



CANopen-PN/2

CANopen[®]-PROFINET[®]-IO Gateway



Manual

For Product C.2931.02

Notes

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, the 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.

All rights to this documentation are reserved by esd electronics. Distribution to third parties, and reproduction of this document in any form, whole or in part, are subject to esd electronics' written approval.

© 2023 esd electronics gmbh, Hannover

esd electronics gmbh

Vahrenwalder Str. 207
30165 Hannover
Germany

Tel.:	+49-511-37298-0
Fax:	+49-511-37298-68
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/2. Carefully read this manual before commencing any work and follow the instructions.
The manual is a product component, please retain it for future use.

Links

esd electronics gmbh assumes no liability or guarantee for the content of Internet pages to which this document refers directly or indirectly. Visitors follow links to websites at their own risk and use them in accordance with the applicable terms of use of the respective websites.

Trademark Notices

CANopen® and CiA® are registered EU trademarks of CAN in Automation e.V.

PROFINET® and PROFIBUS® are registered EU trademarks of PROFIBUS Nutzerorganisation e.V.

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

Document Information

Document file:	I:\Texte\Doku\MANUALS\CAN\CANopen-PN2\Englisch\CANopen-PN2_Manual_en_12.docx
Date of print:	2023-10-19
Document-type number:	DOC0800

Hardware version.:	from Rev. 3.0
Software version:	from Rev. 3.0

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.

Rev.	Chapter	Changes versus previous version	Date
1.0	-	First English manual of CANopen-PN/2	2023-06-20
1.1	5.2	Reference to chapter 15. included	2023-09-14
	15.	New chapter: "Software Licenses"	
1.2	15.2.3	New chapter: "Open Source Software Copy"	2023-10-19

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 indicates that the device contains components sensitive to electrostatic discharge.



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/2 follow the instructions below and read the manual carefully to protect yourself from injury and the CANopen-PN/2 from damage.
- Do not use damaged or defective cables to connect the CANopen-PN/2 and follow the CAN wiring hints in chapter: "Correct Wiring of 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.
- The galvanic isolation of the CANopen-PN/2 has only functional tasks and is not a protection against hazardous electrical voltage.
- The CANopen-PN/2 is a device of protection class III according to DIN EN IEC 61010-2-201 and may only be operated on supply circuits that offer sufficient protection against dangerous voltages.
- External circuits connected to the interfaces of the CANopen-PN/2 must be sufficiently protected against dangerous voltage.
- Compliance with the applicable national safety regulations is the responsibility of the user.
- Do not open the housing of the CANopen-PN/2 .
- The CANopen-PN/2 must be securely installed before commissioning.
- The permitted operating position is specified as shown (Figure 3). Other operating positions are not allowed.
- Never let liquids get inside CANopen-PN/2. Otherwise, electric shocks or short circuits may result.
- Protect the CANopen-PN/2 from dust, moisture, and steam.
- Protect the CANopen-PN/2 from shocks and vibrations.
- The CANopen-PN/2 may become warm during normal use. Always allow adequate ventilation around the CANopen-PN/2 and use care when handling
- Do not operate the CANopen-PN/2 adjacent to heat sources and do not expose it to unnecessary thermal radiation. Ensure an ambient temperature as specified in the technical data.



NOTICE

Electrostatic discharges may cause damage to electronic components.

→ Take the appropriate precautions for handling electrostatic discharge sensitive devices.

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/2 is an industrial product and meets the demands of the EU regulations and EMC standards printed in the conformity declaration at the end of this manual.

Warning: In a residential, commercial, or light industrial environment the CANopen-PN/2 may cause radio interferences in which case the user may be required to take adequate measures.

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/2 is the operation 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/2 is intended for indoor operation only.
- The operation of the CANopen-PN/2 in hazardous areas, or areas exposed to potentially explosive materials is not permitted.
- The operation of the CANopen-PN/2 for medical purposes is prohibited.

Service Note

The CANopen-PN/2 does not contain any parts that require maintenance by the user. The CANopen-PN/2 does not require any manual configuration of the hardware. Unauthorized intervention in the device voids warranty claims

Disposal



Products marked with a crossed-out dustbin must not be disposed of with household waste. Devices which have become defective in the long run must be disposed in an appropriate way or must be returned to the manufacturer for proper disposal. Please, contribute to environmental protection.

Typographical Conventions

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

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

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.

Table of Contents

Safety Instructions	5
1 Overview.....	12
1.1 Description of CANopen-PN/2	12
1.2 Glossary	14
1.3 View with Connectors	15
1.4 LEDs.....	16
1.4.1 Position of the LEDs	16
1.4.2 PROFINET IO LEDs	16
1.4.3 Status LEDs.....	17
1.5 Labels.....	18
2 Installing and Uninstalling Hardware	19
3 Start-Up	20
4 CANopen Protocol	21
4.1 Definition and Terms.....	21
4.2 CANopen Objects	21
4.3 Process Data Objects (PDOs)	22
4.4 Service Data Objects (SDOs)	22
4.4.1 Communication Parameters for SDO Transfers	22
4.4.2 Error Codes of a SDO Transfer.....	24
4.5 Network Management (NMT).....	25
4.6 Node Guarding and Heartbeat	25
4.7 Important CANopen Telegrams	26
5 Software	27
5.1 Functionality	27
5.2 Licenses	27
5.3 Installation	28
5.3.1 Manual Installation of the RNDIS Driver.....	29
5.3.2 Example Project for TIA Portal.....	29
5.4 Configuration	30
5.4.1 Quick Start Guide	30
5.4.2 Configuration of the CANopen Network	30
5.4.3 Installation of the GSDML File	30
5.4.4 Insert the CANopen-PN/2	31
5.4.5 Assign the PROFINET Network.....	32
5.4.6 Assign IP Address and PROFINET Device Name.....	33
5.4.7 Compile and Download Hardware and Software.....	35
5.5 GSDML Composer	37
5.5.1 Quick Start Guide	37
5.5.2 Description.....	38
5.5.3 Features	38
5.5.4 System Requirements.....	39
5.5.5 Compability.....	39
5.5.6 Installation	39
5.5.7 Overview.....	40
5.5.8 Menu Bar	41
5.5.8.1 File	41
5.5.8.2 View.....	42
5.5.8.3 Settings	42
5.5.8.4 Window.....	42
5.5.8.5 Help	42
5.5.9 Device Library.....	43
5.5.10 CANopen Network Editor.....	45
5.5.11 CANopen Manager	47
5.5.12 CANopen Device	50

5.5.12.1	Device Information.....	50
5.5.12.2	RPDO Mapping	51
5.5.12.3	TPDO Mapping.....	54
5.5.12.4	Manager Settings.....	57
5.5.12.5	SYNC / Emergency.....	59
5.5.12.6	Heartbeat / Guarding	61
5.5.12.7	Object Lists.....	63
5.5.12.8	EDS Device Info	64
5.5.12.9	EDS File Info	64
5.5.12.10	EDS Comments.....	64
5.5.13	Output.....	65
5.6	Module In- and Output	66
5.6.1	Overview.....	67
5.6.2	CANopen Manager.....	67
5.6.3	CANopen Devices	68
5.7	Diagnostics	70
5.7.1	Alarms	70
5.7.2	Provider and Consumer Status	75
5.8	Records.....	76
5.8.1	SDO Upload (0xB711)	78
5.8.2	SDO Download (0xB713).....	81
5.8.3	Configure SDO Timeout (0xB715)	82
5.8.4	Start CANopen Device (0xB731)	82
5.8.5	Stop CANopen Device (0xB732).....	82
5.8.6	Set CANopen Device to PRE-OPERATIONAL (0xB733)	83
5.8.7	Reset CANopen Device (0xB734).....	83
5.8.8	Reset Communication (0xB735)	83
5.8.9	Initialize Gateway (0xB751)	83
5.8.10	Set Heartbeat Producer (0xB754).....	84
5.8.11	Set Node ID (0xB755).....	85
5.8.12	Start Emergency Consumer (0xB756)	85
5.8.13	Stop Emergency Consumer (0xB757).....	85
5.8.14	Read Version (0xB762).....	86
5.8.15	Reset CANopen Manager (0xB771).....	86
5.8.16	Start CANopen Manager (0xB772)	86
5.8.17	Stop CANopen Manager (0xB773).....	87
5.8.18	Reset CANopen Device EMCY (0x003A).....	87
5.8.19	PLC Function Blocks	88
5.8.19.1	Read Records.....	88
5.8.19.2	Write Records.....	89
5.9	CANopen-PN/2 Object Directory.....	90
5.9.1	Objects of CiA Specification CiA 301	90
5.9.2	Objects of CiA Specification CiA 302-2	91
6	Firmware Update	92
7	CAN Monitoring	93
8	Compatibility	95
8.1	CANopen-PN Compatibility Mode.....	95
9	Troubleshooting	96
9.1	Faulty PROFINET Connection	96
9.2	Faulty CAN Bus	96
9.3	Faulty CANopen Device.....	97
9.4	Invalid CAN Busload	98
9.5	Support by esd.....	98
10	Technical Data.....	99
10.1	General Technical Data	99
10.2	CPU and Memory	99
10.3	Connectors accessible from Outside.....	100
10.4	PROFINET IO Interface	100

10.5	DIAG Interface	100
10.6	CAN Interface	101
11	Connector Pin Assignments	102
11.1	CAN	102
11.2	24V Power Supply Voltage	103
11.3	PROFINET IO	104
11.4	DIAG	105
12	Correct Wiring of Electrically Isolated CAN Networks	106
12.1	CAN Wiring Standards	106
12.2	Light Industrial Environment (<i>Single Twisted Pair Cable</i>)	107
12.2.1	General Rules	107
12.2.2	Cabling	108
12.2.3	Branching	108
12.2.4	Termination Resistor	108
12.3	Heavy Industrial Environment (<i>Double Twisted Pair Cable</i>)	109
12.3.1	General Rules	109
12.3.2	Device Cabling	110
12.3.3	Branching	110
12.3.4	Termination Resistor	110
12.4	Electrical Grounding	111
12.5	Bus Length	111
12.6	Examples for CAN Cables	112
12.6.1	Cable for Light Industrial Environment Applications (<i>Two-Wire</i>)	112
12.6.2	Cable for Heavy Industrial Environment Applications (<i>Four-Wire</i>)	112
13	CAN Troubleshooting Guide	113
13.1	Electrical Grounding	114
13.2	Short Circuit in CAN Wiring	114
13.3	Correct Voltage Levels on CAN_H and CAN_L	114
13.4	CAN Transceiver Resistance Test	115
13.5	Support by esd	115
14	References	116
15	Software Licenses	117
15.1	3 rd Party Software License Terms	117
15.2	Licence Conditions of the Software Modules	117
15.2.1	Yocto-Linux License Modules	117
15.2.2	Others	120
15.2.3	Open Source Software Copy	120
16	Declaration of Conformity	121
17	PROFINET IO Certificate	122
18	Order Information	123

List of Tables

Table 1:	Description of PROFINET IO LEDs	16
Table 2:	Indicator states of the Status LEDs	17
Table 3:	Description of Status LEDs	17
Table 4:	Installing and uninstalling hardware	19
Table 5:	CANopen Object Directory	21
Table 6:	SDO Communication Parameter 1	22
Table 7:	SDO Communication Parameter 2	23
Table 8:	CANopen SDO Error Codes	24
Table 9:	CANopen NMT States	25
Table 10:	Important CANopen Telegrams	26
Table 11:	Manual installation of the RNDIS driver	29
Table 12:	Configuration - Quick Start Guide	30
Table 13:	GSDML Composer Quick Start Guide	37

Table 14: GSDML Composer <i>File</i> Parameter	41
Table 15: GSDML Composer <i>View</i> Parameter	42
Table 16: GSDML Composer <i>Settings</i> Parameter	42
Table 17: GSDML Composer <i>Device Library</i> Parameter	43
Table 18: GSDML Composer Display Options.....	46
Table 19: GSDML Composer Device Context Menu.....	46
Table 20: GSDML Composer CANopen Manager Parameter.....	49
Table 21: GSDML Composer RPDO Mapping Parameter	52
Table 22: GSDML Composer TPDO Mapping Parameter.....	55
Table 23: GSDML Composer Manager Settings Parameter	58
Table 24: GSDML Composer <i>Sync/Emergency</i> Parameter	60
Table 25: GSDML Composer Sync/Emergency Parameter	62
Table 26: GSDML Composer Object List Parameter	64
Table 27: CANopen-PN/2 Slot Structure	67
Table 28: PLC Address Space \leftrightarrow CANopen Objects	69
Table 29: Read Alarm Function Block Parameter.....	71
Table 30: Alarm Information	72
Table 31: CANopen Error Codes.....	74
Table 32: CANopen Error Register.....	74
Table 33: CANopen-PN/2 Read Records	76
Table 34: CANopen-PN/2 Write Records	77
Table 35: Write Record SDO Upload (0xB711)	78
Table 36: Read Record SDO Upload (0xB711)	78
Table 37: CANopen SDO Data types	79
Table 38: SDO Transfer Error Code	80
Table 39: Write Record SDO Download (0xB713).....	81
Table 40: Write Record Configure SDO Timeout (0xB715).....	82
Table 41: Write Record Start CANopen Device (0xB731).....	82
Table 42: Write Record Stop CANopen Device (0xB732).....	82
Table 43: Write Record Set CANopen Device to PRE-OP. (0xB733).....	83
Table 44: Write Record Reset CANopen Device (0xB734)	83
Table 45: Write Record Reset Communication (0xB735).....	83
Table 46: Write Record Initialize Gateway (0xB751).....	83
Table 47: CAN Bit Timing	84
Table 48: Write Record Set Heartbeat Producer (0xB754)	84
Table 49: Write Record Set Node ID (0xB755).....	85
Table 50: Write Record Start Emergency Consumer (0xB756).....	85
Table 51: Write Record Stop Emergency Consumer (0xB757).....	85
Table 52: Read Record Read Version (0xB762).....	86
Table 53: Write Record Reset CANopen Manager (0xB771).....	86
Table 54: Write Record Start CANopen Manager (0xB772).....	86
Table 55: Write Record Stop CANopen Manager (0xB773).....	87
Table 56: Write Record Reset CANopen Device EMCY (0x003A).....	87
Table 57: Read Record Function Block Parameter.....	88
Table 58: Write Record Function Block Parameter	89
Table 59: CANopen-PN/2 object directory CiA Specification 301	91
Table 60: CANopen-PN/2 object directory CiA Specification 302-2	91
Table 61: Firmware Update	92
Table 62: CAN Monitoring	93
Table 63: General Data of the module	99
Table 64: CPU and Memory	99
Table 65: Connectors, accessible from outside	100
Table 66: Data of the PROFINET IO interface.....	100
Table 67: Data of the USB device interface	100
Table 68: Data of the CAN interface	101
Table 69: Recommended cable lengths at typical bit rates (with esd-CAN interfaces).....	111
Table 70: Order information hardware	123
Table 71: Available manuals	123

List of Figures

Figure 1: PROFINET®-IO-Device to CANopen Manager Gateway	12
Figure 2: Block circuit diagram of CANopen-PN/2.....	12
Figure 3: Connecting diagram of CANopen-PN/2.....	15
Figure 4: LEDs.....	16
Figure 5: View with name plate (example).....	18
Figure 6: View with LED/Connector label	18
Figure 7: Manage GSDML files	30
Figure 8: Inserting the CANopen-PN/2.....	31
Figure 9: Choose GSDML file of the CANopen-PN/2	32
Figure 10: Not assigned CANopen-PN/2.....	32
Figure 11: Assigned CANopen-PN/2.....	32
Figure 12: Assign IP address and device name of the configuration	33
Figure 13: Assign IP address and device name of the gateway	34
Figure 14: Compile hardware and software (detail)	35
Figure 15: Download hardware and software to device (detail)	35
Figure 16: Toolbar with button <i>Go online</i>	36
Figure 17: Online <i>Device overview</i>	36
Figure 18: GSDML Composer diagram	38
Figure 19: GSDML Composer Gateway Selection	39
Figure 20: GSDML Composer Main Window.....	40
Figure 21: GSDML Composer <i>Project Settings</i>	42
Figure 22: GSDML Composer <i>Device Library</i>	43
Figure 23: GSDML Composer <i>CANopen Network Editor</i>	45
Figure 24: GSDML Composer <i>CANopen Manager</i>	47
Figure 25: GSDML Composer Device Information	50
Figure 26: GSDML Composer RPDO Mapping	51
Figure 27: GSDML Composer TPDO Mapping	54
Figure 28: GSDML Composer <i>Manager Settings</i>	57
Figure 29: GSDML Composer Sync/Emergency	59
Figure 30: GSDML Composer <i>Heartbeat/Guarding</i>	61
Figure 31: GSDML Composer Object Lists	63
Figure 32: GSDML Composer <i>Output</i>	65
Figure 33: GSDML Composer <-> Siemens TIA Portal	66
Figure 34: <i>Device Overview</i>	67
Figure 35: <i>Device Overview</i> CANopen Server Mapping.....	68
Figure 36: PDO Mapping CAN-CBX-AO412	69
Figure 37: PDO Mapping CAN-CBX-DIO8	69
Figure 38: Alarm Diagnostics Information	70
Figure 39: CAN Control Panel.....	93
Figure 40: Monitoring the CAN Bus with CANreal	94
Figure 41: Faulty CAN Bus	96
Figure 42: Faulty/Missing CANopen device.....	97
Figure 43: CANopen device EMCY	97
Figure 44: CAN wiring for light industrial environment.....	107
Figure 45: Example for proper wiring with single shielded single twisted pair wires.....	108
Figure 46: CAN wiring for heavy industrial environment.....	109
Figure 47: Example of proper wiring with single shielded double twisted pair cables	110
Figure 48: Simplified diagram of a CAN network.....	113
Figure 49: Simplified schematic diagram of ground test measurement.....	114
Figure 50: Measuring the internal resistance of CAN transceivers	115

1 Overview

1.1 Description of CANopen-PN/2

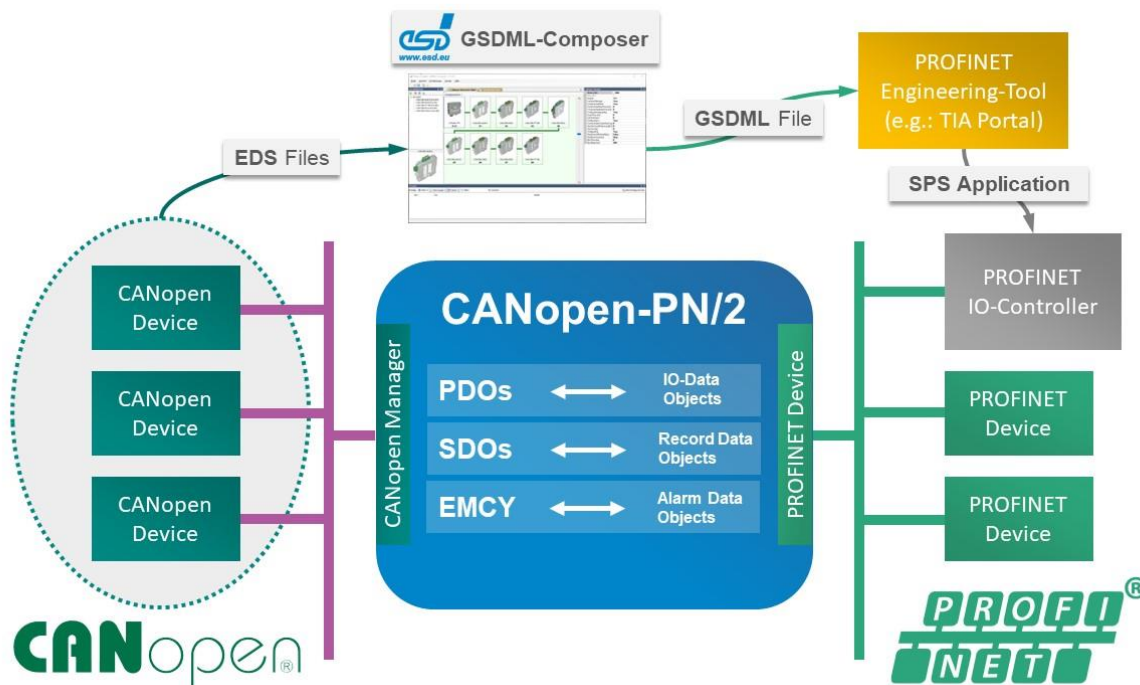


Figure 1: PROFINET®-IO-Device to CANopen Manager Gateway

The CANopen-PN/2 connects any PROFINET IO controller to a CANopen® network. The fieldbus gateway operates as a high-performance PROFINET IO device with a maximum of 1440 bytes of input data and 1440 bytes of output data. It is designed according to the Profibus International PROFIBUS International Document TC2-09-0002 (1) and CANopen Specification CiA 309-1 (2). On the CANopen side it acts as a CANopen manager and supports Network Management (NMT), Node Guarding and Heartbeat.

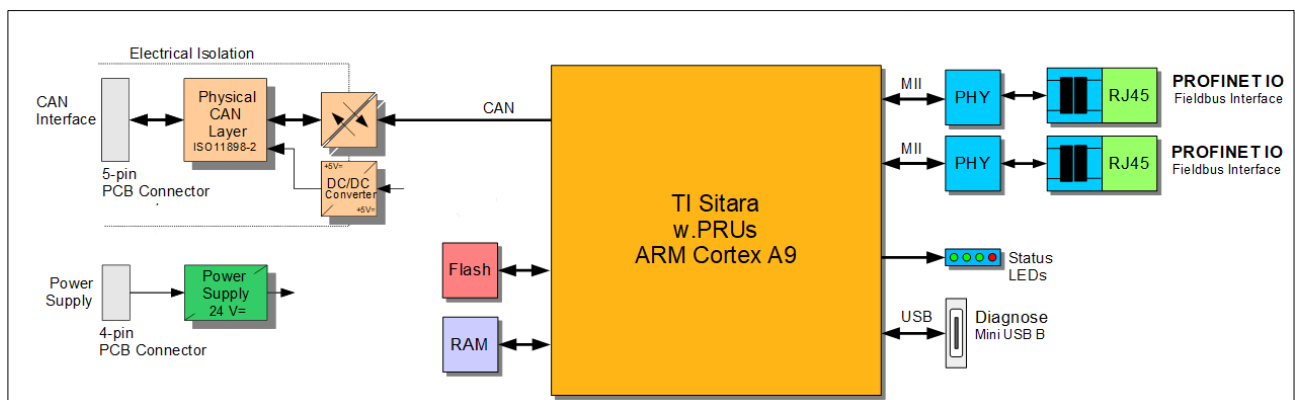


Figure 2: Block circuit diagram of CANopen-PN/2

The CANopen-PN/2 comes in a compact housing for DIN rail mounting with easily accessible connectors. It is equipped with two Ethernet ports via RJ-45 sockets for PROFINET IO, a CAN interface via a connector with spring-cage connection and a Mini-USB-B interface for diagnose and firmware update.

Physical Interfaces

The CAN interface of the gateway is ISO 11898-2:2016 compliant. The 100BASE-TX PROFINET IO interface is IEEE802.3 (3) compliant and allows a data transfer rate of 100 Mbit/s. Both the PROFINET IO and CAN interfaces are galvanically isolated from the rest of the circuit.

Control your CANopen Devices

The CANopen-PN/2 supports the functionality of a CANopen manager according to CiA[®] 302-2 (4). As such, the gateway is able to start, configure and stop all CANopen devices in the network. The gateway supports up to 126 CANopen devices.

High-speed data exchange

The CANopen-PN/2 supports the exchange of data between PROFINET IO and CANopen networks with PROFINET cycle times up to 1ms. It is able to exchange data via PDO and SDO.

Alarm Management

The CANopen-PN/2 supports an extended alarm management to check the CANopen network including the CAN bus status, the CANopen device status as well as Emergency (EMCY) Frames.

Configurable for your needs

The gateway is configurable in a simple manner exactly to fit your needs. It uses a simple configuration tool, called the GSDML-Composer, for individual generation of the matching configuration.

Monitoring the CAN bus

The gateway supports the monitoring of the CAN bus with the included EtherCAN interface via Mini-USB.

Extensive debugging is possible with CAN diagnostic software via the USB interface. Our CAN Tools for esd boards (CANreal, CANplot and COBview) are available free-of-charge.

1.2 Glossary

Abbreviations

Abbreviation	Term
API	Application Programming Interface
BSP	Board Support Package
CAL	CAN Application Layer
CAN	Controller Area Network
CPU	Central Processing Unit
CiA	CAN in Automation
DCF	Device Configuration File
EDS	Electronic Data Sheet
GSD	General Station Description
GSDML	General Station Description Markup Language
HW	Hardware
I/O	Input/Output
IO-CS	PROFINET Consumer Status
IO-PS	PROFINET Provider Status
LSB	Least Significant Bit
MSB	Most Significant Bit
n.a.	not applicable
OS	Operating System
PDO	Process Data Object
PRU	Processor Realtime Unit
RTR	Remote Transmission Request
SDK	Software Development Kit
SDO	Service Data Object
USB	Universal Serial Bus
XML	Extensible Markup Language

1.3 View with Connectors

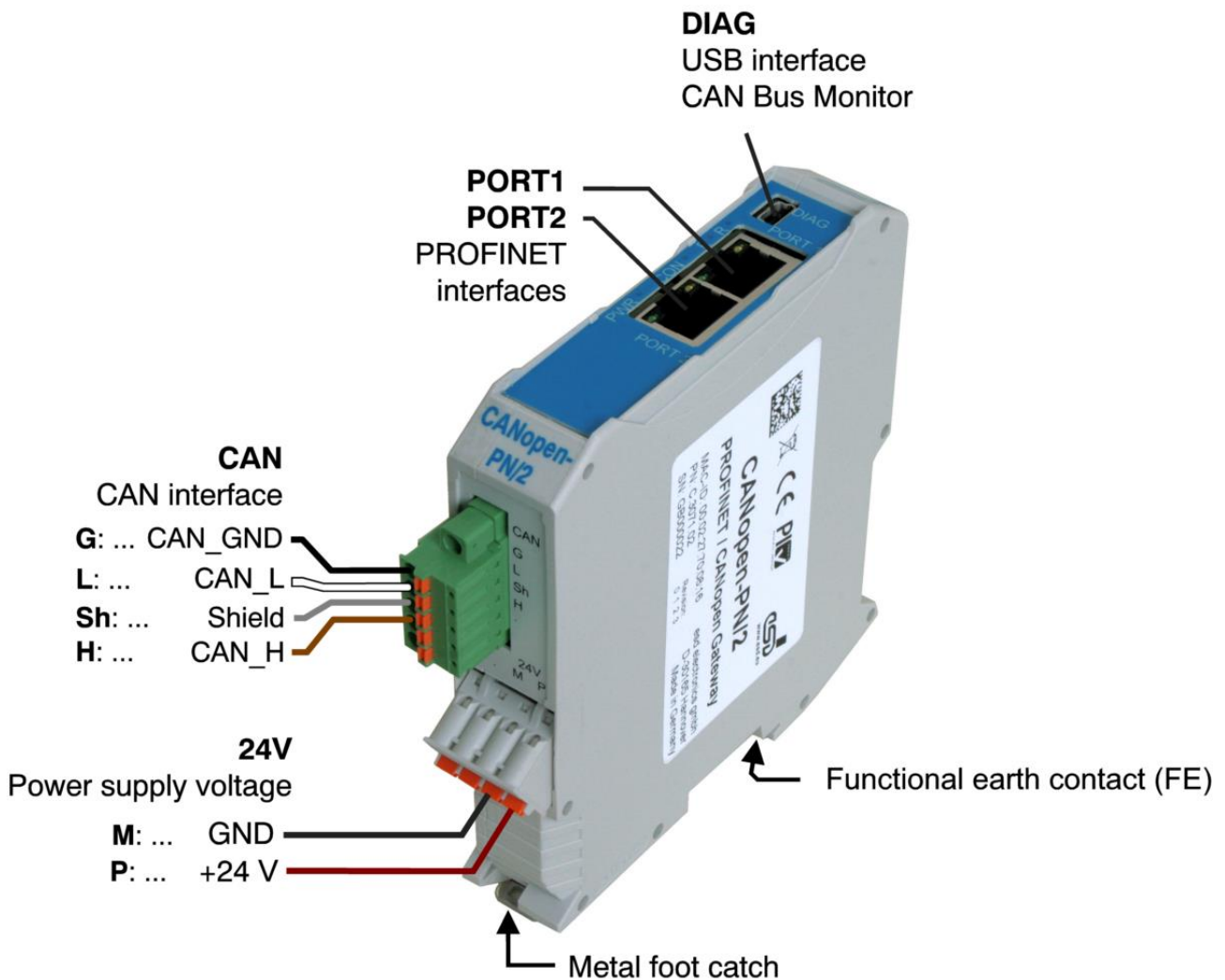


Figure 3: Connecting diagram of CANopen-PN/2

See also page 102 for signal assignment of the CAN connectors.



NOTICE

Read chapter “Installing and Uninstalling Hardware” on page 19, before you start with the installation of the hardware!

1.4 LEDs

1.4.1 Position of the LEDs

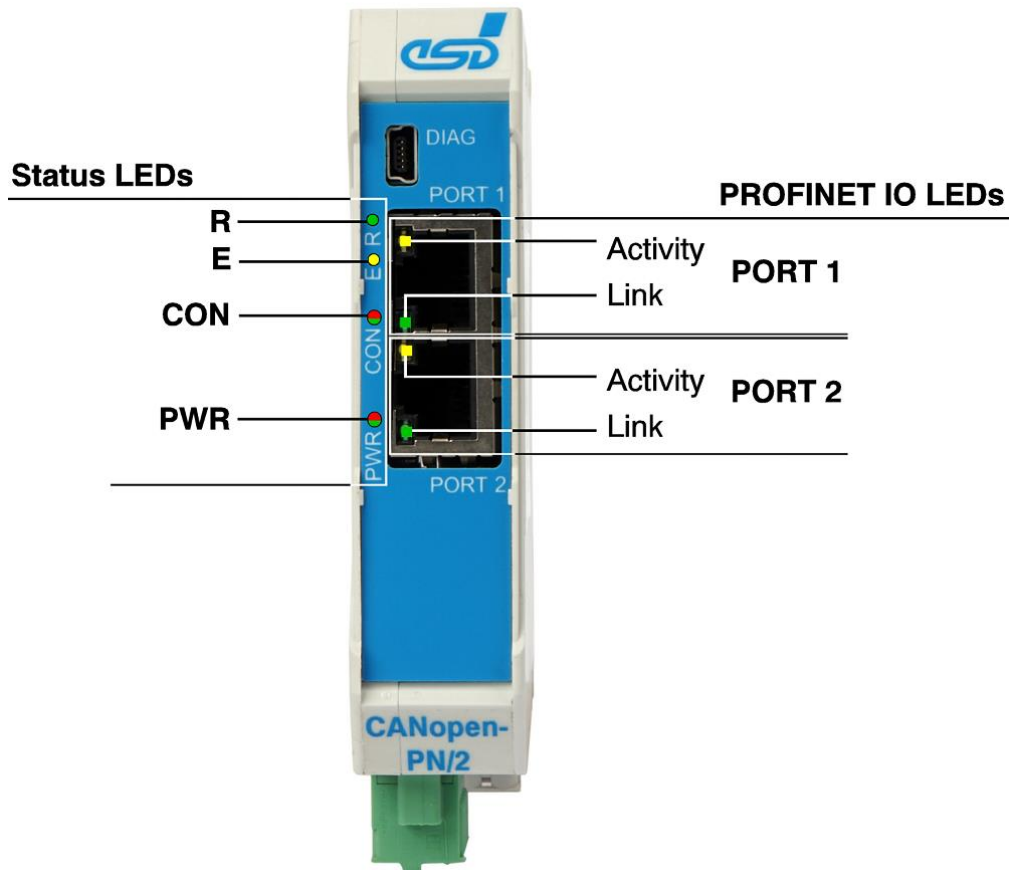


Figure 4: LEDs

1.4.2 PROFINET IO LEDs

The PROFINET IO LEDs of PORT 1 and PORT 2 are integrated in the RJ-45 sockets. The LEDs indicate the status of the corresponding port.

LED	Colour	Indicator State	Description
Activity	Yellow	Off	No Ethernet connection
		Blinking	Ethernet connection is established, data is transferred
		On	Ethernet connection is established
Link	Green	Off	No Ethernet connection
		On	Ethernet connection is established

Table 1: Description of PROFINET IO LEDs

1.4.3 Status LEDs

Indicator State	Description
On	LED on
Off	LED off
Blinking	LED blinks with 1 Hz (PROFINET) / 2,5 Hz (CANopen)
Single flash	LED 200 ms on, 1000 ms off
Double flash	LED 200 ms on, 200 ms off, 200 ms on, 1000 ms off

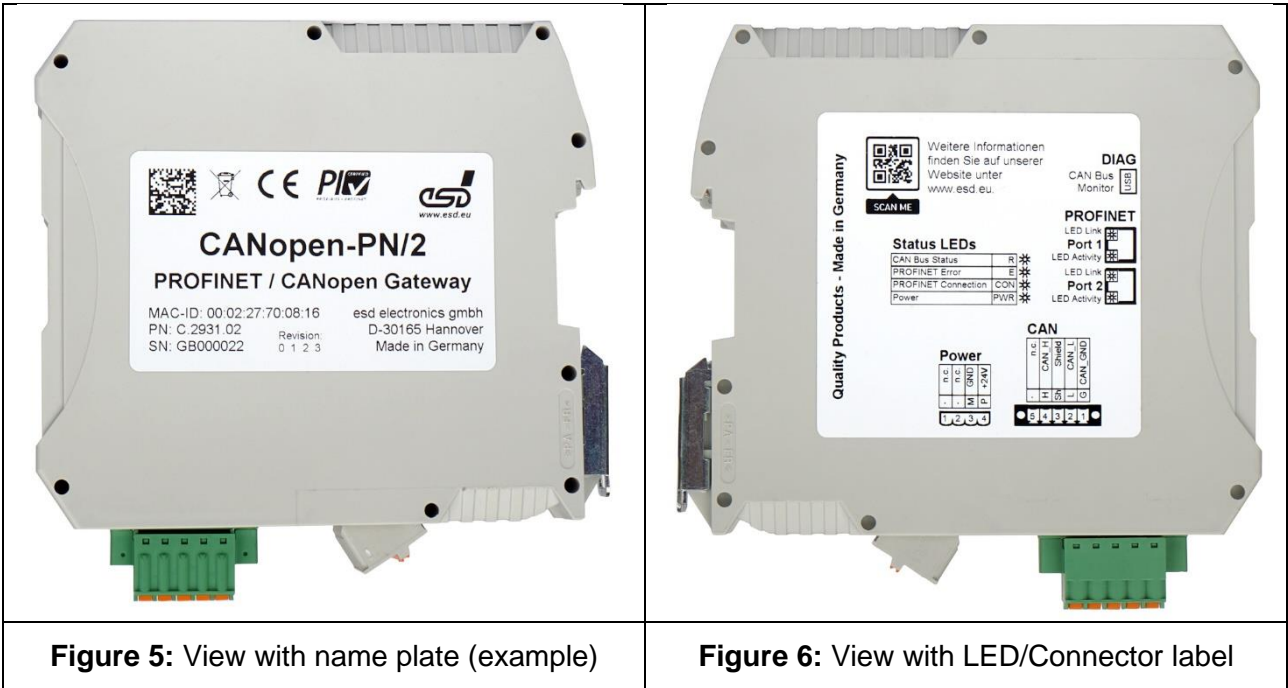
Table 2: Indicator states of the Status LEDs

LED	Function	Colour	Indicator State	Description
R	CANopen Status	Green	Off	<i>CANopen Manager Not Running</i>
			Blinking	<i>PRE-OPERATIONAL</i>
			Single flash	<i>STOPPED</i>
			Double flash	<i>CANopen Manager Recovery</i>
			On	<i>OPERATIONAL</i>
E	CANopen Error	Red	Off	No error
			Single flash	<i>Warning Limit Reached -</i> At least one of the error counters of the CAN controller has reached or exceeded the warning level.
			Double flash	<i>Error Control Event -</i> a Heartbeat- or Nodeguard error occurred
			On	The CAN controller is <i>Bus off</i>
CON	PROFINET IO Connection	Green	Off	No valid PROFINET IO link
			Blinking	Request of the PROFINET IO Controller for the identification of the device
			On	Valid PROFINET IO link is established
PWR	POWER	Green	Off	The application is not running.
			Blinking	The application is running

Table 3: Description of Status LEDs

See also chapter 3 on page 20 for a description of the status LEDs during Start-Up.

1.5 Labels



The name plate (Figure 5) shows among others the name, MAC-ID, esd order No. (PN) and the serial number (SN).

Name plate	CANopen Gateway
Name:	CANopen-PN/2
MAC-ID:	Individual MAC-ID of the module e.g.: 00:02:27:70:08:16
PN (esd order No.):	C.2931.02
SN (Serial number):	Individual number of the module e.g. GB000022

The LED/Connector label (Figure 6) shows short descriptions of the LEDs and connectors and the QR code of esd.

LED/Connector label	CANopen Gateway
LEDs:	Status LEDs, PROFINET LEDs
Connectors:	DIAG, PROFINET, Power, CAN

2 Installing and Uninstalling Hardware

To put the CANopen-PN/2 into operation, please follow the installation notes.




Step	Procedure	See Page
	NOTICE 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/2 is to be integrated. → The CANopen-PN/2 is a device of protection class III according to DIN EN IEC 61010-2-201 and may only be operated on supply circuits that offer sufficient protection against dangerous voltages. → External circuits connected to the interfaces of the CANopen-PN/2 must be sufficiently protected against dangerous voltages. → Compliance with the applicable national safety regulations is the responsibility of the user. → Ensure the absence of voltage before starting any electrical work.	
To install, continue as described from steps 1. to 4. To uninstall, continue from step 5.		
1.	Mount the CANopen-PN/2 module and connect the interfaces (power supply voltage, CAN, PROFINET interface) as described in Figure 3: Connecting diagram of CANopen-PN/2	15
	See also chapter 11 for 'Connector Pin Assignments'.	102
	NOTICE Incorrect wiring of the 24V power supply voltage can cause damage to the module! → Make absolutely sure to connect the cables correctly to the 24V line connector! → Use only suitable cables for the line plug	103
2.	Please note that the CAN bus must be terminated at both ends! esd offers special T-connectors and termination connectors for external termination. Additionally, the CAN_GND signal must be connected to earth at exactly one point in the CAN network. For details, please read chapter 'Correct Wiring of Electrically Isolated CAN Networks'.	106
3.	Switch on the 24 V-power supply voltage of the CANopen-PN/2	
4.	Continue with the installation of the software, as described in chapter 'Software'.	27
To uninstall the CANopen-PN/2 continue as described below.		
5.	Make sure that all connected interfaces are switched off. Disconnect the CANopen-PN/2 from the connected interfaces. If applicable, loosen the fastening of the CANopen-PN/2. Carefully pull the CANopen-PN/2 out.	

Table 4: Installing and uninstalling hardware

3 Start-Up

After switching on the supply voltage, the CANopen-PN/2 starts automatically. During start up the 'R' LED (CANopen Status) turns on. When the device is started successfully 'PWR' LED (Power) turns on and 'R' (CANopen Status) turns off again. This process takes about 10s.

The gateway is now ready to be configured by the PROFINET controller.

When the gateway has established a connection to the PROFINET network, the 'CON' LED (PROFINET Connect) turns ON. When the CAN bus is not faulty, the 'R' LED (CANopen Status) turns on, too.

After the PROFINET controller changes to state RUN, the data exchange is started automatically. When the PLC changes to the state STOP, no more CAN frames are sent.

4 CANopen Protocol

This chapter contains some basic information about the CANopen protocol. It describes only a part in a simplified form for better understanding of the following chapters of this manual. Further information can be found in the CANopen specification CiA 301 and on the knowledge pages of the CiA: (<https://www.can-cia.org/>)

4.1 Definition and Terms

COB	Communication Object
EMCY	Emergency
NMT	Network Management
PDO	Process Data Object
SDO	Service Data Object
SYNC	Sync

Manager/ Device	In accordance with the CiA's recommendation on inclusive language, the terms manager and device are used instead of the previously used terms master and slave.
--------------------	---

4.2 CANopen Objects

Each CANopen device comes with an object directory which is used for configuration and diagnostics. Each object is referenced by a 16-bit index, which is normally displayed as a 4-digit hexadecimal value (e.g. 0x1000), and a 8-bit sub-index, which is normally displayed as a 2-digit hexadecimal value (e.g. 0x10). Each CANopen object can be defined by multiple parameters, which define for example the data type, the value range, or the accessibility of the object.

There are specific index ranges that can be defined as follows:

Index	Description
0x0000	Reserved
0x0001 ... 0x025F	Data types
0x0260 ... 0x0FFF	Reserved
0x1000 ... 0x1FFF	Communication Profile Area
0x2000 ... 0x5FFF	Manufacturer Specific Profile Area
0x6000 ... 0x9FFF	Standardized Device Profile
0xA000 ... 0xFFFF	Reserved

Table 5: CANopen Object Directory

4.3 Process Data Objects (PDOs)

Process Data Objects (PDOs) are used to exchange process data. Process data that should be received by the CANopen device is called RPDO or formerly RX PDO. On the other hand, process data that should be transmitted by the CANopen device is called TPDO or formerly TX PDO.

Each PDO is mapped to a single CAN frame using a CAN-ID determined by the COB-ID parameter. It can hold up to 8 bytes of data and consists of multiple CANopen objects that can be mapped into the PDO. However, only objects that are specified for the usage in PDOs are mappable.

In general, PDOs can be transmitted asynchronously and synchronously. Asynchronous PDOs are triggered in device-internal events. For example, when the process data changes. Synchronous PDOs are transmitted after receiving a SYNC message. It is normally a CAN frame with CAN-ID 0x80 that is periodically sent by the CANopen manager. However, this only shows some basic information about PDOs, there are more parameters that specify the transmission trigger (e.g. triggering the PDO only with every fifth SYNC message).

4.4 Service Data Objects (SDOs)

Service Data Objects (SDOs) can be used to obtain data from the object directory of a CANopen device. It can be used to write (download) or read (upload) the data of an CANopen object in the object directory. SDO transfers are not limited to 8 bytes of data, because they can be sent in segments. Therefore, the amount of data is unlimited. They are used for initialization and parameterization of the device. SDOs use a client-server model for communication. The owner of the accessed object directory acts as a server, while the CANopen device that accesses the object directory of the other device, is the client. The client always requests information and needs to wait for the response (acknowledge) of the server.

4.4.1 Communication Parameters for SDO Transfers

The SDOs are transmitted with CAN-ID '**0x600 + NodeID**' (request). The client acknowledges the parameters with ID '**0x580 + NodeID**' (response). Further information is described in **Table 6** and **Table 7**.

An SDO is structured as follows:

Identifier	Command Code	Index		Sub-index	LSB	Data field		MSB
		(low)	(high)					
0x600+ NodeID	0x23	0x00	0x14	0x01	0x7F	0x04	0x00	0x00
	Write	Index=0x1400		(COB-def.)	Data (here COB-ID) = 0x047F			

Table 6: SDO Communication Parameter 1

Parameter	Description																														
Identifier	The parameters are transmitted with ID '0x600 + NodeID' (request). The receiver acknowledges the parameters with ID '0x580 + NodeID' (response).																														
Command Code	<p>The command code transmitted consists among other things of the Command Specifier and the length.</p> <p>Frequently required combinations are for example: 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</p> <p>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</p> <p>Frequently Used Command Codes</p> <p>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 specification CiA 301, chapter "Service Data Object".</p> <table><tr><th>Command</th><th>Number of data bytes</th><th>Command code</th></tr><tr><td rowspan="4">Write Request (Initiate Domain Download)</td><td>1</td><td>0x2F</td></tr><tr><td>2</td><td>0x2B</td></tr><tr><td>3</td><td>0x27</td></tr><tr><td>4</td><td>0x23</td></tr><tr><td>Write Response (Initiate Domain Download)</td><td></td><td>0x60</td></tr><tr><td>Read Request (Initiate Domain Upload)</td><td></td><td>0x40</td></tr><tr><td rowspan="4">Read Response (Initiate Domain Upload)</td><td>1</td><td>0x4F</td></tr><tr><td>2</td><td>0x4B</td></tr><tr><td>3</td><td>0x47</td></tr><tr><td>4</td><td>0x43</td></tr><tr><td>Error Response (Abort Domain Transfer)</td><td></td><td>0x80</td></tr></table>	Command	Number of data bytes	Command code	Write Request (Initiate Domain Download)	1	0x2F	2	0x2B	3	0x27	4	0x23	Write Response (Initiate Domain Download)		0x60	Read Request (Initiate Domain Upload)		0x40	Read Response (Initiate Domain Upload)	1	0x4F	2	0x4B	3	0x47	4	0x43	Error Response (Abort Domain Transfer)		0x80
Command	Number of data bytes	Command code																													
Write Request (Initiate Domain Download)	1	0x2F																													
	2	0x2B																													
	3	0x27																													
	4	0x23																													
Write Response (Initiate Domain Download)		0x60																													
Read Request (Initiate Domain Upload)		0x40																													
Read Response (Initiate Domain Upload)	1	0x4F																													
	2	0x4B																													
	3	0x47																													
	4	0x43																													
Error Response (Abort Domain Transfer)		0x80																													
Index, Sub-Index	Index and sub-index address the parameters in the object directory.																														
Data field	<p>The data field has a maximum size of 4 bytes and is always structured according to the principle 'LSB first, MSB last'.</p> <p>The least significant byte is always in 'Data 1'.</p> <p>For 16-bit values the most significant byte (bits 8 ... 15) is always in 'Data 2', and for 32-bit values the MSB (bits 24 ... 31) is always in 'Data 4'. For larger SDO transfers there are also segmented transfers, which are not covered in this manual.</p>																														

Table 7: SDO Communication Parameter 2

4.4.2 Error Codes of a SDO Transfer

The following error codes might occur (according to CiA 301 (5), 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 too long
0x06070013	SDO_PARA_TO_SHORT	Service parameter is too 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 cannot be executed because of the operating state

Table 8: CANopen SDO Error Codes

4.5 Network Management (NMT)

Each CANopen device must support a CANopen NMT state machine, which consists of the states INIT, PRE-OPERATIONAL, OPERATIONAL and STOPPED.

After power-on each device starts in the INIT state. When it has finished its initialization, it enters the state PRE-OPERATIONAL and indicates that it is ready by transmitting a boot-up message. Further state changes can be triggered by the CANopen manager or by device-intern events like an error. Only during the state OPERATIONAL PDOs are exchanged.

The CANopen-PN/2 acts as a CANopen manager and configures all CANopen devices. After this it uses a 2-byte CAN frame with the data 0x01 and 0x00 on the CAN-ID 0x0000 to start all CANopen devices.

Each NMT state is represented by the following value:

NMT State	Value
INIT	-
STOPPED	0x04
OPERATIONAL	0x05
PRE-OPERATIONAL	0x7F
UNKNOWN*	0x00

*UNKNOWN is not an official CANopen NMT state but is used from the CANopen-PN/2 to define when the CANopen-PN/2 is unable to detect the NMT state of a CANopen device.

Table 9: CANopen NMT States

4.6 Node Guarding and Heartbeat

The Node Guarding and Heartbeat protocols can be used to check the availability of another CANopen device.

Node Guarding Protocol

Guarding is an outdated method to check whether the guarded CANopen device is still available on the network and has the correct NMT state. The NMT manager triggers Remote-Request-Frames (RTR) to a specific CANopen device. The requested device will answer with its respective NMT state. However, if possible, use the heartbeat protocol.

Heartbeat Protocol

The heartbeat protocol consists of a producer-consumer-model. Each CANopen device configured as a heartbeat provider cyclically transmits its current NMT state in a configured interval. Each CANopen device that is configured as a heartbeat consumer for this specific CANopen device checks if it has received a heartbeat message within its configured interval.



NOTICE

The heartbeat producer interval should always be sufficiently greater than the heartbeat consumer interval, because otherwise heartbeat errors might occur, only due to the jitter at transmission/reception.

4.7 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 *	Start (PRE-OPERATIONAL -> OPERATIONAL)
0	NMT	2	0x80 *	OPERATIONAL -> PRE-OPERATIONAL
0	NMT	2	0x81 *	Reset
0	NMT	2	0x82 *	Reset Communication
0x80	SYNC	0	-	Sync Message
0x80 + Node ID	EMCY	0 ... 8 Bytes	Error code	Emergency Message

* Stands for Node ID of a CANopen module or '00' for a message to all CANopen devices

Table 10: Important CANopen Telegrams

5 Software

This chapter describes the installation, configuration, and the functionality of the gateway.

**NOTICE**

It is highly recommended to have some basic knowledge about the CANopen protocol. For some basic information please read chapter 4. For further information please read the CANopen specification CiA 301 (5).

5.1 Functionality

The firmware of the gateway connects a CANopen network and a PROFINET IO network. To achieve this, the CANopen network is replicated with the GSDML-Composer. The GSDML file that arises from the GSDML-Composer can be included in the PROFINET development environment. It includes all necessary information to configure, control and start the CANopen network as well as data exchange between the PROFINET IO and CANopen PDOs.

After establishing a connection between the CANopen-PN/2 and a PROFINET controller, all CANopen devices that are configured in the CANopen network are parametrized and started. The CANopen-PN/2 gateway also executes the NMT functionality and starts the heartbeat or node guarding protocol.

After a CANopen device changes to the NMT state OPERATIONAL, PDOs and PROFINET process data are exchanged. The PDOs are directly mapped into the PLC address space.

If a CANopen device or the CAN bus is non-functional or faulty, diagnostics alarms are sent via PROFINET to the PLC.

5.2 Licenses

**NOTICE**

The software used for the CANopen-PN/2 from esd and from third parties is subject to licenses. You must read and accept these license conditions before the installation!

The license terms of esd (esd electronics License Conditions) and of 3rd parties (3rd Party Licenses) are displayed and installed on your system during installation via the installation program (CANopen-PN_2_X_X_X.exe, see chapter 5.3).

You can also see also chapter Software Licenses from page 117 for further information.

5.3 Installation

The CANopen-PN/2 includes an installer called `CANopen-PN_2_X_X_X.exe`. The installation of the installer is mandatory to configure the CANopen network, applying firmware updates and monitor the CAN bus.

This installer provides the following packages:

GSDML-Composer	This package includes the GSDML-Composer, which is necessary to configure the CANopen-PN/2 gateway.
RNDIS Driver	This package contains the RNDIS driver. It is used to connect the Mini-USB interface to a Windows® computer. The connection is used for firmware updates and CAN monitoring. The driver is installed automatically.
CAN Driver	This package provides the esd CAN-API ("NTCAN"). It is necessary for the esd CAN-SDK.
esd CAN-SDK	This package contains software for the CAN monitoring and diagnostics, especially the monitoring tool <i>CANreal</i> , which can be used to detect and send CAN frames on the bus.
Example	This package includes a TIA Portal® project with some examples.



NOTICE

The CAN driver and the CAN-SDK are not automatically deleted if the CANopen-PN/2 software is removed. Therefore use "Software" of the Windows system administration and remove "EtherCAN [...] Host Driver" and "CAN SDK for Windows".

5.3.1 Manual Installation of the RNDIS Driver

The RNDIS driver is installed automatically with the installer. To check whether the RNDIS driver is installed correctly, connect the Mini-USB interface with the computer. When the installation has been successful, a new network adapter called *RNDIS based ESD Device* should be displayed in the *Device Manager*. If this is not the case, install the driver manually with the following steps:

Step	Action
1.	Connect the CANopen-PN/2 with the computer using the Mini-USB interface.
2.	Open the <i>Device manager</i> and search for a new <i>Serial USB-Device (COMX)</i> under <i>Ports (COM & LPT)</i> .
3.	Right click on the device and select update driver. In the following dialog select <i>Update driver</i> .
4.	Click <i>Browse my computer for drivers</i> .
4.	Select the installation path of the CANopen-PN/2 and check the checkbox <i>Include subfolders</i> . By default, this installation path is C:\Program Files (x86)\esd\CANopen-PN.
5.	Press <i>Next</i> .
6.	Now there should be a network adapter called <i>RNDIS based ESD Device</i> .

Table 11: Manual installation of the RNDIS driver

5.3.2 Example Project for TIA Portal

The installer comes with an example project for the TIA Portal. This shows some examples how to exchange data via SDO or set the NMT state of a CANopen device.

5.4 Configuration

This chapter describes the steps which are relevant to configure the CANopen-PN/2. The steps are shown with the Siemens TIA Portal as development environment. For further information about your development environment or the TIA Portal, please read the respective documentation.

5.4.1 Quick Start Guide

Step	Action	See Page
1	Disconnect the online connection in the TIA Portal, because the hardware and software must be compiled in offline mode.	-
2	Configure the CANopen network with the GSDML Composer as described in chapter 5.5.	37
2	Change into the project view of the TIA Portal.	-
3	Install the GSDML file as described in chapter 5.4.3.	30
4	Insert the CANopen-PN/2 in your project as described in chapter 5.4.4.	31
5	Configure the PROFINET interface as described in chapters 5.4.5 and 5.4.6.	32, 33
6	Compile and load the hardware and software as described in chapter 5.4.7.	35
7	Go online as described in chapter 5.4.7.	35

Table 12: Configuration - Quick Start Guide

5.4.2 Configuration of the CANopen Network

This step is done by the GSDML Composer. All needed information is provided in chapter 5.5.

5.4.3 Installation of the GSDML File

To use the GSDML file, it has to be installed into the development environment. To achieve this, switch to the project view in the program window of your TIA Portal. Click on *Options* in the taskbar and select *Manage general station description files (GSD)*.

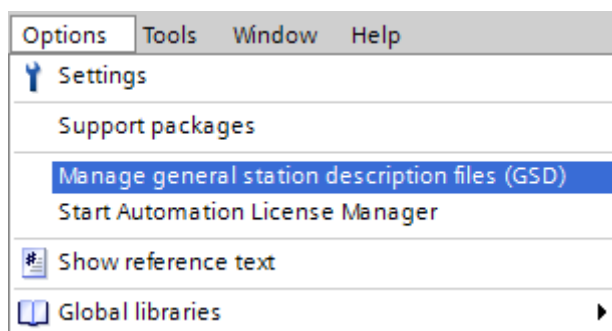


Figure 7: Manage GSDML files

A new dialog appears, in which the path to the directory of the GSDML file must be entered. Choose the export directory of the GSDML project, select the GSDML file and press *Install*.

5.4.4 Insert the CANopen-PN/2

After the installation of the GSDML file, the PROFINET network can be assembled. Therefore, click under *Project tree* → *Devices* onto *Devices & networks* as shown in the following figure. The so-called *Network view* opens. The CANopen-PN/2 can now be added from the *Hardware catalog*. The device can be found under *Other field devices* → *Gateway* → *esd electronics gmbh* → *CANopen/PROFINET-IO* → *CANopen-PN/2*. To insert it, drag it onto the *Network view*.

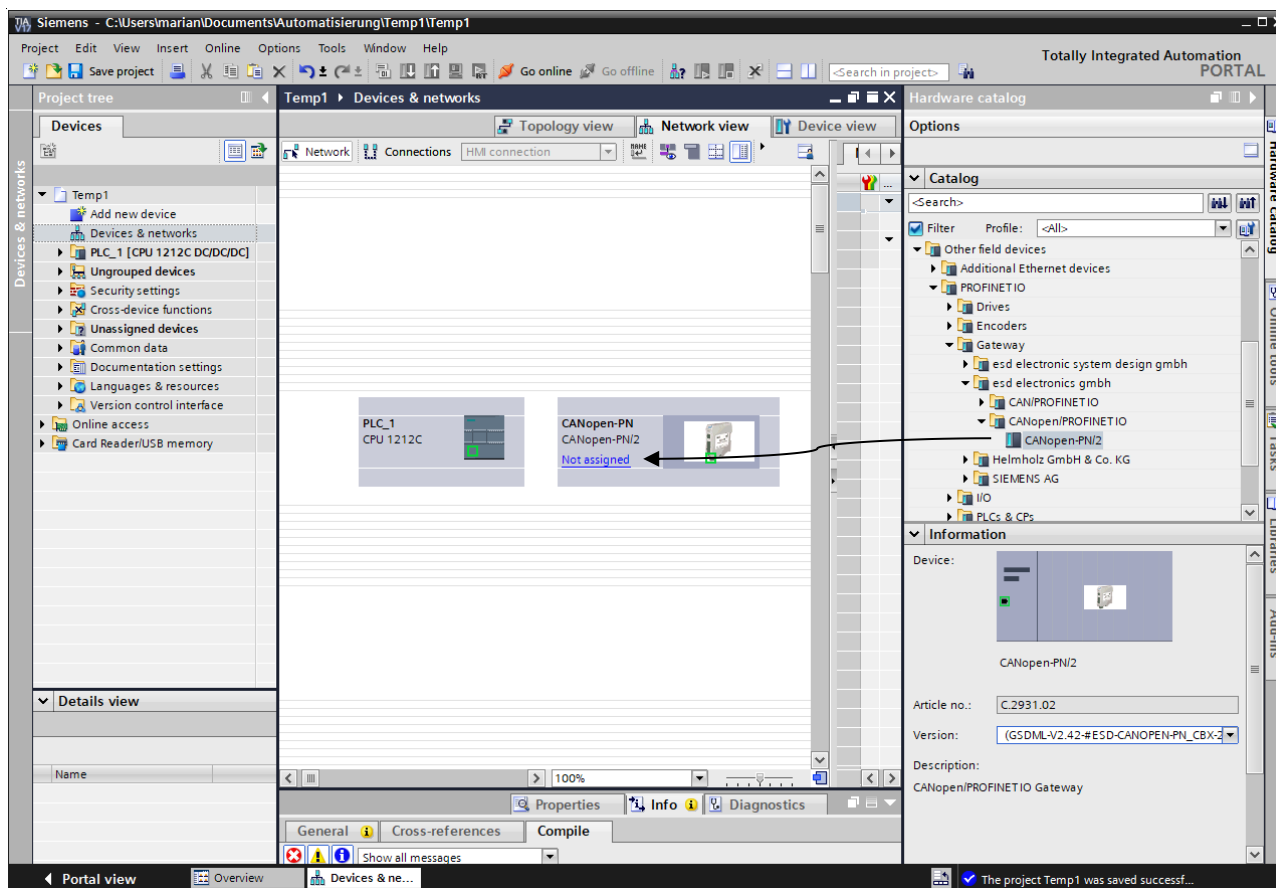


Figure 8: Inserting the CANopen-PN/2



NOTICE

Please note, that it is absolutely necessary that you select the correct GSDML file for your project in the *Information* section of the *Hardware catalog*. By clicking on the input field of *Version*: a list of all available GSDML files is shown (see Figure 9). In the name of the GSDML file the name of the project as well as the date and the time of its creation are encoded. With each new GSDML file installation one entry will be added to the list.

Example:

GSDSML-V2.42-#ESD-CANOPEN_PN_CBX-20220605-110401.xml

GSDML-V2.42	#ESD-CANOPEN_PN	CBX	20220605	110401	.xml
GSDML Version	Product Prefix	Project Name	Date	Time of Creation	File Type

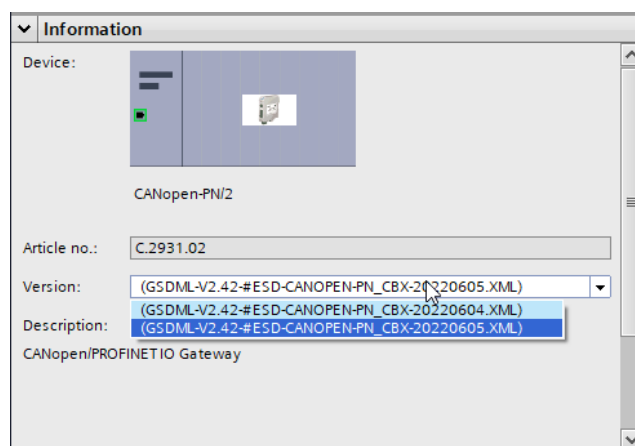


Figure 9: Choose GSDML file of the CANopen-PN/2

5.4.5 Assign the PROFINET Network

First the CANopen-PN/2 must be assigned to a PROFINET network. To accomplish this, go to the *Network view*, press the button Not assigned and click on one of the available PROFINET networks.

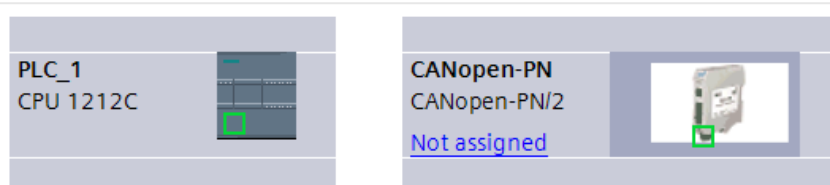


Figure 10: Not assigned CANopen-PN/2

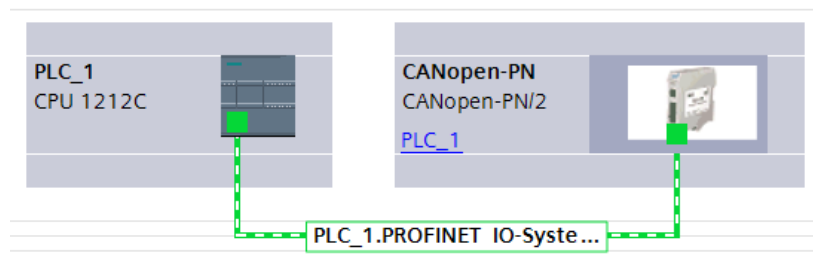


Figure 11: Assigned CANopen-PN/2

5.4.6 Assign IP Address and PROFINET Device Name

In order to work properly the IP address and the PROFINET device name of the configuration must match those persistently saved on the CANopen-PN/2 gateway. Both can be configured separately.



NOTICE

Each IP address and PROFINET device name can only be assigned once per PROFINET network. The IP address normally does not have to be changed manually.

IP Address and Device Name of the Configuration

The IP address and device name of the configuration are generated automatically by default. However, it can be changed manually.

To change it, click on the tab *Device view* of the gateway and select CANopen-PN in *Slot 0* of the section *Device overview*. Now open the tabs *Properties* → *General* → *Ethernet addresses* and search for the parameters *IP address* and *PROFINET device name*.

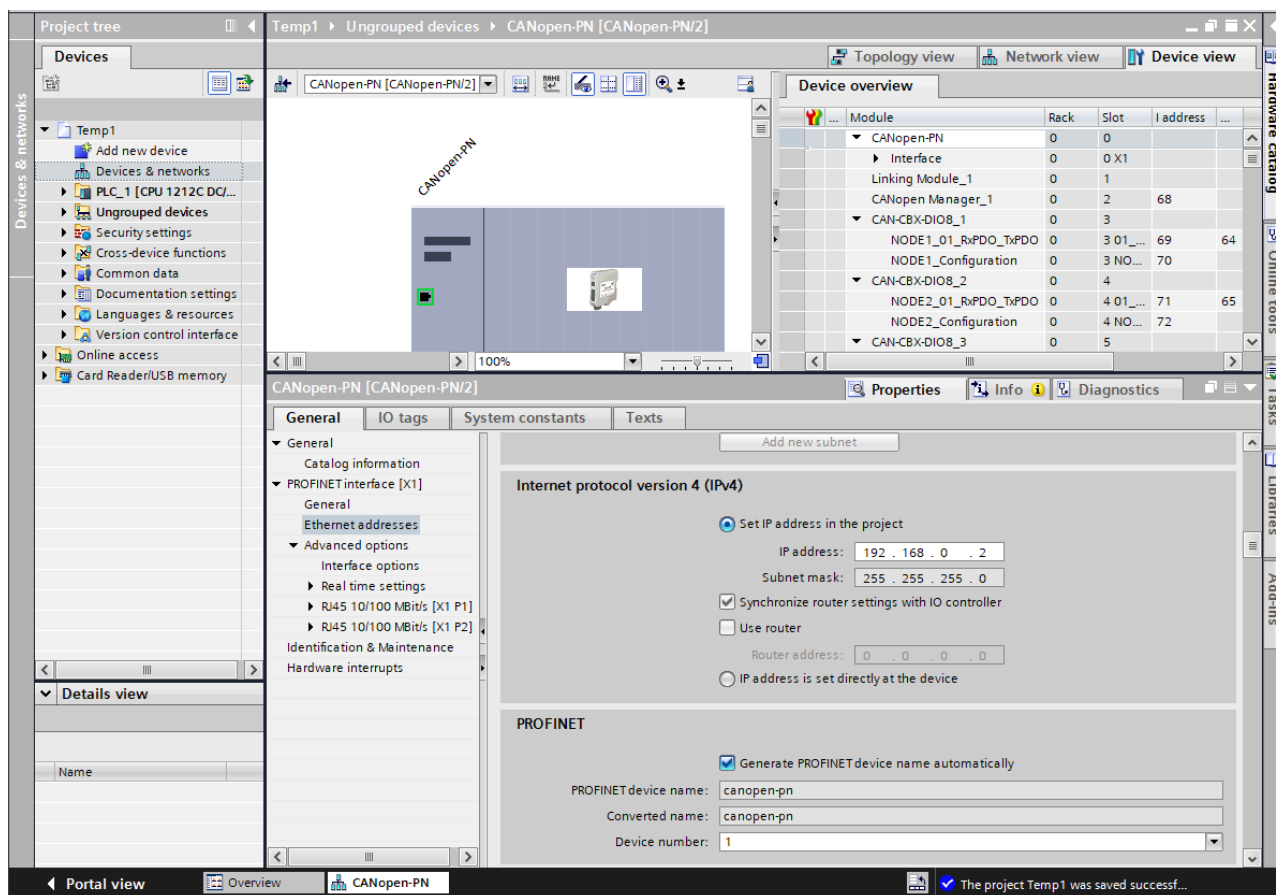


Figure 12: Assign IP address and device name of the configuration

IP Address and Device Name of the Gateway

The current IP address and device name of the gateway can be found in the *Online access* section. To display it, click *Update accessible devices* under *Project tree* → *Devices* → *Online Access* → [Network adapter]. The name as well as the IP address should be displayed.

To change the IP address or the name, expand the device by clicking on the icon and open *Online & diagnostics*. A new dialog will appear as shown in the figure below. Expand *Functions* and select *Assign IP address* or *Assign PROFINET device name*. Insert the new parameter accordingly and press the *Assign IP address* or *Assign name* button.

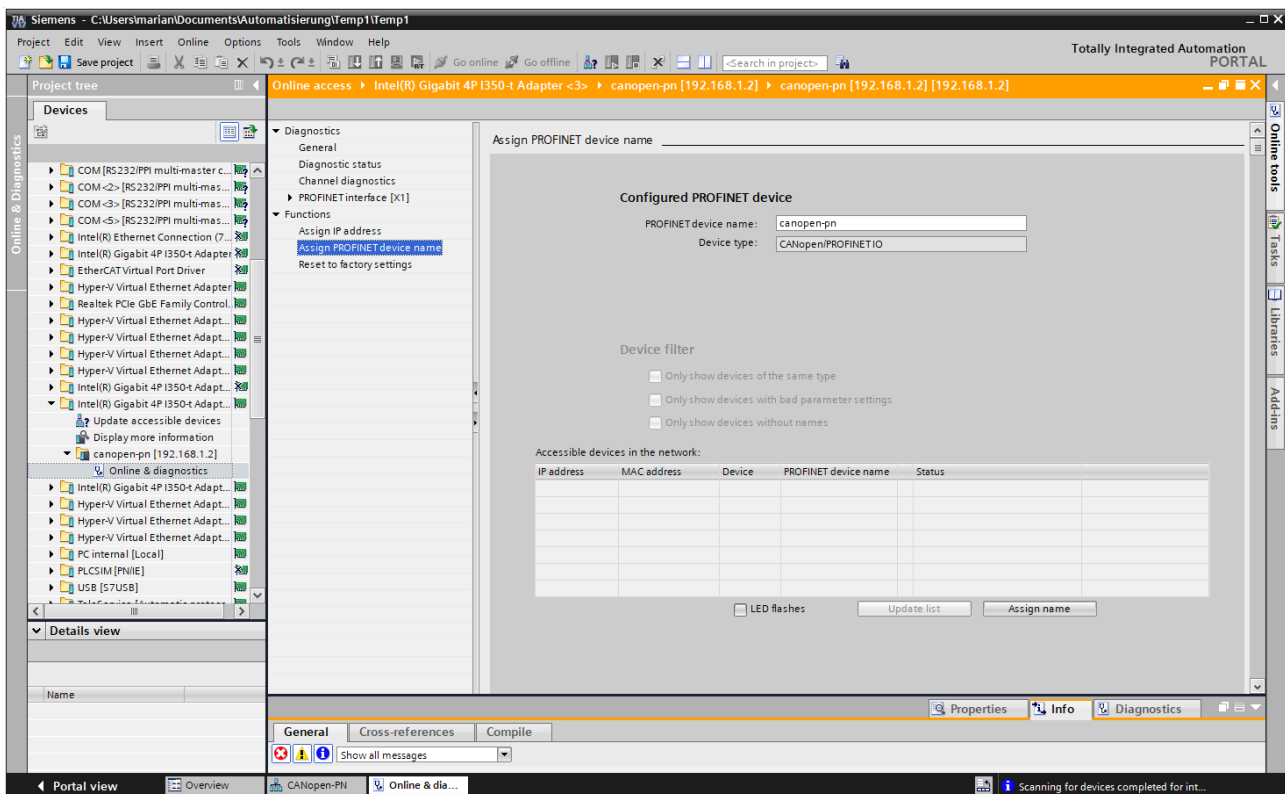


Figure 13: Assign IP address and device name of the gateway

5.4.7 Compile and Download Hardware and Software

Before the software can be download, it must be compiled.

During this process the TIA Portal must be in offline mode!

To compile the software, select the device (PLC_1 in this case) in the field *Project tree* → *Devices* and click *Compile* → *Hardware and software (only changes)* in the pull-down menu.

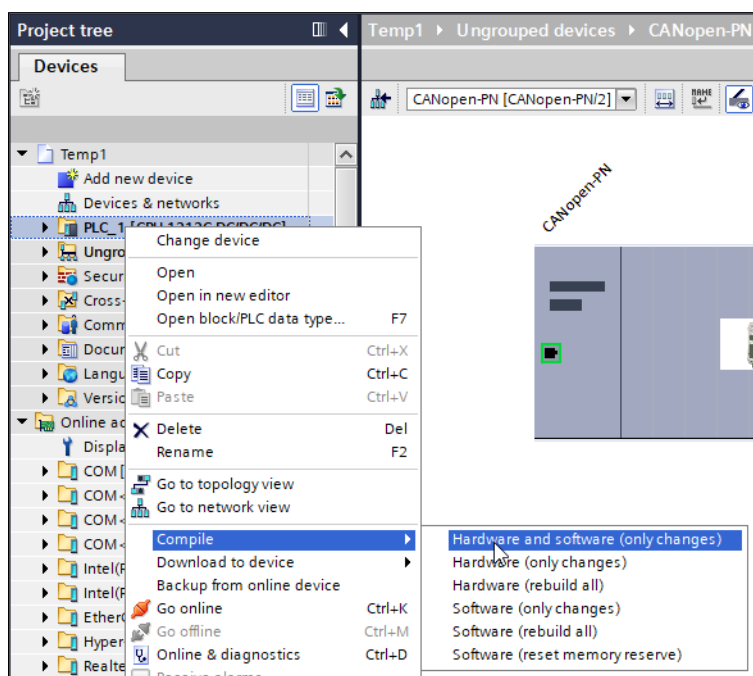


Figure 14: Compile hardware and software (detail)

The configuration is compiled. After this the hard- and software can be downloaded to the device. Select your device (PLC_1 in this case) again and click *Download* → *Hardware and Software* in the pull-down menu. A new dialog opens in which the PLC can be chosen.

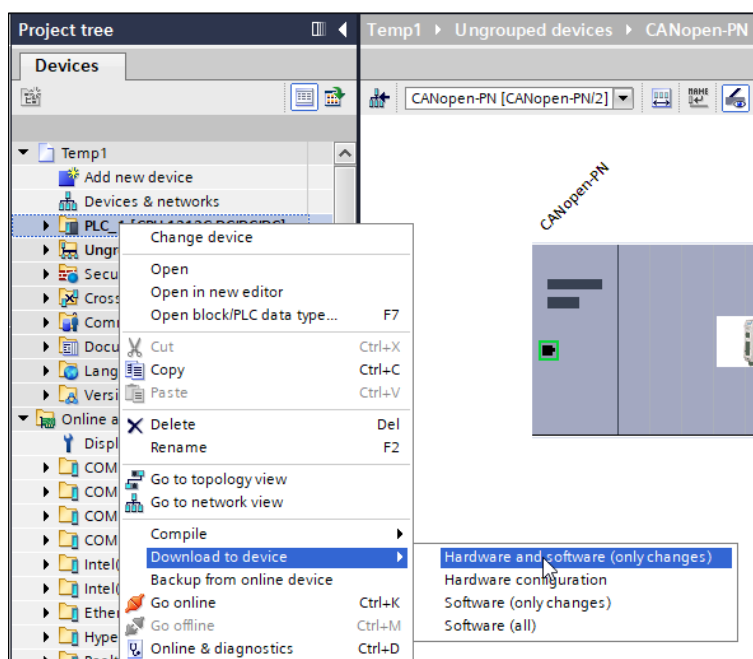


Figure 15: Download hardware and software to device (detail)

Software

The configuration is now successfully passed to the device. Click on the button *Go online* in the toolbar to go online.

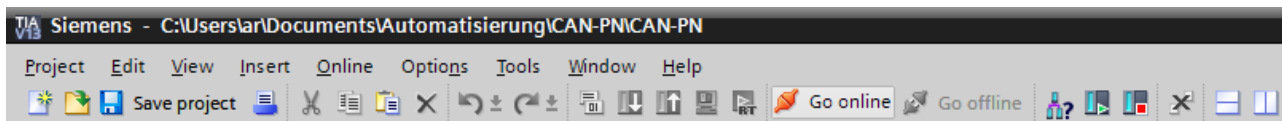


Figure 16: Toolbar with button *Go online*

The online connection is now established.

To check whether the device is working properly, open the *Device View* and see if all check marks are green (see Figure below). If this is not the case, something is wrong. Please read chapters 5.7 and 9 for further information.

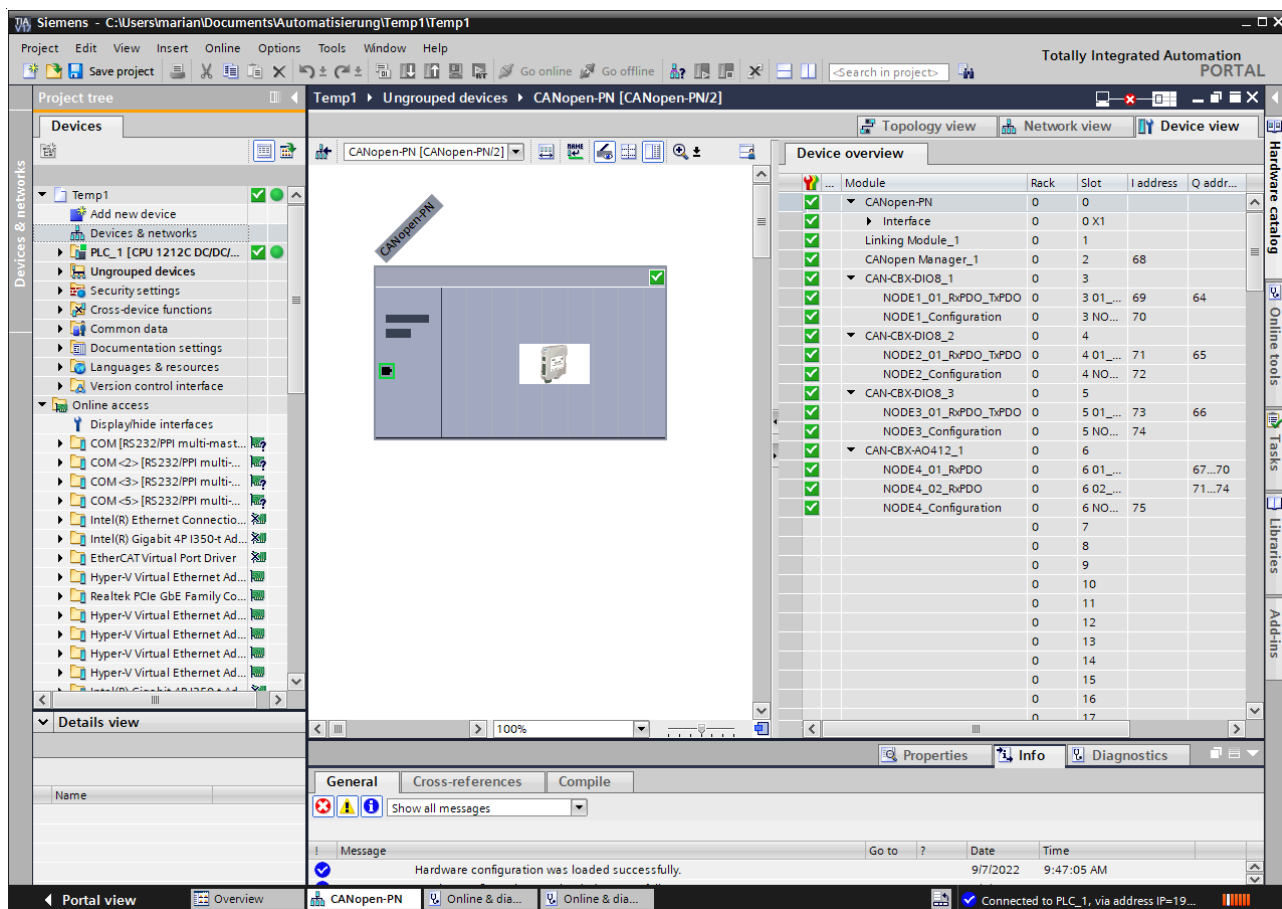


Figure 17: Online *Device overview*

5.5 GSDML Composer

5.5.1 Quick Start Guide

Step	Action	See Page
1	Open the GSDML Composer e.g. via the Windows start menu.	-
2	Import the EDS files of the CANopen devices, that shall be used, into the <i>Device Library</i> (see chapter 5.5.9).	43
3	Insert the selected devices in the <i>CANopen Network Editor</i> (see chapter 5.5.10).	45
4	Configure the CANopen manager (see chapter 5.5.11).	47
5	Configure the CANopen devices (see chapter 5.5.12).	50
6	Save the project and export the GSDML file (see chapter 5.5.8.1).	41

Table 13: GSDML Composer Quick Start Guide

5.5.2 Description

The GSDML Composer is designed to generate and parametrize a GSDML file for the CANopen-PN/2. It can be used to represent the entire CANopen network, configure the PDO mapping and the heartbeat/node guarding.

The CANopen-PN/2 itself always acts as a CANopen manager.

Each CANopen device comes with an EDS file, which can be included into the 'Device Library' of the GSDML Composer. After this, the file can be used to add a CANopen device to the network. When the whole CANopen network is configured, a GSDML file can be generated. This file contains all CANopen information for the CANopen-PN/2. The data cannot be changed with the development environment e.g. the Siemens TIA Portal.

The settings of the PROFINET interface are not set by the GSDML Composer and must be set in the development environment.

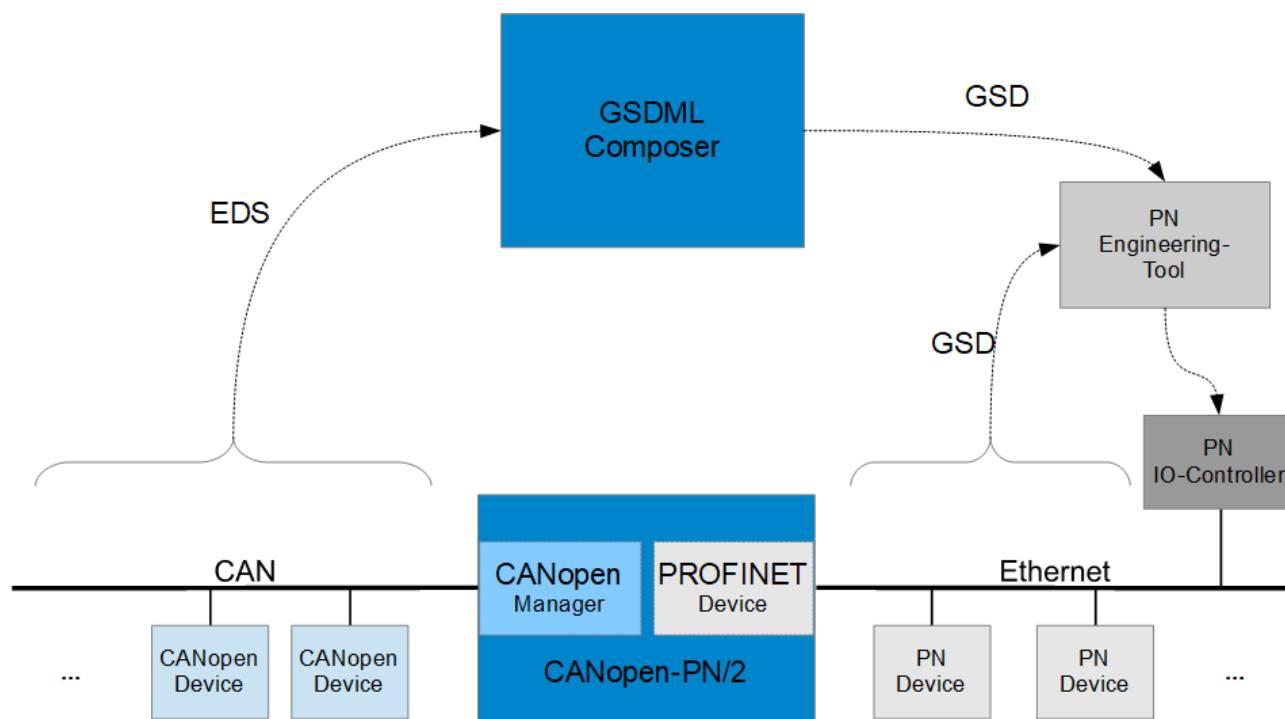


Figure 18: GSDML Composer diagram

5.5.3 Features

- Windows application with intuitive operation
- *CANopen Device Library* based on EDS files
- *CANopen Network Editor* for graphic overview and configuration of the CANopen network
- *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 device
- Export of the GSDML file containing the complete configuration, for the usage of the esd CANopen-PN/2 gateway with the corresponding CANopen net in PROFINET IO

5.5.4 System Requirements

Operating System	Microsoft Windows XP or newer with Microsoft .NET Framework Version 3.5
Memory Space	Approx. 16 MB (CAN Tools approx. 60 MB)

5.5.5 Compatibility

The GSDML Composer can be used to configure the following devices:

- CANopen-PN (C.2921.02)
- CANopen-PN/2 (C.2931.02)

All GSDML Composer Versions above V.1.0.6.7 are able to configure both devices. At the start there is a dialog (see Figure 19) in which the gateway that should be configured has to be selected.



Figure 19: GSDML Composer Gateway Selection

The CANopen-PN (C.2921.02) is the predecessor of the CANopen-PN/2 (C.2931.02). It can only be configured with a GSDML file, which was exported by a GSDML Composer project in which the CANopen-PN was selected or the GSDML Composer version was V.1.0.6.7 or lower. GSDML files for the CANopen-PN start with **GSDML-v2 . 3**. However, this GSDML file can also be used to configure the CANopen-PN/2 for compatibility reason. The GSDML files may not support all new features of the CANopen-PN/2 (see chapter 8).

The CANopen-PN/2 (C.2931.02) GSDML files start with **GSDML-v2 . 42** and can only be used to configure the CANopen-PN/2. They do not work with the CANopen-PN. To update a project from the CANopen-PN to the CANopen-PN/2 and vice versa select the CANopen-PN/2 project at start-up and load an existing project for the CANopen-PN. Save the project again and export the GSDML file.

5.5.6 Installation

The GSDML Composer is installed with the installer described in chapter 5.3.

5.5.7 Overview

The following figure shows the main window of the program.

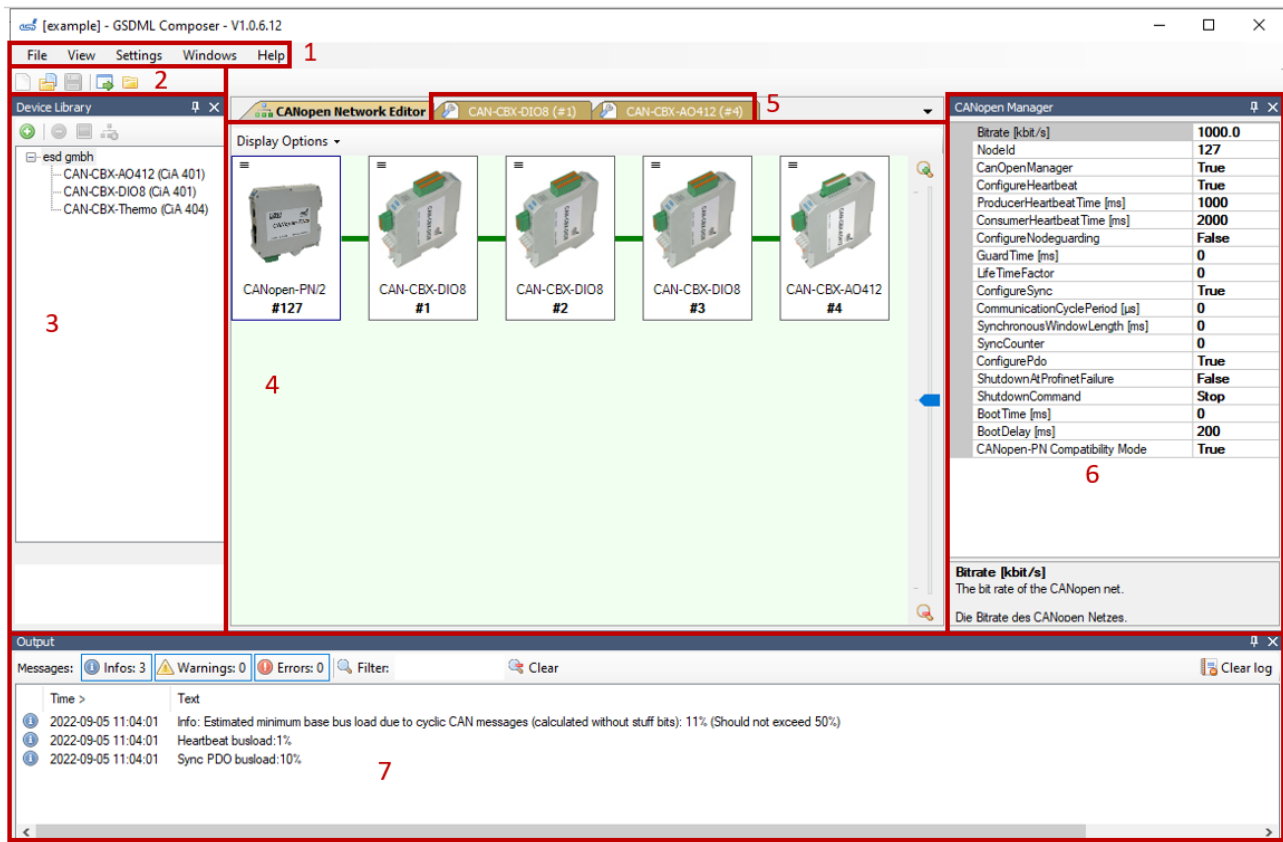


Figure 20: GSDML Composer Main Window

Legend

1. Menu bar (see chapter 5.5.8)
2. Toolbar
3. Device Library (see chapter 5.5.9)
4. CANopen Network Editor (see chapter 5.5.10)
5. Device Editor in background (see chapter 5.5.12)
6. CANopen Manager (see chapter 5.5.11)
7. Output (see chapter 5.5.13)

5.5.8 Menu Bar

The menu bar contains the following menu items:

- *File*
- *View*
- *Settings*
- *Window*
- *Help*

5.5.8.1 File

The menu item *File* can be used to start, open, or save a GSDML project (.xgcp) or to export a DCF or a GSDML file. It can also be used to switch between the CANopen-PN and CANopen-PN/2 or to close the application. Some of its functions are also accessible via icons in the toolbar (see Figure 20, 2).

The following table show all commands in detail:









Command	Symbol	Description
<i>New project</i>		Starts a new empty GSDML project with default values. The current project will be closed.
<i>Load project</i>		Loads an existing GSDML project.
<i>Save project</i>		Saves the current project into a GSDML project file. At the initial saving the file name and path have to be specified. The project file will have the extension .xgcp and holds the complete project data including the EDS information of the CANopen device – it can thus be opened in the GSDML Composer without importing the EDS files again.
<i>Save project as...</i>		Same as <i>Save project</i> , but the file name and path must always be specified in this command.
<i>Restart with gateway swap</i>		Restarts the program with the other gateway type.
<i>Export DCF</i>		Generates the DCF files for the current project and store them in the export directory of the project. If the export directory has not already been specified, the <i>Project settings</i> window (see Figure 21) will be opened automatically.
<i>Export GSDML</i>		Generates the GSDML file for the current project and store it in the export directory of the project. If the export directory has not already been specified, the <i>Project settings</i> window (see Figure 21) will be opened automatically.
<i>Open export directory</i>		Opens the export directory in the Windows Explorer. If the export directory has not already been specified, the project settings window (see Figure 21) will be opened automatically.
<i>Quit</i>		Quits the application.

Table 14: GSDML Composer *File* Parameter

5.5.8.2 View

The menu item *View* offers function to organize the menus of the application.

The following table show all commands in detail:







Command	Symbol	Description
<i>CANopen Network Editor</i>		Opens and move the <i>CANopen Network Editor</i> into foreground.
<i>CANopen Manager</i>		Opens and move the <i>CANopen Manager</i> into foreground.
<i>Device Library</i>		Opens and move the <i>Device Library</i> into foreground.
<i>Output</i>		Opens and move the <i>Output</i> into foreground.
<i>Close all device editors</i>		Closes all open device editors.
<i>Close current device editor window</i>		Closes the current device editor.

Table 15: GSDML Composer *View* Parameter

5.5.8.3 Settings

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

Command	Description
<i>Project settings</i>	Opens a dialog where the export directory can be chosen (see Figure 21).
<i>Disable window docking</i>	Changes the language of the GSDML Composer. The change will only take effect after an application restart.
<i>Language</i>	Opens and moves the <i>Device Library</i> into foreground.
<i>GSDML</i>	This option only contains the checkbox <i>File name with time</i> . It defines whether the name of the GSDML file will contain the time, when exported. This option is helpful especially in the development phase, because the file version of the GSDML file can only be distinguished by date and time.

Table 16: GSDML Composer *Settings* Parameter

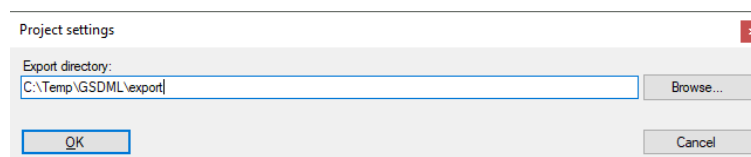


Figure 21: GSDML Composer *Project Settings*

5.5.8.4 Window

This menu item shows a selection of all available windows/tabs (*Device Editor* and the *CANopen Network Editor*). By clicking on an option, the window will be set to foreground.

5.5.8.5 Help

The menu item *Help* offers a link to this manual as well as some additional information.

5.5.9 Device Library

The *Device Library* offers an overview of all installed EDS files sorted by vendor. It is located at the left side of the main window (see Figure 20, 3).

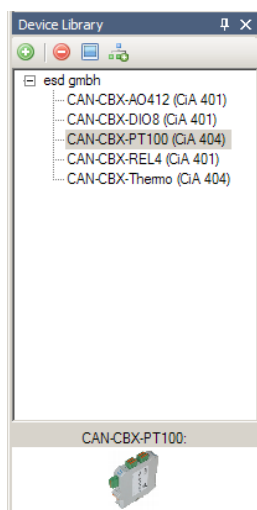


Figure 22: GSDML Composer *Device Library*

The *Device Library* has a toolbar, that has the following options:






Command	Sym bol	Description
<i>Import new device into library</i>		Open a dialog for the selection of the EDS files. The selected files are automatically copied into the <i>Device Library</i> . <div> NOTICE The <i>Device Library</i> is located in the installation path of the CANopen-PN/2. Do not edit it manually.</div>
<i>Delete from device library</i>		Delete the selected device from the library.
<i>Change default image</i>		Shows a dialog 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 test.eds it will be searched for test.jpg, test.png, etc..
<i>Append to network editor</i>		Adds the selected CANopen device to the current project. The device will then be shown in the window of the <i>CANopen Network Editor</i> .

Table 17: GSDML Composer *Device Library* Parameter



NOTICE

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.

Add CANopen Devices to the current Project

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

- Double click with the left mouse button on an entry of the *Device Library*. The chosen device will be appended to the CANopen network as last device.
- 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 network as last device.
- '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*. When the mouse is released in an empty area, the chosen device is appended to the CANopen network as the last device. When the mouse is released on an existing device, it is inserted before the existing device. The sequence is only there for clarity and has no influence on the functionality.

After a device has been inserted, a dialog window for the input of the name and the CANopen node ID opens. The values can be changed afterwards (see chapter 5.5.12.1).

5.5.10 CANopen Network Editor

In the *CANopen Network Editor* displays the CANopen-PN/2 as well as all configured CANopen devices with their CANopen node IDs. It is located at the middle of the main window (see Figure 20, 4) on the tab *CANopen Network Editor*.

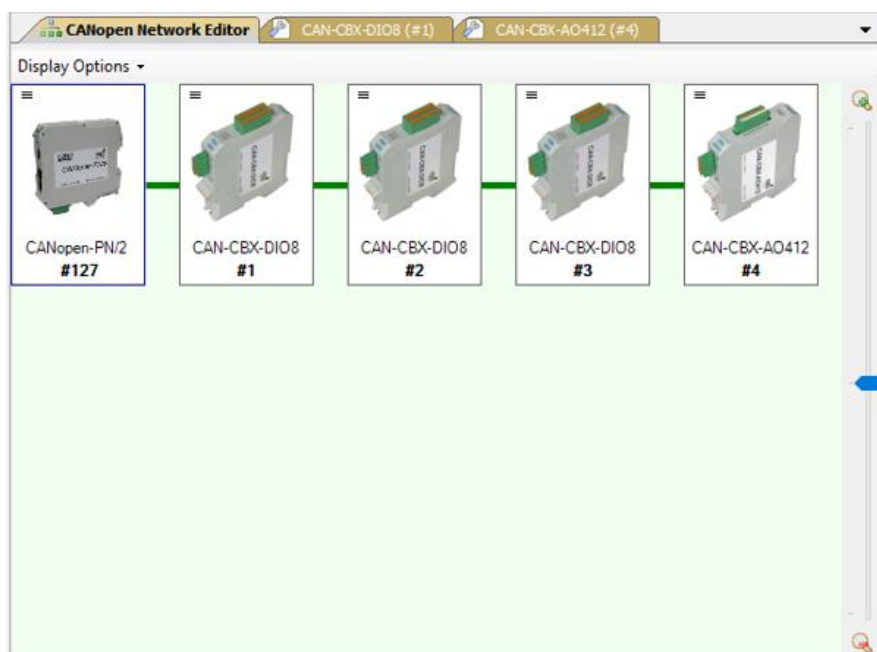


Figure 23: GSDML Composer *CANopen Network Editor*

On the right side there is a bar 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* (chapter 5.5.12) of the respective device.

A single click opens a small information window which contains the name and node ID of the device.

The order of the devices, i.e. the way of the connection line, is only used for clarity – it has no effect on the exported GSDML file. By default, all devices are show in a line and arranged automatically. However, there are multiple *Display Options* that can be used to rearrange the devices (see Table 18).

Furthermore, every device has its own context menu which can be entered by right clicking the device (see Table 19).

Display Options

The following options are available:

Option	Description
<i>Calculate positions automatically</i>	Defines whether the devices are rearranged in the <i>CANopen Network Editor</i> automatically if a device is removed or inserted. If the function is disabled, the devices can be arranged via the mouse in the <i>CANopen Network Editor</i> window in any user-defined order.
<i>Recalculate positions now</i>	Recalculates the position of the devices in the <i>CANopen Network Editor</i> window once. If <i>Calculate position automatically</i> is enabled, this function will be called automatically when a device is removed or inserted.
<i>Reassign Node IDs (by display order)</i>	Changes the node IDs of the devices according to the displayed order. ID '1' is assigned to the first CANopen device and so on.
<i>Reset display order (by Node IDs)</i>	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 and so on.

Table 18: GSDML Composer Display Options

Device Context Menu

The context menu of each device can be entered by right-clicking it. It provides the following options:

Option	Description
<i>Open device editor</i>	Opens the <i>Device Editor</i> (see chapter 5.5.12) for selected device.
<i>Display order</i>	Changes the index of the device in the display order, i.e. if <i>Calculate position automatically</i> is enabled, it interchanges its display position with the preceding/following device (move up / move down). If <i>Calculate position automatically</i> is disabled, only the connection line is affected – the positions of the device images in the <i>CANopen Network Editor</i> window remain unaffected.
<i>Clone</i>	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 dialog allowing a quick change is opened.
<i>Delete</i>	Deletes the selected device.
<i>Change device image</i>	Shows a dialog for the selection of an image file and changes the image of the device.

Table 19: GSDML Composer Device Context Menu

5.5.11 CANopen Manager

This menu is displayed on the right side of the main window (see Figure 20, 6). It contains the settings for the CANopen manager as well as general settings for the CAN interface.

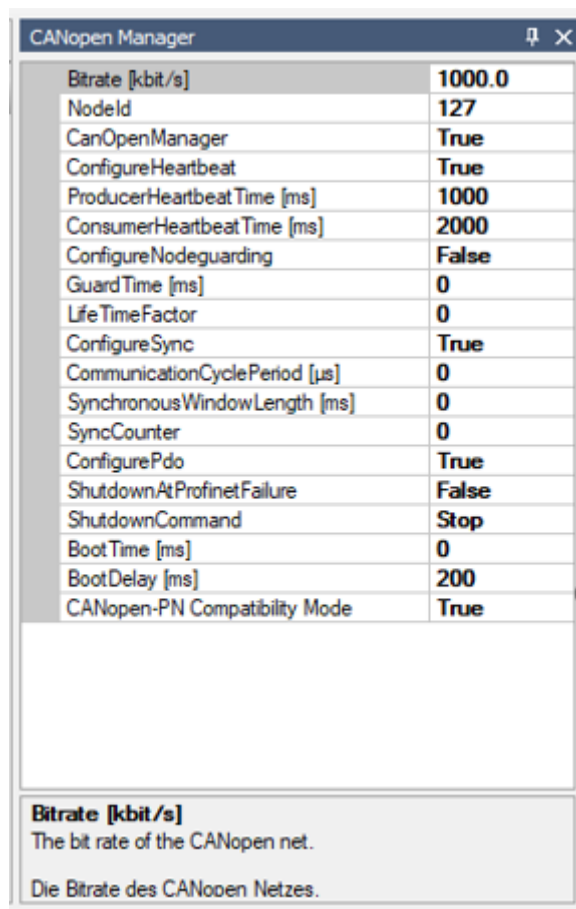



Figure 24: GSDML Composer *CANopen Manager*

The following parameters can be configured:

Option	Description
<i>Bitrate</i>	Bit rate of the CANopen network in kbit/s.
<i>NodeId</i>	CANopen node ID of the gateway itself.
<i>CanOpenManager</i>	<p>Defines whether the CANopen-PN/2 works as CANopen manager. The default value is True (Gateway is CANopen manager).</p> <p>False can only be used if several CANopen-PN(/2) devices are used simultaneously. In this case the following parameters are ignored:</p> <ul style="list-style-type: none"> <i>ConfigureHeartbeat</i> <i>ConfigureSync</i> <i>ConfigurePdo</i> <i>ConfigureNodeguarding</i> <i>ShutdownAtProfinetFailure</i> <i>ShutdownCommand</i> <i>BootTime</i> <i>BootDelay</i>

<i>ConfigureHeartbeat</i>	Defines whether the heartbeat objects of the CANopen devices are written. If the value is False , the heartbeat settings defined via the <i>Device Editor</i> will be ignored and the default configuration will be used.										
<i>ProducerHeartbeatTime</i>	The interval in milliseconds in which the CANopen manager generates the heartbeat messages. ('0' to disable).										
<i>ConsumerHeartbeatTime</i>	The interval in milliseconds in which the CANopen devices, configured as heartbeat consumers, expect the heartbeat messages from the CANopen-PN/2. This value must be sufficiently greater than the interval of the producer, because otherwise heartbeat errors might occur, only due to the jitter at transmission/reception ('0' to disable).										
<i>ConfigureNodeguarding</i>	Defines whether the node guarding objects of the CANopen devices are written. If the value is False , the node guarding settings defined via the <i>Device Editor</i> is ignored and the default configuration is used.										
<i>GuardTime</i>	This parameter is currently not evaluated.										
<i>LifeTimeFactor</i>	This parameter is currently not evaluated.										
<i>ConfigureSync</i>	Defines whether the SYNC objects of the CANopen devices are written. If the value is False , the SYNC-settings defined via the <i>Device Editor</i> are ignored and the default configuration is used.										
<i>CommunicationCyclePeriod</i>	In this interval in microseconds the CANopen-PN/2 generates SYNC messages ('0' to disable). <div style="border: 1px solid black; padding: 10px; margin: 10px 0;">  NOTICE Please note that unlike other times which have to be specified here, the <i>CommunicationCyclePeriod</i> has to be entered in µs! </div>										
<i>SynchronousWindowLength</i>	This parameter is currently not evaluated.										
<i>SyncCounter</i>	This parameter is currently not evaluated.										
<i>ConfigurePdo</i>	Determines whether the PDO objects of the CANopen devices are written. If the value is False , the PDO settings defined via the <i>Device Editor</i> are ignored and the default configuration is used.										
<i>ShutdownAtProfinetFailure</i>	Defines whether the CANopen manager of the CANopen network shuts down in case of a failure on the PROFINET IO side (connection terminated or no connection to PROFINET IO Controller). See also <i>ShutdownCommand</i> below.										
<i>ShutdownCommand</i>	Specifies which command is executed by the CANopen manager in case of <i>ShutdownAtProfinet</i> Failure. (See also CiA 301 [2]) <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Start</td><td>Service start remote node is executed</td></tr> <tr> <td>Stop</td><td>Service stop remote node is executed</td></tr> <tr> <td>PreOp</td><td>Service enter pre-operational is executed</td></tr> <tr> <td>Reset</td><td>Service reset node is executed</td></tr> <tr> <td>ResetComm</td><td>Service reset communication is executed</td></tr> </table>	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
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										



<i>BootTime</i>	<p>Specifies the period in milliseconds that the CANopen manager waits for the necessary CANopen devices before an error is reported.</p> <div>  NOTICE If during the <i>BootTime</i> no communication with the connected CANopen modules has been possible due to a failure of the CAN bus, the CANopen-PN/2 terminates the boot process and does not even try to start the modules after the failure is ended. In this case the write record <i>Reset CANopen Manager</i> (see chapter 5.8.15) must be called to restart the CANopen boot process after the problem has been resolved. </div>
<i>BootDelay</i>	<p>The period in milliseconds that the CANopen manager waits after the <i>NMT Reset Communication</i> command before he continues with the boot process of the CANopen devices.</p>
<i>CANopen-PN Compatibility Mode</i>	<p>This mode reverts some of the improvements of the CANopen-PN/2 to simulate a behaviour that is more similar to the CANopen-PN (see chapter 8.1).</p> <div>  NOTICE When loading an GSDML file of the CANopen-PN (C.2921.02) into the CANopen-PN/2 (C.2931.02) this mode will be activated automatically. </div>

Table 20: GSDML Composer CANopen Manager Parameter



NOTICE

For basic information about CANopen refer to chapter 4. For further information please read the CANopen specification CiA 301 [2].

5.5.12 CANopen Device

The *Device Editor* contains all device-specific configuration options of the selected CANopen device. To open it, go to the *CANopen Network Editor* and use double left-click or use the context menu to open the *Device Editor* of the selected CANopen device. A new window will open, which has a navigation menu on the left side. Each navigation point has different parameters which are described throughout this chapter. All screenshots are based on an esd CAN-CBX-DIO8.



NOTICE

Not every CANopen device supports every option in the *Device Editor*. Therefore, based on the EDS file some options might be disabled for specific CANopen device. Moreover, read only options are also not editable.

5.5.12.1 Device Information

In this menu the device name (*Node Name*) and the CANopen node ID (*Node ID*) can be configured in the section *Device Commissioning*. Furthermore, general information about the device is shown. This information is extracted from the CANopen objects 0x1000, 0x1008 up to 0x100a and 0x1018.

The screenshot shows the 'CAN-CBX-DIO8 Device Information' window in the GSDML Composer. The window has a title bar with 'CANopen Network Editor' and 'CAN-CBX-DIO8 (#6)'. On the left is a navigation tree with the following items: CAN-CBX-DIO8, Device Information, RX PDO Mapping, TX PDO Mapping, Manager Settings, Sync/Emergency, Heartbeat/Guarding, Mandatory Objects [7], Optional Objects [91], Manufacturer Objects [42], All Objects, Search results, EDS Device Info, EDS File Info, and EDS Comments. The main area is divided into several sections:

- Device Commissioning**:
 - Node ID: 6 (dropdown)
 - Node Name: CAN-CBX-DIO8 (text field)
- Manufacturer [1008..100A]**:
 - Device Name: CAN-CBX-DIO8 ext. Eickhoff- (text field)
 - Hardware Version: 1.0 (text field)
 - Software Version: 1.0 (text field)
- Device Identity [1018]**:
 - Vendor Id: 23 (text field) and 0x17 (text field)
 - Product Code: 0 (text field) and 0x0 (text field)
 - Revision Number: 0 (text field) and 0x0 (text field)
 - Serial Number: 0 (text field) and 0x0 (text field)
- Device Type [1000]**:
 - Device Profile Number: 401 (text field)
 - Profile Name: Device profile for generic I/O modules (text field)
 - Additional Information: 3 (text field) and 0x3 (text field)

At the bottom left is a search bar with a magnifying glass icon and the text 'Search objects'. At the bottom right is a button labeled 'Verify and close'.

Figure 25: GSDML Composer Device Information

5.5.12.2 RPDO Mapping

This menu can be used to configure the RPDO mapping, which means the input PDOs for the CANopen device. The RPDOs are sent from the CANopen-PN/2 to the CANopen device. The PLC provides the data for the CANopen-PN/2. Therefore, they are mapped into the PLC output address space.



NOTICE

The number of supported RPDOs per CANopen device is limited to 15. This has no influence on the number of supported TPDOs. This means a CANopen device supports up to 15 RPDOs and 15 TPDOs at the same time.

The window consists of the following sections (see Figure 26):

- 1 - Object Selection
- 2 - PDO Selection
- 3 - Buttons to switch between selections
- 4 - *Communication Parameters*

The functionality is described in Table 21.

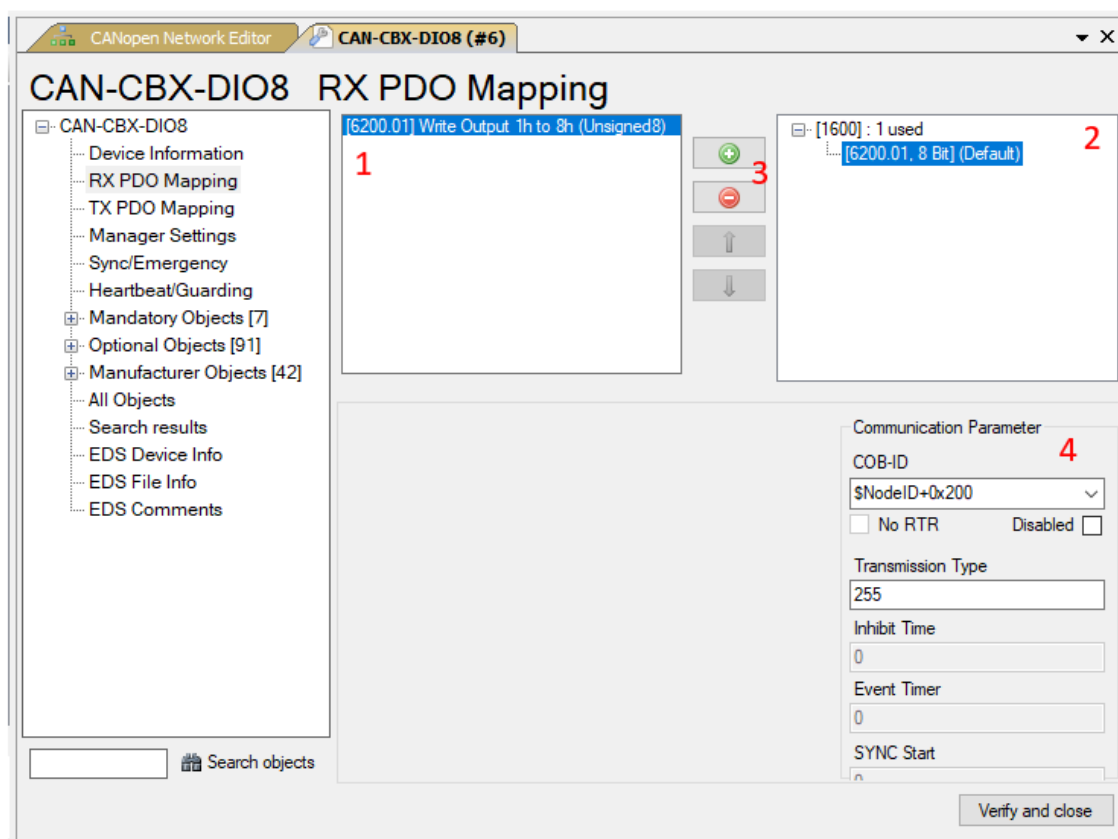



Figure 26: GSDML Composer RPDO Mapping

Parameter	Description																												
<i>Object Selection</i> (see Figure 26, 1)	This list shows all objects that can be mapped in a PDO. It is defined in the EDS file by [PDOMapping] and [AccessType] of the CANopen object.																												
<i>PDO Selection</i> (see Figure 26, 2)	In this list all PDOs and their contents are shown. The example in Figure 26 shows the PDO 0x1600 containing the mapped object “[6200.01, 8 Bit] (Default)”: The mapped object is at index 0x6200, sub-index 0x01 and has 8-bit data width. <i>Default</i> indicates that this value complies with the EDS <i>DefaultValue</i> for the PDO object.																												
<i>Communication Parameter</i> (see Figure 26, 4)	<p>In this section the COB-ID and the transmission type of the PDO can be specified.</p> <table border="1"> <tr> <th>Parameter</th><th>Description</th></tr> <tr> <td><i>COB-ID</i></td><td>The CAN-ID of the selected PDO. In accordance with CiA 306 (6) in the EDS file \$NodeID+ may be used as prefix. This is allowed here too, and the COB-ID is calculated when exported based of the node ID.</td></tr> <tr> <td><i>No RTR</i></td><td>No Remote Transmit Request</td></tr> <tr> <td><i>Disabled</i></td><td>Here the process data channel, containing the COB-ID specified above, can be disabled.</td></tr> <tr> <td><i>Transmission Type</i></td><td> <p>The following transmission types can be configured:</p> <table border="1"> <tr> <td>0x00</td><td>Acyclic with SYNC</td></tr> <tr> <td>0x01</td><td>Cyclic with every SYNC</td></tr> <tr> <td>0x02 ... 0xF0</td><td>Cyclic with every SYNC as specified here (2nd up to 240th)</td></tr> <tr> <td>0xFC</td><td>On RTR (sampling of the values to SYNC)</td></tr> <tr> <td>0xFD</td><td>On RTR (sampling of the values at RTR)</td></tr> <tr> <td>0xFE</td><td>Event based (manufacturer-specific)</td></tr> <tr> <td>0xFF</td><td>Event based (profile-specific)</td></tr> <tr> <td>SYNC:</td><td>Transmission is triggered at reception of a SYNC telegram</td></tr> <tr> <td>Event based:</td><td>Transmission is independent of the SYNC telegram</td></tr> </table> </td></tr> </table> <p>All other parameters do only apply to TPDOs (see chapter 5.5.12.3).</p>	Parameter	Description	<i>COB-ID</i>	The CAN-ID of the selected PDO. In accordance with CiA 306 (6) in the EDS file \$NodeID+ may be used as prefix. This is allowed here too, and the COB-ID is calculated when exported based of the node ID.	<i>No RTR</i>	No Remote Transmit Request	<i>Disabled</i>	Here the process data channel, containing the COB-ID specified above, can be disabled.	<i>Transmission Type</i>	<p>The following transmission types can be configured:</p> <table border="1"> <tr> <td>0x00</td><td>Acyclic with SYNC</td></tr> <tr> <td>0x01</td><td>Cyclic with every SYNC</td></tr> <tr> <td>0x02 ... 0xF0</td><td>Cyclic with every SYNC as specified here (2nd up to 240th)</td></tr> <tr> <td>0xFC</td><td>On RTR (sampling of the values to SYNC)</td></tr> <tr> <td>0xFD</td><td>On RTR (sampling of the values at RTR)</td></tr> <tr> <td>0xFE</td><td>Event based (manufacturer-specific)</td></tr> <tr> <td>0xFF</td><td>Event based (profile-specific)</td></tr> <tr> <td>SYNC:</td><td>Transmission is triggered at reception of a SYNC telegram</td></tr> <tr> <td>Event based:</td><td>Transmission is independent of the SYNC telegram</td></tr> </table>	0x00	Acyclic with SYNC	0x01	Cyclic with every SYNC	0x02 ... 0xF0	Cyclic with every SYNC as specified here (2 nd up to 240 th)	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 is triggered at reception of a SYNC telegram	Event based:	Transmission is independent of the SYNC telegram
Parameter	Description																												
<i>COB-ID</i>	The CAN-ID of the selected PDO. In accordance with CiA 306 (6) in the EDS file \$NodeID+ may be used as prefix. This is allowed here too, and the COB-ID is calculated when exported based of the node ID.																												
<i>No RTR</i>	No Remote Transmit Request																												
<i>Disabled</i>	Here the process data channel, containing the COB-ID specified above, can be disabled.																												
<i>Transmission Type</i>	<p>The following transmission types can be configured:</p> <table border="1"> <tr> <td>0x00</td><td>Acyclic with SYNC</td></tr> <tr> <td>0x01</td><td>Cyclic with every SYNC</td></tr> <tr> <td>0x02 ... 0xF0</td><td>Cyclic with every SYNC as specified here (2nd up to 240th)</td></tr> <tr> <td>0xFC</td><td>On RTR (sampling of the values to SYNC)</td></tr> <tr> <td>0xFD</td><td>On RTR (sampling of the values at RTR)</td></tr> <tr> <td>0xFE</td><td>Event based (manufacturer-specific)</td></tr> <tr> <td>0xFF</td><td>Event based (profile-specific)</td></tr> <tr> <td>SYNC:</td><td>Transmission is triggered at reception of a SYNC telegram</td></tr> <tr> <td>Event based:</td><td>Transmission is independent of the SYNC telegram</td></tr> </table>	0x00	Acyclic with SYNC	0x01	Cyclic with every SYNC	0x02 ... 0xF0	Cyclic with every SYNC as specified here (2 nd up to 240 th)	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 is triggered at reception of a SYNC telegram	Event based:	Transmission is independent of the SYNC telegram										
0x00	Acyclic with SYNC																												
0x01	Cyclic with every SYNC																												
0x02 ... 0xF0	Cyclic with every SYNC as specified here (2 nd up to 240 th)																												
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 is triggered at reception of a SYNC telegram																												
Event based:	Transmission is independent of the SYNC telegram																												


Table 21: GSDML Composer RPDO Mapping Parameter

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 are considered. In the event of a fault the object is not inserted without further acknowledgement.

- **Delete an object from 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 .

5.5.12.3 TPDO Mapping

This menu can be used to configure the TPDO mapping, which means the output PDOs of the CANopen device. The TPDOs are sent to the CANopen-PN/2 from the CANopen device. The PLC receives the data from the CANopen-PN/2. Therefore, they are mapped into the PLC input address space.



NOTICE

The number of supported TPDOs per CANopen device is limited to 15. This has no influence on the number of supported RPDOs. This means a CANopen device supports up to 15 RPDOs and 15 TPDOs at the same time.

The menu itself is similar to the RPDO Mapping (see chapter 5.5.12.2).

The window consists of the following sections (see Figure 27):

- 1 - Object Selection,
- 2 - PDO Selection
- 3 - Buttons to switch between the selections
- 4 - *Communication Parameter*

The functionality is described in Table 22.

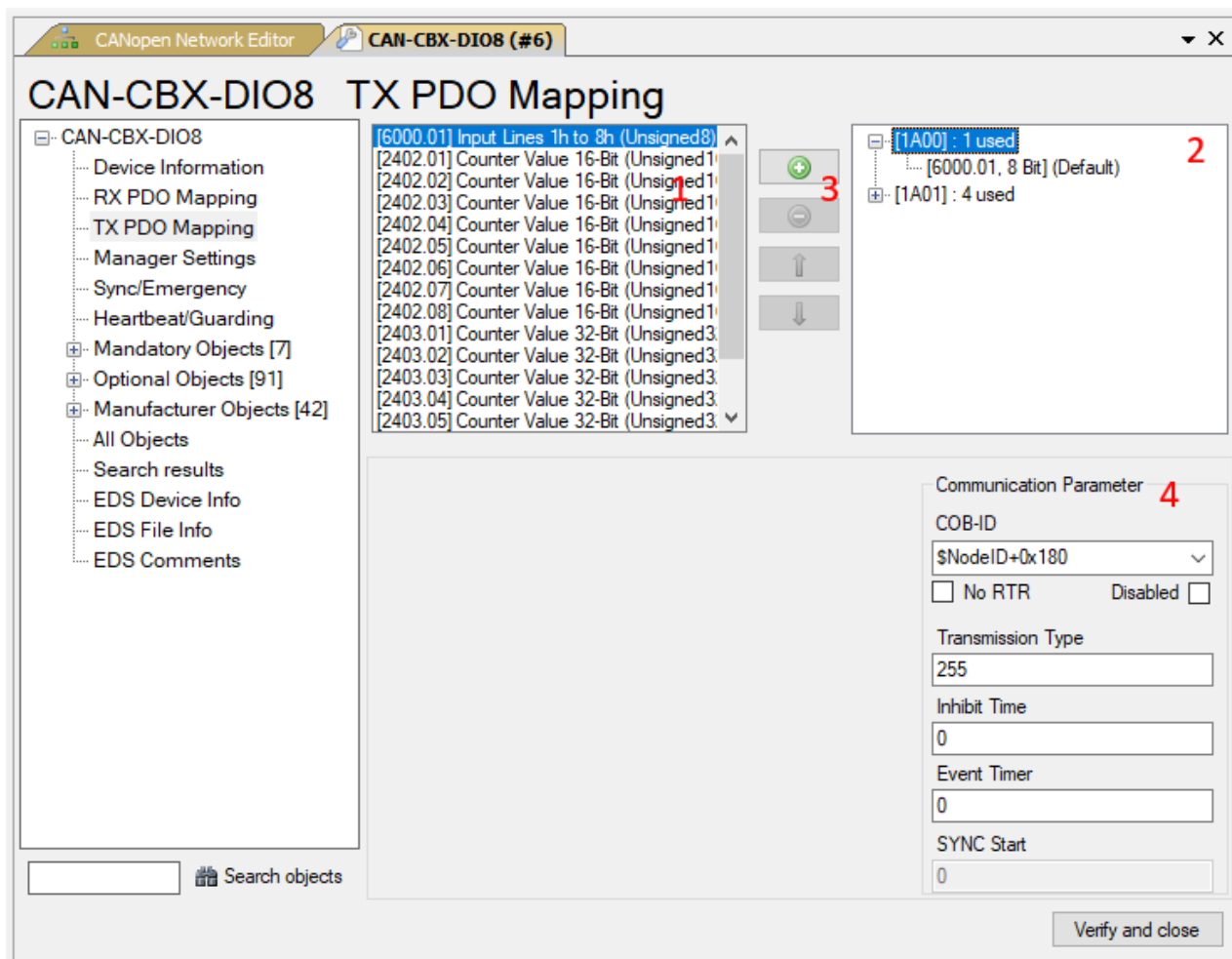



Figure 27: GSDML Composer TPDO Mapping

Parameter	Description																														
<i>Object Selection</i> (see Figure 27, 1)	This list shows all objects that can be mapped in a PDO. It is defined in the EDS file by [PDOMapping] and [AccessType] of the CANopen object.																														
<i>PDO Selection</i> (see Figure 27, 2)	In this list all PDOs and their contents are shown. The example in Figure 27 shows the PDO 0x1A00 containing the mapped object “[6000.01, 8 Bit] (Default)”: The mapped object is at index 0x6200, sub-index 0x01 with 8-bit data width. <i>Default</i> indicates that this value complies with the EDS <i>DefaultValue</i> for the PDO object.																														
<i>Communication Parameter</i> (see Figure 27, 4)	<p>In this section the COB-ID and the transmission type of the PDO can be specified.</p> <table> <tr> <th>Parameter</th><th>Description</th></tr> <tr> <td><i>COB-ID</i></td><td>The CAN-ID of the selected PDO. In accordance with CiA 306 (6) in the EDS file \$NodeID+ may be used as prefix. This is allowed here too, and the COB-ID is calculated when exported based of the node ID.</td></tr> <tr> <td><i>Transmission Type</i></td><td> <p>The following transmission types can be configured:</p> <table> <tr> <td>0x00</td><td>Acyclic with SYNC</td></tr> <tr> <td>0x01</td><td>Cyclic with every SYNC</td></tr> <tr> <td>0x02 ... 0xF0</td><td>Cyclic with every SYNC as specified here (2nd up to 240th)</td></tr> <tr> <td>0xFC</td><td>On RTR (sampling of the values to SYNC)</td></tr> <tr> <td>0xFD</td><td>On RTR (sampling of the values at RTR)</td></tr> <tr> <td>0xFE</td><td>Event based (manufacturer-specific)</td></tr> <tr> <td>0xFF</td><td>Event based (profile-specific)</td></tr> <tr> <td>SYNC:</td><td>Transmission is triggered at reception of a SYNC telegram</td></tr> <tr> <td>Event based:</td><td>Transmission is independent of the SYNC telegram</td></tr> </table> </td></tr> <tr> <td><i>Inhibit Time</i></td><td>Minimum period between two transmissions (As multiple of 100 µs. Only if transmission type is 0xFE/0xFF).</td></tr> <tr> <td><i>Event Timer</i></td><td>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).</td></tr> <tr> <td><i>SYNC Start</i></td><td>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).</td></tr> </table> <p>All other parameters do only apply to RPDOs (see chapter 5.5.12.2).</p>	Parameter	Description	<i>COB-ID</i>	The CAN-ID of the selected PDO. In accordance with CiA 306 (6) in the EDS file \$NodeID+ may be used as prefix. This is allowed here too, and the COB-ID is calculated when exported based of the node ID.	<i>Transmission Type</i>	<p>The following transmission types can be configured:</p> <table> <tr> <td>0x00</td><td>Acyclic with SYNC</td></tr> <tr> <td>0x01</td><td>Cyclic with every SYNC</td></tr> <tr> <td>0x02 ... 0xF0</td><td>Cyclic with every SYNC as specified here (2nd up to 240th)</td></tr> <tr> <td>0xFC</td><td>On RTR (sampling of the values to SYNC)</td></tr> <tr> <td>0xFD</td><td>On RTR (sampling of the values at RTR)</td></tr> <tr> <td>0xFE</td><td>Event based (manufacturer-specific)</td></tr> <tr> <td>0xFF</td><td>Event based (profile-specific)</td></tr> <tr> <td>SYNC:</td><td>Transmission is triggered at reception of a SYNC telegram</td></tr> <tr> <td>Event based:</td><td>Transmission is independent of the SYNC telegram</td></tr> </table>	0x00	Acyclic with SYNC	0x01	Cyclic with every SYNC	0x02 ... 0xF0	Cyclic with every SYNC as specified here (2 nd up to 240 th)	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 is triggered at reception of a SYNC telegram	Event based:	Transmission is independent of the SYNC telegram	<i>Inhibit Time</i>	Minimum period between two transmissions (As multiple of 100 µs. Only if transmission type is 0xFE/0xFF).	<i>Event Timer</i>	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).	<i>SYNC Start</i>	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).
Parameter	Description																														
<i>COB-ID</i>	The CAN-ID of the selected PDO. In accordance with CiA 306 (6) in the EDS file \$NodeID+ may be used as prefix. This is allowed here too, and the COB-ID is calculated when exported based of the node ID.																														
<i>Transmission Type</i>	<p>The following transmission types can be configured:</p> <table> <tr> <td>0x00</td><td>Acyclic with SYNC</td></tr> <tr> <td>0x01</td><td>Cyclic with every SYNC</td></tr> <tr> <td>0x02 ... 0xF0</td><td>Cyclic with every SYNC as specified here (2nd up to 240th)</td></tr> <tr> <td>0xFC</td><td>On RTR (sampling of the values to SYNC)</td></tr> <tr> <td>0xFD</td><td>On RTR (sampling of the values at RTR)</td></tr> <tr> <td>0xFE</td><td>Event based (manufacturer-specific)</td></tr> <tr> <td>0xFF</td><td>Event based (profile-specific)</td></tr> <tr> <td>SYNC:</td><td>Transmission is triggered at reception of a SYNC telegram</td></tr> <tr> <td>Event based:</td><td>Transmission is independent of the SYNC telegram</td></tr> </table>	0x00	Acyclic with SYNC	0x01	Cyclic with every SYNC	0x02 ... 0xF0	Cyclic with every SYNC as specified here (2 nd up to 240 th)	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 is triggered at reception of a SYNC telegram	Event based:	Transmission is independent of the SYNC telegram												
0x00	Acyclic with SYNC																														
0x01	Cyclic with every SYNC																														
0x02 ... 0xF0	Cyclic with every SYNC as specified here (2 nd up to 240 th)																														
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 is triggered at reception of a SYNC telegram																														
Event based:	Transmission is independent of the SYNC telegram																														
<i>Inhibit Time</i>	Minimum period between two transmissions (As multiple of 100 µs. Only if transmission type is 0xFE/0xFF).																														
<i>Event Timer</i>	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).																														
<i>SYNC Start</i>	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).																														


Table 22: GSDML Composer TPDO Mapping Parameter

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 considered. In the event of a fault the object will not be inserted without further acknowledgement.

- **Delete an object from a PDO**

Clicking the -button deletes the object in the PDO, which was 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 .

5.5.12.4 Manager Settings

The settings of the CANopen manager for the CANopen devices can be specified on this menu. It also defines the error handling of the CANopen device.

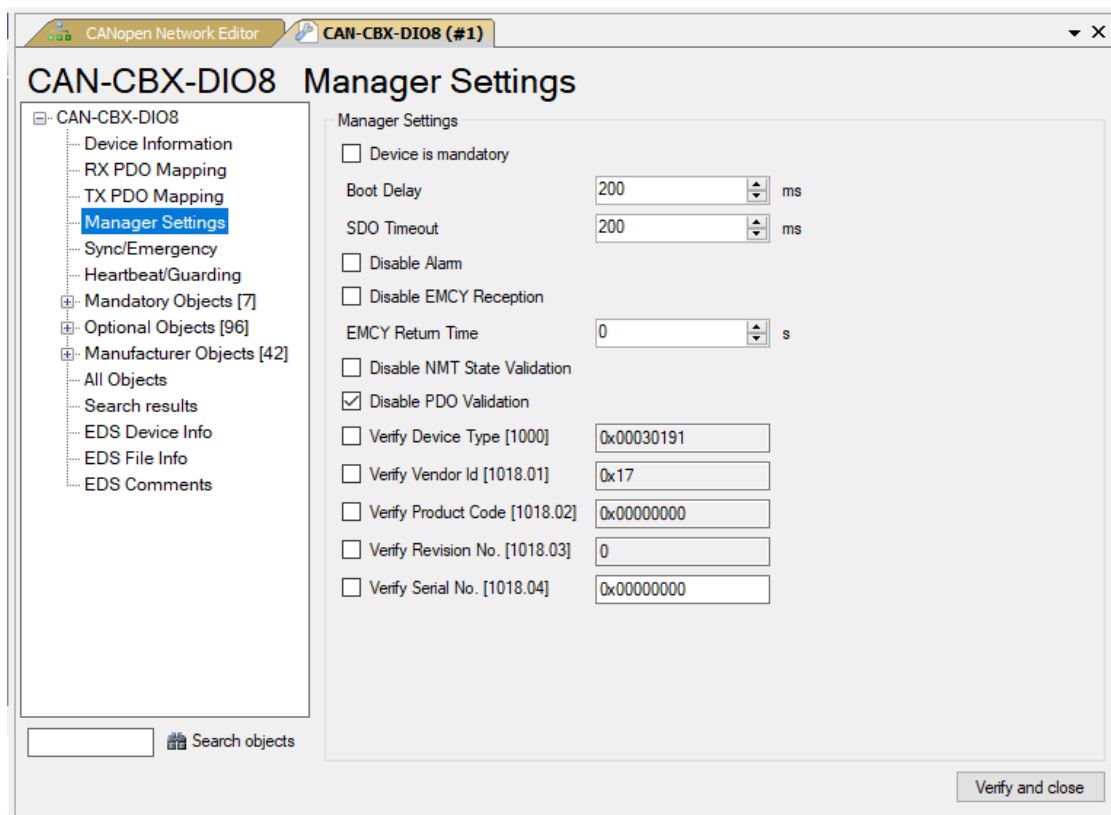


Figure 28: GSDML Composer *Manager Settings*

Parameter	Description
<i>Device is mandatory</i>	Defines whether the complete CANopen network may be started if this CANopen device is missing or if errors occurred during initialisation of this device.
<i>Boot Delay</i>	The period in milliseconds the CANopen manager waits after a reset of this CANopen device in case of an error after the <i>NMT Reset Communication</i> command or <i>NMT Reset Application</i> command before it continues with the boot process of this CANopen devices.
<i>SDO Timeout</i>	The maximum period in milliseconds that the CANopen manager waits for the CANopen device's response to an SDO request.
<i>Disable Alarm</i>	Disables all alarms of the CANopen node (see chapter 5.7.1).
<i>Disable EMCY</i>	Disables the reception of EMCY frames.
<i>EMCY Return Time</i>	When the CANopen manager receives an EMCY frame from the CANopen device, it will be sent to the PLC. However, not all CANopen devices reset the error. Therefore, this parameter defines after which period in seconds the error is resolved automatically, when the CANopen device is still in the NMT state OPERATIONAL.
<i>Disable NMT State Validation</i>	Normally, the PROFINET provider and consumer status of the CANopen device are only valid when the node is in the NMT state OPERATIONAL (see chapter 5.7.2). This safety mechanism is disabled when this checkbox is checked.
<i>Disable PDO Validation</i>	At start-up, all RPDOs have an invalid PROFINET provider status. When the PDO is received once and the NMT state is OPERATIONAL (see chapter 5.7.2). This safety mechanism is disabled when this checkbox is checked.
<i>Verify Device Type [1000]</i>	Defines whether the CANopen device may only be started if the device type exactly matches this value.
<i>Verify Vendor Id [1018.01]</i>	Defines whether the CANopen device may only be started if the vendor Id exactly matches this value.
<i>Verify Product Code [1018.02]</i>	Defines whether the CANopen device may only be started if the product code exactly matches this value.
<i>Verify Revision No. [1018.03]</i>	Defines whether the CANopen device may only be started if the revision number exactly matches this value.
<i>Verify Serial No. [1018.04]</i>	Defines whether the CANopen device may only be started if the serial No. exactly matches this value.

Table 23: GSDML Composer Manager Settings Parameter

5.5.12.5 SYNC / Emergency

This menu can be used to configure the settings of SYNC, TIME and EMCY messages.

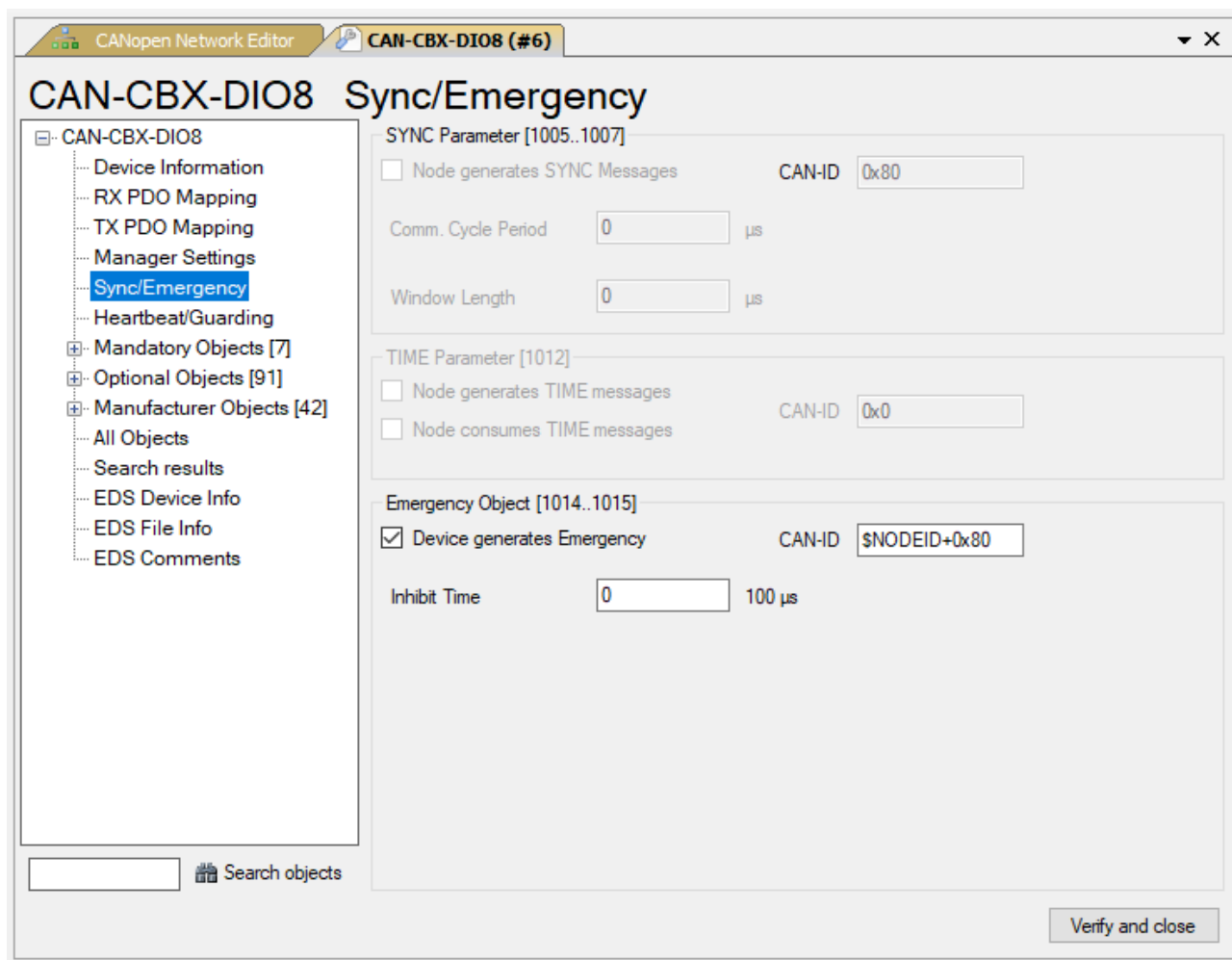


Figure 29: GSDML Composer Sync/Emergency



Parameter	Description
SYNC Parameter	
<i>Node generates SYNC Messages</i>	<p>Defines whether the complete CANopen network may be started if this device is missing or if errors occurred during initialisation of this device.</p> <div>  NOTICE Ensure that only one device in the CANopen network generates the SYNC messages! </div>
<i>CAN-ID</i>	CAN-ID of the SYNC messages. Currently only the CAN-ID 0x80 is supported.
<i>Comm. Cycle Period</i>	Period of the SYNC messages
<i>Window Length</i>	Period after a SYNC message, in which the TPDOs may be transmitted. Only applicable for synchronised PDOs, i.e. PDOs with 'Transmission Type' \leq 0xF0.
TIME Parameter	
<i>Node generates TIME Messages</i>	<p>Defines whether this device generates the TIME messages of the CANopen network.</p> <div>  NOTICE Ensure that only one device in the CANopen network generates the TIME messages! </div>
<i>Node consumes TIME Messages</i>	Defines whether this device is the 'Time Stamp Consumer', i.e. it requires/shall use TIME messages.
<i>CAN-ID</i>	CAN-ID of the TIME messages
Emergency Object	
<i>Device generates Emergency</i>	Specifies whether this device may generate EMCY messages.
<i>Inhibit Time</i>	The minimum period between two EMCY messages (specified as multiple of 100 μ s).
<i>CAN-ID</i>	CAN-ID of the EMCY message

Table 24: GSDML Composer Sync/Emergency Parameter

5.5.12.6 Heartbeat / Guarding

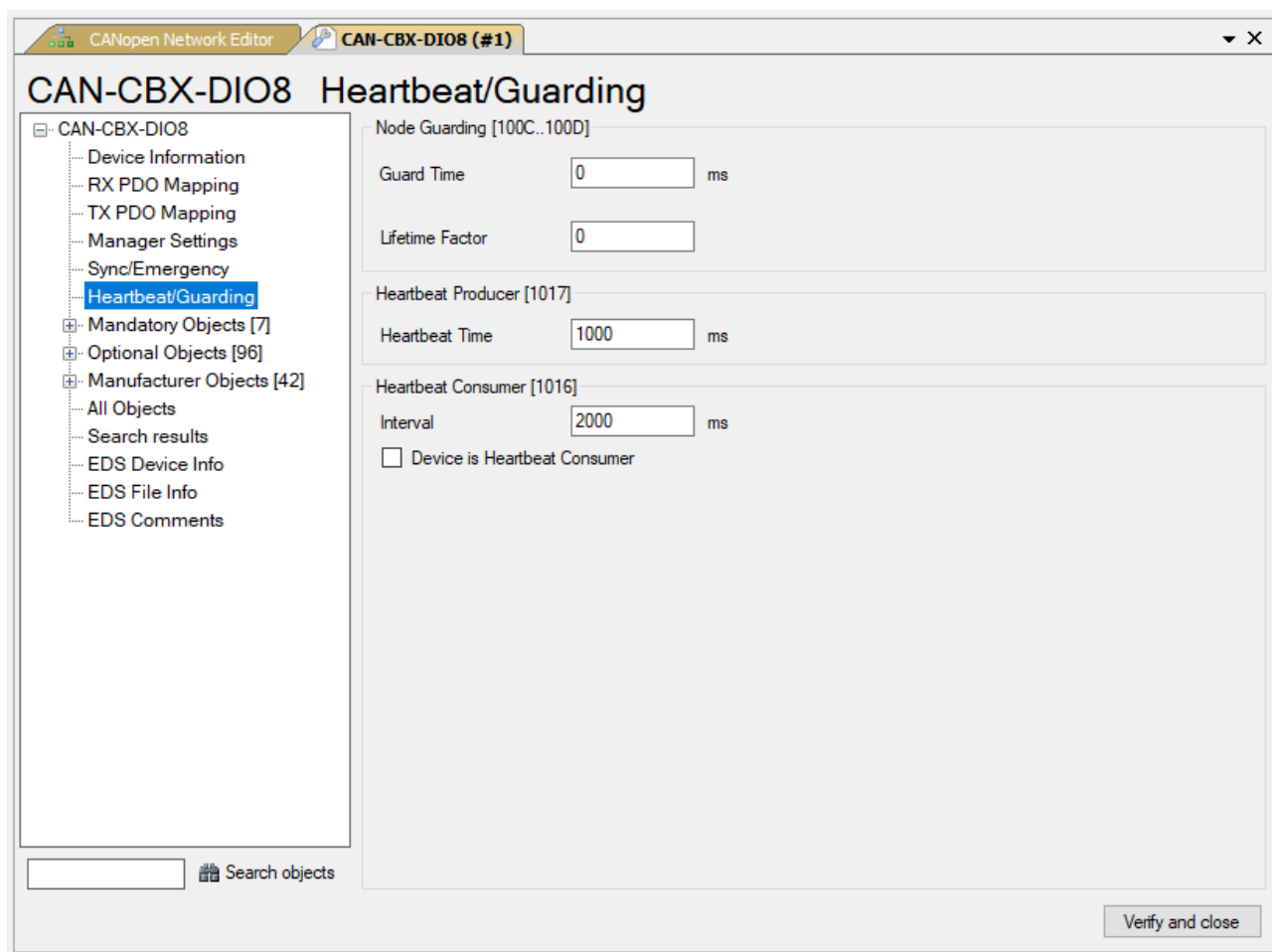


Figure 30: GSDML Composer *Heartbeat/Guarding*




Parameter	Description
Node Guarding	
<i>Guard Time</i>	<p>The NMT manager transmits guarding requests to the device in this interval.</p> <div>  NOTICE Ensure that only one device in the CANopen network generates the SYNC messages! </div>
<i>Lifetime Factor</i>	The <i>Guard Time</i> multiplied by this value results in the node lifetime. This is the period after which the device reports an error if there have not been any guarding requests received.
<div>  NOTICE Both values <i>Guard Time</i> and <i>Lifetime Factor</i> must be greater than '0' to keep the Node Guarding enabled. </div>	
Heartbeat	
<i>Heartbeat Producer, Heartbeat Time</i>	If a value greater than '0' is specified here, this device is <i>Heartbeat Producer</i> and generates heartbeat messages in the interval specified.
<i>Device is Heartbeat Consumer</i>	Defines whether the CANopen device is a heartbeat consumer of the heartbeat messages of the CANopen-PN/2. The time interval is taken from the CANopen manager settings (see chapter 5.5.11).
<i>Heartbeat Consumer, Interval</i>	<p>This value specifies the interval in which the producer, in this case the CANopen-PN/2, expects the heartbeat messages from this CANopen device ('0' to disable).</p> <p>This value must be sufficiently greater than the interval of the producer, because otherwise heartbeat errors might occur, only due to the jitter at transmission/reception.</p>
<div>  NOTICE It is not possible to use both protocols simultaneously. If both protocols are configured, the Heartbeat protocol is used. </div>	

Table 25: GSDML Composer Sync/Emergency Parameter

5.5.12.7 Object Lists

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

The following object lists are available:

<i>Mandatory Objects</i>	Lists the objects of the EDS section [MandatoryObjects]
<i>Optional Objects</i>	Lists the objects of the EDS section [OptionalObjects]
<i>Manufacturer Objects</i>	Lists the objects of the EDS section [ManufacturerObjects]
<i>All Objects</i>	Lists all objects

	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 31: GSDML Composer Object Lists

With the search function *Search Objects* of the *Device Editor* 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.

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 GSDML file and the CANopen manager writes them into the CANopen devices during initialisation.

The following parameter are shown in an object list:

Parameter	Description	
<i>Index / Subindex</i>	Index/Subindex of the object (EDS section name)	
<i>Name</i>	Name of the object (ParameterName in EDS)	
<i>Default</i>	Default value of the object (DefaultValue in EDS)	
<i>Value</i>	Current value of the object (ParameterValue in EDS)	
	Appearance	Description
	Yellow	Value can be edited in the object list
	Red	Invalid value (e.g. value is not in the range between ‘Min’ and ‘Max’)
	Bold	Value differs from default value
<i>Data Type</i>	Data type of the object (DataType in EDS)	
<i>Min</i>	Minimum value (LowLimit in EDS)	
<i>Max</i>	Maximum value (HighLimit in EDS)	
<i>Type</i>	Type of the object (ObjectType in EDS)	
<i>Access</i>	Access rights of the object (AccessType in EDS)	
<i>PDO Mapping</i>	Specifies whether the object is PDO mappable (PDOMapping in EDS)	

Table 26: GSDML Composer Object List Parameter

5.5.12.8 EDS Device Info

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

5.5.12.9 EDS File Info

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

5.5.12.10 EDS Comments

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

5.5.13 Output

The application messages are listed in this window.

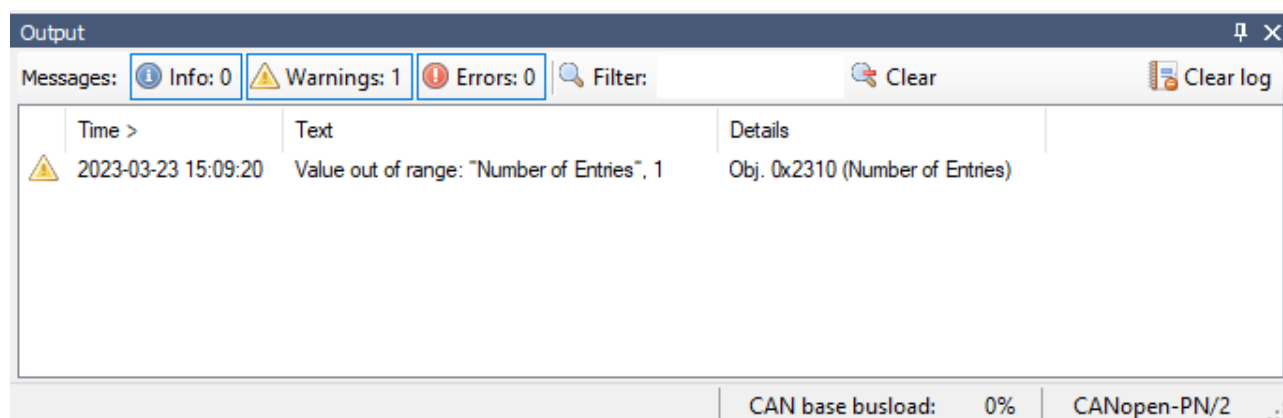


Figure 32: GSDML Composer *Output*

All messages are of type *Info*, *Warnings* or *Errors* and they are marked by different symbols in the first column.

The type can be selected with the corresponding buttons in the toolbar.

Each entry of a message consists of the date and time, a text and if present, detailed information.

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 are deleted. Furthermore, all entries are discarded if the GSDML Composer is closed.

The status bar on the bottom of the *Output* window shows the gateway type that is currently used and the *CAN base busload*. This is the estimated CAN busload from SYNC and guarding telegrams as well as from synchronous RPDOs and TPDOs. A CAN busload of less than 50% is recommended. However, because the sending and receiving interval is not fixed, asynchronous PDOs are not included into the calculation.

5.6 Module In- and Output

This chapter describes how to translate the GSDML Composer configuration to the PROFINET IO device. It is recommended to read chapter 5.4 and 5.5 first, before reading this chapter.

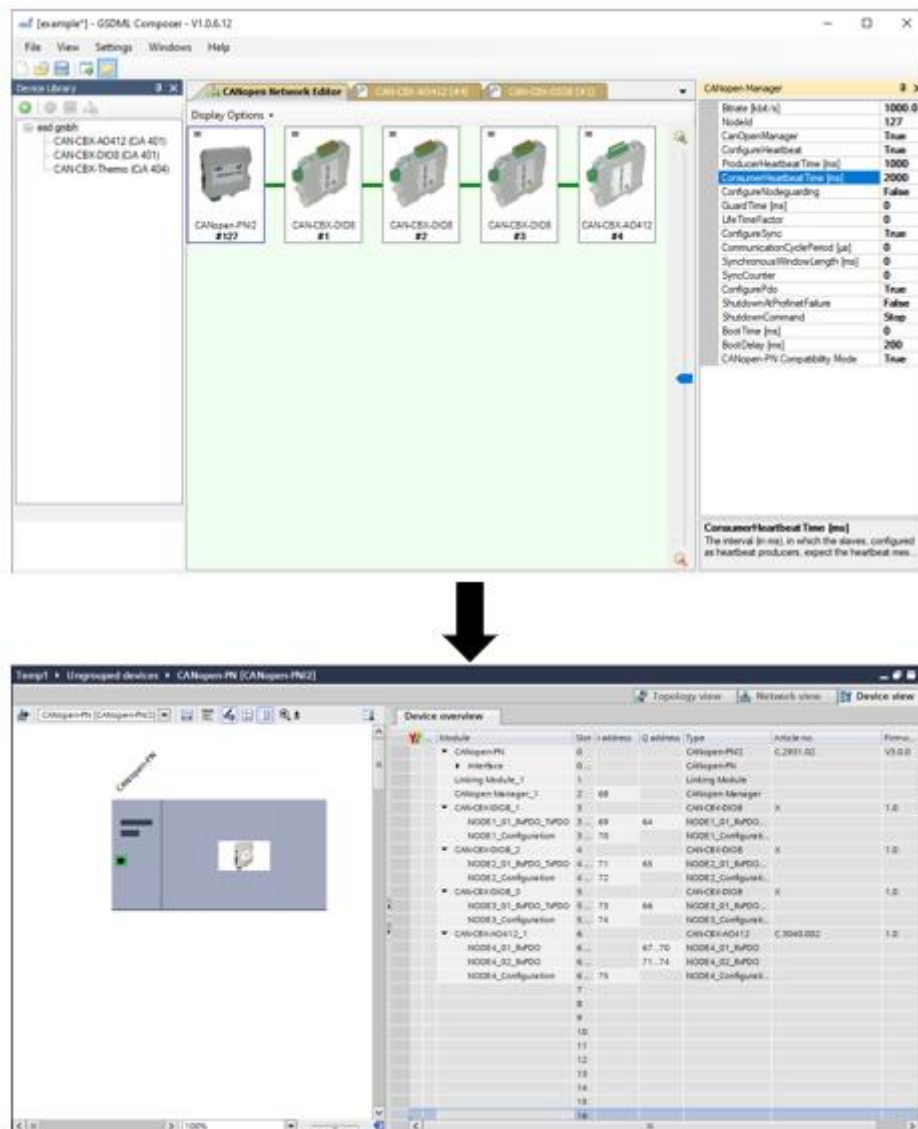
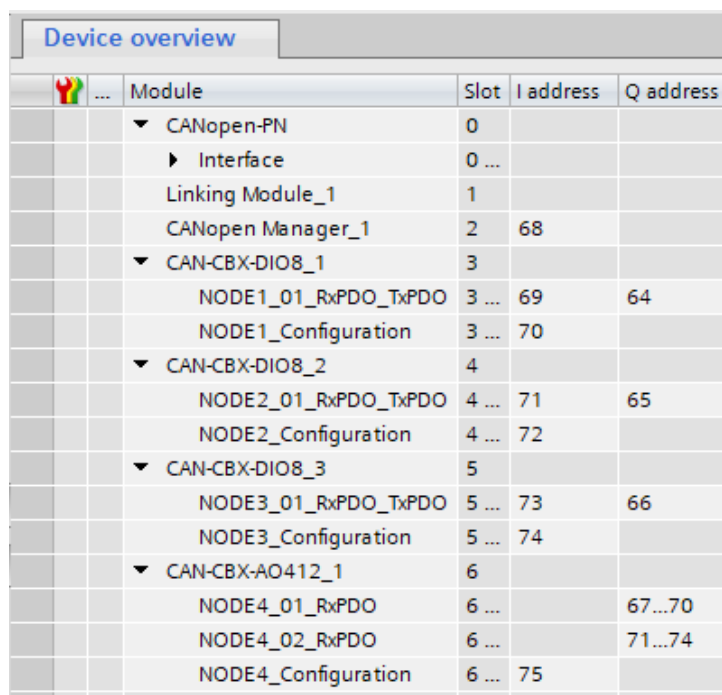


Figure 33: GSDML Composer <-> Siemens TIA Portal

5.6.1 Overview

The configuration of the GSDML project is converted into the module and submodule structure of PROFINET. The best way to describe it is by using the *Device overview* of the Siemens TIA Portal which offers a clearly arranged overview of the configuration.



Module	Slot	I address	Q address
▼ CANopen-PN	0		
▶ Interface	0 ...		
Linking Module_1	1		
CANopen Manager_1	2	68	
▼ CAN-CBX-DIO8_1	3		
NODE1_01_RxPDO_TxPDO	3 ...	69	64
NODE1_Configuration	3 ...	70	
▼ CAN-CBX-DIO8_2	4		
NODE2_01_RxPDO_TxPDO	4 ...	71	65
NODE2_Configuration	4 ...	72	
▼ CAN-CBX-DIO8_3	5		
NODE3_01_RxPDO_TxPDO	5 ...	73	66
NODE3_Configuration	5 ...	74	
▼ CAN-CBX-AO412_1	6		
NODE4_01_RxPDO	6 ...		67...70
NODE4_02_RxPDO	6 ...		71...74
NODE4_Configuration	6 ...	75	

Figure 34: Device Overview

Some modules are always there regardless of the configurations, others are build based of the number of CANopen devices and PDOs. The slots offer the following functionality:

Name	Slot	Description
<i>CANopen-PN</i>	0	This module is used as PROFINET interface. It is used to configure the PROFINET IO parameters.
<i>Linking Module</i>	1	This module has currently no functionality.
<i>CANopen Manager</i>	2	This module represents the CANopen manager. It has a 1-byte input which offers the current NMT state of the manager (see Table 9).
<i>CANopen Device</i>	3 ... 128	This module represents a CANopen device. The submodules are used for PDOs and the NMT state.

Table 27: CANopen-PN/2 Slot Structure

5.6.2 CANopen Manager

The CANopen Manager module is always in slot 2 and contains its configuration. By selecting the submodule `CANopen_Manager_1` in the *Device view*, the CANopen manager settings can be displayed. To do that open *Properties* → *General* → *Module parameters*. However, the values cannot be changed.

5.6.3 CANopen Devices

Each CANopen device is represented by a module and its submodules.

The example configures four CANopen device, which are represented by one module each.

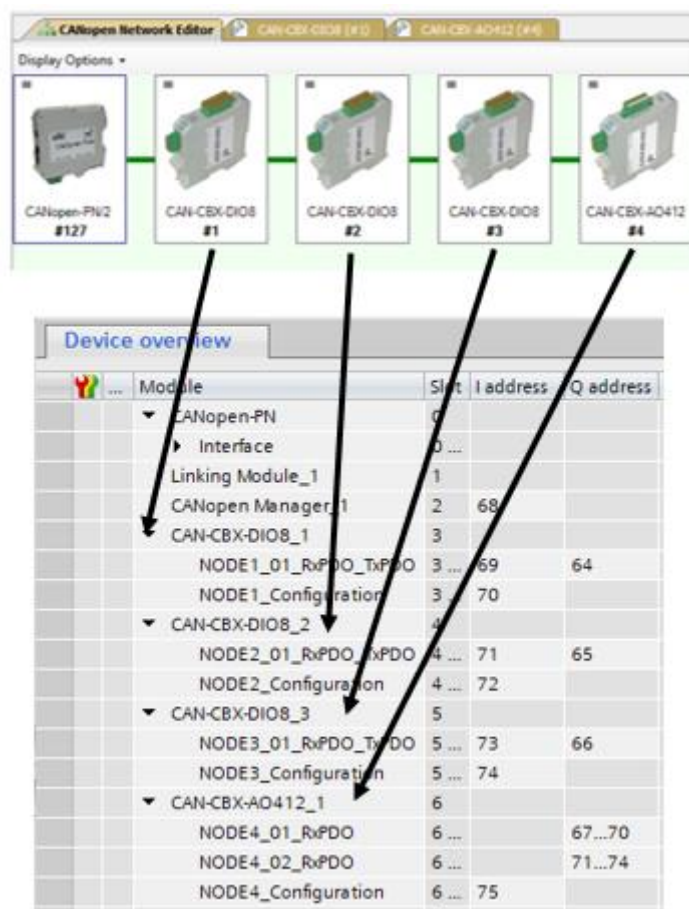


Figure 35: Device Overview CANopen Server Mapping

Each CANopen device has the submodule **NODE_{XXX}_Configuration**. It always has a 1-byte input, which shows the current NMT state (see Table 9) of the CANopen device. It contains the configuration of this node. By selecting the submodule **NODE_{XXX}_Configuration** in the *Device view*, the CANopen device settings can be displayed. To do that open *Properties* → *General* → *Module parameters*. However, the values cannot be changed.

Moreover, the CANopen device has up to 15 PDO submodules. One of the PDO submodules can be shared by one RPDO and one TPDO.

PDOs

The PDOs are directly mapped into the PLC address space. RPDOs are incoming data for the CANopen device, they are therefore represented by output data of the PLC. TPDOs on the other hand are outgoing data of the CANopen device and are therefore represented by input data of the PLC. The PDO length always defines how many bytes are allocated in the PLC address space.

When there are multiple CANopen objects configured within one PDO, the first object is always allocated in the first bytes of the PLC address space. However, the data is already endianness swapped to be directly mapped onto the accurate PLC data type.

Here are some examples:

The CANopen server CAN-CBX-AO412 configures two RPDOs and no TPDO. The two RPDOs have a length of 4 bytes each and are represented by two submodules called **NODE4_01_RxPDO** and **NODE4_02_RxPDO** which also allocate 4 bytes of output data in the PLC address space each.

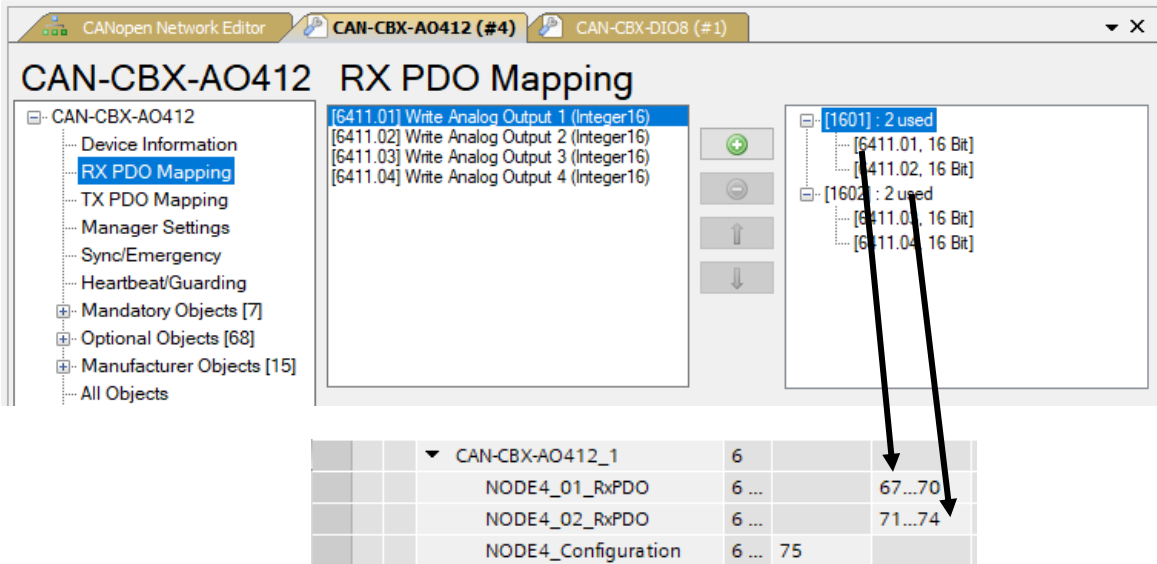


Figure 36: PDO Mapping CAN-CBX-AO412

Both RPDOs contain two CANopen objects which are mapped as follows:

PLC Address Space	CANopen Object
67	0x6411 sub-index 0x01
68	
69	0x6411 sub-index 0x02
70	

Table 28: PLC Address Space ↔ CANopen Objects

As second example the GSDML project configures multiple CAN-CBX-DIO8 modules with one RPDO and one TPDO. Both PDOs have a 1-byte length. Because an RPDO and a TPDO can share one submodule, the submodule **NODE1_01_RxPDO_TxPDO** represents both PDOs and has 1-byte of input and 1-byte of output data in the PLC address space.


CAN-CBX-DIO8_1	3		
NODE1_01_RxPDO_TxPDO	3 ...	69	64
NODE1_Configuration	3 ...	70	

Figure 37: PDO Mapping CAN-CBX-DIO8

5.7 Diagnostics

The gateway has an extensive diagnostics system. There are various ways which the gateway uses to provide diagnostics data. Not every PROFINET controller supports all of them. For further information, please refer to the corresponding PROFINET controller manual. The gateway uses alarms and the PROFINET provider and consumer status to display misconduct. Normally, when the error is gone, all alarms are dismissed, and all invalid provider and consumer stati are set to valid. However, this can take multiple seconds depending on the error.

5.7.1 Alarms

Alarms are used to inform the PROFINET controller about errors on the CANopen devices. The alarms can be sent by any CANopen module. Whenever this symbol  is displayed in the *Device overview*, an alarm is pending on the specific module. Further information about the alarm is described in the diagnostics of the module (Context Menu of Module -> *Online & diagnostics* -> *Diagnostics status* (Figure 38)).

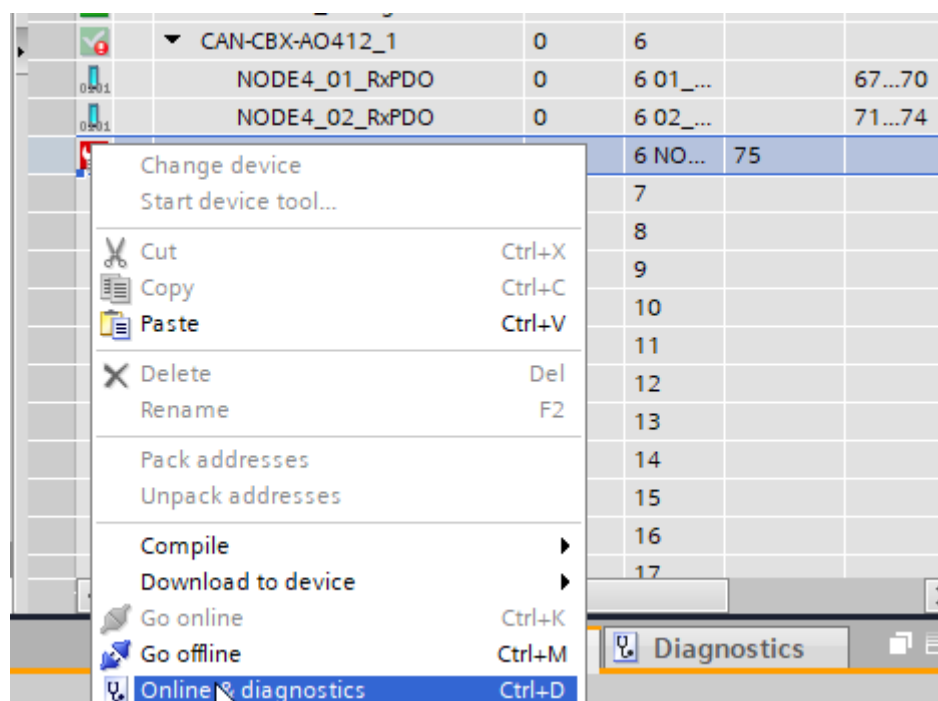


Figure 38: Alarm Diagnostics Information

Because the gateway uses a specific alarm type, it might not be displayed correctly. Therefore, it is recommended to read the alarm within the PLC application.



NOTICE

Some PLCs display the error codes as multiple errors. However, most of the time it is sufficient to check the error code (see Table 31) to get more information for further diagnostics.

Most PROFINET development environments have predefined function blocks to read alarm information. For the TIA Portal it is called **RALRM** and is described in more detail throughout this chapter.

The function block **RALRM** can be used to read diagnostics alarms. It is best to call this function in 'OB82' (Diagnostics Alarm-OB).

The following parameters need to be provided:

MODE	:= INT	(Input)
F_ID	:= HW_IO	(Input)
MLEN	:= INT	(Input)
NEW	:= BOOL	(Output)
STATUS	:= DWORD	(Output)
ID	:= DWORD	(Output)
LEN	:= INT	(Output)
TINFO	:= ANY	(I/O)
AINFO	:= ANY	(I/O)

A data block instance of the function block needs to be added. The data block is automatically generated when the function block is called.

Parameter	Description						
MODE	Operating mode <table> <tr> <td>0</td><td>Only parameter NEW and LEN are written.</td></tr> <tr> <td>1</td><td>All output parameters are rewritten.</td></tr> <tr> <td>2</td><td>All output parameters are rewritten if F_ID has triggered the alarm.</td></tr> </table>	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.
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	HW identifier of a PROFINET IO device						
MLEN	Maximum length of the bytes to be read in AINFO , the actually received length of the data is specified in LEN .						
NEW	A new alarm has been received						
STATUS	Error description						
ID	HW identifier of a module						
LEN	Length of the received alarm information						
TINFO	Administrative information						
AINFO	Alarm information (see Table 30)						

Table 29: Read Alarm Function Block Parameter

The alarm information is based on the CANopen EMCY messages. It has a total length of 36 bytes and is mapped to the diagnostics alarm according to the following table:

Parameter	Byte	Data type	Diagnosis Definition	Emergency Mapping
1	0 ... 1	unsigned16	Block type	
2	2 ... 3	unsigned16	Block length	
3	4 ... 5	unsigned16	Version	
4	6 ... 7	unsigned16	Alarm-type	1 = Diagnostics
5	8 ... 11	unsigned32	API	0x0000 0000
6	12 ... 13	unsigned16	Slot	
7	14 ... 15	unsigned16	Subslot	
8	16 ... 19	unsigned32	Module Ident Number	Defined by the GSDML Composer
9	20 ... 23	unsigned32	Submodule Ident Number	Defined by the GSDML Composer
10	24 ... 25	unsigned16	Alarm Specifier	
11	26 ... 27	unsigned16	User Structure Identifier	0x00CA
12	28 ... 29	unsigned16	User Data 1	Error Code; (see Table 31)
13	30	unsigned8	User Data 2	Error Register / Object 0x1001 (see Table 32)
14	31 ... 35	unsigned8 [5]	User Data 3	Manufacturer Specific Emergency Data (5 octets, see manual of the CANopen device)

Table 30: Alarm Information

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



NOTICE

There are various reasons for a CANopen device to send an EMCY message. Therefore, for further information read the manual of the respective CANopen device. There may be some CANopen devices, that do not reset their errors. Use the EMCY Reset Time (see chapter 5.5.12.4) or the write record *Reset CANopen Device EMCY* (see chapter 5.8.18) to prevent the PLC from a not acknowledged alarm.

The error codes are defined in the CANopen specification CiA 301 (5). In addition, the CANopen-PN/2 uses device specific errors to provide additional information in case of a bootup error. They are not part of the specification and have the error code 0xB0xx. The following table provides descriptions of the error codes:

Error Code (hex)		Description
HEX	DEZ	
0x0000	0	Error reset or no error
0x1000	4.096	Generic error
0x20xx	8.192*	Current
0x21xx	8.448*	Current, CANopen device input side
0x22xx	8.704*	Current inside the CANopen device
0x23xx	8.960*	Current, CANopen device output side
0x30xx	12.288*	Voltage
0x31xx	12.544*	Mains voltage
0x32xx	12.800*	Voltage inside the CANopen device
0x33xx	13.056*	Output voltage
0x40xx	16.384*	Temperature
0x41xx	16.640*	Ambient temperature
0x42xx	16.896*	Device temperature
0x50xx	20.480*	CANopen device hardware
0x60xx	24.576*	CANopen device software
0x61xx	24.832*	Internal software
0x62xx	25.088	User software
0x63xx	25.344*	Data set
0x70xx	28.672*	Additional modules
0x80xx	32.768*	Monitoring
0x81xx	33.024*	Communication
0x8110	33.040	CAN overrun (objects lost)
0x8120	33.056	CAN in error passive mode
0x8130	33.072	Life guard error or heartbeat error
0x8140	33.088	Recovered from bus off
0x8150	33.104	CAN-ID collision
0x82xx	33.280*	Protocol error
0x8210	33.296	PDO not processed due to length error
0x8220	33.312	PDO length exceeded
0x8230	33.328	DAM MPDO not processed, destination object not available
0x8240	33.344	Unexpected SYNC data length
0x8250	33.360	RPDO timeout

Software

0x90xx	36.864*	External error
0xF0xx	61.440*	Additional functions
0xFFxx	65.280*	Device specific
Bootup Error Codes		
0xB0xx	45.056*	CANopen device bootup error
0xB001	45.057	Device removed from network (Error A)
0xB002	45.058	Access to CANopen object 0x1000 failed (Error B)
0xB003	45.059	Wrong device type (Error C)
0xB004	45.060	Wrong vendor ID (Error D)
0xB005	45.061	Wrong product code (Error M)
0xB006	45.062	Wrong revision number (Error N)
0xB007	45.063	Wrong serial number (Error O)
0xB008	45.064	No heartbeat received (Error E)
0xB009	45.065	No guard reply received (Error F)
0xB00A	45.066	No expected SW version (Error G)
0xB00B	45.067	SW update not allowed (Error H)
0xB00C	45.068	SW update failed (Error I)
0xB00D	45.069	Config download failed (Error J)
0xB00E	45.070	No heartbeat received (Error K)
0xB00F	45.071	Device already operational (Error L)
0xB0FE	45.310	Boot process timed out
0xB0FF	45.311	Unknown status

* Start Value of the Emergency Class


Table 31: CANopen Error Codes

The error register is defined in the CANopen specification CiA 301(5). The following table provides descriptions of the error registers:

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

Table 32: CANopen Error Register

5.7.2 Provider and Consumer Status

Besides alarms the gateway also uses the producer and consumer status. This is a PROFINET feature to determine whether the exchanged data is valid. Whenever this symbol  is displayed in the *Device overview*, the provider/consumer status is invalid on the specific module. For PLC input data, the gateway acts as the provider of the data and therefore use the provider status. On the other side PLC output data are consumed by the gateway and therefore use the consumer status. Modules with in- and output data have both statuses.

The in- and output data of the gateway represent PDOs, which are only exchanged in the NMT state OPERATIONAL. Therefore, whenever a CANopen device is in another NMT state, the provider/consumer status is invalid. This behaviour can be disabled by deactivating the NMT validation (see chapter 5.5.12.4). Moreover, as long as the PDO of the module is not received, the provider status of the module is invalid, because it has only its default value '0'. This behaviour can also be disabled by deactivating the PDO validation (see chapter 5.5.12.4).



NOTICE

Some common error cases with alarms and provider/consumer status and its solution are described in chapter 9.

5.8 Records

PROFINET records are asynchronous operations that can be used to exchange noncyclic data between the PLC and the gateway. Read records receive data from the gateway while write records send data to the gateway. Therefore, read records need an input buffer in the PLC, in which the gateway can store the data. In accordance with the PROFIBUS International Document TC2-09-0002 (CANopen-Integration_7012_V10_Mar11) (1) 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 CiA 301 (5), can be controlled via PROFINET side.

The TIA Portal already has implemented function blocks called **RDREC** for read records and **WRREC** for write records. The function blocks can be configured and integrated easily. For further information, see chapter 5.8.19 and read the respective documentation of the TIA Portal.

Records are differentiated by their record index. In addition, the application must clarify the maximum number of bytes that can be read for read records and the maximum number of bytes that should be sent for write records. Throughout this chapter the data from a read record is referred to as input data (Gateway → PLC). The data of write records is referred to as output data (PLC → Gateway). All records are documented in detail throughout this chapter. Some services use a combination of a write record that triggers an event, followed by a read record that gets the result of the event.

The installer comes with a TIA Portal project attached which can be used as example.

The following overview shows all supported read records.

Index	Record Length [Bytes]	Description	Page
0xB711	1..2047*	SDO Upload (Get Result)	78
0xB762	28	Read Version	86

* Record length, value range and data type dependent on the object that is read.

Table 33: CANopen-PN/2 Read Records

The following overview shows all supported write records.

Index	Record Length [Bytes]	Description	Page
0xB711	5	SDO Upload (Start)	78
0xB713	5 ... 2051*	SDO Download	81
0xB715	1	Configure SDO Timeout	82
0xB731	1	Start CANopen Device	82
0xB732	1	Start CANopen Device	82
0xB733	1	Set CANopen Device to PRE-OPERATIONAL	83
0xB734	1	Reset CANopen Device	83
0xB735	1	Reset Communication	83
0xB751	4	Initialize Gateway	83
0xB754	3	Set Heartbeat Producer	84
0xB755	3	Set Node ID	85
0xB756	3	Start Emergency Consumer	85
0xB757	3	Stop Emergency Consumer	85
0xB771	1	Reset CANopen Manager	86
0xB772	1	Start CANopen Manager	86
0xB773	1	Stop CANopen Manager	87
0x003A	1	Reset CANopen Device EMCY	87

* Record length, value range and data type dependent on the object that is written.

Table 34: CANopen-PN/2 Write Records

5.8.1 SDO Upload (0xB711)

The SDO upload service is used to obtain the data from the object directory of a CANopen device. It uses a combination of a write record to start the upload and a read record to obtain the resulting data. It is only allowed to have one SDO interaction at the same time.

To start the SDO upload the following write record needs to be send:

Write Record SDO Upload (Index 0xB711 Record Length 5 Bytes)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>Node ID</i>	0x01 ... 0x7F	unsigned8
2	1..2	<i>Index</i>	0x0000 ... 0xFFFF	unsigned16
3	3	<i>Sub-index</i>	0x00 ... 0xFF	unsigned8
4	4	<i>Data type</i>	0x00 ... 0xFF	unsigned8

Table 35: Write Record SDO Upload (0xB711)

The resulting data of the SDO upload is stored in the gateway internally till they are sent via read record to the PLC. When reading the data is automatically adjusted in the byte order (Endianness) and returned. An overview of all supported datatypes is displayed in Table 37. In brief, the byte order of the objects with the sizes 2, 4 and 8 bytes are swapped 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 for the read record is defined as follows:

Read Record SDO Upload (Index 0xB711 Record Length 1 ... 2047 Bytes)				
Parameter	Byte	Description	Value Range	Data type
1	1 ... 2047*	<i>Data</i>	*	*

* Record length, value range and data type dependent on the object that is read.

Table 36: Read Record SDO Upload (0xB711)

In the event of an error a data response with ErrorCode = 0xDE (IODReadRes) and ErrorDeCode = 0x80 (PNIORW) is returned for read records. In ErrorCode1 – consisting of ErrorClass and ErrorCode - the following values are returned according to the Table 38.

The following table displays all supported data types and the information, whether the endianness of the data is swapped.

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 37: CANopen SDO Data types

PROFINET IO Side			CANopen Side	
Error Class	Error Code	Error Code1	SDO Abort Code	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.
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 a 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 38: SDO Transfer Error Code

5.8.2 SDO Download (0xB713)

The SDO download service is used to change the data in the object directory of a CANopen device. It is implemented as a write record and the following parameters are defined:

Write Record SDO Download (Index 0xB713 Record Length 5 ... 2051 Bytes)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>Node ID</i>	0x01 ... 0x7F	unsigned8
2	1..2	<i>Index</i>	0x0000 ... 0xFFFF	unsigned16
3	3	<i>Sub-index</i>	0x00 ... 0xFF	unsigned8
4	4	<i>Data type</i>	0x00 ... 0xFF	unsigned8
5	5..2051*	<i>Data</i>	*	*

* Record length, value range and data type dependent on the object that is written.

Table 39: Write Record SDO Download (0xB713)

5.8.3 Configure SDO Timeout (0xB715)

This write record can be used to change the SDO timeout. The SDO timeout is the time in which the gateway waits for an answer of the other CANopen device after a SDO request. The CANopen device is selected by the module, which sends the write record.



NOTICE

This value will become active when the CANopen manager is reset (see chapter 5.8.15). Normally, the value is specified by the GSDML Composer and does not have to be changed afterwards.

Write Record Configure SDO Timeout (Index 0xB715 Record Length 2 Bytes)				
Parameter	Byte	Description	Value Range	Data type
1	0 ... 1	<i>SDO Timeout [ms]</i>	0x0000 ... 0xFFFF	unsigned16

Table 40: Write Record Configure SDO Timeout (0xB715)

5.8.4 Start CANopen Device (0xB731)

This write record can be used to start a CANopen device. This means that its NMT state is changed to OPERATIONAL if possible. The only parameter is the node ID of the selected device. When the node ID is set to '0', all CANopen devices are addressed.

Normally, all devices are going to OPERATIONAL automatically after start-up.

Write Record Start CANopen Device (Index 0xB731 Record Length 1 Byte)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x00 ... 0x7F	unsigned8

Table 41: Write Record Start CANopen Device (0xB731)

5.8.5 Stop CANopen Device (0xB732)

This write record can be used to stop a CANopen device. This means that its NMT state is changed to STIPPED. The only parameter is the node ID of the selected device. When the node ID is set to '0', all CANopen devices are addressed.

Write Record Stop CANopen Device (Index 0xB732 Record Length 1 Byte)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x00 ... 0x7F	unsigned8

Table 42: Write Record Stop CANopen Device (0xB732)

5.8.6 Set CANopen Device to PRE-OPERATIONAL (0xB733)

This write record can be used to set a CANopen device to the NMT state PRE-OPERATIONAL. The only parameter is the node ID of the selected device. When the node ID is set to '0', all CANopen devices are addressed.

Write Record Set CANopen Device to PRE-OP. (Index 0xB733 Record Length 1 Byte)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x00 ... 0x7F	unsigned8

Table 43: Write Record Set CANopen Device to PRE-OP. (0xB733)

5.8.7 Reset CANopen Device (0xB734)

This write record resets a CANopen device. The only parameter is the node ID of the selected device. The node ID '0' is not allowed.

Write Record Reset CANopen Device (Index 0xB734 Record Length 1 Byte)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x01 ... 0x7F	unsigned8

Table 44: Write Record Reset CANopen Device (0xB734)

5.8.8 Reset Communication (0xB735)

This write record resets the NMT state machine of a CANopen device. The only parameter is the node ID of the selected device. The node ID '0' is not allowed.

Write Record Reset Communication (Index 0xB735 Record Length 1 Byte)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x01 ... 0x7F	unsigned8

Table 45: Write Record Reset Communication (0xB735)

5.8.9 Initialize Gateway (0xB751)

This write record can be used to change the bitrate of the CAN bus. Subsequently the CANopen manager will be restarted afterwards.



NOTICE

Because the bitrates 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. Normally, the value is specified by the GSDML Composer and does not have to be changed afterwards.

Write Record Initialize Gateway (Index 0xB751 Record Length 4 Byte)				
Parameter	Byte	Description	Value Range	Data type
1	0 ... 3	<i>CAN Bit Timing</i>	0x01 ... 0x08	unsigned32

Table 46: Write Record Initialize Gateway (0xB751)

The parameter *CAN Bit Timing* 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

Table 47: CAN Bit Timing

5.8.10 Set Heartbeat Producer (0xB754)

This write record can be used to set the heartbeat producer interval. The parameter *CANopen Node ID* is not evaluated.



NOTICE

This value will become active when the CANopen manager is reset (see chapter 5.8.15). Normally, the value is specified by the GSDML Composer and does not have to be changed afterwards.

Write Record Set Heartbeat Producer (Index 0xB754 Record Length 3 Byte)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x01 ... 0x7F	unsigned8
2	1 ... 2	<i>HeartbeatProducerTime</i>	0x0000 ... 0xFFFF	unsigned16

Table 48: Write Record Set Heartbeat Producer (0xB754)

5.8.11 Set Node ID (0xB755)

This write record can be used to set the CANopen node ID of the gateway itself.



NOTICE

This value becomes active when the CANopen manager is reset (see chapter 5.8.15). Normally, the value is specified by the GSDML Composer and does not have to be changed afterwards.

Write Record Set Node ID (Index 0xB755 Record Length 3 Bytes)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x01 ... 0x7F	unsigned8
2	1 ... 2	<i>HeartbeatProducerTime</i>	0x0000 ... 0xFFFF	unsigned16

Table 49: Write Record Set Node ID (0xB755)

5.8.12 Start Emergency Consumer (0xB756)

This write record can be used to enable forwarding of CANopen EMCY messages to the PLC via PROFINET. By default, the forwarding is enabled. Therefore, this write record is only useful when the forwarding has been disabled before (see chapter 5.8.13). The parameters are not evaluated.

Write Record Start Emergency Consumer (Index 0xB756 Record Length 3 Bytes)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x01 ... 0x7F	unsigned8
2	1 ... 2	<i>COB-ID</i>	0x0081 ... 0x00FF	unsigned16

Table 50: Write Record Start Emergency Consumer (0xB756)

5.8.13 Stop Emergency Consumer (0xB757)

This write record can be used to disable forwarding of CANopen EMCY messages to the PLC via PROFINET. By default, the forwarding is enabled. The parameters are not evaluated.

Write Record Stop Emergency Consumer (Index 0xB757 Record Length 3 Bytes)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x01 ... 0x7F	unsigned8
2	1 ... 2	<i>COB-ID</i>	0x0081 ... 0x00FF	unsigned16

Table 51: Write Record Stop Emergency Consumer (0xB757)

5.8.14 Read Version (0xB762)

This read request can be used to obtain information about the gateway itself. It returns the data in the following format:

Read Record Read Version (Index 0xB762 Record Length 28 Bytes)				
Parameter	Byte	Description	Value Range	Data type
1	0 ... 3	<i>Vendor-ID</i>	0x0000 0017	unsigned32
2	4 ... 7	<i>Product Code</i>	0x0029 3102	unsigned32
3	8 ... 11	<i>Revision number</i> (CANopen software, not firmware)	0XXXX YY ZZ (various)	unsigned32
4	12 ... 15	<i>Serial Number</i>	0 ... 0xFFFFFFFF	unsigned32
5	16 ... 19	<i>Gateway Class</i>	0x0000 0003	unsigned32
6	20 ... 23	<i>Protocol Version</i>	0x0001 0100	unsigned32
7	24 ... 27	<i>Implementation Class</i>	0x0004 0000	unsigned32

Table 52: Read Record Read Version (0xB762)

5.8.15 Reset CANopen Manager (0xB771)

This write record restarts the CANopen manager within the gateway. The parameter is not evaluated. It should only be used in exceptional cases – because additionally all CANopen devices are restarted. This leads to many PROFINET alarms, which might be possibly not completely processed, e.g. in the TIA portal.

Write Record Reset CANopen Manager (Index 0xB771 Record Length 1 Byte)				
Parameter	Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x01 ... 0x7F	unsigned8

Table 53: Write Record Reset CANopen Manager (0xB771)

5.8.16 Start CANopen Manager (0xB772)

This write record starts the CANopen manager within the gateway. The parameter is not evaluated. This is only useful if the CANopen manager has been stopped before (see chapter 5.8.17).

Write Record Start CANopen Manager (Index 0xB772 Record Length 1 Byte)				
Parameter	Output Byte	Description	Value Range	Data type
1	0	CANopen Node ID	0x01 ... 0x7F	unsigned8

Table 54: Write Record Start CANopen Manager (0xB772)

5.8.17 Stop CANopen Manager (0xB773)

This write record stops the CANopen manager within the gateway. The parameter is not evaluated.

Write Record Stop CANopen Manager (Index 0xB773 Record Length 1 Byte)				
Parameter	Output Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x01 ... 0x7F	unsigned8

Table 55: Write Record Stop CANopen Manager (0xB773)

5.8.18 Reset CANopen Device EMCY (0x003A)

Some CANopen manager can produce EMCY messages without a reset when the error is gone. To prevent pending alarms on the PLC the EMCY can be cleared after a fixed interval (see parameter *EMCY Reset Time* in chapter 5.5.12.4) or with this record.

Write Record Reset CANopen Device EMCY (Index 0x003A Record Length 1 Byte)				
Parameter	Output Byte	Description	Value Range	Data type
1	0	<i>CANopen Node ID</i>	0x01 ... 0x7F	unsigned8

Table 56: Write Record Reset CANopen Device EMCY (0x003A)

5.8.19 PLC Function Blocks

5.8.19.1 Read Records

The function block **RDREC** is used for reading read records asynchronously.

The following parameter needs to be provided:

```

REQ      := BOOL      (Input)
ID       := HW_IO     (Input)
INDEX    := DINT      (Input)
MLEN     := UINT      (Input)
VALID    := BOOL      (Output)
BUSY     := BOOL      (Output)
ERROR    := BOOL      (Output)
STATUS   := DWORD     (Output)
LEN      := UINT      (Output)
RECORD   := VARIANT (I/O)

```

A data block instance of the function block needs to be added. The data block is automatically generated when the function block is called.

Parameter	Description
REQ	Start read operation (always 1)
ID	HW identifier of a module
Index	Record index
MLEN	Minimum length of the bytes to be read. The actually received length of the data is returned in LEN .
VALID	Read operation was successful
BUSY	Read operation is in progress
ERROR	Error occurred during read operation, see parameter STATUS for a further information
STATUS	Error description
LEN	Number of received bytes
RECORD	Received data from the gateway. The received length is specified with the parameter LEN .

Table 57: Read Record Function Block Parameter



NOTICE

The parameter **REQ** is not edge-triggered. If the input is not reset accordingly, the operation will be repeated permanently.



NOTICE

The parameter **ID** needs the HW identifier of the CANopen manager module for every record except 'Set Heartbeat Producer' (0xB754) where the HW identifier is used to determine the device.

5.8.19.2 Write Records

The function block **WRREC** is used for writing write records asynchronously.

The following parameter needs to be provided:

```

REQ    := BOOL    (Input)
ID     := HW_IO   (Input)
INDEX  := DINT    (Input)
LEN    := UINT    (Input)
DONE   := BOOL    (Output)
BUSY   := BOOL    (Output)
ERROR  := BOOL    (Output)
STATUS := DWORD    (Output)
RECORD := VARIANT (I/O)

```

A data block instance of the function block needs to be added. The data block is automatically generated when the function block is called.

Parameter	Description
REQ	Start write operation (always 1)
ID	HW identifier of a module
Index	Record index
LEN	Length of the bytes to be transferred
DONE	Write operation done successfully
BUSY	Write operation is in progress
ERROR	Error occurred during write operation, see parameter STATUS for a further information
STATUS	Error description
RECORD	Data to be transmitted

Table 58: Write Record Function Block Parameter



NOTICE

The parameter **REQ** is not edge-triggered. If the input is not reset accordingly, the operation is repeated permanently.



NOTICE

The parameter **ID** needs the HW identifier of the CANopen manager module for every record.

5.9 CANopen-PN/2 Object Directory

As a CANopen device the CANopen-PN/2 has its own CANopen object directory which contains the following objects:

5.9.1 Objects of CiA Specification CiA 301

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	-	<i>Device Type</i>	unsigned32	ro	Default: 0x0000 0000
0x1001	-	<i>Error Register</i>	unsigned8	ro	Default: 0x00
0x1002	-	<i>Manufacturer Status Register</i>	unsigned32	ro	Default: 0x00
0x1003	0	<i>Pre-defined Error Field</i>	unsigned8	rw	Default: 0x00
	1 ... 254		unsigned32	ro	Default: 0x00
0x1005	-	<i>COB-ID-Sync</i>	unsigned32	rw	Default: 0x80
0x1006	-	<i>Communication Cycle Period</i>	unsigned32	rw	Def. via GSDML Composer
0x1008	-	<i>Manufacturer Device Name</i>	visible string	ro	CANopen-PN/2
0x1009	-	<i>Manufacturer Hardware Version</i>	visible string	ro	x.yy (depending on version)
0x100A	-	<i>Manufacturer Software Version</i>	visible string	ro	x.yy (depending on version)
0x100C	-	<i>Guard Time</i>	unsigned16	rw	Def. via GSDML Composer
0x100D	-	<i>Life Time Factor</i>	unsigned8	rw	Def. via GSDML Composer
0x1014	-	<i>COB-ID Emergency Object</i>	unsigned32	rw	0x80 + node ID
0x1015	-	<i>Inhibit Time Emergency</i>	unsigned16	rw	0
0x1016	0 ... 127	<i>Consumer Heartbeat Time</i>	array	rw	Def. via GSDML Composer
0x1017	-	<i>Producer Heartbeat Time</i>	unsigned16	rw	Def. via GSDML Composer
0x1018	0	<i>Identity Object</i>	unsigned8	ro	Number of Entries = 4
	1		unsigned32	ro	Vendor ID = 0x0000 0017
	2		unsigned32	ro	Device ID = 0x0029 3102
	3		unsigned32	ro	Revision
	4		unsigned32	ro	Serial Number

0x1400 ... 0x15FF	<i>RPDO Communication Parameter</i>	RPDO Communication Parameters of CANopen-PN/2 according to the number of PDOs of the TPDOs of the CANopen devices connected, as defined in the GSDML Composer
0x1600 ... 0x17FF	<i>RPDO Mapping Parameter</i>	RPDO Mapping Parameters of CANopen-PN/2 according to the number of PDOs of the TPDOs of the CANopen devices connected, as defined in the GSDML Composer
0x1800 ... 0x19FF	<i>TPDO Communication Parameter</i>	TPDO Communication Parameter of CANopen-PN/2 according to the number of PDOs of the RPDOs of the CANopen devices connected, as defined in the GSDML Composer
0x1A00 ... 0x1BFF	<i>TPDO Mapping Parameter</i>	TPDO Mapping Parameter of CANopen-PN/2 according to the number of PDOs of the RPDOs of the CANopen devices connected, as defined in the GSDML Composer

Table 59: CANopen-PN/2 object directory CiA Specification 301

5.9.2 Objects of CiA Specification CiA 302-2

CANopen objects according to CiA 302-2 (4) are implemented.



NOTICE

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

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 (4).

Index	Object code	Description	Data type	Access
0x102A	VAR	NMT inhibit time	unsigned16	rw
0x1F80	VAR	NMT start-up	unsigned32	rw
0x1F81	ARRAY	NMT manager 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

Table 60: CANopen-PN/2 object directory CiA Specification 302-2

6 Firmware Update

The CANopen-PN/2 provides the possibility of firmware updates.

To install a firmware update, the following steps need to be done:

Step	Action
1	Install the installer provided with the product (see chapter 5.3) with all packages.
1	Connect the gateway via Mini-USB with a Windows computer.
3	Make sure that CANopen-PN/2 is detected correctly and a network adapter called <i>RNDIS based ESD Device</i> shows up in the <i>Device Manager</i> (see chapter 5.3).
4	Extract the package called provided by the esd support, which is named <code>canopen_pn_2_update_X_X_X.zip</code> .
3	Execute the batch file <code>update_X_X_X.bat</code> .
4	When no error is displayed, the update was successfully transmitted to the gateway and is executed on the gateway. This process is indicated by a green-blinking LED 'R'.
4	Wait till the gateway restarts. This is indicated when the 'PWR' LED lights up again.
5	The update is completed.

Table 61: Firmware Update

7 CAN Monitoring

The CANopen-PN/2 can be used to monitor the CAN interface.
To configure the CAN monitoring, the following steps need to be done:

Step	Action
1	An installer is provided with the CANopen-PN/2. Install with all packages (see chapter 5.3).
2	Connect the CANopen-PN/2 via Mini-USB with a Windows computer.
3	Make sure that CANopen-PN/2 is detected correctly, and a network adapter called <i>RNDIS based ESD Device</i> shows up in the <i>Device Manager</i> (see chapter 5.3).
4	Start the program <i>CAN Control Panel</i> which is installed with the installer. By default, the net number is already set to 100. Set the parameter <i>Hostname / IP address</i> to 192.168.7.1 Make sure that the checkbox <i>Enabled</i> is checked. Do not change any other settings (see Figure 39). Press <i>Apply</i> .
5	Open the esd CAN tool CANreal. Select the net number '100' in the dropdown menu of the input field <i>Net</i> on the top. It is not needed to set the baud rate manually because the PROFINET controller already configured it. However, when the gateway is not connected to a PROFINET network, this parameter can also be set manually. Let all other values unchanged. Press <i>Start</i> .
6	Now CANreal can interact with the CAN interface on the gateway. Some basic functionalities are described in Figure 40. For further information see the CANreal manual (Start menu 'Program/esd/CAN SDK/Documentation').

Table 62: CAN Monitoring

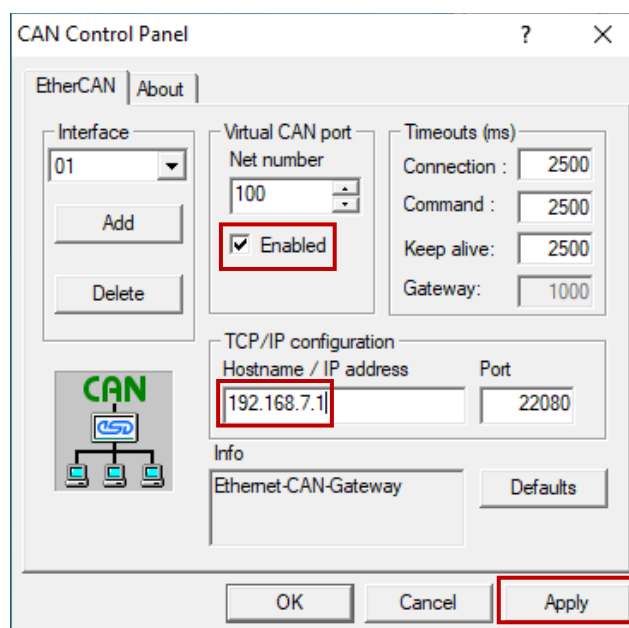


Figure 39: CAN Control Panel

CAN Monitoring

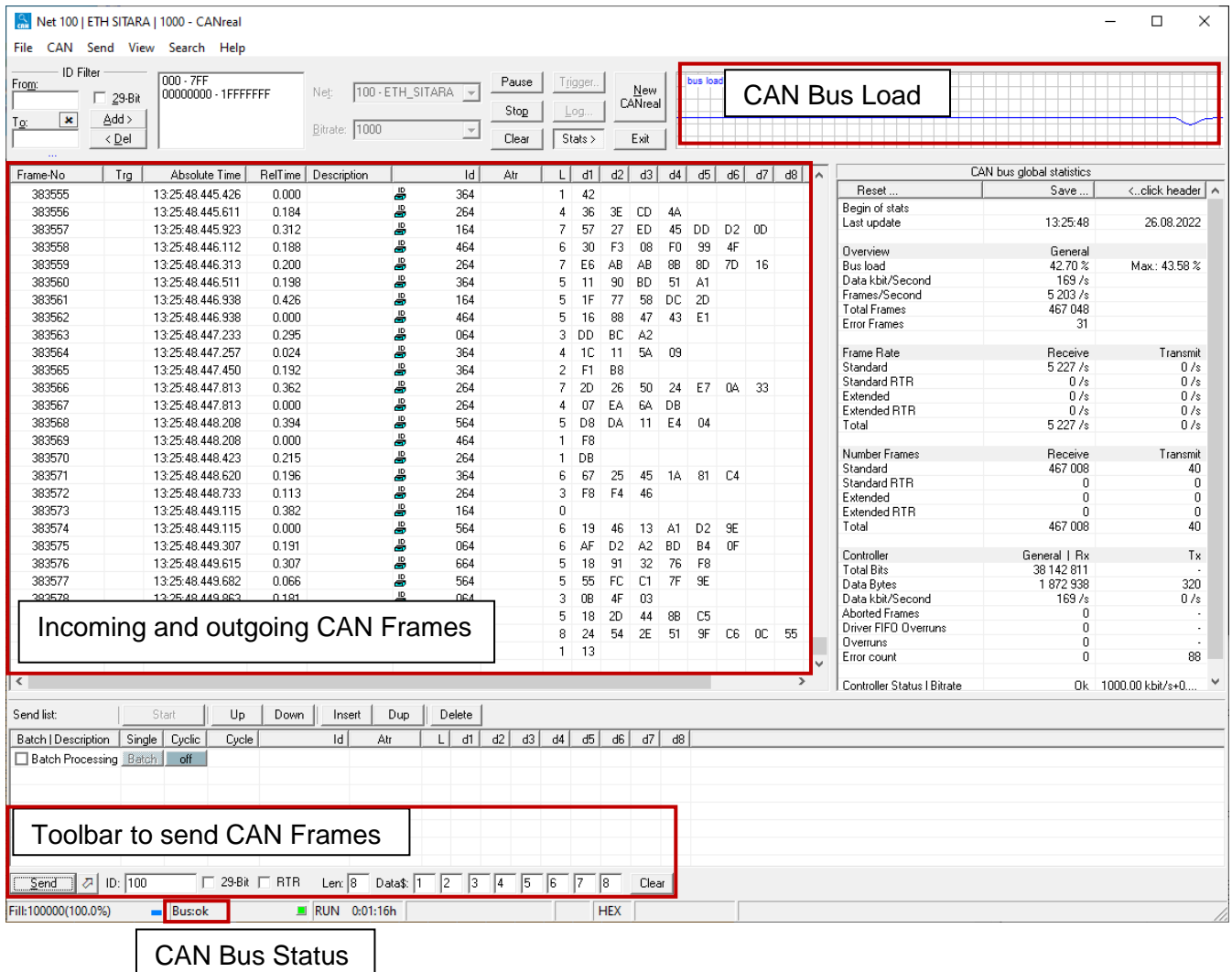


Figure 40: Monitoring the CAN Bus with CANreal



NOTICE

CANreal also offers the possibility to save the current CAN frames by clicking 'File→Save frames...'. In support cases this is useful to track the issue. The log file has the extension **.csplog**.

8 Compatibility

The CANopen-PN/2 (C.2931.02) is the successor of the CANopen-PN (C.2921.02). As such the new gateway is still compatible with the predecessor. That means, it is possible to use the new gateway as replacement for a CANopen-PN (C.2921.02) without any changes to the configuration. The GSDML file as well as the features of the predecessor are still fully supported. However, because the new gateways use a different soft- and hardware, minor timing difference may occur, and some changes have been made to improve the user experience.

**NOTICE**

It is not possible to configure the old CANopen-PN (C.2920.02) with the GSDML file of the CANopen-PN/2 (C.2931.02).

8.1 CANopen-PN Compatibility Mode

The new gateway has a so called 'CANopen-PN Compatibility Mode'. Whenever an old GSDML file for the original CANopen-PN is loaded into the CANopen-PN/2 this mode is activated automatically. Moreover, it is possible to configure a GSDML project for the new gateway and still activate this mode manually for compatibility reasons (see chapter 5.5.11).

The changes lead to the following deviations to this manual:


- The parameter 'CAN device is Heartbeat Consumer' (see chapter 5.5.12.6) is not used to configure the CANopen device. Instead, the time configured in the CANopen manager settings (see chapter 5.5.11) is used again.
- SYNC cannot be configured for CANopen devices (see chapter 5.5.12.5).
- The parameter *EMCY Reset Time* for CANopen nodes (see chapter 5.5.12.4) is set to 5 seconds and cannot be changed.
- The PROFINET provider/consumer status of a module is set to valid with the successful start of the CANopen device without NMT or PDO validation (see chapter 5.5.12.4).
- CANopen device boot failures always result in the error code 0x8130.
- CANopen device alarms are displayed in every submodule of the module and not just the last submodule.

9 Troubleshooting

This chapter shows some common error cases and how to solve them. It is explained on the Siemens TIA Portal as development environment.

9.1 Faulty PROFINET Connection

How does the error present itself?



- The 'CON' LED is not lit.
- The *Device overview* displays multiple missing modules ().

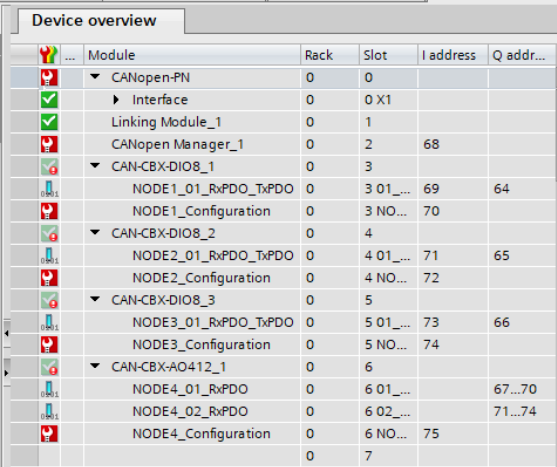
How can the error be solved?

- Check if the PROFINET device name of the gateway and the configuration match.
- Check if the PROFINET wiring is correct.

9.2 Faulty CAN Bus

How does the error present itself?

- The 'E' LED is lit continuously or lit up in single flashes (see chapter 1.4.3).
- The *Device overview* shows multiple alarms (), especially on the CANopen manager module, and multiple invalid provider and consumer statuses () (see Figure 41).



Module	Rack	Slot	I address	Q addr...
CANopen-PN	0	0		
Interface	0	0 X1		
Linking Module_1	0	1		
CANopen Manager_1	0	2	68	
CAN-CBX-DIO8_1	0	3		
NODE1_01_RxPDO_TxPDO	0	3 01...	69	64
NODE1_Configuration	0	3 NO...	70	
CAN-CBX-DIO8_2	0	4		
NODE2_01_RxPDO_TxPDO	0	4 01...	71	65
NODE2_Configuration	0	4 NO...	72	
CAN-CBX-DIO8_3	0	5		
NODE3_01_RxPDO_TxPDO	0	5 01...	73	66
NODE3_Configuration	0	5 NO...	74	
CAN-CBX-AO412_1	0	6		
NODE4_01_RxPDO	0	6 01...		67...70
NODE4_02_RxPDO	0	6 02...		71...74
NODE4_Configuration	0	6 NO...	75	
	0	7		

Figure 41: Faulty CAN Bus

How can the error be solved?

- Check if all CAN devices have the same baud rate.
- Check whether the CAN bus is terminated.
- Check that the CAN bus wiring is correct (see chapter 12).
- Check the error code of the CANopen manager module.



NOTICE

After the CAN bus is ok, it may take several seconds for the alarm to be resolved. When the CAN bus is faulty during configuration, the CANopen manager needs to be reset by restarting the module or using the write record 0xB771 (see chapter 5.8.15).

9.3 Faulty CANopen Device

How does the error present itself?

- The 'E' LED is lit up in double flashes (see chapter 1.4.3).
- The *Device overview* shows an alarm on a specific CANopen device module (🔴) and/or there is an invalid provider and/or consumer status (🔴) (see Figure 42 and Figure 43).

Device overview					
Module	Rack	Slot	I address	Q addr...	
🔴 CANopen-PN	0	0			
▶ Interface	0	0 X1			
✔ Linking Module_1	0	1			
✔ CANopen Manager_1	0	2	68		
✔ CAN-CBX-DIO8_1	0	3			
✔ NODE1_01_RxPDO_TxPDO	0	3 01....	69	64	
✔ NODE1_Configuration	0	3 NO...	70		
✔ CAN-CBX-DIO8_2	0	4			
✔ NODE2_01_RxPDO_TxPDO	0	4 01....	71	65	
✔ NODE2_Configuration	0	4 NO...	72		
✔ CAN-CBX-DIO8_3	0	5			
✔ NODE3_01_RxPDO_TxPDO	0	5 01....	73	66	
✔ NODE3_Configuration	0	5 NO...	74		
🔴 CAN-CBX-AO412_1	0	6			
✔ NODE4_01_RxPDO	0	6 01....		67...70	
✔ NODE4_02_RxPDO	0	6 02....		71...74	
🔴 NODE4_Configuration	0	6 NO...	75		
	0	7			

Figure 42: Faulty/Missing CANopen device

✔	NODE4_01_RxPDO	0	6 01....	67...70	
✔	NODE4_02_RxPDO	0	6 02....	71...74	
🔴	NODE4_Configuration	0	6 NO...	75	

Figure 43: CANopen device EMCY

How can the error be solved?

- The provider/consumer status is invalid (see Figure 42):
 - The server is unreachable or changed its NMT state to another state than operational.
 - Check if the heartbeat/node guarding settings are correct.
 - Check whether the CANopen device is wired up correctly.
 - Check the error code of the CANopen device module for further information.
- The provider/consumer status is valid (see Figure 43):
 - The server is still available and in the NMT state OPERATIONAL but has send an EMCY message.
 - Check the error code of the CANopen device module for further information.





NOTICE

Not every CANopen device resets its EMCY messages reliable. If this is the case, set the parameter *EMCY Reset Time* (see chapter 5.5.12.4). The alarm will be resolved after this time when the module is available and the NMT state is OPERATIONAL.

9.4 Invalid CAN Busload

How does the error present itself?

- The 'E' LED is lit up in double flashes (see chapter 1.4.3).
- The *Device overview* shows one or multiple alarms on CANopen device modules () and/or there is an invalid provider and/or consumer status ()

How can the error be solved?

- Check the 'CAN base busload' in the GSDML Composer (see chapter 5.5.13). This busload includes SYNC, guarding information and synchronous PDOs. The CiA recommends a maximum busload of <50%. Asynchronous PDOs are not included in the calculation because the sending or receiving interval is not fixed. When they are used try to calculate and add the additional busload produced by them on top of the *CAN base busload*.
- There are some ways to reduce the CAN busload.
 - If possible, try to use synchronous PDOs to have a more predictable and better calculatable CAN bus load.
 - For TPDOs use the parameter *Inhibit Time* to reduce the number of asynchronous PDOs on the CAN bus.

9.5 Support by esd

When you have a problem with the CANopen-PN/2 please make sure to check the troubleshooting chapter 9 first. If you still cannot find the solution to the problem, don't hesitate to contact our support team for help.

Please contact our support by email to support@esd.eu or by phone +49-511-37298-130.

In order to provide the fastest and best service, please provide the following information if possible:

- Detail error description
 - How does the error present itself?
 - Are alarms received on the PROFINET controller?
- Serial Number (printed on the device)
- GSDML Project (.xgcp)
- GSDML File (.xml)
- Siemens TIA Portal Project or at least a screenshot of the *Device view*
- CAN Monitoring Log (.csplog) (see chapter 7)

10 Technical Data

10.1 General Technical Data

Power supply voltage	Nominal voltage 18 V/DC ... 32 V/DC Current consumption (24 V, 20 °C): typical: 120 mA
Power consumption	Typical: 4.5 W (FW 50% CPU Load and 24 V power supply) Maximum: 5 W
Protective circuits	Reverse voltage protection Protection against transient overvoltages (triggering from 26 V)
Temperature range	0 °C... +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
Form factor / Dimensions	Width: 22.5 mm, height: 99 mm, depth: 114.5 mm (Without connectors)
Weight	130 g

Table 63: General Data of the module

10.2 CPU and Memory

CPU	ARM Cortex A9, 1 GHz, AM4377, 32-bit
SDRAM	1 Gbyte
EEPROM	256 kBit
NOR Flash	512 Mbit

Table 64: CPU and Memory

10.3 Connectors accessible from Outside

Name	Function, Interfaces	Type
CAN	CAN	5-pos. Phoenix Contact PCB header MC 1,5/5-GF-3,81 with PCB connector FK-MCP 1,5/5-STF-3,81
PORT1	PROFINET Port 1 (EtherCAT IN)	Dual port RJ45 socket with integrated transformer and LEDs
PORT2	PROFINET Port 2 (EtherCAT OUT)	
DIAG	USB-Device	Mini-USB socket, type B
24V	24V-power supply	4-pos. Phoenix Contact PCB header MSTBO 2,5/ 4-G1L KMGY with PCB connector FKCT 2,5/4-ST KMGY

Table 65: Connectors, accessible from outside

10.4 PROFINET IO Interface

Number of PROFINET interfaces	2 ports
Standard	IEEE 802.3, 100BASE-TX,
Bit rate	10/100 Mbit/s
Connection	Twisted Pair (compatible with IEEE 802.3), 100BASE-TX
Controller	Integrated in CPU
Electrical isolation	Via transformer, integrated in RJ-45 socket
Connector	Dual port RJ-45 socket in the front panel with integrated LEDs (Link- and Activity)

Table 66: Data of the PROFINET IO interface

10.5 DIAG Interface

Number	1
Standard	USB Specification Rev. 2.0
Bit rate	Max. 480 Mbit/s (Hi-speed)
Controller	Integrated in CPU
Connector	Mini-USB socket type B

Table 67: Data of the USB device interface

10.6 CAN Interface

Number of CAN interfaces	1
CAN controller	Integrated in CPU, According to ISO 11898-1 (CAN 2.0 A/B)
CAN protocol	According to ISO 11898-1:2015
Physical CAN Layer	High-speed CAN interface according to ISO 11898-2:2016, bit rate up to 1 Mbit/s
Electrical isolation	Separation by means of optocouplers and DC/DC-converters voltage over CAN isolation (CAN to slot bracket/EARTH; CAN to Host/System Ground; CAN to CAN): 1kV DC @ 1s ($I < 1 \text{ mA}$)
Bus termination	Terminating resistor must be set externally, if required
Connector	5-pin PCB connector

Table 68: Data of the CAN interface

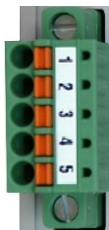
11 Connector Pin Assignments

11.1 CAN

Device connector: Phoenix Contact PCB header MC 1,5/5-GF-3,81
Cable plug: Phoenix Contact PCB connector FK-MCP 1,5/5-STF-3,81,
 Push-in spring connection ¹⁾
 Phoenix Contact Order No.: 1851261, included in delivery

Pin Position:

(Cable plug)



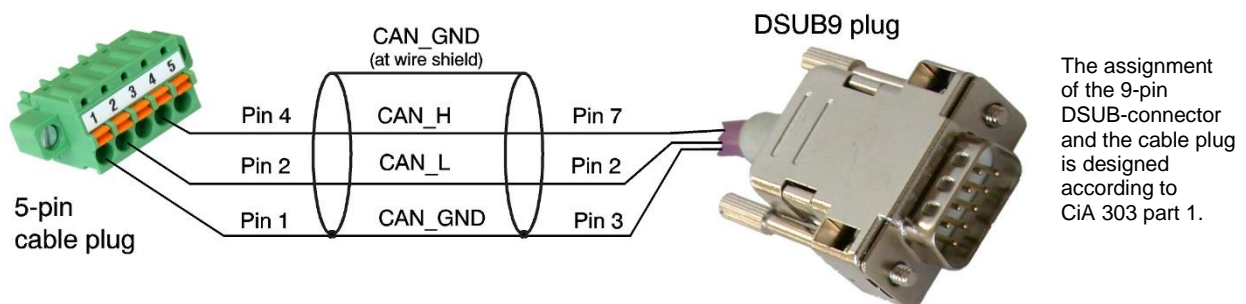
Pin Assignment:

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 signal lines
 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, if it is connected)
 - ... Reserved, do not connect!

Recommendation of an adapter cable from 5-pin cable plug (here Phoenix Contact FK-MCP1,5/5-STF_3,81 with push-in spring connection) to 9-pin DSUB:



¹⁾ For technical data (e.g. conductor cross section) see Phoenix Contact website, PCB Connectors, Product list PCB connectors

11.2 24V Power Supply Voltage



DANGER

The CANopen-PN/2 is a device of protection class III according to DIN EN IEC 61010-2-201 and may only be operated on supply circuits that offer sufficient protection against dangerous voltages.

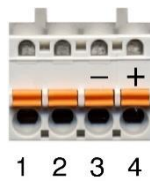
Device socket:

Phoenix Contact PCB header MSTBO 2,5/4-G1L-KMGY

Cable plug:

Phoenix Contact PCB connector FKCT 2,5/4-ST KMGY, 5 mm pitch ²⁾,
Push-in spring connection, included in the scope of delivery
(Phoenix Contact order No.: 19 21 90 0)

Pin Position:



Pin Assignment:

Device housing label	24V			
	.	.	M	P
Connector label	(none)	(none)	-	+

Pin	1	2	3	4
Signal	Do not connect !	Do not connect !	M24 (GND)	P24 (+ 24 V)

Please refer to the connecting diagram page 15.



NOTICE

Feeding through the +24V power supply voltage can cause damage on the modules. It is not permitted to feed through the power supply voltage through this connector and to supply the power supply voltage to another CAN module station!

- Make absolutely sure to connect the cables correctly to the cable plug!
- Use only suitable cables for the line plug.

Signal Description:

P24... Power supply voltage (18 V/DC – 32 V/DC)

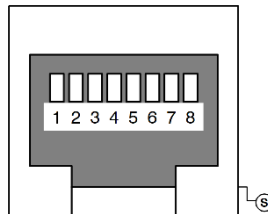
M24... Reference potential

²⁾ For technical data (e.g. conductor cross section) see Phoenix Contact website, PCB Connectors, Product list PCB connectors

11.3 PROFINET IO

Device Connector: RJ45 socket, 8-pin
According to IEEE 802.3-2015,
"Table 25–2—Twisted-pair MDI contact assignments"

Pin Position:



Pin Assignment:

Pin	Signal	Meaning
1	Tx0+ (TxD+)	Transmit Data +
2	Tx0- (TxD-)	Transmit Data -
3	Rx0+ (RxD+)	Receive Data +
4	-	-
5	-	-
6	Rx0- (RxD-)	Receive Data -
7	-	-
8	-	-
S	Shield	

Signal Description:

Tx0+/-, Rx0+/- ...	Ethernet data lines
- ...	reserved for future applications, do not connect!
Shield...	case shield, connected with the front panel of the CANopen-PN/2



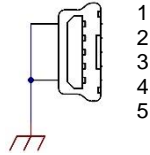
NOTICE

Cables of category CAT5 or higher must be used to grant the function in networks with 100 Mbit/s. esd grants the EU conformity of the product if the wiring is carried out with shielded twisted pair cables.

11.4 DIAG

Device connector: USB 2.0 Mini-B receptacle, standard pinning

Pin Position:



Pin Assignment:

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

Signal Description:

VBUS ...	+5 V power supply voltage
D+, D-...	Data USB 2.0, differential pair +/-
-	Reserved (ID for USB-type). Do not connect!
GND...	Reference potential

12 Correct Wiring of 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), such as regarding electromagnetic compatibility, security distances, cable cross-section or material, must be observed.

12.1 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 the ISO 11898-2 standard offers significant advantages:

- Reliable operation due to proven design specifications
- Minimization of error sources due to sufficient distance to the 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!**



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 12.3 ensures the EU Conformity as well.

1	A suitable cable type with a wave impedance of about $120\ \Omega \pm 10\%$ with an adequate conductor cross-section ($\geq 0.22\ \text{mm}^2$) 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: <ul style="list-style-type: none"> • 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 $120\ \Omega \pm 10\%$) at both ends (between the signals CAN_L and CAN_H and not at CAN_GND).
5	Keep cable stubs as short as possible ($l < 0.3\ \text{m}$).
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.

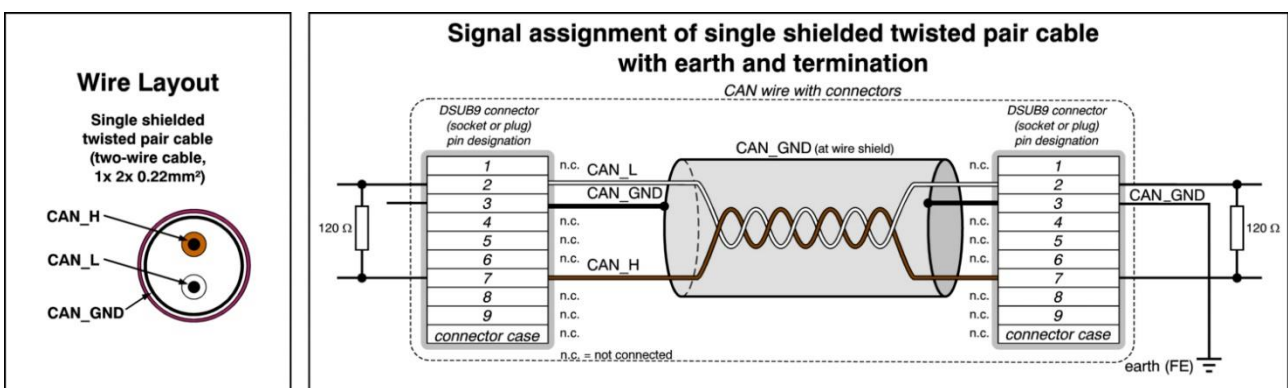


Figure 44: CAN wiring for light industrial environment

12.3 Heavy Industrial Environment (Double Twisted Pair Cable)

12.3.1 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 $120\ \Omega \pm 10\%$ with an adequate conductor cross-section ($\geq 0.22\ \text{mm}^2$) must 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: <ul style="list-style-type: none"> • Two twisted wires must be assigned to the data signals (CAN_H, CAN_L) and • 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 $120\ \Omega \pm 10\%$) at both ends (between the signals CAN_L and CAN_H and not to CAN_GND).
5	Keep cable stubs as short as possible ($l < 0.3\ \text{m}$).
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.

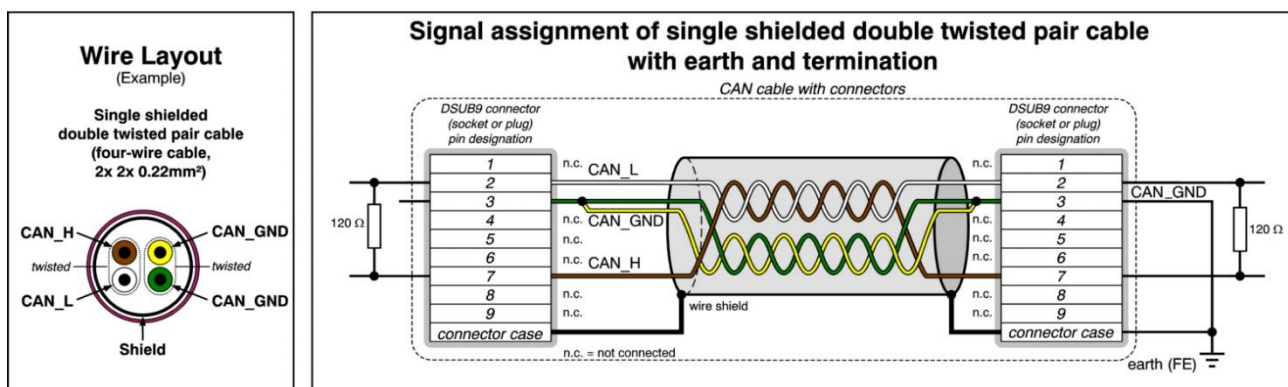


Figure 46: CAN wiring for heavy industrial environment

12.3.2 Device Cabling

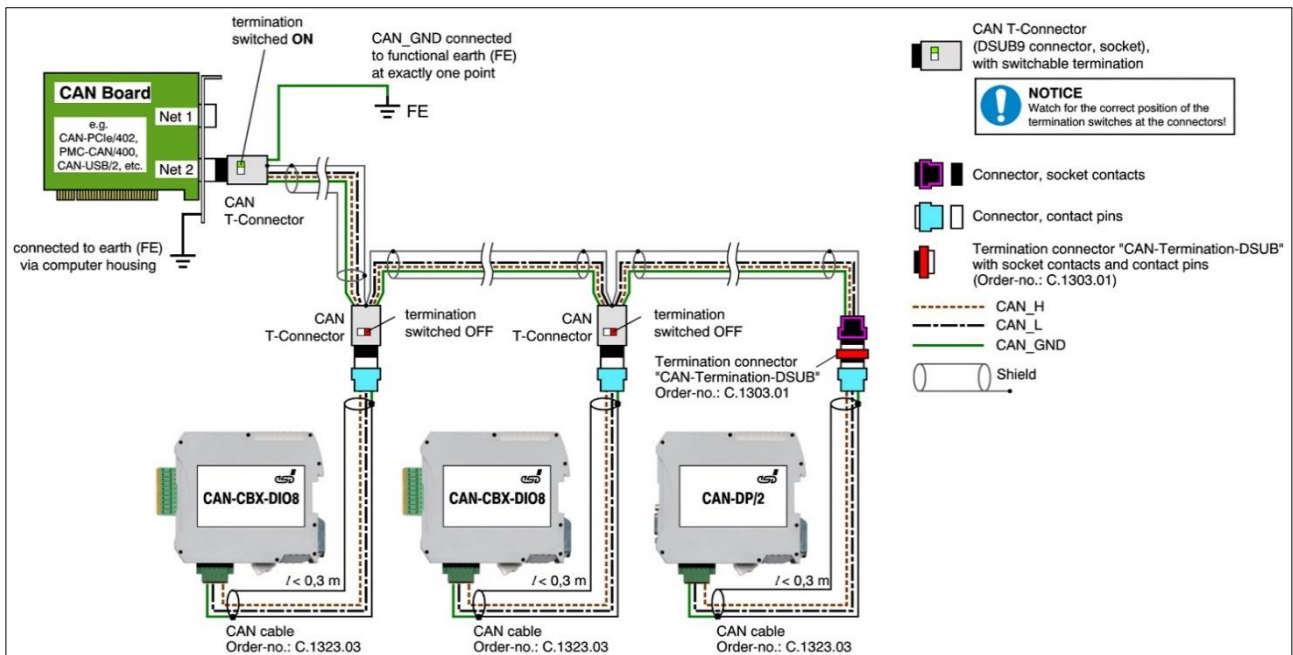


Figure 47: Example of proper wiring with single shielded double twisted pair cables

12.3.3 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 via 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.

12.3.4 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).

12.4 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).

12.5 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 [mm ²]
1000	37	25	0.25 to 0.34
800	59	50	0.34 to 0.6
666. $\overline{6}$	80	-	
500	130	100	
333. $\overline{3}$	180	-	
250	270	250	
166	420	-	0.5 to 0.6
125	570	500	
100	710	650	0.75 to 0.8
83. $\overline{3}$	850	-	
66. $\overline{6}$	1000	-	
50	1400	1000	
33. $\overline{3}$	2000	-	not defined in CiA 303-1
20	3600	2500	
12.5	5400	-	
10	7300	5000	

Table 69: 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 ISO 11898-2. 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”.

12.6 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.

12.6.1 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.)

12.6.2 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**.

13 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.

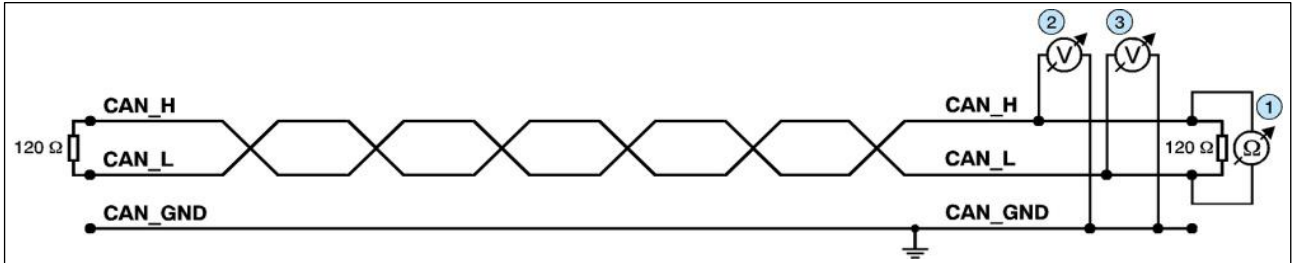


Figure 48: Simplified diagram of a CAN network

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 the 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 determined 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).

13.1 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.

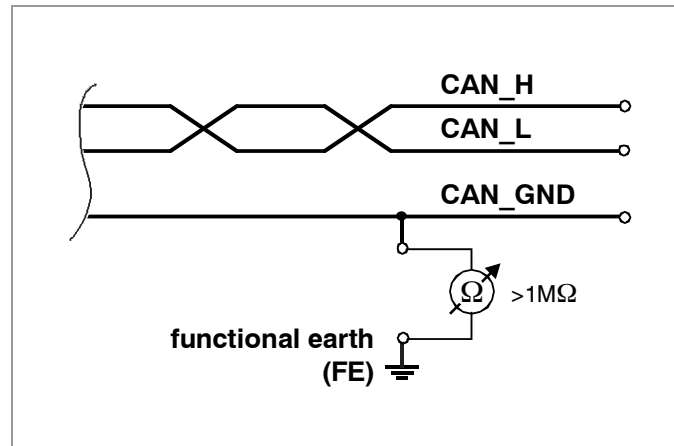
Please note that this test can only be performed with electrically isolated CAN nodes.

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 49: Simplified schematic diagram of ground test measurement



Expected result:

The measured resistance should be greater than 1 MΩ. If it is smaller, please search for additional grounding of the CAN_GND wires.

13.2 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!

13.3 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 a CAN network” on previous page).
4. Measure the DC voltage between CAN_L and CAN_GND, measuring point ③. (See “Simplified diagram of a 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 the voltage 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).

13.4 CAN Transceiver Resistance Test

CAN transceivers have circuits that control CAN_H and CAN_L. Experience shows that electrical damage can 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 CAN 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.)

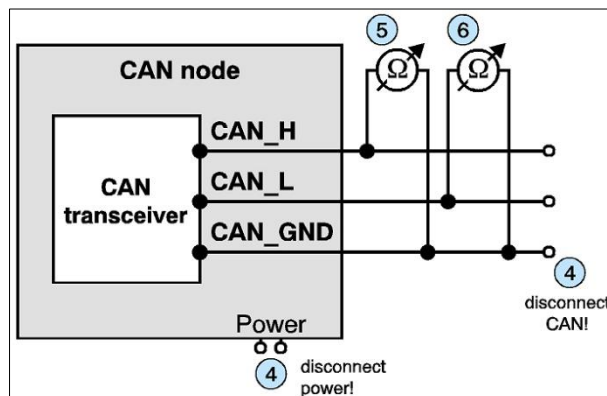


Figure 50: Measuring the internal resistance of CAN transceivers

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 %).

13.5 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**.

14 References

- (1) PROFIBUS International Document TC2-09-0002, (CANopen-Integration_7012_V10_Mar11), PROFIBUS Nutzerorganisation e.V., 76131 Karlsruhe, Germany, V 1.0 2011
- (2) CiA 309-1 Draft Standard, Interfacing CANopen with TCP/IP, Part 1: General principles and services, CAN in Automation (CiA) e. V., Nürnberg, Germany, V 1.1 (12 2006)
- (3) IEEE Standard for Ethernet, IEEE Std 802.3™-2015, IEEE Standards Association, New York, USA,
- (4) CiA 302-2, Draft Standard Proposal, CANopen additional application layer, Part 2: Network management, CAN in Automation (CiA) e. V., Nürnberg, Germany, V.4.1 (02.2009)
- (5) CiA 301, CANopen Application Layer and Communication Profile, CAN in Automation (CiA) e. V., Nürnberg, Germany, V4.2.0 (02.2011)
- (6) CiA 306, 306 WD, Electronic device description, Part 1: Electronic Data Sheet and Device Configuration File, CAN in Automation (CiA) e. V. Nürnberg, Germany, V1.3.3 (08.2010)
- (7) CiA 303-3 Draft Recommendation, CANopen Additional specification Part 3: Indicator specification, CAN in Automation e. V., Nürnberg, Germany, V.1.3 2006
- (8) esd electronics, "Source Code License Agreement", Hannover, esd electronics gmbh, Hannover, 2019-01-21, Rev. 1.1

15 Software Licenses



NOTICE

The software used for the CANopen-PN/2 from esd and from third parties is subject to licenses.

You must read and accept these license conditions before the installation!

The license terms of esd (esd electronics License Conditions) and of 3rd parties (3rd Party Licenses) are displayed and installed on your system during installation via the installation program (CANopen-PN_2_X_X_X.exe, see chapter 5.3).

You can also download the licenses conditions from our website, see the following chapters.

15.1 3rd Party Software License Terms

Lizenz-Name
License Conditions for Siemens Profinet Stack
Apache-2.0
BSD-2-Clause
BSD-3-Clause
BSD-4-Clause
bzip2-1.0.4
GPL-2.0
GPL-3.0
GPL-3.0-with-GCC-exception
ISC
LGPL-2.0
LGPL-2.1
LGPL-3.0
MIT
Spencer-94
TI-TFL
TI-TSPA
Unicode-DFS-2016

15.2 Licence Conditions of the Software Modules

15.2.1 Yocto-Linux License Modules

PACKAGE NAME: amx3-cm3 PACKAGE VERSION: 1.9.2 RECIPE NAME: amx3-cm3 LICENSE: TI-TSPA	PACKAGE NAME: bash PACKAGE VERSION: 5.1.16 RECIPE NAME: bash LICENSE: GPL-3.0-or-later PACKAGE NAME: busybox PACKAGE VERSION: 1.35.0 RECIPE NAME: busybox LICENSE: GPL-2.0-only & bzip2-1.0.4	LICENSE: GPL-2.0-only & bzip2-1.0.4 PACKAGE NAME: busybox-udhcpd PACKAGE VERSION: 1.35.0 RECIPE NAME: busybox LICENSE: GPL-2.0-only & bzip2-1.0.4 PACKAGE NAME: busybox-udhcpd PACKAGE VERSION: 1.35.0 RECIPE NAME: busybox LICENSE: GPL-2.0-only & bzip2-1.0.4	PACKAGE NAME: coreutils PACKAGE VERSION: 9.0 RECIPE NAME: coreutils LICENSE: GPL-3.0-or-later PACKAGE NAME: coreutils-stdbuf PACKAGE VERSION: 9.0 RECIPE NAME: coreutils LICENSE: GPL-3.0-or-later PACKAGE NAME: eudev PACKAGE VERSION: 3.2.10 RECIPE NAME: eudev LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later	PACKAGE NAME: glibc PACKAGE VERSION: 2.35 RECIPE NAME: glibc LICENSE: GPL-2.0-only & LGPL-2.1-only PACKAGE NAME: gmp PACKAGE VERSION: 6.2.1 RECIPE NAME: gmp LICENSE: GPL-2.0-or-later LGPL-3.0-or-later PACKAGE NAME: gnupg PACKAGE VERSION: 2.3.4	RECIPE NAME: gnupg LICENSE: GPL-3.0-only & LGPL-3.0-only PACKAGE NAME: gnupg-gpg PACKAGE VERSION: 2.3.4 RECIPE NAME: gnupg LICENSE: GPL-3.0-only & LGPL-3.0-only PACKAGE NAME: gnutls PACKAGE VERSION: 3.7.4 RECIPE NAME: gnutls LICENSE: LGPL-2.1-or-later
PACKAGE NAME: base-files PACKAGE VERSION: 3.0.14 RECIPE NAME: base-files LICENSE: GPL-2.0-only	PACKAGE NAME: base-passwd PACKAGE VERSION: 3.5.29 RECIPE NAME: base-passwd LICENSE: GPL-2.0-only				

Software Licenses

<p>PACKAGE NAME: init-fupdown PACKAGE VERSION: 1.0 RECIPE NAME: init-fupdown LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: init-system- helpers-service PACKAGE VERSION: 1.62 RECIPE NAME: init-system- helpers LICENSE: BSD-3-Clause & GPL- 2.0-only</p> <p>PACKAGE NAME: initscripts PACKAGE VERSION: 1.0 RECIPE NAME: initscripts LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: initscripts- functions PACKAGE VERSION: 1.0 RECIPE NAME: initscripts LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: inotify-tools PACKAGE VERSION: 3.22.1.0 RECIPE NAME: inotify-tools LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-base PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel- devicetree PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-image PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-image- fitimage PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- auth-rpccss-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- cdc-acm-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- dwc3-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- dwc3-omap-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- ehci-hcd-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- ehci-omap-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- ehci-platform-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p>	<p>PACKAGE NAME: kernel-module- g-ether-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- g-mass-storage-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- irq-pruss-intc-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- libcomposite-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- md-deadline-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- nfsv3-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- nfsv3-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- nfsv3-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- oid-registry-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- ohci-hcd-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- oid-phy-generic-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- pru-rproc-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- prueth-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- pruss-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- roles-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p>	<p>PACKAGE NAME: kernel-module- scsi-mod-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- sd-mod-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- tun-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- u-ether-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- uas-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- udc-core-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- uio-module-drv-5.4.28-rt19+ PACKAGE VERSION: 2.3.1.0+gitAUTOINC+78c535afe8 RECIPE NAME: uio-module-drv LICENSE: BSD-3-Clause</p> <p>PACKAGE NAME: kernel-module- uio-pruss-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- usb-common-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- usb-f-ecm-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- usb-f-ecm-subset-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- usb-f-eeem-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- usb-f-mass-storage-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- usb-f-rndis-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- usb-otg-fsm-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p>	<p>PACKAGE NAME: kernel-module- usb-storage-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- usbcore-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- xhci-hcd-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel-module- xhci-plat-hcd-5.4.28-rt19+ PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kernel- modules PACKAGE VERSION: 5.4.28+gitAUTOINC+c3dd64420d RECIPE NAME: linux-ti-staging-rt LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: kmod PACKAGE VERSION: 29 RECIPE NAME: kmod LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later</p> <p>PACKAGE NAME: ldconfig PACKAGE VERSION: 2.35 RECIPE NAME: glibc LICENSE: GPL-2.0-only & LGPL- 2.1-only</p> <p>PACKAGE NAME: ldd PACKAGE VERSION: 2.35 RECIPE NAME: glibc LICENSE: GPL-2.0-only & LGPL- 2.1-only</p> <p>PACKAGE NAME: libarchive PACKAGE VERSION: 3.6.1 RECIPE NAME: libarchive LICENSE: BSD-2-Clause</p> <p>PACKAGE NAME: libassuan PACKAGE VERSION: 2.5.5 RECIPE NAME: libassuan LICENSE: LGPL-2.1-or-later</p> <p>PACKAGE NAME: libattr PACKAGE VERSION: 2.5.1 RECIPE NAME: attr LICENSE: LGPL-2.1-or-later</p> <p>PACKAGE NAME: libcap PACKAGE VERSION: 2.63 RECIPE NAME: libcap LICENSE: BSD-3-Clause GPL- 2.0-only</p> <p>PACKAGE NAME: libcap-ng PACKAGE VERSION: 0.8.2 RECIPE NAME: libcap-ng LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later</p> <p>PACKAGE NAME: libcrypto PACKAGE VERSION: 3.0.3 RECIPE NAME: openssl LICENSE: Apache-2.0</p> <p>PACKAGE NAME: libgcc PACKAGE VERSION: 11.2.0 RECIPE NAME: libgcc LICENSE: GPL-3.0-with-GCC- exception</p> <p>PACKAGE NAME: libgpg-error PACKAGE VERSION: 1.44 RECIPE NAME: libgpg-error LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later</p> <p>PACKAGE NAME: libidn2 PACKAGE VERSION: 2.3.2 RECIPE NAME: libidn2</p>	<p>LICENSE: (GPL-2.0-or-later LGPL-3.0-only) & Unicode-DFS- 2016</p> <p>PACKAGE NAME: libinotifytools PACKAGE VERSION: 3.22.1.0 RECIPE NAME: inotify-tools LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: libkmod PACKAGE VERSION: 29 RECIPE NAME: kmod LICENSE: LGPL-2.1-or-later</p> <p>PACKAGE NAME: libksba PACKAGE VERSION: 1.6.0 RECIPE NAME: libksba LICENSE: GPL-2.0-or-later LGPL-3.0-or-later</p> <p>PACKAGE NAME: libopkg PACKAGE VERSION: 0.5.0 RECIPE NAME: opkg LICENSE: GPL-2.0-or-later</p> <p>PACKAGE NAME: libsolv PACKAGE VERSION: 0.7.22 RECIPE NAME: libsolv LICENSE: BSD-3-Clause</p> <p>PACKAGE NAME: libstdc++ PACKAGE VERSION: 11.2.0 RECIPE NAME: gcc-runtime LICENSE: GPL-3.0-with-GCC- exception</p> <p>PACKAGE NAME: libubootenv PACKAGE VERSION: 0.3.2 RECIPE NAME: libubootenv LICENSE: LGPL-2.1-only</p> <p>PACKAGE NAME: libubootenv-bin PACKAGE VERSION: 2.35 RECIPE NAME: libubootenv LICENSE: LGPL-2.1-only</p> <p>PACKAGE NAME: libxcrypt PACKAGE VERSION: 4.4.28 RECIPE NAME: libxcrypt LICENSE: LGPL-2.1-only</p> <p>PACKAGE NAME: libzstd PACKAGE VERSION: 1.5.2 RECIPE NAME: zstd LICENSE: BSD-3-Clause & GPL- 2.0-only</p> <p>PACKAGE NAME: Imsensors- config-libsensors PACKAGE VERSION: 1.0 RECIPE NAME: Imsensors-config LICENSE: MIT</p> <p>PACKAGE NAME: Imsensors- libsensors PACKAGE VERSION: 3.6.0 RECIPE NAME: Imsensors LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later</p> <p>PACKAGE NAME: Imsensors- sensors PACKAGE VERSION: 3.6.0 RECIPE NAME: Imsensors LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later</p> <p>PACKAGE NAME: Isof PACKAGE VERSION: 4.94.0 RECIPE NAME: Isof LICENSE: Spencer-94</p> <p>PACKAGE NAME: Izo PACKAGE VERSION: 2.10 RECIPE NAME: Izo LICENSE: GPL-2.0-or-later</p> <p>PACKAGE NAME: memtool PACKAGE VERSION: 2018.03.0 RECIPE NAME: memtool LICENSE: GPLv2</p> <p>PACKAGE NAME: modutils- initscripts PACKAGE VERSION: 1.0 RECIPE NAME: modutils- initscripts LICENSE: MIT</p>	<p>PACKAGE NAME: mtd-utils PACKAGE VERSION: 2.1.4 RECIPE NAME: mtd-utils LICENSE: GPL-2.0-or-later</p> <p>PACKAGE NAME: mtd-utils-ubifs PACKAGE VERSION: 2.1.4 RECIPE NAME: mtd-utils LICENSE: GPL-2.0-or-later</p> <p>PACKAGE NAME: ncurses- libncurses PACKAGE VERSION: 6.3 RECIPE NAME: ncurses LICENSE: MIT</p> <p>PACKAGE NAME: ncurses- libncursesw PACKAGE VERSION: 6.3 RECIPE NAME: ncurses LICENSE: MIT</p> <p>PACKAGE NAME: ncurses-libinfo PACKAGE VERSION: 6.3 RECIPE NAME: ncurses LICENSE: MIT</p> <p>PACKAGE NAME: ncurses- terminfo-base PACKAGE VERSION: 6.3 RECIPE NAME: ncurses LICENSE: MIT</p> <p>PACKAGE NAME: netbase PACKAGE VERSION: 6.3 RECIPE NAME: netbase LICENSE: GPL-2.0-only</p> <p>PACKAGE NAME: nettle PACKAGE VERSION: 3.7.3 RECIPE NAME: nettle LICENSE: LGPL-3.0-or-later GPL-2.0-or-later</p> <p>PACKAGE NAME: nph PACKAGE VERSION: 1.6 RECIPE NAME: nph LICENSE: LGPL-2.0-or-later</p> <p>PACKAGE NAME: openssl PACKAGE VERSION: 8.9p1 RECIPE NAME: openssl LICENSE: BSD-2-Clause & BSD- 3-Clause & ISC & MIT</p> <p>PACKAGE NAME: openssl- keygen PACKAGE VERSION: 8.9p1 RECIPE NAME: openssl LICENSE: BSD-2-Clause & BSD- 3-Clause & ISC & MIT</p> <p>PACKAGE NAME: openssl-scp PACKAGE VERSION: 8.9p1 RECIPE NAME: openssl LICENSE: BSD-2-Clause & BSD- 3-Clause & ISC & MIT</p> <p>PACKAGE NAME: openssl-ssh PACKAGE VERSION: 8.9p1 RECIPE NAME: openssl LICENSE: BSD-2-Clause & BSD- 3-Clause & ISC & MIT</p> <p>PACKAGE NAME: openssl-sshhd PACKAGE VERSION: 8.9p1 RECIPE NAME: openssl LICENSE: BSD-2-Clause & BSD- 3-Clause & ISC & MIT</p> <p>PACKAGE NAME: openssl-conf PACKAGE VERSION: 3.0.3 RECIPE NAME: openssl LICENSE: Apache-2.0</p> <p>PACKAGE NAME: openssl-oss- module-legacy PACKAGE VERSION: 3.0.3 RECIPE NAME: openssl LICENSE: Apache-2.0</p> <p>PACKAGE NAME: opkg PACKAGE VERSION: 0.5.0 RECIPE NAME: opkg LICENSE: GPL-2.0-or-later</p> <p>PACKAGE NAME: opkg-arch- config PACKAGE VERSION: 1.0 RECIPE NAME: opkg-arch-config LICENSE: MIT</p> <p>PACKAGE NAME: os-release PACKAGE VERSION: 1.0 RECIPE NAME: os-release</p>
--	---	---	---	---	---

Software Licenses

[illegible]

Software Licenses

LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-wall PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause
PACKAGE NAME: util-linux-nologin PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-renice PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-scriptreplay PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-swaplabel PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-wdctl PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause
PACKAGE NAME: util-linux-nsenter PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-resizepart PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-setarch PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-swapon PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-whereis PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause
PACKAGE NAME: util-linux-partx PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-rev PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-setpriv PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-swapon PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-wipefs PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause
PACKAGE NAME: util-linux-pivot-root PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-rfkill PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-setsid PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-switch-root PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-write PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause
PACKAGE NAME: util-linux-primit PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-rtwake PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-setterm PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-taskset PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-zramctl PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause
PACKAGE NAME: util-linux-readprofile PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-script PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-sfdisk PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-uclampset PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-zlog PACKAGE VERSION: 1.2.15 RECIPE NAME: zlog LICENSE: LGPL-2.1-only
PACKAGE NAME: util-linux-rename PACKAGE VERSION: 2.37.4	PACKAGE NAME: util-linux-scriptlive PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux	PACKAGE NAME: util-linux-slogin PACKAGE VERSION: 2.37.4	PACKAGE NAME: util-linux-uclampsset PACKAGE VERSION: 2.37.4 RECIPE NAME: util-linux LICENSE: GPL-2.0-or-later & LGPL-2.1-or-later & BSD-3-Clause & BSD-4-Clause	PACKAGE NAME: util-linux-zlog PACKAGE VERSION: 1.2.15 RECIPE NAME: zlog LICENSE: LGPL-2.1-only

15.2.2 Others

NAME: U-Boot
VERSION: v2020.01
LICENSE: GPL-2.0-or-later

NAME: PROFINET Stack-
Lizenzbedingungen
VERSION: 2011-08-01
LICENSE: License Conditions for
Siemens Profinet Stack

15.2.3 Open Source Software Copy

You may obtain a copy of the source code, if and as required under the license by sending a mail to oss-compliance@esd.eu

You may also obtain a copy of the source code, if and as required under the license, by sending a check or money of EUR 25.00 to:

esd electronics gmbh
Vahrenwalder Str. 207
30165 Hannover, Germany

16 Declaration of Conformity

EU-KONFORMITÄTSERKLÄRUNG EU DECLARATION OF CONFORMITY



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

esd erklärt, dass das Produkt
esd declares, that the product

CAN-PN/2

CAN-PN/2-FD

CANopen-PN/2

CAN-CBX-CPU/A9-1CFBRJ45

Typ, Modell, Artikel-Nr.
Type, Model, Article No.

C.2924.02

C.2924.62

C.2931.02

C.3073.01

die Anforderungen der Normen
fulfills the requirements of the standards

EN 61000-6-2:2005,
EN 61000-6-4:2007/A1:2011

gemäß folgendem Prüfbericht erfüllt.
according to test certificate.

EMVP No.: 0226-202305

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
Funktion / Title
Datum / Date

T. Bielert
QM-Beauftragter / QM
Representative
Hannover, 2023-05-22

Rechtsgültige Unterschrift / authorized signature

\\files\er\pub\prijesd\Gilewsys\FB-GW_HW_Plattform\Test\EMV\EU_Declaration_of_Conformity_IFB-CAN-FD-GW4377_2023-05-22.docx

17 PROFINET IO Certificate



Certificate

PROFIBUS Nutzerorganisation e.V. grants to

esd electronics gmbh

Vahrenwalder Str. 207, 30165 Hannover, Germany

the Certificate No: **Z13447** for the PROFINET Device:

Model Name: CANopen-PN/2
Revision: SW/FW: V3.0.0; HW: 301
Identnumber: 0x015D; 0x0002
GSD: GSDML-V2.42-#esd-CANopen-PN_CBX-20220605.xml
DAP: Linking Device: CANopen-PN/2; 0x20000101

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

<input checked="" type="checkbox"/> PNIO_Version	V2.4
<input checked="" type="checkbox"/> Conformance Class	B
<input checked="" type="checkbox"/> Optional Features	Legacy
<input checked="" type="checkbox"/> Netload Class	I
<input checked="" type="checkbox"/> PNIO_Tester_Version	Version V2.42.1
<input checked="" type="checkbox"/> Tester	SIEMENS AG, Fürth, Germany; PN728-1

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 **August 04, 2025** the certificate is valid for life.

Karlsruhe, November 07, 2022

Board of PROFIBUS Nutzerorganisation e. V.

(Official in Charge)

(Karsten Schneider)

(Frank Moritz)



18 Order Information

Type	Properties	Order No.
CANopen-PN/2	<p>High-performance PROFINET-IO-Device to CANopen Manager Gateway.</p> <p>CAN-Physical-Layer according to ISO-11898-2 and PROFINET-Physical -Layer 100BASE-TX according to IEEE802.3 with galvanic isolation on both sides.</p> <p>Compact housing for DIN rail mounting with easily accessible connectors.</p> <p>System integration using the esd GSDML Composer for individual generation of the matching configuration.</p> <p>Extensive debugging with CAN diagnostic software (CANreal, CANplot and COBview) via USB interface.</p>	C.2931.02

Table 70: Order information hardware

PDF Manuals

For the availability of the manuals see table below.

Please download the manuals as PDF documents from our esd website <https://www.esd.eu> for free.

Manuals		Order No.
CANopen-PN/2-ME	Hardware and software manual for CANopen-PN/2 in English	C.2931.21
CAN-API-ME	NTCAN-API, Part 1: Application Developers Manual NTCAN-API, Part 2: Driver Installation Guide	C.2001.21
CANopen-ME	CANopen Manuals in English	C.2002.21

Table 71: 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.