BECKHOFF New Automation Technology

Documentation | EN

EL6633

PROFINET Controller

2025-08-21 | Version: 1.0.0

Table of contents

1 Foreword

1.1 Notes on the documentation

Intended audience

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

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

Disclaimer

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

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

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

Trademarks

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

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

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

Copyright

Third-party brands

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

1.2 Guide through documentation

NOTICE

Further components of documentation

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

Title	Description
EtherCAT System Documentation (PDF)	• System overview • EtherCAT basics • Cable redundancy • Hot Connect • EtherCAT devices configuration
Infrastructure for EtherCAT/Ethernet (PDF)	Technical recommendations and notes for design, implementation and testing
Software Declarations I/O (PDF)	Open source software declarations for Beckhoff I/O components

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

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

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

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

1.4 Safety instructions

Safety regulations

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

Exclusion of liability

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

Personnel qualification

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

Signal words

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

Personal injury warnings

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

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

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

Warning of damage to property or environment

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

Information on handling the product

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

1.5 Documentation issue status

Version	Comment
1.0.0	• Amendments • First release
0.3.0	• Amendments
0.2.0	• Amendments
0.1	• Preliminary version EL6633

1.6 Version identification of EtherCAT devices

1.6.1 General notes on marking

Designation

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

- family key
- type
- version
- revision

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

Notes

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

1.6.2 Version identification of EL terminals

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

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

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

1.6.3 Beckhoff Identification Code (BIC)

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

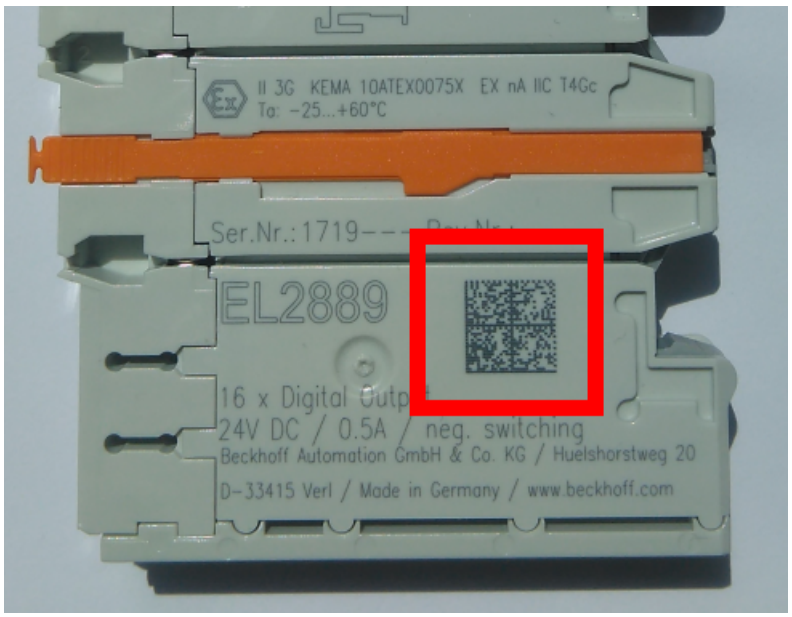


Fig. 2: BIC as data matrix code (DMC, code scheme ECC200)

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

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

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

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

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

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

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

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

Structure of the BIC

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

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

Accordingly as DMC:

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

BTN

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

NOTICE

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

1.6.4 Electronic access to the BIC (eBIC)

Electronic BIC (eBIC)

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

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

K-bus devices (IP20, IP67)

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

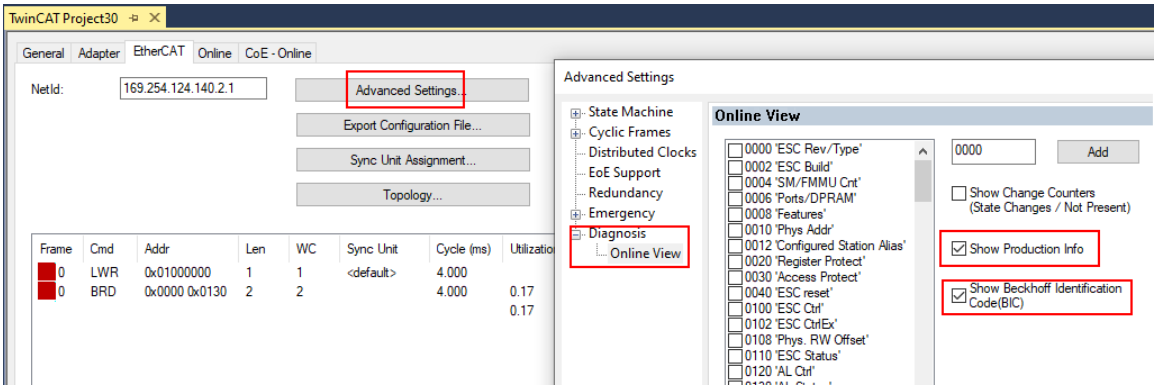
EtherCAT devices (IP20, IP67)

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

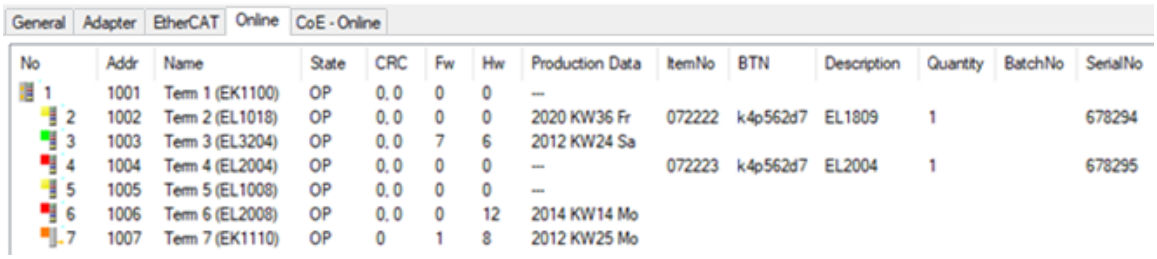
Beckhoff also stores the eBIC in the ESI-EEPROM. The eBIC was introduced into Beckhoff IO production (terminals, box modules) in 2020; as of 2023, implementation is largely complete.

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

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

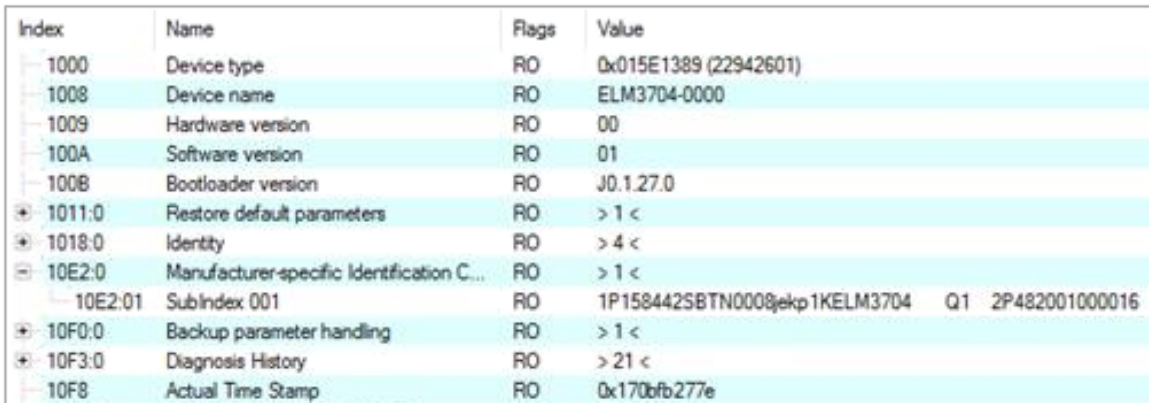


  - The BTN and its contents are then displayed:



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

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



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

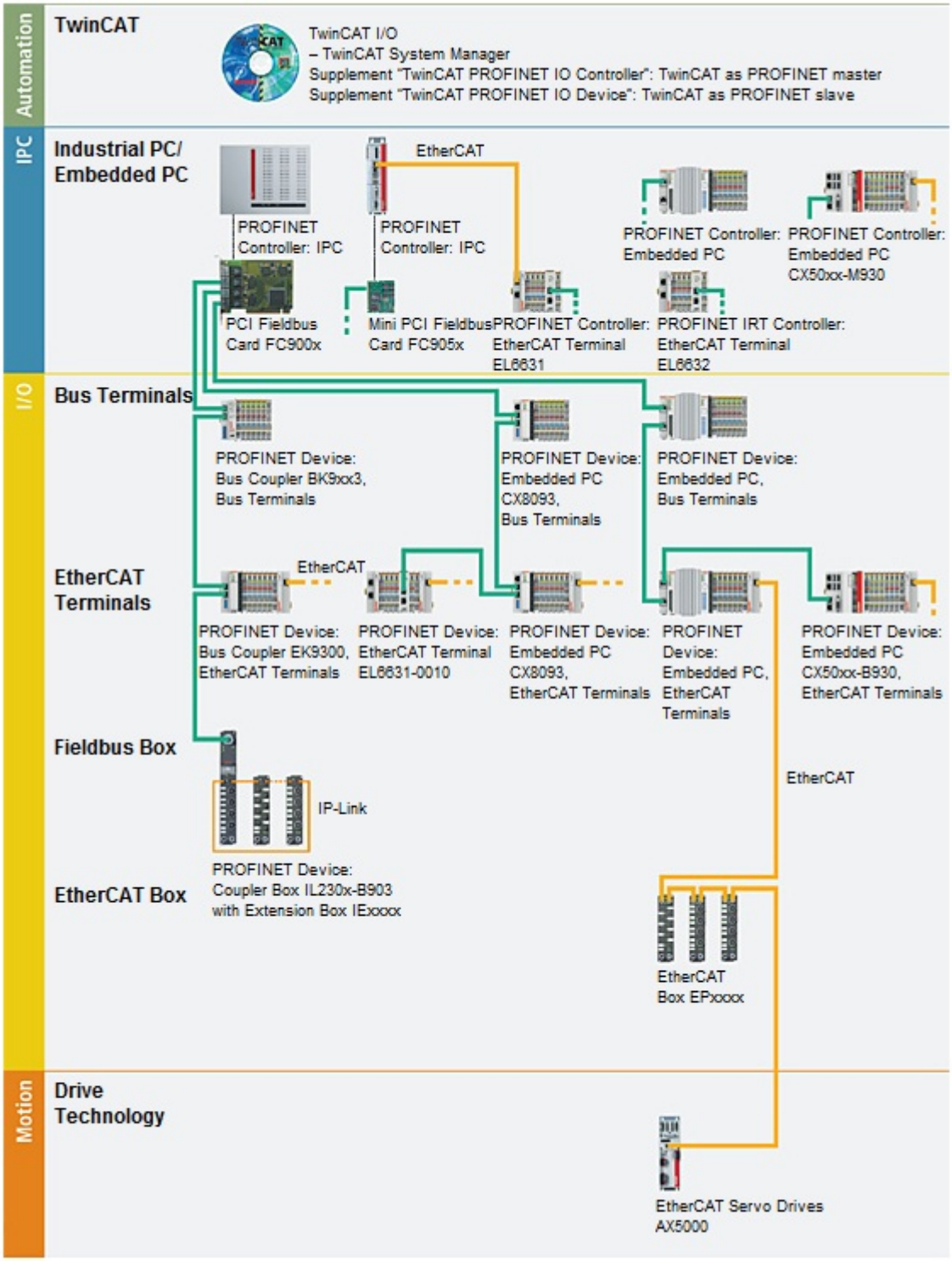
  - The object 0x10E2 will be preferentially introduced into stock products in the course of necessary firmware revision.
  - From TwinCAT 3.1. build 4024.24, the functions *FB_EcCoEReadBIC* and *FB_EcCoEReadBTN* for reading into the PLC are available in the Tc2_EtherCAT library from v3.3.19.0
- The following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

PROFIBUS, PROFINET, and DeviceNet devices

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

2 PROFINET system presentation

PROFINET is the Industrial Ethernet standard of the PNO (PROFIBUS user organization). Internationally established IT standards such as TCP/IP are used for communication.



PROFINET system description

PROFINET IO describes the data exchange between controllers and field devices in several real-time classes: RT (software-based Real-Time) and IRT (hardware-based Isochronous Real-Time). In addition, further Ethernet traffic can be transmitted in the NRT (non-real-time) time slot of the PROFINET cycle. PROFINET RT can be operated with commercially available switches, but in this case topology detection is not supported. For better diagnostics and automatic name assignment (alias name), commercially available switches should be used that support at least topology detection; they are referred to as LLDP. All PROFINET devices from Beckhoff (with or without switch) have this feature.

Beckhoff PROFINET components

Components	Comment
TwinCAT	
TwinCAT PROFINET IO Controller	TwinCAT as PROFINET master
Embedded PCs	
CXxxxx-M930	Embedded PC with optional interface PROFINET RT Controller
EtherCAT Terminals	
EL6631	PROFINET IO controller
EL6632	PROFINET-IRT controller
EL6633	PROFINET IO controller
PC Fieldbus cards	
FC90xx	PCI Ethernet card for all Ethernet-based protocols (IEEE 802.3)
FC9x51	Mini PCI Ethernet card for all Ethernet-based protocols (IEEE 802.3)

3 Product description

3.1 General technical data

Technical Data	RT Ethernet (TF6271)	Optional interface M930	EtherCAT (EL6631)	EtherCAT (EL6632)	EtherCAT (EL6633)
Ethernet hardware	RT Ethernet hardware	RT Ethernet hardware	EL6631 PROFINET terminal	EL6632 PROFINET terminal	EL6633 PROFINET terminal
Cycle time	min.1 ms			min. 500 µs	min.1 ms
Number of possible IO devices	Limited by CPU power and memory		up to 15 devices	for IRT up to 5 devices, for RT up to 15 devices	up to 15 devices
Max. number of process data	depending on cycle time, number and slave type, etc.		1 kB input data and 1 kB output data		1.4 kB input data and 1.4 kB output data
I-Device	no				yes
Controller/Device	yes/yes	yes/no	yes/no	yes/no	yes/yes
Switch	depending on Ethernet interface	100 Mbit/s			10/100/1000 Mbit/s
Requirement	TwinCAT 2				TwinCAT 3.1 Build 4024

3.2 Introduction

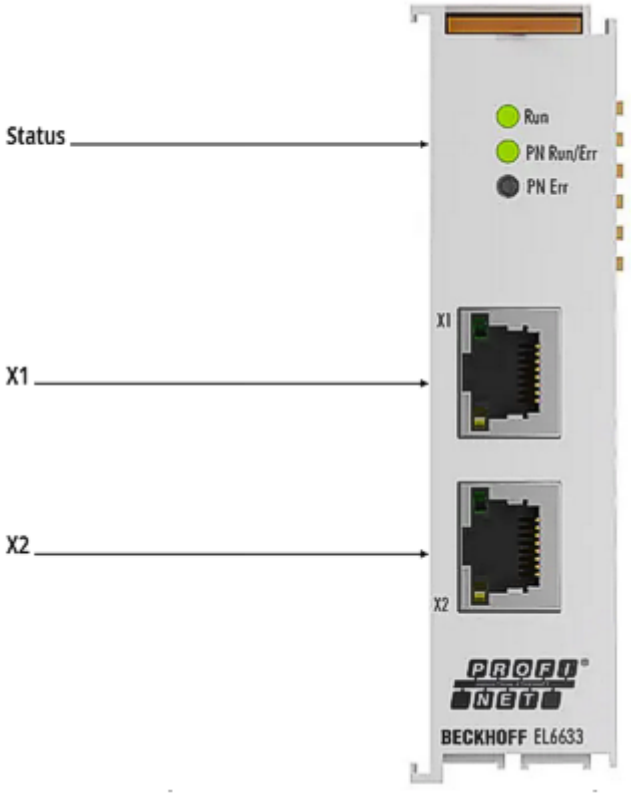


Fig. 4: EL6633

2-port communication interface, PROFINET RT, controller/i-device

The EL6633 PROFINET RT controller terminal is the successor to the EL6631 and supports full real-time (RT) functionality as well as extensive diagnostic options. All services in accordance with Conformance Class B are supported. Up to 15 PROFINET RT devices can be projected on the EL6633.

Protocols such as LLDP or SNMP can be used for network diagnostics.

Supplement

The TwinCAT Supplement is not required for the EL6633.

TwinCAT version

The released TwinCAT version is TwinCAT 2.11 R3.
It must be ensured that the target system also corresponds to the TwinCAT version.
Older TwinCAT versions cannot be used!

3.3 EL6633 – Technical data

Technical data	EL6633
Technology	PROFINET RT
Number of ports/channels	2 (switched)
Fieldbus	PROFINET RT Controller and/or Device
Max. process data size	1.4 kB input/1.4 kB output data
Ethernet interface	10/100/1000BASE-TX Ethernet with 2 x RJ45
Cable length	up to 100 m twisted pair
Hardware diagnostics	Status LEDs
Power supply	via the E-bus
Electrical isolation	500 V (E-bus/Ethernet)
Protocol	RT
Configuration	via the EtherCAT master
Current consumption via E-bus	400 mA typ.
Special features	Conformity class B, supported service protocols RPC and RSI
Maximum number of process data	1 kB input data and 1 kB output data
Maximum number of submodules per PROFINET device	238
Configuration	via the EtherCAT master
Installation [▶ 29]	on 35 mm mounting rail, conforms to EN 60715
Weight	approx. 75 g
Permissible ambient temperature during operation**)	0…+55 °C
Permissible ambient temperature during storage	-25…+85 °C
Permissible relative humidity	95% no condensation
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
Installation position	Standard installation position (an operating temperature of 45 °C applies to other installation positions), see also notice [▶ 26]
Protection rating	IP20
Approvals/markings*)	CE, cULus [▶ 24]

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

**) If another terminal with high power loss (e.g. E-bus current >250 mA) is present next to the terminal, an EL9xx0 power supply terminal or separation terminal must be connected between them (recommendation: terminal with E-bus ASIC).

3.4 EL6633 - LEDs

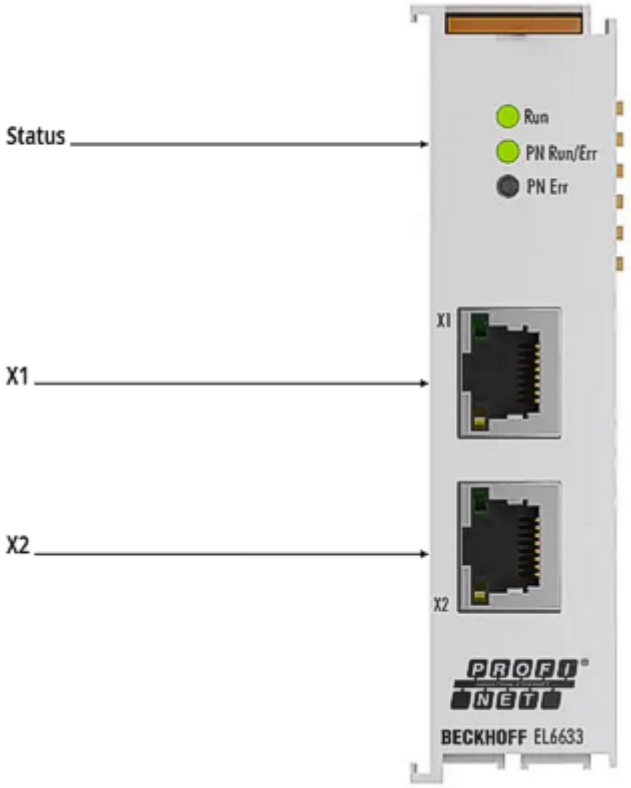


Fig. 5: EL6633 - LEDs

LEDs for EtherCAT diagnosis

LED		Display	Description
RUN	green	off	State of the EtherCAT State Machine: **INIT** = initialization of the terminal; **BOOTSTRAP** = function for firmware updates of the terminal
		flashing 200 ms	State of the EtherCAT State Machine: **PREOP** = function for mailbox communication and different standard-settings set
		off (1 s) on (200 ms)	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

LED diagnosis PROFINET RUN/Err

Color green	Color red	Meaning
off	flashing 200 ms	Terminal starts
flashing 200 ms	off	No name
1 s off, 200 ms on	off	No IP address
on	off	EL terminal is parameterized

LED diagnosis PROFINET Err

Colours green	Colours red	Meaning
off	flashing 500 ms	no AR established with any device
off	1 s off, 200 ms on	At least one device has not established an AR
1 s off, 200 ms on	off	At least one device has signaled an error, e.g. that there is a module difference or that the error bit for an IO-CR is set (Problem Indicator)
flashing 200 ms	off	At least one device is signaling that its status is ‘Stop’ (Provider State Stop)
on	off	All PROFINET devices are in data exchange mode

If there are several different errors, then the error that is located at the top of (or higher in) the table is always displayed.

LEDs starting up

Run	PN Run/Err	PN Err	Meaning
off	off	off	No electrical voltage connected to E-bus. The EL6631 must be exchanged if EtherCAT terminals behind it function.
off	off	red on	EL terminal is starting up; after approx. 10 seconds, the LED should go out. If this does not happen, the EL6631 module must be exchanged.

4 Mounting and wiring

4.1 Instructions for ESD protection

NOTICE

Destruction of the devices by electrostatic discharge possible!

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

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

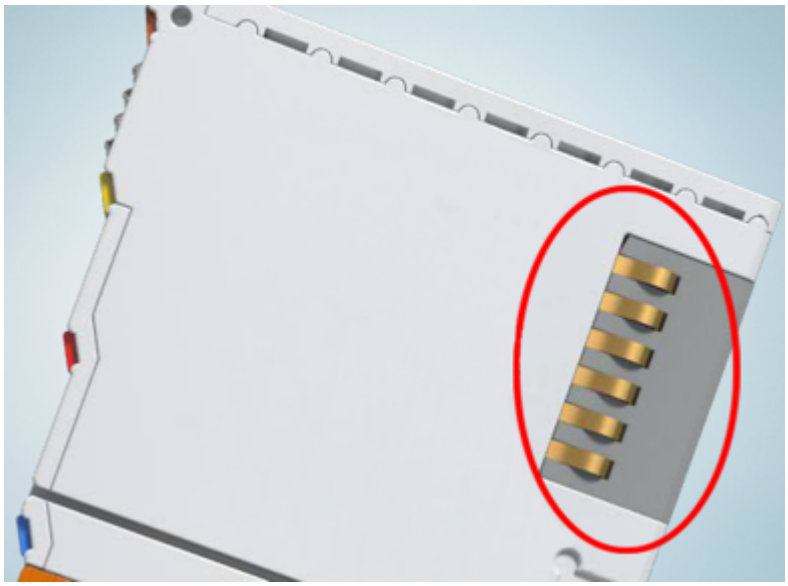


Fig. 6: Spring contacts of the Beckhoff I/O components

4.2 UL notice

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

⚠ CAUTION	
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).

⚠ CAUTION	
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:

4.3 Note - power supply

⚠ WARNING

Power supply from SELV / PELV power supply unit!

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

Notes:

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

4.4 Installation positions

NOTICE

Constraints regarding installation position and operating temperature range

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

Optimum installation position (standard)

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

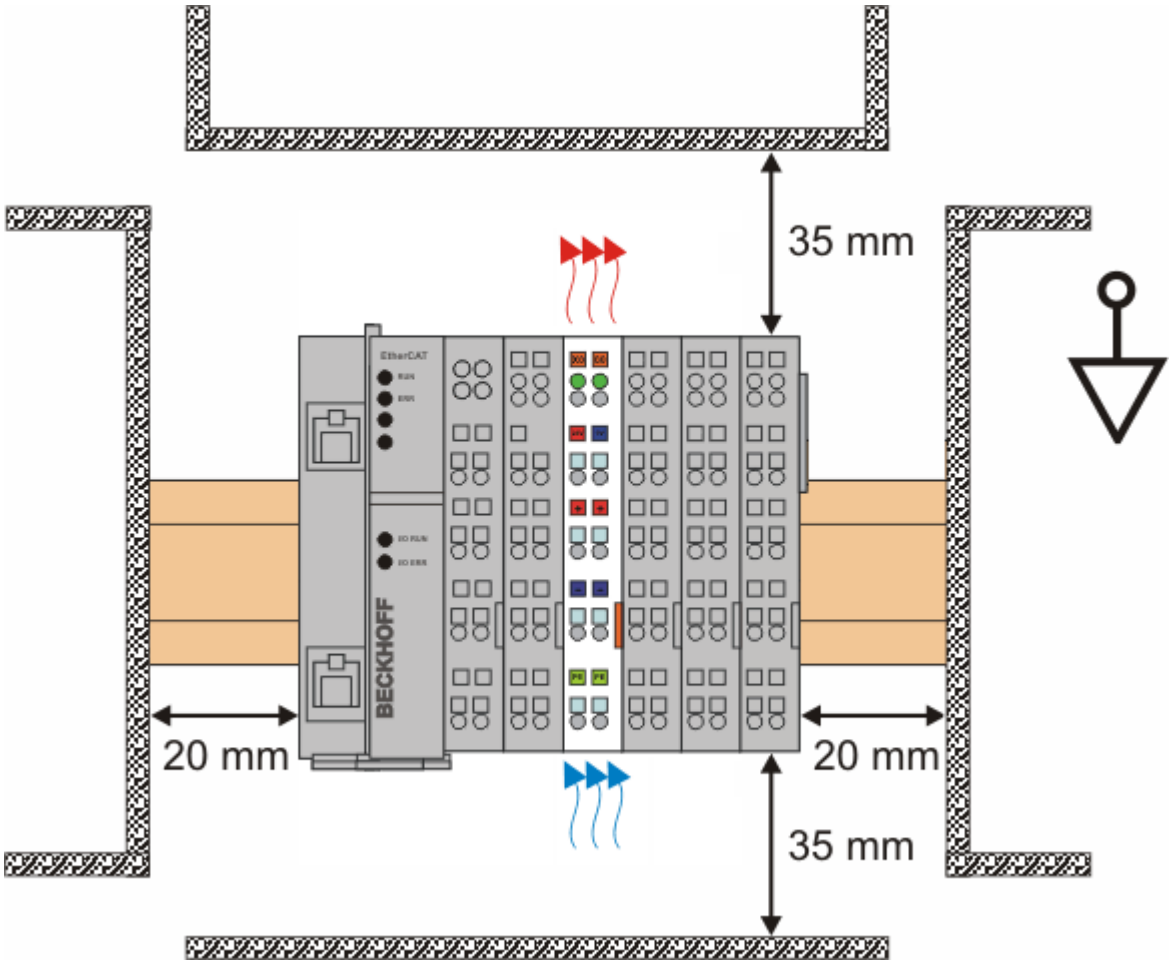


Fig. 7: Recommended distances for standard installation position

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.

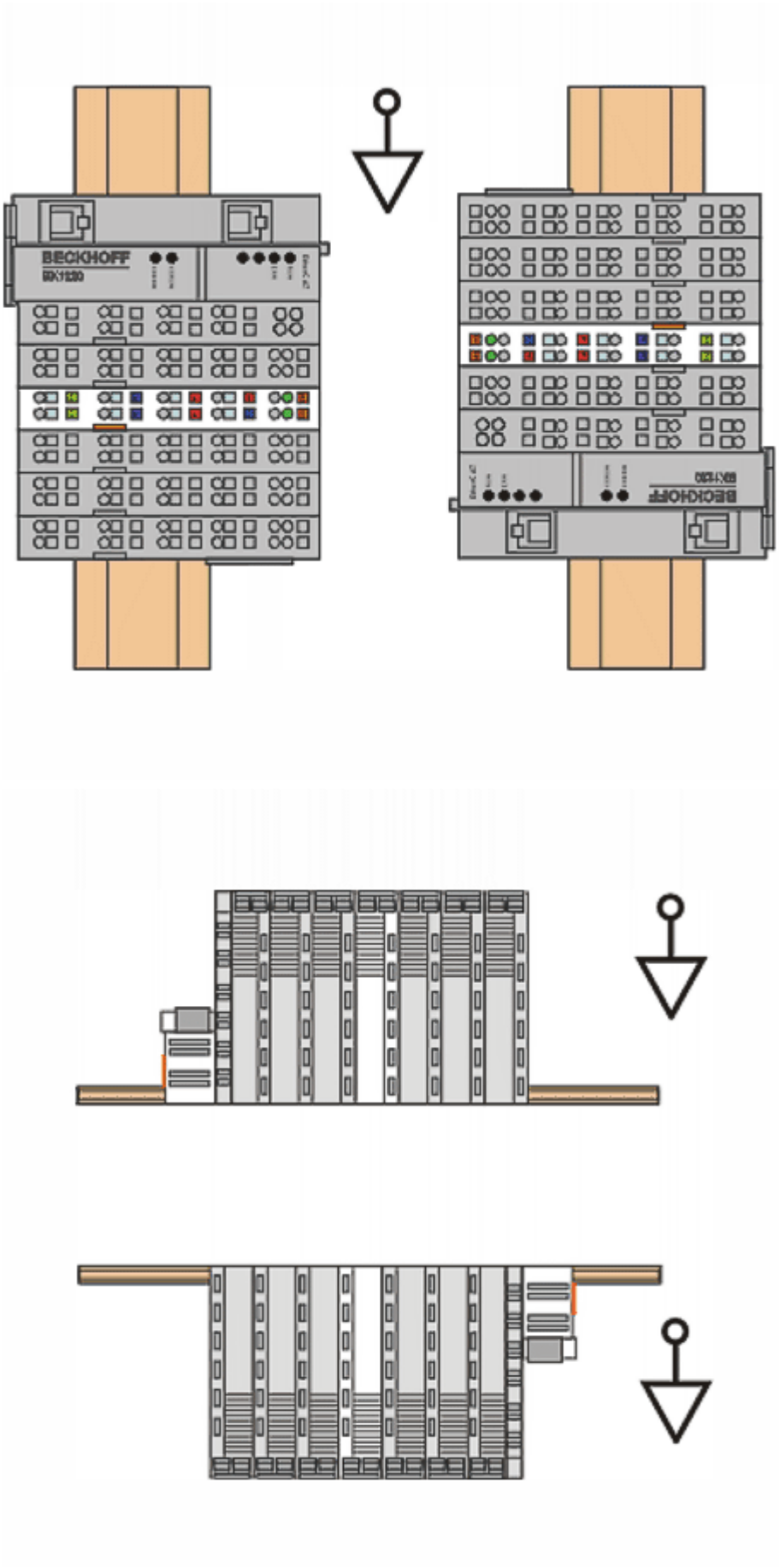


Fig. 8: Other installation positions

4.5 Note on shielding

NOTICE

Low-resistance shielding through external shield connection

The shielding of the EL66xx is capacitively connected to the mounting rail.

If low-resistance shielding is desired or required, the shield must be connected to an external shield support (e.g. with the clamp strap for shield connection ZB5800).

4.6 Mounting and demounting - top front unlocking

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

Fixing of mounting rails

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

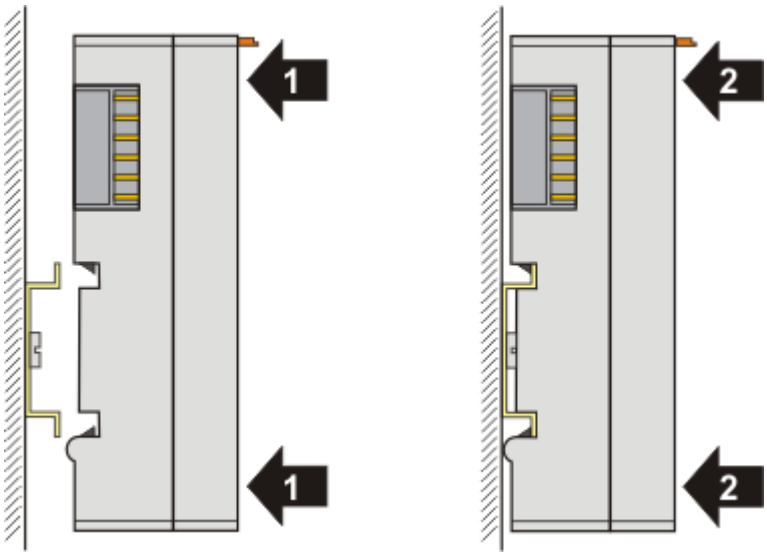
⚠ WARNING

Risk of electric shock and damage of device!

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

Mounting

- Fit the mounting rail to the planned assembly location.

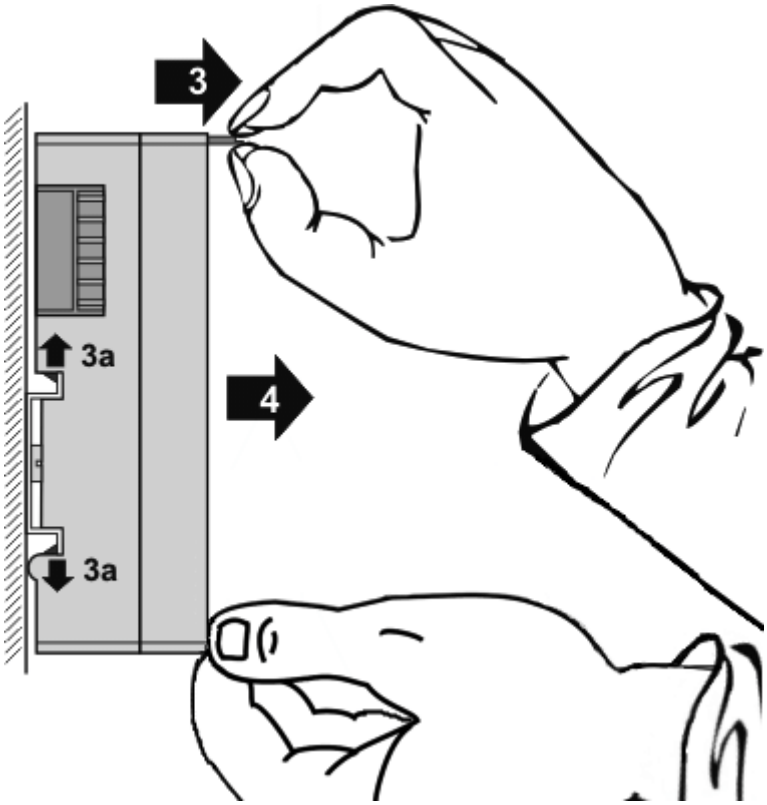


and press (1) the terminal module against the mounting rail until it latches in place on the mounting rail (2).

- Attach the cables.

Demounting

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



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

4.7 Disposal

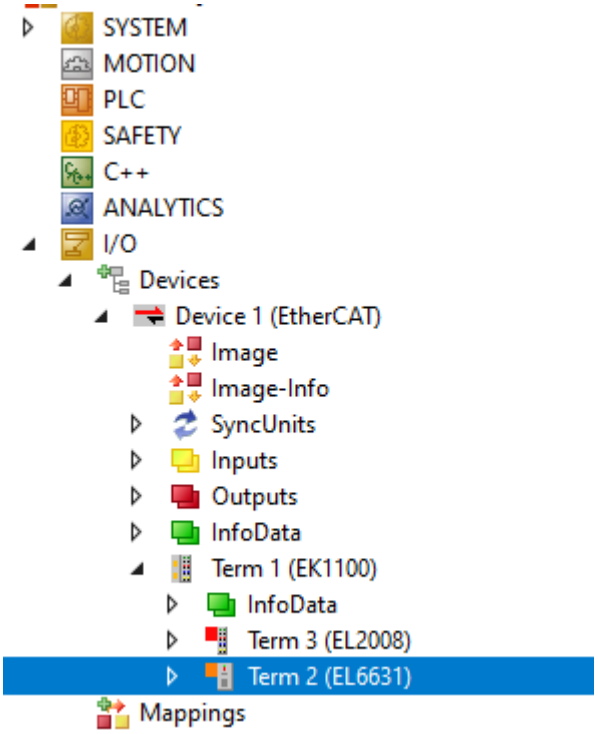
Products marked with a crossed-out wheeled bin shall not be discarded with the normal waste stream. The device is considered as waste electrical and electronic equipment. The national regulations for the disposal of waste electrical and electronic equipment must be observed.

5 Commissioning & Configuration

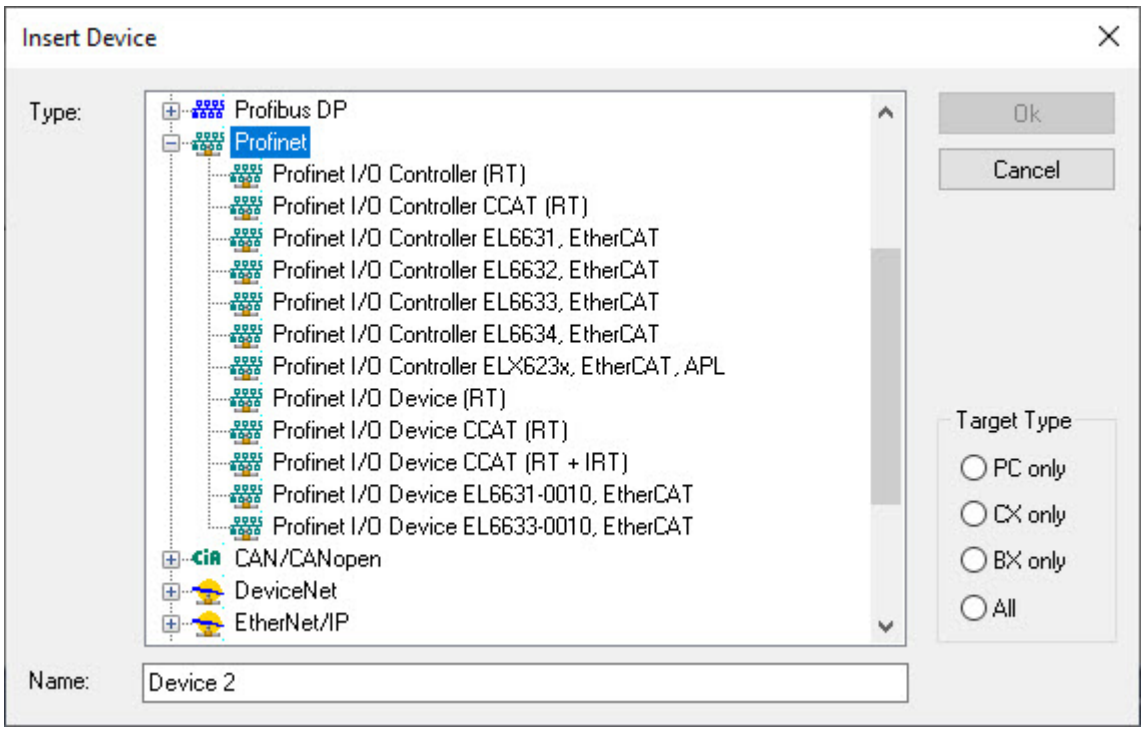
5.1 Integration via an EL663x

The following section shows how a PROFINET Controller protocol is integrated using the EL663x PROFINET gateway terminal.

The EL663x is primarily an EtherCAT slave, which means that it must be included in the TwinCAT configuration, since it serves as an adapter for the PROFINET Controller.



The controller protocol is attached directly by right-clicking on the I/O device. Select the appropriate protocol according to the configuration with the EL663x. If such a terminal is located on the projected EtherCAT segment, the associated adapter is entered directly when the protocol is added. For the operation of several EL663x terminals the corresponding PROFINET protocol must be appended several times. If the terminal assignment is to be modified or checked afterwards, this can take place on the *Adapter* tab.

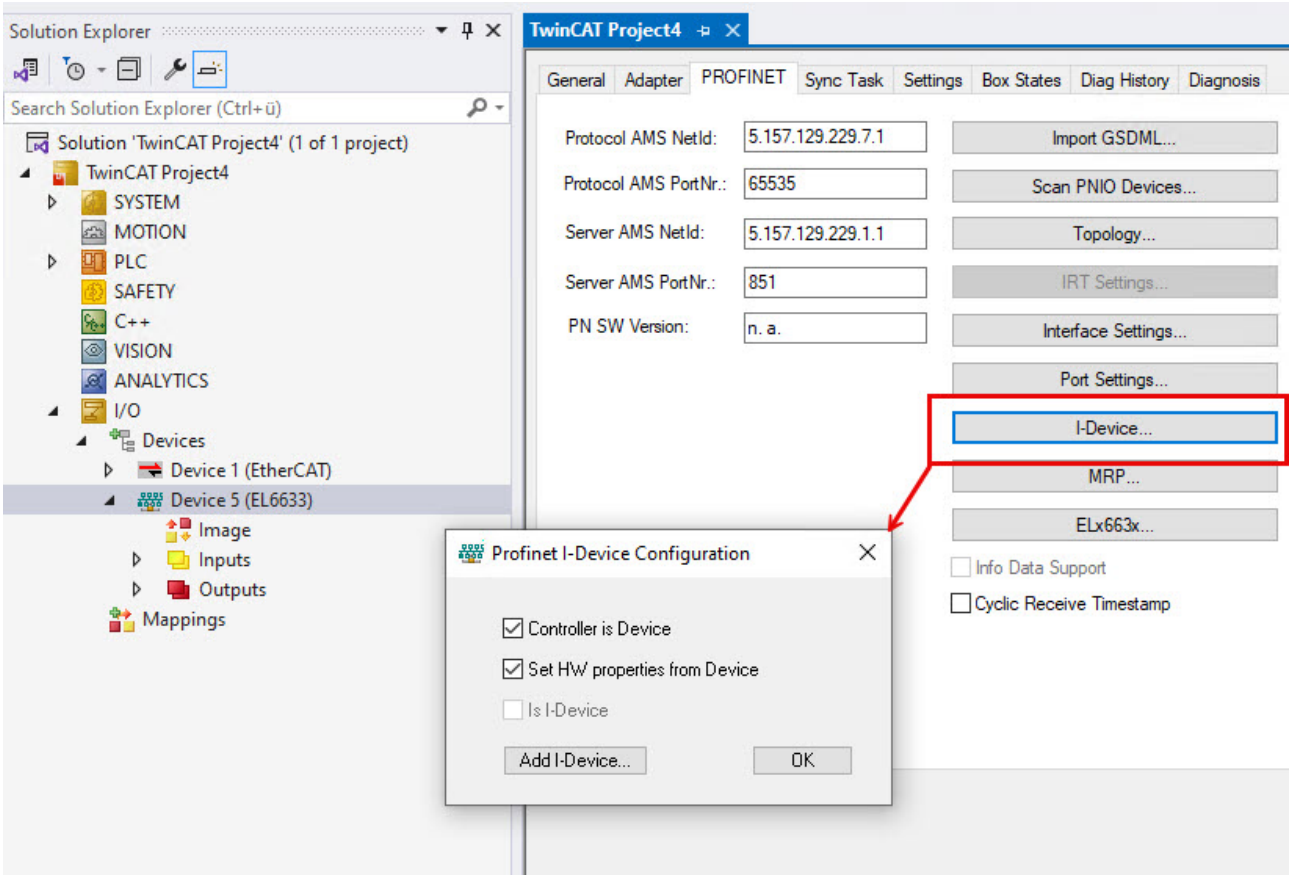


Further information in the chapter: Appending PROFINET devices [▶ 36]

5.1.1 Creating an I-Device

Using the I-Device functionality, a controller and a device interface can be operated simultaneously via a physical interface. The requirements for this are the EL6633 PROFINET RT Controller terminal and at least TwinCAT version 4026.12

The "I-Device" button on the "PROFINET" tab opens a "Profinet I-Device Configuration" dialog box in which the I-Device functionality can be configured and activated.



Controller is Device:

This activates the controller to also function as a device at the same time.

Set HW properties from Device:

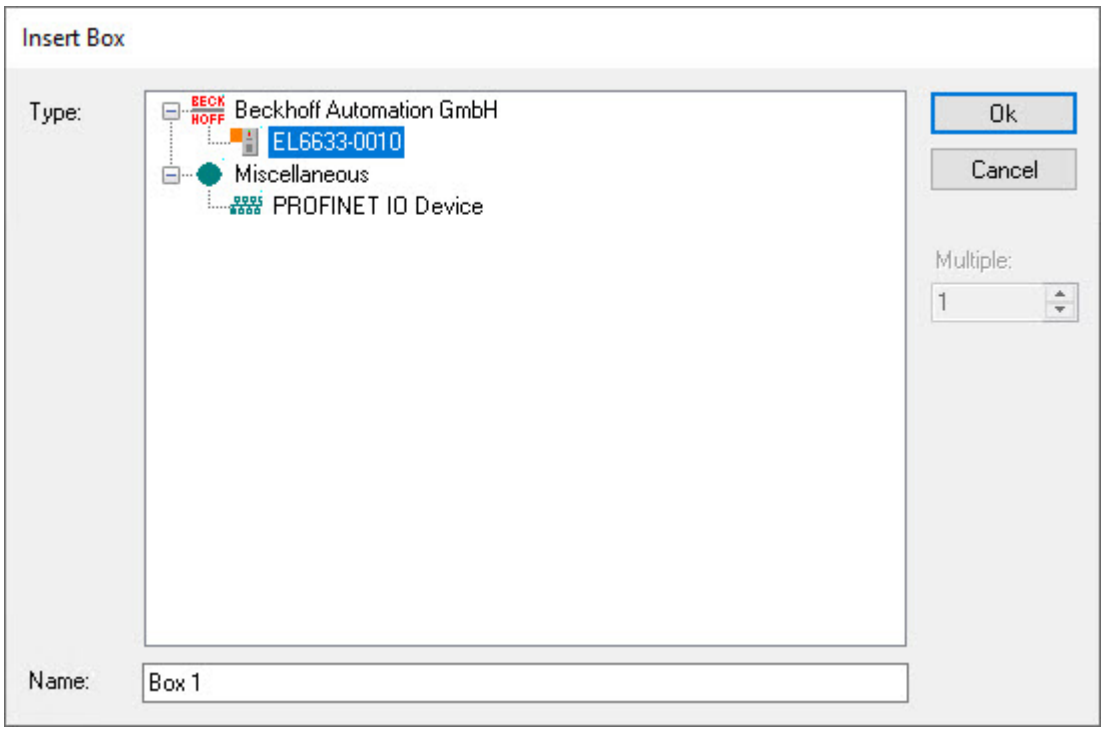
Here you can optionally define the hardware properties, e.g. what is signaled via the fieldbus LED. By default this is the controller diagnostics, although it is possible to change to device. This option also affects the port properties (e.g. LLDP). When enabled, the higher-level controller can parameterize the properties of the physical interface.

Is I-Device:

Only available in connection with the optional CCAT -M930 interface.

Add I-Device:

Opens a dialog for selecting the I-Device to be attached.


Insert Box
Type:
Beckhoff Automation GmbH
EL6633-0010
Miscellaneous
PROFINET IO Device
Ok
Cancel
Multiple:
1
Name:
Box 1

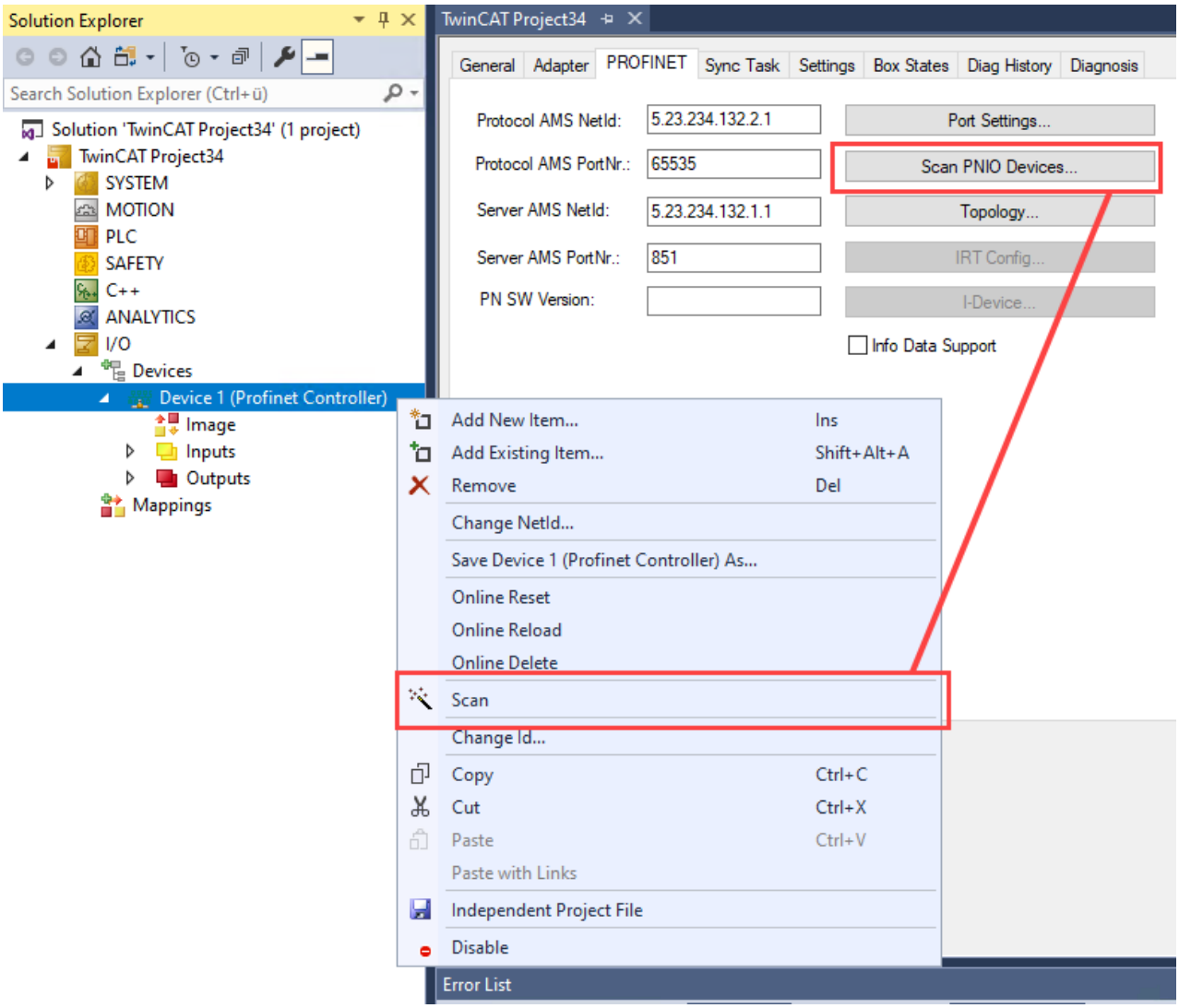
5.2 Configuration

5.2.1 Appending PROFINET devices

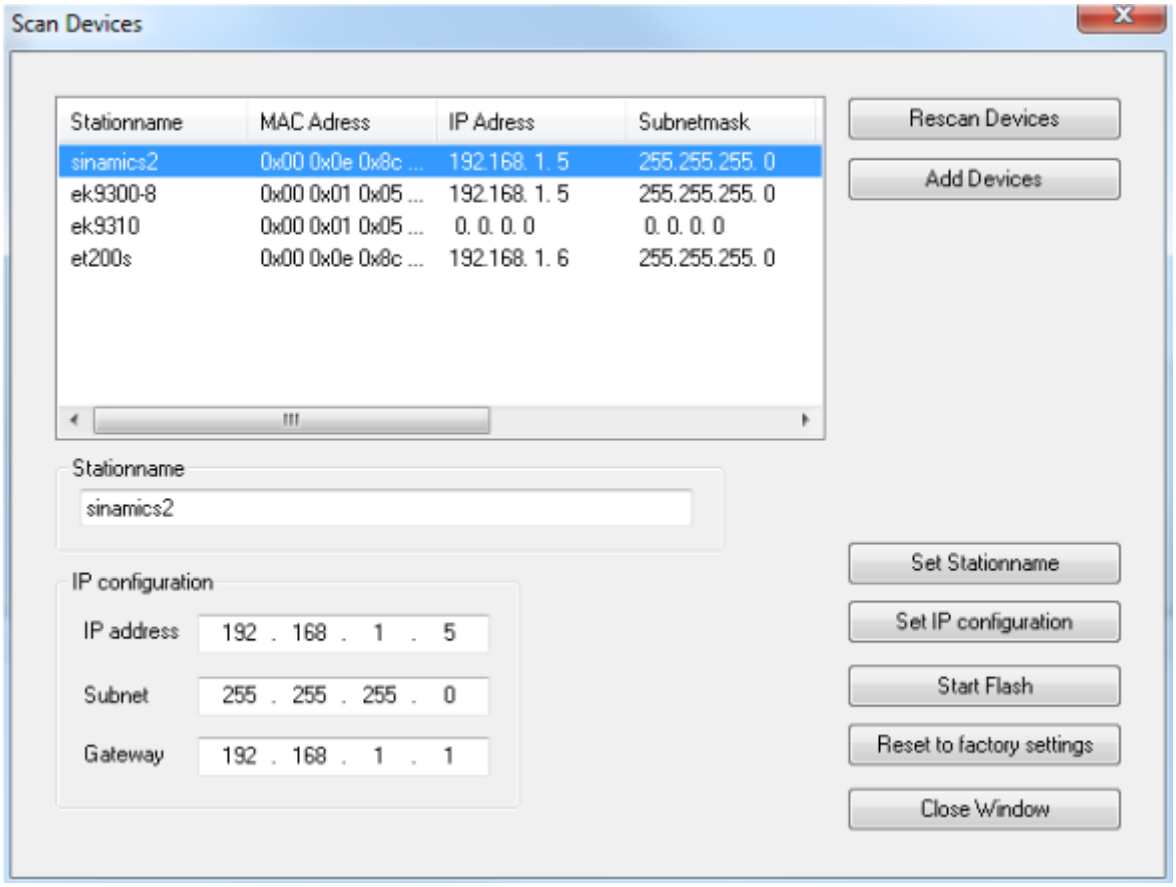
Once the PROFINET Controller has been created, there are two options for adding new PROFINET devices.

5.2.1.1 Appending via the Scan function

The first and recommended option is to use **Scan PNIO Devices**. This feature is comparable with the "ScanBoxes" feature which, however, is available only in CONFIG mode. After successful scanning, if PROFINET devices were found a dialog opens in which settings and configuration can be made on the devices and devices with module configuration can be added.

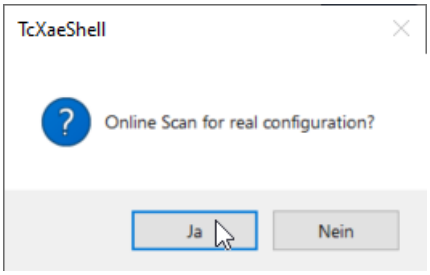


The prerequisite for finding PROFITNET devices on the bus is that they are present and switched on and that the GSDML file of the respective devices is located in the path *C:\TwinCAT\3.1\Config\Io\Profinet*.



Various settings or project planning can be carried out for the devices here. These are adopted only when the corresponding button is explicitly pressed. When setting the name, care must be taken that only PROFINET-compliant characters are used. This also applies to the IP address; only valid combinations of IP and subnet are to be used. Name and IP are checked for correctness when setting PROFINET devices. If this is not the case, DCP_SET is acknowledged with an error. By pressing the **Rescan** button, any changes that were made can be read back. In addition the selected device can be signaled. This functionality is PROFINET-specific. The signaling is vendor-specific. As standard, however, the signal must arrive with a frequency of 2 Hz. For example, the Beckhoff BK9103 Bus Coupler signals itself by the alternate flashing of two LEDs at a rate of 2 Hz. This function is very helpful for identifying the devices in this list. The flashing is stopped again by pressing the button once more. The flashing is stopped by closing the **ScanDevices** window. Subsequently, one or more devices can be marked with the Ctrl button. The selected devices are adopted into the project by pressing **Add Devices**

Upon pressing **Add Devices**, the following question box opens:



Yes button:

An attempt is initially made to determine the ModuleIdentNumber of the DAP (Device Access Point) by an implicit read access. If this fails a corresponding dialog opens containing the possible DAPs, which must then be selected manually. If all boxes have been appended, a ‘Reload Devices’ automatically takes place, i.e. the devices (adaptors) created are transmitted to the PROFINET driver. Subsequently, a distinction is made as to whether the box is a normal device or a drive with Profidrive support. For a normal device the actual module population (RealIdentificationData) is read out again via an implicit read access. If it is a

Profidrive device, conversely, the required information is read out by a Profidrive access. A Supervisor AR is established for this purpose. The required write accesses can take place within this. The Submodule interface on the DAP is taken here as the Parameter Access Point. The parameter access takes place via data record 47, much like the case of the Profibus beforehand. When using Sinamics, however, it must be noted that such an access is only permitted from version 4.3 SP2. If an older version is used, a corresponding error message appears and the parameterization must take place manually.

Once the automatic module parameterization has been completed, a question box appears asking whether the port data should be read in automatically. Here again, the port connection for the individual devices is read out via an implicit read access. The real port connection must be known for the various services. This can be limited to diagnostic services, although automatic device startup also requires this option (via alias), or specification of IRT planning. If this dialog is acknowledged with "no" or if the read access has failed, such a connection can also be made manually at the individual ports in the TwinCAT project.

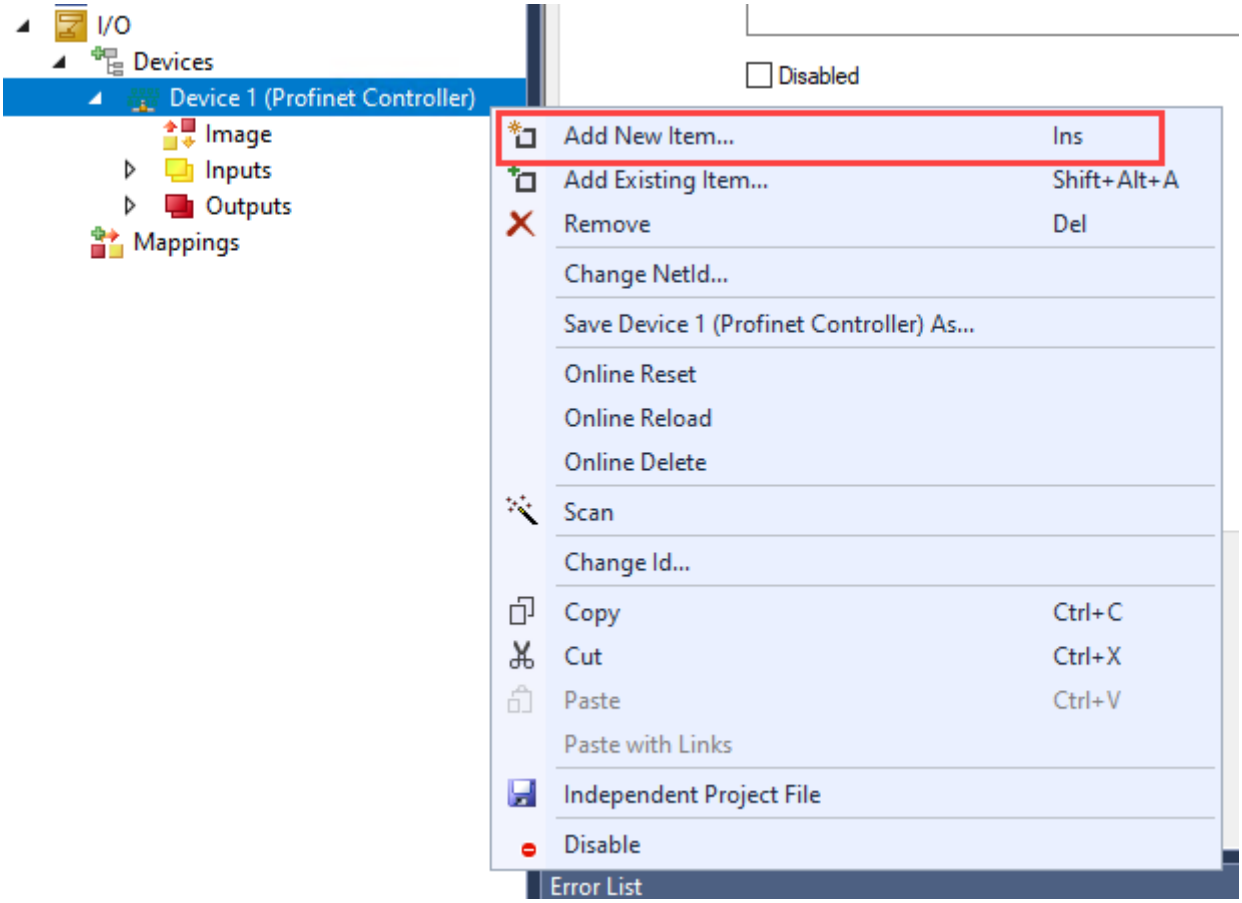
If the port connection has been successfully generated, a query also appears in the case of an IRT Controller (e.g. project planning on an EL6632) asking whether all devices are to be automatically connected in IRT Mode (provided they support it). If this is affirmed the cable length is additionally set to 10 m copper cable on all projected ports. The IRT algorithm requires this information for the calculation of the signal propagation delays. The precise cable length is not so important here (approx. +/-10 m), because the runtime delays tend to be small at 100 Mbit/s (5 ns/m). If the automatic switchover is not to take place immediately, these points can also be modified later either on the protocol or on the individual devices (on the interface or port submodule).

No button:

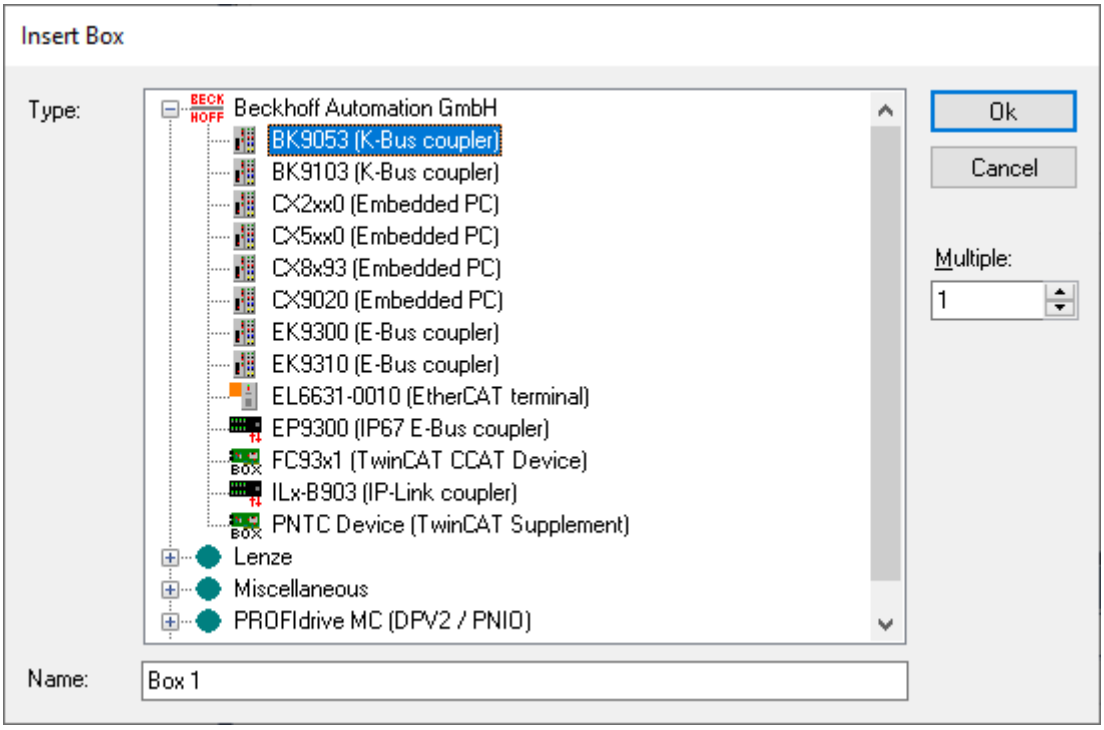
For each device a check is performed to see if the GSDML exists in the appropriate folder (..\ *TwinCAT\3.1\Config\Io\Profinet*). If this is the case, the list of possible DAPs is read in. Subsequently a selection dialog is opened so that the corresponding DAP can be selected. Once the devices have been appended in the project, the API below the Box can be accessed and the modules and submodules can be manually appended via it.

5.2.1.2 Appending via the dialog selection

The second configuration option involves adding the individual devices via the project tree. Right-click on the created PROFINET Controller and select **Add New Item**.



The following dialog opens.



There is a possibility here to select various PROFINET devices. The Beckhoff devices search for the GSDML under a defined path (..*TwinCAT\3.1\Config\Io\Profinet*). These should be already present with the TwinCAT installation. If there are several GSDMLs for the same device here, the one with the latest date is taken. If no device description is found, a corresponding error message appears. Either the GSDML is copied into the

folder and the menu is opened again, or the same procedure is selected as for the third-party devices. Click on **PROFINET IO Device** to navigate to the corresponding GSDML in Windows Explorer. This is then integrated into the project.

The DNS name from the GSDML is taken as the default name. When adding several devices at the same time, the default name is always supplemented by "-No." (where No. = 1..n). The assigned name (under which the device appears in the tree) is also the PROFINET Station Name, i.e. the name that must match the name in the device. The device name can be checked by scanning.

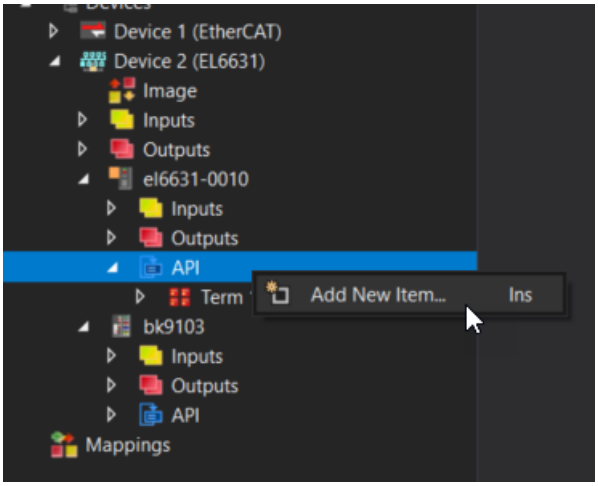
The modules can be attached to the API (Application Profile Interface). The DAP (Device Access Point), which already brings along fixed properties from the GSDML (e.g. process data, interface and port submodules, etc.), is always on Slot 0. This module is always present and cannot be deleted or moved. Each further module is assigned to a certain API. The information regarding its identity comes from the GSDML. By default this is always API 0. Alternatively, an API e.g. for the PROFIDRIVE profile or a fieldbus API is also conceivable. In the API click on **Append PROFINET Module...** to open a device catalog from which you can select and add the required modules. If the modules support it (described in GSDML), the submodules can in turn be appended to them in the same way.

5.2.2 Creating modules/process data

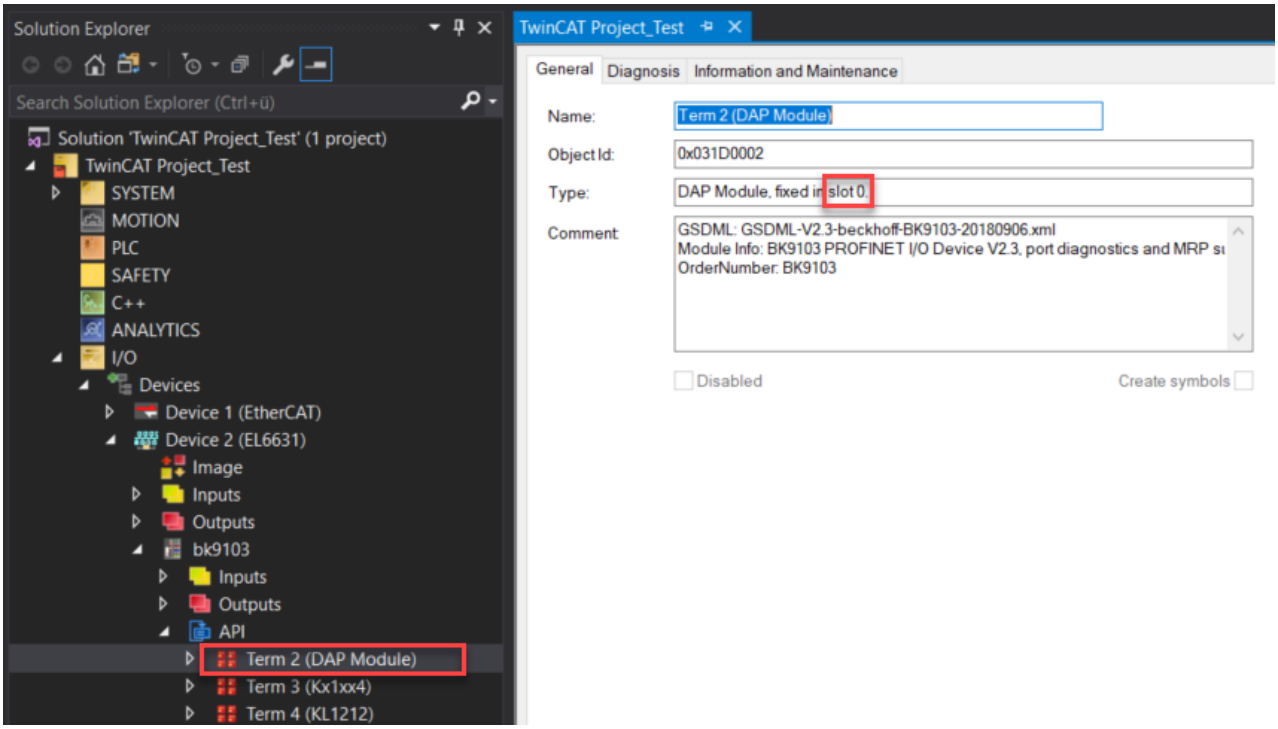
The procedure for creating modules/process data on a PROFINET Controller is the same for all controller types. In the following sections, the BK9053 Bus Coupler and the EL6631-0010 EtherCAT Terminal are used as PROFINET slaves to illustrate the addition of process data in various forms.

5.2.2.1 Up to TwinCAT Version 3.1 Build 4024

The modules can be attached to the API (Application Profile Interface). Open by right-clicking on **Add New Item**



The order of the modules in the tree always corresponds to the populated slot, starting at 0.

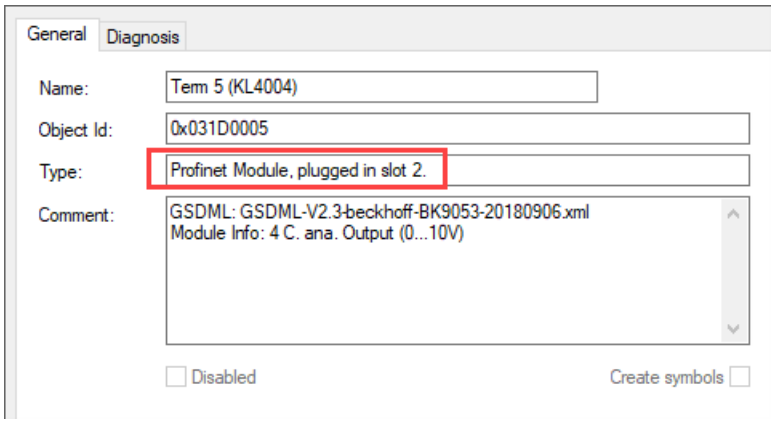


The DAP (DeviceAccessPoint) is added directly with the device. If, according to GSMDL, this is not plugged into slot 0, empty placeholder modules are inserted before it. The DAP is a special module that comes with device properties from the GSDML. The PDEV properties are attached to it in the form of submodules (interface and port). In addition, the DAP can also contain normal submodules with process and record data. It is always fixed and cannot be deleted.

Each additional module is assigned to a specific API. The information regarding its identity comes from the GSDML. By default this is always the API 0. Alternatively, an API e.g. for the PROFIDRIVE profile or a fieldbus API is also conceivable.

If the modules (described in GSDML) support it, the submodules can be projected below them. The subslots are also simply numbered continuously, starting at 1 (modules at 0). The PDEV submodules (interface and port) are exceptions; these are plugged into a fixed subslot specified via the GSDML.

The current slot or subslot can be checked via the associated object.



5.2.2.2 From TwinCAT Version 3.1 Build 4024

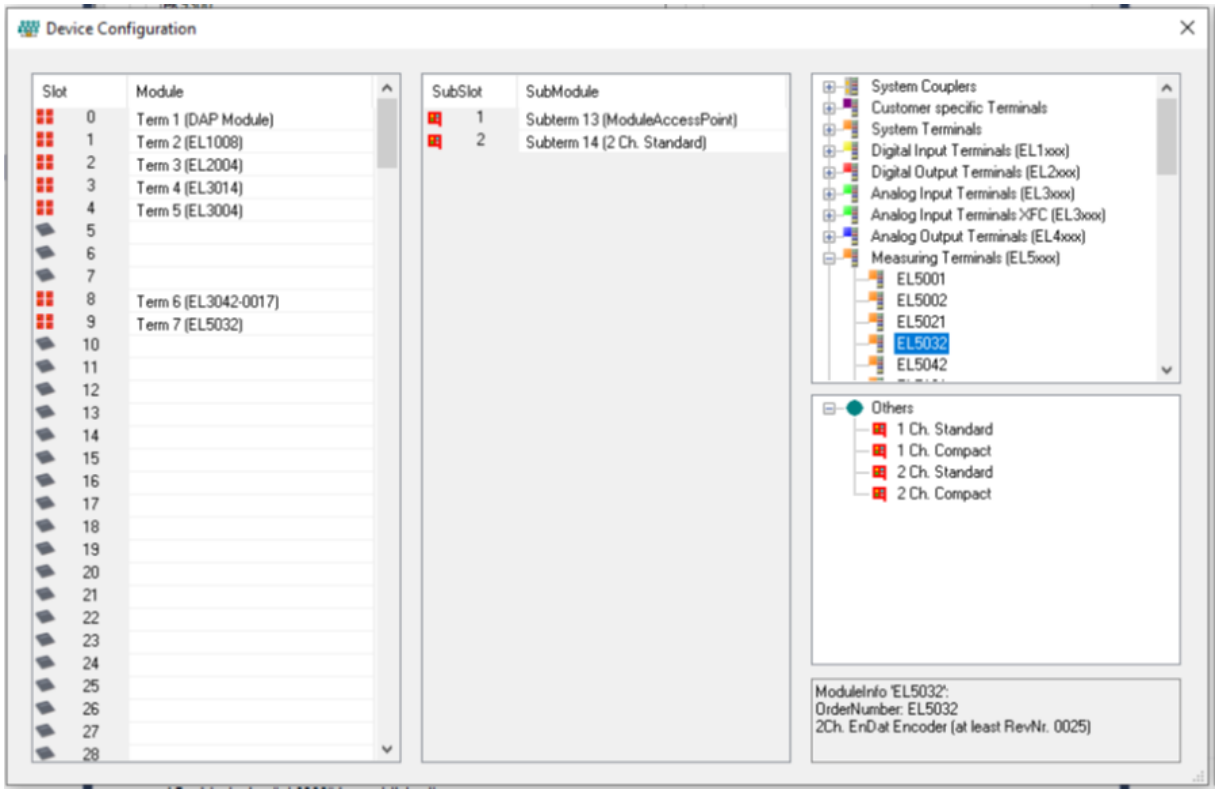
With TwinCAT Version 4024, the classic tree view for module/submodule population was replaced by a rack view.

In this view slots can be populated and removed without shifting the following slots. Empty slots can remain free and do not have to be provided with placeholders.

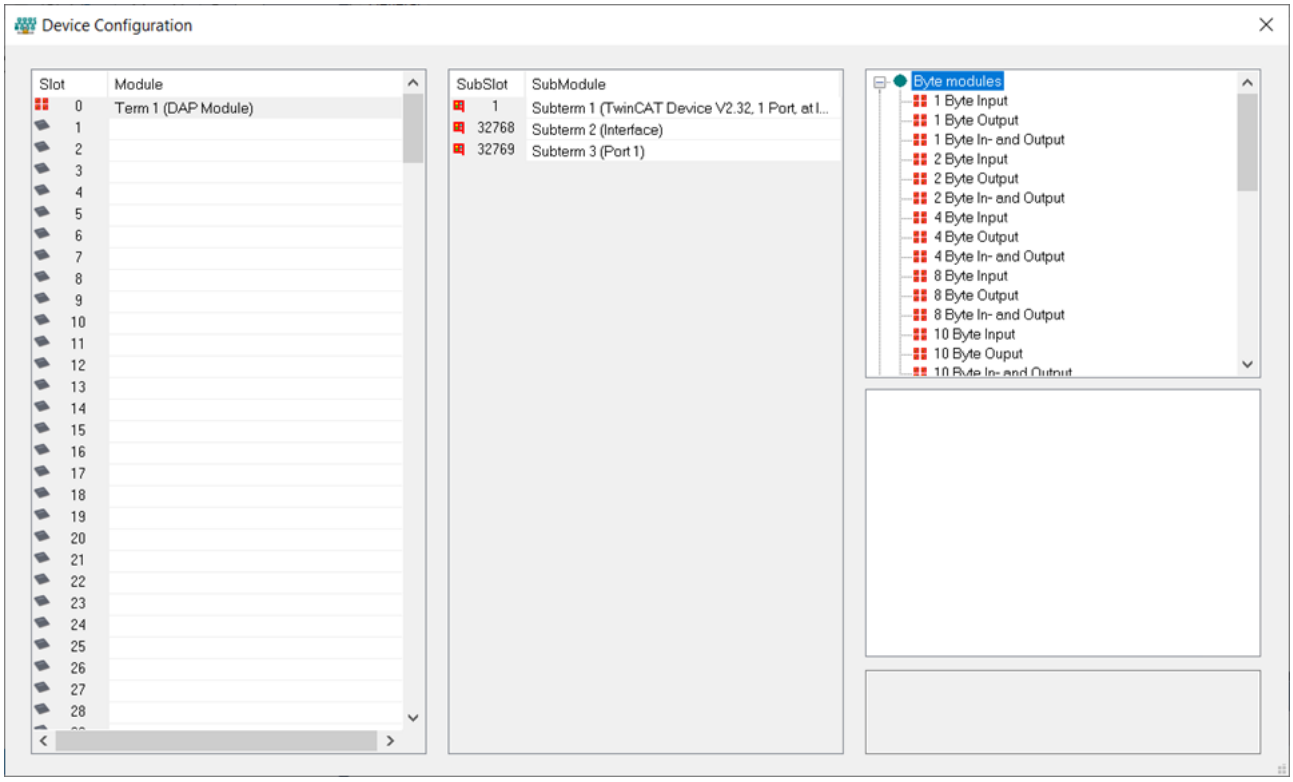
Furthermore, it is possible to generate the created module configuration in a GSDML file using the GSDML Generator. The generated file can then be integrated into the controller. The device configuration is thus fixed and does not have to be reprojected on the controller side.

For more information see chapter GSDML Generator.

The new rack view is illustrated below:



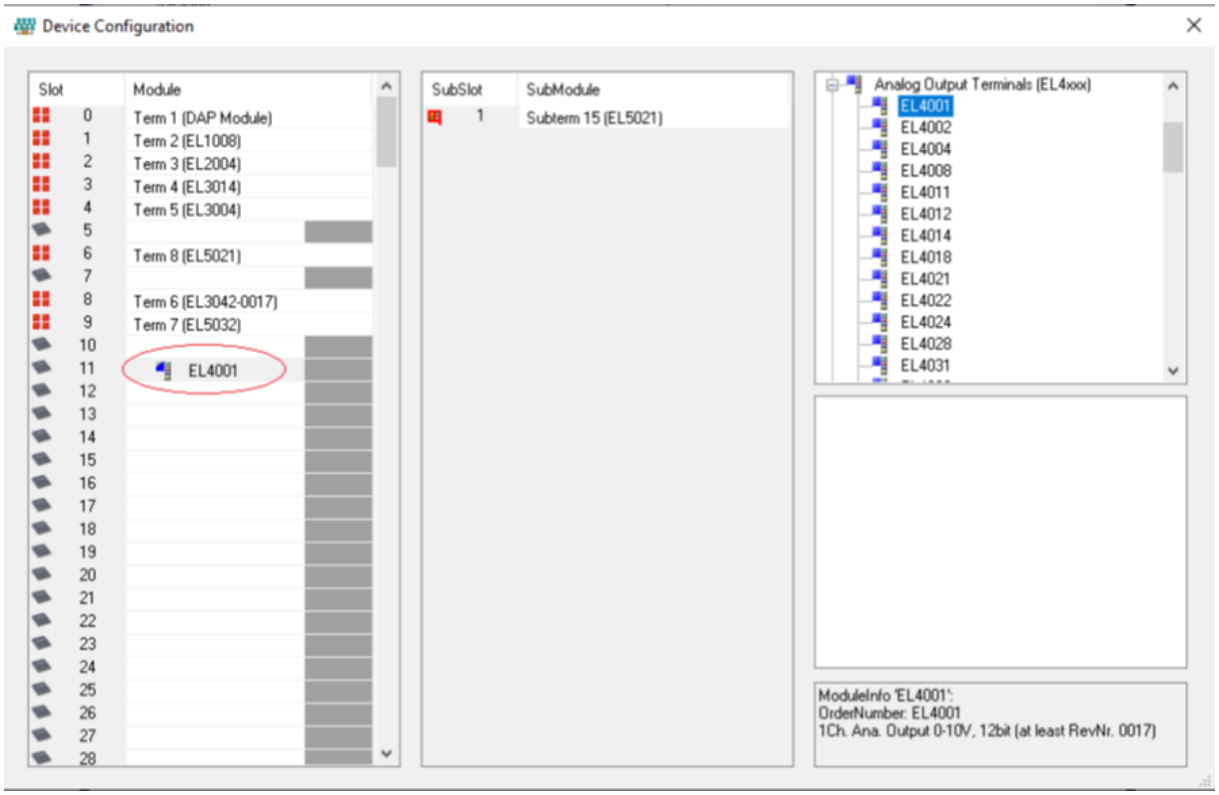
The list on the left shows the available device slots; here you can see which slots are occupied or free. Clicking on a slot updates the list in the center, showing the available subslots at the selected slot. If supported by the module, the submodules can then be populated here.



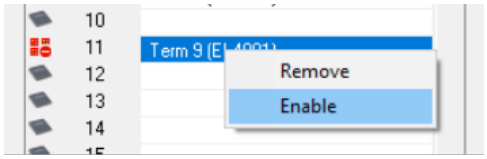
The list on the right shows the available modules and submodules. The list of submodules is always updated depending on the selected module.

The slots and subslots can be populated simply by double-clicking or via drag and drop.

- Double-clicking always triggers insertion at the next free and available slot from the cursor position.
- With drag and drop the available slots are selected, and a module can then be dragged to the required slot and released.



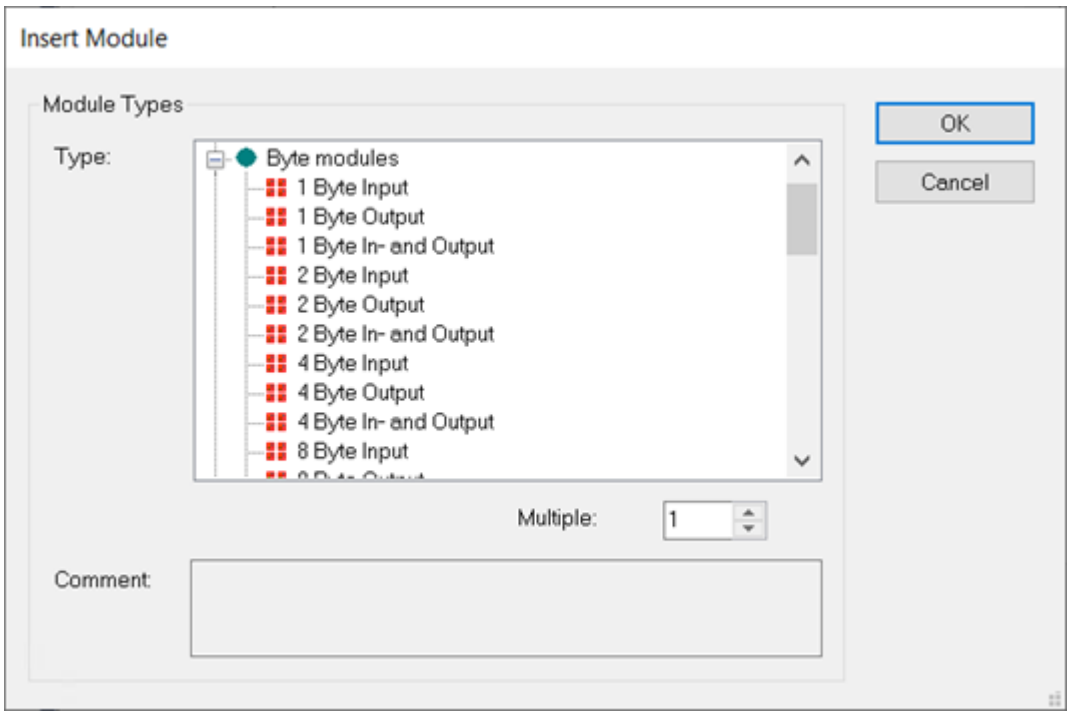
In the slot and subslot list, populated modules or submodules can be disabled via the context menu (right mouse click).



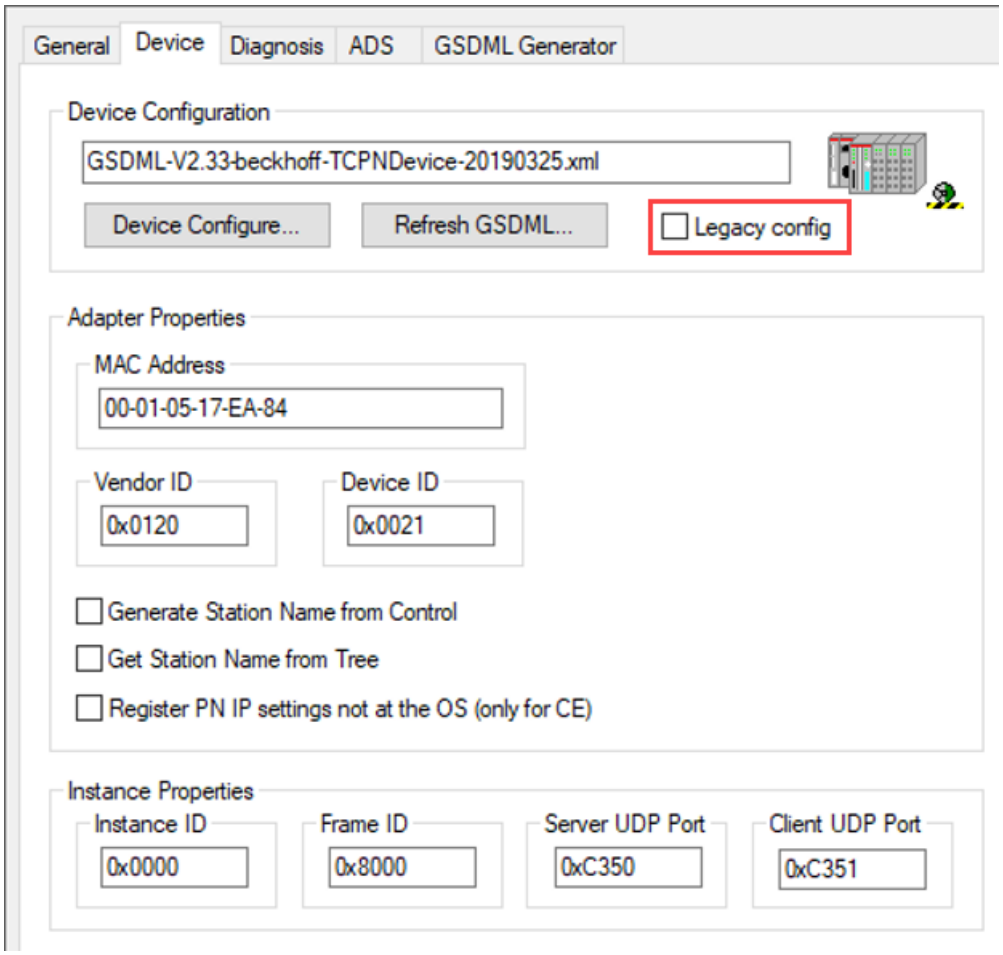
The icon indicates disabled status, which is also apparent in the module tree. Disabled objects are ignored during project planning, i.e. corresponding slots or subslots are considered empty.

Classic tree view

It is possible to switch to classic tree view for module/submodule population, if required.



To do this, check **Legacy Config** on the **Device** tab of the attached box.

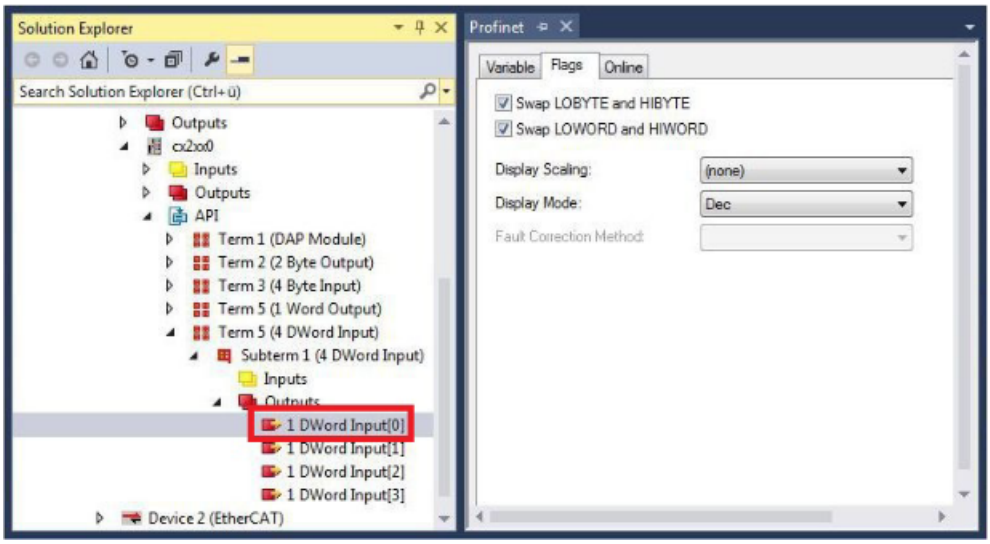


5.2.2.3 'Turning' process data

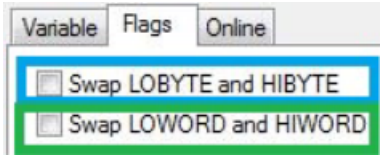
The process data are transferred in Intel format as standard. If the data is required in Motorola format, the data must be rotated accordingly. This step illustrates how to 'turn' the data in TwinCAT.

'Turn' the process data as follows:

1. Click the process data you want to rotate on the right in the tree view.
2. Click on the **Flags** tab



3. Click on the required option. For WORD variables, only LOBYTE and HIBYTE can be swapped. With DWORD process data you can additionally swap the WORD



⇨ In this way you can 'turn' process data. Use the following example to see how the data change for the individual options. Example for DWORD.

Controller data	Data received by the device			
Original data	No option selected	Swap Byte (blue)	Swap Word (green)	Swap both (blue and green)
0x01020304	0x01020304	0x02010403	0x03040102	0x04030201

5.2.3 Gigabit switch: Configuration and loss-free data transmission

The EL6633(-0010) EtherCAT Terminal is equipped with a 2-port Ethernet Switch that supports baud rates of 10/100/1000 Mbit/s.

The baud rate of each port can be set to a fixed value via PROFINET. Autonegotiation is active by default. The switch uses the Cut-Through method to transmit frames with minimal latency.

For optimum performance, both ports should be operated at the same baud rate. At different speeds, the switch automatically adapts to the store-and-forward method. To avoid frame loss, it is advisable to adjust the data rate to a lower speed in this case.

First example: Different baud rates with data loss

- **Port 1:** 100 Mbit/s
- **Port 2:** 1000 Mbit/s

An industrial PC connected to port 2 sends large amounts of data to an industrial PC connected to port 1 of the switch. Due to different port speeds, the switch cannot operate in Cut-Through mode and must buffer the frames. However, the buffer capacity (e.g. in the EL6633) is not sufficient for the amount of data generated.

Result: Frames are discarded → retries by the protocol → increased network load → lower overall performance.

Second example: Different baud rates without problems

A pure PROFINET network without additional Ethernet services:

- **Port 1:** PROFINET controller with 100 Mbit/s
- **Port 2:** PROFINET devices with 1000 Mbit/s

As the data is transferred from the slower (100 Mbit/s) to the faster (1000 Mbit/s) network segment - i.e. from the PROFINET controller to various PROFINET devices - there are no bottlenecks. The network load is already limited by the 100 Mbit/s segment. Example: A utilization of 25% at 100 Mbit/s corresponds to only 2.5% utilization on the gigabit side.

Setting a fixed baud rate

The baud rate is set via the PROFINET controller in the manufacturer's project engineering tool. The "Adjustable MauType" function must be supported for this; the manufacturer of the PROFINET controller should be checked to see whether this is the case.

- The setting requires active PROFINET communication
- When setting for the first time, the link can be briefly interrupted (restart of the Ethernet port)
- The baud rate is retained after the restart
- 10 Mbit/s is only available via autonegotiation

Recommendation: Keep the default setting Autonegotiation.

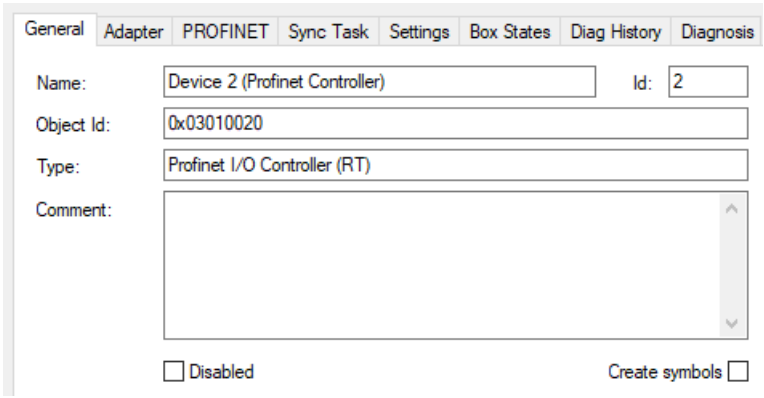
Table 1: Baud rate configuration and behavior of the switch.

Port 1 (Mbit/s)	Port 2 (Mbit/s)	Transmission direction	Procedure	Possible effects	Recommendation
100	100	both directions	Cut-Through	Optimum performance, no buffering problems	OK
1000	1000	both directions	Cut-Through	Optimum performance, no buffering problems	OK
100	1000	100 → 1000	Cut-Through	No bottlenecks, load is limited by the slower port	OK
1000	100	1000 → 100	Store-and-Forward	Buffer overflow possible → frame losses → higher network load	Adjust speeds or limit data rate
10	variable	both directions	Autonegotiation required	Very low speed, potential bottleneck	Not recommended
variable	10	both directions	Autonegotiation required	Very low speed, potential bottleneck	Not recommended

6 Settings and diagnostics

6.1 Settings on the PROFINET Controller protocol

6.1.1 General



Name

Identifier for the PROFINET device protocol object

Id

The device ID is set by the TwinCAT System Manager during configuration and cannot be configured by the user.

Object Id

Identification number of the PROFINET device protocol object in the TwinCAT object context.

Type

Shows the selected object type and its property.

Comment

Freely editable comment to describe the object used.

Disabled

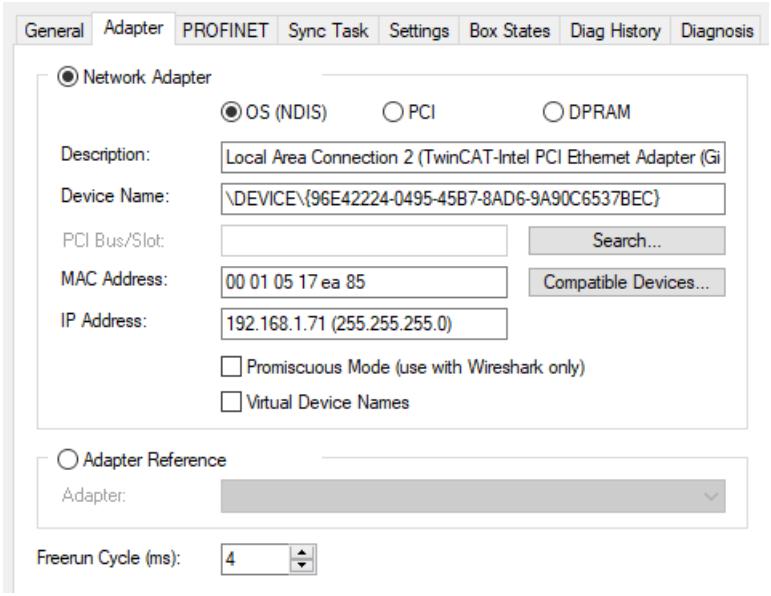
This option sets the PROFINET controller to inactive (transparent) for the current configuration. If this option is activated, the corresponding object is ignored in the IO configuration.

Create symbols

Creating variables as symbolic names.

6.1.2 Adapter

This dialog is used to specify and parameterize the network card to be used for communication with the PROFINET Controller.



OS (NDIS)

This option uses the operating system (OS) settings for installed network cards. The name of the network card is displayed under Description. Device Name contains the Device Manager path of the installed network card

PCI

This option controls the network card via the PCI bus address, which is specified in the PCI Bus/Slot field.

The PCIBus/Slot field is not enabled until the PCI option is selected

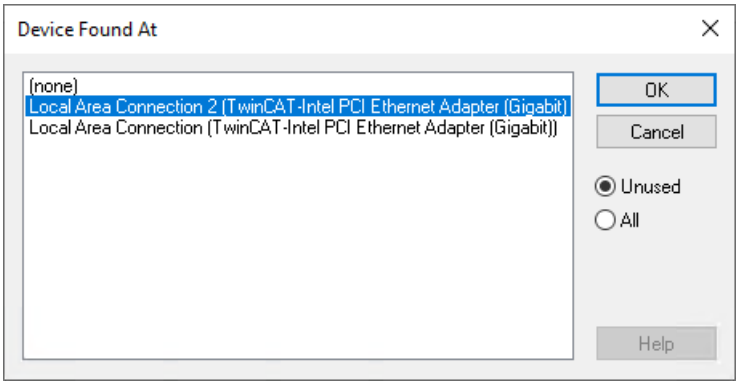
DPRAM

This option controls the network card via the DPRAM address, which is specified in the Address field.

The Address field is not enabled until the DPRAM option is selected.

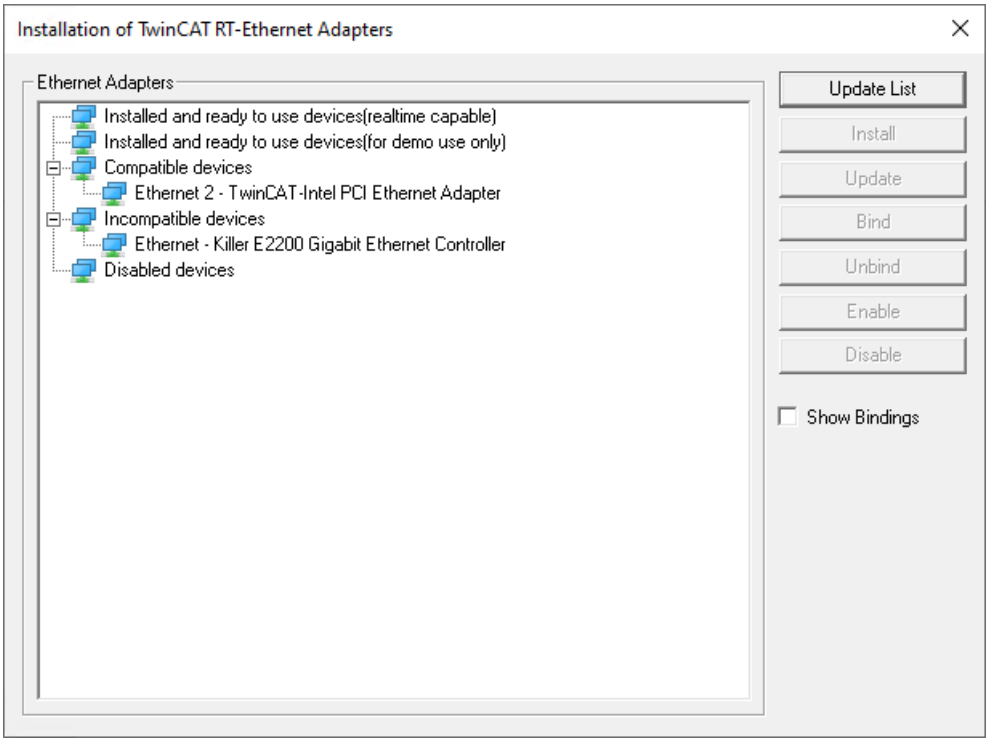
Search... button

This button opens a dialog in which all unused or all compatible devices (adapters) can be selected.



Compatible Device… button

This button opens the same dialog as "TWINCAT\ Show Real-time Ethernet Compatible Devices..." in the main menu. Use the dialog box to determine if compatible Ethernet adapters are available on the system.



MAC Address

MAC address of the Ethernet card (read-only).

IP Address

IP address of the card (read-only). The IP address is read from the operating system. It is unrelated to the PROFINET IP address that will be used later.

Promiscuous Mode

This is required in order to record Ethernet frames, and should normally be switched off.

Virtual Device Names

A virtual name is used for the network card.

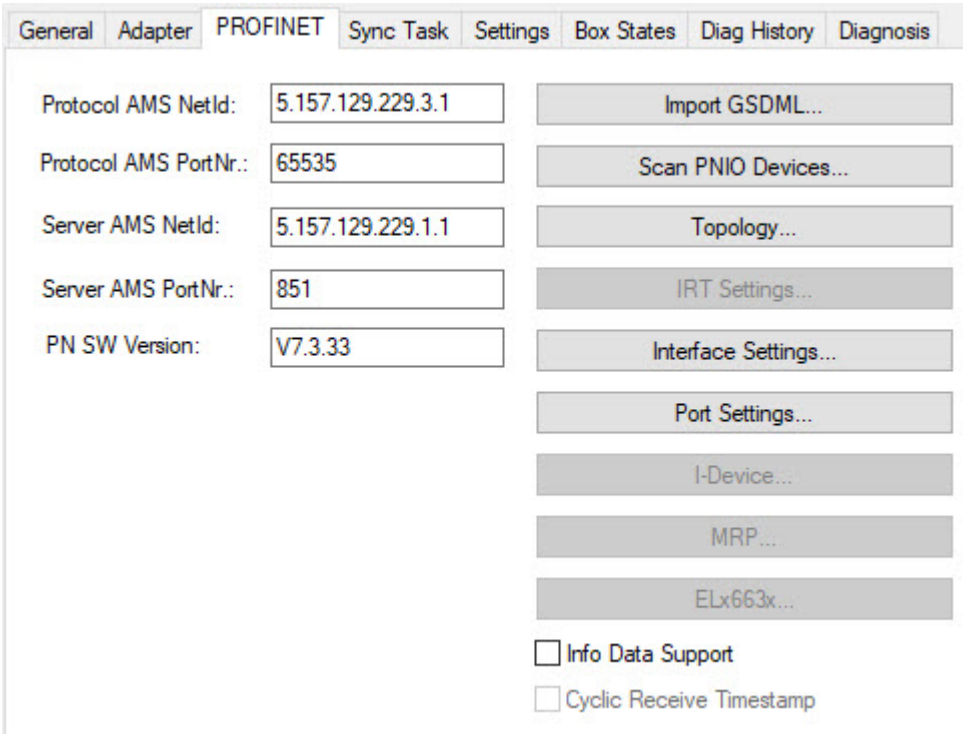
Adapter Reference

If the network adapter is referenced to another device, this option must be selected. This is used, for example, when using the "Multiple Protocol Handler".

Free Cycle

Cycle time in Config mode (no real-time). If TwinCAT is operated in FREERUN mode, care must be taken that the freerun cycle set is no longer than the PROFINET cycle!

6.1.3 PROFINET



Protocol AMS NetId

This is the NetID via which the PROFINET controller protocol can be reached via AMS.

Protocol AMS PortNr

This is the PortNo via which the PROFINET controller protocol can be reached via AMS. This is always fixed to 0xFFFF

Server AMS NetId

This is the NetID to which certain AMS messages (e.g. PN records within the index range 0x1000 - 0x1FFF) are forwarded by the PROFINET driver. Currently this is always the SystemNetId.

Server AMS PortNr

This is the PortNo to which certain AMS messages (e.g. PN records within the index range 0x1000 - 0x1FFF) are forwarded by the PROFINET driver. By default this is the PLC Port 802 of runtime system 1.

PN SW Version

Firmware version of the device

Import GSDML... From TwinCAT version 3.1 Build 4026

Imports and verifies GSDML and DSDMX files into the TwinCAT system.

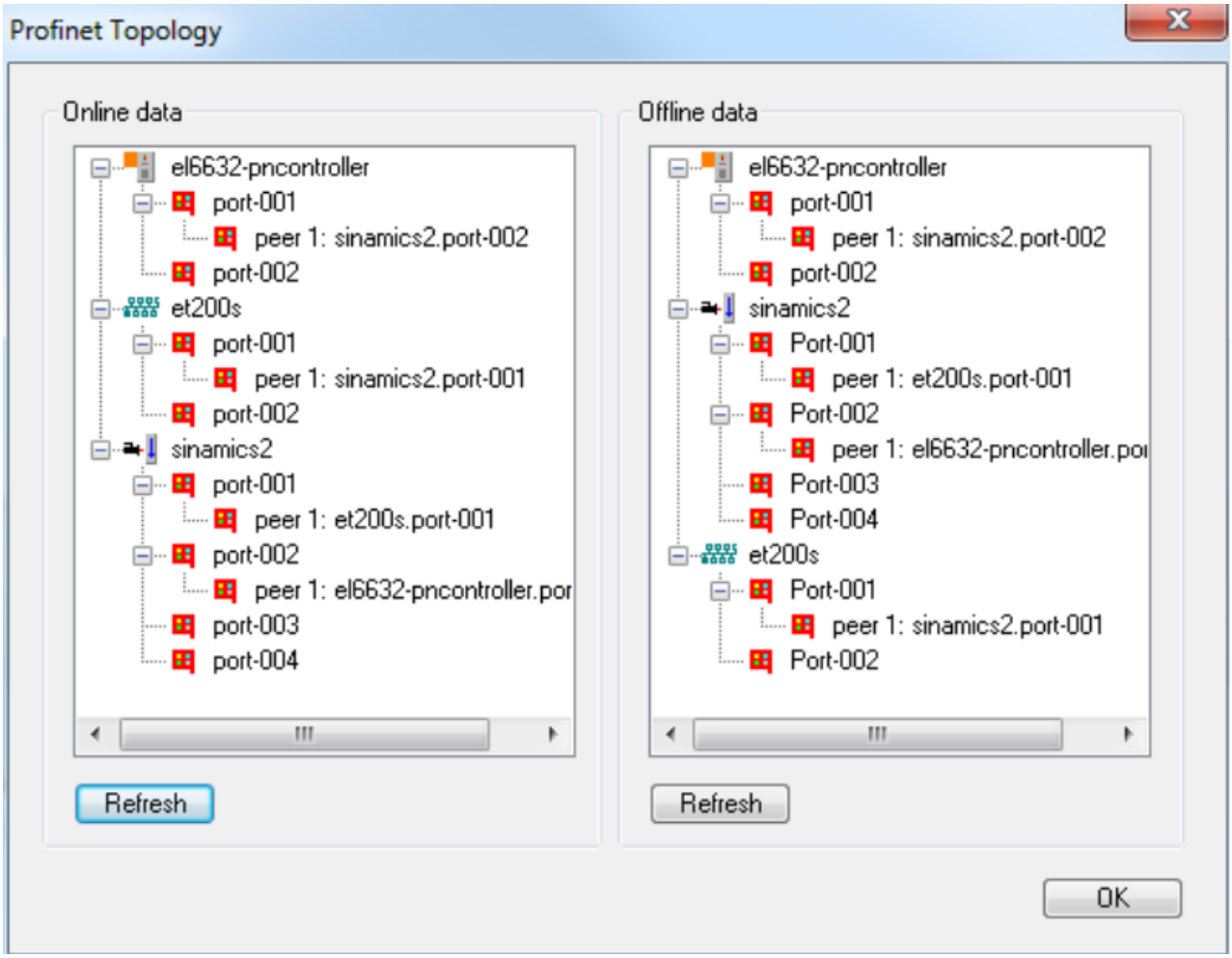
Scan PNIO Devices

Opens a search dialog for PROFINET devices; only available for the controller.

Appending via the Scan function [▶ 36]

Topology…

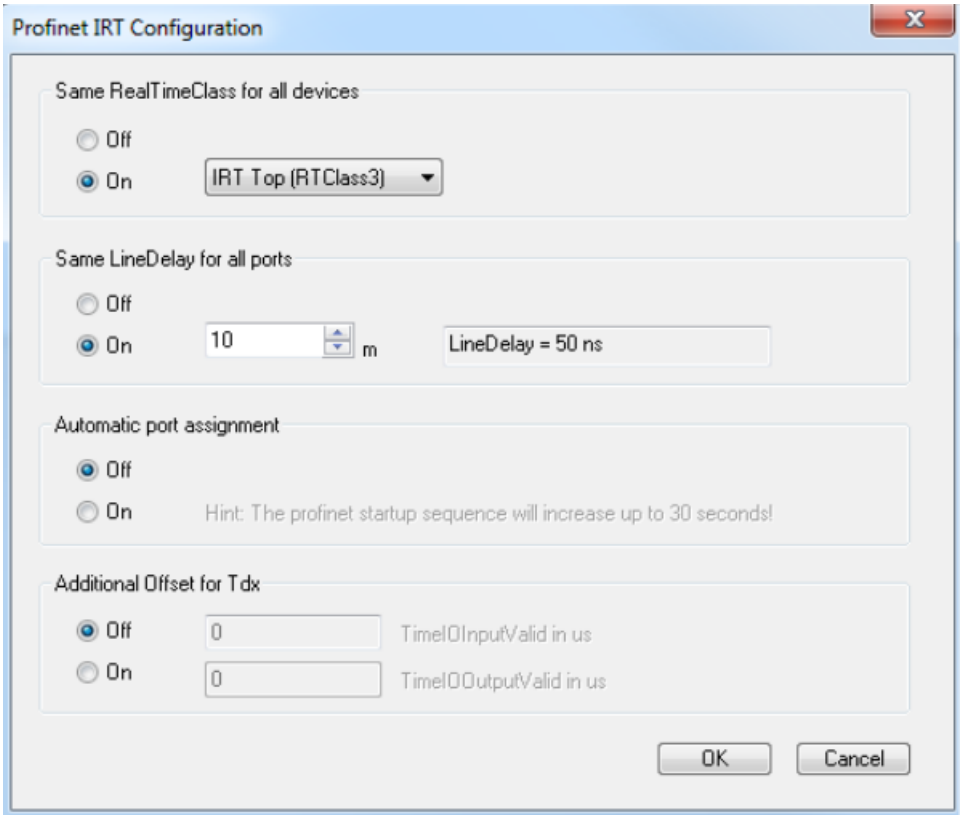
The offline topology can be compared with the online topology via this dialog.



It is quite possible for a device to have several partners on a port in the online view. This is the case, for example, if a switch is used in PROFINET that does not support LLDP (protocol for neighborhood ID). In offline view partners may have been assigned that do not exist in the project. This takes place if the reading of the port properties was enabled during scanning and automatic appending. In this case the device has a ‘neighbor’ that is adopted into the project, but the associated device box is missing in the *.tsm file. When this project is enabled, the "neighbor" that is not present in the *.tsm file is ignored in the driver.

IRT Settings…

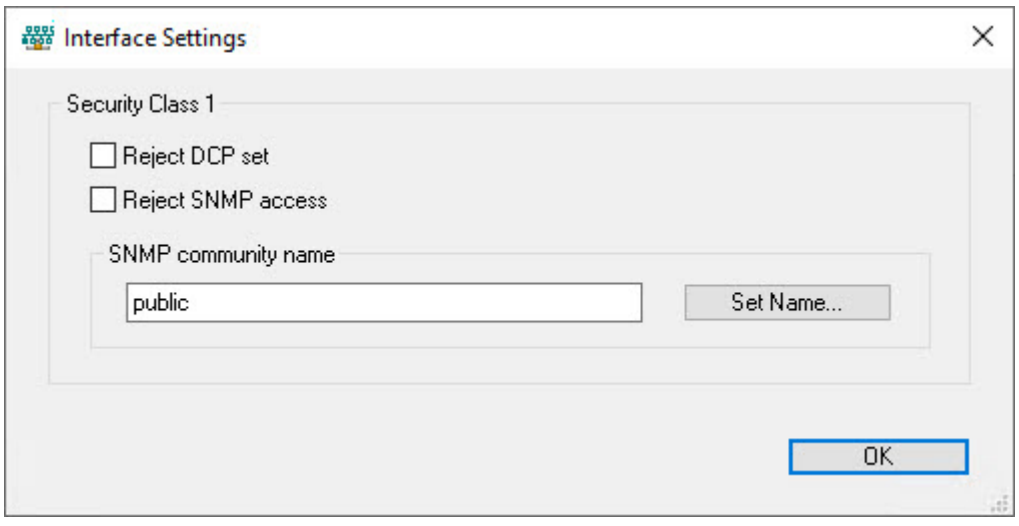
This menu is only enabled for an IRT-capable controller (currently only EL6632). A global setting can be made for all projected devices via this menu.



On the one hand the type of communication can be specified here. Currently, only RTClass1 (RT) and RTClass3 (IRT Top) are supported. This dialog also allows you to specify a general cable length (only for IRT). An approximate value or the max. cable length is sufficient here, because for the calculation of IRT communication this value tends to be lower (at 100 Mbaud and copper cable 5 ns/m). For optimization this feature can also be deactivated again later and the exact cable length can then be entered for each individual device (on the port submodules). Furthermore there is an option here to activate an 'automatic port assignment'. This makes the port connection set in the TwinCAT project irrelevant. Before each restart of PN communication, the topology is read out and used to calculate IRT communication. This has the advantage of minimizing potential cabling errors. In addition to that the ports can be simply replugged without having to change and reload the TwinCAT project. Only a restart of the PN communication is required (e.g. switch terminal to PREOP or disconnect the cable). This can extend the start-up time for PROFINET communication to up to 30 seconds. The reason for this is the TTL (TimeToLive) factor in the LLDP MIB. These are set by default to 20 seconds, i.e. only after this time can it be guaranteed that the port connection read is also the current one. Also, an additional offset for all Ti / To values can be specified in this menu.

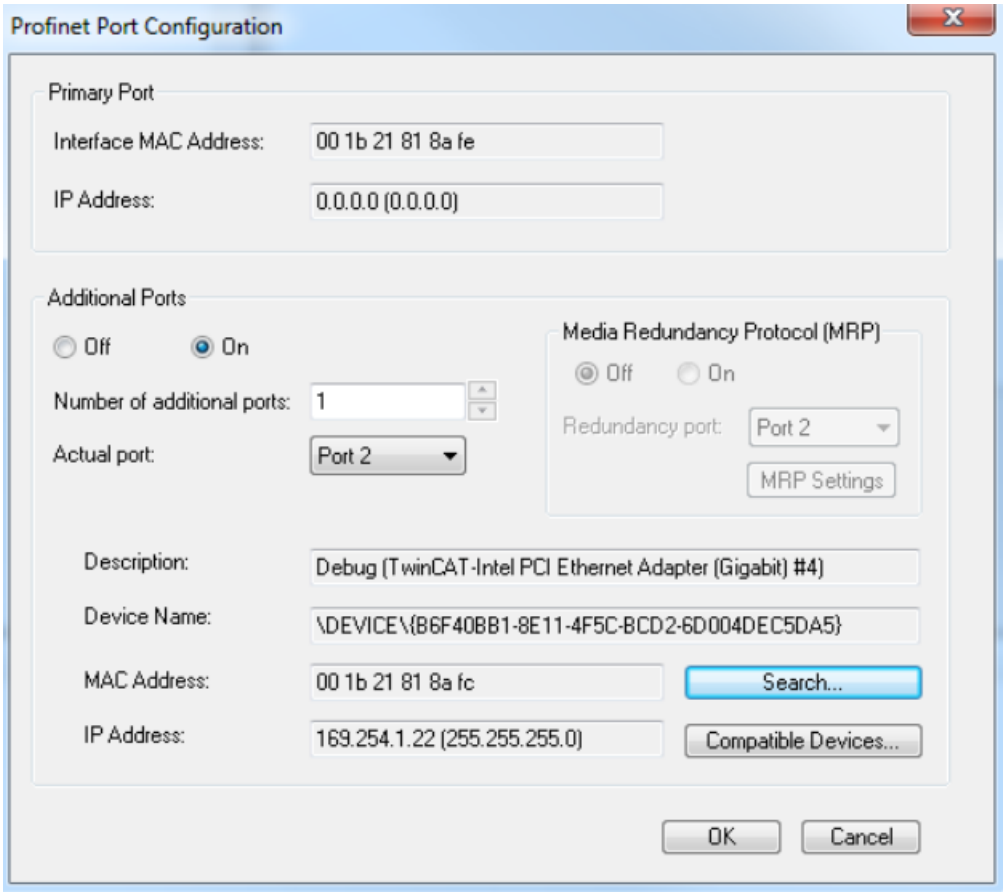
Interface Settings... From TwinCAT version 3.1 Build 4026

Opens a dialog with configuration options for the interface for security class 1 regarding DCP and SNMP.



Port Settings From TwinCAT version 3.1 Build 4026

This feature is currently only approved for the Real-Time Ethernet protocol (no EL663x). With this a second PROFINET port and an intelligent switch can thus be realized with a second network interface card (Intel® chipset). It is intended to repeat this feature x times; however, it is presently limited to one additional port.



I-Device

Opens a dialog for simultaneous parameterization of a controller AND device interface. Only available at CCAT.

Creating an I-Device

MRP… From TwinCAT version 3.1 Build 4026

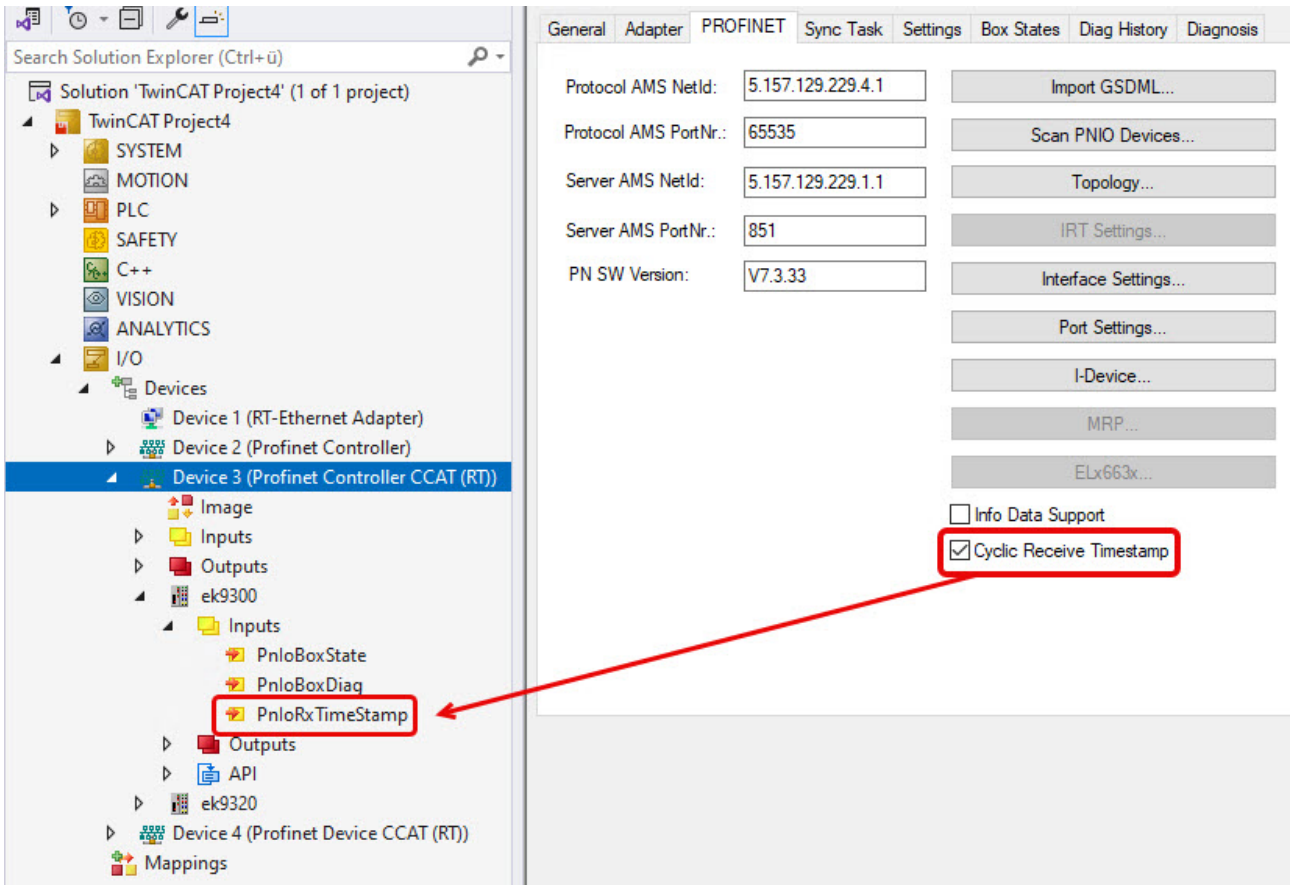
Opens a dialog to set MRP-specific parameters. Only available on the controller.

Info Data Support

If this option is activated, the AMSNETID is also available in the TwinCAT tree and can then be linked accordingly.

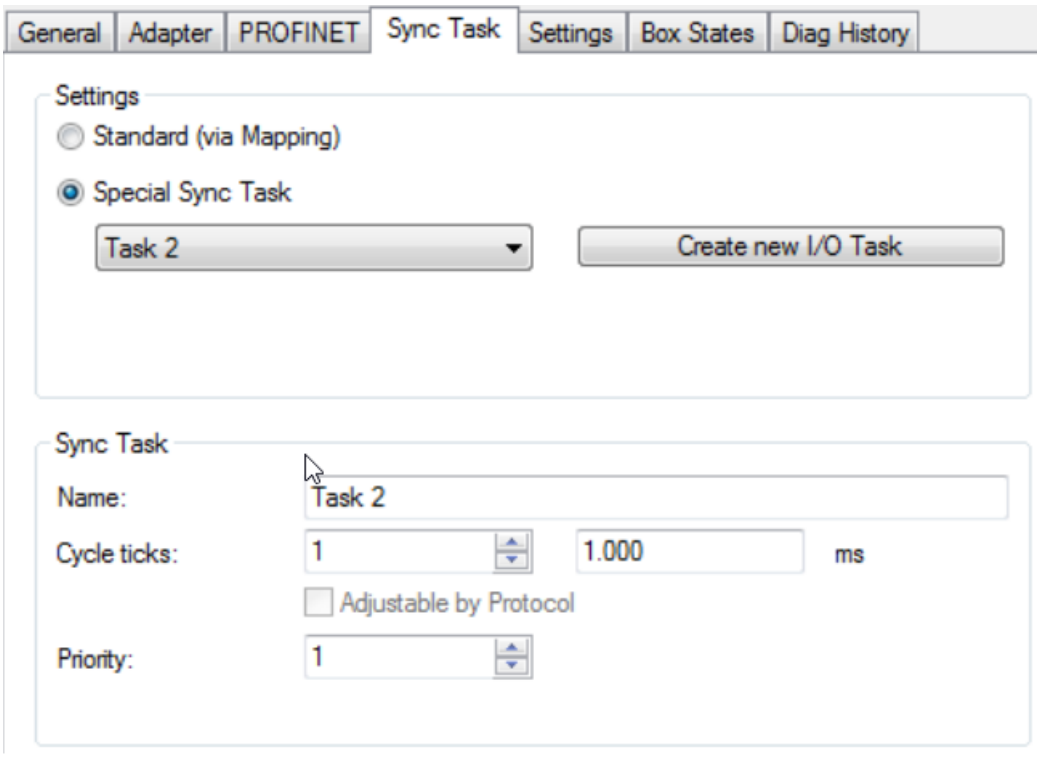
Cycle Receive Timestamp From TwinCAT version 3.1 Build 4026

All projected devices receive a cyclic timestamp. This means that the data received is stamped with the local clock. Only available at CCAT.



6.1.4 Sync Task

The PROFINET controller protocol must always be linked with a task. The protocol is also processed with the set task time. Theoretically the controller can also be jointly processed, for example, via a PLC or NC task. However, if a PLC project, for example, is stopped (e.g. by restart or debugging), this results in the PROFINET part also being stopped. In order to avoid such a side effect is it advisable to always create a free-running SyncTask.



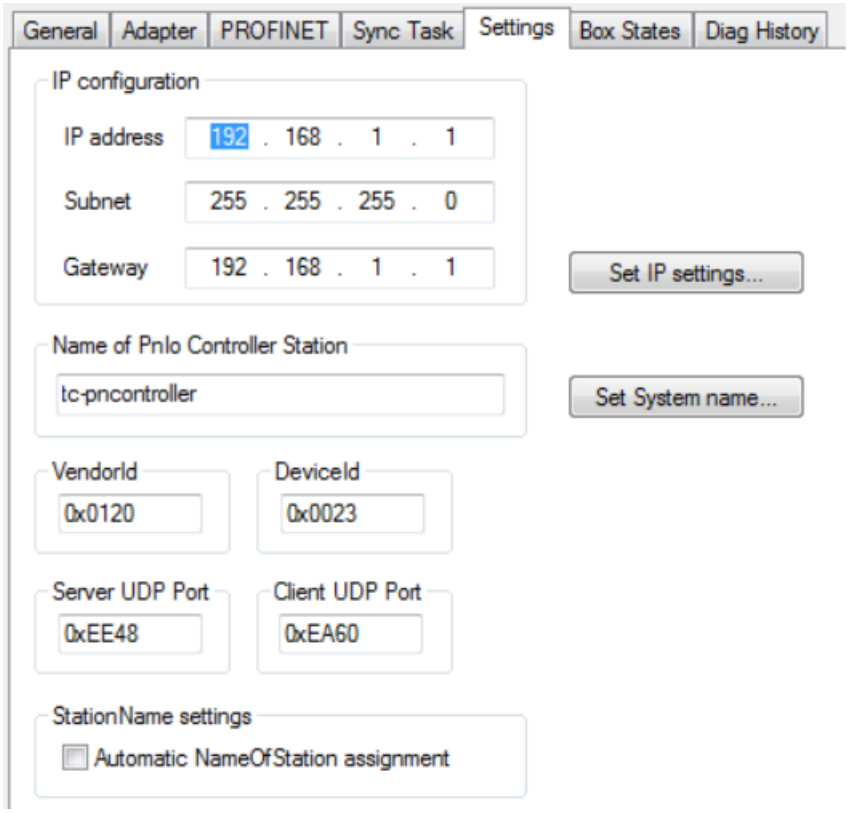
Make sure that the task cycle is in a PROFINET cycle. For PROFINET the basic cycle is 31.25 µs. The basic cycle is obtained by multiplying this cycle by the SendClockFactor (SCF). The SendClockFactor is usually set to 32 for RTClass1. For the Beckhoff PROFINET controller this is also the minimum PN cycle for RTClass1. The smallest resulting cycle time is 1 ms. Further reductions take place using a ReductionRatioFactor. It always corresponds to a multiple of the minimum PN cycle. For RTClass1, the smallest cycle must always be doubled **(permissible cycle times (for RTC1) with an SCF of 32 are 1, 2, 4, 8, ... 512 ms)**. In order to realize faster cycle times for RTClass3, the SCF must be reduced. This is presently at least 16 for a Beckhoff IRT controller (EL6632), which corresponds in turn to a basic cycle of 500 µs. When reducing the PROFINET cycle in this way, note that the time of the triggering task must also be adjusted accordingly.

Using EL663x

When using the EtherCAT PROFINET gateway terminal, please note that the process data always arrive one cycle late in the PLC, since one cycle is required to transfer the process data from PROFINET to EtherCAT

6.1.5 Settings

Specific settings that directly affect the controller can be made via the **Settings** tab.



An IP setting can take place here. The selection of the address range need not correspond to the network card settings. The PROFINET communication spreads its own net, which can be selected here. The IP Settings shown in the image above are the default settings. If no changes are made here, the controller will use these settings. The same applies to the controller name (system name). To change both settings you will need to select the appropriate button. A check is made to ascertain that the input is correct (e.g. the format of the controller name must correspond to the PN spec.). This data is transferred permanently. When changing the subnet or gateway, the settings are also applied to any projected devices.
It is also possible to change these settings via a supervisor tool. In addition the VendorID and DeviceID of the controller can be read out in this dialog. The server and client UDP port employed can also be set here. The default settings should be adequate in most cases. Furthermore there is a possibility in this dialog to enable an automatic PROFINET start-up following a device exchange (including devices without removable media). The target topology must be specified to ensure correct function. On the basis of this information the controller can query the alias names of the individual devices. Every device that supports alias names generates such a name for each of its ports. This is composed of the neighborhood IDs (PortId.ChassisId). If

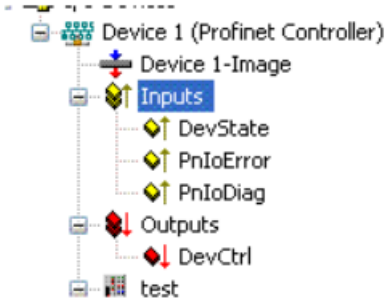
this name is queried, the ‘new’ device answers. If VendorId and DeviceId are correct the device is named with the actual name and a normal PROFINET start-up can subsequently take place. With this mechanism a complete PROFINET system could also start up without having named an individual device beforehand.

When using the TF627x supplement, note that the IP addresses of the operating system and PROFINET are different.

6.2 PROFINET controller protocol diagnostics

6.2.1 Box States

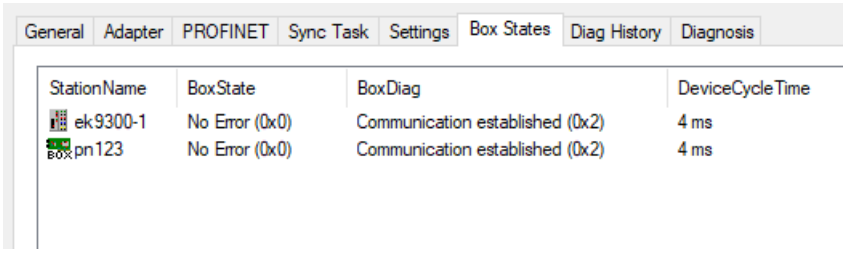
Directly below the PROFINET controller protocol there is a collective PROFINET error and a collective PROFINET status. Both show the number of devices for which a problem has occurred or for which a diagnosis is available. The error indicates possible problems with the connection establishment or reasons for a termination. The diagnosis provides status information about an existing connection.



PnIoError - number of PROFINET IO devices that have an error

PnIoDiag - number of PROFINET IO devices where a diagnosis is available

It is possible to check at a glance which device or box has a problem in the protocol under Box States.



Presently the following error messages are displayed under the “PnIoState”.

Number	Text	Description	Remedy/Cause
0	No Error	No error	No error
1	PROFINET Device state machine is in boot mode	PROFINET DeviceStateMachine is still in the start-up phase	Not an error, wait
2	Device not found	Device does not reply to the Identify Request	Check the connection; device switched on and connected; was the device named correctly?
3	The station name is not unique	The station name is not unique	There are two or more devices in the network with the same PROFINET name. A correct identification cannot take place.
4	IP could not set	IP address could not be set.	The PROFINET device has rejected the IP settings for some reason. Check whether the IP settings are correct.
5	IP conflict	IP conflict in the network	A possible cause is that several devices have the same IP address.
6	DCP set was not successful	There was no reply or an erroneous reply to a DCP Set	Check connection, device connected, was the device called by its correct name?
7	Watchdog error	The connection was broken off with a Watchdog error	Check the cycle time, check the connection, if necessary increase the Watchdog factor.
8	Datahold error	The connection was broken off with a Datahold error	Frame Data status was invalid for the length of the DataHoldTimer. Restart the device if necessary
9	RTC3: Sync signal could not be started	For IRT only: the Sync signal could not be started.	Is EtherCAT Sync signal correct or has Sync0 started?
10	PROFINET Controller has a link error	The PROFINET controller has no link	Check cable and connection.
11	The alias name is not unique	The alias name is not unique	There are two or more devices in the network with the same alias name. This is made up of the neighborhood information (PortId.ChassisId). A correct identification cannot take place.
12	The automatic name assignment is not possible - wrong device type	The automatic name assignment is not possible	The expected PROFINET device is not in the projected position (VendorId or DeviceId does not correspond). This means that no automatic naming and thus no device start-up is possible.
31	Only for EtherCAT gateways: WC-State of cyclic EtherCAT frame is 1	For EL6631 only: EtherCATWC state is set to 1	Check the mode on the EtherCAT master & slave (OP?).

As opposed to the state, more than one status can be displayed in the “BoxPnIoDiag”, i.e. the whole thing is bit-coded and up to 16 pieces of information can be displayed. The following statuses are currently displayed.

0x0000 = No diagnosis

0xXXX1 = IOC-AR is not established

0xXXX2 = IOC-AR is established

0xXXX4 = IOC-AR is established but no ApplReady

0xXXX8 = IOC-AR is established but module difference

0xXX1X = At least one AlarmCR get diagnosis alarm

0xX1XX = At least one InputCR is invalid

0xX2XX = At least one InputCR Provider is in stop

0xX4XX = At least one InputCR Problemindicator is set

0x1XXX = At least one OutputCR is invalid

0x2XXX = At least one OutputCR Provider is in stop

0x4XXX = At least one OutputCR Problemindicator is set

On the one hand information about the status of the IO Controller Single AR is displayed here. In addition, collective statuses are formed from the Frame Data statuses of the individual CRs. This applies to the input and output CRs (currently only one CR is possible, in future several). A PROFINET alarm is also displayed in "PnIoDiag"

Readout via ADS

The Box Status can be read out via an ADS Read.

ADS Read:

NetId = AMSNETID of the PROFINET controller

Port = BoxPort (0x1000 + BoxId)

Indexgroup = 0xF829

IndexOffset = 0

Length = sizeof(TPnIoDeviceDiagData);

where:typedef struct{

WORD pnioState;

WORD pnioDiag;

WORD NrOfInputCRs;

WORD NrOfOutputCRs;

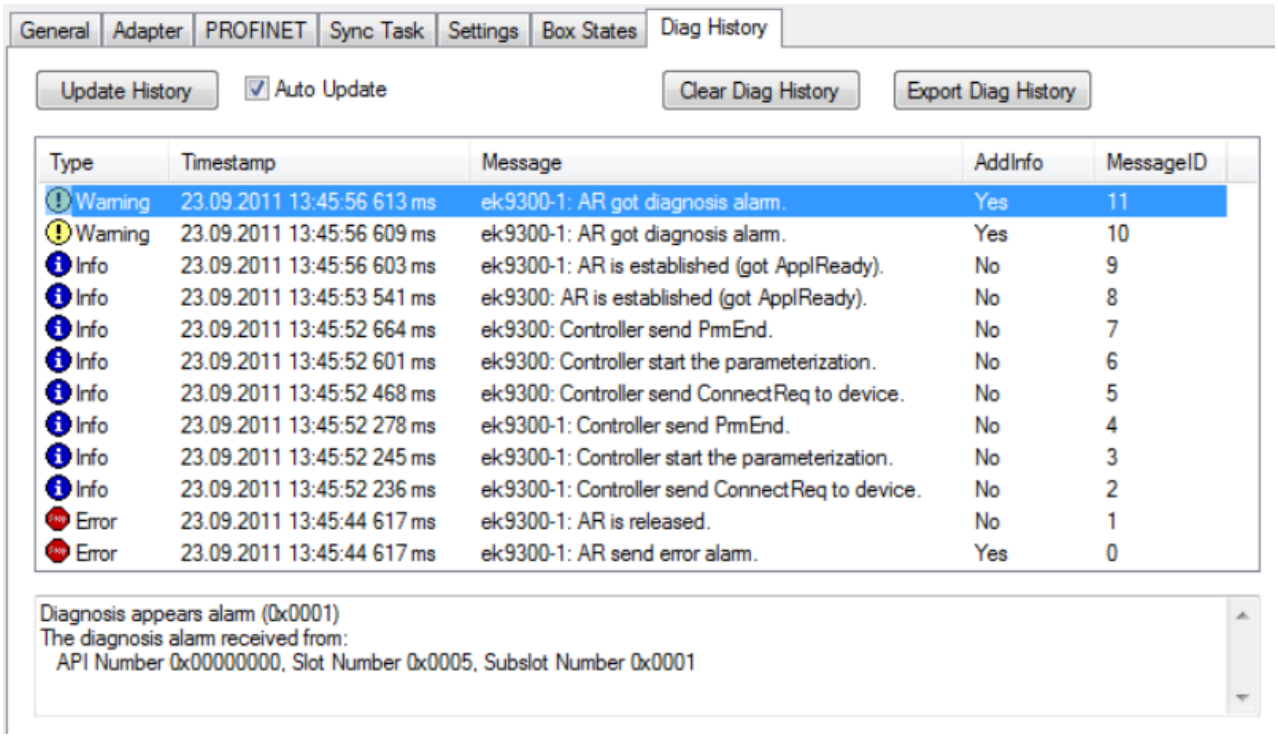
WORD reserved[8];} TPnIoDeviceDiagData, *PTPnIoDeviceDiagData;

Readout via CoE (for EL663x)

The Box Status can also be read out via CoE for the EL663x. The index 0xAyy0 (where yy is the Adaptor / Device number) and the subindex 0x001 must be taken for this.

6.2.2 Diagnostics history

Logged diagnosis messages from the controller protocol can be read out on the **Diag History** tab. The diagnosis buffer operates as a ring buffer with a current maximum size of 1000 entries.



The possible errors are grouped into three types:

Info: e.g. information on the connection establishment

Warning: e.g. PROFINET diagnosis alarms

Error: e.g. loss of connection

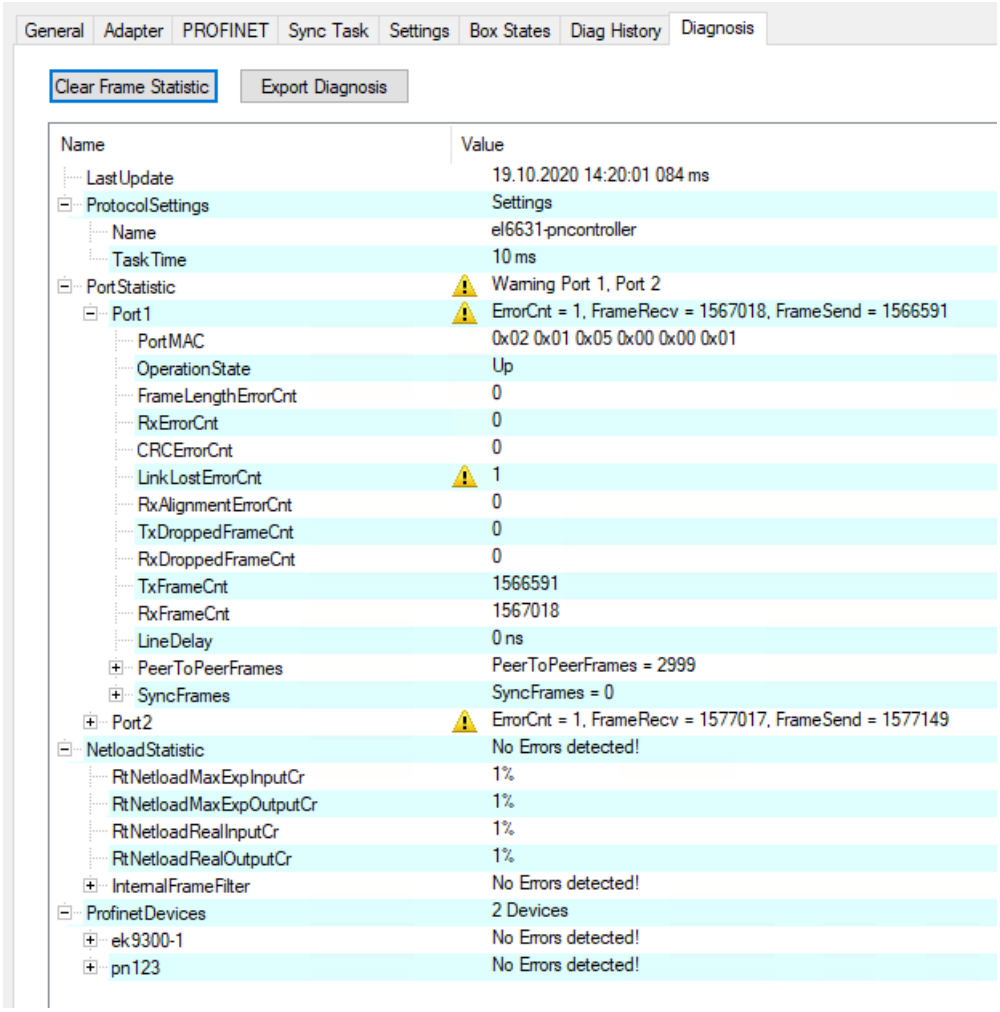
“AddInfo” indicates whether additional information about the event is available. If this is marked by “Yes”, the additional information can be fetched and displayed by clicking on the respective message. In the case of a diagnosis alarm ("Diagnosis appears"), the precise diagnosis information can be fetched at the corresponding level (device, API or module).

The complete diagnosis buffer is cleared by pressing the **Clear Diag History** button.

The displayed messages can be saved in a .TXT file by pressing the **Export Diag History** button.

6.2.3 Diagnosis

This tab contains the frame statistics; this list contains various information and statistics.

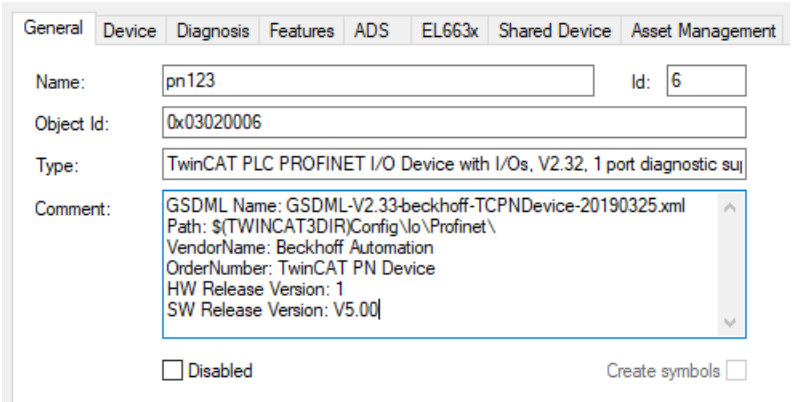


These can be reset or exported using the corresponding buttons; they are divided into the following sections.

1. Protocol settings: Include the name and cycle time of the PROFINET protocol.
2. Port statistics: Port-specific statistics and diagnostics.
3. Network load statistics: Percentage display of the expected bus load associated with the cyclic process data.
4. Also diagnostic counter of the internal network load filter to avoid possible frame bursts.
5. PROFINET devices: Diagnosis and statistics for the projected PROFINET devices.

6.3 Settings on the PROFINET device

6.3.1 General



Name

Identifier for the PROFINET device protocol object

Id

The device ID is set by the TwinCAT System Manager during configuration and cannot be configured by the user.

Object Id

Identification number of the PROFINET device protocol object in the TwinCAT object context

Type

Displays the selected/integrated GSDML and its property.

Comment

Displays further information on the selected/integrated GSDML, if available. Also freely editable commentary.

Disabled

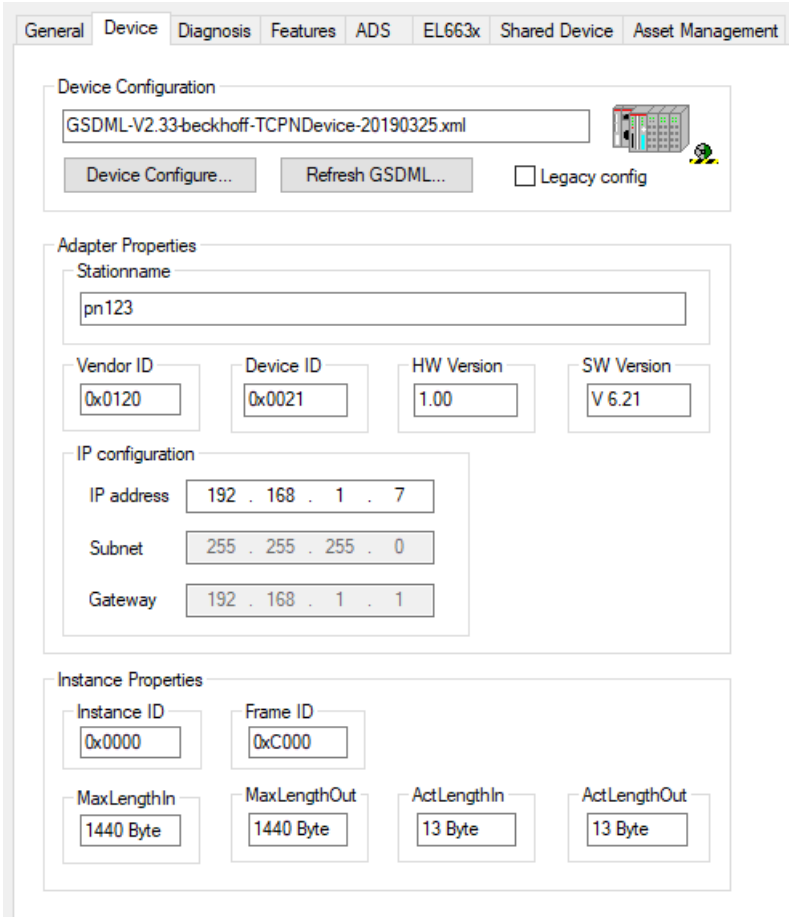
This option sets the PROFINET device to inactive (transparent) for the current configuration. If this option is activated, the corresponding object is ignored in the IO configuration.

Create symbols

Creating variables as symbolic names

6.3.2 Project planning of the PROFINET device

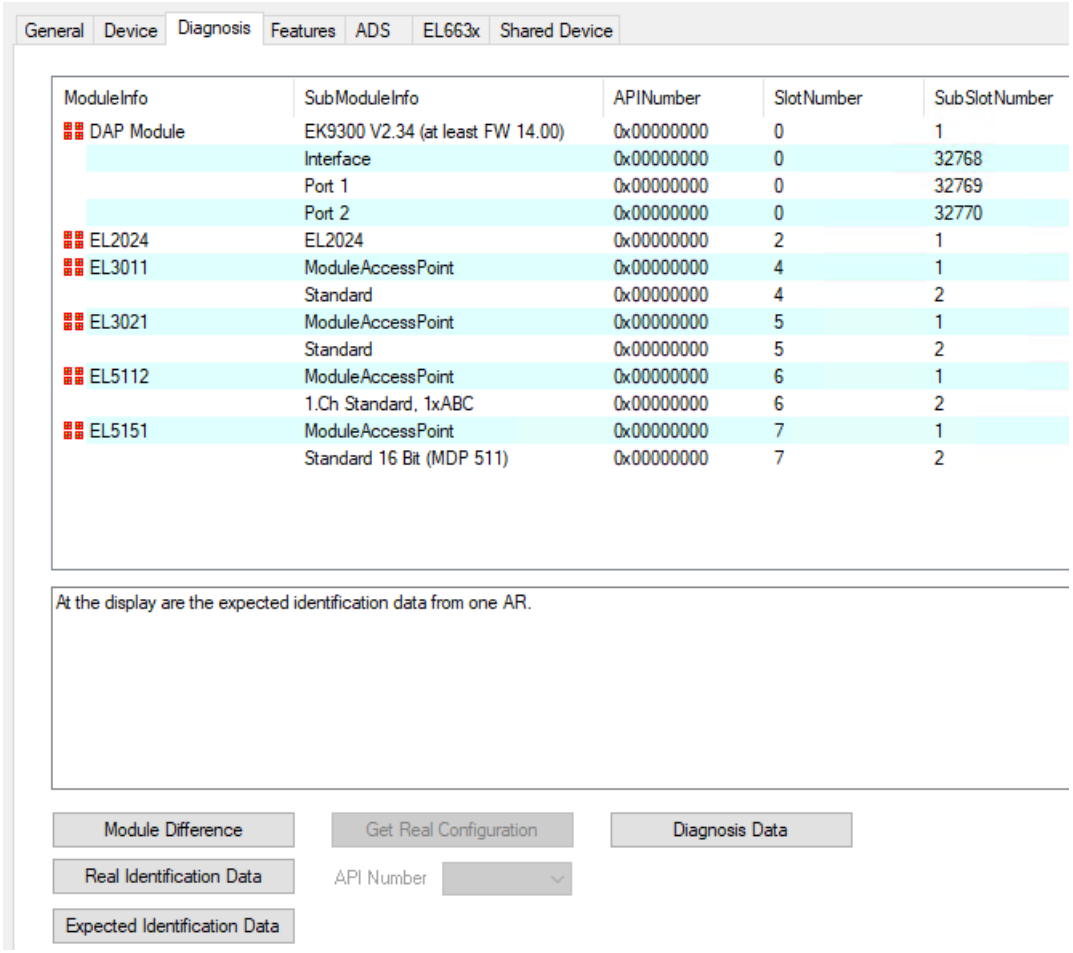
When establishing a PROFINET connection the controller always assigns an IP address to the device from its own address space (if the device does not yet have one or if it has a different one). In TwinCAT the next higher address is taken for a device by default (starting from the controller adaptor class); the subnet and gateway are the same as those of the controller. Before the actual assignment of the IP address to the device by the controller, an ARP is used to test for a possible address conflict or to check whether the device already has this IP address. If there is a conflict, e.g. that the IP address is already assigned in the network, the IO driver determines this and outputs a corresponding message in the logger window. If there is no reply to the ARP, this means that no device (the projected device included) is using this IP configuration, which in turn results in the controller assigning the IP settings to the device via a DCP_SET. Setting is skipped if it is determined via the ARP that the device sought already has the projected IP address. If the IP addresses of the PROFINET device and the operating system are identical, this can lead to unexpected behavior such as errors in the structure of the AR.



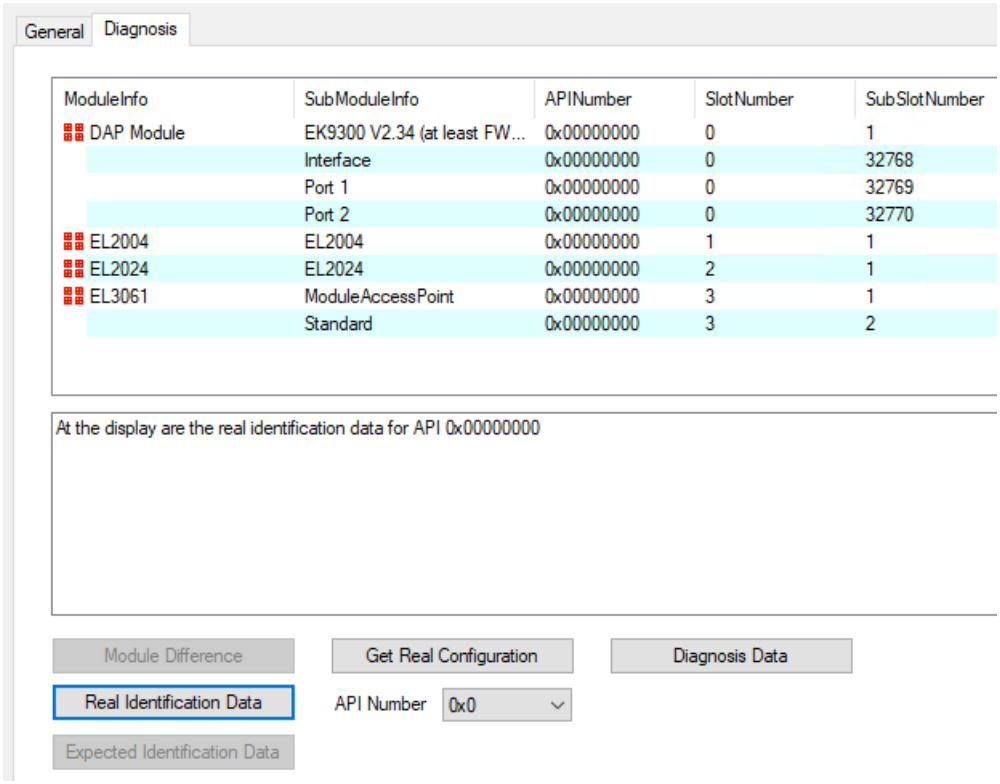
In this window you can also change the **InstanceID** and the **FrameID**. However, the default settings are adequate for most applications. The Instance ID is incorporated into the formation of the UUID object. A change should therefore be made only in exceptional cases. When changing the FrameID, the RTClass used must be taken into account (e.g. for RTClass1 unicast0xC000 - 0xFAFF). If the device is on an IRT controller and all devices have been switched automatically to RTClass3, the FrameID is managed automatically and there is no input option (marked by "Fast Config"). The current process data length can also be checked in this menu. **MaxLength** indicates which process data size is supported by the corresponding device, **ActLength** indicates the current process data length (including IOPS and IOCS). The corresponding error message appears if the maximum lengths are exceeded on appending further modules/ submodules.

6.3.3 Comparison of nominal and actual population

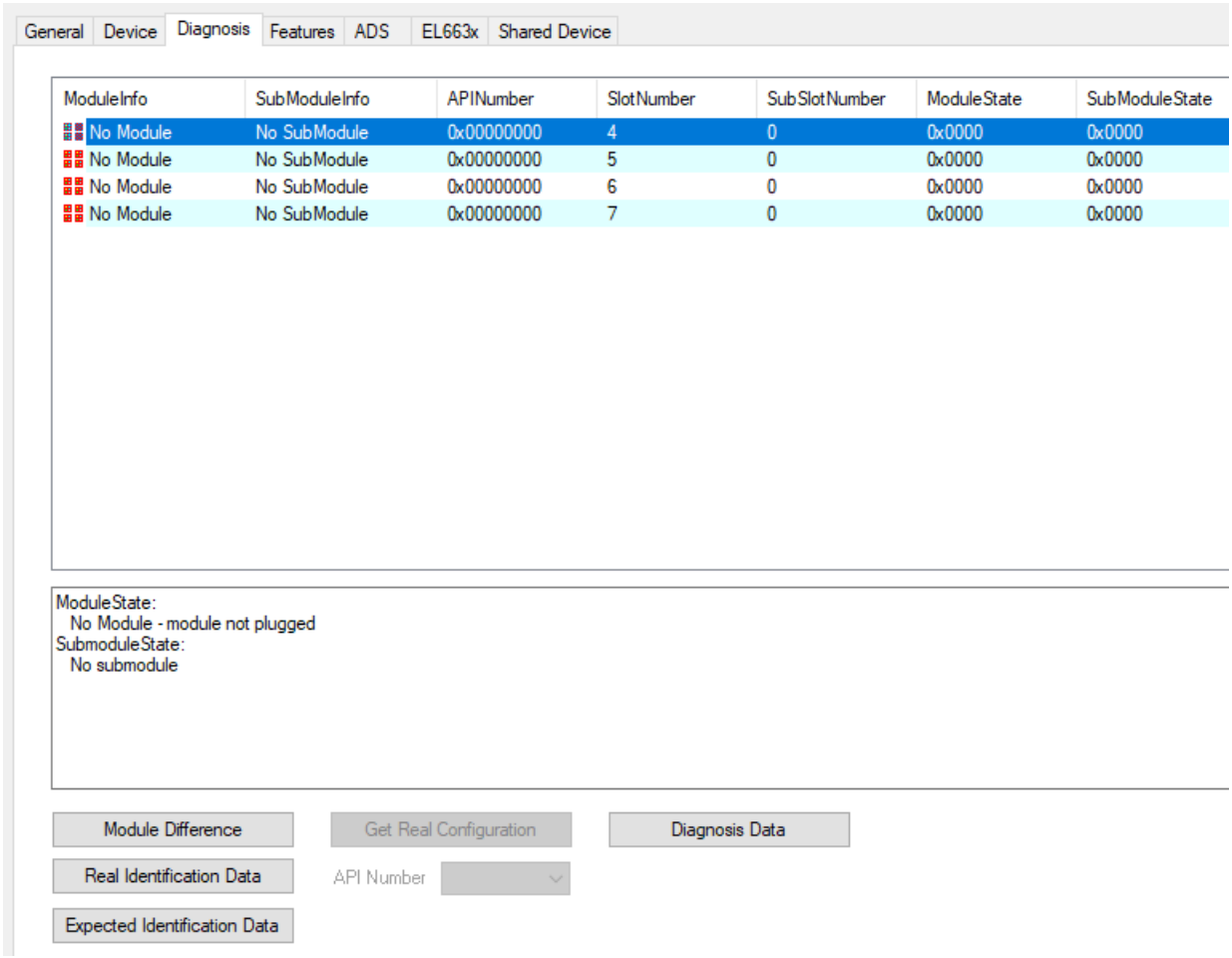
If a connection exists, the project planning can be checked on the **Diagnosis** tab. **Real Identification Data** indicates the existing modules within an AR at this level, **Expected Identification Data** indicates the expected modules (i.e. those projected in the controller). **Module Difference** shows the differences detected by the device during the target/actual comparison.



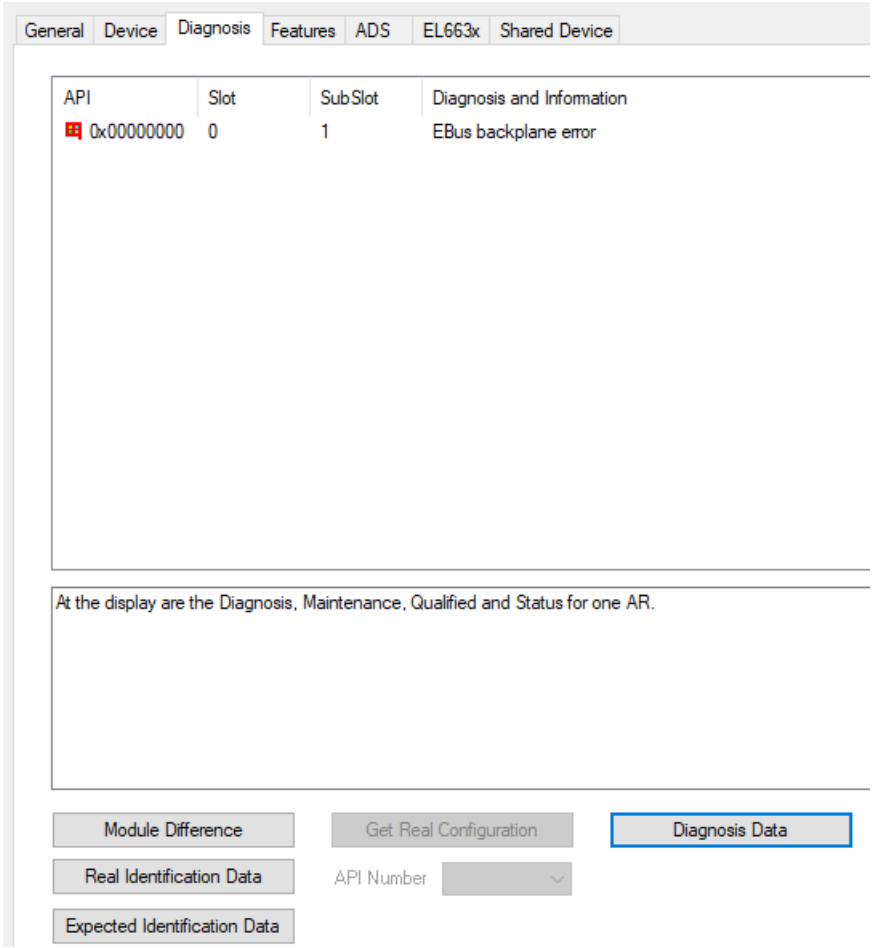
On the **Diagnosis** tab within the API the corresponding API can be selected from which information is to be obtained. If, for example, the PROFINET device is a drive, then this usually supports the Profidrive profile, which is identified in turn via API0x3A00. If the **Real Identification Data** is to be read from this API, for example, then this access takes place via the Profidrive profile.



The **Get Real Configuration** button becomes active within an API (except for drives). Here you can transfer the read-in data set into the current project. Note that modules that have already been created will be overwritten when doing this. This means that the links are lost, even in the case of previously correctly created modules. When displaying the module differences, additional information is displayed by marking the message.



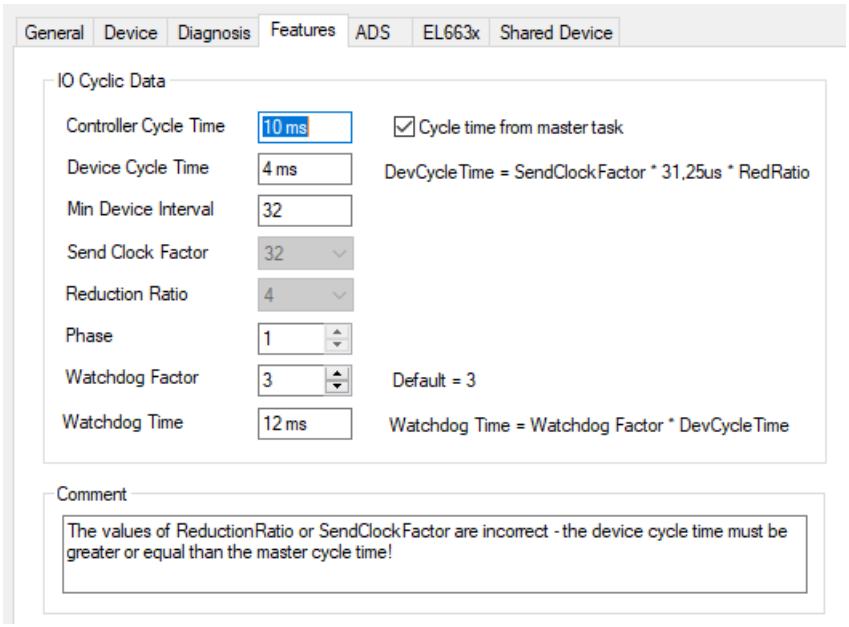
The available diagnosis can be read out by pressing the **Diagnosis Data** button. At device level all available diagnosis data for the existing AR is read out here.



Only two diagnosis parameters are displayed in the list, others are marked with "...". If the individual message is clicked, all available diagnosis information is displayed in the window below.

6.3.4 Features

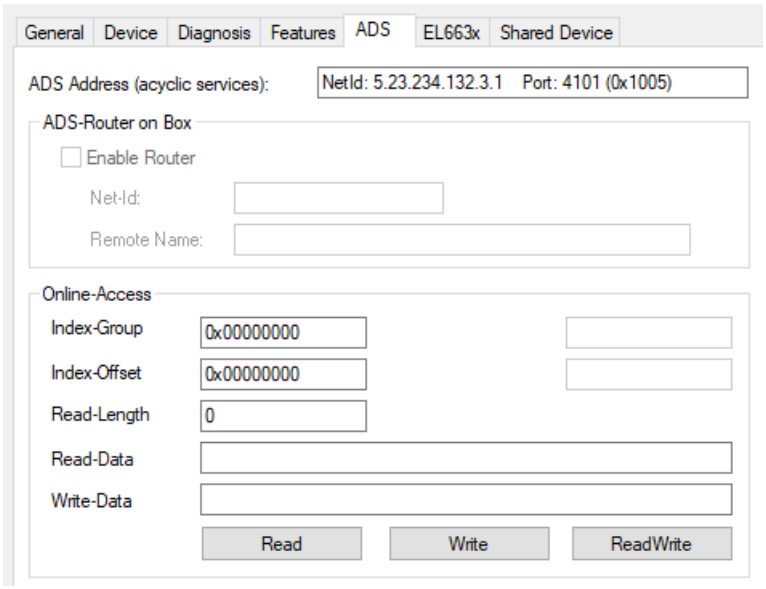
Various settings for the cycle time can be made on the **Features** tab. The cycle time of the controller must always correspond to a power of two for RTCLass1, starting at 1 ms (1, 2, 4, 8...). If an incorrect base time has been selected, this is indicated by a corresponding message. For RTClass3 the 1 ms base time can be divided again and again by two (down to min. 31.25 µs). The device cycle time can be changed via the exponents. The minimum is always the Controller Cycle Time, unless a larger minimum cycle time than that of the controller is defined in the GSDML. The maximum for RTClass1 is 512 ms. The **SendClockFactor** is fixed here as a time base to the value 32 (31.25 µs * 32 = 1 ms). The **Reduction Ratio** also refers to this, i.e. a factor of 4 means a cycle time of 4 ms. The transmission point can be shifted again within a cycle via the phase; i.e. where RR = 4 the phase can be 1 - 4. However, this value is only of importance in the case of a synchronized transmission.



The PROFINET **Watchdog Factor** can also be adjusted here. Each device monitors the input of the cyclic data based on this factor. If the factor is set to the default value (3) this means that, with an RR of 4, three cycles require 12 ms. Hence, a device reacts after 12 ms to missing telegrams (e.g. with an alarm and/or disconnection of the AR). The limits and values are recalculated each time when adjusting the individual factors.

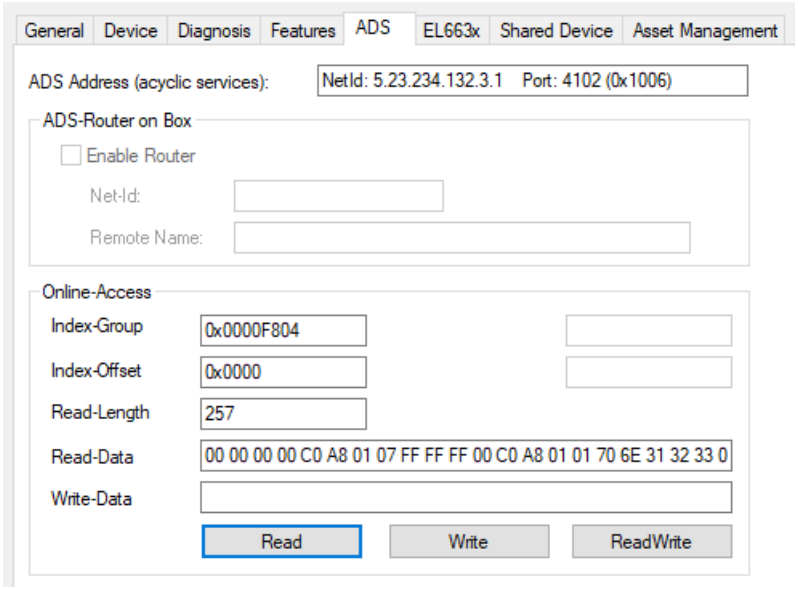
6.3.5 ADS

ADS messages can be sent directly from here. The NetId and the port are taken directly from the corresponding adapter.



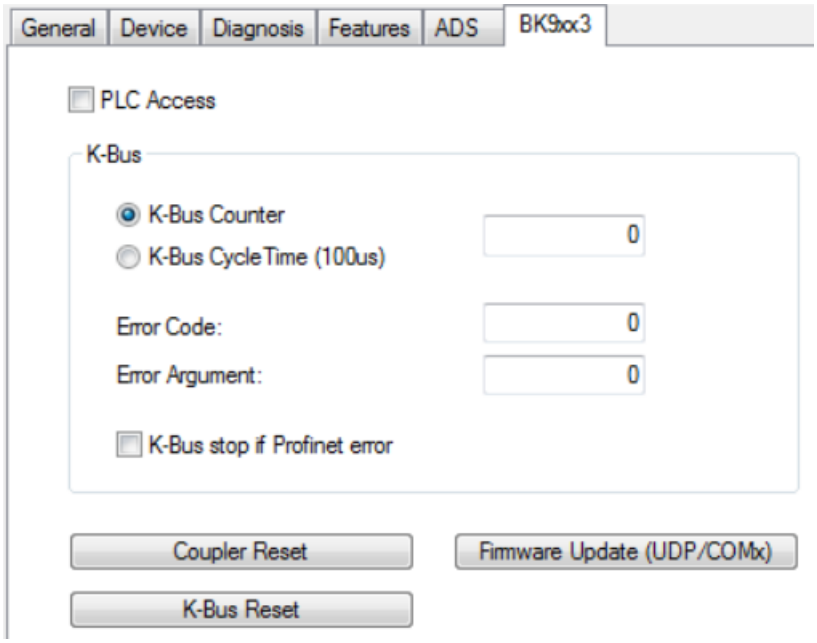
A large number of PROFINET functions can be triggered directly via the correct settings for index group and index offset. This includes, for example, the setting of alarms or record data.

Example: Read out the PROFINET name and the IP settings



6.3.6 BK9xx3

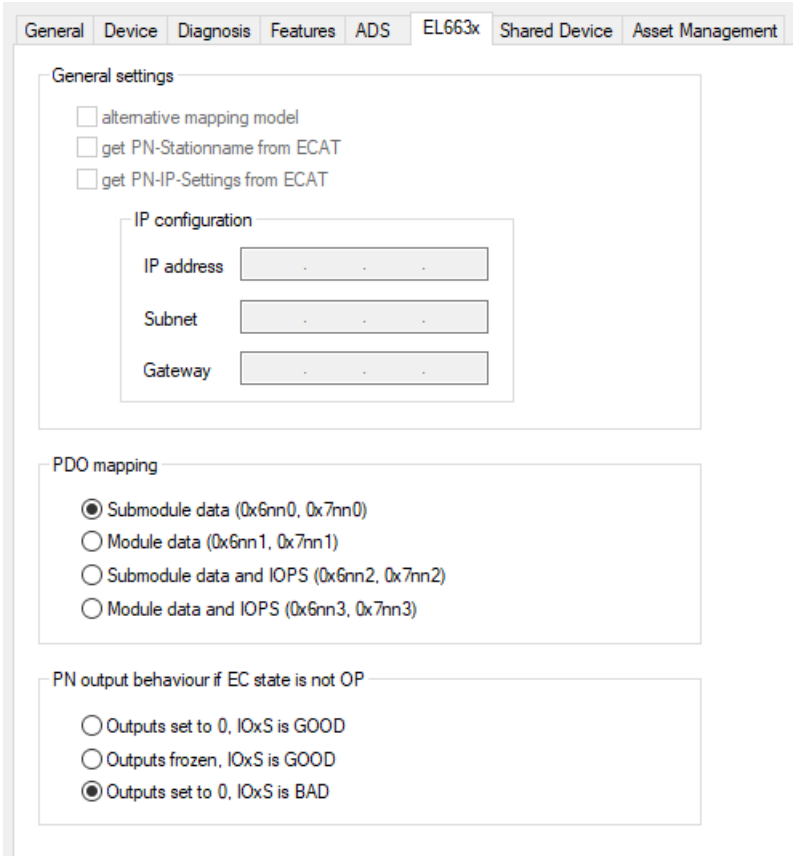
In the case of the Beckhoff K-bus Coupler (at present BK9103 or BK9053) that is not connected to an EL663x, an additional menu appears here.



This allows easy access to the cyclic process data in the DAP of the Bus Coupler. In addition, a firmware update from the System Manager to the Bus Coupler can be carried out via this menu. If the update takes place by IP it is important to ensure that the IP address is obtained via the DIP switches. If this is not the case the connection breaks off during the update, since the memory area of the IP settings is also formatted and rewritten.

6.3.7 EL663x

If the controller protocol is operated via an EL663x, then an additional menu appears on the devices.



Currently only the PDO mapping can be selected for the controller. This sets the form in which the PROFINET process data are mapped to the EtherCAT-side PDOs.

6.3.8 IRT Controller

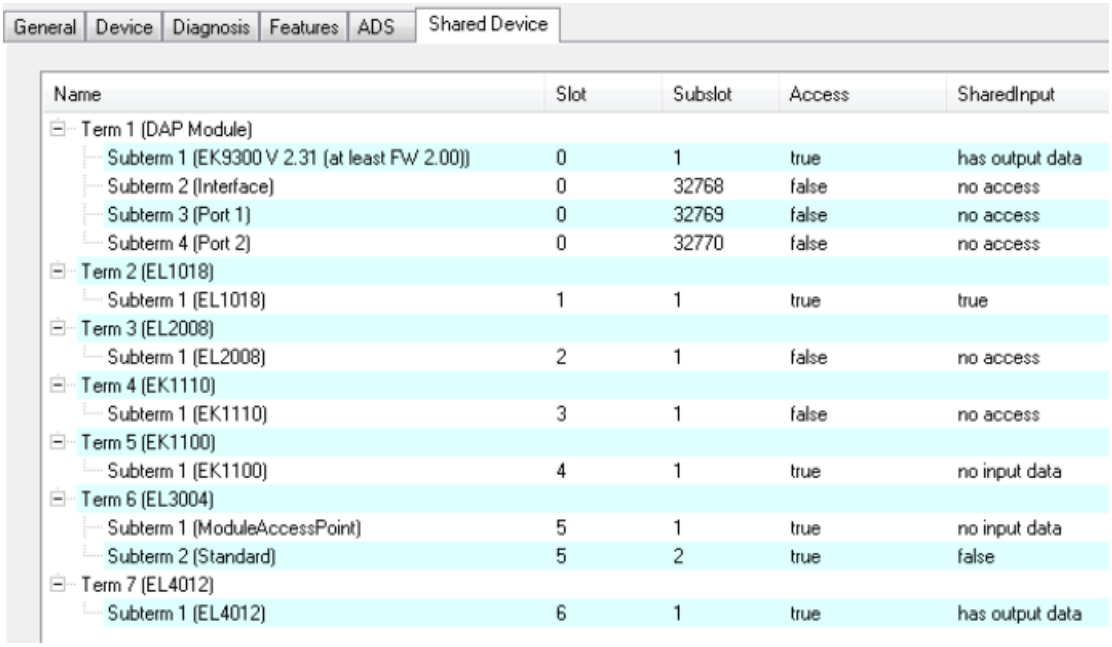
If the device is operated with an IRT-capable controller, an additional **Synchronization** menu appears.

Here it is possible to specify the Ti and To factors (**Time Ti, Time To**) for IRT-capable devices. This means the time during which the data in the device are valid within a cycle, or should be set to valid. The prerequisite is that this feature is also supported. The GSMDL supplies the information about this. There is always a basic cycle here (**base time**). A statement about the minimum possible time comes via the GSDML on the basis of a minimum factor. The upper limit of the factor is limited by the cycle time employed. The shortest possible time in which the data could be valid over PROFINET (always in reference to the cycle) is displayed via the **Time Input Valid** or **Time Output Valid** parameter.

6.3.9 Shared Device

The Shared Device feature is available from TwinCAT 2 Build 22.50 or TwinCAT 3 Build 4019.

The dialog appears if the device supports **Shared Device**. The information for this comes from the GSDML.



Name	Slot	Subslot	Access	SharedInput
Term 1 (DAP Module)				
Subterm 1 (EK9300 V 2.31 (at least FW 2.00))	0	1	true	has output data
Subterm 2 (Interface)	0	32768	false	no access
Subterm 3 (Port 1)	0	32769	false	no access
Subterm 4 (Port 2)	0	32770	false	no access
Term 2 (EL1018)				
Subterm 1 (EL1018)	1	1	true	true
Term 3 (EL2008)				
Subterm 1 (EL2008)	2	1	false	no access
Term 4 (EK1110)				
Subterm 1 (EK1110)	3	1	false	no access
Term 5 (EK1100)				
Subterm 1 (EK1100)	4	1	true	no input data
Term 6 (EL3004)				
Subterm 1 (ModuleAccessPoint)	5	1	true	no input data
Subterm 2 (Standard)	5	2	true	false
Term 7 (EL4012)				
Subterm 1 (EL4012)	6	1	true	has output data

There is an option here to allow or forbid the controller to access the individual submodules. By default the controller may access all submodules; if SharedInput is supported it is switched off.

The text messages for SharedInput have the following meanings:

"not supported" - SharedInput is not supported by the device (info from the GSDML)

"has output data" - the submodule has outputs - activation of SharedInput not possible

"no input data" - the submodule has no inputs (and also no outputs)

"no access" - access is blocked

"true" or "false" - set value for SharedInput

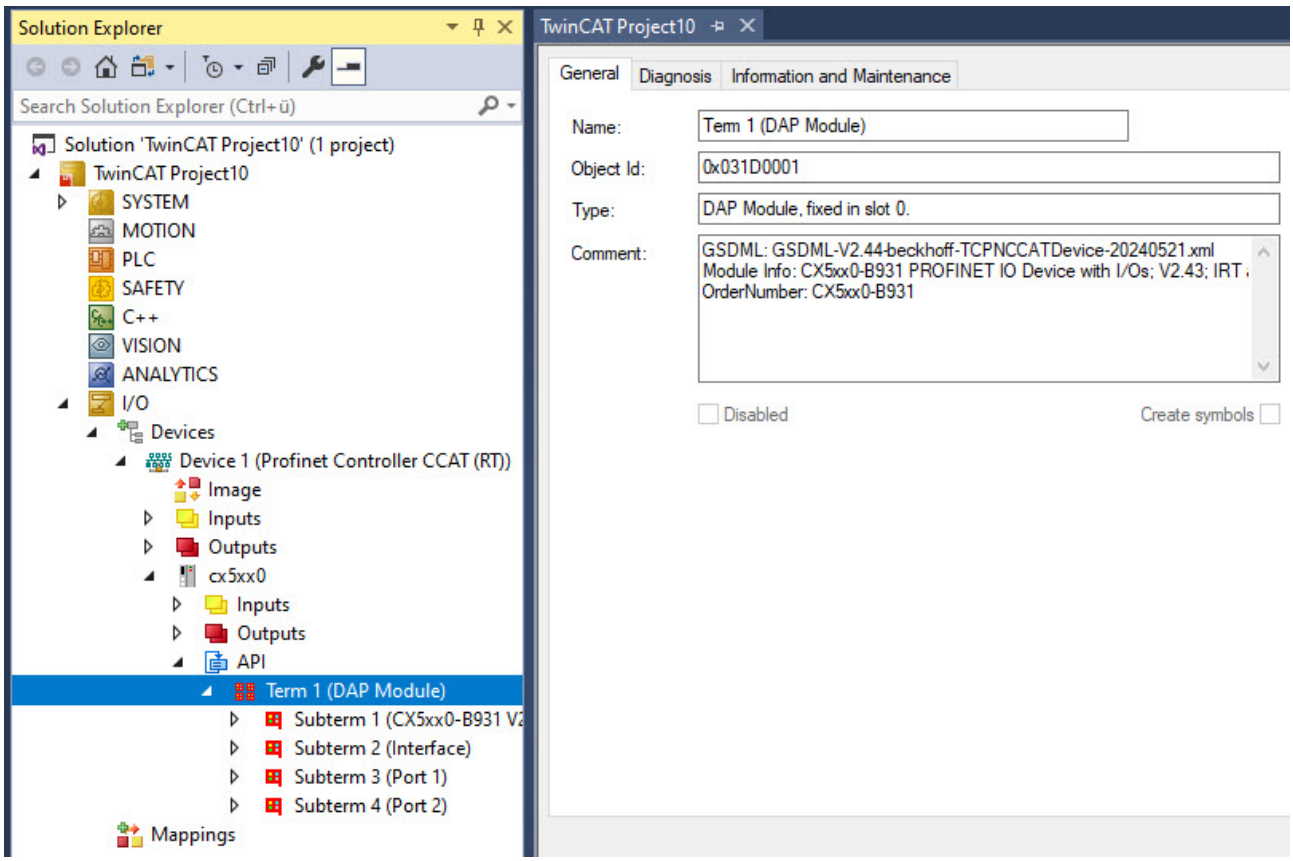
The settings can be changed by double-clicking on the individual submodules. If the access to a port or interface submodule is changed, then it is changed for all ports or interfaces.

6.3.10 Module and submodules

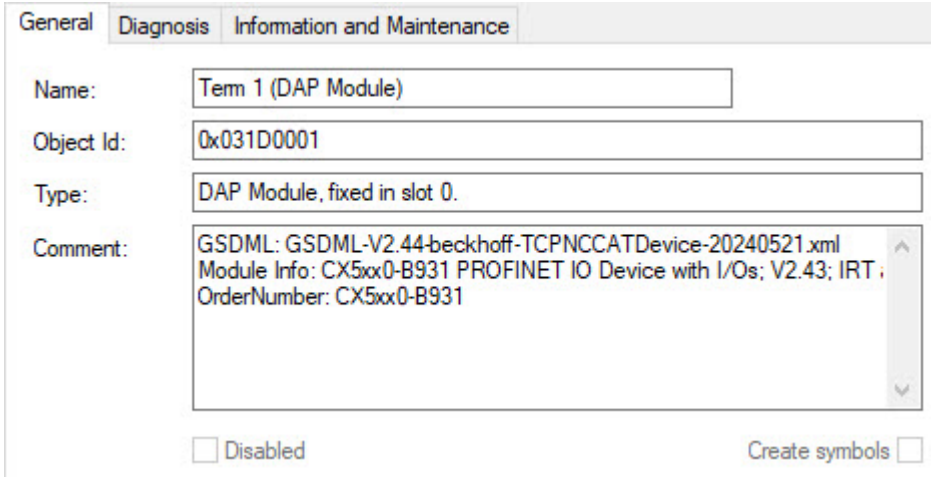
6.3.10.1 DAP (Device Access Point) modules

The device model chosen for PROFINET is the decentralized periphery, which is familiar from PROFIBUS DP.

The slots of the modular devices are represented via slots and subslots.



6.3.10.1.1 General



Name

Identifiers for the PPROFINET module

Object Id

Identification number of the DAP object in the TwinCAT object context.

Type

Displays the selected object type and its properties.

Comment

Freely editable comment to describe the object used

6.3.10.1.2 Diagnosis

Diagnosis at submodule level [▶ 76]

6.3.10.1.3 Information and Maintenance

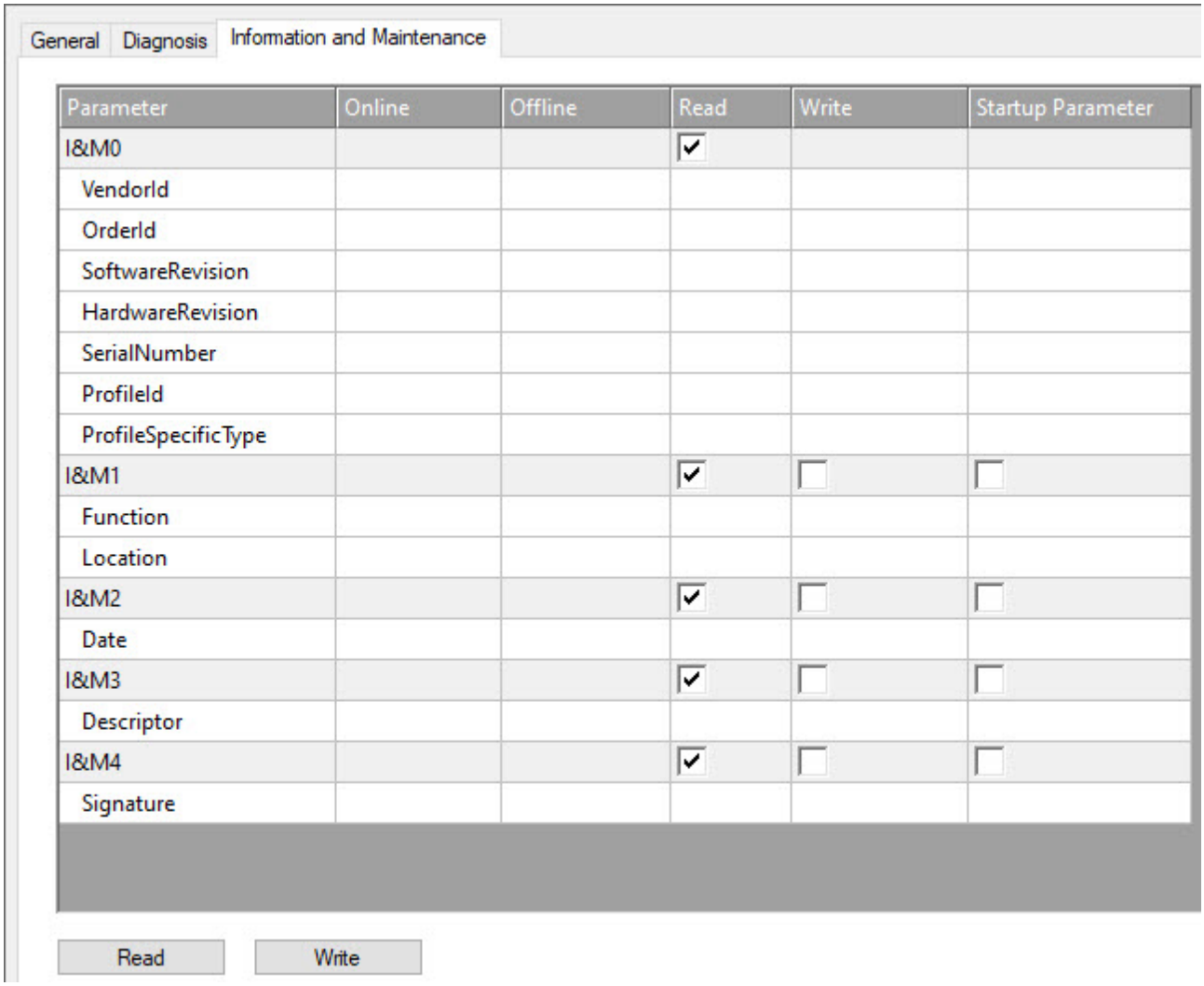


Fig. 9: If the device supports I&M according to GSDML, this dialog is displayed on the corresponding module (usually the DAP)

6.3.10.2 Submodule level

PROFINET currently distinguishes between 4 types of submodule.

Virtual submodules: The virtual submodules are always permanently connected to a module. This means that when a module is inserted, the co-defined virtual submodules are also always inserted on the specified subslot. This kind of submodule is presently the commonest method.

Real submodules: Here there is a possibility to select the pluggable submodules from a submodule list and to append them to the module. The necessary information is procured from the GSDML. In TwinCAT a module can be selected from such a list with the right mouse button (provided this is supported by the device).

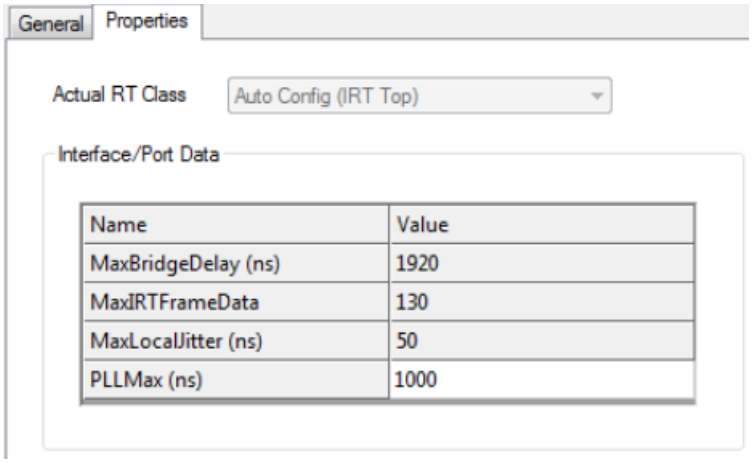
Interface submodules: Device-specific properties are defined in the interfaces submodules. These can be, for example, additionally supported protocols, timing properties, supported MIBs etc.

Port submodules: The physical properties of a network port are reproduced in such a submodule.

6.3.10.3 Interface submodule

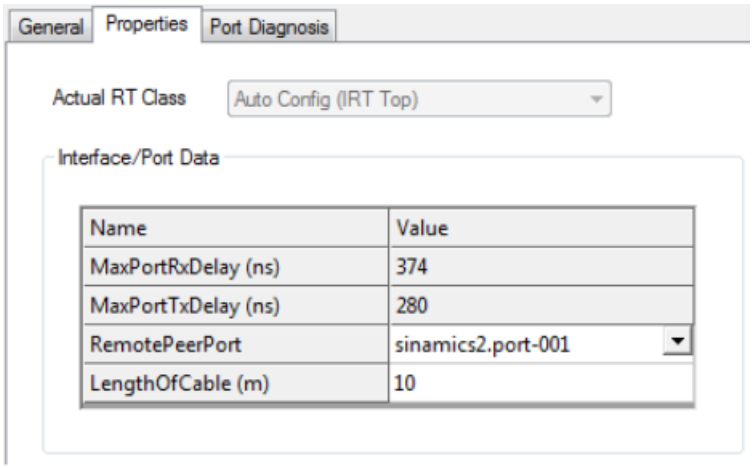
The type of communication can always be set on the interface submodel (at present RTClass1 or RTClass3). The only exception is in the case that a generally valid RTClass was set via the **Auto Config...** menu.

If communication takes place over RTClass3, then the PLL window can additionally be set at the interface.

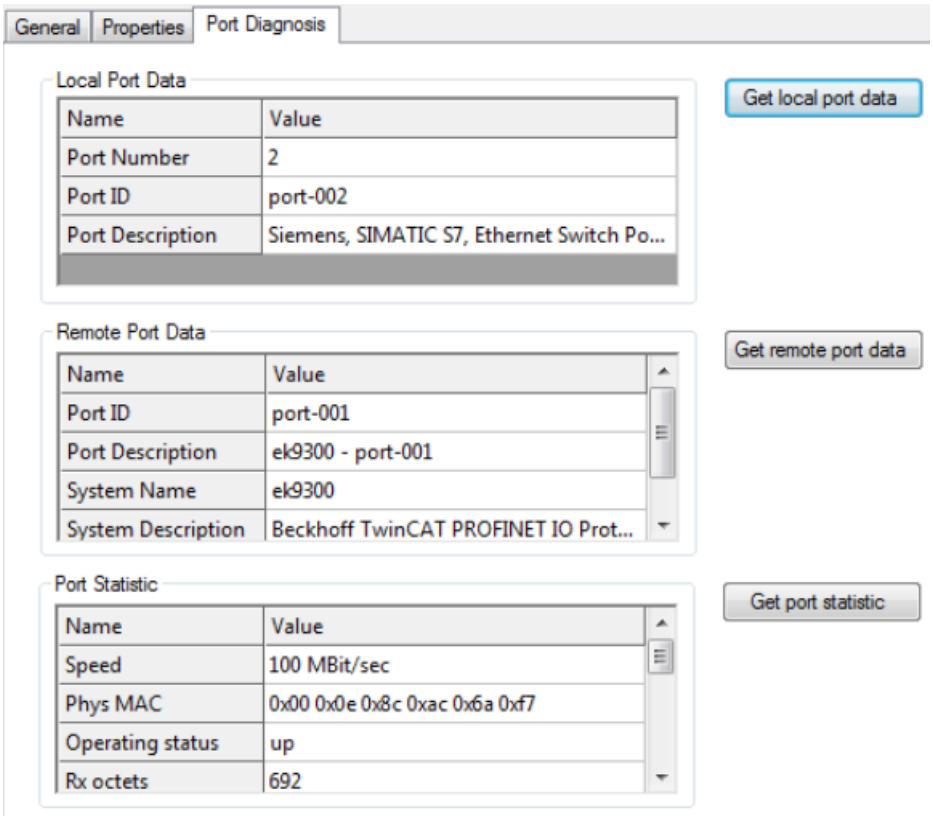


6.3.10.4 Port Submodule

In the **Properties** tab you can make port-specific settings. The menu of possible settings depends on the RTClass used.



Here you can read out some port properties.

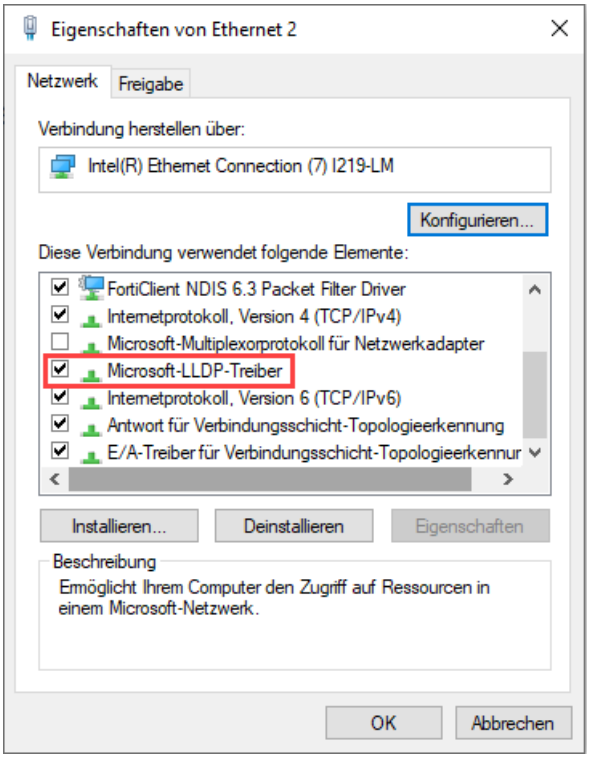


The information here is subdivided into local port information and remote port properties. In PROFINET the LLDP protocol (IEEE Std 802.1AB) is mandatory from conformance class A (CCA). The devices exchange neighborhood IDs via this protocol, so that each port is known to its neighbor. Furthermore, the Simple Network Management Protocol (SNMP) can be used as an aid at this point. On opening the **Port Diagnosis** tab, TwinCAT acts as a Network Management Station (NMS) and collects the required device information via SNMP. The previous images shows that the local port 1 of the BK9053 is linked to port 2 of the BK9103. For correct topology recognition it is important that only devices are present in the segment that also support the LLDP protocol (this is also applies to switches!).

Undesirable behavior

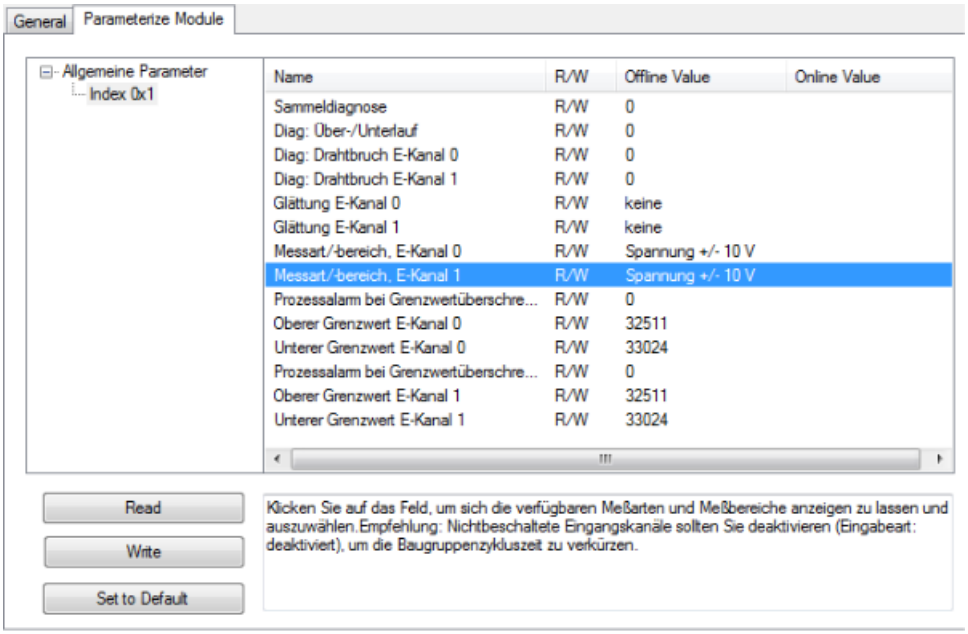
Unwanted behavior may occur in Windows 10. This is reflected in inconsistencies in topology detection.

Inconsistencies in topology detection.
Windows 10 provides an LLDP driver which is active by default. The PROFINET device also contains an LLDP driver. The system then sends two LLDP telegrams from one port to the remaining nodes on the bus. These remaining devices also expect two ports due to two LLDP telegrams, which do not exist.
To prevent this behavior, disable the Windows LLDP driver. To do this, uncheck **Microsoft-LLDP-Driver** (see red rectangle).



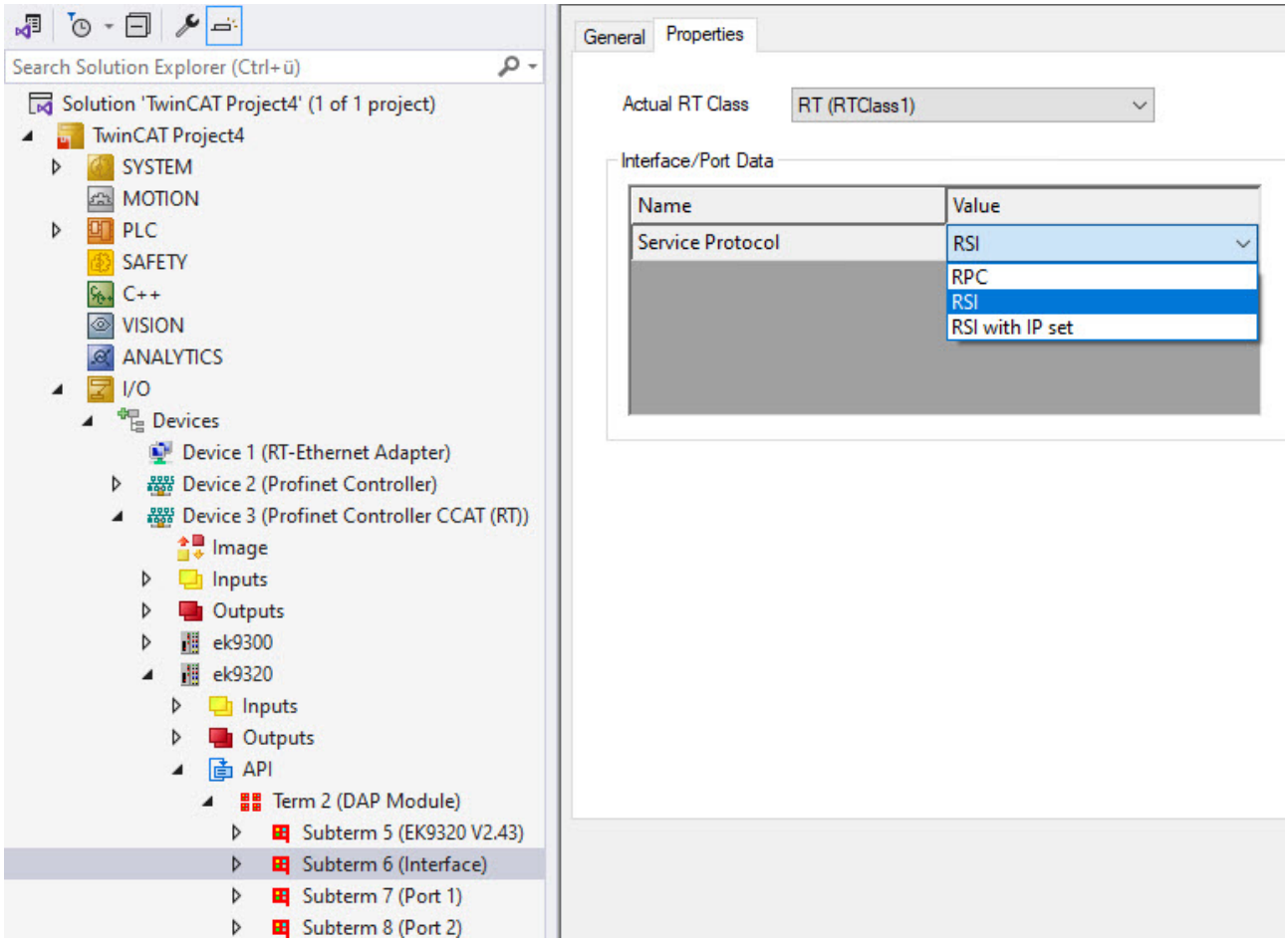
6.3.10.5 Real / virtual submodules

If these submodules have parameterization data they will be displayed as shown in the illustration below.



Selection can be made here between the individual indices. The data can be read and/or written depending on the access method. The online values are updated when reading back. If a single index is selected, clicking **Set to Default** will set all values within an index to default; selecting individual values will reset only those values. The writable values can be changed by double-clicking on the corresponding line.

6.3.11 Interface settings



Is only available on the EK9320 and the EL6633.

Service protocol:

The service protocol used for the acyclic services can be manually selected here. The SystemManager recognizes which protocol is supported based on the GSDML of the device.

RPC:

RPC uses the UDP protocol as the transport layer. This requires the assignment of an IP address, which is limited to IPv4 in PROFINET.

RSI (default):

RSI only uses layer 2 communication, all services that are used with RPC via UDP are routed via layer 2 communication with RSI. This means that no IP address is required and none is assigned via PROFINET by default.

RSI with IP set:

RSI is used as the service protocol, but the PROFINET device is also assigned an IP address by the controller via DCP during connection start-up. This means that the use of IP protocols or services (e.g. SNMP or WebServer) is still possible without the need for an additional "IP assignment mechanism".

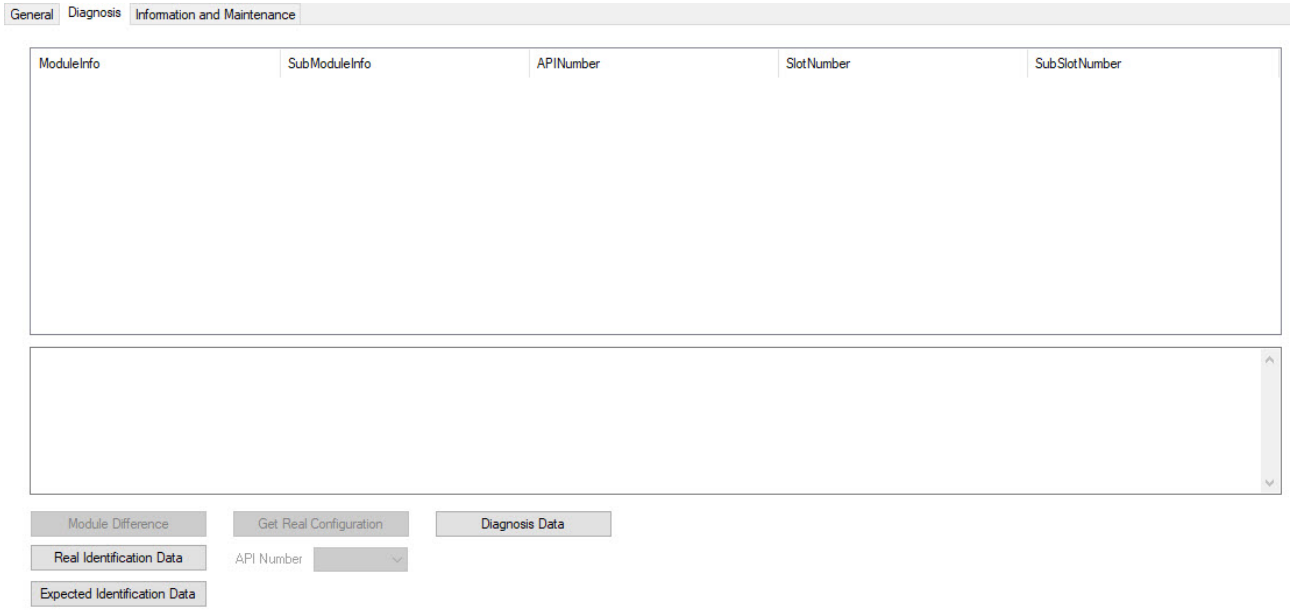
6.4 PROFINET device diagnostics

6.4.1 Diagnosis at module level

At module level there is an option on the Diagnosis tab to compare the nominal and actual data for the respective module. In addition the existing diagnosis for the module can be read out.

Comparison of nominal and actual population [▶ 62]

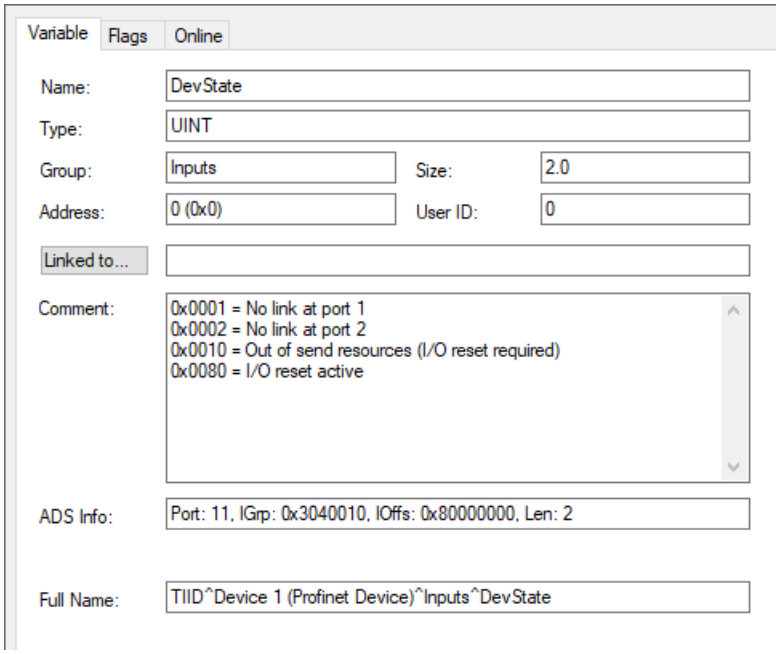
6.4.2 Diagnosis at submodule level



In general, the submodules have the same diagnostic properties as the modules, i.e., currently, it is only possible to read out the target and actual configuration in TwinCAT. The order of the subslot numbers is not necessarily the same as the order in the TwinCAT project. For example, in the DAP the starting point is always the Interface Submodule (ISM), but the subslot number of the ISM is defined in the GSDML and starts at 0x8000. There are 16 possible interfaces (0x8x00), each with up to 256 ports (0x80xx). An ISM is followed by the port submodules with the subslot number referred to above.

6.4.3 Cyclic diagnostics

Cyclic process data can be found directly below the PROFINET controller protocol. This data is only exchanged between the PROFINET driver and TwinCAT 3. It provides general information about the status of the device and the configured boxes.



The process data **DevState**. contains information about the physical communication status of the device, such as the link status or whether the sender resources are still adequate.

The output process data **DevCtrl** currently has no function.

7 TwinCAT library and programming

7.1 Overview

There are ready-to-use function blocks for the use of the PROFINET controller. The library contains further function blocks for the EL6631-0010 PROFINET Device Terminal, but these are not part of this documentation.

(https://infosys.beckhoff.com/content/1033/el6633/Resources/2595517963.zip)

I&M functions

Block I&M functions	Meaning	Description
FB_PN_IM0_READ [▶ 79]	Read the I&M function 0	Supplement and EL663x
FB_PN_IM1_READ [▶ 80]	Read the I&M function 1	Supplement and EL663x
FB_PN_IM2_READ [▶ 82]	Read the I&M function 2	Supplement and EL663x
FB_PN_IM3_READ [▶ 84]	Read the I&M function 3	Supplement and EL663x
FB_PN_IM4_READ [▶ 87]	Read the I&M function 4	Supplement and EL663x
FB_PN_IM1_WRITE [▶ 81]	Write the I&M function 1	Supplement and EL663x
FB_PN_IM2_WRITE [▶ 83]	Write the I&M function 2	Supplement and EL663x
FB_PN_IM3_WRITE [▶ 85]	Write the I&M function 3	Supplement and EL663x
FB_PN_IM4_WRITE [▶ 87]	Write the I&M function 4	Supplement and EL663x

Statistics and diagnostic information

Block I&M functions	Meaning	Description
FB_PN_GET_PORT_STATISTIC [▶ 88]	Read the port statistics	Supplement and EL663x
FB_PN_READ_PORT_DIAG [▶ 89]	Read the port diagnosis	Supplement and EL663x

Development environment	Target platform	PLC libraries to include
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcEtherCAT.lib TcPlcIoFunction.lib TcUtilities.lib TcSystem.lib TcBase.lib

7.2 Functions

7.2.1 I&M

7.2.1.1 FUNCTION_BLOCK FB_PN_IM0_READ

Fig. 10: FUNCTION_BLOCK FB_PN_IM0_READ

Using this function block the PROFINET controller reads all I&M 0 data (Identification & Maintenance) from a device referenced via the *Port* input.
The frame structure of the I&M0 function corresponds to the index 0xAFF0 [▶ 92] according to PROFINET standard.

VAR_INPUT

```
VAR_INPUT
    bStart   : BOOL;
    NETID    : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT     : T_AmsPort; (* Port used by Controller to communicate with Device *)
END_VAR
```

bStart: The block is activated by a rising edge on this input

NETID: AMS Net ID des Controllers

PORT: Port via which the controller communicates with the device (port = Device ID + 1000 hex)

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy    : BOOL;
    IM_AFF0  : str_IM_0xAFF0;
    bError   : BOOL;
    iErrorID     : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

IM_AFF0: Output of the I&M0 frame supplied by the device in a structure. str_IM_0xAFF0.

bError: In the event of an error during the command transfer, this output is set once the *bBusy* output has been reset.

iErrorID: Supplies an ADS error number when the output *bError* is set.

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcProfinetDiag.Lib

7.2.1.2 FUNCTION_BLOCK FB_PN_IM1_READ

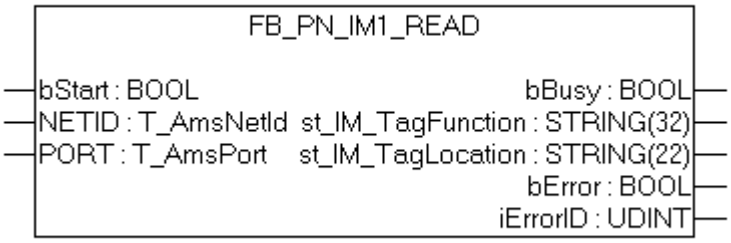


Fig. 11: FUNCTION_BLOCK FB_PN_IM1_READ

Using this function block the PROFINET controller reads all I&M1 data (Information & Maintenance) from a device referenced via the *Port* input.
The frame structure of the I&M1 function corresponds to the index 0xAFF1 [▶ 92] according to PROFINET standard.

VAR_INPUT

```
VAR_INPUT
    bStart  : BOOL;
    NETID   : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT    : T_AmsPort; (* Port used by Controller to communicate with Device *)
END_VAR
```

bStart: The block is activated by a rising edge on this input

NETID: AMS Net ID of the controller

PORT: Port via which the controller communicates with the device (port = Device ID + 1000 hex)

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    st_IM_TagFunction : STRING(32);
    st_IM_TagLocation : STRING(22);
    bError      : BOOL;
    iErrorID        : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

st_IM_TagFunction: label read out for the function of the device.

st_IM_TagLocation: label read out for the installation site of the device.

bError: In the event of an error during the command transfer, this output is set once the *bBusy* output has been reset.

iErrorID: Supplies an ADS error number when the output *bError* is set.

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcProfinetDiag.Lib

7.2.1.3 FUNCTION_BLOCK FB_PN_IM1_WRITE

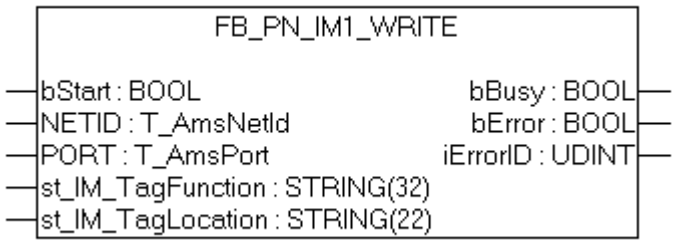


Fig. 12: FUNCTION_BLOCK FB_PN_IM1_WRITE

Using this function block the PROFINET controller writes all I&M1 data (Identification & Maintenance) data to a device referenced via the *Port* input.
The frame structure of the I&M1 function corresponds to the index 0xAFF1 [▶ 92] according to PROFINET standard.

VAR_INPUT

```
VAR_INPUT
    bStart      : BOOL;
    NETID    : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT     : T_AmsPort; (* Port used by Controller to communicate with Device *)
    st_IM_TagFunction     : STRING(32);
    st_IM_TagLocation     : STRING(22);
END_VAR
```

bStart: The function block is activated by a positive edge at this input

NETID: AMS Net ID of the controller

PORT: Port via which the controller communicates with the device (port = Device ID + 1000 hex)

st_IM_TagFunction: With this string the functional description is saved to the device.

st_IM_TagLocation: With this string the installation site is saved to the device.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    iErrorID        : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

bError: In the event of an error during the command transfer, this output is set once the *bBusy* output has been reset.

iErrorID: Supplies an ADS error number when the output *bError* is set..

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcProfinetDiag.Lib

7.2.1.4 FUNCTION_BLOCK FB_PN_IM2_READ

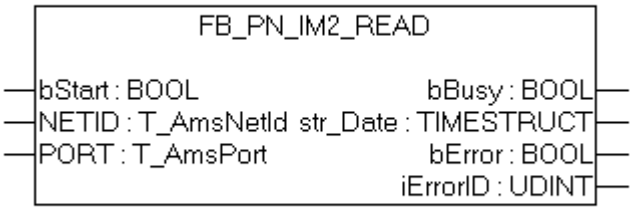


Fig. 13: FUNCTION_BLOCK FB_PN_IM2_READ

Using this function block the PROFINET controller reads all I&M 2 data (Identification & Maintenance) from a device referenced via the *Port* input.
The frame structure of the I&M2 function corresponds to the index 0xAFF2 [▶ 93] according to PROFINET standard.

VAR_INPUT

```
VAR_INPUT
    bStart  : BOOL;
    NETID   : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT    : T_AmsPort; (* Port used by Controller to communicate with Device *)
END_VAR
```

bStart: The function block is activated by a positive edge at this input

NETID: AMS Net ID of the controller

PORT: Port via which the controller communicates with the device (port = Device ID + 1000 hex)

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    str_Date        : TIMESTRUCT; (*YYYY-MM-DD HH:MM*)
    bError      : BOOL;
    iErrorID        : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

str_Date: Returns the date of installation of the device in the format < YYYY-MM-DD HH:MM >.

bError: In the event of an error during the command transfer, this output is set once the *bBusy* output has been reset.

iErrorID: Supplies an ADS error number when the output *bError* is set.

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcProfinetDiag.Lib

7.2.1.5 FUNCTION_BLOCK FB_PN_IM2_WRITE

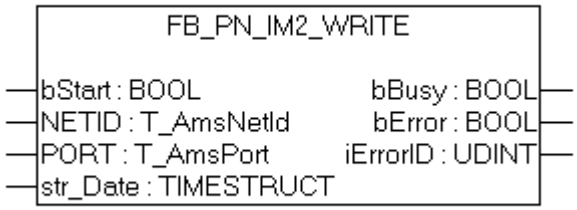


Fig. 14: FUNCTION_BLOCK FB_PN_IM2_WRITE

Using this function block the PROFINET controller writes all I&M 2 data (Identification & Maintenance) data to a device referenced via the *Port* input.
The frame structure of the I&M2 function corresponds to the index 0xAFF2 [▶ 93] according to PROFINET standard.

VAR_INPUT

```
VAR_INPUT
    bStart   : BOOL;
    NETID    : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT     : T_AmsPort; (* Port used by Controller to communicate with Device *)
    str_Date : TIMESTRUCT;(*YYYY-MM-DD HH:MM*)
END_VAR
```

bStart: The function block is activated by a positive edge at this input

NETID: AMS Net ID of the controller

PORT: Port via which the controller communicates with the device (port = Device ID + 1000hex)

str_Date : Writes a date (e.g. date of installation of the device) to the device in the format < YYYY-MM-DD HH:MM >.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    iErrorID        : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

bError: If an error should occur during the transmission of the command, this output is set after the *bBusy* output has been reset.

iErrorID: Supplies an ADS error number when the output *bError* is set.

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcProfinetDiag.Lib

7.2.1.6 FUNCTION_BLOCK FB_PN_IM3_READ

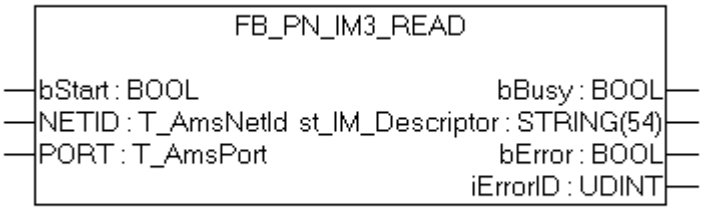


Fig. 15: FUNCTION_BLOCK FB_PN_IM3_READ

Using this function block the PROFINET controller reads all I&M3 data (Identification & Maintenance) from a device referenced via the *Port* input.
The frame structure of the I&M3 function corresponds to the index 0xAFF3 [▶ 93] according to PROFINET standard.

VAR_INPUT

```
VAR_INPUT
    bStart  :
    BOOL; NETID   : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT    : T_AmsPort; (* Port used by Controller to communicate with Device *)
END_VAR
```

bStart: The function block is activated by a positive edge at this input

NETID: AMS Net ID of the controller

PORT: Port via which the controller communicates with the device (port = Device ID + 1000 hex)

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    st_IM_Descriptor    : STRING(54);
    bError      : BOOL;
    iErrorID        : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

st_IM_Descriptor: Returns the manufacturer description stored for the device.

bError: In the event of an error during the command transfer, this output is set once the *bBusy* output has been reset.

iErrorID: Supplies an ADS error number when the output *bError* is set.

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcProfinetDiag.Lib

7.2.1.7 FUNCTION_BLOCK FB_PN_IM3_WRITE

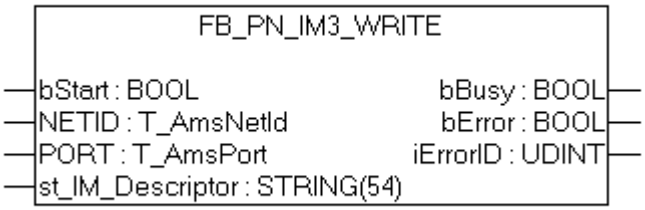


Fig. 16: FUNCTION_BLOCK FB_PN_IM3_WRITE

Using this function block the PROFINET controller writes all I&M3 data (Identification & Maintenance) data to a device referenced via the *Port* input.
The frame structure of the I&M3 function corresponds to the index 0xAFF3 [▶ 93] according to PROFINET standard.

VAR_INPUT

```
VAR_INPUT
    bStart      : BOOL;
    NETID   : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT    : T_AmsPort; (* Port used by Controller to communicate with Device *)
    st_IM_Descriptor     : STRING(54);
END_VAR
```

bStart: The function block is activated by a positive edge at this input

NETID: AMS Net ID of the controller

PORT: Port via which the controller communicates with the device (port = Device ID + 1000 hex)

st_IM_Descriptor: Returns the manufacturer description stored for the device.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    iErrorID        : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

bError: In the event of an error during the command transfer, this output is set once the *bBusy* output has been reset.

iErrorID: Supplies an ADS error number when the output *bError* is set.

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcProfinetDiag.Lib

7.2.1.8 FUNCTION_BLOCK FB_PN_IM4_READ

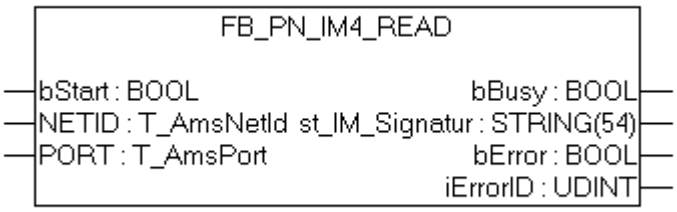


Fig. 17: FUNCTION_BLOCK FB_PN_IM4_READ

Using this function block the PROFINET controller reads all I&M4 data (Identification & Maintenance) from a device referenced via the *Port* input.
The frame structure of the I&M4 function corresponds to the index 0xAFF4 [▶ 93] according to PROFINET standard.

VAR_INPUT

```
VAR_INPUT
    bStart  : BOOL;
    NETID   : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT    : T_AmsPort; (* Port used by Controller to communicate with Device *)
END_VAR
```

bStart: The function block is activated by a positive edge at this input

NETID: AMS Net ID of the controller

PORT: Port via which the controller communicates with the device (port = Device ID + 1000 hex)

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    st_IM_Signatur  : STRING(54);
    bError      : BOOL;
    iErrorID        : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

st_IM_Signatur: Returns the manufacturer signature stored for the device.

bError: In the event of an error during the command transfer, this output is set once the *bBusy* output has been reset.

iErrorID: Supplies an ADS error number when the output *bError* is set.

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcProfinetDiag.Lib

7.2.1.9 FUNCTION_BLOCK FB_PN_IM4_WRITE

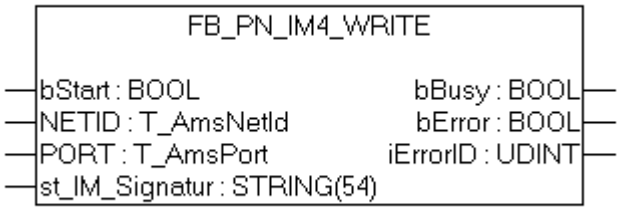


Fig. 18: FUNCTION_BLOCK FB_PN_IM4_WRITE

Using this function block the PROFINET controller writes all I&M4 data (Identification & Maintenance) data to a device referenced via the *Port* input.
The frame structure of the I&M4 function corresponds to the index 0xAFF4 [▶ 93] according to PROFINET standard.

VAR_INPUT

```
VAR_INPUT
    bStart      : BOOL;
    NETID   : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT    : T_AmsPort; (* Port used by Controller to communicate with Device *)
    st_IM_Signatur   : STRING(54);
END_VAR
```

bStart: The function block is activated by a positive edge at this input

NETID: AMS Net ID of the controller

PORT: Port via which the controller communicates with the device (port = Device ID + 1000 hex)

st_IM_Signatur: Signature of the manufacturer is written to the device.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    iErrorID        : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

bError: In the event of an error during the command transfer, this output is set once the *bBusy* output has been reset.

iErrorID: Supplies an ADS error number when the output *bError* is set.

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcProfinetDiag.Lib

7.2.2 Port

7.2.2.1 FUNCTION_BLOCK FB_PN_GET_PORT_STATISTIC

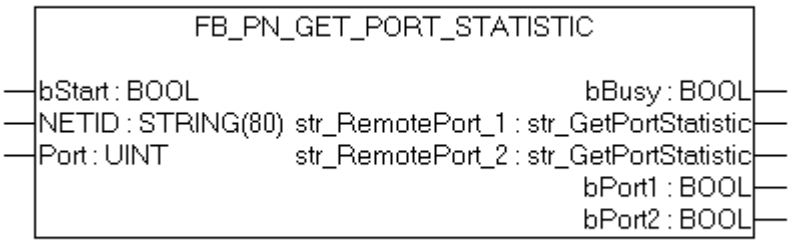


Fig. 19: FUNCTION_BLOCK FB_PN_GET_PORT_STATISTIC

When called, this module supplies the statistical data for the ports of a PROFINET device.

VAR_INPUT

```
VAR_INPUT
    bStart      : BOOL;
    NETID   : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT    : T_AmsPort; (* Port used by Controller to communicate with Device *)
END_VAR
```

bStart: The function block is activated by a positive edge at this input

NETID: AMS Net ID of the controller

PORT: Port via which the controller communicates with the device (port = Device ID + 1000 hex)

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    str_RemotePort_1    : str_GetPortStatistic [▶ 94];
    str_RemotePort_2    : str_GetPortStatistic [▶ 94];
    bPort1      : BOOL;
    bPort2      : BOOL;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

str_RemotePort_1: This structure contains the statistical data for Port 1.

str_RemotePort_2: This structure contains the statistical data for Port 2.

bPort1: Is TRUE if the port has a link.

bPort2: Is TRUE if the port has a link.

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcProfinetDiag.Lib

7.2.2.2 FUNCTION_BLOCK FB_PN_READ_PORT_DIAG

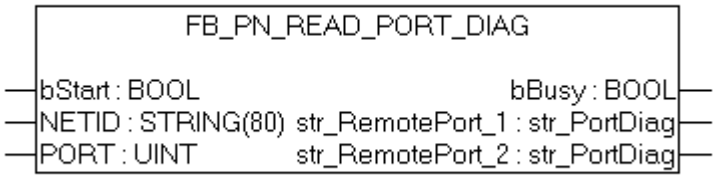


Fig. 20: FUNCTION_BLOCK FB_PN_READ_PORT_DIAG

This block calls the port diagnostic information of a PROFINET device.

VAR_INPUT

```
VAR_INPUT
    bStart      : BOOL;
    NETID    : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT     : T_AmsPort; (* Port used by Controller to communicate with Device *)
END_VAR
```

bStart: The function block is activated by a positive edge at this input

NETID: AMS Net ID of the controller

PORT: Port via which the controller communicates with the device (port = Device ID + 1000 hex)

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    str_RemotePort_1     : str_PortDiag [▶ 95];
    str_RemotePort_2     : str_PortDiag [▶ 95];
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

str_RemotePort_1: This structure contains the diagnostic information for Port 1.

str_RemotePort_2: This structure contains the diagnostic information for Port 2.

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcPROFINETDiag.Lib

7.2.3 AlarmDiag

7.2.3.1 FUNCTION_BLOCK FB_PN_ALARM_DIAG

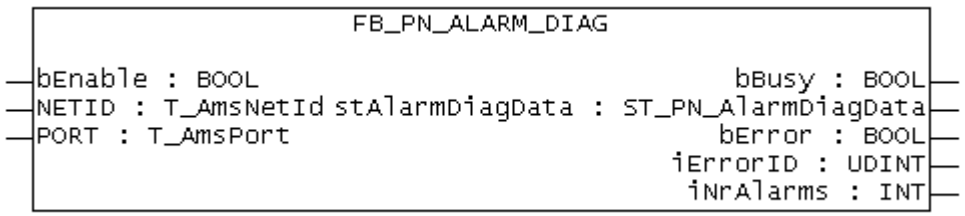


Fig. 21: FUNCTION_BLOCK FB_PN_ALARM_DIAG

Diagnosis alarms can be read out using this function block. Each instance of this function block makes a PLC input available ("PnIoBoxDiag"). This input must be linked to the “PnIoBoxDiag” input of the device that is to be evaluated. After successful reading of the diagnosis alarms/warnings, the alarm status of the device is reset. The function block must be called once for each PROFINET device. A running index (iNrAlarms) indicates how many diagnosis alarms have been read from the buffer.

VAR_INPUT

```
VAR_INPUT
    bEnable : BOOL;
    NETID   : T_AmsNetId;(* AMS Net ID from Controller *)
    PORT    : T_AmsPort; (* Port used by Controller to communicate with Device *)
END_VAR
```

bEnable: Activation of the function block

NETID: AMS Net ID of the controller

PORT: Port via which the controller communicates with the device (port = Device ID + 1000 hex)

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    stAlarmDiagData : ST_PN_AlarmDiagData;
    bError      : BOOL;
    iErrorID        : UDINT;
    iNrAlarms           : INT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until a feedback is received. While Busy = TRUE, no new command will be accepted at the inputs.

stAlarmDiagData: Diagnosis messages are output via this structure. An alarm is output via the structure in each cycle as long as the status bit [0x0010 = at least one AlarmCR got a diagnosis alarm] is present at the PLC input.

bError: If an error should occur during the transmission of the command, this output is set after the *bBusy* output has been reset.

iErrorID: Supplies an ADS error number when the output *bError* is set.

iNrAlarms: Number of alarms last read out.

VAR

```
VAR
        PnIoBoxDiag AT %I* : WORD; (*Hardware Input*)
END_VAR
```

PnIoBoxDiag : Hardware input: this variable must be linked to the PROFINET device. A change of state of this variable indicates to the PLC program that there are new diagnosis alarms in the linked PROFINET device.

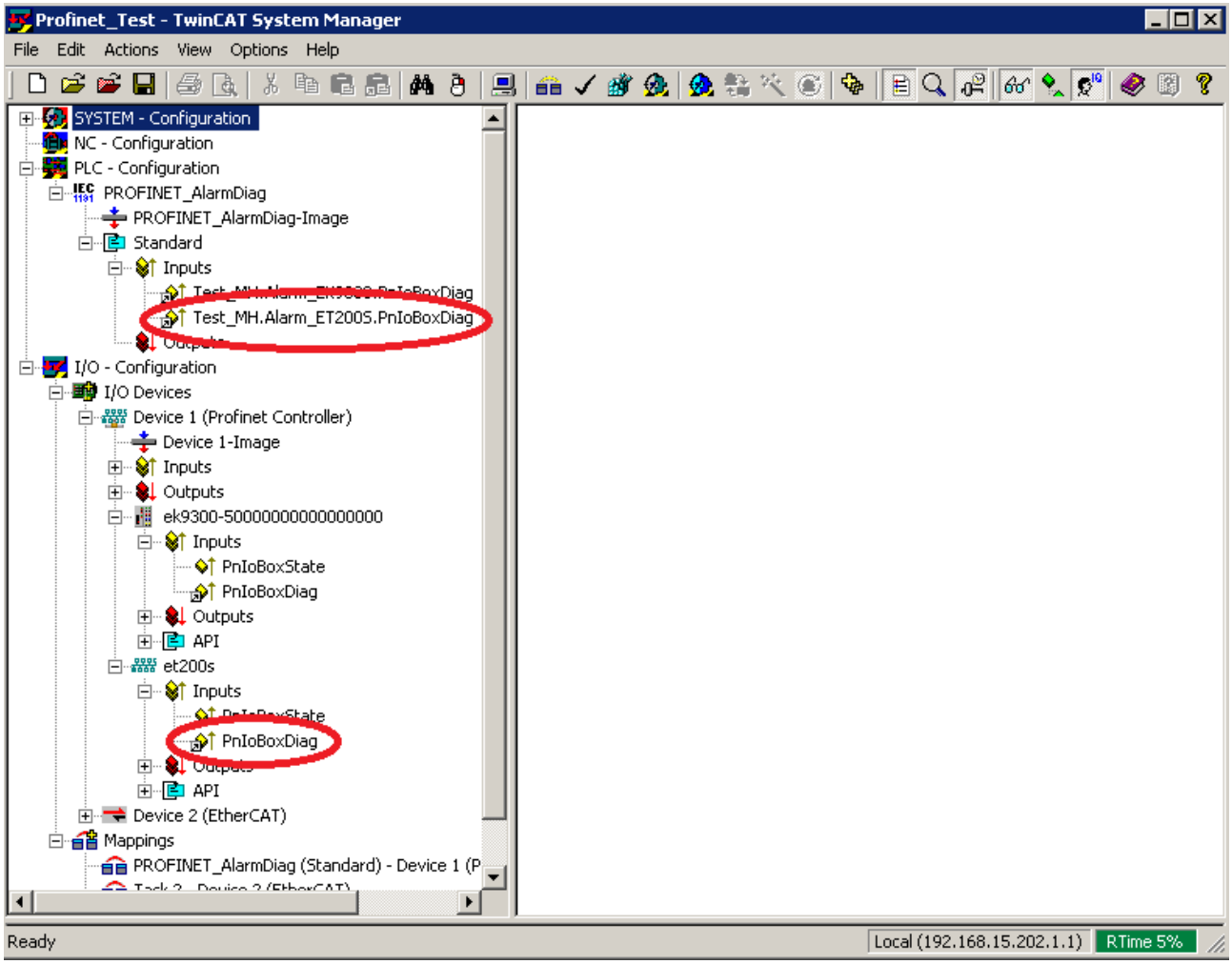


Fig. 22: Linking of the variables in the TwinCAT tree

Development environment	Target platform	PLC libraries to be linked
TwinCAT v2.11.0 R3	PC or CX (x86, ARM)	TcProfinetDiag.Lib

7.3 Data Structures

7.3.1 I&M

7.3.1.1 str_SW_Rec

The data structure **str_IM_0xAFF0** maps the structure of the I&M0 frame in the PLC. Which contains information that is permanently stored in PROFINET devices.

```
TYPE str_IM_0xAFF0 :

STRUCT
    nBlockTyp       : WORD;
    nBlockLen       : WORD;
    nBlockVersion   : WORD;
    nVendorID       : WORD;
    cOrderID        : STRING(21);
    cSerialNumber   : STRING(17);
    nHW_Rev         : WORD;
    strSW_Rev       : str_SW_Rec;
    nRevCount       : WORD;
    nProfileID      : WORD;
    nProfileSpecType    : WORD;
    arIM_Version    : ARRAY[0..1] OF BYTE;
    nSupport        : WORD;
END_STRUCT

END_TYPE
```

The data structure **str_SW_REC** contains the software version of the PROFINET device.

```
TYPE str_SW_Rec :

STRUCT
    cSWRevPrefix    :STRING(2);
    nSWRevFuncEnhance       :BYTE;
    nSWRevBugFix    :BYTE;
    nSWRevIntCha    :BYTE;
END_STRUCT

END_TYPE
```

7.3.1.2 str_IM_0xAFF1

The data structure **str_IM_0xAFF1** maps the structure of the I&M1 frame in the PLC. This structure is used both for writing to, and for reading from a PROFINET device.

```
TYPE str_IM_0xAFF1 :

STRUCT
    nBlockTyp       : WORD;
    nBlockLen       : WORD;
    nBlockVersion   : WORD;
    st_IM_TagFunction : STRING(32);
    st_IM_TagLocation : STRING(22);
END_STRUCT

END_TYPE
```

7.3.1.3 str_IM_0xAFF2

The data structure **str_IM_0xAFF2** maps the structure of the I&M2 frame in the PLC. This structure is used both for writing to, and for reading from a PROFINET device.

```
TYPE str_IM_0xAFF1 :

STRUCT
    nBlockTyp       : WORD;
    nBlockLen       : WORD;
    nBlockVersion   : WORD;
    st_IM_Date      : STRING(16);
END_STRUCT

END_TYPE
```

7.3.1.4 str_IM_0xAFF3

The data structure **str_IM_0xAFF3** maps the structure of the I&M3 frame in the PLC. This structure is used both for writing to, and for reading from a PROFINET device.

```
TYPE str_IM_0xAFF3 :

STRUCT
    nBlockTyp       : WORD;
    nBlockLen       : WORD;
    nBlockVersion   : WORD;
    st_IM_Descriptor    : STRING(54)
END_STRUCT

END_TYPE
```

7.3.1.5 str_IM_0xAFF4

The data structure **str_IM_0xAFF4** maps the structure of the I&M4 frame in the PLC. This structure is used both for writing to, and for reading from a PROFINET device.

```
TYPE str_IM_0xAFF3 :

STRUCT
    nBlockTyp       : WORD;
    nBlockLen       : WORD;
    nBlockVersion   : WORD;
    st_IM_Signatur  : STRING(54)
END_STRUCT

END_TYPE
```

7.3.2 Port

7.3.2.1 str_GetPortStatistic

All statistical information of a device is represented in the data structure **str_GetPortStatistic**.

```
TYPE str_GetPortStatisitc :

STRUCT
    Speed             : DWORD;
    PhyMAC            : STRING(50);
    OperatingStatus   : STRING(16);
    RxOctets          : DWORD;
    RxUniCastPackets  : DWORD;
    RxBadPackets      : DWORD;
    RxDroppedFrames   : DWORD;
    RxUnknownProtocol : DWORD;
    TxOctets          : DWORD;
    TxUniCastPackets  : DWORD;
    TxBadPackets      : DWORD;
    TxDroppedPackets  : DWORD;
END_STRUCT

END_TYPE
```

7.3.2.2 str_PortDiag

All port diagnostic information is represented in the data structure **str_PortDiag**.

```
TYPE str_PortDiag :

STRUCT
    PortId              : STRING(128);
    PortDescription     : STRING(128);
    SystemName          : STRING(128);
    SystemDescription   : STRING(128);
    ChassisId           : STRING(128);
END_STRUCT

END_TYPE
```

7.3.3 AlarmDiag

7.3.3.1 ST_PN_DiagMessage

The data structure **ST_PN_DiagMessage** contains the complete data stream of a diagnostic message that is sent by a PROFINET on request. This data stream is evaluated in the FB_PN_ALARM_DIAG function block and is copied to a readable structure.

```
TYPE ST_PN_DiagMessage :

STRUCT
    nFlags    : WORD;
    nTextID   : WORD;
    TimeStamp : ARRAY[0..7] OF BYTE;
    nData     : ARRAY[0..299] OF BYTE;
END_STRUCT

END_TYPE
```

7.3.3.2 ST_PN_Diag

The data structure **ST_PN_Diag** contains a diagnostic message from a terminal that is connected via a PN device and a controller.

```
TYPE str_PortDiag :

STRUCT
    strTimeStamp           : ARRAY[0..7] OF BYTE;
    nAPI                   : DWORD;
    nSlot                  : WORD;
    nSubSlot               : WORD;
    nAlarmType             : WORD;
    nAlarmSpecifier        : WORD;
    nUserStructIdentifier  : WORD;
    nChannelNumber         : WORD;
    nChannelErrorTyp       : WORD;
    nChannelProperties     : WORD;
    nExtChannelErrorTyp    : WORD;
    arSpare                : ARRAY [1..9] OF WORD;
    arUserSpecificData     : ARRAY [0..19] OF BYTE;
END_STRUCT

END_TYPE
```

The information content of the structure corresponds to that of the Diag History, which is displayed in the system manager.

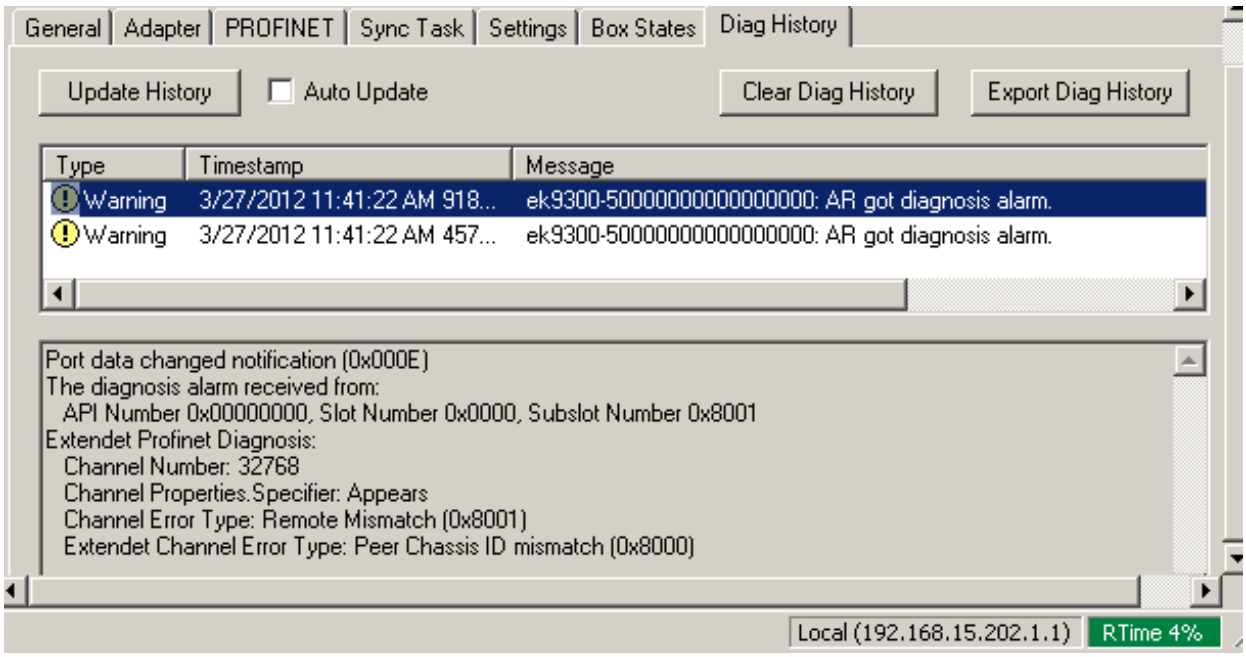


Fig. 23: "Diag History" tab

7.3.3.3 ST_PN_AlarmDiagData

The data structure **ST_PN_AlarmDiagData** contains the alarm diagnosis data record read from a device, including a time stamp that indicates when the event occurred and a flag that indicates that 'user-specific' data are present.

```
TYPE ST_PN_AlarmDiagData :

STRUCT
    ST_TimeStamp    : TIMESTRUCT;
    sNameOfStation  : STRING(20);
    ST_Diag         : ST_PN_Diag [▶ 95];
    bUserSpecData   : BOOL;
END_STRUCT

END_TYPE
```

7.3.4 Enumeration types for PROFINET alarms

E_PN_ALARM_TYP

The enumeration type **E_PN_ALARM_TYP** lists all PROFINET communication alarms.

```
TYPE E_PN_ALARM_TYP :
(
    PN_ALARM_RESERVE           :=0,
    PN_ALARM_DIAGNOSE_APPEARS  :=1,
    PN_ALARM_PROCESS           :=2,
    PN_ALARM_PULL              :=3,
    PN_ALARM_PLUG              :=4,
    PN_ALARM_STATUS            :=5,
    PN_ALARM_UPDATE            :=6,
    PN_ALARM_REDUNDANCY        :=7,
    PN_ALARM_Controlled_by_Supervisor          :=8,
    PN_ALARM_Released                          :=9,
    PN_ALARM_Plug_Wrong_Submodule              :=16#A,
    PN_ALARM_Diagnosis_Disappears              :=16#B,
    PN_ALARM_Multicast_Communication_Mismatch  :=16#C,
    PN_ALARM_Multicast                         :=16#D,
    PN_ALARM_STATUS_                           :=16#E,
    PN_ALARM_Sync_                             :=16#F,
    PN_ALARM_Isochronous_Mode_Problem_Notification :=16#10
);
END_TYPE
```

8 Appendix

8.1 FAQ

The following information answers frequently asked questions and gives hints for the configuration of the PROFINET system. If these are not observed, undesired behavior may occur. Here you will find approaches to diagnostics.

8.1.1 Device description file (GSDML) / DAP (DeviceAccessPoint)

Device description file (GSDML) / DAP (DeviceAccessPoint)

- Is the GSDML available on the system?
- Do the versions of both systems match?
  - It is recommended to use the same GSDML/DAP versions on both systems.
  - Is the latest version used?
- Is the GSDML in the correct path?
  - TwinCAT 2: TwinCAT2: C:\TwinCAT\Io\ProfiNet
  - TwinCAT 3: C:\TwinCAT\3.1\Config\Io\Profinet
- Is the correct GSDML used?
  - Version
  - It may be necessary to contact the vendor/manufacturer or search for the appropriate GSDML on the vendor's website.
- Where can I find GSDML files?
  - From Beckhoff products the GSDML files are usually delivered with the installation of TwinCAT.
  - On the Beckhoff website, use the "Download Finder" and its filter options for this purpose

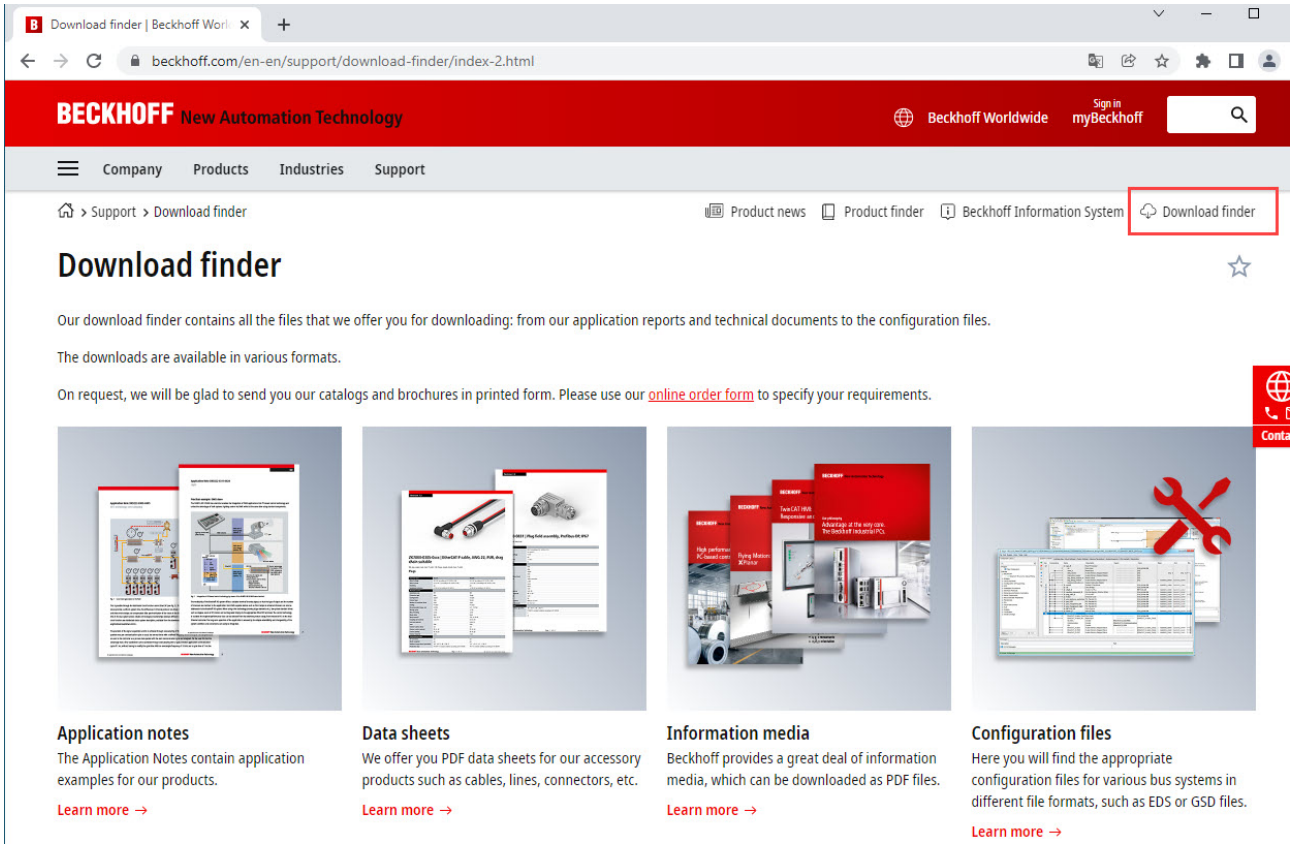


Fig. 24: Website Download finder

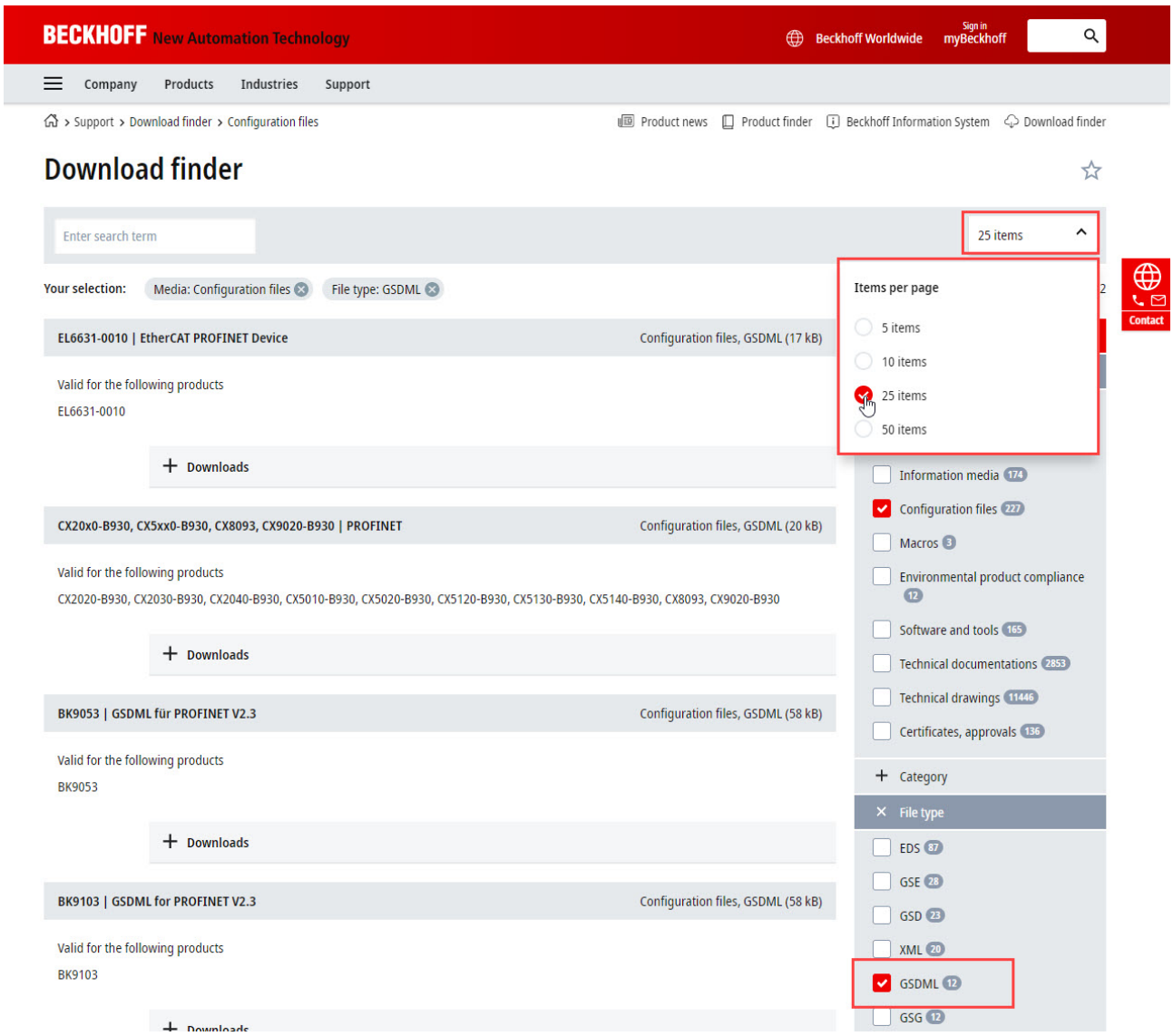


Fig. 25: Website Download finder (filtered)

- For products from other suppliers/manufacturers, the supplier must be contacted or the GSDML files can be downloaded from the website.

8.1.2 Task configuration

Task configuration

- Has a free-running task been created?
  - Or a "special sync task" used?
- Cycle time to base 2?
  - 1ms, 2ms, 4ms, 8ms, ...

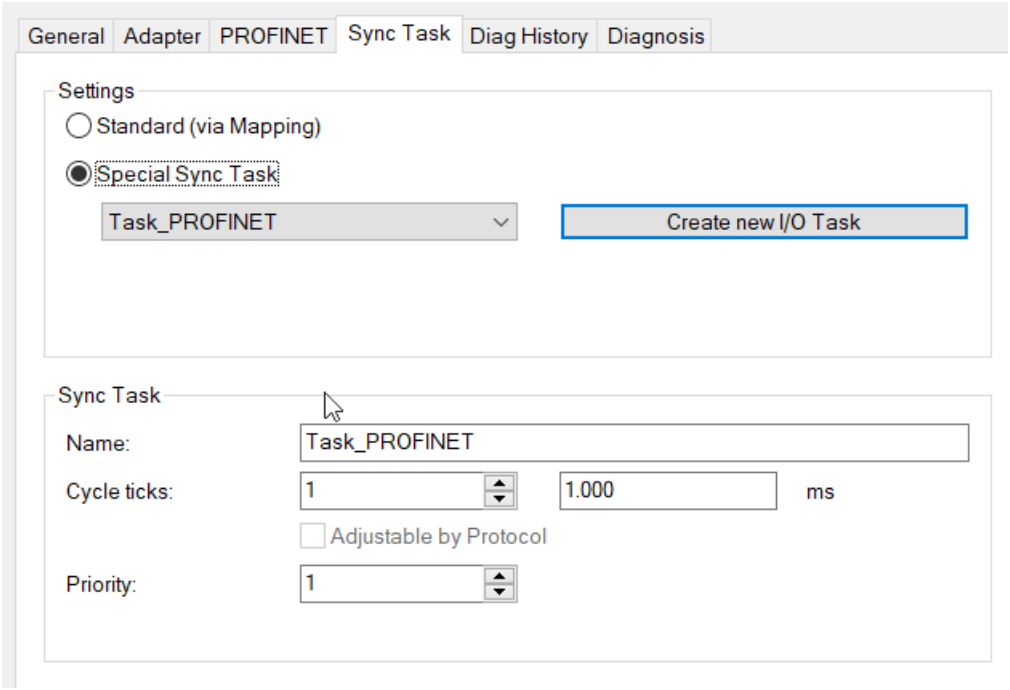


Fig. 26: Setting "Special Sync Task

- Further notes in chapter Sync Task

8.1.3 EL663x-00x0 EtherCAT Terminals

EL663x-00x0 EtherCAT Terminals

- Was the correct terminal used?
  - EL663x-0000 cannot be used as device
  - EL6631-0010 cannot be used as controller

EL663x-00x0 EtherCAT Terminals

- Was the correct terminal used?
  - EL663x-0000 cannot be used as device
  - EL6631-0010 cannot be used as controller

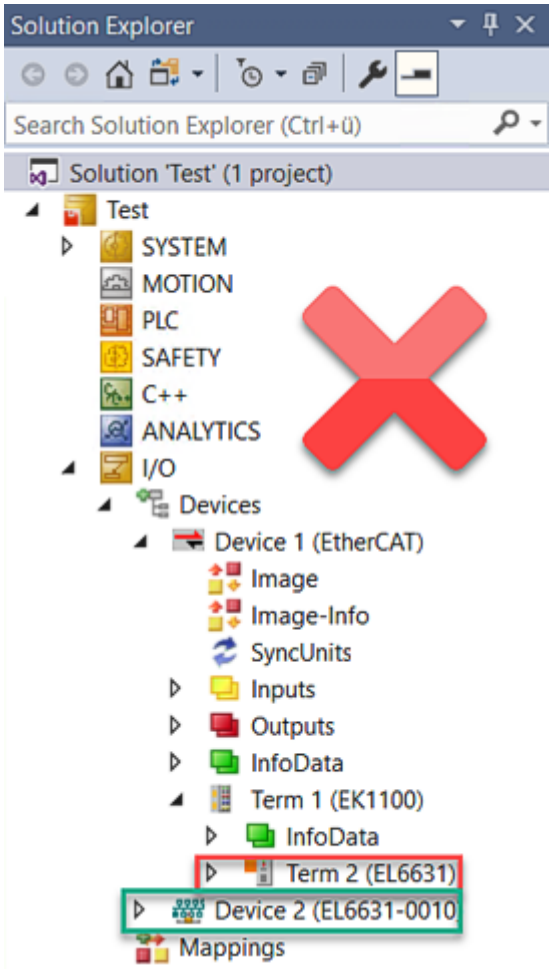


Fig. 27: Wrong configuration

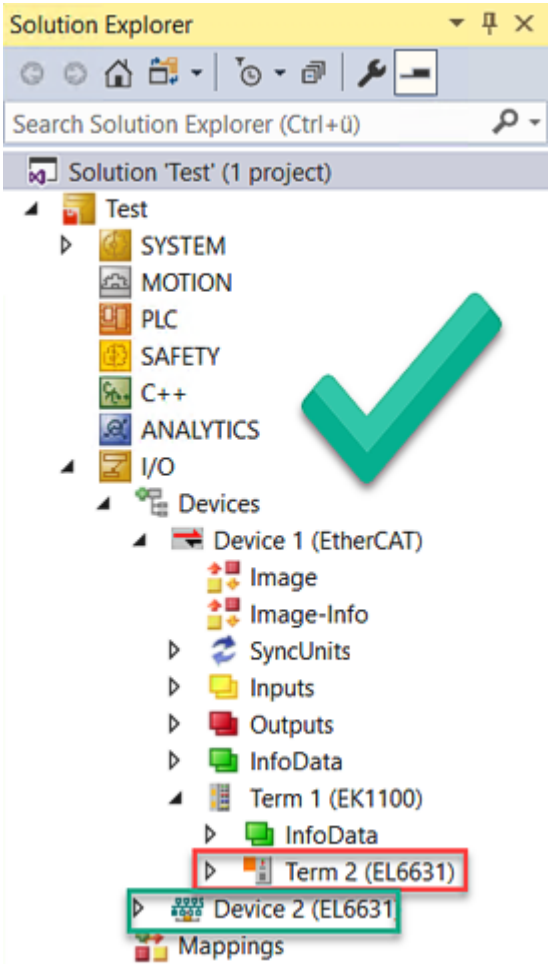


Fig. 28: Correct configuration

- EtherCAT diagnostics
  - EtherCAT state = Operational (OP)
  - WcState = 0 (Data valid)
- EtherCAT diagnostics
  - EtherCAT state = Operational (OP)
  - WcState = 0 (Data valid)

8.1.4 BoxStates of the PROFINET devices

BoxStates of the PROFINET devices

- Communication established?
  - See Box States

8.1.5 Diagnostic status under TIA

- I get a message that the installed firmware is not identical to the version of the configured firmware, what should I do?
  - If this message appears under the TIA software (see illustration), it can be ignored. The products are always downward compatible, i.e. the old GSDML file can still be used with newer software without having to update it. There is no technical reason to take action here.

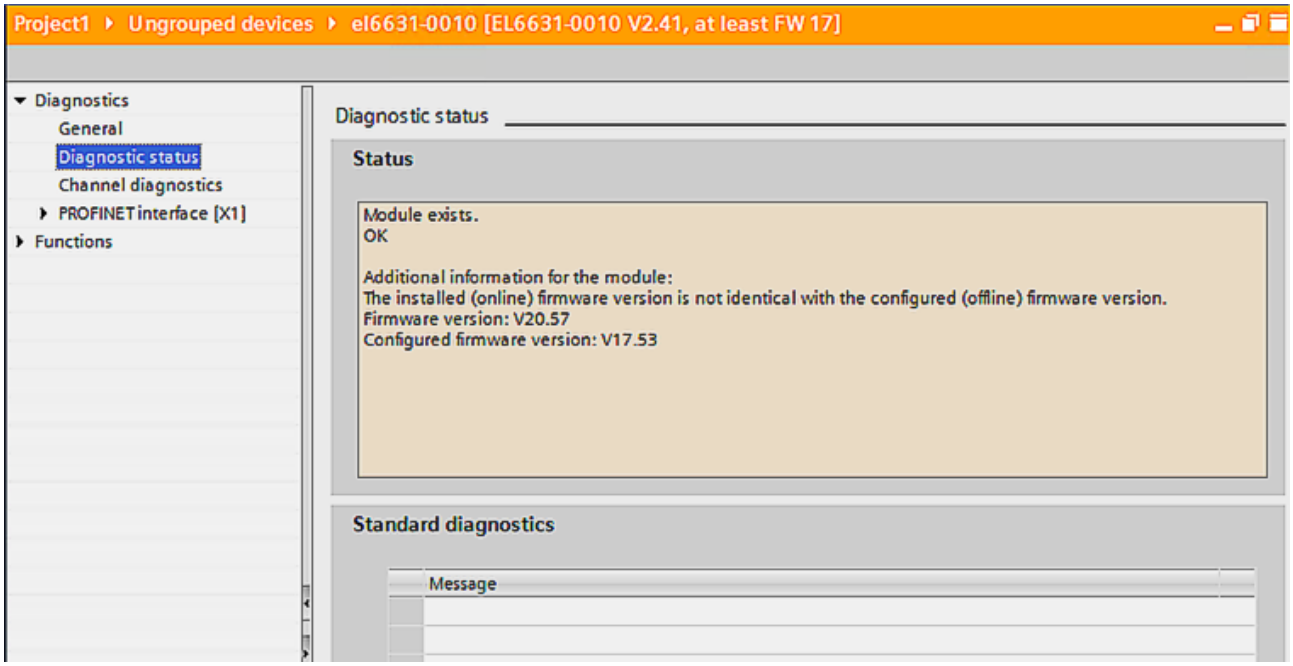


Fig. 29: Note on the firmware in Diagnostic Status

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.

NOTICE

Risk of damage to the device!

Pay attention to the instructions for firmware updates on the separate page [▶ 106].
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!

EL6633			
Hardware (HW)	**Firmware**	**Revision no.**	**Release date**
00 - 01	01	EL6633-0000/0016	2025/02
	02	EL6633-0000/0017	2025/05
	03		2025/08

*) 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/EP/EPP/ERPxxxx

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

NOTICE

Only use TwinCAT 3 software!

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

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

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

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

Storage locations

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

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

NOTICE

Application-specific writing of the ESI-EEPROM

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

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

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

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

Simplified update by bundle firmware

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

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

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

Following the update, its success should be verified

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

NOTICE

Risk of damage to the device!

✓ Note the following when downloading new device files

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

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

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

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

8.4.1 Device description ESI file/XML

NOTICE

Attention regarding update of the ESI description/EEPROM

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

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

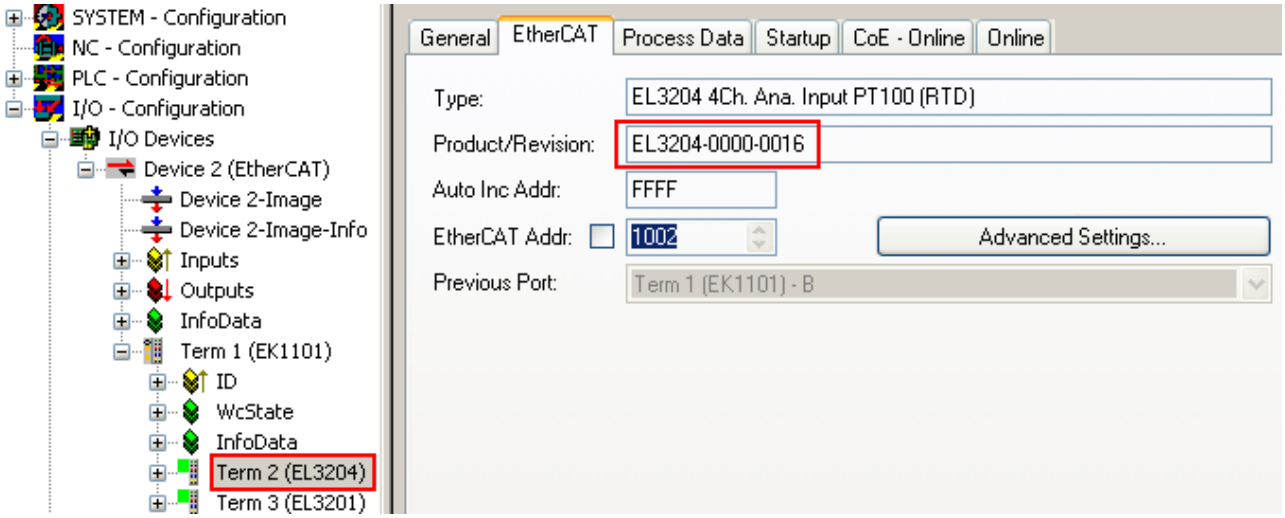


Fig. 30: Device identifier consisting of name EL3204-0000 and revision -0016

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:

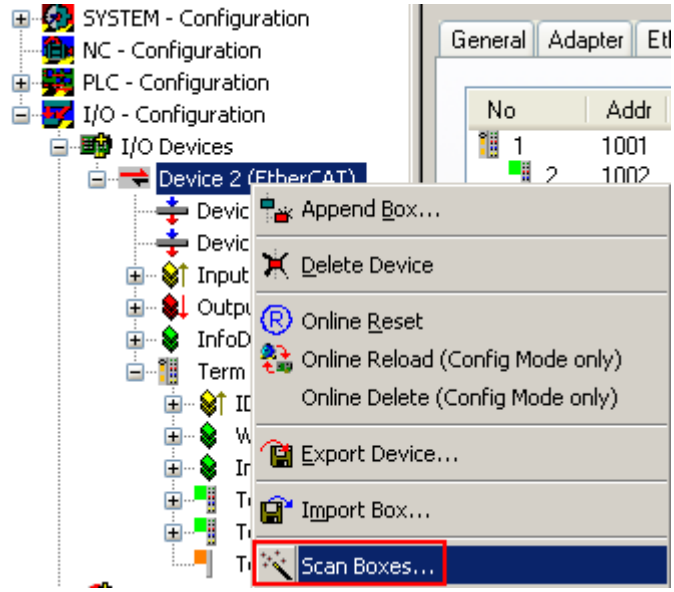


Fig. 31: Scan the subordinate field by right-clicking on the EtherCAT device

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

Fig. 32: Configuration is identical

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

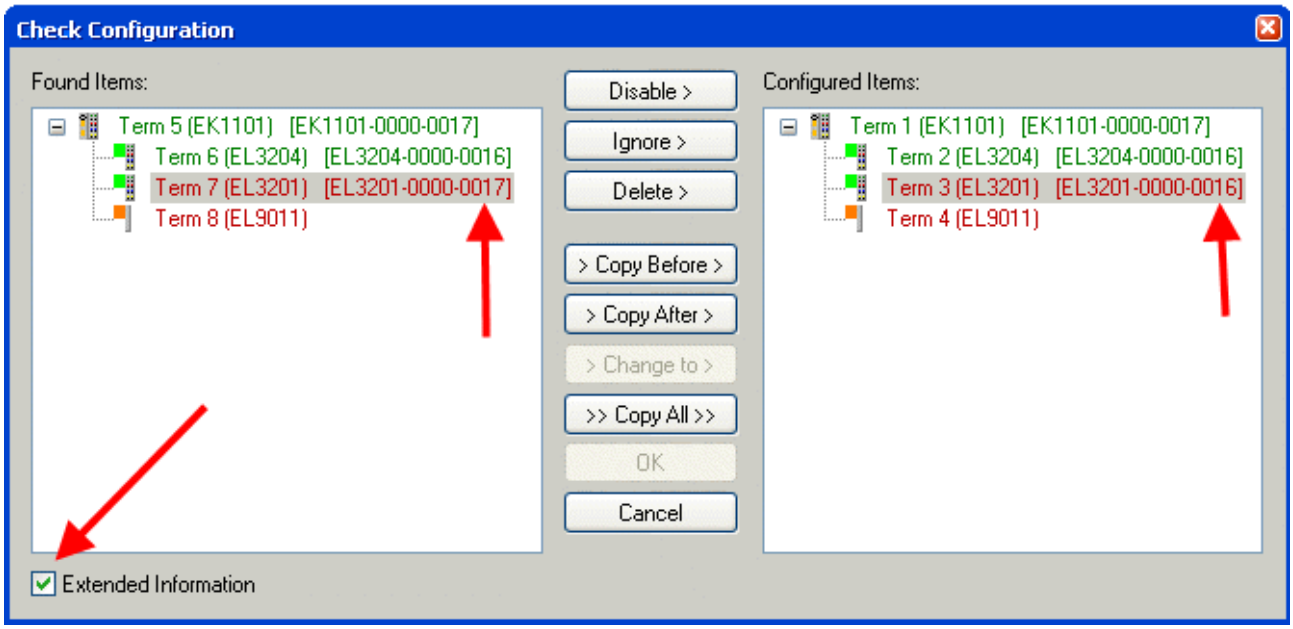


Fig. 33: Change dialog

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*

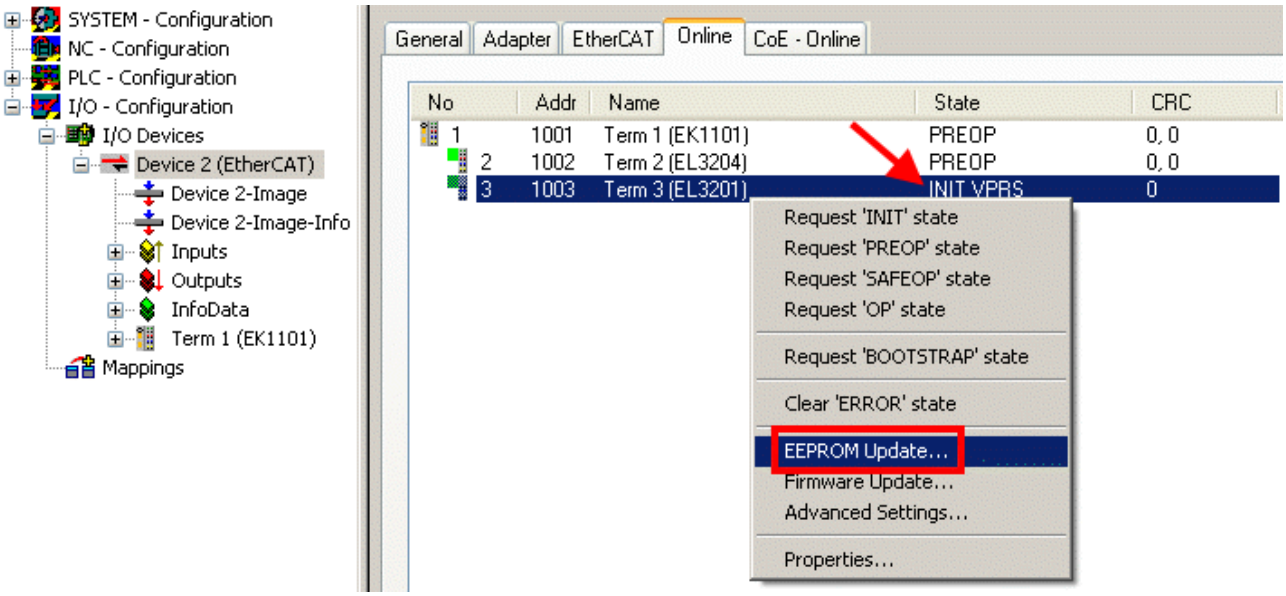


Fig. 34: EEPROM Update

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.

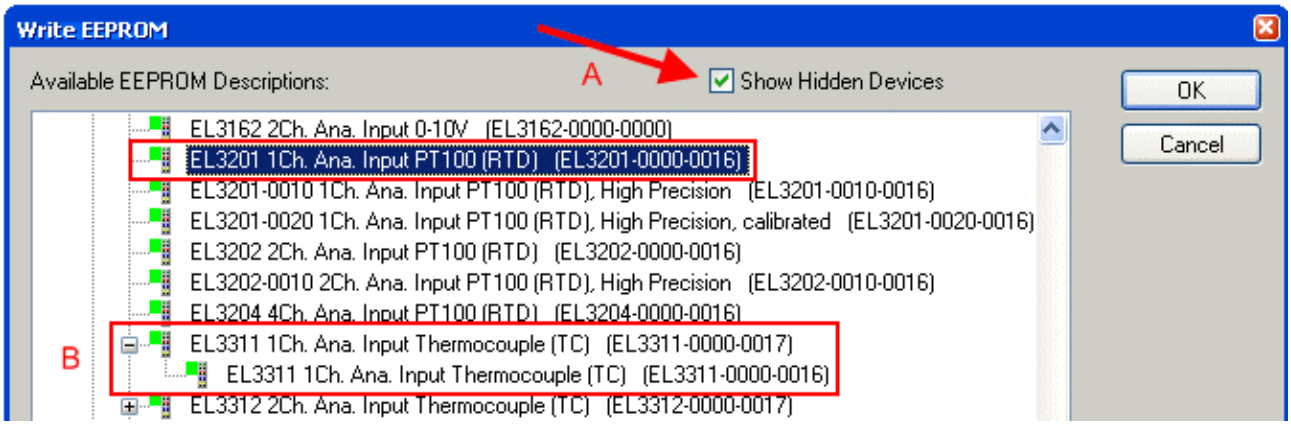


Fig. 35: Selecting the new ESI

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 via the TwinCAT System Manager

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

CoE Online and Offline CoE

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

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

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

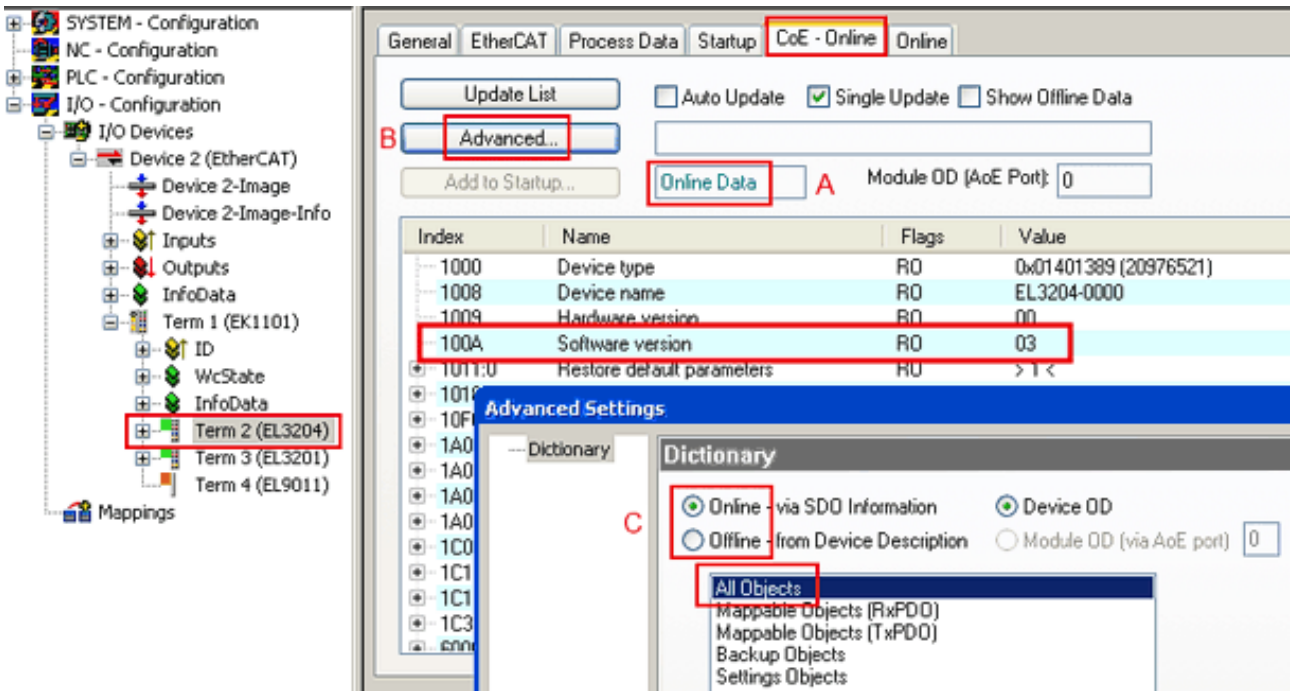


Fig. 36: Display of EL3204 firmware version

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.*

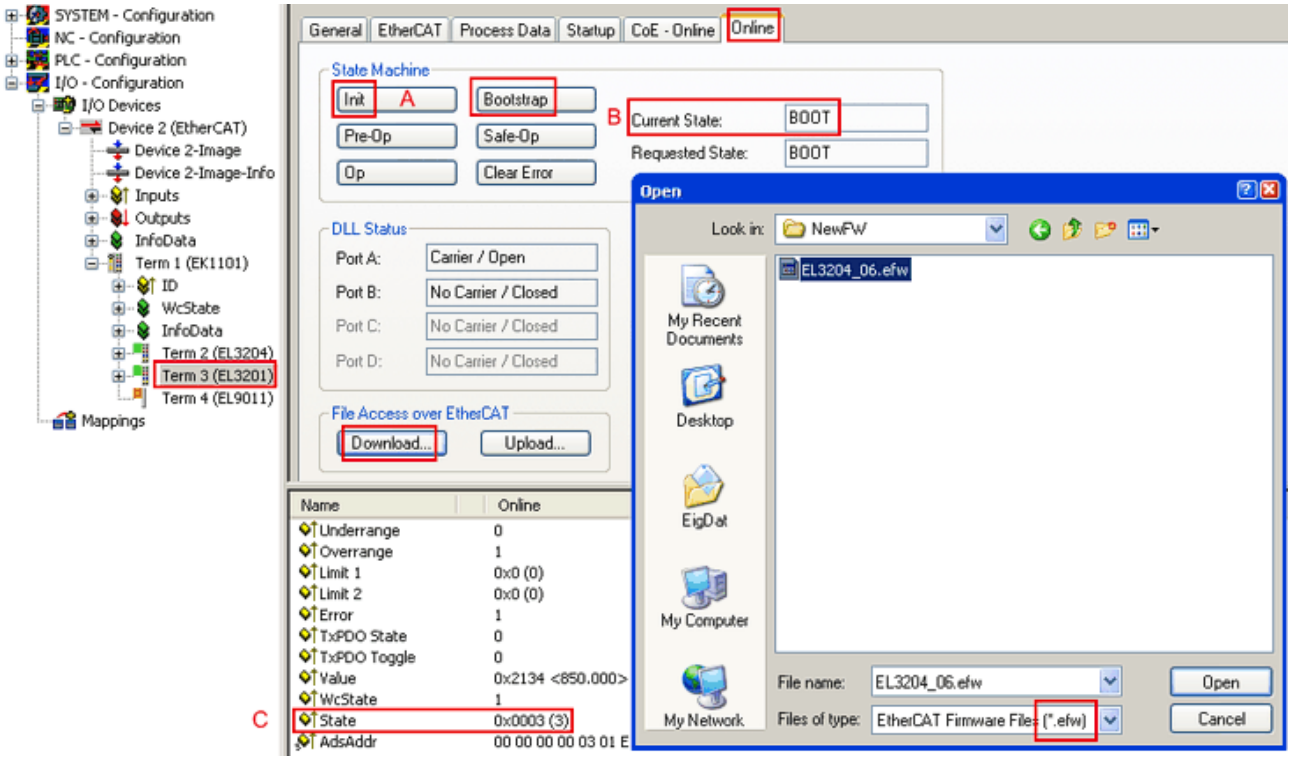
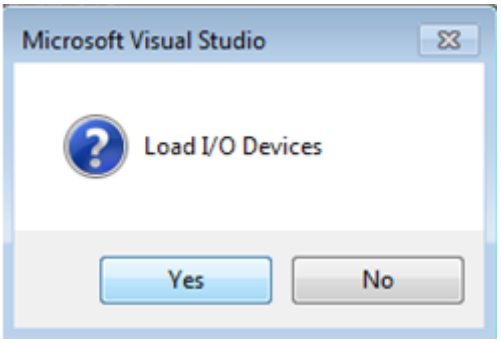
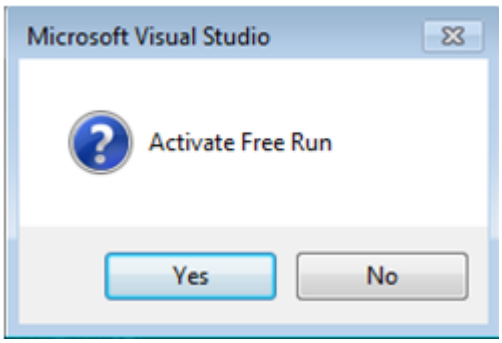


Fig. 37: Firmware Update

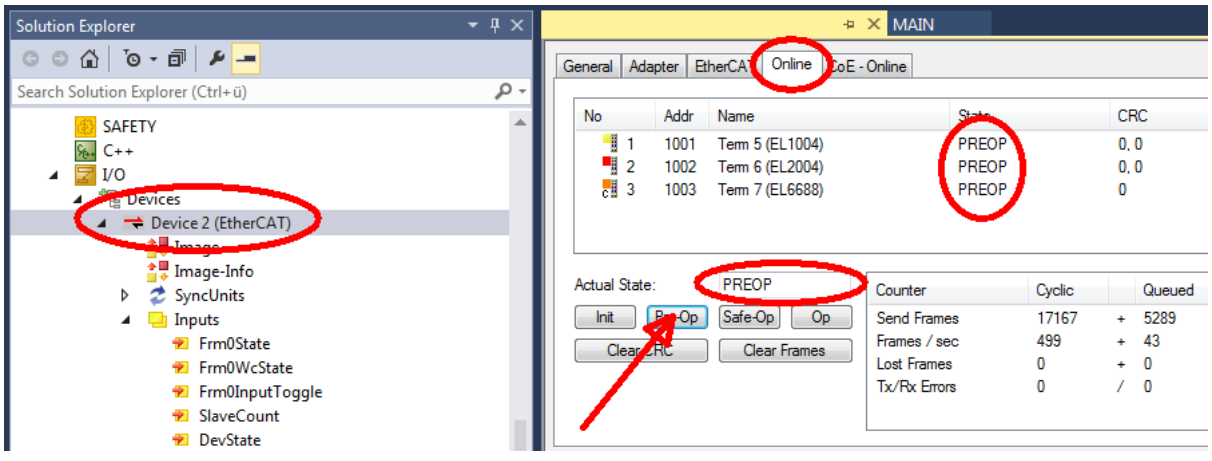
Proceed as follows, unless instructed otherwise by Beckhoff support. Valid for TwinCAT 2 and 3 as EtherCAT master.

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



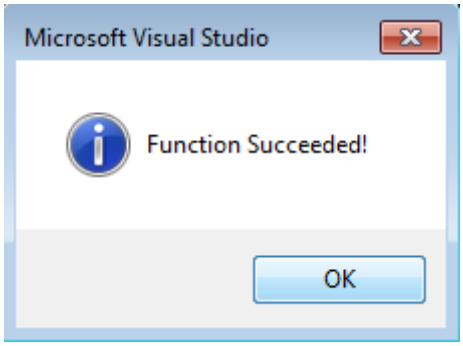


- Switch EtherCAT Master to PreOP



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

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



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

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 TwinCAT System Manager

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

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

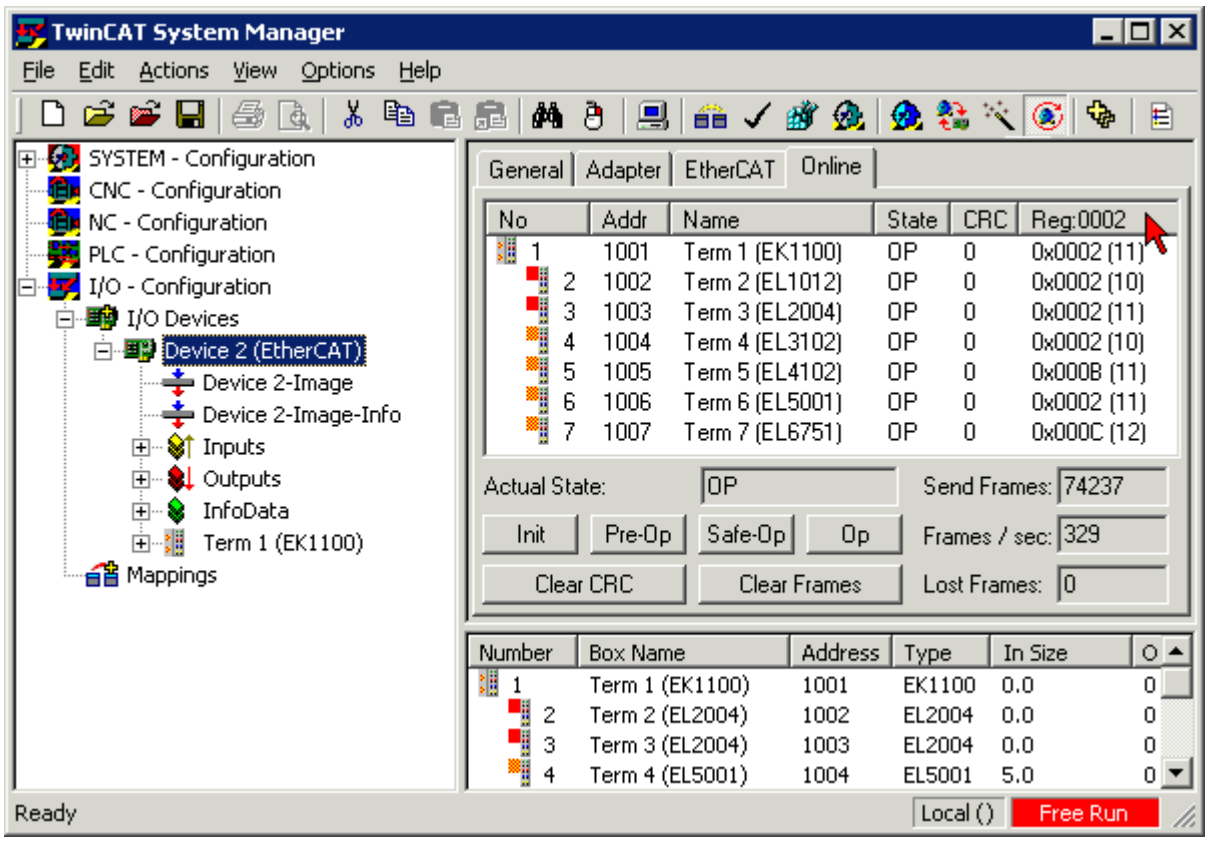


Fig. 38: FPGA firmware version definition

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

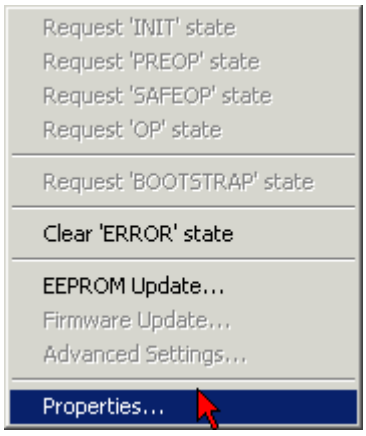


Fig. 39: Context menu *Properties*

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.

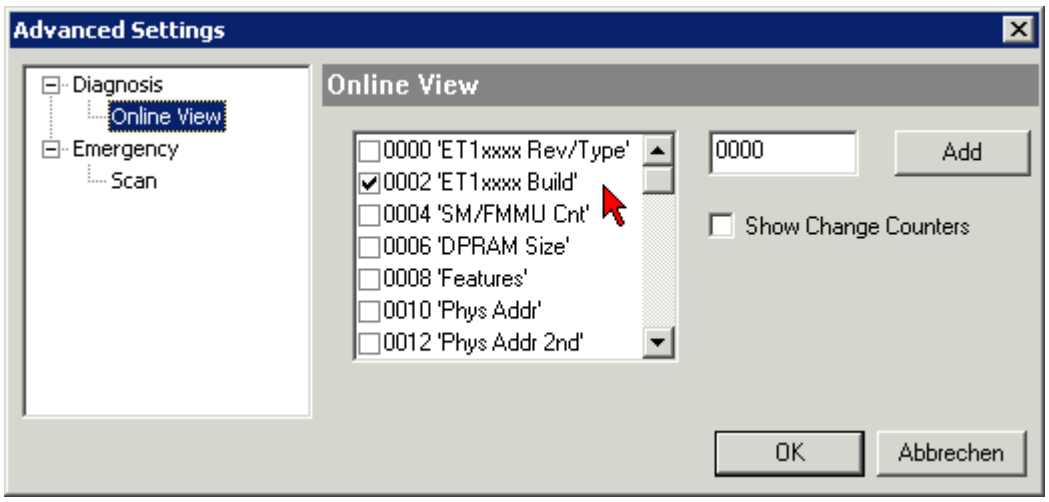


Fig. 40: Dialog *Advanced Settings*

Update

For updating the FPGA firmware

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

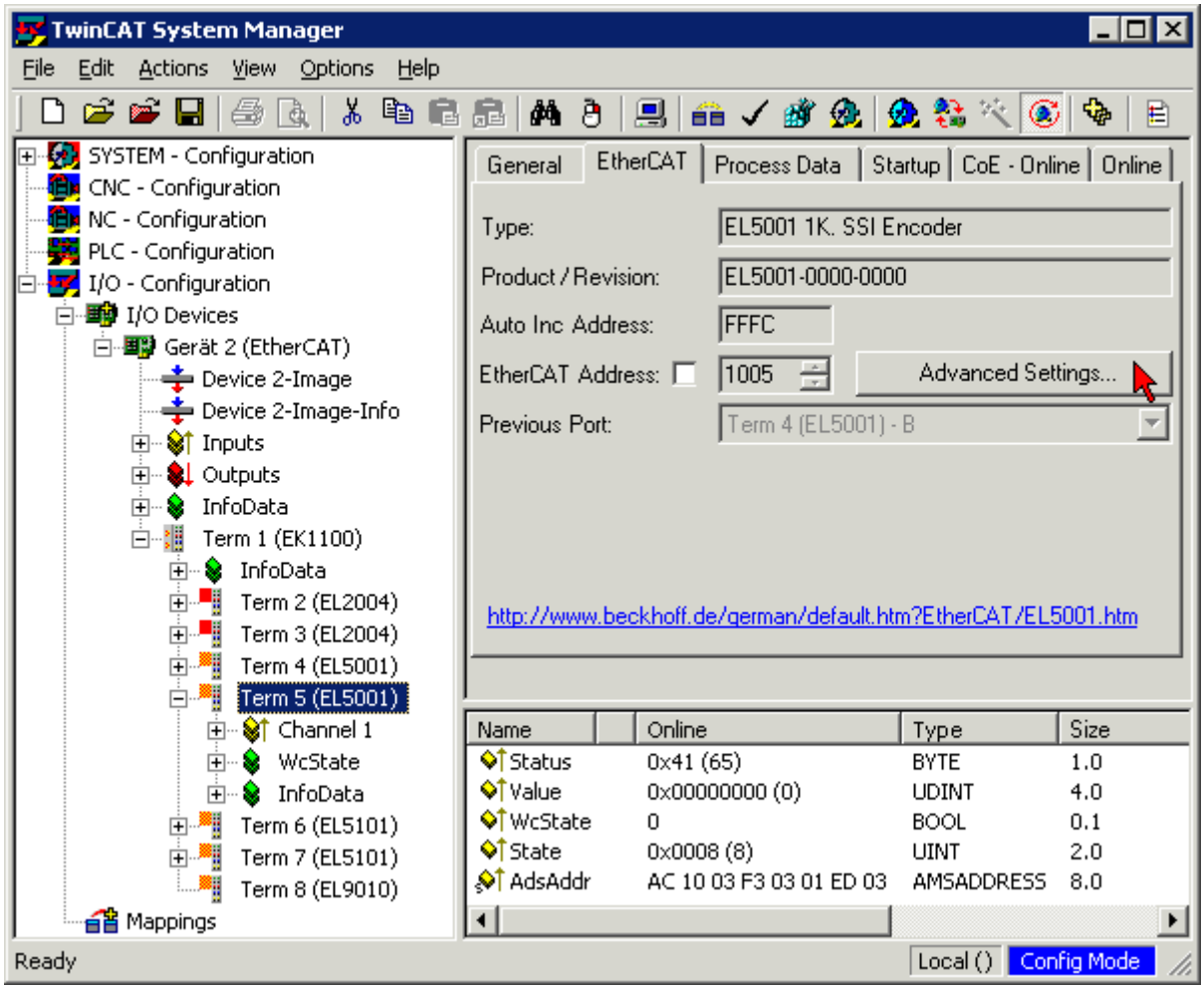
Older firmware versions can only be updated by the manufacturer!

Updating an EtherCAT device

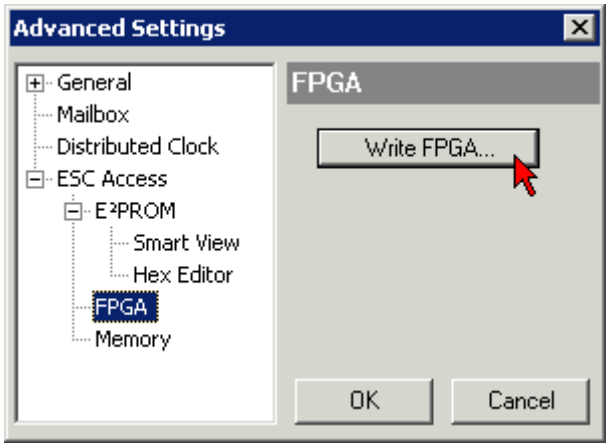
The following sequence order have to be met if no other specifications are given (e.g. by the Beckhoff support):

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

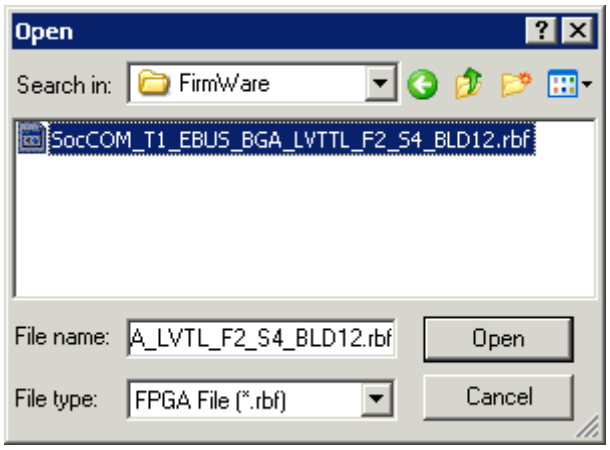
- In the TwinCAT System Manager select the terminal for which the FPGA firmware is to be updated (in the example: Terminal 5: EL5001) and
  click the *Advanced Settings* button in the *EtherCAT* tab:



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



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



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

NOTICE

Risk of damage to the device!

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

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.

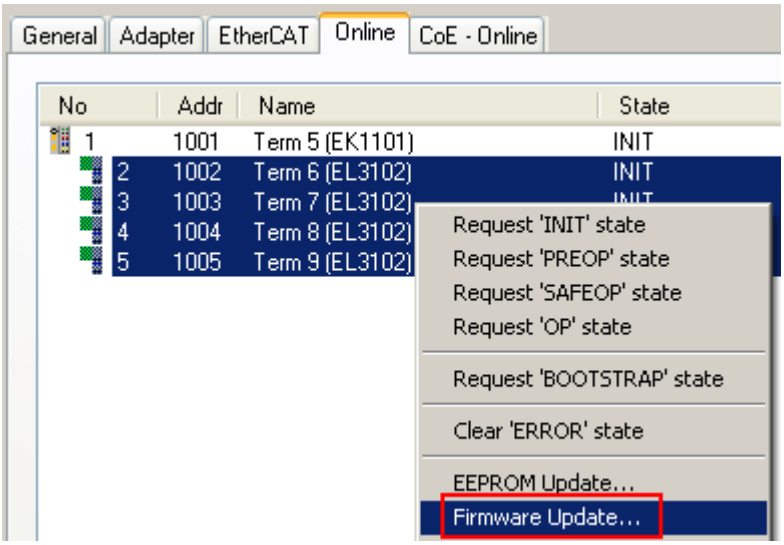


Fig. 41: Multiple selection and firmware update

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

8.5 Support and Service

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

Beckhoff's branch offices and representatives

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

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

You will also find further documentation for Beckhoff components there.

Support

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

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

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

Service

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

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

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

Headquarters Germany

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Germany

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

Trademark statements

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

Third-party trademark statements

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

Intel, the Intel logo, Intel Core, Xeon, Intel Atom, Celeron and Pentium are trademarks of Intel Corporation or its subsidiaries.

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

More Information:
www.beckhoff.com/EL6xxx

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