CANopen-PN

PROFINET IO / CANopen Gateway

Manual

to Product C.2921.02

esd electronics gmbh
Vahrenwalder Str. 207 • 30165 Hannover • Germany
http://www.esd.eu
Phone: +49 (0) 511 3 72 98-0 • Fax: +49 (0) 511 3 72 98-68

N O T E

The information in this document has been carefully checked and is believed to be entirely reliable. **esd electronics** makes no warranty of any kind with regard to the material in this document, and assumes no responsibility for any errors that may appear in this document. In particular descriptions and technical data specified in this document may not be constituted to be guaranteed product features in any legal sense.

esd electronics reserves the right to make changes without notice to this, or any of its products, to improve reliability, performance or design.

esd electronics gmbh
Vahrenwalder Str. 207
30165 Hannover
Germany

Phone:	+49-511-372 98-0
Fax:	+49-511-372 98-68
E-Mail:	info@esd.eu
Internet:	www.esd.eu

This manual contains important information and instructions on safe and efficient handling of the CANopen-PN. Carefully read this manual before commencing any work and follow the instructions.
The manual is a product component, please retain it for future use.

Trademark Notices

CANopen® and CiA® are registered community trademarks of CAN in Automation e.V.
PROFINET® is a registered trademark of PROFIBUS and PROFINET International (PI).
ARM® is a registered trademark of Advanced RISC Machines, Ltd or its subsidiaries.

All other trademarks, product names, company names or company logos used in this manual are reserved by their respective owners.

Document file:	I:\Texte\Doku\MANUALS\CAN\CANopen-PN\English\CANopen-PN_Manual_en_17.odt
Date of print:	2023-07-28
Document type number:	DOC0800

Firmware version:	CANopen-PN firmware from rev.1.0.6

Document History

The changes in the document listed below affect changes in the hardware as well as changes in the description of the facts, only.

Revision	Chapter	Changes versus previous version	Date
1.2	6	Notes concerning PROFINET IO services revised or inserted	2015-06-04
	6.1.9	Note concerning bit rate updated	
	6.2.1	Description of the Error Codes revised	
	9., 10.	Chapter updated	
1.3	-	Safety Information revised	2015-11-27
	4.	Note to compatibility of CANopen-PN firmware and specific GSDML Composer versions inserted	
	5.1.3	New chapter „Compatibility CANopen-PN Firmware and GSDML Composer“	
	5.2	Note to compatibility of CANopen-PN firmware and specific GSDML Composer versions inserted	
	5.7	Note to *CommunicationCyclePeriod* inserted	
	5.11	New chapter “Using GSDML File and CANopen-PN Gateway with the TIA Portal“	
1.4	-	Classification of warning messages moved and revised	2017-05-08
	4.	Warning message concerning power supply inserted	
	5.1	Note on CANopen-Slave inserted	
	5.6	“CANopen-PN” inserted in first sentence	
1.5	1.1	Note on CANopen Slave functions deleted	2017-07-20
	15.	New PROFINET Certificate	
1.6	14.	New Declaration of Conformity	2020-07-24
	15.	New PROFINET Certificate	
1.7	15.	PROFINET Certificate updated	2023-07-28

Technical details are subject to change without further notice.

Classification of Warning Messages and Safety Instructions

This manual contains noticeable descriptions, warning messages and safety instructions, which you must follow to avoid personal injuries or death and property damage.

This is the safety alert symbol.
It is used to alert you to potential personal injury hazards. Obey all safety messages and instructions that follow this symbol to avoid possible injury or death.

DANGER, WARNING, CAUTION

Depending on the hazard level the signal words DANGER, WARNING or CAUTION are used to highlight safety instructions and warning messages. These messages may also include a warning relating to property damage.

DANGER
Danger statements indicate a hazardous situation which, if not avoided, will result in death or serious injury.

WARNING
Warning statements indicate a hazardous situation that, if not avoided, could result in death or serious injury.

CAUTION
Caution statements indicate a hazardous situation that, if not avoided, could result in minor or moderate injury.

NOTICE

Notice statements are used to notify people on hazards that could result in things other than personal injury, like property damage.

NOTICE
This NOTICE statement contains the general mandatory sign and gives information that must be heeded and complied with for a safe use.

INFORMATION

INFORMATION
Notes to point out something important or useful.

Safety Instructions

- When working with the CANopen-PN follow the instructions below and read the manual carefully to protect yourself from injury and the CANopen-PN from damage.
- Do not open the housing of the CANopen-PN.
- The permitted operating position is specified as shown (Figure: 47). Other operating positions are not allowed.
- Never let liquids get inside the CANopen-PN. Otherwise, electric shocks or short circuits may result.
- Protect the CANopen-PN from dust, moisture and steam.
- Protect the CANopen-PN from shocks and vibrations.
- The CANopen-PN may become warm during normal use. Always allow adequate ventilation around the CANopen-PN and use care when handling.
- Do not operate the CANopen-PN adjacent to heat sources and do not expose it to unnecessary thermal radiation. Ensure an ambient temperature as specified in the technical data.
- Do not use damaged or defective cables to connect the CANopen-PN and follow the CAN wiring hints in chapter: "Correctly Wiring Electrically Isolated CAN Networks".
- In case of damages to the device, which might affect safety, appropriate and immediate measures must be taken, that exclude an endangerment of persons and domestic animals and property.
- Current circuits which are connected to the device have to be sufficiently protected against hazardous voltage (SELV according to EN 60950-1).
- The CANopen-PN may only be driven by power supply current circuits, that are contact protected. A power supply, that provides a safety extra-low voltage (SELV or PELV) according to EN 60950-1, complies with this conditions.

Qualified Personnel

This documentation is directed exclusively towards personnel qualified in control and automation engineering.
The installation and commissioning of the product may only be carried out by qualified personnel, which is authorized to put devices, systems and electric circuits into operation according to the applicable national standards of safety engineering.

Conformity

The CANopen-PN meets the demands of the EU regulations and EMC standards printed in the conformity declaration at the end of this manual.

Data Safety

This device is equipped with an Ethernet or other interface which is suitable to establish a connection to data networks. Depending on the software used on the device, these interfaces may allow attackers to compromise normal function, get illegal access or cause damage.
esd does not take responsibility for any damage caused by the device if operated at any networks. It is the responsibility of the device's user to take care that necessary safety precautions for the device's network interface are in place.

Intended Use

The intended use of the CANopen-PN is the operation as PROFINET IO / CANopen Gateway.
The guarantee given by esd does not cover damages which result from improper use, usage not in accordance with regulations or disregard of safety instructions and warnings.

- The CANopen-PN is intended for indoor operation only.
- The operation of the CANopen-PN in hazardous areas, or areas exposed to potentially explosive materials is not permitted.
- The operation of the CANopen-PN for medical purposes is prohibited.

Service Note

The CANopen-PN does not contain any parts that require maintenance by the user. The CANopen-PN does not require any manual configuration of the hardware. Unauthorized intervention in the device voids warranty claims.

Disposal

Devices which have become defective in the long run have to be disposed in an appropriate way or have to be returned to the manufacturer for proper disposal. They must not be disposed of with household waste.
Please, make a contribution to environmental protection.

Typographical Conventions

Throughout this manual the following typographical conventions are used to distinguish technical terms.

Convention	Example
File and path names	**`/dev/null`** or **`<stdio.h>`**
Function names	***open()***
Programming constants	`NULL`
Programming data types	`uint32_t`
Variable names	*Count*

Number Representation

All numbers in this document are **base 10** unless designated otherwise. Hexadecimal numbers have a prefix of **0x**, and binary numbers have a prefix of **0b**. For example, 42 is represented as 0x2A in hexadecimal and 0b101010 in binary.

Abbreviations

Abbreviation	Term	Description
DCF	Device Configuration File	
EDS	Electronic Data Sheet	Description file for CANopen devices
GSD	General Station Description	Description file for PROFINET devices
GSDML	General Station Description Markup Language	Language of the GSD files (XML-based)
PDO	Process Data Object	
RTR	Remote Transmission Request	
SDO	Service Data Object	
XML	Extensible Markup Language	

Table of contents

Index of Tables

List of Figures

1. Overview

1.1 Description of CANopen-PN

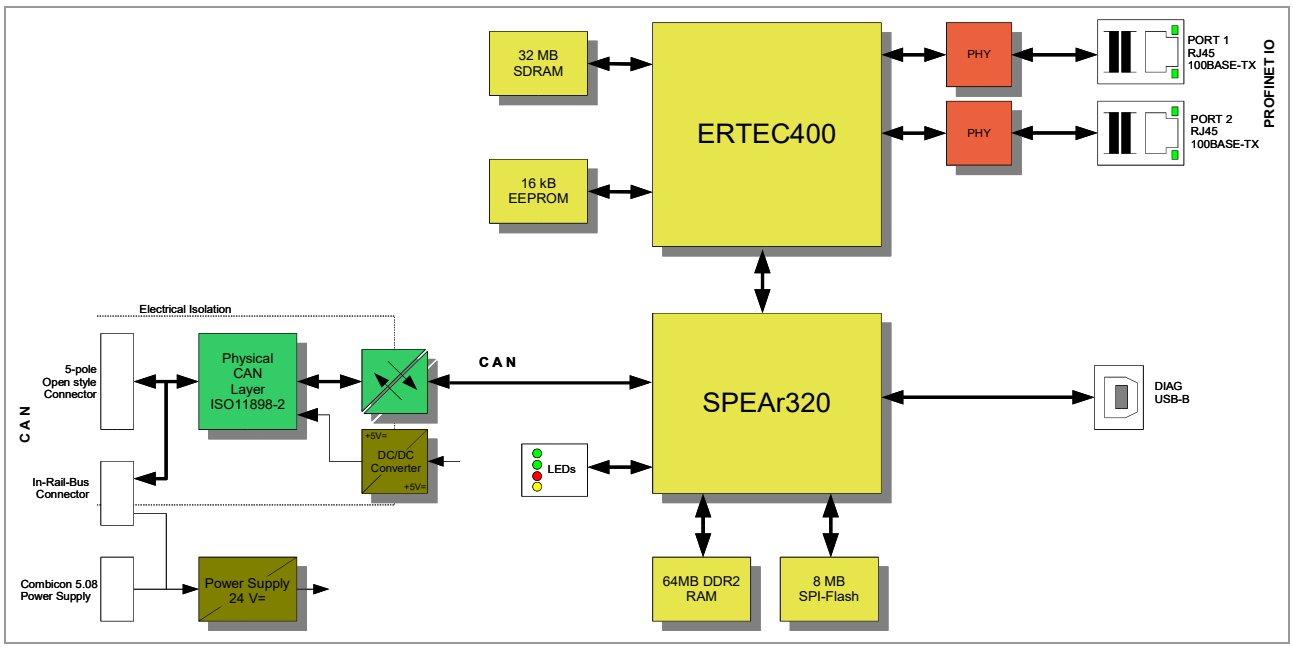


Figure 1: Block circuit diagram of CANopen-PN

The CANopen-PN offers a fast and reliable link of PROFINET IO® controllers to a CANopen network. The gateway operates as a PROFINET IO device with a maximum of 1440 bytes input data and 1440 bytes output data on the PROFINET IO bus.
The CANopen-PN is designed according to Profibus International Document TC2-09-0002 (CANopen-Integration_7012_d07_Jul09) [6].
The firmware controls the connection of CAN bus units via the CANopen® protocol [2] to PROFINET IO, thus the CANopen-PN can be e.g. connected to a SIMATIC-S7. The number of CAN participants is not limited by the CANopen-PN. The device is CANopen master.
The CANopen implementation supports CANopen master functions according to CiA® 302-2 [3].

The CANopen-PN module works with an ERTEC400 ARM® processor for the PROFINET IO communication, which buffers the PROFINET IO data in a local SDRAM.
Furthermore the CANopen-PN uses a SPEAr320 ARM®-processor, with integrated CAN controller, that controls the CANopen side and buffers the data in a local DDR2-RAM.
The firmware and the configuration files are stored in the flash.

The high-speed CAN interface (compatible to ISO 11898-2) allows a maximum data-transfer rate of 1 Mbit/s.
The 100 BASE-TX PROFINET IO interface is compatible to IEEE802.3 and runs at 100 Mbit/s.

The PROFINET IO and the CANopen interface are electrically isolated against the other components. The PROFINET IO interface is equipped with two RJ45 sockets in accordance with the standard. An Ethernet switch is integrated in the ERTEC400.
The CAN interface is connected via a 5-pin COMBICON connector with spring-cage connection.

The module can be configured via the GSDML Composer, which is a PROFINET IO configuration tool. Additionally, the esd CAN tools (CANreal, CANplot, CANrepro, CANscript, COBview) can be used for CAN diagnostics.

2. PCB View with Connectors

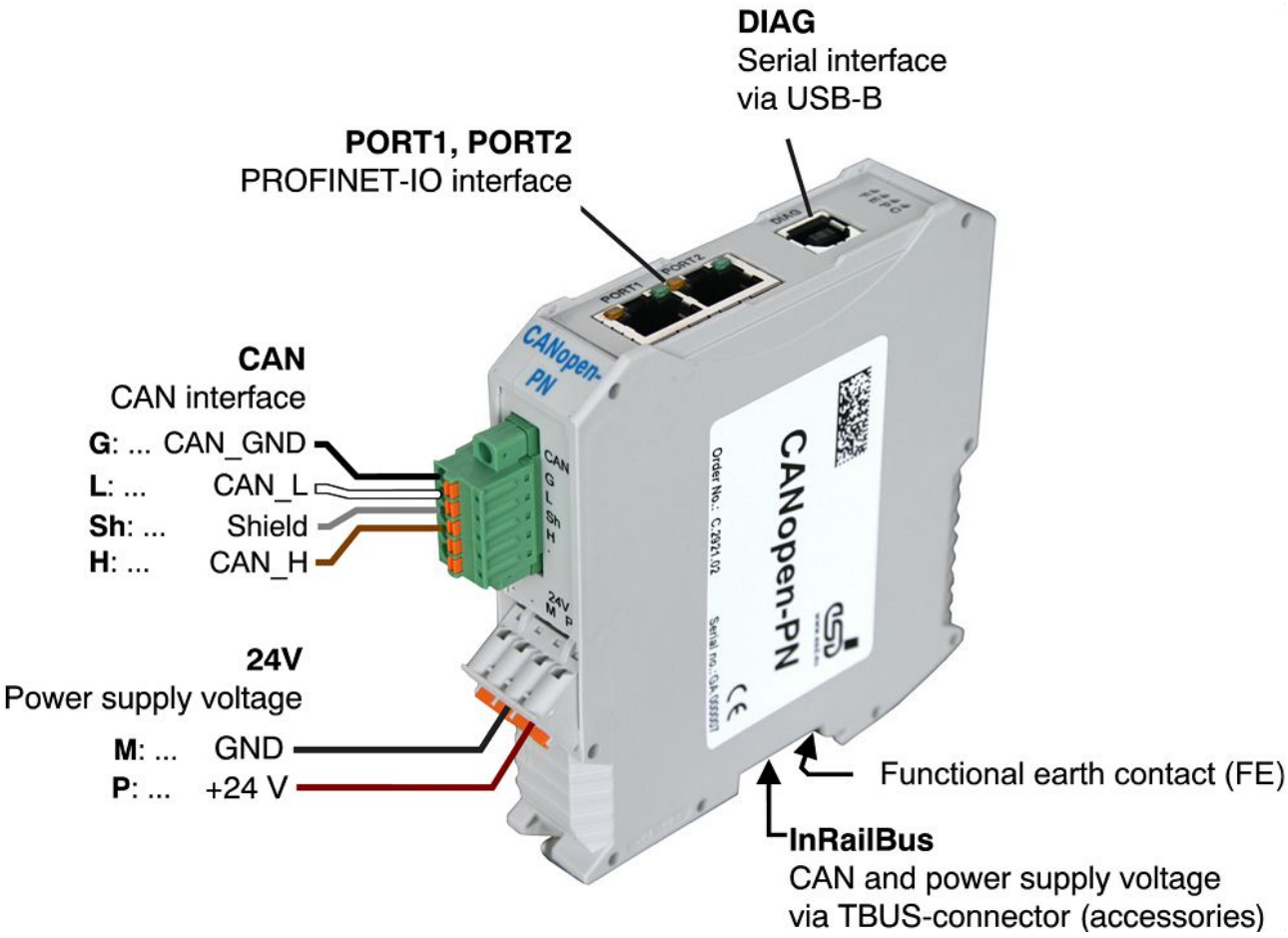


Figure 2: Connecting diagram of CANopen-PN

See also page 73 for signal assignment of the CAN connectors.

3. LEDs

3.1 Position of the LEDs

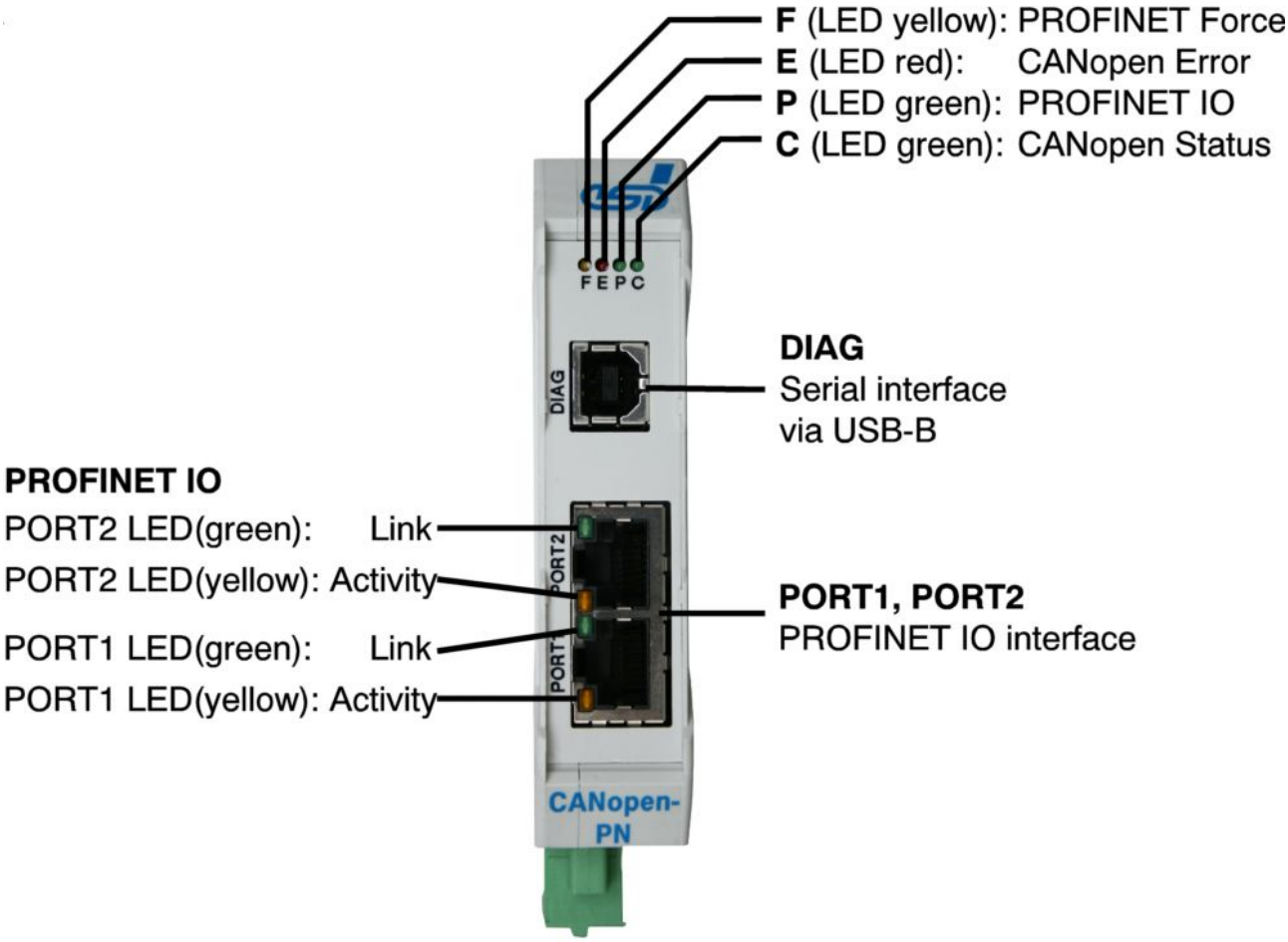


Figure 3: Front panel view with LEDs

3.2 Indication of the LEDs

PROFINET IO RJ-45 socket LEDs of PORT1 and PORT2

LED	Colour	Indicator state	Meaning
Link	green	off	no Ethernet connection
		on	Ethernet connection is established
Activity	yellow	off	no Ethernet connection
		blinking	Ethernet connection is established, data is transferred
		on	Ethernet connection is established

Table 1: Indication of the RJ45-LEDs

LEDs F, E, P and C

LED	Colour	Function	Indicator state	State / Meaning	LED-Name in PCB
F	yellow	PROFINET IO Force	off	there is no requirement of the PROFINET IO-controller for the identification of the unit	LED1A
			blinking	Requirement of the PROFINET IO controller for identification of the unit	
E	red	CANopen Error	off	no error - the CANopen-PN is in working condition	LED1B
			single flash	*Warning Limit Reached* - At least one of the error counters of the CAN controller has reached or exceeded the warning level (too many error frames)	
			double flash	*Error Control Event* - a Heartbeat- or Nodeguard error occurred	
			on	the CAN controller is *Bus off*	
P	green	PROFINET IO	off	no valid PROFINET IO link	LED1C
			on	valid PROFINET IO link is established	
C	green	CANopen Status	blinking	*PREOPERATIONAL*	LED1D
			single flash	*STOPPED*	
			on	*OPERATIONAL*	

Table 2: Indication of the LEDs F, E, P and C

The indicator states of the CANopen LEDs are described in chapter 3.2.1.

3.2.1 CANopen Indicator States

Description of the CANopen indicator states (according to CiA 303 [7]):

Indicator State	Indication
on	LED on
off	LED off
blinking	LED blinking with 2.5 Hz
single flash	LED 200 ms on, 1000 ms off
double flash	LED 200 ms on, 200 ms off, 200 ms on, 1000 ms off

4. Hardware Installation

To put the CANopen-PN into operation, please follow the installation notes.

Step	Procedure	see page
	**Read the safety instructions at the beginning of this document carefully, before you start with the hardware installation!**	5
	**Danger** Hazardous Voltage - Risk of electric shock due to unintentional contact with uninsulated live parts with high voltages inside of the system into which the CANopen-PN is to be integrated. → All current circuits which are connected to the device have to be sufficiently protected against hazardous voltage (SELV according to EN 60950-1) before you start with the installation. → Ensure the absence of voltage before starting any electrical work.	
1.	Mount the CANopen-PN module and connect the interfaces (power supply voltage, CAN, PROFINET IO).	13
2.	Please note that the CAN bus has to be terminated at both ends! **esd** offers special T-connectors and termination connectors. Additionally the CAN_GND signal has to be connected to earth at **exactly one** point in the CAN network. A CAN participant with a CAN interface which is not electrically isolated corresponds to the grounding of the CAN-GND.	78
3.	If you use the InRailBus, read chapter 12. “Appendix InRailBus (Option)”	97
	**INFORMATION** If you want to use the CAN interface and the esd CAN tools under Windows® (see chapter 5.1.4.2 “CAN Interface and Tools under Windows"), install the software now as described in chapter "5.1.4 Software Installation".	19
4.	Switch on the 24 V-power supply voltage of the CANopen-PN.	
5.	End of hardware configuration	
	**NOTICE** To guarantee a proper operation of the CANopen-PN make absolutely sure to use version 1.0.6 of the GSDML Composer, that is compatible with the CANopen-PN firmware version 1.0.6, before you start with the software configuration!	-
...	Start with the software configuration now (For a software quick start see chapter 5.2 “GSDML Composer Quick Start“)	22

5. Software Configuration with the GSDML Composer

5.1 Description of the GSDML Composer

The GSDML Composer is designed to generate and parametrize a GSD file for the CANopen-PN.

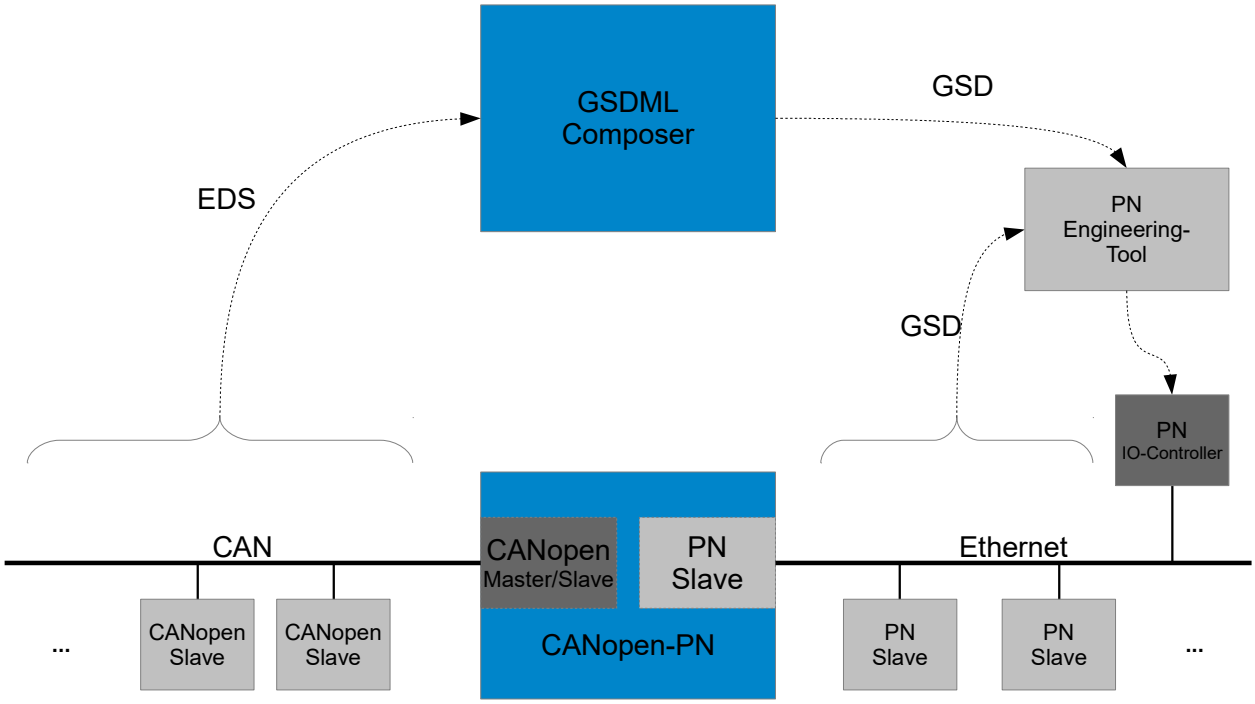


Figure 4: Block circuit diagram GSDML Composer

The configuration of the CANopen side of the gateway is performed with the GSDML Composer, the PROFINET IO side is usually configured via a so called engineering tool (e.g. SIMATIC-S7).

The CANopen-PN is CANopen master and cannot be used as CANopen slave. The internal CANopen slave is implicitly used by the PROFINET slave.

The EDS files with information about the CANopen slaves which are required on CANopen side, must be imported in the GSDML Composer.

The GSD file for the description of CANopen-PN as PROFINET IO slave, which is used on PROFINET IO side, is generated with the GSDML Composer.
Thus the GSD file contains all information, the PROFINET IO control needs for the operation of the Gateway, and the information, the CANopen Manager integrated in the gateway needs for his CANopen net.

5.1.1 Features of the GSDML Composer

- Windows application with intuitive operation
- CANopen device library based on EDS files
- CANopen Network Editor for graphic overview and configuration of the CANopen networks
- Device editor for easy configuration of the CANopen objects
  - PDO Mapping, Error Control Services (Node guarding, Heartbeat), etc.
- Export of the DCF files of the parametrized CANopen Slaves
- Export of the GSD file containing the complete configuration, for the usage of the esd CANopen-PN gateway with the corresponding CANopen net in PROFINET IO

5.1.2 System Requirements

Operating system	Microsoft Windows XP or newer with Microsoft .NET Framework Version 3.5
Hard disk memory space	approx. 12 MB (CAN Tools approx. 42 MB)
Random access memory	according to recommendations for the operating system used

5.1.3 Compatibility of CANopen-PN Firmware and GSDML Composer

NOTICE
To guarantee a proper operation of the CANopen-PN make absolutely sure to use version 1.0.6 of the GSDML Composer, that is compatible with the CANopen-PN firmware version 1.0.6, before you start with the software configuration!

5.1.4 Software Installation

The software installation is started by calling the following installation program on the data CD (which is contained in the scope of delivery):
`CANopen-PN_X_X_X_Setup.exe` (`X` according to the version of the GSDML Composer)

INFORMATION
Administrator rights are required for the installation.

5.1.4.1 GSDML Composer

The GSDML Composer will be automatically installed with the setup specified above. After software installation the program can be started, e.g. via the Windows start menu.

5.1.4.2 CAN Interface and Tools under Windows

Optionally a Windows driver together with its tools can be installed, to use the CAN interface of the CANopen-PN under Windows.

Therefore select *Incl. tools for CAN diagnostics* in the *Select Components* window during installation, see Figure 5:

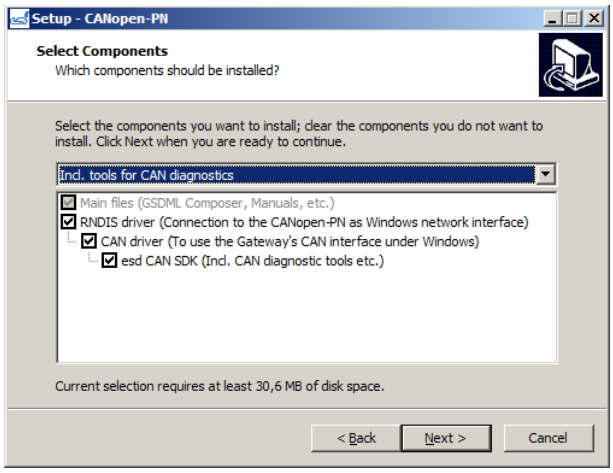


Figure 5: Component selection during setup

RNDIS driver:	Provides a network interface under Windows, which represents the interface to CANopen-PN.
CAN driver:	Provides the esd CAN-API (“NTCAN”) via a TCP/IP connection.
esd CAN-SDK:	Amongst others it contains software for the CAN diagnostics, especially CANreal, which will be used for testing (5.1.4.2.1).

INFORMATION
The CAN driver and the CAN-SDK will not be automatically deleted if the CANopen-PN software is removed. Therefore use “Software” of the Windows system administration and remove “EtherCAN […] Host Driver” and “CAN SDK for Windows”.

5.1.4.2.1 Configuration/Test of the CAN Interface under Windows

First connect the CANopen-PN with your PC via USB. Windows will detect the device and provide a new network interface (by means of the RNDIS driver which is installed with the setup).
The CAN software uses the CAN API, which is provided in this case via a TCP/IP connection – which first has to be configured: Therefore start "`Program/esd/EtherCAN/CAN Control Panel`" in the *start menu*:

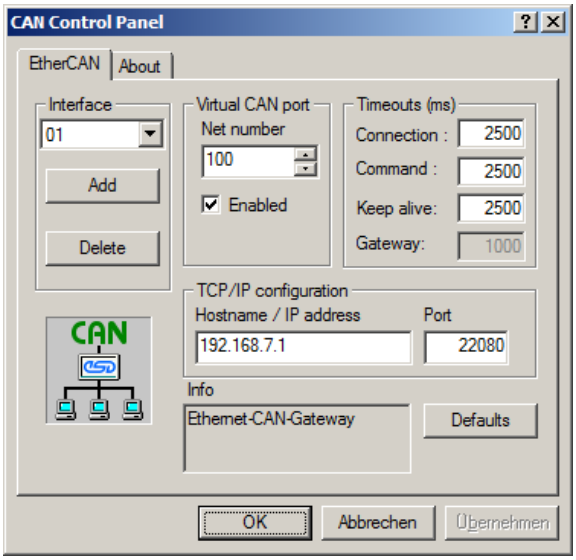


Figure 6: *CAN Control Panel*

When the *CAN Control Panel* window is opened it does already contain the entry of the *Net number* in the field *Virtual CAN port*. The Net number is 100 for the CANopen-PN (also ETHopen_PN, see Figure 7).

The *IP address* is defined for the CANopen-PN by esd. In the field *TCP/IP configuration* the following *IP address* has to be entered: **192.168.7.1**

Do not change the other settings (see NTCAN API Manual Part 2 for details).

Now you can use e.g. CANreal (part of the esd CAN SDK) for testing. It can be started with the start menu entry "`Program/esd/CAN SDK/CANreal`".

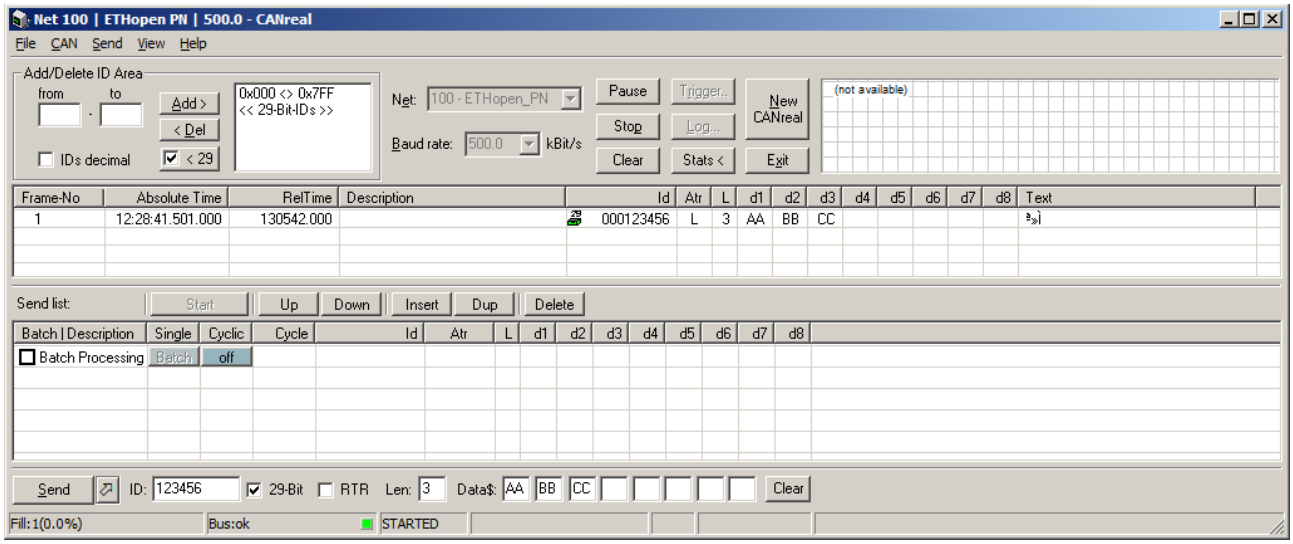


Figure 7: *CANreal*

In the CANreal program window now the net number has to be selected in the input field *Net:*. Select the net No. 100, which is defined for the CANopen-PN.
The entry of the field *Net:* is then: **100 - ETHopen-PN**

Select the baud rate in the input field *Baud rate:* and click the *Start* button. Now all CAN frames received by the CANopen-PN are shown, depending on the configuration of the ID-filter as displayed in the upper left of the CANreal window.

To transmit frames specify the CAN-ID of the frame (e.g. 123456 here) in the input field *ID:* in the lower part of the program window. Then enter the data bytes in the input fields *Data:* and click *Send* to transmit the data.

INFORMATION
For further information see CANreal Manual
(Start menu "`Program/esd/CAN SDK/Documentation`").

5.2 GSDML Composer Quick Start

Step	Action	see page
	NOTICE To guarantee a proper operation of the CANopen-PN make absolutely sure to use version 1.0.6 of the GSDML Composer, that is compatible with the CANopen-PN firmware version 1.0.6, before you start with the software configuration!	-
1.	Open the GSDML Composer e.g. via the Windows start menu. (The installation is described in chapter 5.1.4)	19
2.	Import the EDS files of the CANopen devices, that shall be used, into the device library	27
3.	Insert the selected devices in the CANopen Network Editor now (see chapter 5.6) – e.g. per double click or per “Drag'n'Drop” (see chapter 5.5.1)	28 26
4.	Configure the devices – the Device Editor (see chapter 5.9) can be opened e.g. by double clicking on a device	34
5.	Configure the CANopen-PN with the CANopen Manager (see chapter 5.7)	30
6.	Completed – export the GSD file (menu *File*/*Toolbar*)	
7.	(Usually the file will subsequently be imported into a PROFINET IO engineering-tool, this is described exemplary for the SIMATIC manager in chapter 5.10)	43

5.3 Main Window of the GSDML Composer

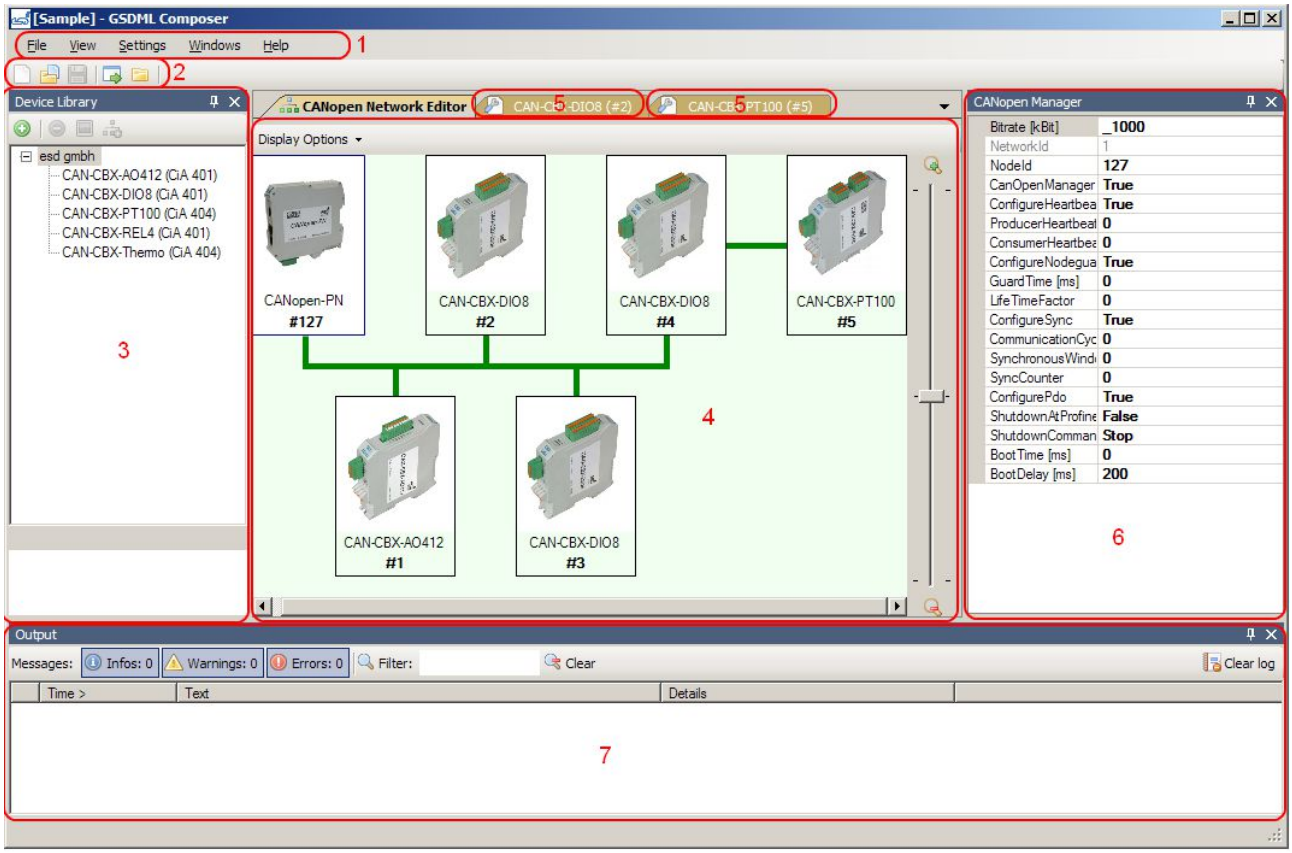


Figure 8: Main window of the GSDML Composer

Legend

1. Menu bar, see 5.4 f.
2. Toolbar, see 5.4.1
3. Device Library, see 5.5
4. CANopen Network Editor, see 5.6
5. Device Editor (in background), see 5.9
6. Configuration of the CANopen Net with the CANopen Manager, see 5.7
7. Output, see 5.8

5.4 Menu Bar of the GSDML Composer

The menu bar contains the following menu items:
- *File*
- *View*
- *Setting*
- *Window*
- *Help*

5.4.1 Menu Item *File,* General Options

The menu item *File* shows the following selection of commands for the processing of the project files. The corresponding buttons of the commands are contained in the toolbar (see Figure 8).

Command	Description	Button in the toolbar
New project	Starts a new project, i.e. all slaves will be removed and all project-specific settings will be reset	
Load project	Loads a project stored on the hard disk etc.	
Save project	Saves the current project into a file. At the initial saving the file name and path have to be specified. The project file comes with the extension **`.xgcp`** and holds the complete project data including the EDS information of the slaves – it can thus be opened in the GSDML Composer without importing the EDS files again.	
Save project as…	As *Save Project*, but file name and path must always be specified in this command.	-
Export GSDML	Generates the GSD file for the current project and stores it in the export directory of the project. If the export directory has not already been specified, the *Project settings* window (see figure 9) will be automatically opened.	
Export DCF	Generates the DCF-files for the current project and stores them in the export directory of the project. If the export directory has not already been specified, the *Project settings* window (see figure 9) will be automatically opened	-
Open export directory	Opens the export directory in the Windows Explorer. If the export directory has not already been specified, the *Project settings* window (see figure 9) will be automatically opened	

5.4.2 Menu Item *View*

This menu item shows a selection of the application windows. By clicking on it the chosen one will be opened or if it is already opened it will be moved into the foreground.
By selecting the commands *Close all device editors* and *Close current device editor window* the corresponding Device Editor windows can be closed.

5.4.3 Menu Item *Settings*

The menu item *Settings* shows the following selection of commands to change the settings:

Command	Description
Project settings	Opens the Project settings, see 5.4.3.1
Disable window docking	Disables moving or docking of windows in the main window
Language	Changes the language of the GSDML Composer. The change will only take effect after an application restart.
GSDML	*File name with time*: Defines whether the name of the GSD file will contain the time, when exported. This option is helpful especially in the development phase, because the file version of the GSD file can only be distinguished by date and time.

5.4.3.1 Project settings

Shows a dialogue for the processing of the project settings:

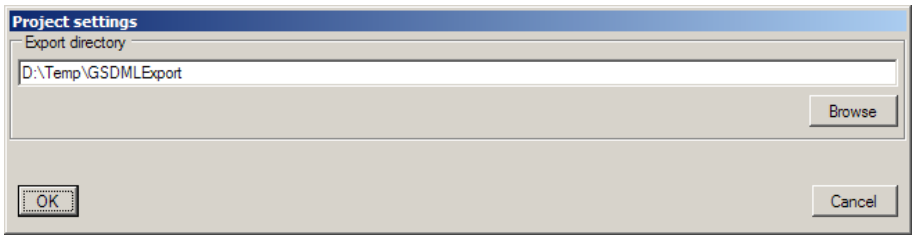


Figure 9: Dialogue *Project settings*

In the input field *Export directory* it can be specified in which directory the GSD and the DCF files will be stored during the export.

5.4.4 Menu Item *Window*

This menu item shows a selection of all available windows/tabs (Device Editor and the CANopen Network Editor). Click on a selected window to activate it.

5.5 Device Library

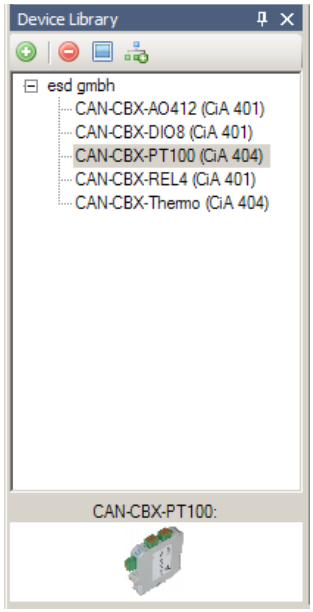


Figure 10: Device Library

The device library contains a list of all installed CANopen device descriptions. The devices can be added to the current net; see next section.

5.5.1 Add CANopen Devices to the current Project

There are three ways to add CANopen devices to the current project:

1. Double click with the left mouse button on an entry of the Device Library
   - The chosen device will be appended to the CANopen net as last device

2. Via the context menu item *Attach in the Network Editor* (Click with the right mouse button on an entry in the Device Library)
   - The chosen device will be appended to the CANopen net as last device

3. “Drag'n'Drop” (Click with the left mouse button on an entry in the device library, hold the button and move the cursor to the CANopen Network Editor)
   - Release the mouse button in an empty range of the CANopen Network Editor: The chosen device will be appended to the CANopen net as last device
   - Release the mouse button on an existing device in the CANopen Network Editor: The chosen device will be inserted before the existing device.

The order of the devices corresponds to the “display order“, see chapter "5.6 CANopen Network Editor".
After a device has been inserted, a dialogue window for the input of the name and the node-ID is opened. This simplifies the first generation of the CANopen network – the values may be changed at any time, see chapter "5.9.1 Configuration of a CANopen Slave via Device Editor".

5.5.2 Edit Device Library

To edit the entries of the device library the context menu of the device library or its toolbar can be used.

Figure 11: Toolbar of the *Device Library*

The following commands can be enabled via the context menu or their corresponding symbols in the toolbar:

Import new slaves into library

Opens a dialogue for the selection of the EDS files.
The selected files are automatically copied into the device Library

INFORMATION
The device library is the "`DeviceLibrary`" subdirectory of the CANopen-PN.
Do not edit it manually!
Additionally the files will be changed when copied to the directory and may therefore not be copied back or used somewhere else.

Delete from slave library

Deletes the selected device from the library

Change default image

Shows a dialogue for the selection of a new graphic file for the selected device.
Supported file extensions are: `.jpg`, `.jpeg`, `.bmp`, `.gif` and `.png`
At the import the standard graphic will be searched for according to the file name of the corresponding EDS file, i.e. for `abc.eds` it will be searched for `abc.jpg`, `abc.png`, etc.

Append to new editor

Adds the selected CANopen device to the current project.
The device will then be shown in the window of the CANopen network editor.

EDS information and device graphics are also copied when the device is inserted in the project, i.e. changing or deleting of entries of the device library will not influence existing projects.

5.6 CANopen Network Editor

In the CANopen Network Editor window the CANopen-PN and all slaves of the CANopen net are displayed and may be visually rearranged.

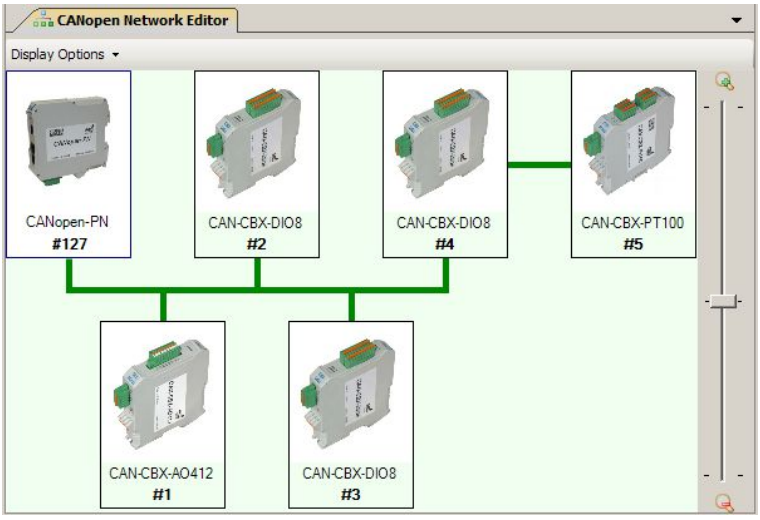


Figure 12: CANopen Network Editor window with manually positioned slaves

Use the scroll bar (on the right in Figure 12) to scale the view of the CANopen Network Editor.

Double click with the left mouse button on a device in the CANopen Network Editor window to open the Device Editor (see 5.9) of the respective device.
A single click opens a small information window which contains e.g. name and node-ID of the device . Furthermore for every slave there is a context menu, see page 29.

The order of the devices, i.e. the way of the connection line, only gives an overview – it has no effect on the exported GSD file. It can be adapted to the node-IDs (and vice versa, see *Reassign Node IDs (by display order)* or *Reset display order (by Node IDs)* below).

5.6.1 Display Options

Via the menu item *Display Options* the arrangement of the devices in the CANopen Network Editor window can be changed.

Calculate positions automatically	Defines whether the devices are rearranged in the CANopen Network Editor automatically if a device is removed or inserted. If the function is disabled, the devices can be arranged via the mouse in the CANopen Network Editor window in any user-defined order.
Recalculate positions now	Recalculates the position of the devices in the CANopen Network Editor window once. If *Calculate position automatically* is enabled, this function will be called automatically when a device is removed or inserted.
Reassign Node IDs (by display order)	Changes the node-IDs of the devices according to the displayed order. ID 1 is assigned to the first device, etc.
Reset display order (by Node IDs)	Resets the way of the connection line on the basis of the node-IDs, i.e. the device with the lowest ID becomes the first, then the connection line is drawn to the device with the next higher ID, etc.

5.6.2 Device Context Menu

Open device editor	Opens the Device Editor (see 5.9) for selected device.
Display order	Changes the index of the device in the display order, i.e. if *Calculate position automatically* is enabled, it will interchange its display position with the preceding/following device (*move up* / *move down*). If *Calculate position automatically* is disabled, only the connection line will be affected – the positions of the device images in the CANopen network editor window remain unaffected.
Clone	Duplicates a device together with its settings and inserts the copy behind the device. At first the copy is assigned with the lowest unused node-ID, but a dialogue allowing a quick change is opened. (As with the inserting of a new device, see 5.5.1)
Delete	Deletes the device
Change device image	Shows a dialogue for the selection of an image file and changes the image of the device accordingly.

INFORMATION
Supported file extensions for the image files are:
`.jpg`, `.jpeg`, `.bmp`, `.gif` and `.png`

5.7 Configuration of the CANopen Net with the CANopen Manager

Here the general settings of the CANopen-PN and its CANopen Manager are specified.

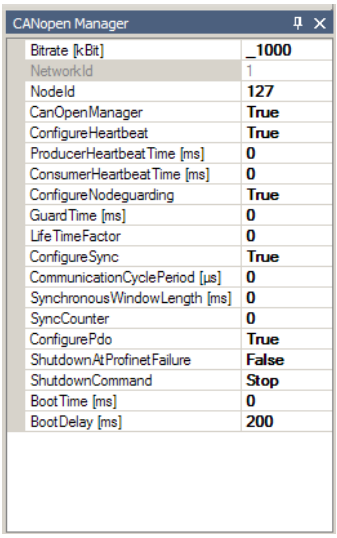


Figure 13: CANopen Manager window

Bitrate — The bit rate of the CANopen net, "_1000" represents 1000 kbit/s, etc.

NetworkId — Currently not supported

NodeId — The CANopen device address of CANopen-PN

CanOpenManager — Defines whether the CANopen-PN works as CANopen Manager.
The default value is `True` (module is CANopen Manager).
`False` can only be used if several CANopen-PN modules are used simultaneously.
In addition in this case the following options can not be used:

ConfigureHeartbeat	*ShutdownAtProfinetFailure*
ConfigureSync	*ShutdownCommand*
ConfigurePdo	*BootTime*
ConfigureNodeguarding	*BootDelay*

ConfigureHeartbeat — Defines whether the heartbeat objects of the slaves are written. (For the value `False` the heartbeat settings defined via the Device Editor will be ignored and the default configuration will be used)

ProducerHeartbeatTime — The interval (in ms), in which the CANopen Manager generates the heartbeat messages. ('0' to disable)

ConsumerHeartbeatTime — The interval (in ms), in which the slaves, configured as heartbeat producers, expect the heartbeat messages.
The *ConsumerHeartbeatTime* must exceed the maximum *Heartbeat Producer* interval of a slaves. ('0' to disable)

ConfigureNodeguarding — Defines whether the Node guarding objects of the slaves are written. (If the value is `False`, the Node guarding settings defined via the Device Editor will be ignored and the default configuration will be used)

GuardTime — The NMT-Master transmits the Guarding-Requests to the CANopen-PN with this interval (in ms).
(Only valid if the *LifeTimeFactor* exceeds 0. '0' to disable)

LifeTimeFactor — The *GuardTime* multiplied by this factor results in the *Node Lifetime*: the period, after which an error is reported, if no Guarding-Requests are received.
(Only valid if *GuardTime* exceeds 0. '0' to disable)

ConfigureSync — Defines if the SYNC objects of the slaves are written.
(If the value is `False`, the SYNC-settings defined via the Device Editor will be ignored and the default configuration will be used.)

CommunicationCyclePeriod — In this period (in µs) the CANopen-PN generates SYNC messages. ('0' to disable)

INFORMATION
Please note that unlike other times which have to be specified here, the *CommunicationCyclePeriod* has to be entered in **µs**!

SynchronousWindowLength — Currently not evaluated

SyncCounter — Currently not evaluated

ConfigurePdo — Determines whether the PDO objects of the slaves are written. (If the value is `False`, the PDO settings defined via the Device Editor will be ignored and the default configuration will be used.)

ShutdownAtProfinetFailure — Defines whether the manager of the CANopen net shuts down in case of a failure on the PROFINET IO side (connection terminated or no connection to PROFINET IO master). See also *ShutdownCommand* below.

ShutdownCommand — Specifies which command is executed by the NMT master in case of *ShutdownAtProfinet Failure*. (See also CiA 301 [2])

- *Start*: *Service start remote node* is executed
- *Stop*: *Service stop remote node* is executed
- *PreOp*: *Service enter pre-operational* is executed
- *Reset*: *Service reset node* is executed
- *ResetComm*: *Service reset communication* is executed

BootTime

Specifies the period (in ms), the manager waits for the necessary slaves (see 5.9.4) before an error is reported.

INFORMATION
If during the *BootTime* no communication with the connected CANopen modules has been possible due to a failure of the CAN bus, the CANopen-PN terminates the boot process and does not even try to start the modules after the failure is ended.
In this case the command "Reset Controller" (see 6.1.15) must be called to restart the CANopen boot process after the problem has been resolved.

BootDelay

The period (in ms), the manager waits after the *NMT Reset Communication* command, before he continues with the boot process of the slaves.

INFORMATION
For basic information about CANopen refer to chapter “CANopen Firmware” on page 88.
For further information please read the CANopen specification CiA 301 [2].

5.8 Output

The application messages are listed in this window.

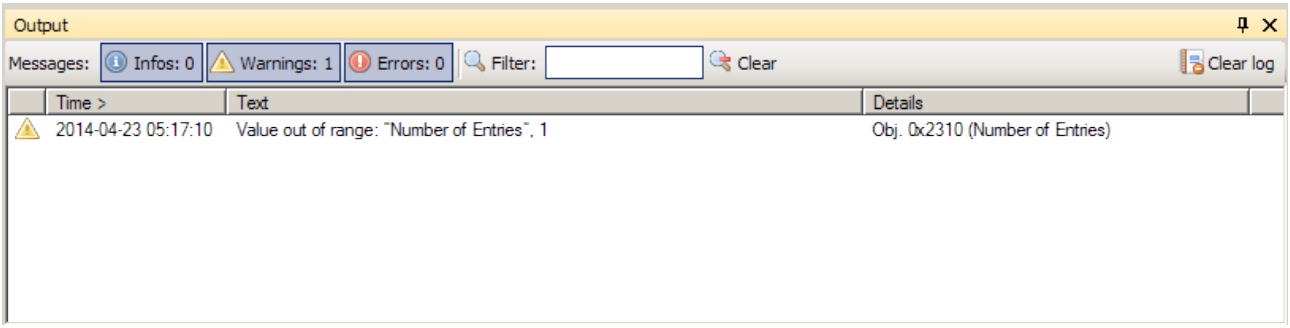


Figure 14: Output window

All messages are of type `Information`, `Warning` or `Error` and they are marked by different symbols in the first column.
The type can be selected with the corresponding buttons *Infos*, *Warnings* and *Errors* in the Output menu bar under *Messages:*
The entry of a message consists of the time, a text and further details, see Figure 14.

With *Filter:* the messages can be filtered by a user-defined text. Only the messages which contain this text in the *Text* column are displayed.
Via the *Clear* button the filter can be deleted.

Via *Clear log* the complete list is deleted. Entries which are currently filtered out are also deleted (The complete log file can be stored in a text file via the context menu of the list.)
The number of entries in the list is limited. If the number is exceeded, the oldest entries will be deleted. Furthermore all entries will be rejected if the GSDML Composer is closed.

5.9 Configuration of a CANopen Slave via Device Editor

The Device Editor contains all device-specific configuration options of the selected CANopen slave (exemplary for CAN-CBX-PT100 here).
Depending on the selection in the tree structure (Figure 4, left) another page of the Device Editor is shown. The different pages are described below.

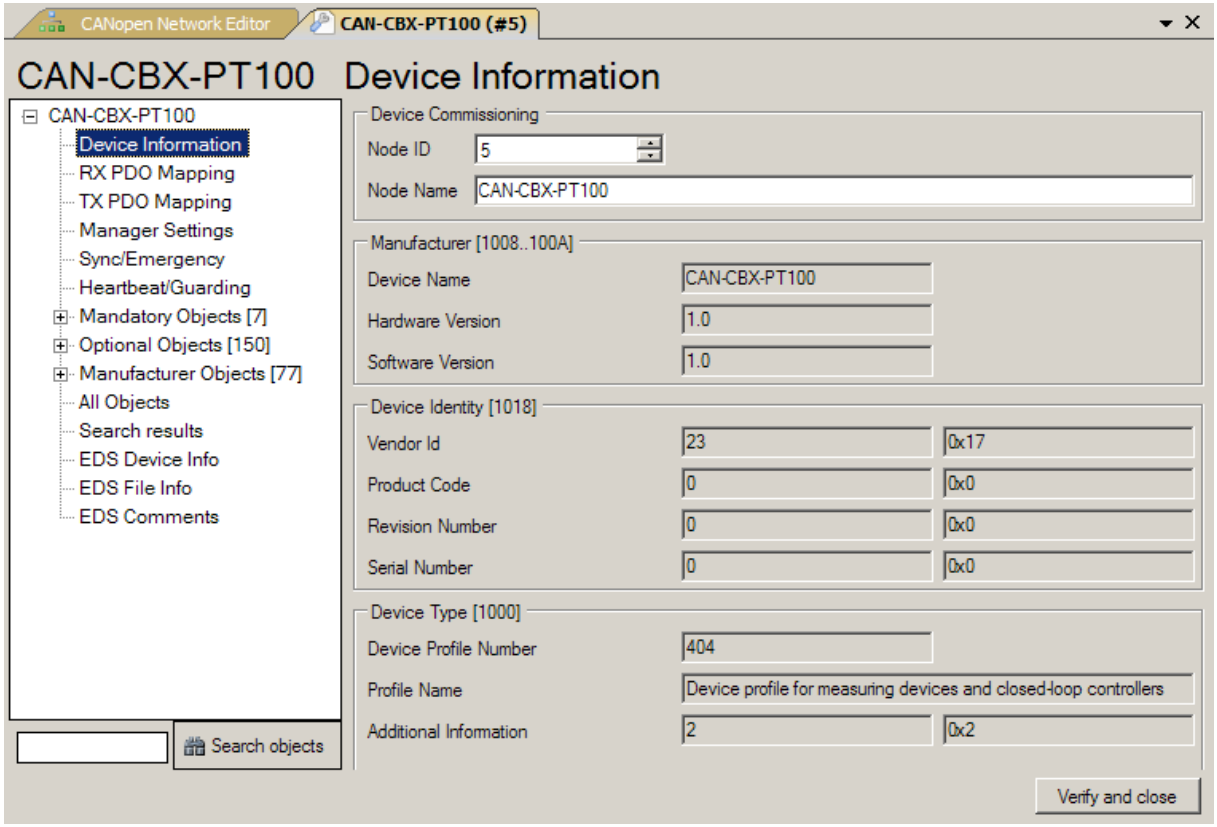


Figure 15: Device Editor with page *Device information* (Example: CAN-CBX-PT100)

INFORMATION
For basic information about CANopen refer to chapter “CANopen Firmware” on page 88.
For further information please read the CANopen profile CiA 301 [2].

5.9.1 Device Information

On this page the device name (*Node Name)* and the *Node ID* can be configured (according to the DCF-section `[DeviceComissioning]`.).
If the entered node-ID is already assigned to another device, the entry will not be accepted.

Furthermore the general information about the device are shown. (Extracted from the CANopen objects 0x1000, 0x1008 up to 0x100a and 0x1018)

5.9.2 RPDO Mapping

On the *RPDO Mapping* page the PDOs can be configured.

INFORMATION
The number of PDO pairs (TPDO, RPDO, T/RPDO) is limited to 15 per connected CANopen participant.

The current Composer (version 1.0.0) observes this limit and does not allow the generation of a GSDML file.

The window consists of the *object selection* (top left), the *PDO selection* (top right), the buttons to change the mapping parameters (top centre) and the *Communication Parameters*.

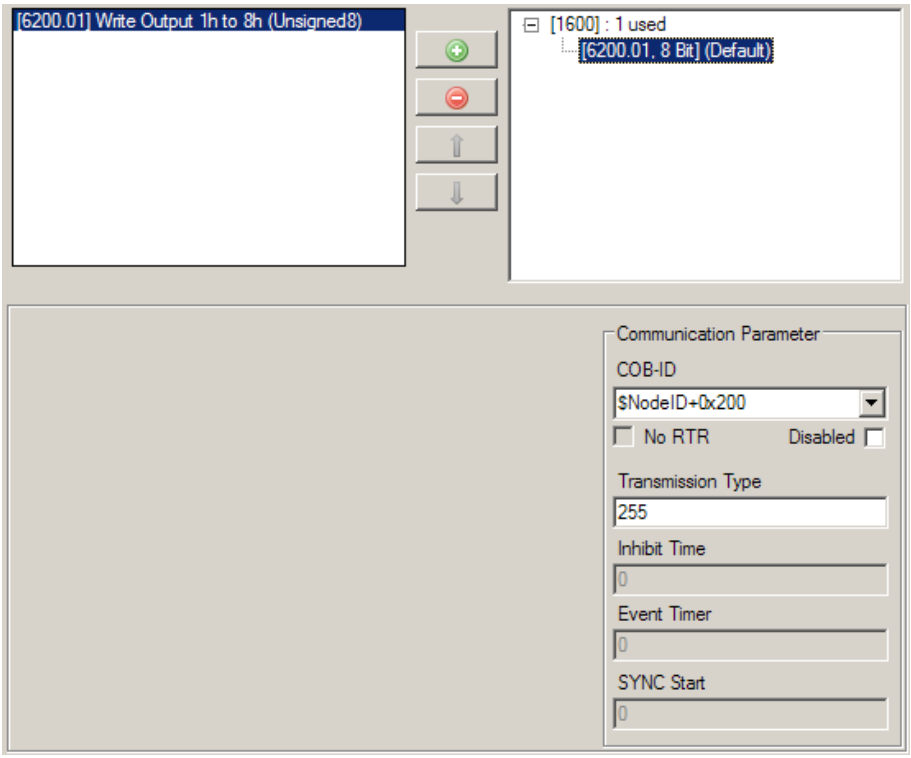


Figure 16: PDO Mapping

Object selection
(Figure 6, top left)

This list shows all objects that can be mapped in a PDO.
(Defined in the EDS file by *PDOMapping* and *AccessType* of the object)

PDO selection
(Figure 6, top right)

In this tree structure all PDOs and their contents are shown.

The example in Figure 16 shows the PDO 0x1600 containing the mapped object “[6200.01, 8 Bit] (Default)”:
The object mapped is the object 0x6200, sub-index 1 with 8-bit data width.
“Default” indicates that this value complies with the EDS-“DefaultValue” of the PDO object

Communication Parameter

In this field the COB-ID and the transmission type of the PDO can be specified.

COB-ID
The CAN-ID of the selected PDO.
In accordance with CiA 306 [4] in the EDS file "$NodeID+" may be used as prefix. This is allowed here too, and the COB-ID is calculated when exported.

No RTR
Remote Transmit Request

Disabled
Here the process data channel, containing the COB-ID specified above, can be disabled.

Transmission Type
Equivalent to the *Transmission Type* of the TPDOs as seen from the CANopen-PN, see *Transmission Type* in chapter 5.9.3

All other parameters do only apply for TPDOs, see chapter 5.9.3

5.9.2.1 Change Mapping Parameter via Buttons

Insert an object in a PDO
With the -button the object selected in the field *Object selection* is inserted in the selected PDO. Size and granularity will be taken into account.
In the event of a fault the object will not be inserted (without further acknowledgement).

Dummy Objects
Via the context menu of the *PDO selection* dummy objects can be inserted into RPDOs (types/indices according to CiA 306 , e.g. "Unsigned8" as object Index 5)

Delete an object in a PDO
Clicking the -button deletes the object in the PDO, which is selected in the *PDO selection*.

Move an object in a PDO
A selected object can be positioned in the *PDO selection* with the buttons and .

5.9.3 TPDO Mapping

The structure of the page *TPDO Mapping* corresponds to the structure of the page *RPDO Mapping* but with the difference that TPDOs are configured and, if applicable, additional communication parameters are available.
The TPDO Mapping window consists of the *object selection* (top left), the *PDO selection* (top right), the buttons to change the mapping parameters (top centre) and the *Communication Parameter*.

Communication Parameter:

COB-ID — See chapter 5.9.2

Transmission Type — Specifies the transmission type/rate:
- 0x00: Acyclic with SYNC
- 0x01: Cyclic with every SYNC
- 0x02 .. 0xF0: Cyclic with every SYNC as specified here (2nd up to 240th.)
- 0xFC: On RTR (sampling of the values to SYNC)
- 0xFD: On RTR (sampling of the values at RTR)
- 0xFE: Event based/manufacturer-specific
- 0xFF: Event based/profile-specific

SYNC: Transmission will be triggered at reception of a SYNC message (and within the SYNC-time window)

Event based: The transmission is independent of the SYNC

Inhibit Time — Minimum period between two transmissions
(As multiple of 100 µs. Only if *Transmission Type* is 0xFE/0xFF)

Event Timer — For a value which does not equal zero, this value specifies the cycle time of the transmission in ms (Only if *Transmission Type* is 0xFE/0xFF).

SYNC Start — The SYNC message with this counter value will be evaluated as first received SYNC message
(0: The counter value will be ignored. Only if *Transmission Type* is ≤ 0xF0)

Changing the mapping parameters via the buttons can be carried out as described for the RPDO Mapping, see page 36.

INFORMATION
The number of PDO pairs (TPDO, RPDO, T/RPDO) is limited to 15 per connected CANopen participant.

The current Composer (version 1.0.0) observes this limit and does not allow the generation of a GSDML file.

5.9.4 Manager Settings

The settings of the CANopen Manager for the slave can be specified on this page.

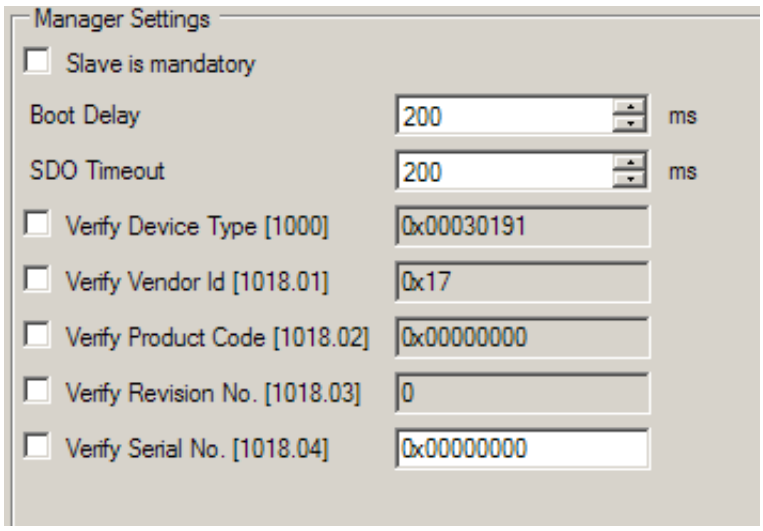


Figure 17: Manager Settings

Slave is mandatory — Defines whether the complete CANopen network may be started if this slave is missing or if errors occurred during initialisation of this slave.

Boot Delay — The period (in ms), the CANopen manager waits at a reset of this slave in case of an error after the *NMT Reset Communication* command or *NMT Reset Application*-command, before he continues with the boot process of this slaves.

SDO Timeout — The maximum period (in ms), the CANopen Manager waits during an SDO request for the answer of the slave.

Verify Device Type [1000] — Defines whether the slave may only be started if the device type exactly matches this value.

Verify Vendor Id [1018.01] — Defines whether the slave may only be started if the vendor Id exactly matches this value.

Verify Product Code [1018.02] — Defines whether the slave may only be started if the product code exactly matches this value.

Verify Revision No. [1018.03] — Defines whether the slave may only be started if the revision number exactly matches this value.

Verify Serial No. [1018.04] — Defines whether the slave may only be started if the serial No. exactly matches this value.

5.9.5 SYNC/Emergency

On the *SYNC/Emergency* page the settings for the transmission of SYNC-, TIME- and Emergency messages can be specified.

SYNC Parameter:

Node generates SYNC Messages — Defines whether this slave generates the SYNC messages of the CANopen network.

> **NOTICE**
> Ensure that only one device in the CANopen network generates the SYNC messages!

CAN-ID — The CAN-ID of the SYNC messages

Comm. Cycle Period (Communication cycle period) — Interval of the SYNC messages

Window Length — The period after a SYNC message, in which the TPDOs may be transmitted.
(Only applicable for synchronised PDOs, i.e. PDOs with *Transmission Type* ≤ 0xF0)

TIME Parameter:

Node generates TIME messages — Defines whether this slave generates the TIME messages of the CANopen network.

> **NOTICE**
> Ensure that only one device in the CANopen network may generate the TIME messages!

Node consumes TIME messages — Defines whether this slave is the *Time Stamp Consumer*, i.e. it requires/shall use TIME messages.

CAN-ID — The CAN-ID of the TIME messages

Emergency Object:

Device generates Emergency — Specifies whether this slave may generate Emergency messages.

CAN-ID — The CAN-ID of the Emergency message

Inhibit Time — The minimum period between two Emergency messages (specified as multiple of 100 µs)

5.9.6 Heartbeat/Guarding

Use this page to specify the settings of the *Heartbeat* and *Node Guarding* protocols.

Node Guarding:

Guard Time — The NMT master transmits guarding requests to the slave in this interval.

Lifetime Factor — The *Guard Time* multiplied by this value results in the *Node Lifetime.* This is the period after which the slave reports an error if there have not been any guarding requests received.

INFORMATION
Both values *Guard Time* and *Lifetime Factor* must be greater than '0' to keep the Node Guarding enabled.

Heartbeat:

Heartbeat Producer, Heartbeat Time — If a value greater than '0' is specified here, this slave is *Heartbeat Producer* and generates Heartbeat messages in the interval specified.

Heartbeat Consumer, Interval — This value specifies the interval in which the *Producer* (the CANopen-PN here) expects the Heartbeat messages. ('0' to disable)
This value must be sufficiently greater than the interval of the *Producer*, because otherwise missing heartbeats might result, only due to the jitter at transmission/reception.

INFORMATION
It is not possible to use both protocols simultaneously. If both protocols are configured, the Heartbeat protocol is used.

5.9.7 Object Lists (various)

On the pages *Mandatory Objects, Optional Objects* and *Manufacturer Objects* the corresponding object lists are displayed.
The object lists display the CANopen objects with their values and types etc.

The following pages of the Device Editor are **object lists**:

Mandatory Objects Lists the objects of the EDS section `[MandatoryObjects]`

Optional Objects Lists the objects of the EDS section `[OptionalObjects]`

Manufacturer Objects Lists the objects of the EDS section `[ManufacturerObjects]`

All Objects All objects are listed

Index [Hex]	Subindex	Name	Default	Value	Data Type	Min	Max	Type	Access	PDO Mapping
1000	0	Device Type	0x00030191	0x00030191	Unsigned32			Var	Ro	False
1001	0	ErrorRegister	0x0	0x0	Unsigned8	0x0	0xff	Var	Ro	False
1003		Pre-defined Error Field			None			Array	None	False
1003	0	Number of Errors	0x0	**10**	Unsigned8			Var	Ro	False
1003	1	Standard Error Field	0x0	0x0	Unsigned32			Var	Ro	False
1003	2	Standard Error Field	0x0	0x0	Unsigned32			Var	Ro	False
1003	3	Standard Error Field	0x0	0x0	Unsigned32			Var	Ro	False
1003	4	Standard Error Field	0x0	0x0	Unsigned32			Var	Ro	False
1003	5	Standard Error Field	0x0	0x0	Unsigned32			Var	Ro	False
1003	6	Standard Error Field	0x0	0x0	Unsigned32			Var	Ro	False
1003	7	Standard Error Field	0x0	0x0	Unsigned32			Var	Ro	False
1003	8	Standard Error Field	0x0	0x0	Unsigned32			Var	Ro	False
1003	9	Standard Error Field	0x0	0x0	Unsigned32			Var	Ro	False
1003	A	Standard Error Field	0x0	0x0	Unsigned32			Var	Ro	False
1005	0	COB-ID SYNC-message	0x80		Unsigned32	0x1	0x7FF	Var	Rw	False
1008	0	ManufacturerDeviceName	CAN-CBX-DIO8 e...	CAN-CBX-DIO8 e...	VisibleString			Var	Const	False
1009	0	ManufacturerHardwareVersion	1.0	1.0	VisibleString			Var	Const	False
100A	0	ManufacturerSoftwareVersion	1.0	1.0	VisibleString			Var	Const	False
100C	0	GuardTime	0x0		Unsigned16	0x0	0xffff	Var	Rw	False
100D	0	LifeTimeFactor	0x0		Unsigned8	0x0	0xff	Var	Rw	False

Figure 18: Object list

With the search function *Search Objects* of the device manager (Figure 15, page 34 bottom left) on the page *Search results* a list with all objects is generated whose name contain the specified text.
If a four-digit text is entered, it will be interpreted as object index in hexadecimal form and the corresponding object with its sub-objects will be shown.

The entries of the column *value* can be modified if the access rights of the object contain a write access and the values are not edited by other pages.

Generally all objects with indices < 0x2000 are considered to be fixed, with exception of: 0x1028, 0x1029, [0x1200..0x12FF] and [0x1FA0..0x1FFF].

The modified values are transferred in the GSD file and the CANopen Manager will write them into the slaves during initialisation.

The following **columns** are contained in an object list:

Index / Sub-index	Index/Sub-index of the object (EDS section name)
Name	Name of the object (`ParameterName` in EDS)
Default	Default value of the object (`DefaultValue` in EDS)
Value	Current value of the object (`ParameterValue` in EDS)
	Yellow background: value can be edited in the object list
	Red background: invalid value (e.g. value lies outside the value range Min./Max.)
	Font „**bold**“: value differs from default value
Data type	Data type of the object (`DataType` in EDS)
Min	Minimum value (`LowLimit` in EDS)
Max	Maximum value (`HighLimit` in EDS)
Type	Type of the object (`ObjectType` in EDS)
Access	Access rights of the object (`AccessType` in EDS)
PDO Mapping	Specifies whether the object is PDO mappable (`PDOMapping` in EDS)

5.9.8 EDS Device Info

Displays the EDS section `[DeviceInfo]`.
For information only – cannot be changed.

5.9.9 EDS File Info

Displays the EDS section `[FileInfo]`.
For information only – cannot be changed.

5.9.10 EDS Comments

Displays the EDS section `[Comments]`.
For information only – cannot be changed.

5.10 Insert the GSD File in the SIMATIC-Manager

Installation of the GSD File
To install the GSD file select the menu item *Options* in the Hardware Configurator and go on with *Install GSD Files*.

The window *Install GSD Files* opens.

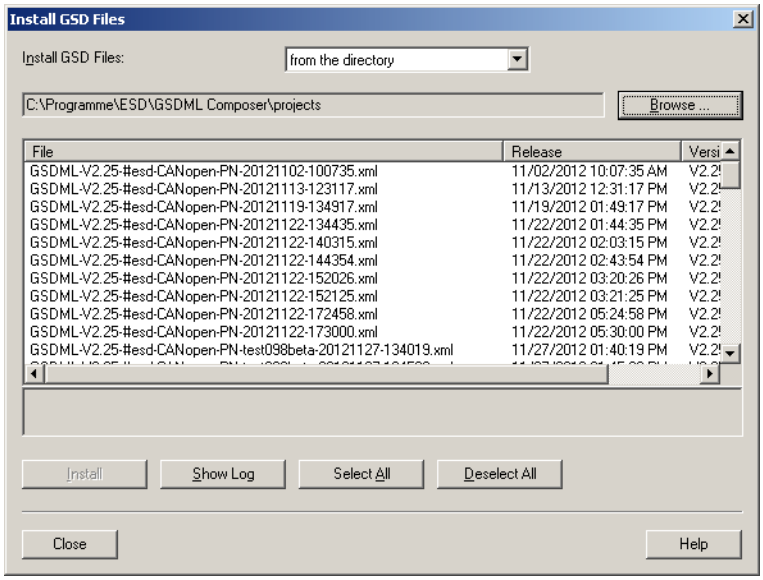


Figure 19: Install GSD files

Select the GSD file you want to install and confirm the selection with the button *Install* (see Figure 19). The GSD file will be installed now.

After installation the CANopen-PN is shown in the tree structure in the window of the hardware configurator (see Figure 20, right bottom) under: *Additional Field Devices/Gateway/CANopen Profinet-IO*.

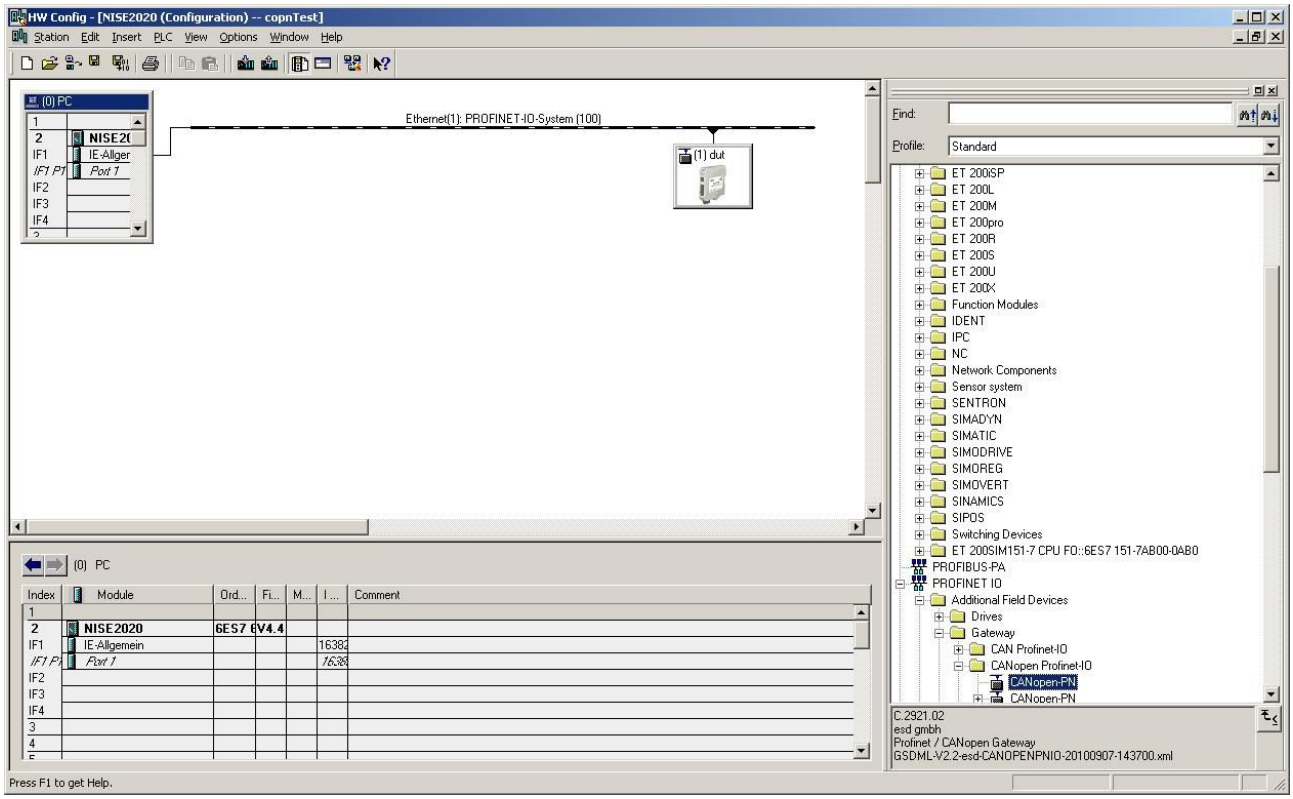


Figure 20: Select GSD file

Select the CANopen-PN in the tree structure and move it via “Drag'n'Drop” to the chosen Ethernet PROFINET IO system, which is shown in the upper left window area.

In the table (bottom left) only the CANopen-PN and the properties contained in the GSD file are displayed.
To open the *Properties* window double-click with the left mouse button on a row in the table.
In the *Properties* window the parameters are displayed as contained in the GSD file. The parameters cannot be changed here. See the following examples.

Example 1: Properties - CANopen Manager

In this example the CAN bus parameters of the CANopen Manager, like CAN bit rate, node-ID etc. are displayed. For further information about the parameters see chapter 5.7.

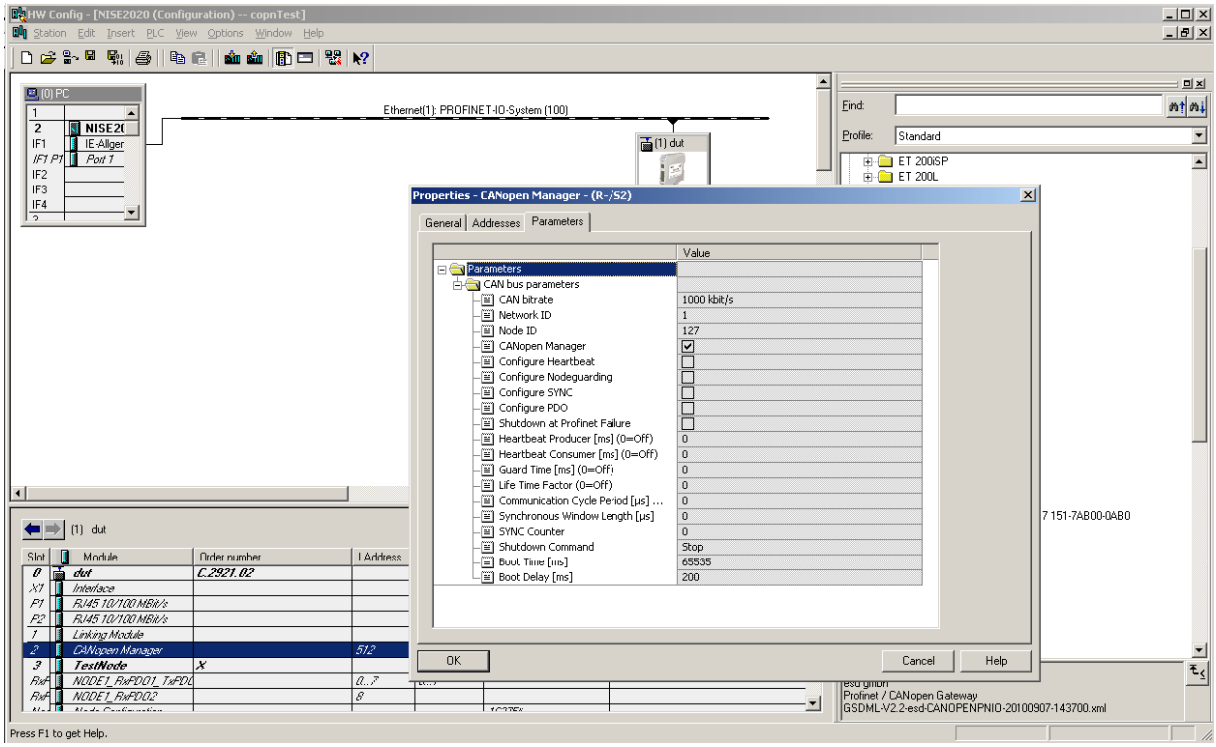


Figure 21: CANopen Manager properties

Example 2: Properties - Node Configuration

In this example the properties of the first connected node are shown. For further information about the parameters see chapter 5.9.

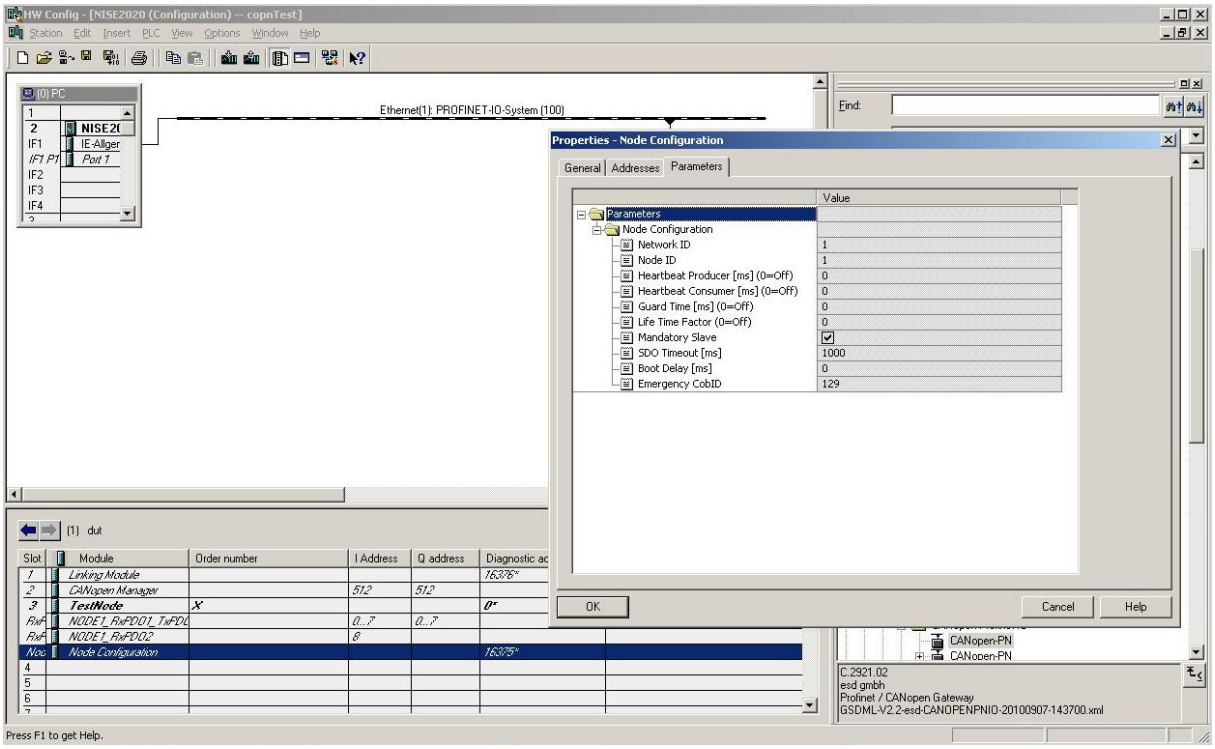


Figure 22: Node Configuration

Example 3: Properties - NODE1_RPDO1_TPDO1

In this example the properties of the PDOs of the first connected node are shown. For further information about the parameters see chapters 5.9.2 or 5.9.3

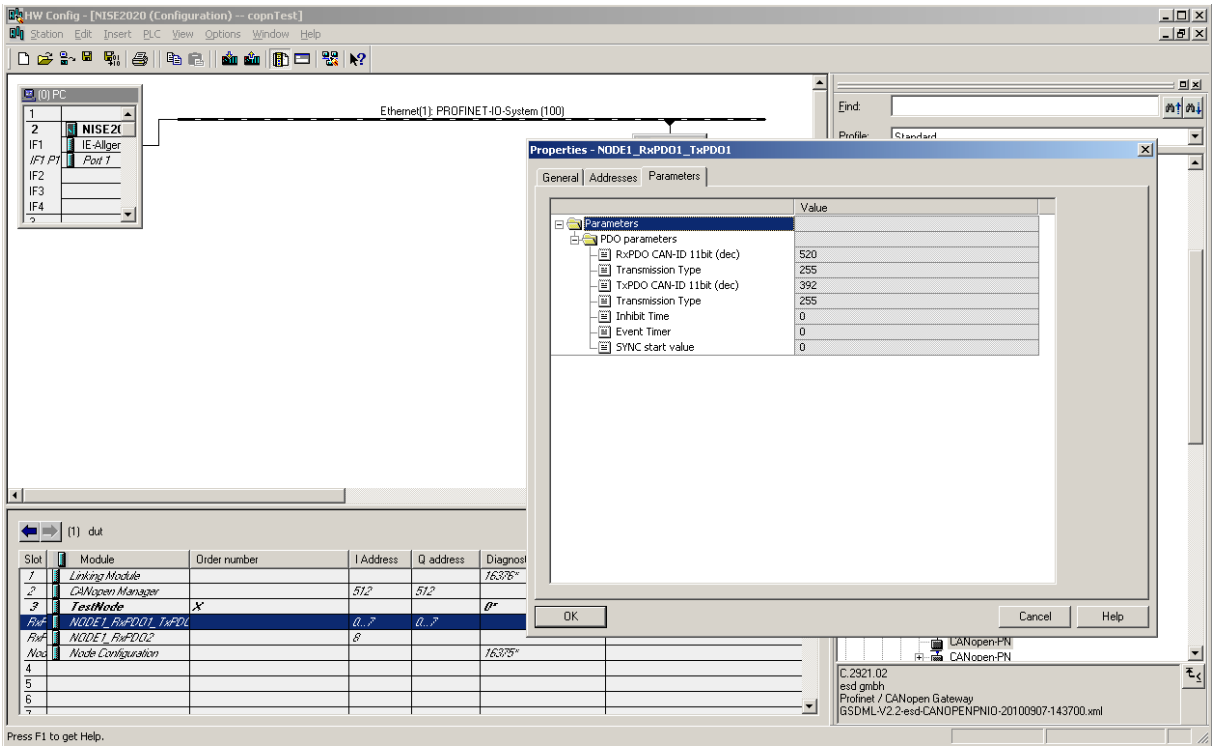


Figure 23: RPDO TPDO

INFORMATION

The number of PDO pairs (TPDO, RPDO, T/RPDO) is limited to 15 per connected CANopen participant.

The current Composer (version 1.0.0) observes this limit and does not allow the generation of a GSDML file.

5.11 Using GSDML File and CANopen-PN Gateway with the TIA Portal

INFORMATION
This chapter does only describe the steps which are relevant for the usage of the GSDML file and the CANopen-PN gateway with the TIA Portal.
Please read the documentation of your TIA Portal for a detailed description.

5.11.1 Quick Start

Step	Action	see page
	NOTICE To guarantee a proper operation of the CANopen-PN make absolutely sure to use version 1.0.6 of the GSDML Composer, that is compatible with the CANopen-PN firmware version 1.0.6, before you start with the software configuration!	-
1.	Disconnect the online connection in your TIA-Portal, because the hardware and software have to be complied in offline mode!	-
2.	Change into the project view of your TIA-Portal.	-
3.	Install the GSDML file as described in chapter „Installation of the GSDML File“.	48
4.	Insert the CANopen-PN in your project, see chapter „Insert CANopen-PN Hardware and Network Configuration“.	50
5.	Compile and load the hardware and software as described in chapter „Compile and Download Hardware and Software“.	53
6.	Go online again. See Fig. 36.	54

5.11.2 Installation of the GSDML File

Change in the program window of your TIA-Portal into project view to import the device description file. Click onto menu item *Options* in the main menu.

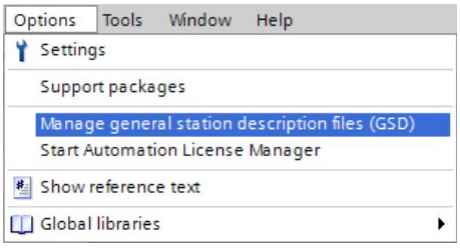


Figure 24: *Extras/Manage GSD*

Choose *Manage general station description files (GSD)* in the pull-down menu, see Figure 24.

Now the window *Manage general station description files* opens.

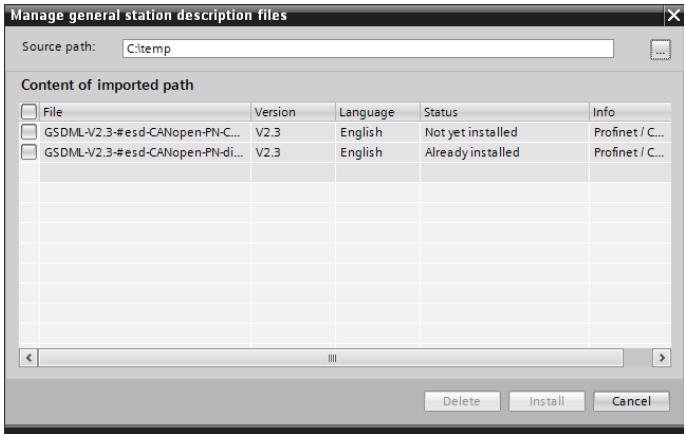


Figure 25: Window *Manage general station description files*

If the path to the folder is not already entered in the input field *Source path* correctly, click onto the button [...] in the upper right part of the window *Manage general station description files* to choose the path.

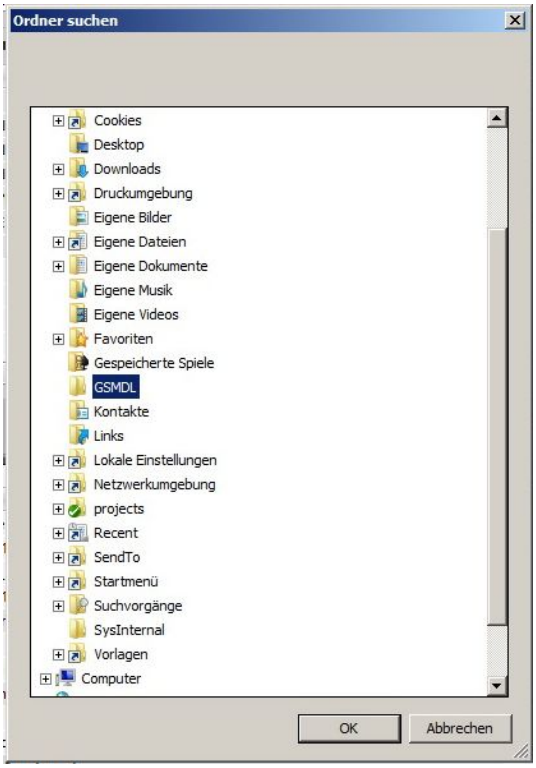
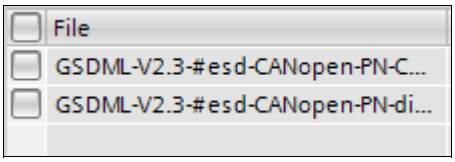


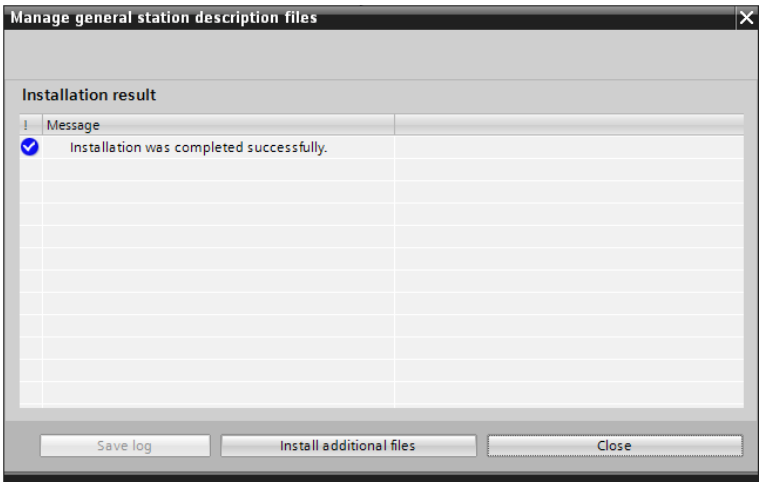
Figure 26: Select the path to the folder

In this window you can now select the path to your GSDML folder.
Click on the button *OK* to confirm the path.



All available GSDML files are listed in the *Manage general station description files window* now.
Activate the corresponding check-box on the left, to select a file, see Figure 27.
Click onto the button *Install* to confirm the selection.

Figure 27: Activate check-box



If the installation of the selected file has been successful, you see the message as shown in the window on the left.

Figure 28: Installation was completely successful

5.11.3 Insert CANopen-PN Hardware and Network Configuration

Now you are able to assemble the devices and nets for your project in the TIA-Portal.
Therefore click under *Project tree / Devices* in the upper left part of the window onto *Devices & networks* as shown in the following figure.

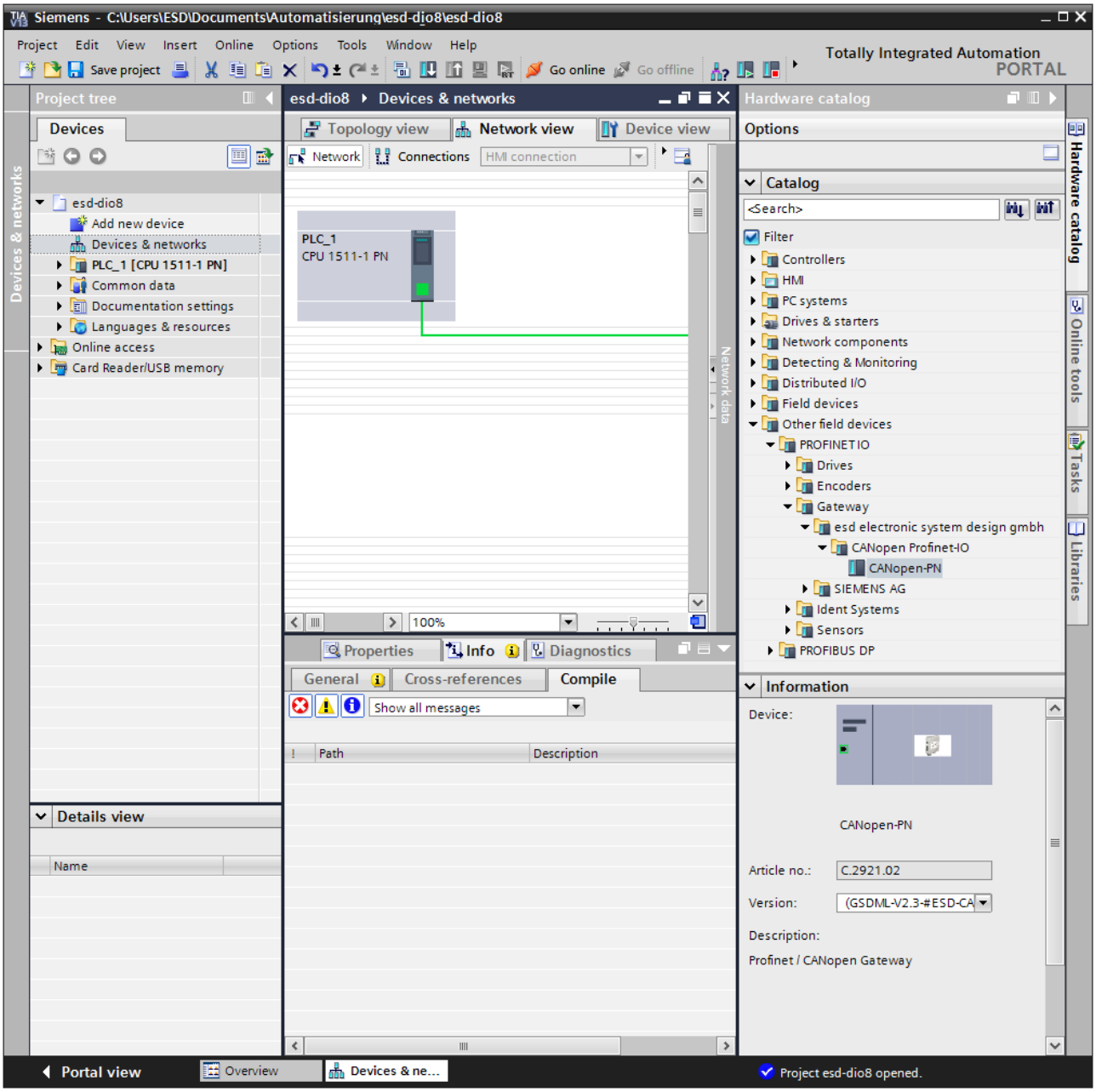


Figure 29: Devices and networks

Choose now your CANopen-PN device the in the tree structure of the Hardware C*atalogue,* which is in the right part of the window.

IMPORTANT
Please note, that it is absolutely necessary that you select the appropriate CAN network configuration for your CANopen-PN device!
Therefore please proceed as described in the following.

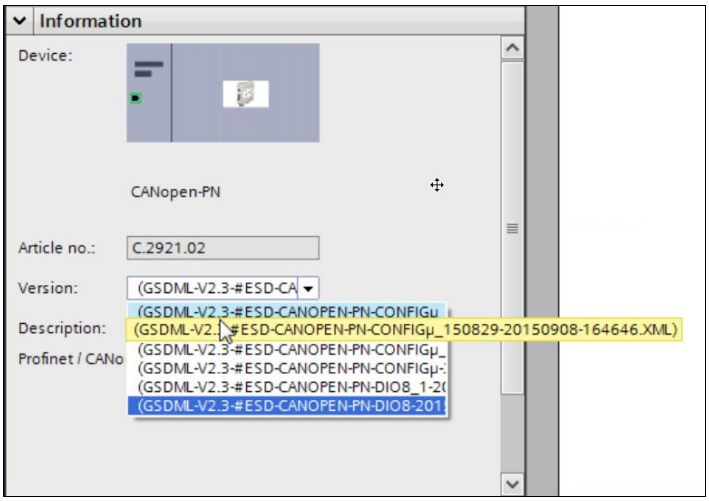


Figure 30: Choose the version (detail)

In the field *Information,* which is in the lower right part of the TIA-Portal program window, the CANopen-PN is listed with *Article no.* and *Version*.

Select the needed CAN network configuration in the pull-down menu of *Version*.

In the example (see Figure 30) the following file is displayed:
`GSDML_V2.3_#ESD-CANOPEN-PN-CONFIGµ_0150829-20150908-164646.XML`

The name of the file is composed of the preceding `GSDML_V2.3_#ESD-CANOPEN-PN-` which is created by the GSDML-Composer and following: *`Project name - Date - Time of creation .XML`*
(in the example : `CONFIGµ_0150829 - 20150908 - 164646 .XML`).

To import the device in your project, select the CANopen-PN with the left mouse button in the tree-structure in the field *Catalog,* drag it into the field *Network view* and drop it.

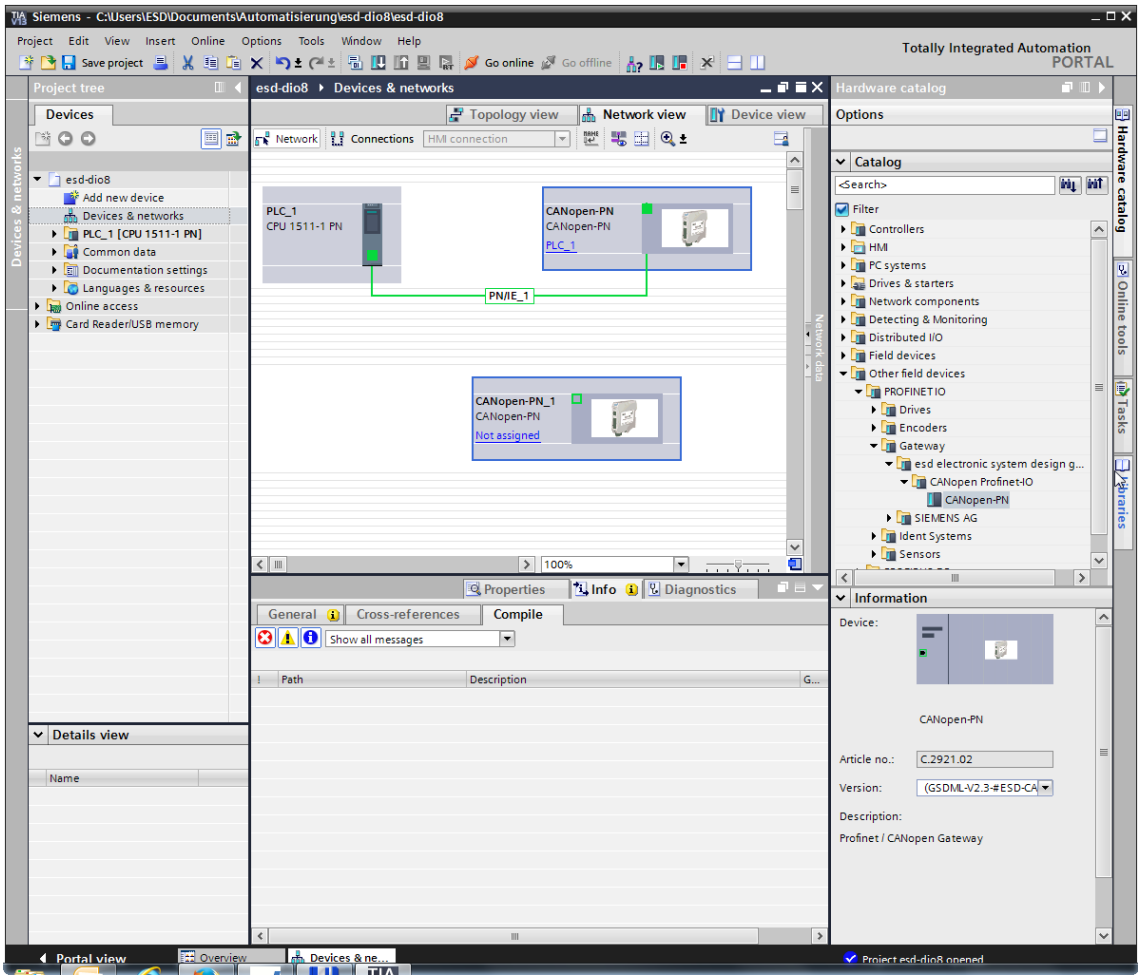


Figure 31: Program window with new CANopen-PN in the window *Network view*

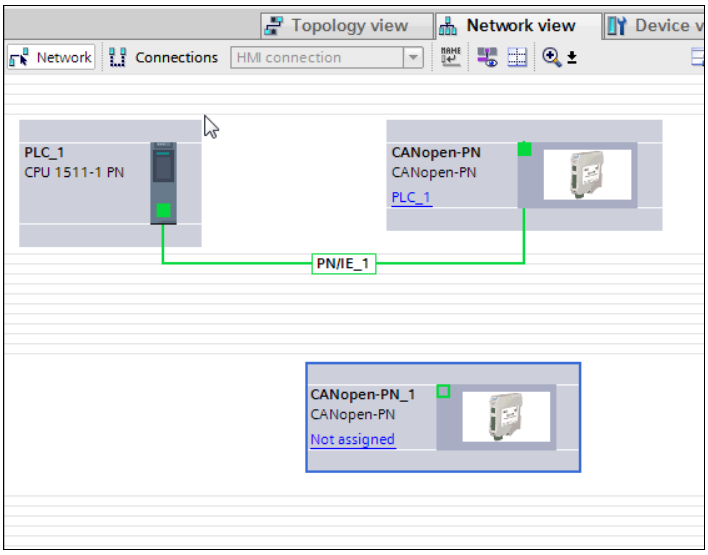


Figure 32: Net view (detail)

The CANopen-PN is now displayed in the field *Network view* as CANopen-PN_1, but it is still not connected to PROFINET (see Figure 32).

To assign the device, click onto the button *Not assigned.*

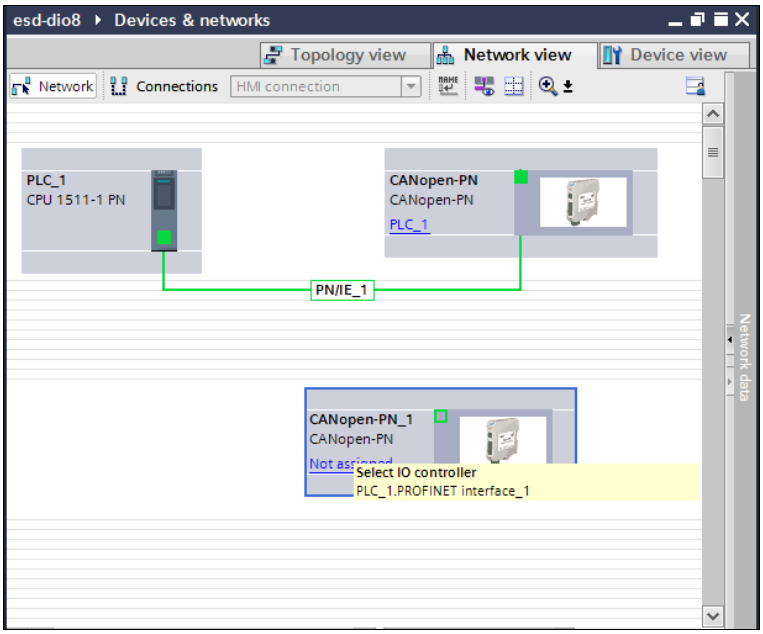


Figure 33: Select IO controller

The drop-down menu *Select IO controller* is opened. In this example the *PLC_1.PROFINET Interface_1* can be chosen.

Now choose the network, to which the CANopen-PN shall be connected, by a mouse-click.

The CANopen-PN is connected to the network now.

5.11.4 Compile and Download Hardware and Software

Before you can download the software, it must be compiled.
To compile the software select your device (*PLC_1 here)* in the field *Project tree / Devices* and further in the pull-down menu *Compile* and then *Hardware and Software (only Changes)*.
You must be still in offline mode for this.

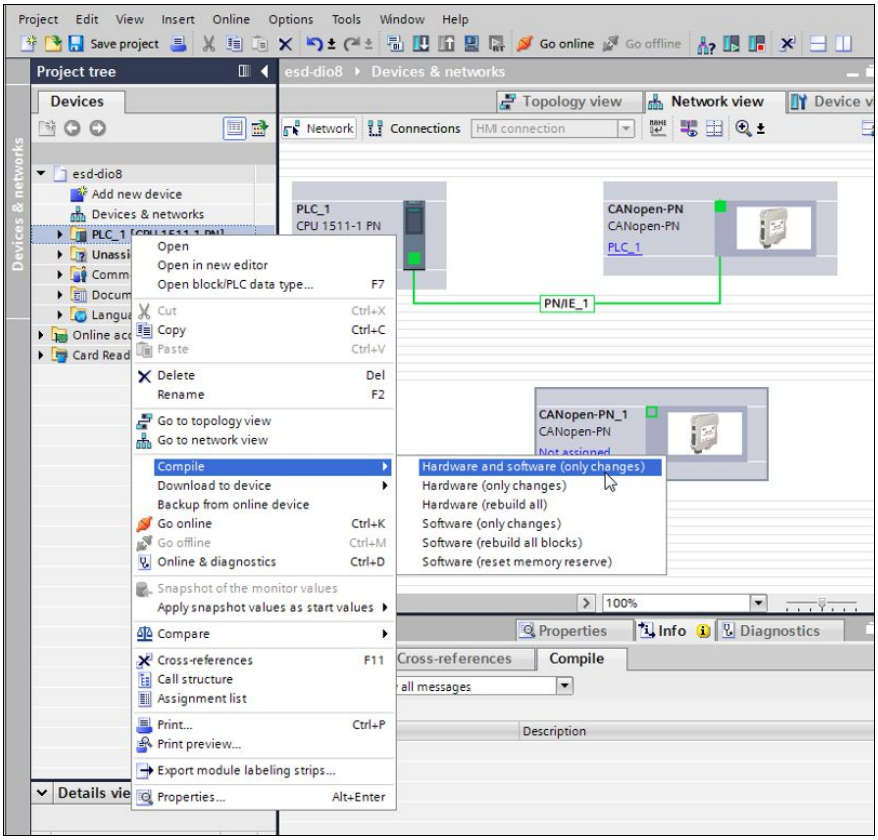


Figure 34: Compile Hardware and Software (detail)

The configuration is compiled. Continue to work offline!
Now the hardware and software can be downloaded in the device as described in the following:
Select your device (*PLC_1 here)* again and further in the pull-down menus the menu items *Download to device* and then *Hardware and software*.

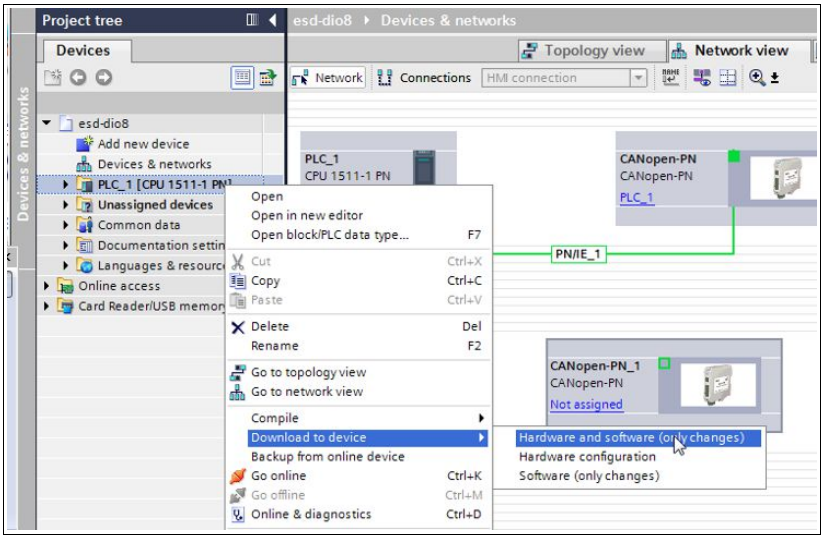


Figure 35: Download hardware and software to device (detail)

Hardware and software are downloaded now.

Click on the button *Go online* in the tool bar under the main menu to go online.

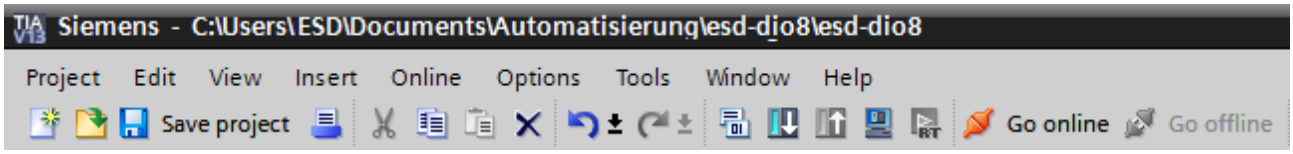


Figure 36: Toolbar with button *Go online*

The online connection is now established.
In the program window the components of the decentralised periphery are marked with check marks.
The I/O addresses of the single units are shown in detail in the *Device overview*.

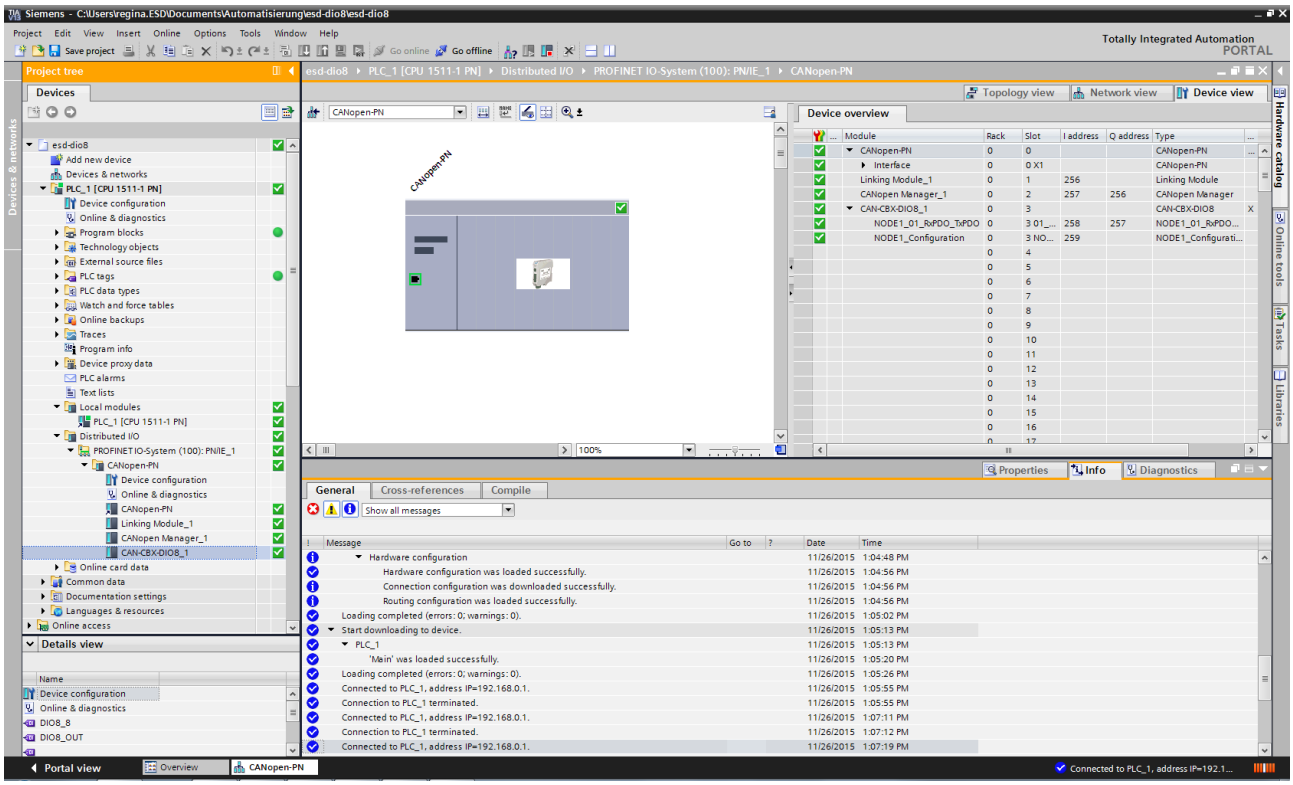


Figure 37: Main window with device overview

6. Control via PROFINET IO

6.1 CANopen Manager Services via PROFINET IO

In accordance with the PROFIBUS International Document TC2-09-0002 (CANopen-Integration_7012_V10_Mar11) [6] which is supported by the CANopen-PN, the services of the CANopen Manager, which are described in the following chapters and correspond to the CANopen specification [5], can be controlled via PROFINET side.
For this purpose the read- and write services of PROFINET IO are used.

S7-Function Block

The S7 function blocks which have to be called for the asynchronous writing and reading of data (e.g. SDO) are described in the following.
In the Simatic-S7 the function block SFB 53 (WRREC) is used for writing of asynchronous data and the function block SFB 52 (RDREC) is used for reading of asynchronous data on PROFINET IO.

Asynchronous **Writing** of data (PROFINET IO Write Service):

```
CALL  SFB 53 , DBx
  REQ    :=BOOL    (Input)
  ID     :=DWORD   (Input)
  INDEX  :=INT     (Input)
  LEN    :=INT     (Input)
  DONE   :=BOOL    (Output)
  BUSY   :=BOOL    (Output)
  ERROR  :=BOOL    (Output)
  STATUS:=DWORD    (Output)
  RECORD:=ANY      (I/O)
```

To the function block SFB 53 (WRREC) a data block (DBx here) has to be added. The data block will be automatically generated when the function block is called.

Parameter	Meaning	
REQ	=1: a write operation is started	
ID	Logical address of the PROFINET IO component (module or submodule). For an output unit bit 15 has to be set (e.g. for address 5: ID:=DW#16#8005). For a combination module (input and output) the lower of both addresses has to be specified (e.g.: transmit SDO to CANopen unit in slot 3, then the corresponding I/O-address of this slot has to be selected)	
INDEX	Index as described in the respective chapters (6.1.1 - 6.1.17).	
LEN	Number of the bytes to be transmitted (e.g. Upload SDO thus 5).	
DONE	= 1:	this write operation is finished
BUSY	= 1:	this write operation is still in process
ERROR	= 1:	an error occurred during the write operation, see parameter STATUS for a detailed error description (error number)
STATUS	Error description	
RECORD	Transmitted data according to the corresponding index tables The length must not be smaller than LEN	

Asynchronous **Reading** of data (PROFINET Read Services):

```
CALL  SFB 52 , DBx
  REQ    :=BOOL     (Input)
  ID     :=DWORD    (Input)
  INDEX  :=INT      (Input)
  MLEN   :=INT      (Input)
  VALID  :=BOOL     (Output)
  BUSY   :=BOOL     (Output)
  ERROR  :=BOOL     (Output)
  STATUS:=DWORD     (Output)
  LEN    :=INT      (Output)
  RECORD:=ANY       (I/O)
```

To the function block SFB52 a data block (DBx here) has to be added. The data block is automatically generated when the function block is called.

Parameter	Meaning	
REQ	=1: a read operation is started	
ID	Logical address of the PROFINET IO component (module or submodule). For an output unit bit 15 has to be set (e.g. for address 5: ID:=DW#16#8005). For a combination module (input and output) the lower of both addresses has to be specified (e.g.: transmit SDO to CANopen unit in slot 3, then the corresponding I/O-address of this slot has to be selected)	
INDEX	Index as described in the respective chapters (6.1.1 - 6.1.17).	
MLEN	minimum length of the bytes to be read e.g. for „Get Version“ 28 bytes The actually received length of the data is returned in LEN	
VALID	= 1:	this read operation is completed successfully
BUSY	= 1:	this read operation is still in process
ERROR	= 1:	an error occurred during the read operation, see parameter STATUS for a detailed error description (error number)
STATUS	Error description	
LEN	Number of the received bytes	
RECORD	Transmitted data according to the corresponding index tables The received length is of these data is specified in LEN	

Note “REQ”:
Please refer to the S7 documentation about this data blocks. In particular, the „REQ“-input is not edge-triggered. If the input is not reset accordingly, the operation will be repeated permanently!

Note “ID”:
In Step 7 5.x usually the input address of the CANopen Manager module has to be used. In the TIA portal (e.g. when using S7-1500) the HW identifier of the CANopen Manager has to be used instead.

6.1.1 Upload SDO (Index 0xB711)

By means of this service the object (SDO), described via index and sub-index, of the selected node with the node number *Node-ID* is read.

First the Write service is used to activate the CANopen-PN to read the data and to store it internally. For reading these data the Read service is used.

The parameters used when writing (Write services) are defined as described in the following (the parameter *Data Type* has to be set according to table 4):

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB711	1	*CANopen Node-ID*	0x01..0x7F	unsigned 8
	2	*Index*	0x0000..0xFFFF	unsigned 16
	3	*Sub-index*	0x00..0xFF	unsigned 8
	4	*Data Type*	0x00..0xFF	unsigned 8

When reading the data are automatically adjusted in the byte order (Endianness) and returned.
I.e., the byte order of the objects with the sizes 2, 4 and 8 bytes will be interchanged from Little-Endian (Intel) format, as used for CANopen, into Big-Endian (Motorola) format, as used for PROFINET IO. All other objects are transmitted unchanged.
The parameter of the following reading (Read Service) are defined as follows:

PROFINET IO Read Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB711	1	*Data*	*	*

*... Value range and data type dependent on the object that is read.

In the event of errors a data response with ErrorCode = 0xDE (IODReadRes) and ErrorDeCode = 0x80 (PNIORW) is returned for the read service. In *ErrorCode1* – consisting of *ErrorClass* and *ErrorCode* - the following values are returned according to the error table "Assignment of SDO abort codes of CANopen" in [6]:

PROFINET IO Side			**CANopen Side**	
Error Class	**Error Code**	**ErrorCode1**	**SDO Abort Code [2]**	**Description of SDO Abort Code**
11	8	0xB8	0x0503 0000	Toggle bit not alternated.
12	3	0xC3	0x0504 0000	SDO protocol timed out.
11	8	0xB8	0x0504 0001	Client/server command specifier not valid or unknown.
11	1	0xB1	0x0504 0002	Invalid block size (block mode only).
11	8	0xB8	0x0504 0003	Invalid sequence number (block mode only).
11	8	0xB8	0x0504 0004	CRC error (block mode only).
12	3	0xC3	0x0504 0005	Out of memory.
11	6	0xB6	0x0601 0000	Unsupported access to an object.

PROFINET IO Side			CANopen Side	
Error Class	**Error Code**	**ErrorCode1**	**SDO Abort Code [2]**	**Description of SDO Abort Code**
11	6	0xB6	0x0601 0001	Attempt to read a write only object.
11	6	0xB6	0x0601 0002	Attempt to write a read only object.
11	0	0xB0	0x0602 0000	Object does not exist in the object dictionary.
11	6	0xB6	0x0604 0041	Object cannot be mapped to the PDO.
11	1	0xB1	0x0604 0042	The number and length of the objects to be mapped would exceed PDO length.
11	8	0xB8	0x0604 0043	General parameter incompatibility reason.
10	8	0xA8	0x0604 0047	General internal incompatibility in the device.
10	2	0xA2	0x0606 0000	Access failed due to an hardware error.
11	1	0xB1	0x0607 0010	Data type does not match, length of service parameter does not match
11	1	0xB1	0x0607 0012	Data type does not match, length of service parameter too high
11	1	0xB1	0x0607 0013	Data type does not match, length of service parameter too low
11	0	0xB0	0x0609 0011	Sub-index does not exist.
11	8	0xB8	0x0609 0030	Invalid value for parameter (download only).
11	3	0xB3	0x0609 0031	Value of parameter written too high (download only).
11	7	0xB7	0x0609 0032	Value of parameter written too low (download only).
11	7	0xB7	0x0609 0036	Maximum value is less than minimum value.
12	3	0xC3	0x060A 0023	Resource not available: SDO connection
12	0	0xC0	0x0800 0000	General error
10	0	0xA0	0x0800 0020	Data cannot be transferred or stored to the application.
11	6	0xB6	0x0800 0021	Data cannot be transferred or stored to the application because of local control.
11	5	0xB5	0x0800 0022	Data cannot be transferred or stored to the application because of the present device state.
11	4	0xB4	0x0800 0023	Object dictionary dynamic generation fails or no object dictionary is present (e.g. object dictionary is generated from file and generation fails because of an file error).
11	0	0xB0	0x0800 0024	No data available

Table 3: SDO abort codes

6.1.2 Download SDO (0xB713)

By means of this service the Object (SDO) is written, which is described via index and sub-indeces of the selected node with the node number *Node-ID*.
The parameters used when writing (Write Service) are defined as follows:

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB713	1	*CANopen Node-ID*	0x01..0x7F	unsigned 8
	2	*Index*	0x0000..0xFFFF	unsigned 16
	3	*Sub-index*	0x00..0xFF	unsigned 8
	4	*Data Type*	0x00..0xFF	unsigned 8
	5	*Data*		depending on *Data Type*

The following data types contained in *Data Type* are supported and the byte order (Endianness) is swapped according to the entry in the table 4 (Swapping = yes) or not (Swapping = no):

Data Type	**Value**	**Swapping**
Boolean	0x01	no
Integer8	0x02	no
Integer16	0x03	yes
Integer32	0x04	yes
Unsigned8	0x05	no
Unsigned16	0x06	yes
Unsigned32	0x07	yes
Floating32	0x08	yes
VisibleString	0x09	no
OctetString	0x10	no
TimeOfDay	0x12	yes
TimeDifference	0x13	yes
Floating64	0x15	yes
TimeOfDay without date indication	0x52	yes
TimeDifference with date indication	0x53	yes
TimeDifference without date indication	0x54	yes
INTEGER64	0x55	yes
UNSIGNED64	0x56	yes

Table 4: Supported data types according to PROFINET IO

For the return in the event of an error, please note Table 3 in chapter 6.1.1.

6.1.3 Configure SDO Timeout (0xB715)

With this service the SDO Timeout is changed.
SDO Timeout is the time the CANopen-PN waits for an answer of the node defined via the module slot after an SDO request
The CANopen-PN will not accept the value until the CANopen Manager is reset (see chapter 6.1.15). Normally this value is specified in the GSDML Composer (see chapter 5.9.4) and does not have to be changed.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB715	1	*SDO Timeout [ms]*	0x0000..0xFFFF	unsigned 16

6.1.4 Start Node (0xB731)

This service can start a single CANopen node (*Node-ID* = node number from 0x01 up to 0x7F), i.e. the node is transferred into CANopen state *Operational*.
If *Node-ID* contains the value '0', all CANopen nodes will be addressed together at once.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB731	1	*CANopen Node-ID*	0x00..0x7F	unsigned 8

6.1.5 Stop Node (0xB732)

This service can stop a single CANopen node *(Node-ID* = node number from 0x01 up to 0x7F), i.e. the node is transferred into CANopen state *Stopped*.
If *Node-ID* contains the value '0', all CANopen nodes will be addressed together at once.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB732	1	*CANopen Node-ID*	0x00..0x7F	unsigned 8

6.1.6 Set Node to Pre-Operational (0xB733)

This service can transfer a single CANopen node (*Node-ID* = node number from 0x01 up to 0x7F), into CANopen state *Pre-Operational*.
If *Node-ID* contains the value '0', all CANopen nodes will be addressed together at once.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB733	1	*CANopen Node-ID*	0x00..0x7F	unsigned 8

6.1.7 Reset Node (0xB734)

This service resets a single CANopen node (*Node-ID* = node number).
(Might be necessary if an error occurred during boot-up of the slave and the CANopen Manager has terminated the boot-up for this reason.)

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB734	1	*CANopen Node-ID*	0x01..0x7F	unsigned 8

6.1.8 Reset Communication (0xB735)

By means of this service only the CANopen state machine of a single CANopen node (*Node-ID* = node number) will be reset.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB735	1	*CANopen Node-ID*	0x01..0x7F	unsigned 8

6.1.9 Initialize Gateway (0xB751)

With this service the bit rate of the CAN bus can be set and subsequently the CANopen Manager in the CANopen-PN will be restarted.

NOTICE
Because the bit rates of the connected CAN nodes do not change automatically, the bit rates of the nodes should be set to the new value first. Please refer to the manual of the connected CANopen devices.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB751	1	*CAN Bit Timing*	0..8	unsigned 32

The bit rate is defined as follows:

CAN Bit Timing	**Bit rate [kbit/s]**
0	1000
1	800
2	500
3	250
4	125
5	100
6	50
7	20
8	10

6.1.10 Set Heartbeat Producer (0xB754)

By means of this service the Heartbeat-Producer time of the CANopen-PN can be set. This service can only be proceeded for the CANopen-PN itself, the parameter *Node-ID* will not be evaluated.
The CANopen-PN will not accept the value (see chapter 6.1.15) until the CANopen Manager is reset.
Normally this value is specified in the GSDML Composer (see chapter 5.7) and does not have to be changed.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB754	1	*CANopen Node-ID*	0x01..0x7F	unsigned 8
	2	*HeartbeatProducerTime*	0x0000..0xFFFF	unsigned 16

6.1.11 Set Node-ID (0xB755)

With this service the node number of the CANopen-PN can be changed.
The CANopen-PN will not accept the value (see chapter 6.1.15) until the CANopen Manager is reset.
Normally this value is specified in the GSDML Composer (see chapter 5.7) and does not have to be changed.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB755	1	*CANopen Node-ID*	0x01..0x7F	unsigned 8

6.1.12 Start Emergency Consumer (0xB756)

This service enables the forwarding of CANopen emergency messages to the PROFINET IO side.
Per default the forwarding is enabled. Therefore this service is only useful if the forwarding has been disabled before (see chapter 6.1.13).
The parameters are not evaluated.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB756	1	*CANopen Node-ID*	0x01..0x7F	unsigned 8
	2	*COB-ID*	0x0081..0x00FF	unsigned 16

6.1.13 Stop Emergency Consumer (0xB757)

This service disables the forwarding of CANopen emergency messages to the PROFINET IO side.
The parameters are not evaluated.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB757	1	*CANopen Node-ID*	0x01..0x7F	unsigned 8
	2	*COB-ID*	0x0081..0x00FF	unsigned 16

6.1.14 Get Version (0xB762)

This service can request information about the CANopen-PN.
Therefore a Write service with a parameter "Dummy" has to be called previously.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB762	1	*Dummy*	-	unsigned 8

The service returns the data with the Read service in the following format:

PROFINET IO Read Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB762	1	*Vendor-ID*	0x0000 0017	unsigned 32
	2	*Product code*	0x2292 1002	unsigned 32
	3	*Revision number* *(CANopen software, not firmware)*	0xXXXX YYZZ (depending on version, Version XXXX.YY.ZZ)	unsigned 32
	4	*Serial number*	[0 .. 0xFFFF FFFF]	unsigned 32
	5	*Gateway class*	0x0000 0003	unsigned 32
	6	*Protocol version*	0x0001 0100	unsigned 32
	7	*Implementation class*	0x0004 0000	unsigned 32

6.1.15 Reset Controller (0xB771)

This service restarts the CANopen-Manager in the CANopen-PN.
The parameter is not evaluated. It should only be used in exceptional cases – because additionally all CANopen slaves are restarted. This will lead to many PROFINET alarms, which might be possibly not completely processed, e.g. in the TIA portal.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB771	1	*CANopen Node-ID*	0x01..0x7F	unsigned 8

6.1.16 Start Controller (0xB772)

This service starts the CANopen Manager in the CANopen-PN. This is only sensible if the CANopen Manager has been stopped before (see 6.1.17)
The parameter is not evaluated.

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB772	1	*CANopen Node-ID*	0x01..0x7F	unsigned 8

6.1.17 Stop Controller (0xB773)

This service stops the CANopen Manager in the CANopen-PN.
The parameter is not evaluated.
Normally this service should not be necessary. – Please note that the termination is proceeded immediately, without generating alarms, etc. before!

PROFINET IO Write Service				
Index	**Parameter**	**Description**	**Value range**	**Data type**
0xB773	1	*CANopen Node-ID*	0x01..0x7F	unsigned 8

6.2 PROFINET IO Diagnosis

S7-Function Blocks

For reading of diagnosis data from a PROFINET IO device the function block SFB 54 (RALRM) is used. It is best to call this function in OB82 (Diagnosis alarm-OB).

The call is as shown in the following:

```
CALL  SFB 54 , DBx
 MODE      := INT      (Input)
 F_ID      := DWORD    (Input)
 MLEN      := INT      (Input)
 NEW       := BOOL     (Output)
 STATUS    := DWORD    (Output)
 ID        := DWORD    (Output)
 LEN       := INT      (Output)
 TINFO     := ANY      (I/O)
 AINFO     := ANY      (I/O)
```

To the function block SFB 54 a data block (DBx here) has to be added. The data block is automatically generated when the function block is called.

Parameter	Meaning
MODE	Operating mode of this SFB 0 – only parameter NEW and LEN are written 1 – all output parameters are rewritten 2 – all output parameters are rewritten, if F_ID has triggered the alarm
F_ID	logical start address of the selected PROFINET IO component (refer to note “ID” in chapter 6.1)
MLEN	maximum length of the bytes to be read in AINFO, the actually received length of the data is specified in LEN
NEW	= 1: a new alarm has been received
STATUS	error description (see online help of S7)
ID	logic start address of the PROFINET IO component, that has received an alarm (The I/O identifier is contained in Bit 15: 0: input address; 1: output address)
LEN	length of the received alarm information
TINFO	administrative information
AINFO	alarm information (see table 5 in chapter 6.2.1)

6.2.1 Emergency Mapping

CANopen Emergency messages are mapped to Diagnosis Alarms of PROFINET IO according to the following table:

Offset	Diagnosis Definition	Emergency Mapping
0	Block type	
2	Block length	
4	Version	
6	Alarm-type	1 = Diagnosis
8	API	always 0x0000 0000
12	Slot	
14	Subslot	
16	Module Ident Number	always defined by the Composer
20	Submodule Ident Number	always defined by the Composer
24	Alarm Specifier	
26	User Structure Identifier	always 0x00CA
28	User Data 1	Error Code; definition see table 6 and [2]
30	User Data 2	Error Register / Object 0x1001; (see table 7 and [2])
31	User Data 3	Manufacturer Specific Emergency Data (5 octets, see manual of the CANopen device)
36	(Length of the data = 36)	--

Table 5: Emergency Mapping

Alarm specifier shall be set to 0 in case of an error code of 0. This means that the CANopen node is error free.

Please read the manual of the CANopen device for a detailed description of the error codes and their meaning.

Error Code	**Description** (from [2])
0x0000	Error reset or no error
0x1000	Generic error
0x20xx	Current
0x21xx	Current, CANopen device input side
0x22xx	Current inside the CANopen device
0x23xx	Current, CANopen device output side
0x30xx	Voltage
0x31xx	Mains voltage
0x32xx	Voltage inside the CANopen device

0x33xx	Output voltage
0x40xx	Temperature
0x41xx	Ambient temperature
0x42xx	Device temperature
0x50xx	CANopen device hardware
0x60xx	CANopen device software
0x61xx	Internal software
0x62xx	User software
0x63xx	Data set
0x70xx	Additional modules
0x80xx	Monitoring
0x81xx	Communication
0x8110	CAN overrun (objects lost)
0x8120	CAN in error passive mode
0x8130	Life guard error or heartbeat error
0x8140	recovered from bus off
0x8150	CAN-ID collision
0x82xx	Protocol error
0x8210	PDO not processed due to length error
0x8220	PDO length exceeded
0x8230	DAM MPDO not processed, destination object not available
0x8240	Unexpected SYNC data length
0x8250	RPDO timeout
0x90xx	External error
0xF0xx	Additional functions
0xFFxx	Device specific

Table 6: CANopen Emergency Codes and Classes

Bit	**Description** (from [2])
0	General error (always set)
1	Current
2	Voltage
3	Temperature
4	Communication
5	Profile-specific
6	Reserved
7	Manufacturer-specific

Table 7: CANopen Error Register

7. Technical Data

7.1 General Technical Data

Power supply voltage	nominal voltage 24 V/DC ±10%, current consumption (24 V, 20 °C): typical: 150 mA
Connectors	24V 24V-power supply voltage (X1, 4-pin COMBICON-plug with spring-cage connection) CAN CAN bus interface (X2, 5-pin Phoenix Contact MC 1,5/5-GF-3,81) PORT1, PORT2 PROFINET IO-interface PORT1, PORT2 (X4, 2x RJ-45-socket with integrated transformer and LEDs) InRailBus CAN bus interface and power supply voltage via InRailBus (X5, 5-pin ME-MAX-TBUS-connector, accessories) The following interface is only for test- and diagnosis DIAG DIAG-interface (X3, USB-socket type B)
Temperature range	0...50 °C ambient temperature
Humidity	max. 90%, non-condensing
Protection class	IP20
Pollution degree	maximum permissible according to DIN EN 61131-2: Pollution Degree 2
Housing	plastic housing for carrier rail mounting NS35/7,5 DIN EN 60715
Dimensions	width: 22.5 mm, height: 99 mm, depth: 114.5 mm (without connectors)
Weight	approx. 135 g

Table 8: General data of the module

7.2 Microcontroller

CPU (PROFINET IO):	
Microcontroller	ERTEC400, 150 MHz
Memory	SDRAM: 32 MB EEPROM 16 Kbit

CPU (Application):	
Microcontroller	SPEAr320S, 320 MHz
Memory	DDR2 RAM: 64 MB SPI-Flash: 8 MB

Table 9: Microcontroller units

7.3 CAN Interface

Number of CAN interfaces	1 x CAN
CAN controller	ISO 11898-1, integrated in SPEAr320S
Electrical isolation	CAN interface isolated by digital isolators and DC/DC converter
Physical CAN Layer	Physical layer according to ISO 11898-2, transmission rate programmable from 10 Kbit/s to 1 Mbit/s
Bus termination	Terminating resistor has to be set externally, if required
Connector	5-pin COMBICON with spring-cage connection or optional via InRailBus connector (CAN-CBX-TBUS)

Table 10: Data of the CAN interface

7.4 PROFINET IO Interface

Number of interfaces	2, PROFINET IO PORT1, PORT2
Connection	RJ-45, BASE-TX, 100 Mbit/s, compatible to IEEE 802.3, electrical isolation via RJ-45-socket with integrated transformer
Controller	ERTEC400
Connector	PORT1, PORT2 (X4), 2x RJ-45 socket with integrated transformer and LEDs

Table 11: Data of PROFINET IO interface

7.5 DIAG, Serial Interface via USB Interface

Type	USB
USB-specification	USB 2.0 High Speed (480 Mbit/s)
Connector	DIAG (X3), USB socket type-B

Table 12: Data of the USB interface

7.6 Software Support

The firmware of the CANopen-PN supports the CANopen® specification CiA 301 [2].

The configuration of the CANopen side of the gateway is done via the GSDML Composer. The GSDML Composer is used for generation and parametrization of a GSD file for the CANopen-PN.
The PROFINET IO side of the CANopen-PN is configured via a so called engineering tool (e.g. SIMATIC-S7).

8. Interfaces and Connector Assignments

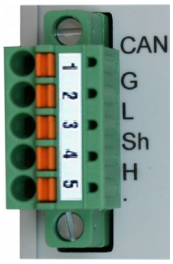
8.1 CAN

The CAN interface can be connected via CAN connector (X2) or optional via InRailBus.
Use the mounting-rail bus connector of the CBX-InRailBus for the connection via the InRailBus, see order information in the appendix (page 105).

Device connector: Phoenix Contact MC 1,5/5-GF-3,81
Line connector: Phoenix Contact FK-MCP 1,5/5-STF-3,81, spring-cage connection
Phoenix Contact order No.: 1851261 (included in delivery)
For conductor connection and conductor cross section see page 77.

Pin Position:

(line connector)



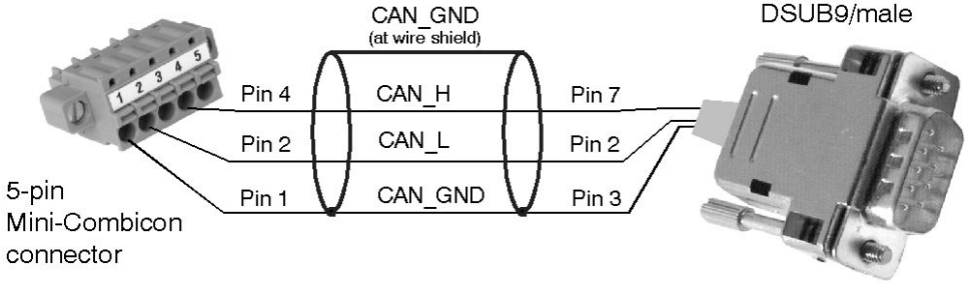
Pin Assignment:

Imprint	Signal	Pin
G	CAN_GND	1
L	CAN_L	2
Sh	Shield	3
H	CAN_H	4
•	-	5

Signal Description:

CAN_L, CAN_H ... CAN signals
CAN_GND ... reference potential of the local CAN physical layer
Shield ... pin for line shield connection (using hat rail mounting direct contact to the mounting rail potential)
- ... not connected

Recommendation of an adapter cable from 5-pin COMBICON (here line connector FK-MCP1,5/5-STF_3,81 with spring-cage-connection) to 9-pin DSUB:

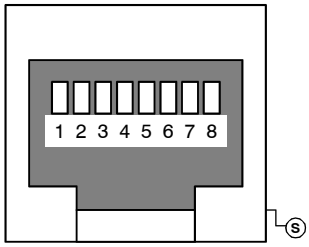


The assignment of the 9-pin DSUB-connector and the 5-pin Mini-COMBICON is designed according to CiA 303 Part 1.

8.2 PROFINET IO

Device connector: RJ45 socket, 8-pin,
Ethernet 100BASE-TX, assigned according to IEEE 802.3-2008,
Table 25-3 “UTP MDI Contact Assignment”

Pin Position:



Pin Assignment:

Pin	Signal	Meaning
1	MDI0+ (TxD+)	Transmit Data +
2	MDI0- (TxD-)	Transmit Data -
3	MDI1+ (RxD+)	Receive Data +
4	-	-
5	-	-
6	MDI1- (RxD-)	Receive Data -
7	-	-
8	-	-

S	Shield	

The pins 1 to 8 are connected to line terminations

Signal Description:

MDI0+/-, MDI1+/- ...	PROFINET IO data lines
- ...	reserved for future applications, do not connect!
Shield...	case shield, connected to earth potential (FE).

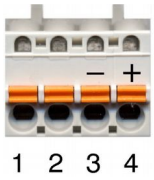
NOTICE
Permissible cable types: Cables of category 5e or higher have to be used to grant the function in networks with up to 100 Mbits/s.
esd grants the EC conformity of the product if the wiring is carried out with shielded twisted pair cables of class SF/UTP or higher.

8.3 24V-Power Supply Voltage

The power supply voltage can be fed via connector 24V (X1) or optional via InRailBus (connector assignment see page 76).

Device connector: Phoenix Contact MSTBO 2,5/4-G1L-KMGY
Line connector: Phoenix Contact FKCT 2,5/4-ST, 5.0 mm pitch, spring-cage connection
Phoenix Contact order no.: 1921900 (included in the scope of delivery)
For conductor connection and conductor cross section see page 77.

Pin Position:



Pin Assignment:

Imprint of the housing			24V	
	.	.	M	P
Imprint of the connector	(none)	(none)	-	+

Pin no.	1	2	3	4
Signal	**Do not connect!**	**Do not connect!**	M24 (GND)	P24 (+ 24 V)

Please refer to the connecting diagram page 13.

NOTICE
It is **not permissible** to feed through the power supply voltage through the 24V connector and to supply the power supply voltage to another CAN module station!
A feed through of the +24 V power supply voltage can cause damage on the modules.

Signal Description:

P24... power supply voltage +24 V ±10 %
M24... reference potential

8.4 24V and CAN via InRailBus

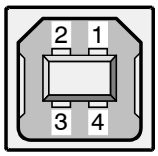
Power supply voltage and CAN can optionally be fed via InRailBus.
Use the mounting-rail bus connector of the CBX-InRailBus for the connection via the InRailBus, see order information in the appendix (page 105).
Take notice of the instructions for connecting power supply and CAN signals via InRailBus in the appendix!

8.5 DIAG

The CANopen-PN realises a virtual network interface. Via this interface firmware can be updated and CAN messages can be recorded and transmitted via the implemented esd EtherCAN functionality, see chapter 5.1.4.2.

NOTICE
The module may only be connected to USB nets with USB interfaces with version 2.0!
Operability can only be guaranteed for these USB interfaces.

Pin Position:



Pin Assignment:

Pin	Signal
1	V_{BUS}
2	D-
3	D+
4	GND
Shell	Shield

USB socket (Type B)

8.6 Conductor Connection/Conductor Cross Sections

The following table contains an extract of the technical data of the line connectors [1].

Interface	**Power Supply 24 V**	**CAN Connector**
Connector type plug component (Range of articles)	FKCT 2,5/..-ST KMGY	FK-MCP 1,5/5-STF-3,81
Connection method	spring-cage connection	spring-cage connection
Stripping length	10 mm	9 mm
Conductor cross section solid min.	0.2 mm²	0.14 mm²
Conductor cross section solid max.	2.5 mm²	1.5 mm²
Conductor cross section stranded min.	0.2 mm²	0.14 mm²
Conductor cross section stranded max.	2.5 mm²	1.5 mm²
Conductor cross section stranded, with ferrule without plastic sleeve min.	0.25 mm²	0.25 mm²
Conductor cross section stranded, with ferrule without plastic sleeve max.	2.5 mm²	1.5 mm²
Conductor cross section stranded, with ferrule with plastic sleeve min.	0.25 mm²	0.25 mm²
Conductor cross section stranded, with ferrule with plastic sleeve max.	2.5 mm²	0.5 mm²
Conductor cross section AWG/kcmil min.	24	26
Conductor cross section AWG/kcmil max	12	16
2 conductors with same cross section, solid min.	not allowed	not allowed
2 conductors with same cross section, solid max.	not allowed	not allowed
2 conductors with same cross section, stranded min.	not allowed	not allowed
2 conductors with same cross section, stranded max.	not allowed	not allowed
2 conductors with same cross section, stranded, ferrules without plastic sleeve, min.	not allowed	not allowed
2 conductors with same cross section, stranded, ferrules without plastic sleeve, max.	not allowed	not allowed
2 conductors with same cross section, stranded, TWIN ferrules with plastic sleeve, min.	0.5 mm²	not allowed
2 conductors with same cross section, stranded, TWIN ferrules with plastic sleeve, max.	1.0 mm²	not allowed
Minimum AWG according to UL/cUL	26	28
Maximum AWG according to UL/cUL	12	16

Table 13: Conductor connection/Conductor cross section

9. Correctly Wiring Electrically Isolated CAN Networks

NOTICE
This chapter applies to CAN networks with bit rates up to 1 Mbit/s.
If you work with higher bit rates, as for example used for CAN FD, the information given in this chapter must be examined for applicability in each individual case.
For further information refer to the CiA® CAN FD guidelines and recommendations (https://www.can-cia.org/).

For the CAN wiring all applicable rules and regulations (EU, DIN), e.g. regarding electromagnetic compatibility, security distances, cable cross-section or material, have to be observed.

9.1 CAN Wiring Standards

The flexibility in CAN network design is a major strength of the various extensions based on the original CAN standard ISO 11898-2, such as CANopen®, ARINC825, DeviceNet® and NMEA2000. However, taking advantage of this flexibility absolutely requires a network design that considers the interactions of all network parameters.

In some cases, the CAN organizations have adapted the scope of CAN in their specifications to enable applications outside the ISO 11898 standard. They have imposed system-level restrictions on data rate, line length and parasitic bus loads.

However, when designing CAN networks, a margin must always be planned for signal losses over the entire system and cabling, parasitic loads, network imbalances, potential differences against earth potential, and signal integrities. **Therefore, the maximum achievable number of nodes, bus lengths and stub lengths may differ from the theoretically possible number!**

esd has limited its recommendations for CAN wiring to the specifications of ISO 11898-2.
A description of the special features of the derived specifications CANopen, ARINC825, DeviceNet, and NMEA2000 is omitted here

The consistent compliance with ISO 11898-2 standard offers significant advantages:

- Reliable operation due to well proven design specifications
- Minimization of error sources due to sufficient distances to physical limits
- Easy maintenance because there are no "special cases" to consider for future network modifications and troubleshooting

Of course reliable networks can be designed according to the specifications of CANopen, ARINC825, DeviceNet and NMEA2000, **however it must be observed that it is strictly not recommended to mix the wiring guidelines of the various specifications!**

9.2 Light Industrial Environment (*Single* Twisted Pair Cable)

9.2.1 General Rules

NOTICE
esd grants the EU Conformity of the product, if the CAN wiring is carried out with at least single shielded **single** twisted pair cables that match the requirements of ISO 11898-2. Single shielded *double* twisted pair cable wiring as described in chapter 9.3. ensures the EU Conformity as well.

The following **general rules** for CAN wiring with single shielded *single* twisted pair cable should be followed:

1	A suitable cable type with a wave impedance of about 120 Ω ±10% with an adequate conductor cross-section (≥ 0.22 mm²) must be used. The voltage drop over the wire must be considered.
2	For light industrial environment use at least a two-wire CAN cable, the wires of which must be assigned as follows: • Two twisted wires must be assigned to the data signals (CAN_H, CAN_L). • The cable shield must be connected to the reference potential (CAN_GND).
3	The reference potential CAN_GND must be connected to the functional earth (FE) at exactly **one** point.
4	A CAN bus line must not branch (exception: short cable stubs) and must be terminated with the characteristic impedance of the line (generally 120 Ω ±10%) at both ends (between the signals CAN_L and CAN_H and **not** at CAN_GND).
5	Keep cable stubs as short as possible (l < 0.3 m).
6	Select a working combination of bit rate and cable length.
7	Keep away cables from disturbing sources. If this cannot be avoided, double shielded wires are recommended.

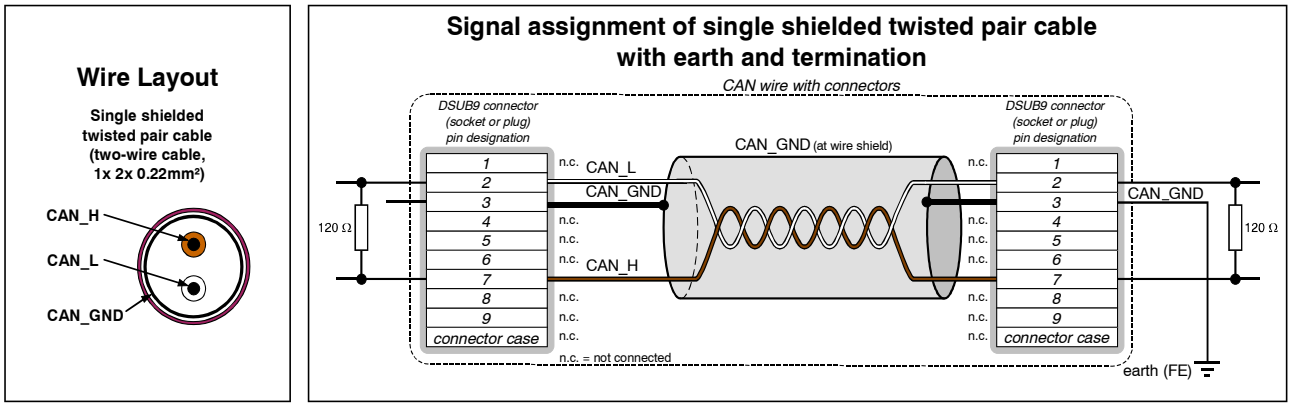


Figure 38: CAN wiring for light industrial environment

9.2.2 Cabling

- To connect CAN devices with just one CAN connector per net use a short stub (< 0.3 m) and a T-connector (available as accessory). If this devices are located at the end of the CAN network, the CAN terminator “CAN-Termination-DSUB9” can be used.

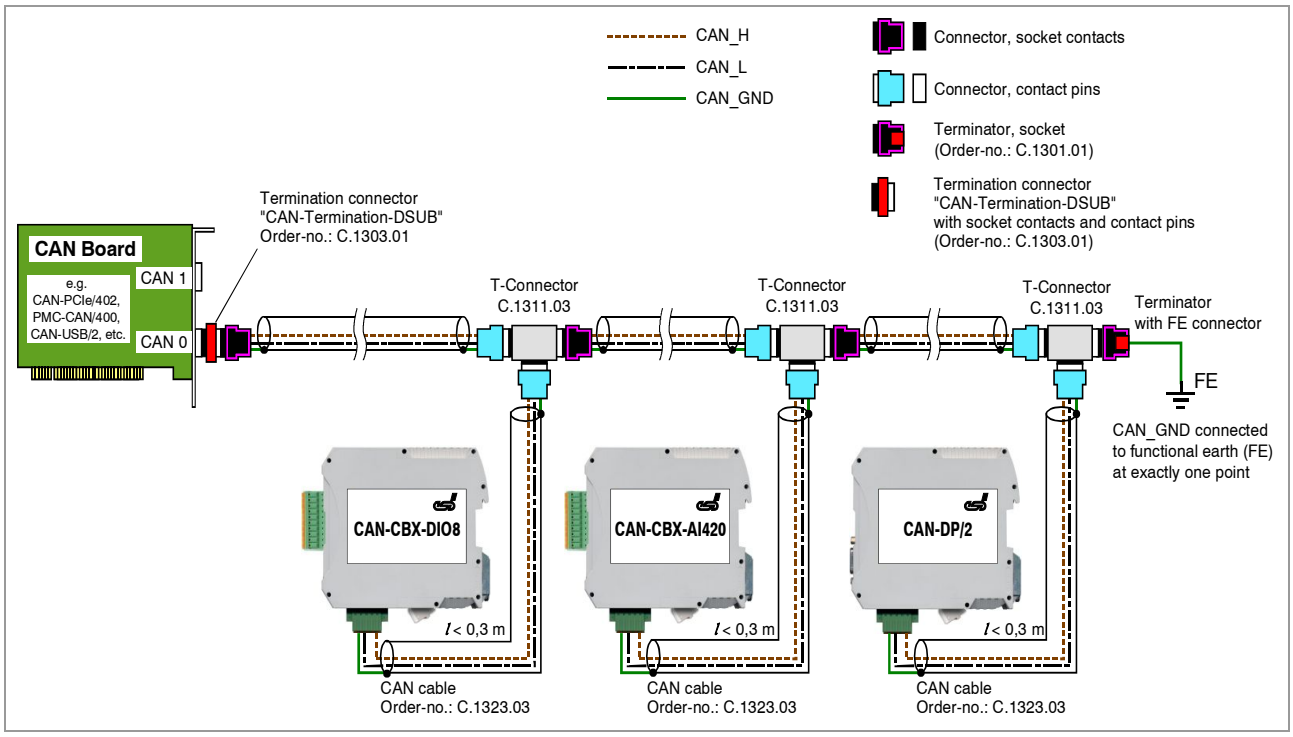


Figure 39: Example for proper wiring with single shielded single twisted pair wires

9.2.3 Branching

- In principle the CAN bus must be realized in a line. The nodes are connected to the main CAN bus line via short cable stubs. This is normally realised by so called T-connectors. esd offers the CAN-T-Connector (Order No.: C.1311.03)
- If a mixed application of single twisted and double twisted cables cannot bei avoided, ensure that the CAN_GND line is not interrupted!
- Deviations from the bus structure can be realized by the usage of repeaters.

9.2.4 Termination

- A termination resistor must be connected at both ends of the CAN bus. If an integrated CAN termination resistor is connected to the CAN interface at the end of the CAN bus, this integrated termination must be used instead of an external CAN termination resistor.
- 9-pole DSUB-termination connectors with integrated termination resistor and pin contacts and socket contacts are available from esd (order no. C.1303.01).
- For termination of the CAN bus and grounding of the CAN_GND, DSUB terminators with pin contacts (order no. C.1302.01) or socket contacts (order no. C.1301.01) and with additional functional earth contact are available.

9.3 Heavy Industrial Environment (*Double* Twisted Pair Cable)

9.3.1 General Rules

The following **general rules** for the CAN wiring with single shielded *double* twisted pair cable should be followed:

1	A suitable cable type with a wave impedance of about 120 Ω ±10% with an adequate conductor cross-section (≥ 0.22 mm²) has to be used. The voltage drop over the wire must be considered.
2	For heavy industrial environment use a four-wire CAN cable, the wires of which must be assigned as follows: • Two twisted wires must be assigned to the data signals (CAN_H, CAN_L). • The other two twisted wires must be assigned to the reference potential (CAN_GND). • The cable shield must be connected to functional earth (FE) at least at one point.
3	The reference potential CAN_GND must be connected to the functional earth (FE) at exactly **one** point.
4	A CAN bus line must not branch (exception: short cable stubs) and must be terminated with the characteristic impedance of the line (generally 120 Ω ±10%) at both ends (between the signals CAN_L and CAN_H and **not** to CAN_GND).
5	Keep cable stubs as short as possible (l < 0.3 m).
6	Select a working combination of bit rate and cable length.
7	Keep away CAN cables from disturbing sources. If this cannot be avoided, double shielded cables are recommended.

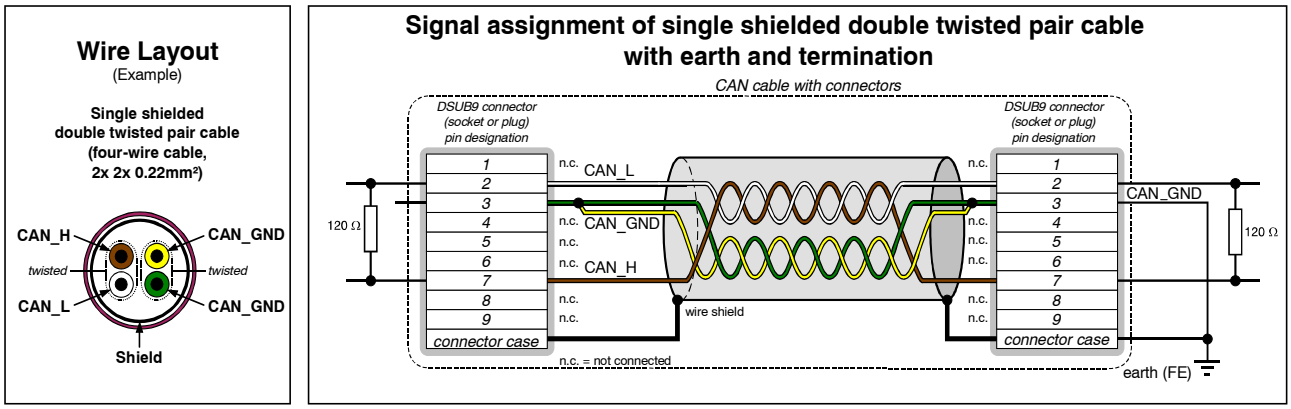


Figure 40: CAN wiring for heavy industrial environment

9.3.2 Device Cabling

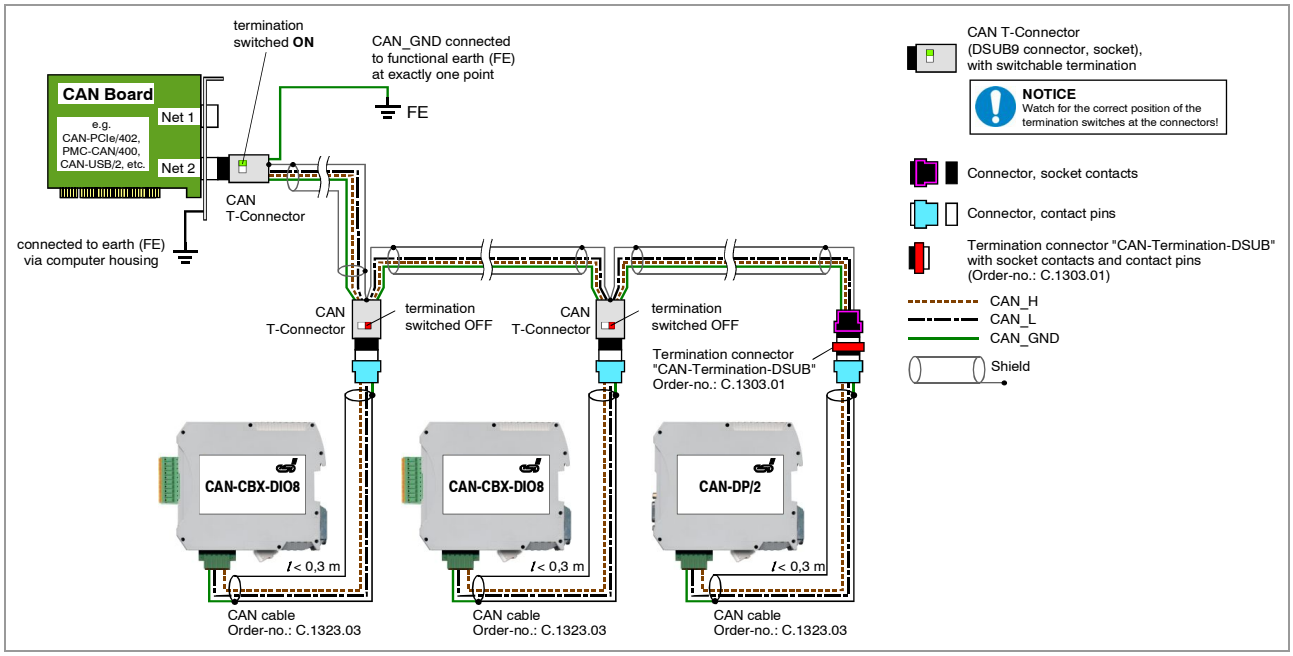


Figure 41: Example of proper wiring with single shielded double twisted pair cables

9.3.3 Branching

- In principle the CAN bus must be realized in a line. The nodes are connected to the main CAN bus line via short cable stubs. This is usually realised by so called T-connectors. When using esd's CAN-T-Connector (order no.: C.1311.03) in heavy industrial environment and with four-wire twisted cables, it must be noted that the shield potential of the conductive DSUB housing is not looped through this type of T-Connector.This interrupts the shielding. Therefore, you must take appropriate measures to connect the shield potentials, as described in the manual of the CAN-T-Connector. For further information on this, please refer to the CAN-T-Connector Manual (order no.: C.1311.21).
  Alternatively, a T-connector can be used, in which the shield potential is looped through, for example the DSUB9 connector from ERNI (ERBIC CAN BUS MAX, order no.:154039).
- If a mixed application of single twisted and double twisted cables cannot be avoided, ensure that the CAN_GND line is not interrupted!
- Deviations from the bus structure can be realized by using repeaters.

9.3.4 Termination Resistor

- A termination resistor must be connected at both ends of the CAN bus. If an integrated CAN termination resistor is connected to the CAN interface at the end of the CAN bus, this integrated termination must be used instead of an external CAN termination resistor.
- 9-pole DSUB-termination connectors with integrated termination resistor and pin contacts and socket contacts are available from esd (order no. C.1303.01).
- 9-pole DSUB-connectors with integrated switchable termination resistor can be ordered for example from ERNI (ERBIC CAN BUS MAX, socket contacts, order no.:154039).

9.4 Electrical Grounding

- For CAN devices with electrical isolation the CAN_GND must be connected between the CAN devices.
- CAN_GND should be connected to the earth potential (FE) at **exactly one** point of the network.
- Each *CAN interface with electrical connection to earth potential* acts as a grounding point. For this reason it is recommended not to connect more than one *CAN device with electrical connection to earth potential.*
- Grounding can be done for example at a termination connector (e.g. order no. C.1302.01 or C.1301.01).

9.5 Bus Length

The bus length of a CAN network must be adapted to the set bit rate. The maximum values result from the fact that the time required for a bit to be transmitted in the bus system is shorter the higher the transmission rate is. However, as the line length increases, so does the time it takes for a bit to reach the other end of the bus. It should be noted that the signal is not only transmitted, but the receiver must also respond to the transmitter within a certain time. The transmitter, in turn, must detect any change in bus level from the receiver(s). Delay times on the line, the transceiver, the controller, oscillator tolerances and the set sampling time must be considered.
In the following table you will find guide values for the achievable bus lengths at certain bit rates.

Bit-Rate [kbit/s]	Theoretical values of reachable wire length **with esd interface** l_{max} [m]	**CiA recommendations** (07/95) for reachable wire lengths l_{min} [m]	Standard values of the cross-section according to CiA 303-1 [mm²]
1000	37	25	0,25 to 0,34
800	59	50	0,34 to 0,6
666,6̄	80	-	
500	130	100	
333,3̄	180	-	
250	270	250	
166	420	-	0,5 to 0,6
125	570	500	
100	710	650	0,75 to 0,8
83,3̄	850	-	
66,6̄	1000	-	
50	1400	1000	
33,3̄	2000	-	not defined in CiA 303-1
20	3600	2500	
12,5	5400	-	
10	7300	5000	

Table 14: Recommended cable lengths at typical bit rates (with esd-CAN interfaces)

Optical couplers are delaying the CAN signals. esd modules typically achieve a wire length of 37 m at 1 Mbit/s within a proper terminated CAN network without impedance disturbances, such as those caused by cable stubs > 0.3 m.

NOTICE
Please note that the cables, connectors and termination resistors used in CANopen networks shall meet the requirements defined in ISO11898-2.
In addition, further recommendations of the CiA, like standard values of the cross section, depending on the cable length, are described in the CiA recommendation CiA 303-1 (see CiA 303 CANopen Recommendation - Part 1: „Cabling and connector pin assignment“, Version 1.9.0, Table 2). Recommendations for pin-assignment of the connectors are described in CiA 106: “Connector pin-assignment recommendations ”.

9.6 Examples for CAN Cables

esd recommends the following two-wire and four-wire cable types for CAN network design. These cable types are used by esd for ready-made CAN cables, too.

9.6.1 Cable for Light Industrial Environment Applications (Two-Wire)

Manufacturer	Cable Type
U.I. LAPP GmbH Schulze-Delitzsch-Straße 25 70565 Stuttgart Germany www.lappkabel.com	e.g. UNITRONIC ®-BUS CAN UL/CSA (1x 2x 0.22) (UL/CSA approved) Part No.: 2170260 UNITRONIC ®-BUS-FD P CAN UL/CSA (1x 2x 0.25) (UL/CSA approved) Part No.: 2170272
ConCab GmbH Äußerer Eichwald 74535 Mainhardt Germany www.concab.de	e. g. BUS-PVC-C (1x 2x 0.22 mm²) Order No.: 93 022 016 (UL appr.) BUS-Schleppflex-PUR-C (1x 2x 0.25 mm²) Order No.: 94 025 016 (UL appr.)

9.6.2 Cable for Heavy Industrial Environment Applications (Four-Wire)

Manufacturer	Cable Type
U.I. LAPP GmbH Schulze-Delitzsch-Straße 25 70565 Stuttgart Germany www.lappkabel.com	e.g. UNITRONIC ®-BUS CAN UL/CSA (2x 2x 0.22) (UL/CSA approved) Part No.: 2170261 UNITRONIC ®-BUS-FD P CAN UL/CSA (2x 2x 0.25) (UL/CSA approved) Part No.: 2170273
ConCab GmbH Äußerer Eichwald 74535 Mainhardt Germany www.concab.de	e. g. BUS-PVC-C (2x 2x 0.22 mm²) Order No.: 93 022 026 (UL appr.) BUS-Schleppflex-PUR-C (2x 2x 0.25 mm²) Order No.: 94 025 026 (UL appr.)

INFORMATION
Ready-made CAN cables with standard or custom length can be ordered from **esd**.

10. CAN Troubleshooting Guide

The CAN Troubleshooting Guide is a guide to finding and eliminating the most common problems and errors when setting up CAN bus networks and CAN-based systems.

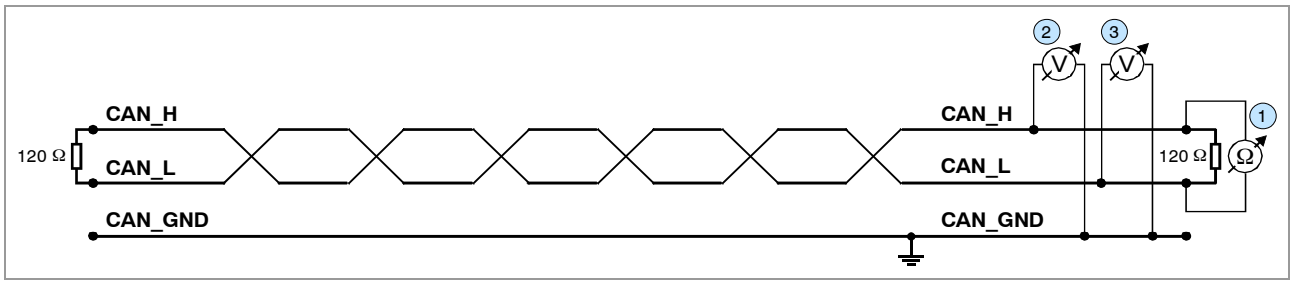


Figure 42: Simplified diagram of a CAN network

10.1 Termination

The bus termination is used to match impedance of a node to the impedance of the bus line used. If the impedance is mismatched, the transmitted signal is not completely absorbed by the load and will be partially reflected back into the transmission line.
If the impedances of the sources, transmission lines and loads are equal, the reflections are avoided. This test measures the total resistance of the two CAN data lines and the connected terminating resistors.

To test this, please proceed as follows:

1. Switch off all supply voltages of all connected CAN nodes.
2. Measure the DC resistance between CAN_H and CAN_L at one end of the network, measuring point ① (see figure above).

Expected result:
The measured value should be between 50 Ω and 70 Ω.

Possible causes of error:
▪ If the value is below 50 Ω, please make sure that:
- There is no **short circuit** between CAN_H and CAN_L wiring.
- **No more than two** terminating resistors are connected.
- The transceivers of the individual nodes are not defective.

▪ If the determined value is higher than 70 Ω, please make sure that:
- All CAN_H and CAN_L lines are correctly connected.
- Two terminating resistors of 120 Ω each are connected to your CAN network (one at each end).

10.2 Electrical Grounding

The CAN_GND of the CAN network should be connected to the functional earth potential (FE) at only **one** point. This test indicates whether the CAN_GND is grounded at one or more points.

Please note that this test can only be performed with electrically isolated CAN nodes.

To test this, please proceed as follows:

1. Disconnect the CAN_GND from the earth potential (FE).
2. Measure the DC resistance between CAN_GND and earth potential (see figure on the right).

Do not forget to reconnect CAN_GND to earth potential after the test!

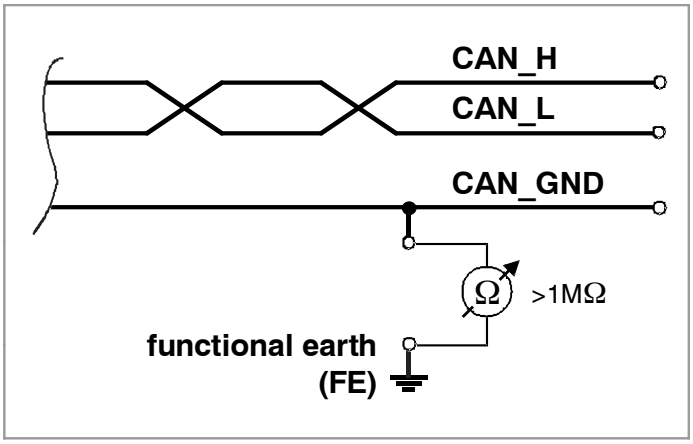


Figure 43: Simplified schematic diagram of ground test measurement

Expected result:
The measured resistance should be higher than 1 MΩ. If it is lower, please search for additional grounding of the CAN_GND wires.

10.3 Short Circuit in CAN Wiring

A CAN bus might possibly still be able to transmit data even if CAN_GND and CAN_L are short-circuited. However, this will usually cause the error rate to rise sharply.
Ensure that there is no short circuit between CAN_GND and CAN_L!

10.4 Correct Voltage Levels on CAN_H and CAN_L

Each node contains a CAN transceiver that outputs differential signals. When the network communication is idle the CAN_H and CAN_L voltages are approximately 2.5 V measured to CAN_GND. Defective transceivers can cause the idle voltages to vary and disrupt network communication.

To test for defective transceivers, please proceed as follows:

1. Switch on all supply voltages.
2. Terminate all network communication.
3. Measure the DC voltage between CAN_H and CAN_GND, measuring point ② (see “Simplified diagram of CAN network” on previous page).
4. Measure the DC voltage between CAN_L and CAN_GND, measuring point ③ (see “Simplified diagram of CAN network” on previous page).

Expected result:
The measured voltage should be between 2.0 V and 3.0 V.

Possible causes of error:
- If it is lower than 2.0 V or higher than 3.0 V, it is possible that one or more nodes have defective transceivers.
  - If the voltage is lower than 2.0 V, please check the connections of the CAN_H and CAN_L lines.

- To find a node with a defective transceiver within a network, please check individually the resistances of the CAN transceivers of the nodes (see next section).

10.5 CAN Transceiver Resistance Test

CAN transceivers have circuits that control CAN_H and CAN_L. Experience shows that electrical damage of the circuits may increase the leakage current in these circuits.

To measure the current leakage through the CAN circuits, please use an ohmmeter and proceed as follows:

1. Switch **off** the node (4) and **disconnect** it from the network (see figure below).

2. Measure the DC resistance between CAN_H and CAN_GND, measuring point (5) (see figure below).

3. Measure the DC resistance between CAN_L and CAN_GND, measuring point (6) (see figure below).

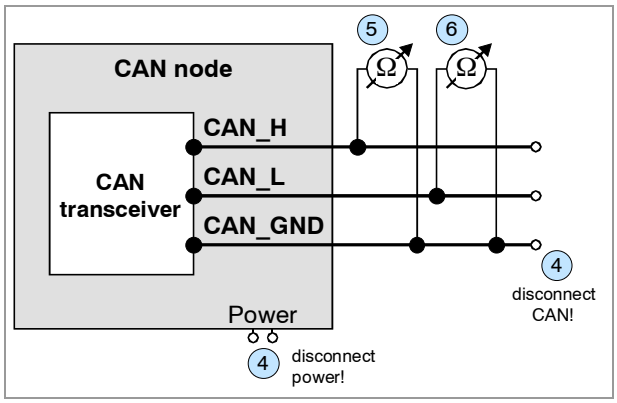


Figure 44: Measuring the internal resistance of CAN transceivers

Expected result:
The measured resistance should be greater than 10 kΩ for each measurement.

Possible causes of error:
- If the resistance is significantly lower, the CAN transceiver may be defective.
- Another indication of a defective CAN transceiver is a very high deviation of the two measured input resistances (>> 200 %).

10.6 Support by esd

If you have followed the troubleshooting steps in this troubleshooting guide and still cannot find a solution to your problem, our support team can help.
Please contact our support by email to **support@esd.eu** or by phone **+49-511-37298-130**.

11. CANopen Firmware

Apart from basic descriptions of CANopen, this chapter contains the most significant information about the implemented functions.
A complete CANopen description is too extensive for the purpose of this manual. Further information can therefore be taken from the CANopen specification CiA 301 [2].

11.1 Definition of Terms

COB ...	Communication Object
Emergency-Id...	Emergency Data Object
n.a.	not applicable
NMT...	Network Management (Master)
Rx...	receive
SDO...	Service Data Object
SYNC...	SYNC (frame) Telegram
tbd.	to be defined
Tx...	transmit

PDOs (Process Data Objects)
PDOs are used to transmit up to 8 bytes process data.
In the ‘Receive’-PDO (RPDO) the process data are received.
In the ‘Transmit’-PDO (TPDO) the process data are transmitted.

SDOs (Service Data Objects)
SDOs are used to transmit module internal configuration- and parameter data. The object directory is accessed via SDOs.
In opposition to the PDOs SDO-messages are confirmed. A write or read request on a data object is always answered by a response telegram with an error index.

NMT State Machine
For the control of the device functionality all CANopen devices come with an internal state machine. In the individual states only defined operations are allowed. The state transitions can be triggered by internal events (e.g. boot-up, error, reset) or by the NMT master.

11.2 NMT-Boot-up

After power on each CANopen devices will enter an initialization phase and will switch automatically into the state *Pre-Operational* then.
Usually a telegram is sufficient to switch from *Pre-Operational* status to *Operational* status after boot-up. For this for example the 2-byte telegram ‘01h’, ‘00h’ has to be transmitted with CAN identifier ‘0000h’ (= Start Remote Node all Devices).

11.3 The CANopen Object Directory

The object directory is basically a (sorted) group of objects which can be accessed via the CAN network. Each object in this directory is addressed with a 16-bit index. The index in the object directories is represented in hexadecimal format.

The index can address parameters defined in CANopen specification or a manufacturer-specific code. By means of the MSBs of the index the object class of the parameter is defined.

Part of the object directory are among others:

Index	Object	Example
0x0001 ... 0x009F	definition of data types	
0x1000 ... 0x1FFF	Communication Profile Area	0x1001 = Error register
0x2000 ... 0x5FFF	Manufacturer Specific Profile Area	-
0x6000 ... 0x9FFF	Standardized Device Profile	according to user profile DS-40x
0xA000 ... 0xFFFF	reserved	-

The kind and number of the supported objects depends on the CANopen module type.

11.4 Accessing the Communication Parameters via SDO Telegrams

The SDOs (Service Data Objects) are used to access the object directory of a device.
They are used for initialization of a device and for parameter transmission.
SDO-access is only possible if the device is in *operational* or *pre-operational* state.

The SDOs are transmitted with ID '**0x600 + NodeID**' (request). The receiver acknowledges the parameters with ID '**0x580 + NodeID**' (response).

An SDO is structured as follows:

Identifier	Command code	Index		Sub-index	LSB	Data field	MSB
		(low)	(high)				

Example:

0x600+ NodeID	0x23	0x00	0x14	0x01	0x7F	0x04	0x00	0x00
	(write)	(Index=0x1400) (Receive-PDO-Comm-Para)		(COB-def.)	Data (here COB-ID) = 0x047F			

Description of the SDOs:

Identifier ... The parameters are transmitted with ID '0x600 + NodeID' (request).
The receiver acknowledges the parameters with ID '0x580 + NodeID' (response).

Command code … The command code transmitted consists among other things of the Command Specifier and the length.

Frequently required combinations are, for instance:

0x40 = 64 : Read Request, i.e. a parameter is to be read

0x23 = 35 : Write Request with 32-bit data, i.e. a parameter is to be set

The addressed module responds to every received telegram with a response telegram. This can contain the following command codes:

0x43 = 67 : Read Response with 32 bit data, this telegram contains the parameter requested

0x60 = 96 : Write Response, i.e. a parameter has been set successfully

0x80 = 128 : Error Response, i.e. the CAN-module reports a communication error

Frequently Used Command Codes

The following table summarizes frequently used command codes. The command frames must always contain 8 data bytes. Notes on the syntax and further command codes can be found in CiA 301 [2], chapter “Service Data Object”.

Command	Number of data bytes	Command code
Write Request (Initiate Domain Download)	1 2 3 4	0x2F 0x2B 0x27 0x23
Write Response (Initiate Domain Download)	-	0x60
Read Request (Initiate Domain Upload)	-	0x40
Read Response (Initiate Domain Upload)	1 2 3 4	0x4F 0x4B 0x47 0x43
Error Response (Abort Domain Transfer)	-	0x80

Index, Sub-Index ... Index and sub-index address the parameters in the object directory.

Data field ... The data field has got a size of a maximum of 4 bytes and is always structured ‘LSB first, MSB last’.

The least significant byte is always in ‘Data 1’. With 16-bit values the most significant byte (bits 8...15) is always in ‘Data 2’, and with 32-bit values the MSB (bits 24...31) is always in ‘Data 4’.

Error codes of the SDO Domain Transfer

The following error codes might occur (according to CiA 301 [2], chapter “Abort SDO Transfer Protocol”):

Error code	Name	Description
0x05040001	SDO_CS_UNKNOWN	wrong command specifier
0x06010000	SDO_WRONG_ACCESS	wrong access
0x06010001	SDO_WRITE_ONLY	wrong read access
0x06010002	SDO_READ_ONLY	wrong write access
0x06020000	SDO_WRONG_INDEX	wrong index
0x06040043	SDO_PARA_INCOMPATIBLE	parameter address incompatible
0x06070010	SDO_WRONG_LENGTH	wrong number of data bytes
0x06070012	SDO_PARA_TO_LONG	service parameter is to long
0x06070013	SDO_PARA_TO_SHORT	service parameter is to short
0x06090011	SDO_WRONG_SUBIND	wrong sub-index
0x06090030	SDO_VALUE_EXCEEDED	transmitted parameter is outside the accepted value range
0x06090031	SDO_VALUE_TOO_HIGH	transmitted parameter exceeds the accepted value range
0x06090032	SDO_VALUE_TOO_LOW	transmitted parameter is below the accepted value range
0x08000000	SDO_OTHER_ERROR	undefined cause of error
0x08000021	SDO_LOCAL_CONTROL	request can not be executed because of the operating state

11.5 Accessing the Process Data via PDOs

For transmission of the process data of a CANopen device during operation PDOs (Process Data Objects) are used.

Four standard PDOs per data direction are available. The communication parameters of these RPDOs and TPDOs can be programmed via SDOs. For example the COB-ID and the transmission type (e.g. sync/async) can be programmable communication parameters.
The objects of the object directory which contain the relevant parameters are assigned to the PDOs via the communication parameters. The CANopen specification CiA 301 [2] provides default assignments of the PDOs for some typical applications.

It depends on the individual CANopen firmware of the CANopen devices how many and which PDOs are supported and which communication parameters can be changed.

11.6 Important CANopen Telegrams

The following table shows a short listing of important common CANopen telegrams.

CAN Identifier	Name	Length	Data	Description
0	NMT	2	0x01 xx	Start (*Preoperational -> Operational*)
0	NMT	2	0x80 xx	*Operational -> Preoperational*
0	NMT	2	0x81 xx	Reset
0	NMT	2	0x82 xx	Reset Communication
0x80	SYNC	0	-	Sync at all
0x80 + *Node-ID*	EMCY	0...8 Bytes	Error code	Emergency Message

Node-ID. ... Node-ID of the addressed CANopen module

xx... Node-ID of a CANopen module or ‘00’ for a message to all CANopen nodes

11.7 Implemented CANopen Objects of CANopen-PN

11.7.1 CiA Specification CiA 301 Objects

The following table shows the implemented CANopen objects according to CiA 301.
For a detailed description of the objects refer to CiA 301 [2].

Index	Sub-index	Description	Data type	Access	Product-specific Properties
0x1000	-	Device Type	unsigned32	ro	default: 0x0000 0000
0x1001	-	Error Register	unsigned8	ro	default: 0x00
0x1002	-	Manufacturer Status Register	unsigned32	ro	default: 0x00
0x1003	0	Pre-defined Error Field	unsigned8	rw	default: 0x00
	1 ... 254		unsigned32	ro	default: 0x00
0x1005	-	COB-ID-Sync	unsigned32	rw	default: 0x80
0x1006	-	Communication Cycle Period	unsigned32	rw	defined via composer
0x1008	-	Manufacturer Device Name	visible string	ro	CANopen-PN
0x1009	-	Manufacturer Hardware Version	visible string	ro	x.yy (depending on version)
0x100A	-	Manufacturer Software Version	visible string	ro	x.yy (depending on version)
0x100C	-	Guard Time	unsigned16	rw	defined via composer
0x100D	-	Life Time Factor	unsigned8	rw	defined via composer
0x1014	-	COB-ID Emergency Object	unsigned32	rw	0x80 + node number
0x1015	-	Inhibit Time Emergency	unsigned16	rw	0
0x1016	0...127	Consumer Heartbeat Time	array	rw	defined via composer
0x1017	-	Producer Heartbeat Time	unsigned16	rw	defined via composer
0x1018	0	Identity Object	unsigned8	ro	Number of Entries = 4
	1		unsigned32	ro	Vendor Id = 0000 0017_h
	2		unsigned32	ro	Device Id = 2292 1002_h
	3		unsigned32	ro	Revision = depending on version
	4		unsigned32	ro	Serial Number

Index	Name	Description
0x1400 ...0x15FF	RPDO Communication Parameter	RPDO Communication Parameters of CANopen-PN according to the number of PDOs of the TPDOs of the slaves connected, as defined in the composer
0x1600 ..0x17FF	RPDO Mapping Parameter	RPDO Mapping Parameters of CANopen-PN according to the number of PDOs of the TPDOs of the slaves connected, as defined in the composer
0x1800 ..0x19FF	TPDO Communication Parameter	TPDO Communication Parameter of CANopen-PN according to the number of PDOs of the RPDOs of the slaves connected, as defined in the composer
0x1A00 ..0x1BFF	TPDO Mapping Parameter	TPDO Mapping Parameter des CANopen-PN according to the number of PDOs of the RPDOs of the slaves connected, as defined in the composer

Table 15: Implemented CANopen objects of the CANopen-PN gateway

11.7.2 Objects of CiA Specification CiA 302-2

CANopen objects according to CiA 302-2 [3] are implemented.

NOTICE
It is strongly recommended that changes of the objects of CiA 302-2 are carried out by experienced users with detailed knowledge of the CANopen specification.

For further information please contact our support team (support@esd.eu).

The following table shows the implemented CANopen objects according to CiA 302-2.
For a detailed description of the objects refer to CiA 302-2 [3].

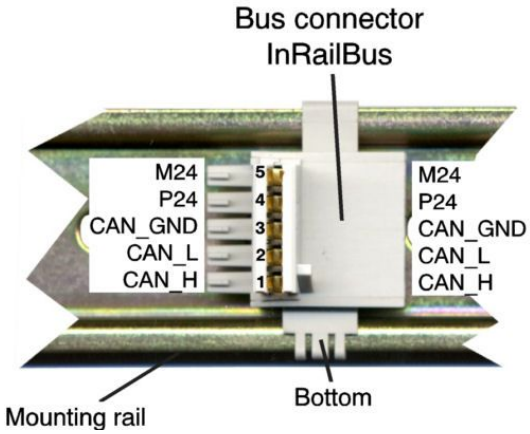
Index	Object code	Description	Data type	Access
0x102A	VAR	NMT inhibit time	unsigned16	rw
0x1F80	VAR	NMT startup	unsigned32	rw
0x1F81	ARRAY	NMT slave assignment	unsigned32	rw
0x1F82	ARRAY	Request NMT	unsigned8	-
0x1F83	ARRAY	Request guarding	unsigned8	-
0x1F84	ARRAY	Device type identification	unsigned32	rw
0x1F85	ARRAY	Vendor identification	unsigned32	rw
0x1F86	ARRAY	Product code	unsigned32	rw
0x1F87	ARRAY	Revision number	unsigned32	rw
0x1F88	ARRAY	Serial number	unsigned32	rw
0x1F89	VAR	Boot time	unsigned32	rw

12. Appendix InRailBus (Option)

12.1 Connector Assignment

Connector type: Mounting-rail bus connector of the CBX-InRailBus
Phoenix Contact ME 22,5 TBUS 1,5/5-ST-3,81 KMGY

Connector View:



Pin Assignment:

Pin	Signal
5	M24 (GND)
4	P24 (+24 V)
3	CAN_GND
2	CAN_L
1	CAN_H
S	FE (PE_GND)

Signal Description:

CAN_L,
CAN_H ... CAN signals
CAN_GND ... reference potential of the local CAN-Physical layers
P24... power supply voltage +24 V ±10 %
M24... reference potential
FE... functional earth contact (EMC) (connected to mounting rail potential)

12.2 Using InRailBus (Option)

INFORMATION
This chapter describes the installation of the module using InRailBus for CAN-CBX-modules. For the CANopen-PN module the following points apply accordingly.

12.3 Installation of the Module Using InRailBus-Connector

If the CAN bus signals and the power supply voltage shall be fed via the InRailBus, please proceed as follows:

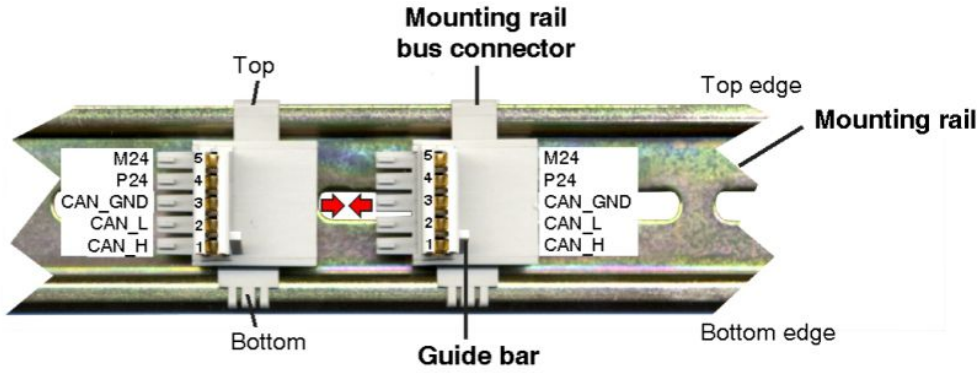


Figure 45: Mounting rail with bus connector

1. Position the InRailBus connector on the mounting rail and snap it onto the mounting rail using slight pressure. Plug the bus connectors together to contact the communication and power signals (in parallel with one). The bus connectors can be plugged together before or after mounting the CAN-CBX modules.

2. Place the CAN-CBX module with the DIN rail guideway on the top edge of the mounting rail.

Figure 46: Mounting CAN-CBX modules

3. Swivel the CAN-CBX module onto the mounting rail in pressing the module downwards according to the arrow as shown in figure 45. The housing is mechanically guided by the DIN rail bus connector.

4. When mounting the CAN-CBX module the metal foot catch snaps on the bottom edge of the mounting rail. Now the module is mounted on the mounting rail and connected to the InRailBus via the bus connector. Connect the bus connectors and the InRailBus, if not already done.

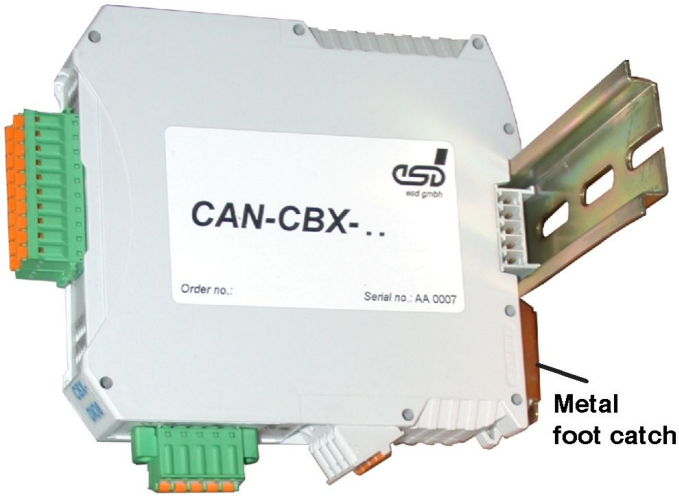


Figure 47: Mounted CAN-CBX module

12.3.1 Connecting Power Supply and CAN Signals to CBX-InRailBus

To connect the power supply and the CAN-signals via the InRailBus, a terminal plug (order no.: C.3000.02) is needed. The terminal plug is not included in delivery and must be ordered separately (see order information).

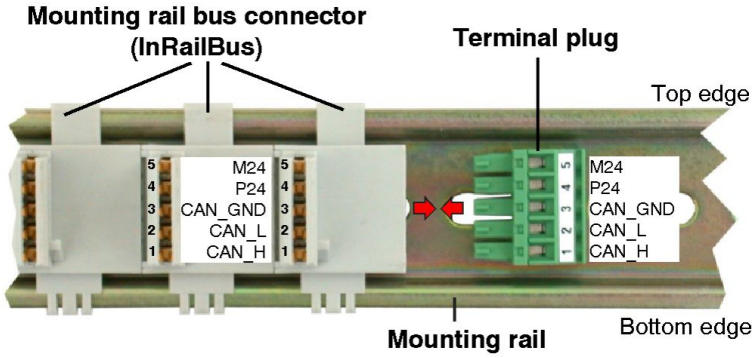


Figure 48: Mounting rail with InRailBus and terminal plug

Plug the terminal plug into the socket on the right of the mounting-rail bus connector of the InRailBus, as described in Figure 48. Then connect the CAN interface and the power supply voltage via the terminal plug.

12.3.2 Connection of the Power Supply Voltage

NOTICE
It is **not permissible** to feed through the power supply voltage through the CBX station and to supply it to another CBX station via 24V connector! A feed through of the +24 V power supply voltage can cause damage on the CBX modules.

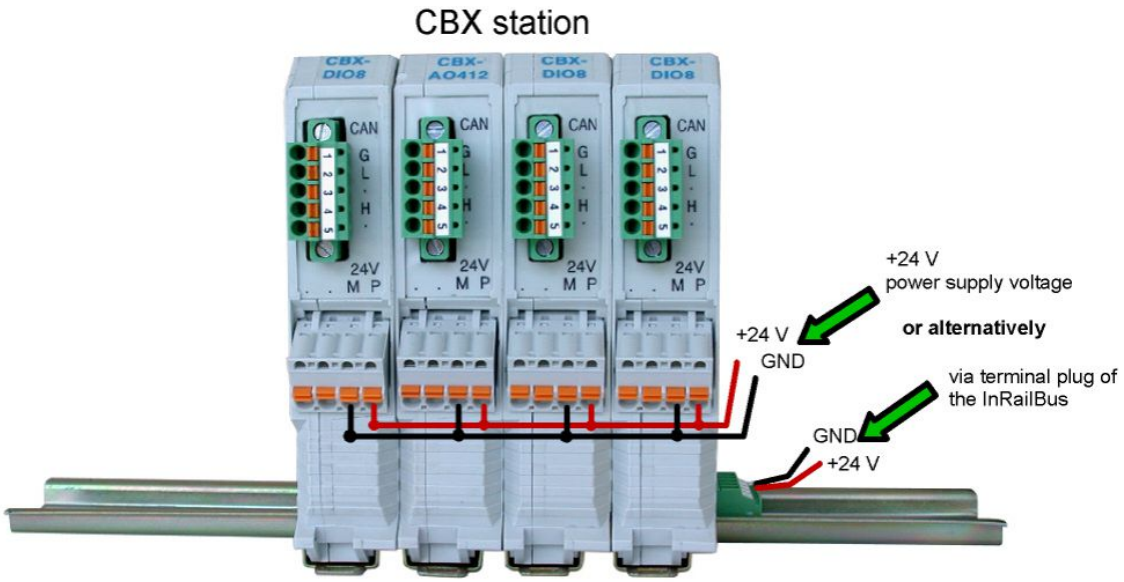


Figure 49: Connecting the power supply voltage to the CAN-CBX station

12.3.3 Connection of CAN

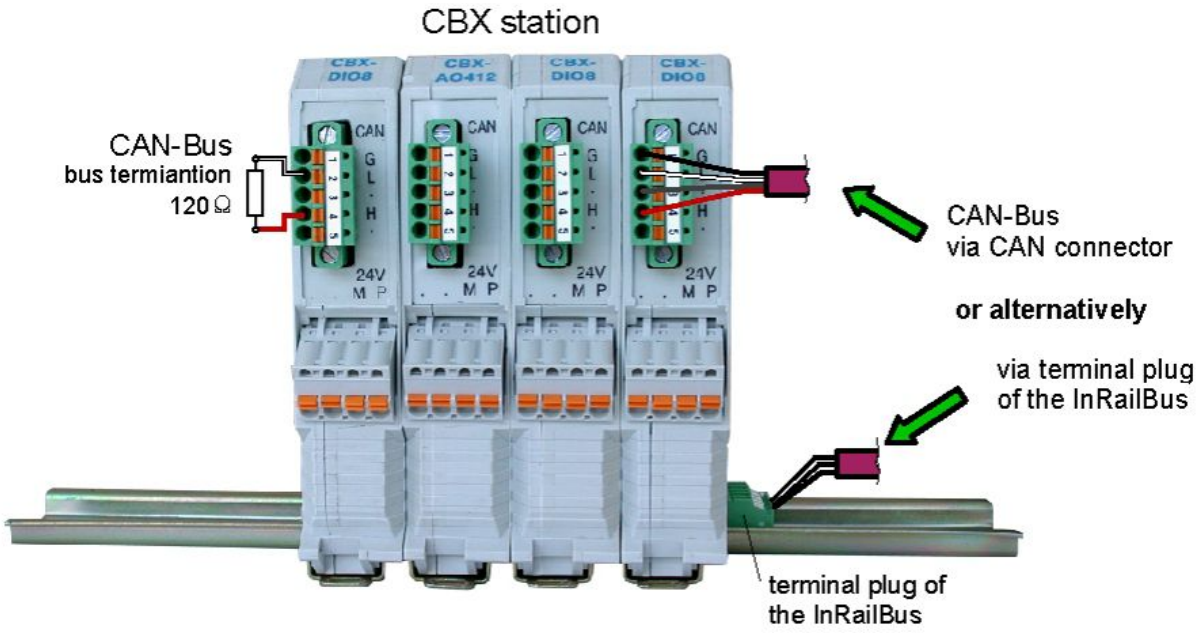


Figure 50: Connecting the CAN signals to the CAN-CBX station

Generally the CAN signals can be fed via the CAN connector of the first CAN-CBX module of the CBX station. The signals are then connected through the CAN-CBX station via the InRailBus. To lead through the CAN signals the CAN bus connector of the last CAN-CBX module of the CAN-CBX station has to be used. The CAN connectors of the CAN-CBX modules which are not at the ends of the CAN-CBX station must not be connected to the CAN bus, because this would cause incorrect branching.
A bus termination must be connected to the CAN connector of the CAN-CBX module at the end of the CBX-InRailBus (see Fig. 50), if the CAN bus ends there.

12.4 Removal of the CAN-CBX Module from InRailBus

If the CAN-CBX module is connected to the InRailBus please proceed as follows:

Release the module from the mounting rail in moving the foot catch (see Fig. 46) downwards (e.g. with a screwdriver). Now the module is detached from the bottom edge of the mounting rail and can be removed.

INFORMATION
It is possible to remove individual devices from the whole without interrupting the InRailBus connection, because the contact chain will not be interrupted.

13. References

[1] Phoenix Contact GmbH & Co. KG, Blomberg.
Technical data taken from COMBICON online catalogue,
http://www.phoenixcontact.com/assets/interactive_ed/local_de/modules/0000156/index.html
LPrinted-circuit board connector- FKCT-2,5/4-ST KMGY - 1921900,

http://www.phoenixcontact.com/assets/interactive_ed/local_de/modules/0000156/index.html
Printed-circuit board connector - FK-MCP 1,5/ 5-STF-3,81 - 1851261,
last downloaded 2012-11-21

[2] CiA 301 Specification V4.2.0 (02.2011)
CANopen application layer and communication profile,
CAN in Automation e. V., Nürnberg, Germany

[3] CiA 302-2 Draft Standard Proposal, V.4.1 (02.2009)
CANopen additional application layer,
Part 2: Network management,
CAN in Automation e. V., Nürnberg, Germany

[4] CiA 306 WD, V1.3.3 (08.2010)
Electronic device description,
Part 1: Electronic Data Sheet and Device Configuration File,
CAN in Automation e. V., Nürnberg, Germany

[5] CiA 309-1 Draft Standard V 1.1 (12 2006)
CANopen Interfacing CANopen with TCP/IP
Part 1: General principles and services,
CAN in Automation e. V., Nürnberg, Germany

[6] PROFIBUS International Document TC2-09-0002 V 1.0 2011
(CANopen-Integration_7012_V10_Mar11),
PROFIBUS Nutzerorganisation e.V., 76131 Karlsruhe, Germany

[7] CiA 303-3 Draft Recommendation V.1.3 2006
CANopen Additional specification
Part 3: Indicator specification
CAN in Automation e. V., Nürnberg, Germany

14. Declaration of Conformity

EU-KONFORMITÄTSERKLÄRUNG
EU DECLARATION OF CONFORMITY

Adresse / *Address*
esd electronics gmbh
Vahrenwalder Str. 207
30165 Hannover
Germany

esd erklärt, dass das Produkt *esd declares, that the product*	Typ, Modell, Artikel-Nr. *Type, Model, Article No.*
CANopen-PN	**C.2921.02**
die Anforderungen der Normen *fulfills the requirements of the standards*	**EN 61000-6-2:2005,** **EN 61000-6-3:2007/A1:2011**
gemäß folgendem Prüfbericht erfüllt. *according to test certificate.*	**H-K00-0461-12**
Das Produkt entspricht damit der EU-Richtlinie „EMV" *Therefore the product conforms to the EU Directive 'EMC'*	**2014/30/EU**
Das Produkt entspricht den EU-Richtlinien „RoHS" *The product conforms to the EU Directives 'RoHS'*	**2011/65/EU, 2015/863/EU**

Diese Erklärung verliert ihre Gültigkeit, wenn das Produkt nicht den Herstellerunterlagen entsprechend eingesetzt und betrieben wird, oder das Produkt abweichend modifiziert wird.
This declaration loses its validity if the product is not used or run according to the manufacturer's documentation or if non-compliant modifications are made.

Name / *Name*	T. Bielert
Funktion / *Title*	QM-Beauftragter / *QM Representative*
Datum / *Date*	Hannover, 2020-06-30

Rechtsgültige Unterschrift / *authorized signature*

I:\Texte\Doku\MANUALS\CAN\CANopen-PN\Konformitaetserklaerung\CANopen-PN_Konformitaetserklaerung_2020-06-30.docx

15. PROFINET IO Certificate

PROFIBUS • PROFINET

Certificate

PROFIBUS Nutzerorganisation e.V. grants to

esd electronics gmbh
Vahrenwalder Str. 207, 30165 Hannover, Germany

the Certificate No: **Z10767** for the PROFINET IO Device:

Model Name:	CANopen-PN
Revision:	SW/FW: V1.0.5; HW: 100
Identnumber:	0x015D; 0x0002
GSD:	GSDML-V2.3-#esd-CANopen-PN-test105-20140602.xml
DAP:	Linking Device: CANopen-PN, 0x20000000

This certificate confirms that the product has successfully passed the certification tests with the following scope:

☑	PNIO_Version	V2.2
☑	Conformance Class	B
☑	PNIO_Tester_Version	V2.2.4
☑	Tester	SIEMENS AG, Fürth, Germany, PN292-1

This certificate is granted according to the document:
"Framework for testing and certification of PROFIBUS and PROFINET products".
For all products that are placed in circulation by **June 13, 2026** the certificate is valid for life.

Karlsruhe, June 15, 2023

(Official in Charge)

Board of PROFIBUS Nutzerorganisation e. V.

(Xaver Schmidt)

(Frank Moritz)

PROFINET

16. Order Information

Type	Features	Order No.
CANopen-PN	PROFINET IO CANopen Gateway, Documentation and GSDML Composer	C.2921.02
Accessories (for usage with InRailBus only)		
 **CAN-CBX-TBUS**	Mounting-rail bus connector of the CBX-InRailBus for CAN-CBX-modules, (not included in delivery of C.2921.02!)	C.3000.01
 **CAN-CBX-TBUS-Connector**	Terminal plug of the CBX-InRailBus for the connection of the +24 V power supply voltage and the CAN interface Female type	C.3000.02
 **CAN-CBX-TBUS-Connection adapter**	Terminal plug of the CBX-InRailBus for the connection of the +24 V power supply voltage and the CAN interface Male type	C.3000.03

Table 16: Order information

PDF Manuals
Manuals are available in English and usually in German as well. For availability of English manuals see table below.
Please download the manuals as PDF documents from our esd website www.esd.eu for free.

Manuals		Order No.
CANopen-PN-ME	English manual of CANopen-PN	C.2921.21
CAN-API-ME	NTCAN-API manual 1/2: Functions (English) NTCAN-API manual 2/2: Installation (English)	C.2001.21
CANopen-ME	CANopen manuals in English	C.2002.21

Table 17: Available manuals

Printed Manuals
If you need a printout of the manual additionally, please contact our sales team: sales@esd.eu for a quotation. Printed manuals may be ordered for a fee.