8	RO	0x10 (16_{dec})
F100:01	Number of Slaves not in Run	Number of DeviceNet slaves that are not in the RUN state	UINT8	RO	0x00 (0_{dec})
F100:10	TxPdoState	Status of the Tx-PDO	BOOLEAN	RO	0x00 (0_{dec})

Index F101 Network status

Index (hex)	Name	Meaning	Data type	Flags	Default
F101:0	Network status	Max. Subindex	UINT8	RO	0x11 (17_{dec})
F101:01	Device status	0: RUN MODE 1: IDLE MODE 2: Duplicate MacId Check failed, MAC ID used 3: Status: Selftest 4: Status: Standby 5: Status:Major Recoverable Fault 6: Status:Minor Recoverable Fault 7: DeviceNet Voltage Error 8: DeviceNet Access Error	UINT8	RO	0x00 (0_{dec})
F101:09	CAN BUS-OFF	CAN controller of the EL6752 is in state bus-off	BOOLEAN	RO	0x00 (0_{dec})
F101:0A	CAN warning limit	CAN controller of the EL6752 has exceeded the warning limit	BOOLEAN	RO	0x00 (0_{dec})
F101:0B	CAN Overrun	CAN controller of the EL6752 is in state bus-off	BOOLEAN	RO	0x00 (0_{dec})
F101:11	CAN BUS load	CAN bus load 0 - 100%	UINT16	RO	0x0000 (0_{dec})

Index F800 bus parameter set

Index (hex)	Name	Meaning	Data type	Flags	Default
F800:0	Bus Parameter set	Max. Subindex	UINT8	RW	0x08 (8_{dez})
F800:01	MAC ID	Device address of the DeviceNet device (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
F800:03	Product Name	Product name of the DeviceNet device (see DeviceNet specification)	OCTET-STRING[32]	RW	{0}
F800:04	DeviceType	Device type of the DeviceNet device (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
F800:05	Vendor ID	Manufacturer ID of the DeviceNet device (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
F800:06	Product Code	Product code of the DeviceNet device (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
F800:07	Revision Number	Version number of the DeviceNet device (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
F800:08	Serial Number	Serial number of the DeviceNet device (see DeviceNet specification)	UINT32	RW	0x00350000 (3473408_{dec})
F800:09	Baud rate	DeviceNet Baud Rate	UINT16	RW	0x0100 (256_{dec})

6.7.2.2 DeviceNet Slave - EL6752-0010

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 [▶ 34] when using/manipulating the CoE parameters:

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

Introduction

The CoE overview contains objects for different intended applications:

- Objects required for parameterization during commissioning
- Objects for indicating internal settings (may be fixed)

The parameterization and the objects required for normal operation will be presented first of all below. All further objects that are not needed for the normal application case can be found in the lower section of the table.

6.7.2.2.1 Objects for the parameterization

Index 8000 configuration data

Index (hex)	Name	Meaning	Data type	Flags	Default
8000:0	Configuration Data	Max. Subindex	UINT8	RW	0x33 (51_{dez})
8000:01	MAC ID	DeviceNet device address (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
8000:03	ProductName	Product name	OCTET-STRING[32]	RW	{0}
8000:04	Device Type	Device type (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
8000:05	Vendor ID	Manufacturer ID (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
8000:06	Product Code	Product code (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
8000:07	Revision Number	Version number (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
8000:08	Serial Number	Serial number (see DeviceNet specification)	UINT32	RW	0x00000000 (0_{dec})
8000:1D	Network Flags	Reserved for AMS via DeviceNet	UINT16	RW	0x0000 (0_{dec})
8000:1E	Network Port	Reserved for AMS via DeviceNet	UINT16	RW	0x0000 (0_{dec})
8000:1F	Network Segment Address	Reserved for AMS via DeviceNet	OCTET-STRING[2]	RW	{0}
8000:20	Allocation Choice	DeviceNet mode selection (see DeviceNet specification) Bit 0: reserved (0) Bit 1: Polled Bit 2: Bit-Strobed Bit 3: reserved (0) Bit 4: Change of State Bit 5: Cyclic Bit 6: Acknowledge Suppression Bit 7: reserved(0)	UINT16	RW	0x0100 (256_{dec})
8000:21	Expected Packet Rate - Poll	Timing parameter for the poll connection (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
8000:22	Expected Packet Rate - Bit Strobe	Timing parameter for the bit strobe connection (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
8000:23	Expected Packet Rate - COS/Cyclic	Timing parameter for the COS/cyclic connection (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
8000:24	Produced Data Size - Poll	Data length in poll mode	UINT16	RW	0x0000 (0_{dec})
8000:25	Produced Data Size - Bit Strobe	Data length in bit strobe mode	UINT16	RW	0x0000 (0_{dec})
8000:26	Produced Data Size - COS/Cyclic	Data length in change of state / cyclic mode	UINT16	RW	0x0000 (0_{dec})
8000:27	Consumed Data Size - Poll	Data length in poll mode	UINT16	RW	0x0000 (0_{dec})
8000:28	Consumed data size - Bit strobe	Data length in bit strobe mode	UINT16	RW	0x0000 (0_{dec})
8000:29	Consumed Data Size - COS/Cyclic	Data length in change of state / cyclic mode	UINT16	RW	0x0000 (0_{dec})

6.7.2.2.2 Objects for internal settings

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	0x145A1389 (341447561_{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	EL6752-0010

Index 1009 Hardware version

Index (hex)	Name	Meaning	Data type	Flags	Default
1009:0	Hardware version	Hardware-Version des EtherCAT-Slaves	STRING	RO	00

Index 100A Software version

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

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	0x1A603052 (442511442_{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	0x0010000A (1048586_{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 1600 DNS RxPDO-Map

Index (hex)	Name	Meaning	Data type	Flags	Default
1600:0	DNS RxPDO-Map	PDO Mapping RxPDO 1	UINT8	RW	0x00 (0_{dec})
1600:01					
...					
1600:FF					

Index 1A00 DNS TxPDO-Map

Index (hex)	Name	Meaning	Data type	Flags	Default
1A00:0	DNS TxPDO-Map	PDO Mapping TxPDO 1	UINT8	RW	0x00 (0_{dec})
1A00:01					
...					
1A00:FF					

Index 1A01 DNM TxPDO-Map Device

Index (hex)	Name	Meaning	Data type	Flags	Default
1A01:0	DNM TxPDO-Map Device	PDO Mapping TxPDO 2	UINT8	RW	0x09 (9_{dec})
1A01:01	SubIndex 001	1. PDO Mapping entry (object 0xF100 (DeviceNet status), entry 0x01 (Communication status))	UINT32	RW	0xF100:01, 8
1A01:02	SubIndex 002	2. PDO Mapping entry (7 bits align)	UINT32	RW	0x0000:00, 7
1A01:03	SubIndex 003	3. PDO Mapping entry (object 0xF100 (DeviceNet status), entry 0x10 (TxPdoState))	UINT32	RW	0xF100:10, 1
1A01:04	SubIndex 004	4. PDO Mapping entry (object 0xF101 (Network status), entry 0x01 (Device status))	UINT32	RW	0xF101:01, 8
1A01:05	SubIndex 005	5. PDO Mapping entry (object 0xF101 (Network status), entry 0x09 (CAN BUS-OFF))	UINT32	RW	0xF101:09, 1
1A01:06	SubIndex 006	6. PDO Mapping entry (object 0xF101 (Network status), entry 0x0A (CAN warning limit))	UINT32	RW	0xF101:0A, 1
1A01:07	SubIndex 007	7. PDO Mapping entry (object 0xF101 (Network status), entry 0x0B (CAN Overrun))	UINT32	RW	0xF101:0B, 1
1A01:08	SubIndex 008	8. PDO Mapping entry (5 bits align)	UINT32	RW	0x0000:00, 5
1A01:09	SubIndex 009	9. PDO Mapping entry (object 0xF101 (Network status), entry 0x11 (CAN BUS load))	UINT32	RW	0xF101:11, 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	UINT8	RW	0x01 (1_{dec})
1C12:01	SubIndex 001	1st allocated RxPDO (contains the index of the associated RxPDO mapping object)	UINT16	RW	0x1600 (5632_{dec})

Index 1C13 TxPDO assign

Index (hex)	Name	Meaning	Data type	Flags	Default
1C13:0	TxPDO assign	PDO Assign Inputs	UINT8	RW	0x02 (2_{dec})
1C13:01	SubIndex 001	1st allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A00 (6656_{dec})
1C13:02	SubIndex 002	2nd allocated TxPDO (contains the index of the associated TxPDO mapping object)	UINT16	RW	0x1A01 (6657_{dec})

Profile-specific objects (0x6000-0xFFFF)

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

Index 6000 Poll Produced Data

Index (hex)	Name	Meaning	Data type	Flags	Default
6000:0	Poll Produced Data	Output data of the polling connection	UINT8	RO	0x01 (1_{dec})
6000:01					
...					
6000:01					

Index 6001 COS Produced Data

Index (hex)	Name	Meaning	Data type	Flags	Default
6001:0	COS Produced Data	Output data of the change of state connection	UINT8	RO	0x01 (1_{dec})
6001:01					
...					
6001:01					

Index 7000 Poll Consumed Data

Index (hex)	Name	Meaning	Data type	Flags	Default
7000:0	Poll Consumed Data	Input data of the polling connection	UINT8	RO	0x01 (1_{dec})
7000:01					
...					
7000:01					

Index 7001 COS Consumed Data

Index (hex)	Name	Meaning	Data type	Flags	Default
7001:0	COS Consumed Data	Input data of the change of state connection	UINT8	RO	0x01 (1_{dec})
7001:01					
...					
7001:01					

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 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	List of connected devices	UINT8	RW	0x01 (1_{dec})
F010:01	SubIndex 001	Product code EL6752-0010	UINT32	RW	0x0000145A (5210_{dec})

Index F100 DeviceNet status

Index (hex)	Name	Meaning	Data type	Flags	Default
F100:0	DeviceNet status	DeviceNet status of the EL6752-0010	UINT8	RO	0x10 (16_{dec})
F100:01	Communication status	Communication status of the EL6752-0010: 0 = No error 1 = Station deactivated 2 = Station not exists 18 = Station ready 31 = only for EtherCAT gateways: WC-State of cyclic EtherCAT frame is 1	UINT8	RO	0x00 (0_{dec})
F100:10	TxPdoState	Status of the Tx-PDO	BOOLEAN	RO	0x00 (0_{dec})

Index F101 Network status

Index (hex)	Name	Meaning	Data type	Flags	Default
F101:0	Network status	Max. Subindex	UINT8	RO	0x11 (17_{dec})
F101:01	Device status	0: RUN MODE 1: IDLE MODE 2: Duplicate MacId Check failed, MAC ID used 3: Status: Selftest 4: Status: Standby 5: Status:Major Recoverable Fault 6: Status:Minor Recoverable Fault 7: DeviceNet Voltage Error 8: DeviceNet Access Error	UINT8	RO	0x00 (0_{dec})
F101:09	CAN BUS-OFF	CAN controller of the EL6752-0010 is in state bus-off	BOOLEAN	RO	0x00 (0_{dec})
F101:0A	CAN warning limit	CAN controller of the EL6752-0010 has exceeded the warning limit	BOOLEAN	RO	0x00 (0_{dec})
F101:0B	CAN Overrun	CAN controller of the EL6752-0010 is in state bus-off	BOOLEAN	RO	0x00 (0_{dec})
F101:11	CAN BUS load	CAN bus load 0 - 100 %	UINT16	RO	0x0000 (0_{dez})

Index F800 bus parameter set

Index (hex)	Name	Meaning	Data type	Flags	Default
F800:0	Bus Parameter set	Max. Subindex	UINT8	RW	0x08 (8_{dec})
F800:01	MAC ID	Device address of the DeviceNet device (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
F800:03	Product Name	Product name of the DeviceNet device (see DeviceNet specification)	OCTET-STRING[32]	RW	{0}
F800:04	DeviceType	Device type of the DeviceNet device (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
F800:05	Vendor ID	Manufacturer ID of the DeviceNet device (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
F800:06	Product Code	Product code of the DeviceNet device (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
F800:07	Revision Number	Version number of the DeviceNet device (see DeviceNet specification)	UINT16	RW	0x0000 (0_{dec})
F800:08	Serial Number	Serial number of the DeviceNet device (see DeviceNet specification)	UINT32	RW	0x00350000 (3473408_{dec})
F800:09	Baud rate	DeviceNet Baud Rate	UINT16	RW	0x0100 (256_{dec})

7 Error handling and diagnostics

7.1 EL6752 - LED description

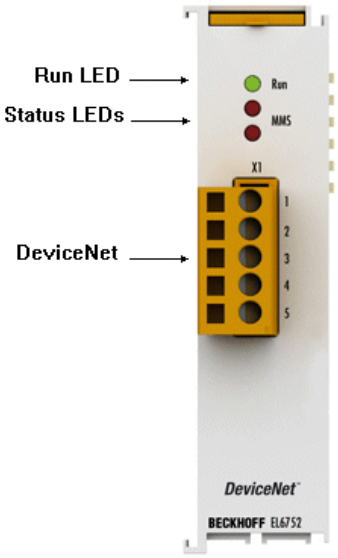


Fig. 71: LEDs

LED behaviour

The LEDs facilitate diagnosing of the main terminal states:

EL6752-0000 (DeviceNet master terminal)

LED	Colours	Meaning	
RUN	green	This LED indicates the terminal's operating state:	
		off	State of the EtherCAT State Machine: **INIT** = initialization of the terminal; **BOOTSTRAP** = function for terminal firmware updates
		flashing	State of the EtherCAT State Machine: **PREOP** = function for mailbox communication and different standard-settings set
		Single flash	State of the EtherCAT State Machine: **SAFEOP** = verification of the sync manager channels and the distributed clocks. Outputs remain in safe state
		on	State of the EtherCAT State Machine: **OP** = normal operating state; mailbox and process data communication is possible
MNS green	green	off	Master is offline
		flashing	Master is online and is performing the Duplicate MAC-ID check
		on	Master is online and is communicating with the configured slaves
MNS red	red	flashing	Communication error of the master with one of the configured slaves
		on	DeviceNet Bus OFF, DeviceNet voltage error, Master failed Duplicate MAC-ID check

EL6752-0010 (DeviceNet slave terminal)

LED	Color	Meaning	
RUN	green	This LED indicates the terminal's operating state:	
		off	State of the EtherCAT State Machine: **INIT** = initialization of the terminal; **BOOTSTRAP** = function for terminal firmware updates
		flashing	State of the EtherCAT State Machine: **PREOP** = function for mailbox communication and different standard-settings set
		Single flash	State of the EtherCAT State Machine: **SAFEOP** = verification of the sync manager channels and the distributed clocks. Outputs remain in safe state
		on	State of the EtherCAT State Machine: **OP** = normal operating state; mailbox and process data communication is possible
MNS green	green	off	Slave is offline
		flashing	Slave port has ended the Duplicate MAC-ID check (Network OK), communication error with the master.
		on	Slave port is online and is communicating with the master.
MNS red	red	flashing	Communication error of the slave port with the master, timeout of the slave port
		on	DeviceNet Bus OFF, DeviceNet voltage error, slave port error, error in Duplicate MAC-ID check

7.2 EL6752/-0010 diagnostics

The EL6752/-0010 feature various diagnostic variables that describe the state of the terminal and the DeviceNet and can be linked in the PLC:

We recommend monitoring the following process data during each cycle:

- **WcState**: if ≠ 0, this EtherCAT device does not take part in the process data traffic
- **State**: if ≠ 8, the EtherCAT device is not in OP (operational) status

We recommend monitoring the following process data:

- **Error**: if ≠ 0, the indicated number of DeviceNet devices has a BoxState not equal zero, i.e. check which DeviceNet devices are not operating correctly in the bus
- **DiagFlag**: indicates pending diagnostic data

7.2.1 EL6752/-0010 - WC-State

For monitoring the EtherCAT communication the WC state (working counter) of the EL6752/-0010 must be checked. If the WC state is not equal "0" the EtherCAT communication is disturbed, i.e. data sent to the slave or the master are no longer transferred correctly and are not valid.

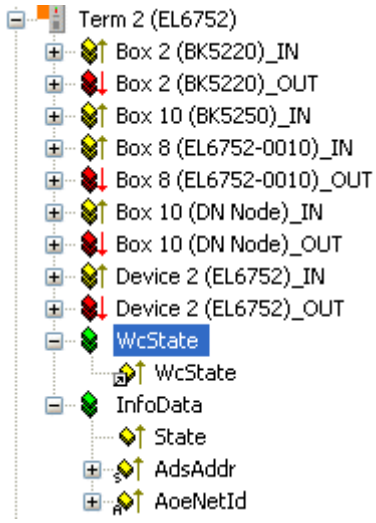


Fig. 72: WCState in the TwinCAT tree

WcState

0: data are valid
1: data are not valid, problem in the EtherCAT communication

7.2.2 EL6752/-0010 - State

The diagnostic state variable indicates the current EtherCAT state of the EL6752/-0010.

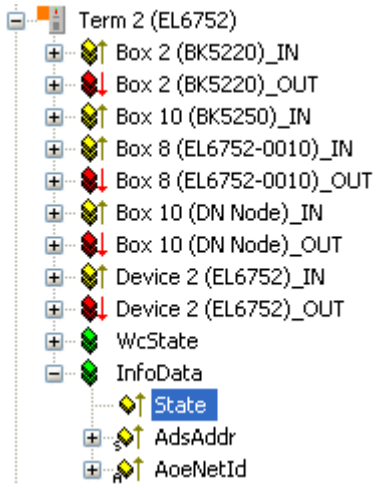


Fig. 73: State diagnostic variable in the TwinCAT tree

State

0x___1 = Slave in 'INIT' state
0x___2 = Slave in 'PREOP' state
0x___3 = Slave in 'BOOT' state
0x___4 = Slave in 'SAFEOP' state
0x___8 = Slave in 'OP' state
0x001_ = Slave signals error
0x002_ = Invalid vendorId, productCode... read
0x004_ = Initialization error occurred
0x010_ = Slave not present
0x020_ = Slave signals link error
0x040_ = Slave signals missing link
0x080_ = Slave signals unexpected link
0x100_ = Communication port A
0x200_ = Communication port B
0x400_ = Communication port C
0x800_ = Communication port D

7.2.3 EL6752/-0010 - Error / DiagFlag

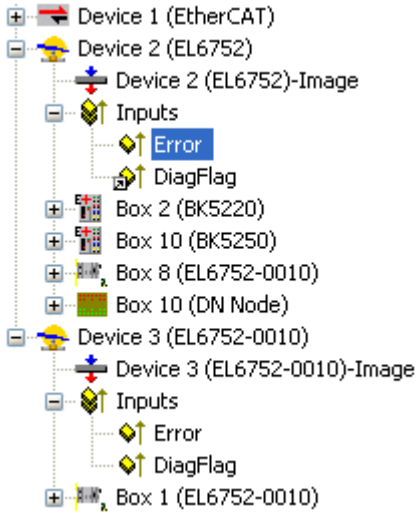


Fig. 74: Error and DiagFlag in the TwinCAT tree

Error

0: all DeviceNet devices have BoxState zero
>0: number of DeviceNet devices with BoxState not equal zero.

DiagFlag

0 = no diagnostic data are pending
1 = diagnostic data are pending and can be read via AdsRead services

7.3 DeviceNet device diagnostics

DeviceNet slave devices feature different diagnostic variables that describe the DeviceNet communication state and can be linked in the PLC:

We recommend monitoring the following process data during each cycle:

- **MacState**: if ≠ 0, this DeviceNet device is not participating correctly in the process data traffic
- **CouplerState**: for Beckhoff Bus Couplers, the terminal communication of the Bus Coupler may be disturbed or diagnostic data may be present if ≠ 0

7.3.1 DeviceNet slave device / EL6752-0010 - MacState

For monitoring the DeviceNet communication the MacState of the DeviceNet device / EL6752-0010 must be checked. If the MacState is not equal zero, the DeviceNet slave is not participating correctly in the DeviceNet data exchange.

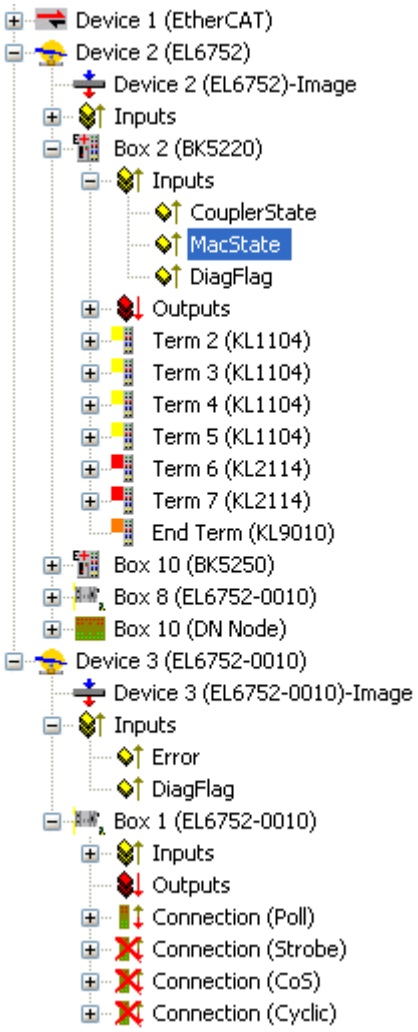


Fig. 75: MacState in the TwinCAT tree

MacState

0 = No error
1 = Station deactivated
2 = Station not exists
18 = Station ready
40 = Heartbeat Message not received
41 = Shutdown Message received
42 = Electronic Key Fault: Vendor Id
43 = Electronic Key Fault: Device Type
44 = Electronic Key Fault: Product Code
45 = Electronic Key Fault: Revision
46 = Fault while writing Start-Up Attributs
47 = wrong Produced IO-Data Size
48 = wrong Consumed IO-Data Size
49 = Idle Mode

7.3.2 DeviceNet slave device / EL6752-0010 - DiagFlag

The DiagFlag indicates pending diagnostic data. Pending diagnostic data can be read via an AdsRead command.

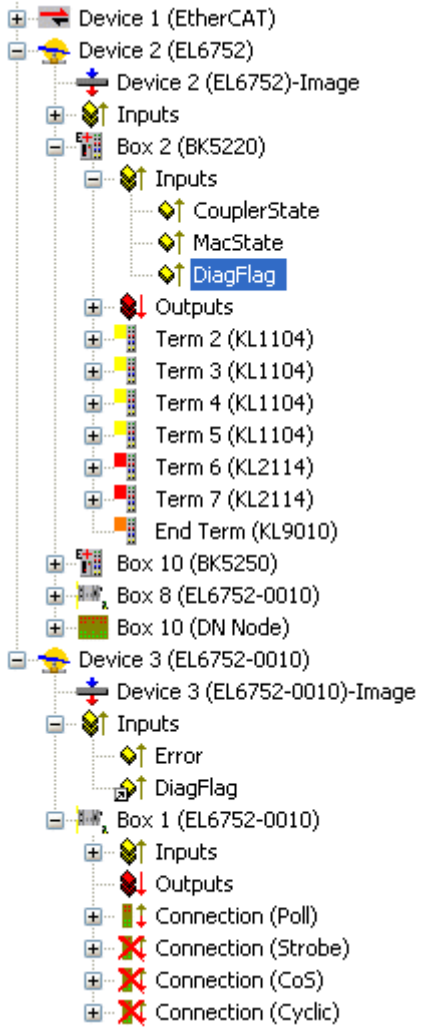


Fig. 76: DiagFlag in the TwinCAT tree

DiagFlag

0 = no diagnostic data are pending
1 = diagnostic data are pending and can be read via AdsRead services

7.3.3 Beckhoff DeviceNet slave device - CouplerState

The CouplerState provides information on the terminal bus communication of the Beckhoff Bus Coupler. This information is available for Beckhoff BK52x0 Bus Couplers, devices from the IPxxxx-B520 IP Box-family and the IP Link family.

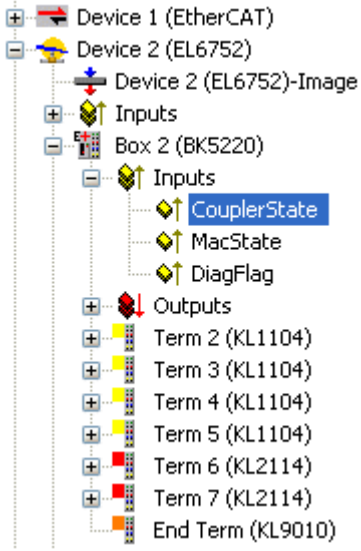


Fig. 77: CouplerState in the TwinCAT tree

CouplerState

0x00 = I/O Run
0x01 = I/O Error (KBus, IO or Terminal Error)
0x80 = I/O idle mode / fieldbus error, no output data are written
0x08= diagnostic information for an analog/special function terminal is pending. This function first hast to be activated at the couplers. The diagnostic data can then be read in the associated registers of the terminals, IP/IL modules

7.4 EL6752/-0010 - ADS Error Codes

The ADS error codes have the following meaning:

Error	Description
	Error during ADS/AMS data exchange
0x1001	Insufficient memory for AMS command
0x1101	Incorrect data length at StartFieldbus
0x1102	Incorrect DeviceState at StartFieldbus
0x1103	Device cannot change from INIT to RUN
0x1104	Incorrect AdsState in INIT state
0x1105	Incorrect DeviceState at StopFieldbus
0x1106	Device cannot change from STOP to RUN if a CDL is not defined
0x1107	Device cannot change from STOP to RUN if a box is not defined
0x1108	Incorrect data length at StartDataTransfer
0x1109	Incorrect DeviceState at StartDataTransfer
0x110A	Incorrect AdsState in STOP state
0x110B	Device cannot change from RUN to INIT
0x110C	Incorrect data length at StopDataTransfer
0x110D	Incorrect DeviceState at StopDataTransfer
0x1110	Incorrect AdsState in RUN state
0x1111	Loading the device parameters is only permitted in the INIT state
0x1112	Incorrect data length at SetDeviceState
0x1113	AddBox not allowed in INIT state
0x1114	Incorrect data length at AddBox
0x1115	DeleteBox not allowed in INIT state
0x1116	Incorrect IndexOffset at DeleteBox
0x1117	Incorrect data length at DeleteBox
0x1118	ReadBox only with AdsRead
0x1119	AddCdl not allowed in INIT state
0x111A	Incorrect data length at AddCdl
0x111B	DeleteCdl not allowed in INIT state
0x111C	Incorrect IndexOffset at DeleteCdl
0x111D	Incorrect data length at DeleteCdl
0x111E	Incorrect IndexGroup at AdsWrite
0x111F	Device parameters cannot be read

Error	Description
	Error during ADS/AMS data exchange
0x1120	Box parameters cannot be read
0x1121	Cdl parameters cannot be read
0x1122	DeleteBox or DeleteCdl only with AdsWrite
0x1123	ReadBox only possible in STOP state
0x1124	Incorrect IndexOffset at ReadBox
0x1125	Incorrect data length at ReadBox
0x1126	Incorrect IndexGroup at AdsRead
0x1127	AddDeviceNotification not allowed in INIT state
0x1128	DelDeviceNotification not allowed in INIT state
0x1129	IndexOffset too large during reading of the device diagnostic data
0x112B	IndexOffset too large during reading of the box diagnostic data
0x112F	Insufficient memory for ReadBox response
0x1201	AddCdl: CDL no. is too large
0x1202	DeleteCdl only possible when CDL is stopped
0x1203	DeleteCdl not possible as no CDL defined
0x1204	Cycle could not be completed within the internal watchdog time
0x1301	AddCdl: I/O access multiplier is too large
0x1302	AddCdl: Start cycle must be smaller than I/O access multiplier
0x1303	AddCdl: Incorrect data length for output area
0x1304	AddCdl: Incorrect data offset for output area
0x1305	AddCdl: Output area is already defined
0x1306	AddCdl: Incorrect data length for input area
0x1307	AddCdl: Incorrect data offset for input area
0x1308	AddCdl: Input area is already defined
0x1309	AddCdl: Incorrect area type
0x130A	AddCdl: BoxNo has not been defined with AddBox
0x130B	AddCdl: Incorrect action type
0x130C	AddCdl: Insufficient memory for poll list
0x130D	AddCdl: Insufficient memory for poll list array
0x130E	AddCdl: Insufficient memory for actions
0x130F	AddCdl: CdlNo already exists

Error	Description
	Error during ADS/AMS data exchange
0x1310	DeleteCdl: CDL is not stopped
0x1311	AddCdl: Insufficient memory for asynchronous transmit list
0x1312	AddCdl: Insufficient memory for synchronous receive list
0x1313	AddCdl: Insufficient memory for asynchronous receive list
0x1316	AddCdl: Insufficient memory for synchronous receive list
0x1318	AddCdl: Only slave action allowed
0x1319	AddCdl: Insufficient memory for slave list
0x1601	AddBox: BoxNo is too large
0x1602	AddBox: Insufficient memory for ADS StartUp telegram
0x1604	DeleteBox: Box is not stopped
0x1605	AddBox: Insufficient memory for CDL telegram
0x1606	AddBox: Number of CDL telegrams is too large
0x1607	BoxRestart: Box is not stopped
0x1608	BoxRestart: AdsWriteControl syntax error
0x1609	BoxRestart: Incorrect AdsState
0x160A	Syntax error in AdsWrite to box port
0x160B	AMS CmdId is not supported by box port
0x160E	AdsReadState is not supported by box port
0x160F	AddBox: Insufficient memory for the ADS interface
0x1610	AddBox: AMS channel is invalid
0x1611	Error communicating with an AMS box
0x1613	Error communicating with an AMS box: Incorrect offset
0x1614	Error communicating with an AMS box: Data packet is too large
0x1615	Error communicating with an AMS box: AMS command is too large
0x1616	Error communicating with an AMS box: First data packet is too large
0x1617	Error communicating with an AMS box: First offset is incorrect

Error	Description
	Error during ADS/AMS data exchange
0x1701	AddDeviceNotification: Length of device diagnostic data to small
0x1702	AddDeviceNotification: Length of device diagnostic data to large
0x1703	AddDeviceNotification: Length of box diagnostic data to small
0x1704	AddDeviceNotification: Length of box diagnostic data to large
0x1705	AddDeviceNotification: Box is not defined
0x1706	AddDeviceNotification: Incorrect IndexGroup
0x1707	AddDeviceNotification: No more resources for client
0x1708	DelDeviceNotification: Incorrect handle
0x1801	StartFieldbus: In equidistant operation, shift time + safety time + 2*PLL sync. time must be greater than the cycle time
0x1802	StartFieldbus: Cycle time is too large
0x1803	StartFieldbus: Cycle time is too large
0x1804	StartFieldbus: Shift time is too large
0x1805	StartFieldbus: PLL sync time is too large
0x1806	StartFieldbus: Safety time is too large
0x1807	StartFieldbus: Cycle times shorter than 1 ms must be integral divisors of 1 ms
0x1A01	Memory could not be allocated from the huge heap, because it is larger than 0x8000 bytes
0x1A02	Memory could not be allocated from the near heap, because it is larger than 0x1000 bytes
0x1A03	Memory could not be allocated from the huge heap, because it is 0 bytes
0x1A04	Memory could not be allocated from the near heap, because it is 0 bytes
	Error during initialization of the DeviceNet configuration
0x2001 .. 0x2xxx	
	Error during explicit DeviceNet data exchange
0x2300	GENERR_RESUNAVAILABLE
0x2301	ADSERR_DEVICE_SRVNOTSUPP
0x2302	GENERR_INVALATTRVAL
0x2303	GENERR_ALRERADYINREQU
0x2304	GENERR_OBJECTSTATECONF
0x2305	GENERR_ATTRNOTSETABLE
0x2306	GENERR_PRIVVIOLATION
0x2307	GENERR_REPLDATTOOLARGE
0x2308	GENERR_NOTENOUGHDATA
0x2309	GENERR_ATTRNOTSUPP
0x230A	GENERR_TOOMUCHDATA
0x230B	GENERR_OBJECTNOTEXIST
0x230C	GENERR_NOSTOREATTRDATA
0x230D	GENERR_STOREOPFAIL
0x230E	GENERR_VENDORSPEC
0x230F	GENERR_INVALPARAM
0x2310	GENERR_INVALMEMBERID
0x2311	GENERR_MEMBERNOTSET
0x2312	ADSERR_DEVICE_SYMBOLNOTFOUND
0x2313	GENERR_OBJECTSTATECONF

7.5 DeviceNet / CAN Trouble Shooting

Error Frames

One sign of errors in the CAN wiring, the address assignment or the setting of the baud rate is an increased number of error frames: the diagnostic LEDs then show *Warning Limit exceeded* or *Bus-off state entered*.

DeviceNet / CAN network analysis

CAN warning limit exceeded, passive error or bus-off state are indicated first of all at the node that has detected the most errors. These nodes are not necessarily the cause for the occurrence of error frames! If, for instance, one node contributes unusually heavily to the bus traffic (e.g. because it is the only one with analog inputs, the data for which triggers event-driven messages at a high rate), then the probability of its telegrams being damaged increases. Its error counter will, correspondingly, be the first to reach a critical level.

MAC ID / baud rate setting

Duplicate allocation of node addresses / MAC IDs must be avoided.

Test 1

Check MAC ID. If the DeviceNet communication works at least temporarily and all devices support the duplicate MAC ID check, the address assignment can also be checked by logging the duplicate MAC ID check messages when the devices are switched on, although this procedure does not detect incorrect allocation of node addresses.

Test 2

Check that the same baud rate has been set everywhere.

Testing the DeviceNet/CAN cabling

These tests should not be carried out if the network is active: No communication should take place during the tests. The following tests should be carried out in the stated sequence, because some of the tests assume that the previous test was successful. Not all the tests are generally necessary.

Network terminator and signal leads

The nodes should be switched off or the CAN cable unplugged for this test, because the results of the measurements can otherwise be distorted by the active CAN transceiver.

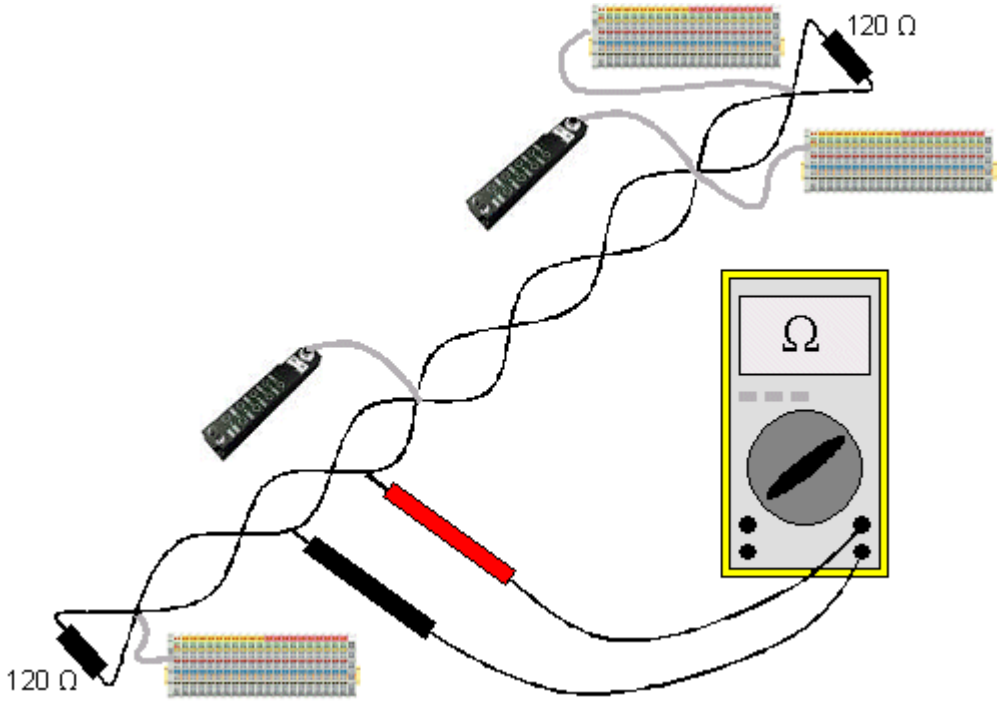


Fig. 78: Wiring diagram for test setup

Test 3

Determine the resistance between CAN high and CAN low - at each device, if necessary.

If the measured value is greater than 65 Ohms, it indicates the absence of a terminating resistor or a break in a signal lead. If the measured value is less than 50 Ohms, look for a short circuit between the CAN lines, more than the correct number of terminating resistors, or faulty transceivers.

Test 4

Check for a short circuit between the CAN ground and the signal leads, or between the screen and signal leads.

Test 5

Remove the earth connection from the CAN ground and screen. Check for a short circuit between the CAN ground and screen.

Topology

The possible cable length in CAN networks depends heavily on the selected baud rate. CAN will tolerate short drop lines - although this again depends on the baud rate. The maximum permitted drop line length should not be exceeded. The length of cable that has been installed is often underestimated - estimates can even be a factor of 10 less than the actual length. The following test is therefore recommended:

Test 6

Measure the drop line lengths and the total bus length (a rough estimate is not sufficient) and compare the values with the topology rules (depending on the baud rate).

Screening and earthing

The power supply and the screen should be carefully earthed at the power supply unit, once only and with low resistance. At all connecting points, branches and so forth the screen of the CAN cable (and possibly the CAN GND) must also be connected, as well as the signal leads. In the Beckhoff IP20 Bus Couplers, the screen is grounded for high frequencies via an R/C element.

Test 7

Use a DC ammeter (16 A max.) to measure the current between the power supply ground and the screen at the end of the network most remote from the power supply unit. An equalization current should be present. If there is no current, then either the screen is not connected all the way through, or the power supply unit is not properly earthed. If the power supply unit is somewhere in the middle of the network, the measurement should be performed at both ends. When appropriate, this test can also be carried out at the ends of the drop line.

Test 8

Interrupt the screen at a number of locations and measure the connection current. If current is flowing, the screen is earthed at more than one place, creating a ground loop.

Potential differences

The screen must be connected all the way through for this test, and must not be carrying any current - this has previously been tested.

Test 9

Measure and record the voltage between the screen and the power supply ground at each node. The maximum potential difference between any two devices should be less than 5 volts.

Detect and localize faults

The "low-tech approach" rule is the best localisation method: disconnect parts of the network, and observe when the fault disappears.

But: the approach based on this method is inadequate in the event of problems such as excessive potential differences, ground loops, EMC and signal corruption, since in many cases making the network smaller solves the problem notwithstanding the fact that the "missing" component may not have caused it. The bus load also changes as the network is reduced in size, which can mean that external interference "hits" CAN telegrams less often.

Diagnosis with an oscilloscope is not usually successful: even when they are in good condition, CAN signals can look really chaotic. It may be possible to trigger on error frames using a storage oscilloscope - this type of diagnosis, however, is only possible for expert technicians.

Protocol problems

In rare cases, protocol problems (e.g. faulty or incomplete DeviceNet implementation, unfavorable timing at boot up, etc.) can be the cause of faults. In this case it is necessary to trace the bus traffic for evaluation by DeviceNet experts - the Beckhoff support team [▶ 113] can help here.
A free channel on a Beckhoff FC5102 CANopen PCI card is appropriate for such a trace - Beckhoff make the necessary trace software available on the internet. Alternatively, it is of course possible to use a normal commercial CAN analysis tool.

8 Appendix

8.1 UL notice

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

Basic principles

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

c UL us

8.2 EtherCAT AL Status Codes

For detailed information please refer to the EtherCAT system description.

8.3 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 [▶ 101]. 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!

EL6752			
Hardware (HW)	**Firmware (FW)**	**Revision no.**	**Release date**
06 - 20*	07	EL6752-0000-0016	2008/06
	08		2008/11
	09		2010/05
		EL6752-0000-0017	2011/10
	10		2012/01
		EL6752-0000-0018	2012/10
	11	EL6752-0000-0019	2014/07
	12	EL6752-0000-0020	2014/06
	13*		2015/02

EL6752-0010			
Hardware (HW)	**Firmware (FW)**	**Revision no.**	**Release date**
06 - 20*	06	EL6752-0010-0016	2008/04
	07		2008/06
	08		2008/11
		EL6752-0010-0017	2011/10
	09		2012/01
	10		2012/05
		EL6752-0010-0018	2012/10
	11	EL6752-0010-0019	2014/07
	12	EL6752-0010-0020	2014/06
	13*		2015/02

*) 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.

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

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

Storage locations

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

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

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

The TwinCAT System Manager offers mechanisms for programming all 3 parts with new data, if the slave is set up for this purpose. Generally the slave does not check whether the new data are suitable, i.e. it may no longer be able to operate if the data are unsuitable.

Simplified update by bundle firmware

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

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

• Firmware downloads to an EtherCAT device must not be interrupted
• Flawless EtherCAT communication must be ensured. CRC errors or LostFrames must be avoided.
• The power supply must adequately dimensioned. The signal level must meet the specification.

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

8.4.1 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:

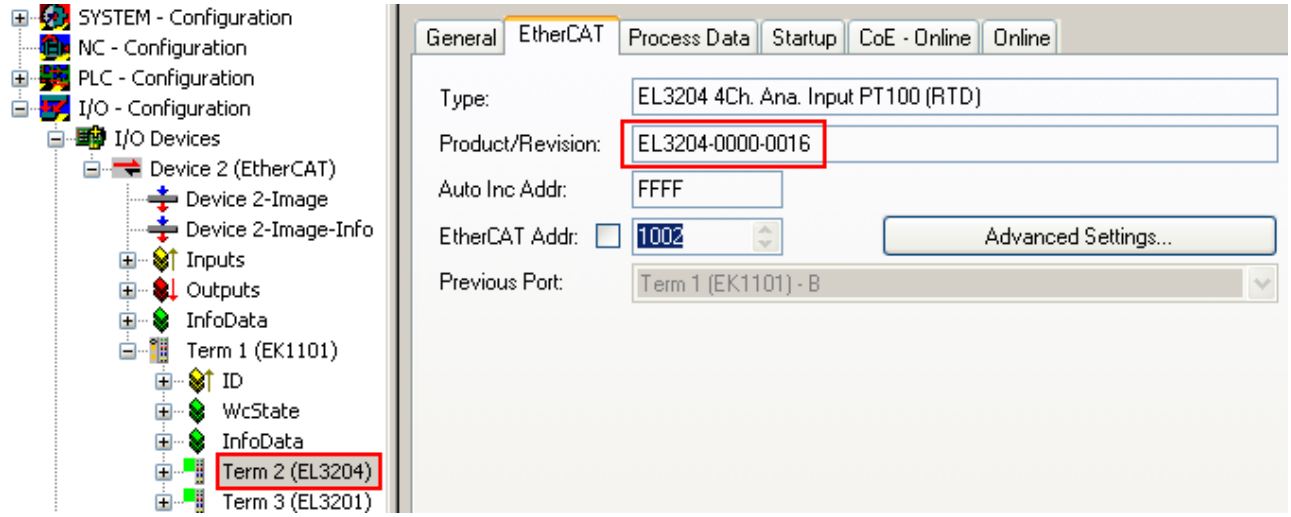


Fig. 79: Device identifier consisting of name EL3204-0000 and revision -0016

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:

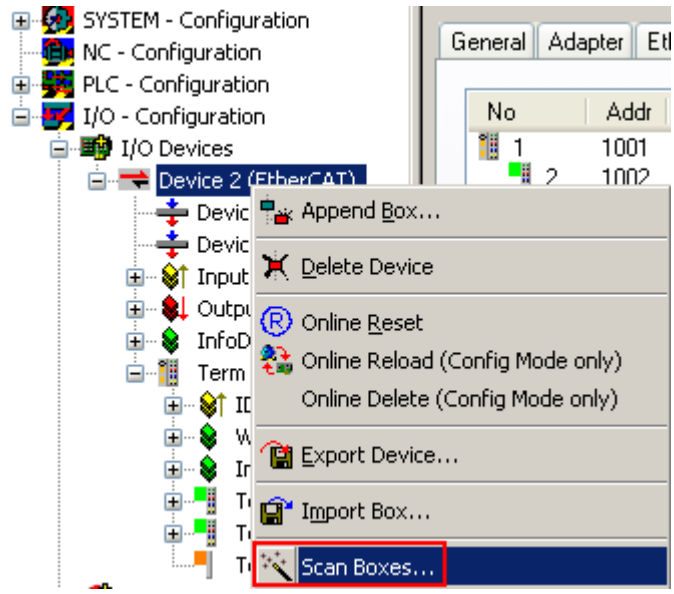


Fig. 80: Scan the subordinate field by right-clicking on the EtherCAT device

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

Fig. 81: Configuration is identical

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

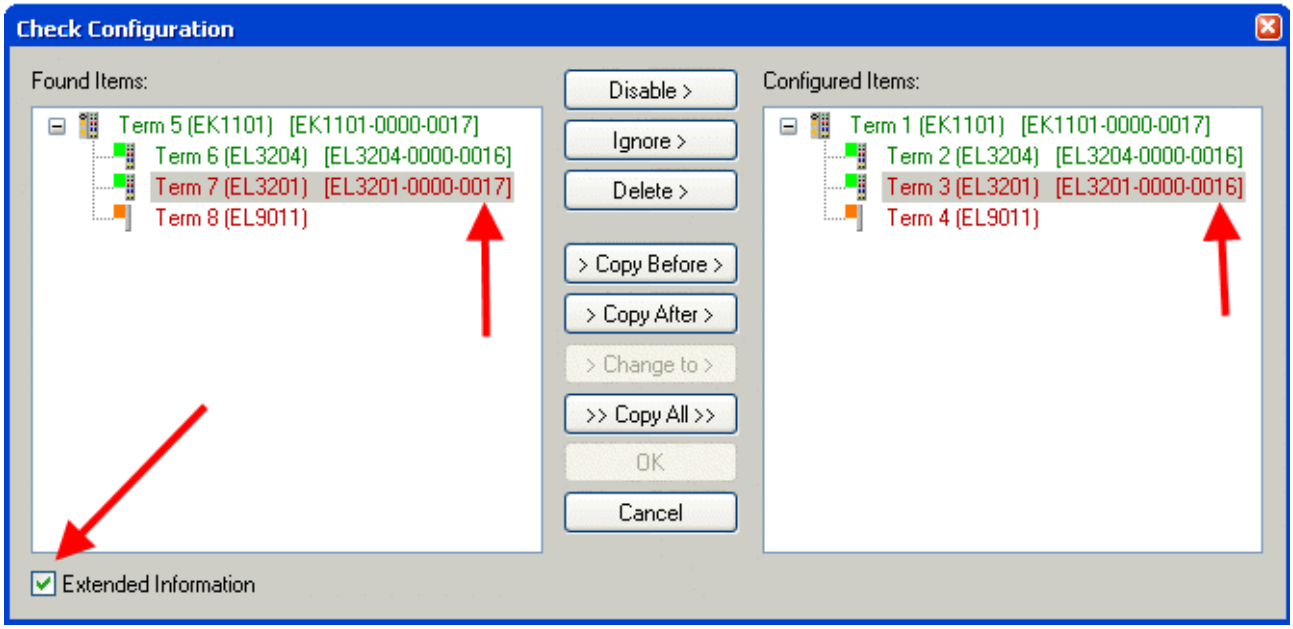


Fig. 82: Change dialog

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*

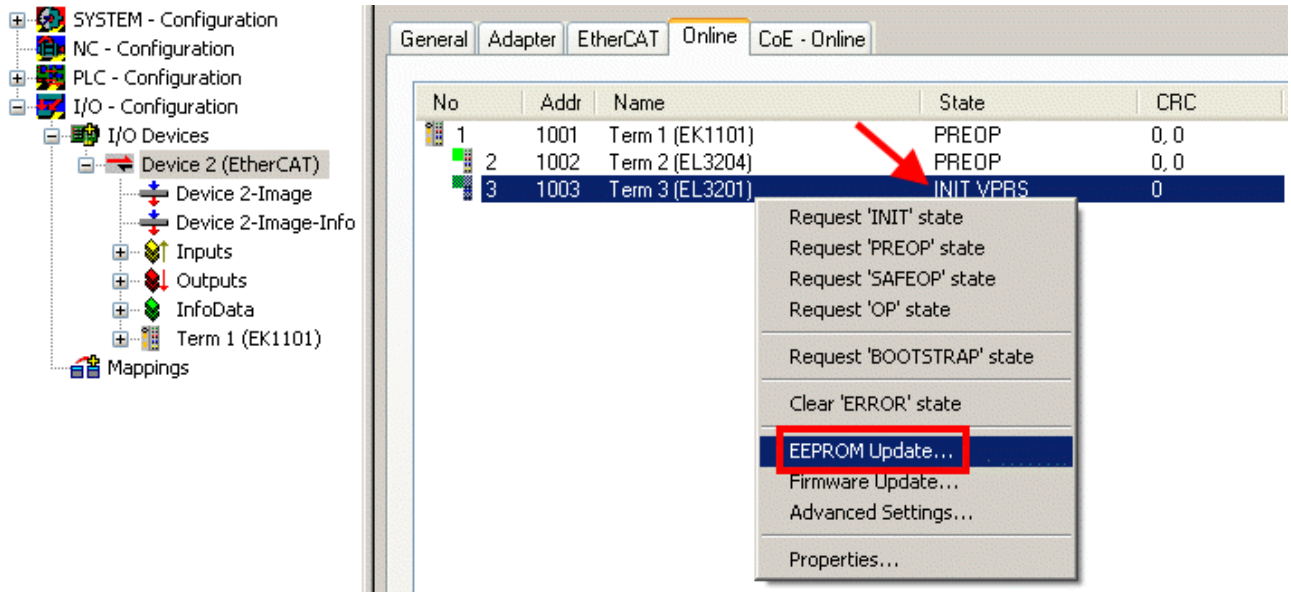


Fig. 83: EEPROM Update

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.

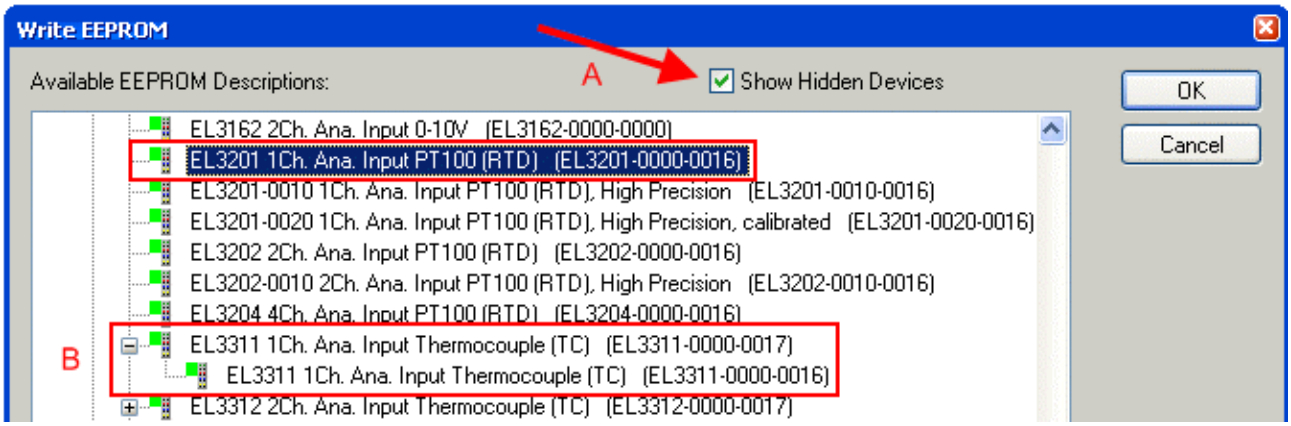


Fig. 84: Selecting the new ESI

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.

8.4.2 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.

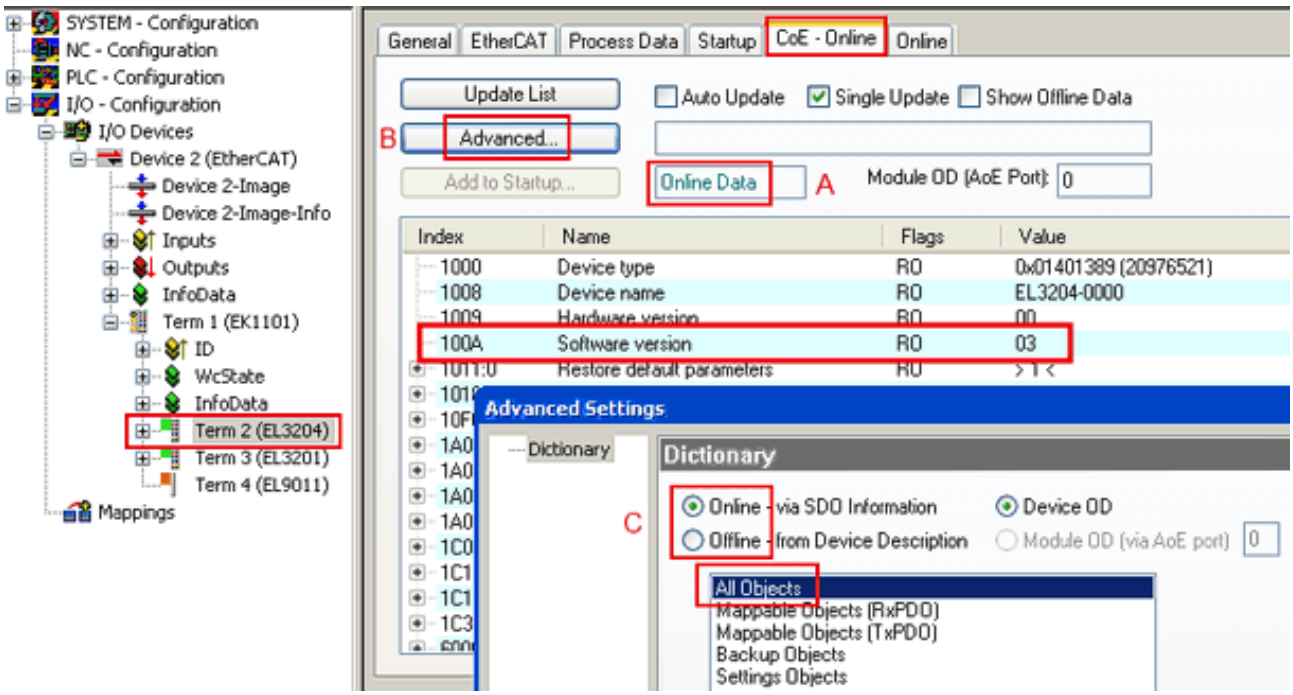


Fig. 85: Display of EL3204 firmware version

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*.

8.4.3 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.*

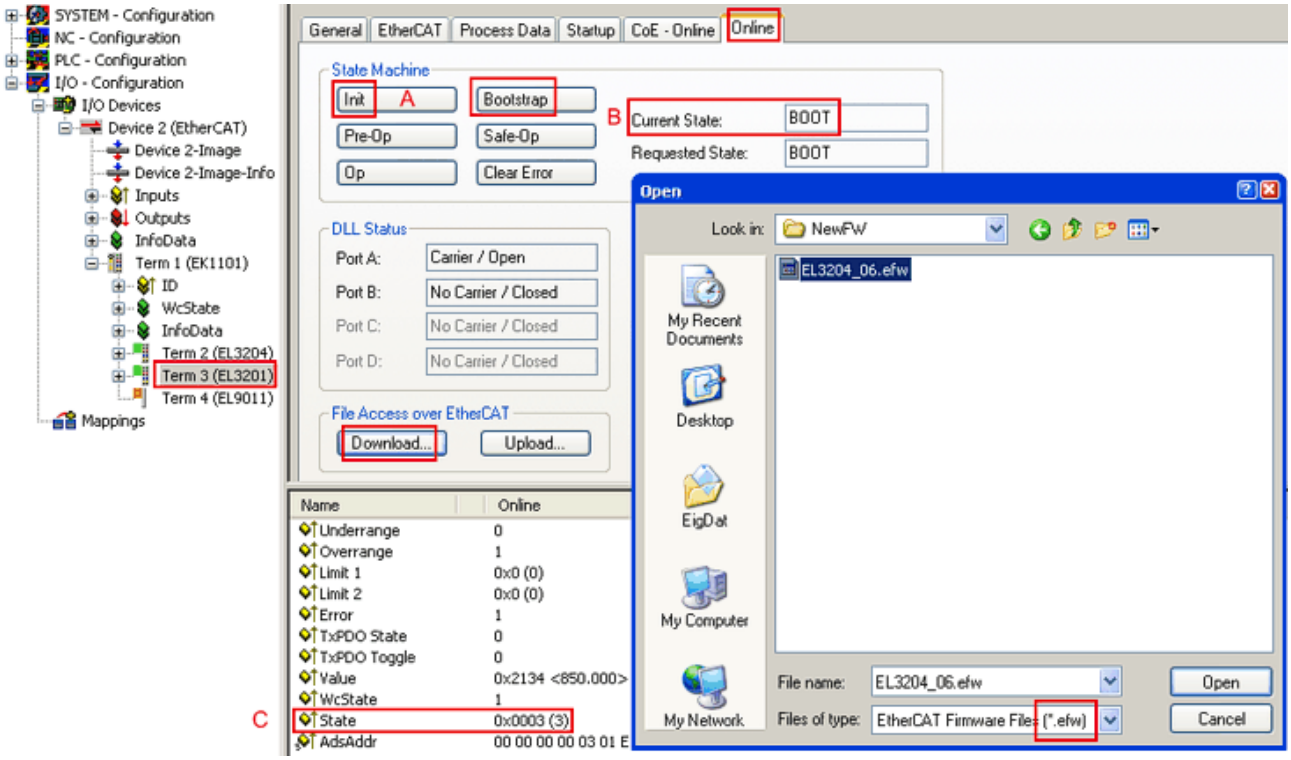
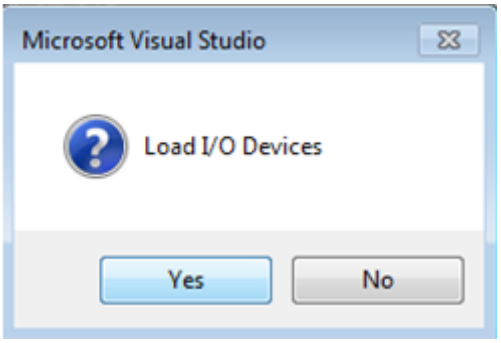
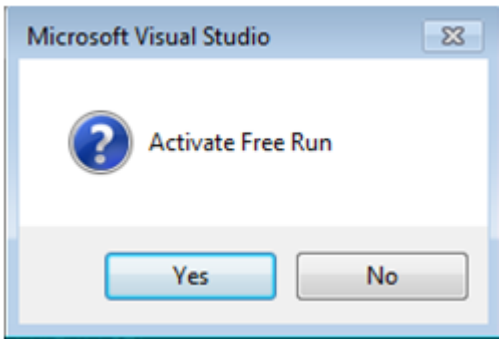


Fig. 86: Firmware Update

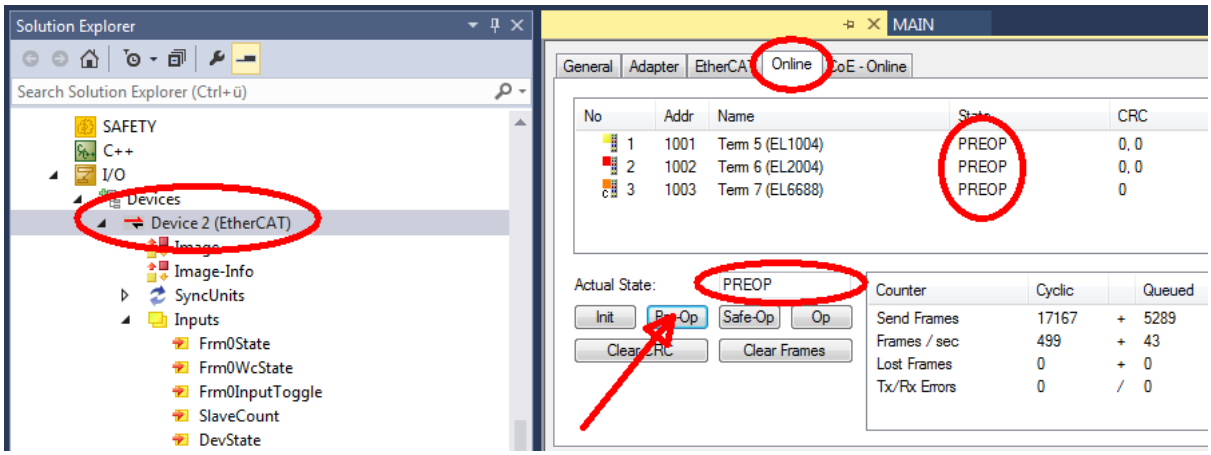
Proceed as follows, unless instructed otherwise by Beckhoff support. Valid for TwinCAT 2 and 3 as EtherCAT master.

- 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.

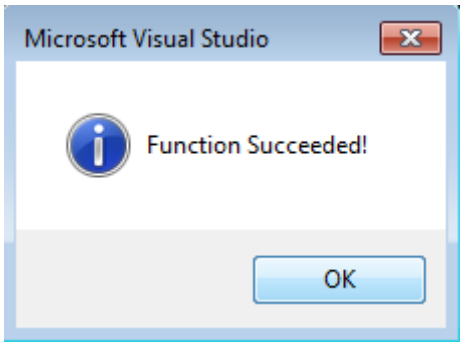




- Switch EtherCAT Master to PreOP



- 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.

8.4.4 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.

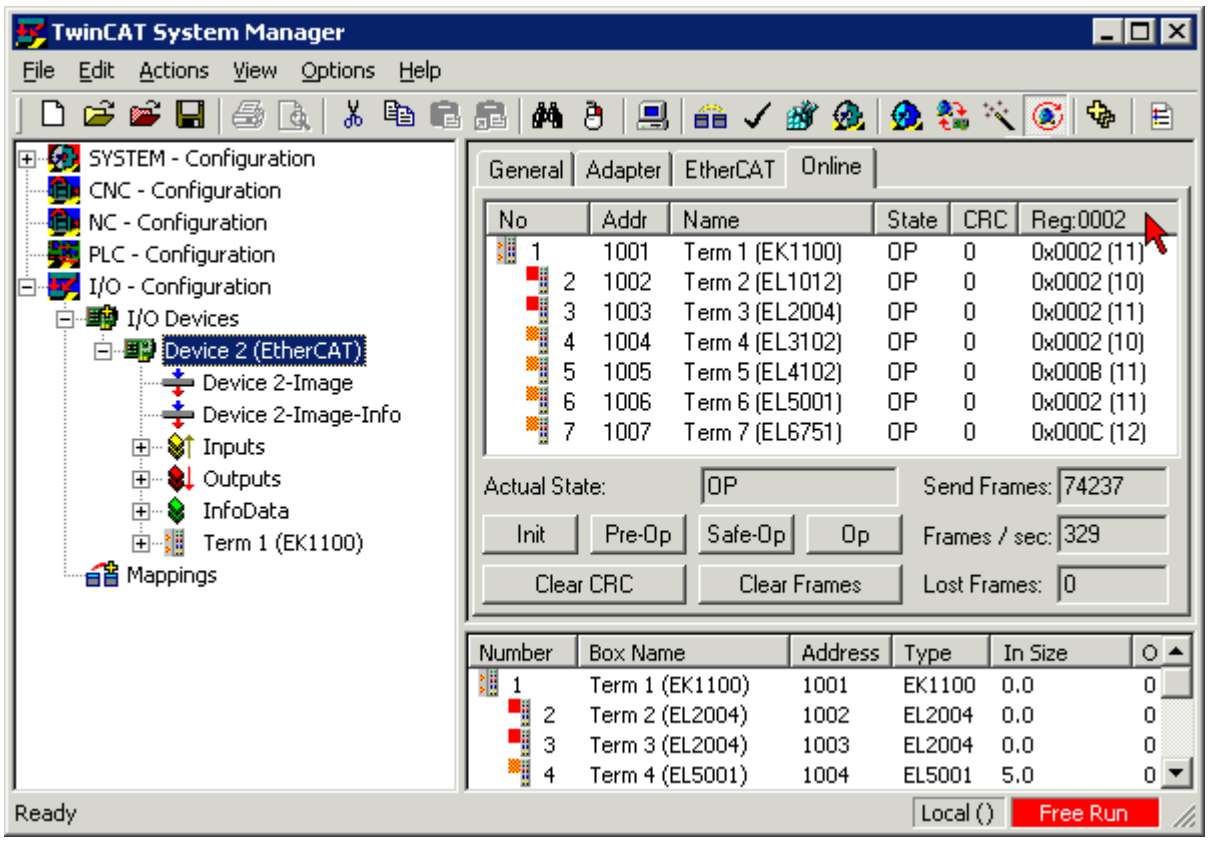


Fig. 87: FPGA firmware version definition

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

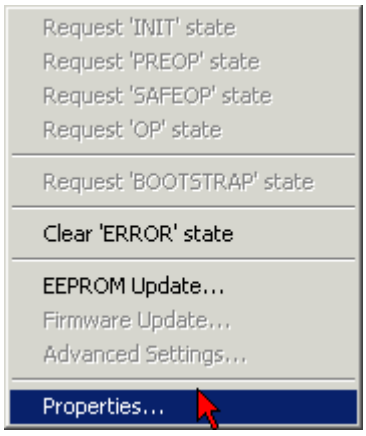


Fig. 88: Context menu *Properties*

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.

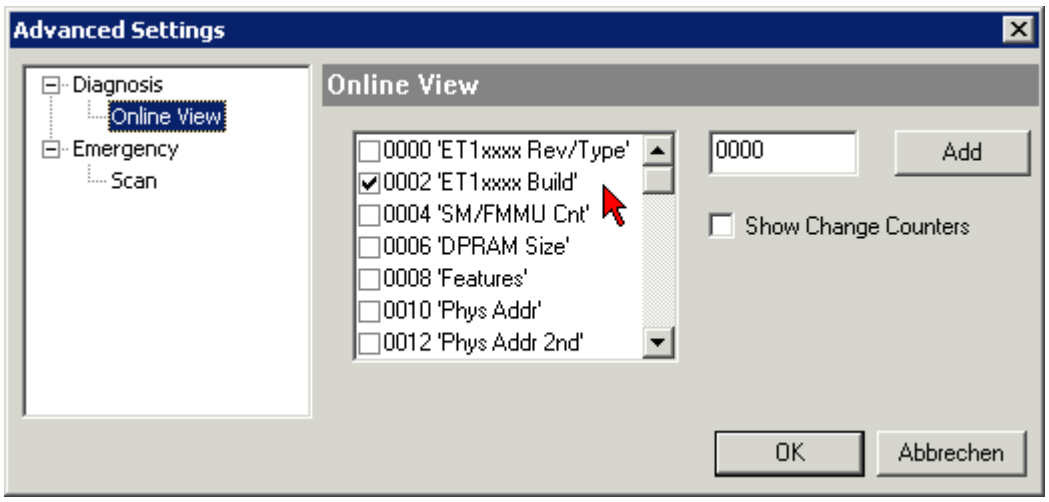


Fig. 89: Dialog *Advanced Settings*

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.

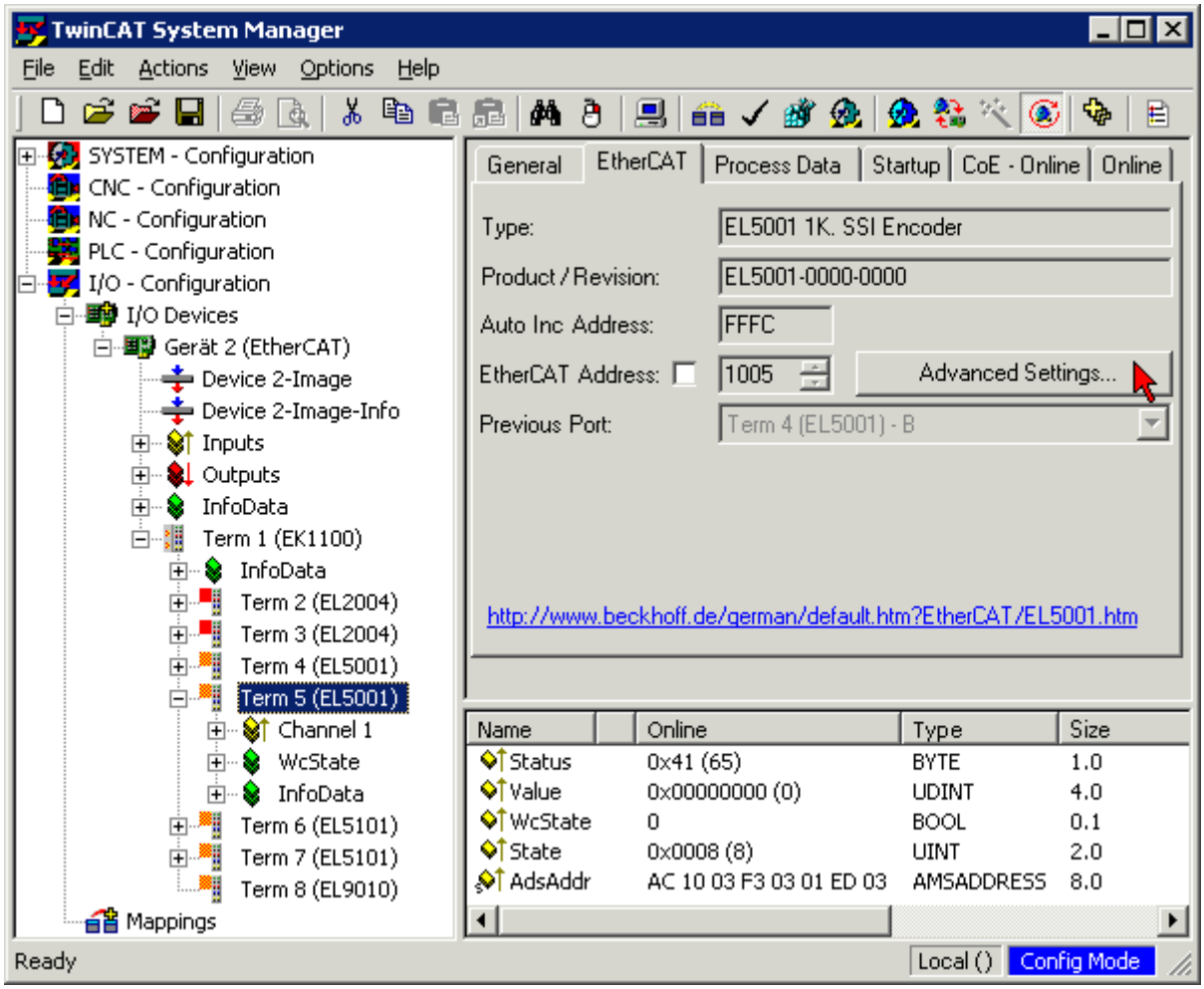
Older firmware versions can only be updated by the manufacturer!

Updating an EtherCAT device

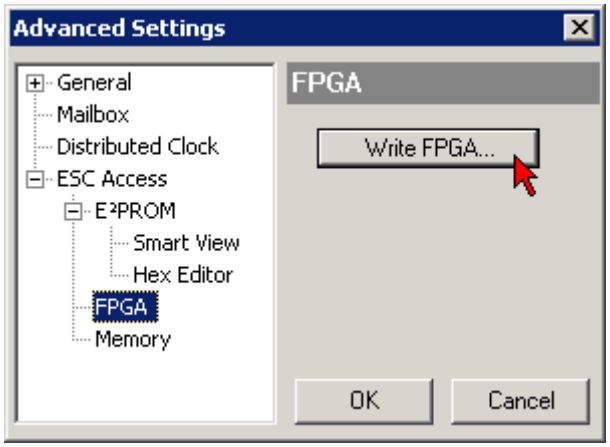
The following sequence order have to be met if no other specifications are given (e.g. by the Beckhoff support):

- 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.

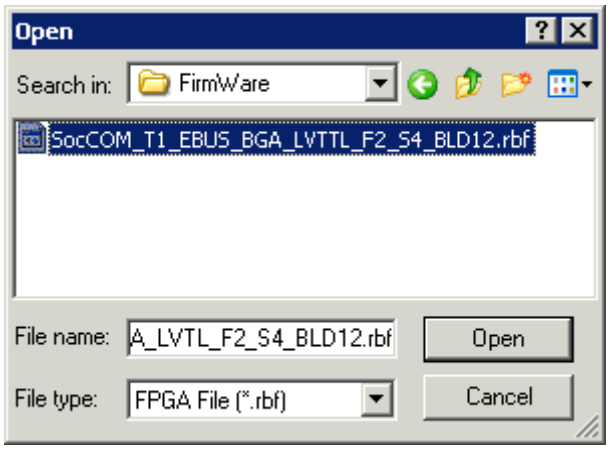
- In the TwinCAT System Manager select the terminal for which the FPGA firmware is to be updated (in the example: Terminal 5: EL5001) and
  click the *Advanced Settings* button in the *EtherCAT* tab:



- 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!

8.4.5 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.

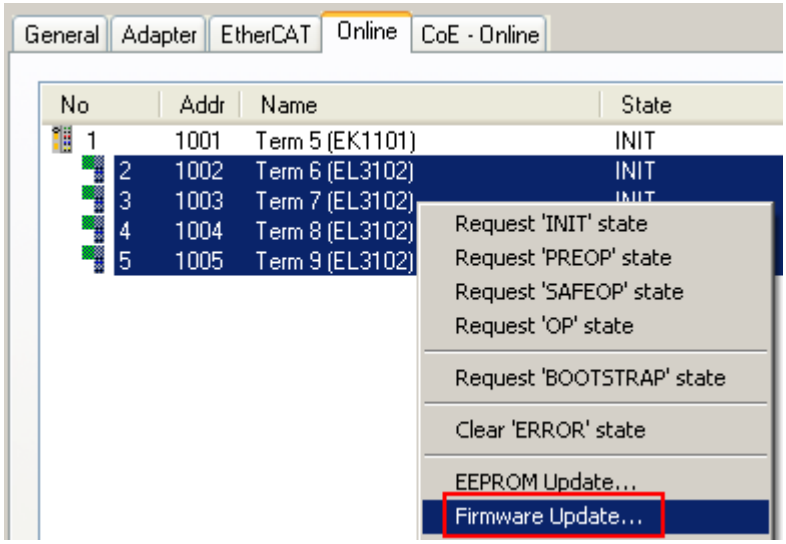


Fig. 90: Multiple selection and firmware update

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

8.5 ATEX Documentation

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

Pay also attention to the continuative documentation

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

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

8.6 Abbreviations

Abbreviation	Description
CAN	Controller Area Network. Serial bus system standardised in ISO 11898 that is used as the basic technology for CANopen
CiA	CAN in Automation e.V.. An international association of manufacturers and users based in Erlangen, Germany.
CoB	Communication Object. A CAN telegram with up to 8 data bytes.
CoB-ID	Communication Object Identifier. Telegram address (not to be confused with the node address). CANopen uses the 11-bit identifier according to CAN 2.0A.
CoE	CANopen over EtherCAT
ESC	EtherCAT Slave Controller
FBM	Fieldbus master
MC	Motion Control
NMT	Network Management. One of the service primitives of the CANopen specification. Network management is used to initialize the network and to monitor nodes.
OP	OPERATIONAL
PDO	Process Data Object. A CAN telegram for the transfer of process data (e.g. I/O data).
PREOP	PRE OPERATIONAL
RxPDO	Receive PDO. PDOs are always identified from the point of view of the device under consideration. Thus a TxPDO with input data from an I/O module becomes an RxPDO from the controller's point of view.
SAFEOP	SAFE OPERATIONAL
SDO	Service Data Object. A CAN telegram with a protocol for communication with data in the object directory (typically parameter data).
SI	Subindex
SM	SyncManager
SoE	Servo Profile over EtherCAT
TxPDO	Transmit PDO (named from the point of view of the CAN node).

8.7 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