CAN-EtherCAT

EtherCAT-CAN Gateway

Manual

to Product C.2922.02

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

N O T E

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

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

esd electronic system design gmbh
Vahrenwalder Str. 207
30165 Hannover
Germany

Phone: +49-511-372 98-0

Fax: +49-511-372 98-68

E-Mail: info@esd.eu

Internet: www.esd.eu

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

Document file:	I:\Texte\Doku\MANUALS\CAN\CAN-EtherCAT\Englisch\CAN-EtherCAT_manual_en_14.odt
Date of print:	2017-02-22
Document type number:	DOC0800

Firmware version:	from Version 2.3

Document History

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

Revision	Chapter	Changes versus previous version	Date
1.3	-	Classification of Safety Information inserted and Safety messages revised	2016-04-08
	2.	Note concerning connectors inserted	
	3.	Safety message inserted	
	5.1.5.1	Figure about functional principle of the CAN-Rx-Message-Queues inserted and figure about transmission CAN-message -> EtherCAT-message	
	5.1.5.3	Figure about functional principle of the CAN-Tx-Message-Queues inserted and note to figure about transmission CAN-message -> EtherCAT-message	
	5.1.4.1, 5.1.5.3, 5.1.5.4, 5.1.5.5	Note inserted - some write accesses only in PreOp	
	5.3.1	New chapter: “Firmware Update with the esd Workbench”	
	5.3.2	Figure 27 new- with Slave in Bootstrap	
	8,1, 8.2	Note to chapter 8.7 inserted	
	8.7	New chapter „Conductor Connection/Conductor Cross Sections“	
	9	Updated chapter 'Correct Wiring of Electrically Isolated CAN Networks'	
	10	Updated chapter 'CAN Troubleshooting Guide'	
	12	Declaration of Conformity new	
	13	Updated chapter 'Order Information'	
1.4	-	Service Note revised	2017-02-22
	5.1.5.1, 5.1.5.2	Chapter revised, Figure with sequence of Rx-Counter inserted	
	5.1.5.3, 5.1.5.4	Chapter revised, Figure with sequence of Tx-Counter inserted	

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 and property damage.

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

DANGER, WARNING, CAUTION

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

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

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

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

NOTICE

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

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

INFORMATION

INFORMATION
Notes to point out something important or useful.

Safety Instructions

- When working with the CAN-EtherCAT follow the instructions below and read the manual carefully to protect yourself from injury and the CAN-EtherCAT from damage.
- Do not use damaged or defective cables to connect the CAN-EtherCAT 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.
- Current circuits which are connected to the device have to be sufficiently protected against hazardous voltage (SELV according to EN 60950-1).
- The CAN-EtherCAT may only be driven by power supply current circuits, that are contact protected.
  A power supply, that provides a safety extra-low voltage (SELV) according to EN 60950-1, complies with this conditions.

- Do not open the housing of the CAN-EtherCAT.
- The CAN-EtherCAT has to be securely installed before commissioning.
- Never let liquids get inside the CAN-EtherCAT. Otherwise, electric shocks or short circuits may result.
- Protect the CAN-EtherCAT from dust, moisture and steam.
- Protect the CAN-EtherCAT from shocks and vibrations.
- The CAN-EtherCAT may become warm during normal use. Always allow adequate ventilation around the CAN-EtherCAT and use care when handling.
- Do not operate the CAN-EtherCAT adjacent to heat sources and do not expose it to unnecessary thermal radiation. Ensure an ambient temperature as specified in the technical data.

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 CAN-EtherCAT is to be integrated.

→ All current circuits which are connected to the device have to be sufficiently protected against hazardous voltage (SELV according to EN 60950-1) before you start with the installation.

Qualified Personal

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

Conformity

The CAN-EtherCAT 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 CAN-EtherCAT may cause radio interference 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 CAN-EtherCAT is the operation as CAN-EtherCAT 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 CAN-EtherCAT is intended for indoor use.
- The operation of the CAN-EtherCAT in hazardous areas, or areas exposed to potentially explosive materials is not permitted.
- The operation of the CAN-EtherCAT for medical purposes is prohibited.

Service Note

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

Disposal

Devices which have become defective in the long run have to be disposed in an appropriate way or have to be returned to the manufacturer for proper disposal. Please, make a contribution to environmental protection.

Typographical Conventions

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

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

Number Representation

All numbers in this document are base 10 unless designated otherwise. Hexadecimal numbers a followed by "$_h$". For example, 42 is represented as $2A_h$ in hexadecimal.

Table of contents

1. Overview

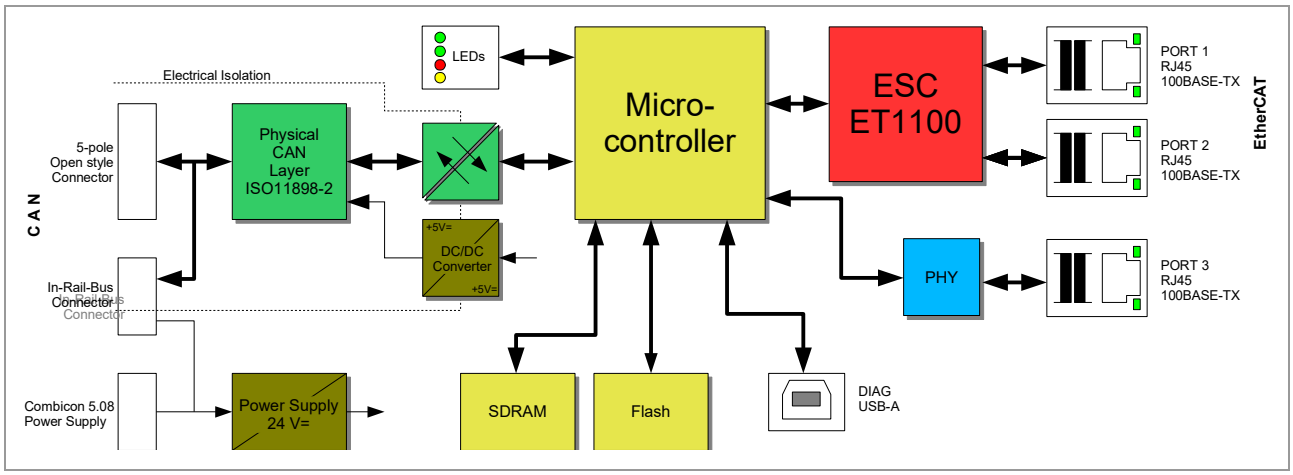


Figure 1: Block circuit diagram

The CAN-EtherCAT device connects an EtherCAT® network with one CAN network. In this case the gateway acts as an EtherCAT slave device according to „Module Profile Number 5000“ of the „Modular Device Profile Description“ (ETG.5001 documents).

The CAN-EtherCAT gateway allows CAN modules with CANopen® (CiA® DS 301) or Layer 2 (ISO 11898-1) implementations to connect with a real-time EtherCAT network. The gateway does not limit the number of CAN nodes.

The CAN-EtherCAT is also configurable as “Ethernet Switch Port” by Ethernet over EtherCAT (EoE), see section 5.2.

The high-speed CAN interface is compliant with ISO 11898-2 and it supports transfer rates from 50 kBit/s up to 1 MBit/s. The 100BASE-TX EtherCAT interface is IEEE802.3 compatible and runs at 100 MBit/s. The CAN interface, as well as the EtherCAT interface, is electrically isolated.

The configuration of the CAN-EtherCAT is accomplished through the EtherCAT master.
CAN diagnostics and firmware updates are realized via web interface. (Firmware updates are also possible via File access over EtherCAT (FoE), see section 5.3)

2. Hardware Installation

2.1 Connections

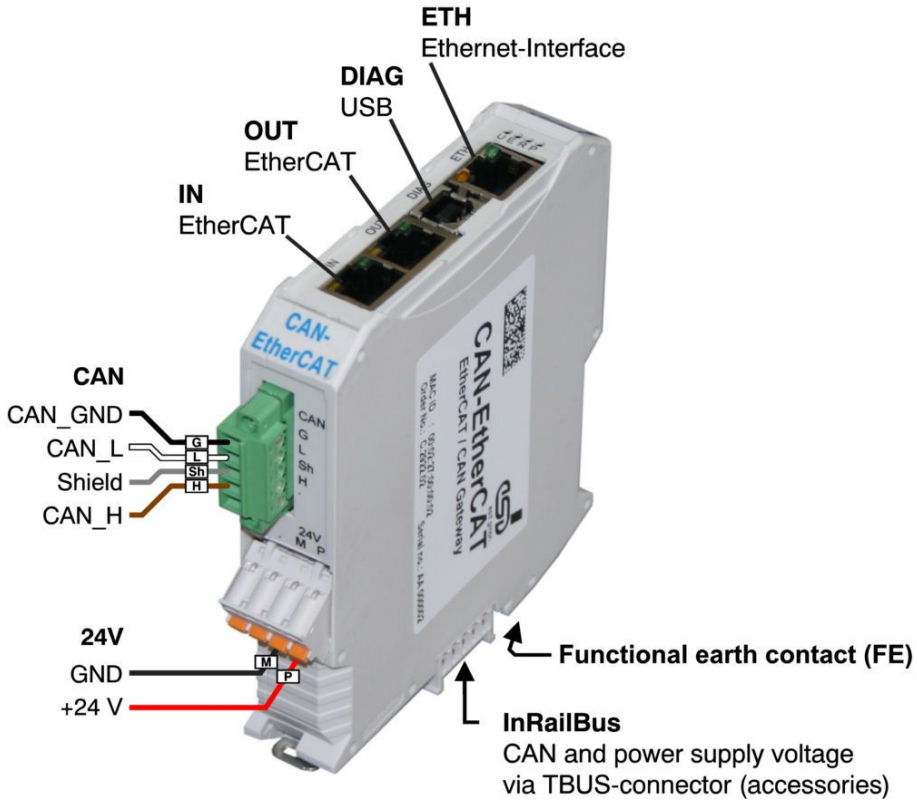


Figure 2: Connections for operating condition

NOTICE
Read chapter “Hardware Installation ” on page 14, before you start with the installation of the hardware!
Please refer to page 68 ff. for signal assignment of the connectors.

INFORMATION
The “DIAG” USB interface is currently available only for internal use at the factory!

2.2 LEDs

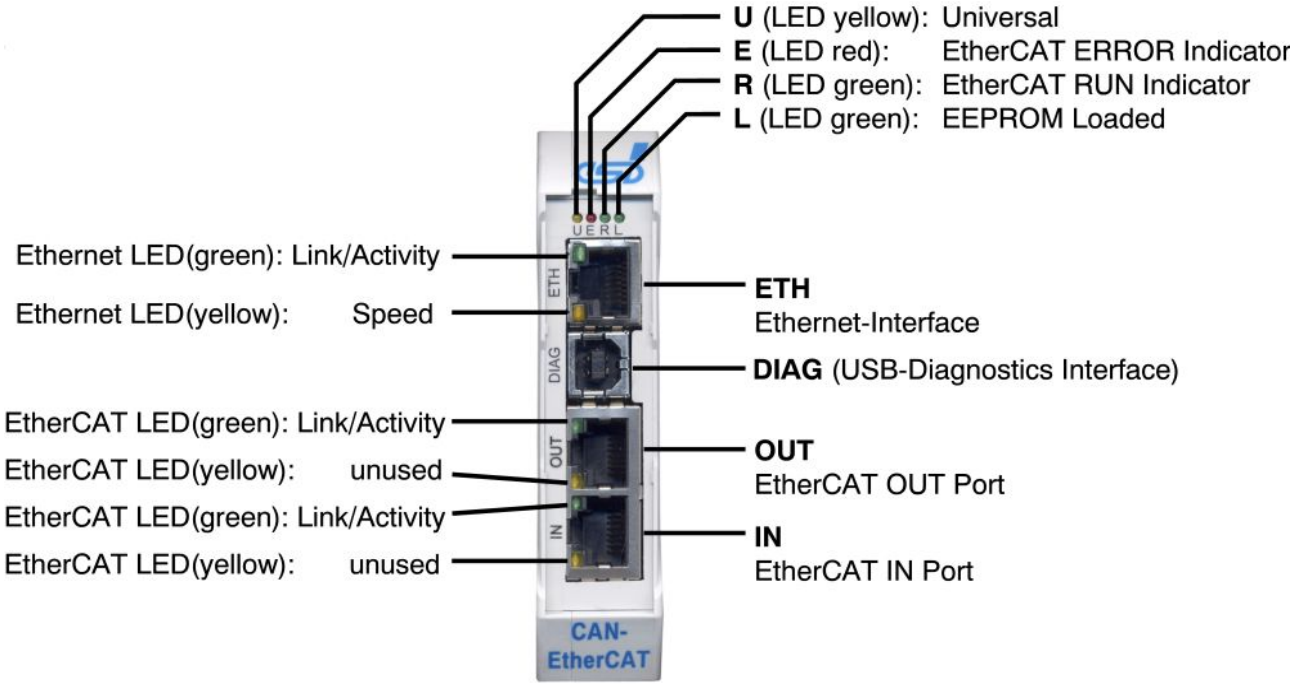


Figure 3: Connectors and LEDs

2.2.1 LED Assignment

LEDs at Ethernet RJ45 connector *ETH:*

LED	Color	LED Status	Description
Link/ Activity	green	off	no Ethernet link present
		blinking	Ethernet link present, Ethernet activity (reception of Ethernet data packages)
Speed	yellow	off	10 MBit/s
		on	100 MBit/s

Table 1: Ethernet-LED functionality

LEDs at EtherCAT-RJ45 Connectors *IN* and *OUT*

LED	Color	LED Status	Description
Link/ Activity	green	off	no EtherCAT link present
		blinking	EtherCAT link present, EtherCAT activity (reception of Ethernet data)
Spare	yellow	-	unused

Table 2: EtherCAT LED functionality (integrated in RJ45)

EtherCAT-LEDs *U, E, R, L*

LED Status	Description
blinking	LED repeats: 200 ms on, 200 ms off.
flicker	LED repeats: 50 ms on, 50 ms off.
single flash	LED repeats: 200 ms on, 1000 ms off.
double flash	LED repeats: 200 ms on, 200 ms off, 200 ms on, 1000 ms off.

Table 3: LED states (according to ETG.1300 documents)

LED	Color	Function	LED Status	Description	Schematic Reference
U	yellow	Universal	off	No information available	LED1A
			blinking	Device is in “Local IP Port Mode”, i.e. its Webserver etc. is directly accessible, see see chapter 6.	
			flicker	FoE firmware transfer is in progress	
			on	FoE firmware transfer finished. Visible only for a few seconds – then actual firmware update is started	
			any	Set by CoE object 0x2000.2, see 5.1.4.1	
E	red	EtherCAT ERROR Indicator	off	No error	LED1B
			blinking	State change failed	
			single flash	State changed due to configuration error	
			double flash	SM watchdog time out	
R	green	EtherCAT RUN Indicator*	off	Init	LED1C
			blinking	Pre-Operational	
			single flash	Safe-Operational	
			on	Operational	
			flicker	Bootstrap	
L	green	EEPROM Loaded*	off	unable to retrieve ET1100 configuration from EEPROM	LED1D
			on	successful retrieval of ET1100 configuration from EEPROM	

*Directly connected to the ET1100

Table 4: EtherCAT LED functionality

3. Hardware Installation

For proper installation and setup please follow the recommended steps as shown here:

Step	Procedure	see page
	**Read the safety instructions at the beginning of this document carefully, before you start with the hardware installation!**	5
	**Danger** Hazardous Voltage - Risk of electric shock due to unintentional contact with uninsulated live parts with high voltages inside of the system into which the CAN-EtherCAT is to be integrated. → All current circuits which are connected to the device have to be sufficiently protected against hazardous voltage (SELV according to EN 60950-1) before you start with the installation. → Ensure the absence of voltage before starting any electrical work.	
1.	Mount and connect the CAN-EtherCAT gateway and connect the interfaces (Power supply, CAN bus, EtherCAT, and – if applicable – Ethernet).	11
2.	Please note that the CAN bus has to be terminated at both ends! **esd** offers special T-connectors and termination connectors for external termination. Additionally the CAN_GND signal has to be connected to earth at **exactly one** point in the CAN network. All esd termination devices will provide a corresponding contact. For details please read chapter “Correct Wiring of Electrically Isolated CAN Networks” . Any CAN node that does not support a galvanic isolation represents the equivalent of a Ground (GND) connection.	76
3.	Turn on the 24 V-power supply voltage of the CAN-EtherCAT.	-
4.	Copy the enclosed EtherCAT slave information file (ESI) into the corresponding folder.	20
5.	Configure the CAN-EtherCAT gateway with an EtherCAT configurator.	15

4. Configuration with an EtherCAT Configurator

4.1 CAN-EtherCAT Gateway Application Example

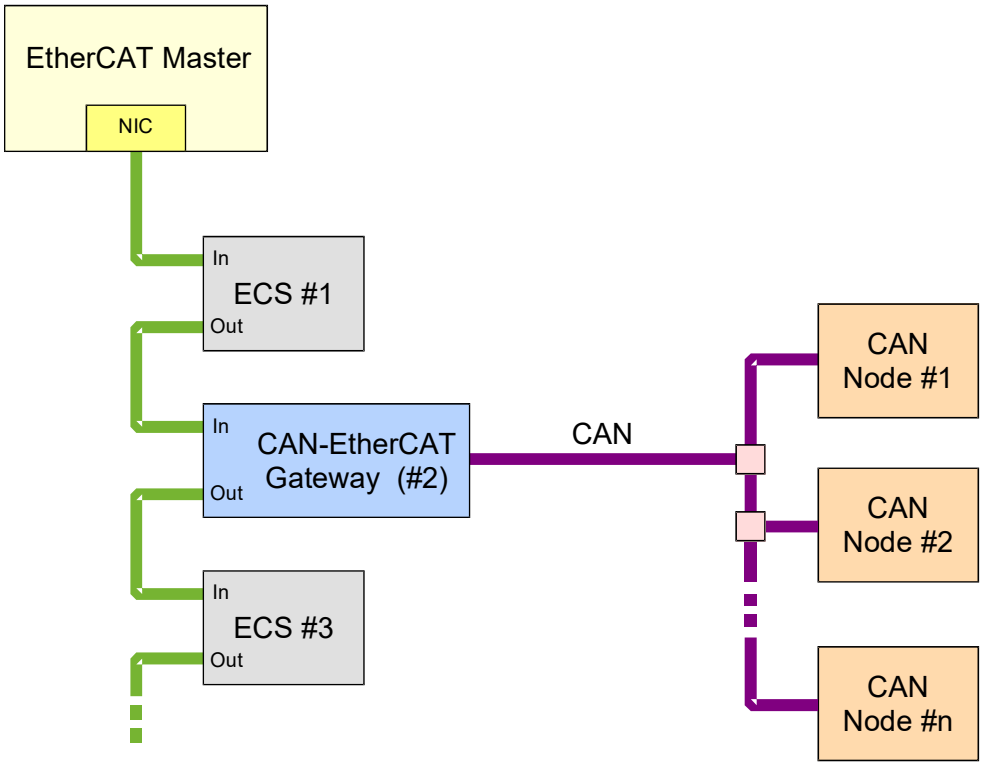


Figure 4: CAN-EtherCAT gateway connection example

The CAN-EtherCAT gateway can take any position in an EtherCAT network.

4.2 Configuration Sequence, esd EtherCAT Workbench

The following chapter describes the CAN interface configuration of the CAN-EtherCAT gateway for example by means of the esd EtherCAT Workbench.

First, the enclosed EtherCAT Slave information file (ESI)

ESD CAN-EtherCAT.xml

must be copied to the corresponding folder.

When the Workbench is running, this can be done by the menu entry “Copy ESI file(s) to slave library” (Under “Tools”), see Figure 5. Otherwise the Workbench's start menu entry “Open slave library folder” can be used to copy the file manually.

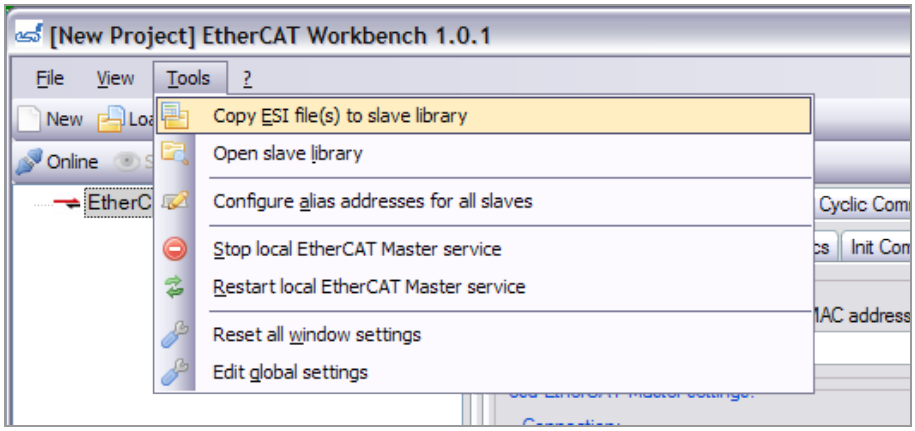


Figure 5: Installing ESI file with the Workbench

Now the Workbench has to be (re)started and a network scan will show the device:

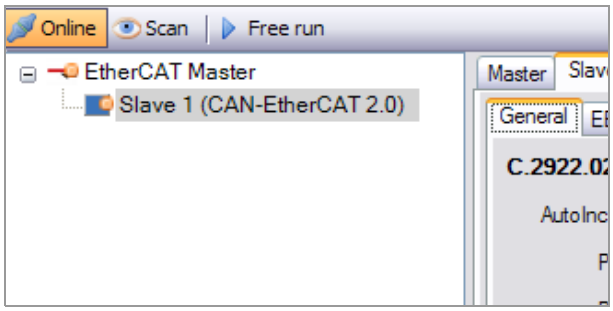


Figure 6: Slave tree view

Now go to the “CoE Dictionary” tab page and recreate the dictionary by the menu item “Recreate dictionary”, “Online from slave” as shown in Figure 7:

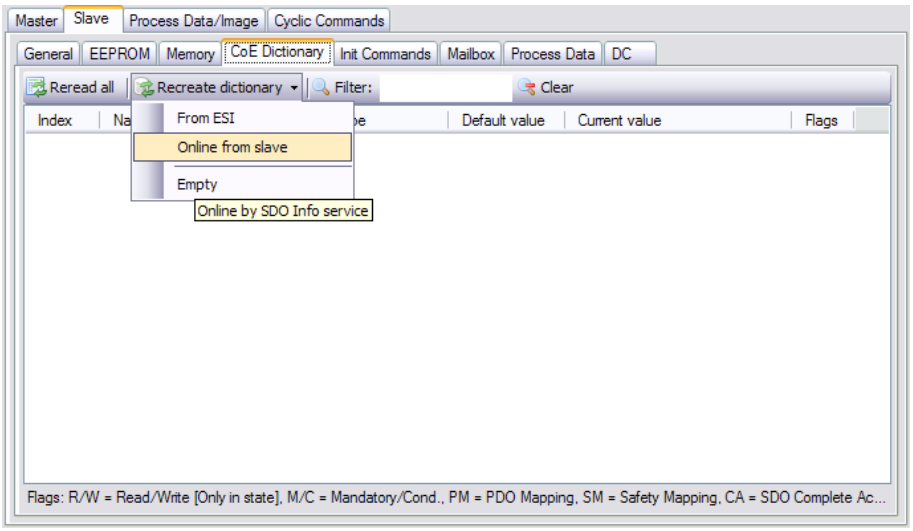


Figure 7: Recreating the CoE dictionary

Click “Reread all” to update the items, then select the “Process Data” tab page.

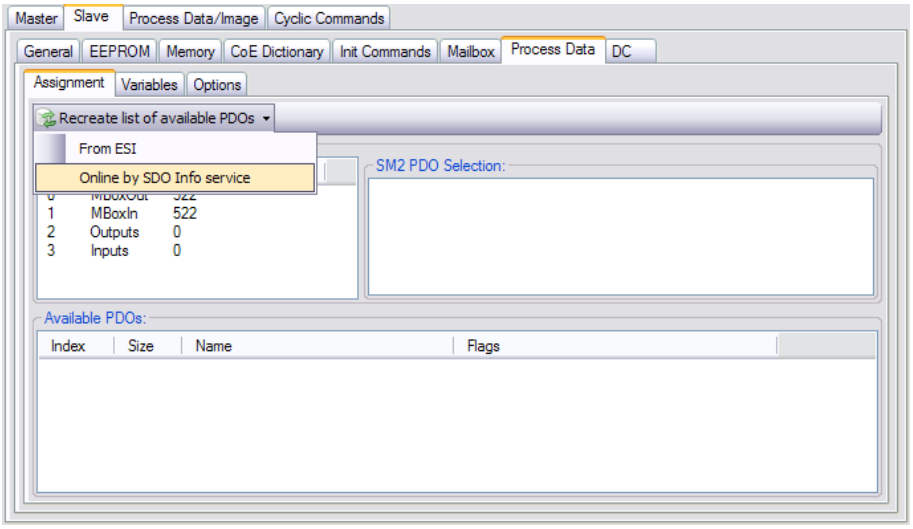
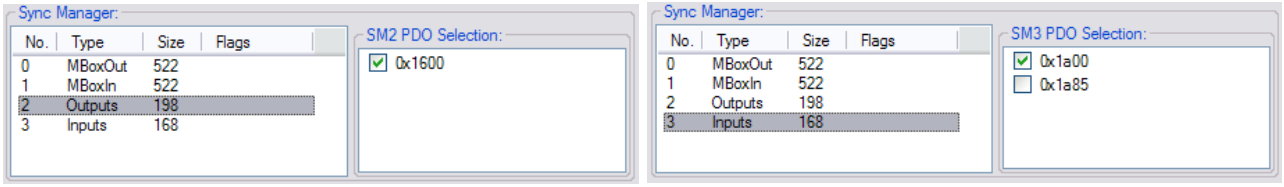


Figure 8: Recreating the list of available PDOs

Recreate list of available PDOs by SDO Info service as shown in Figure 8. As a result the PDOs 0x1600 and 0x1a00 should be mapped:



PDO 0x1600 contains the CAN Tx messages, as described in 5.1.3.7, 0x1a00 contains the Rx messages as described in 5.1.3.8. (The optional PDO 0x1a85 contains CAN Status information, see 5.1.3.9 and 5.1.5.8)

Different queue sizes or 29 bit CAN IDs

If you want to use extended CAN IDs or change Rx- or Tx- queue size, you have to do this in the CoE dictionary, right after you clicked “Reread all” as described above, but it must be done **before** you recreate the list of available PDOs.
To use 29 bit IDs for example, you have to write object 0x8000.20 as described in 5.1.5.5. When the list of available PDOs is recreated afterwards you'll notice a PDO size change (With the standard queue sizes for example, the 198 bytes for the outputs will change to 262 byte).

When changing the queue sizes or CAN ID type this has to be done during slave start up, too. Section 4.2.1 shows how this is done for the CAN baud rate object – this works for other objects as well.

4.2.1 Setting the Baud Rate during Slave Initialization

Go to the “Slave”, “Mailbox”, “CoE” tab page, right click in the init. commands list and select “Append new item”:

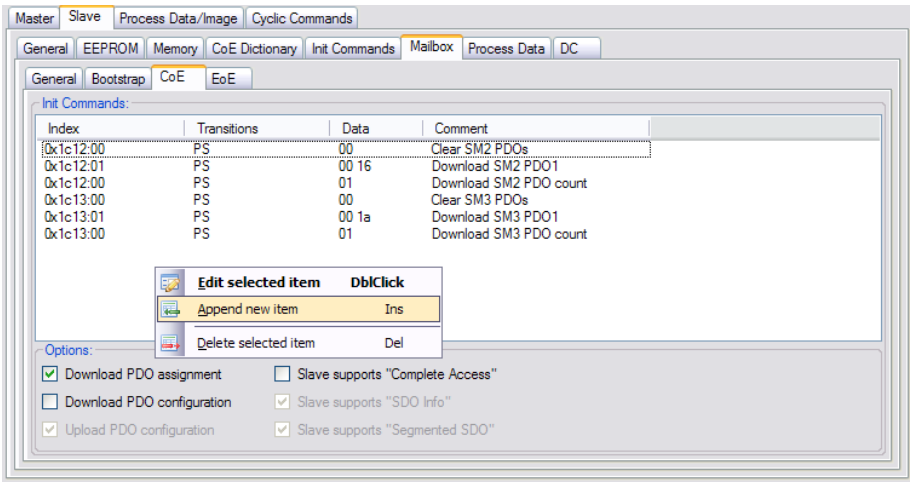


Figure 9: Appending CoE init command

In the following dialog window click “Select from object dict.” and select the baud rate object ($F800_h$:02) in the context menu that appears. Now the dialog should look like this:

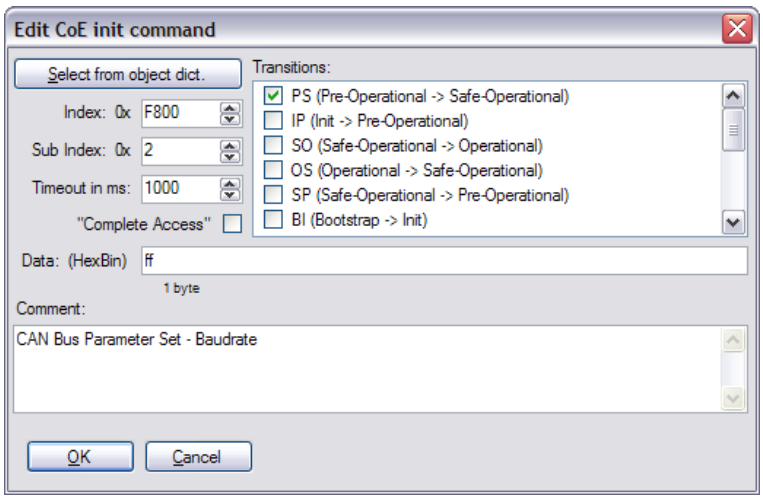


Figure 10: CoE init. command for baud rate object

Just enter the desired baud rate index (described in 5.1.5.9) at the “Data:” input box and leave the dialog with the “OK” button. (Other settings should be left untouched)
Now the “Init Commands” list contains an additional command that sets the baud rate during the slave's “PreOp → SafeOp” transition.

This can be done for other objects, especially the “CAN Interface Configuration” objects (8000_h, see 5.1.5.5), too.

4.2.2 Export ENI

To export the ENI for the EtherCAT network and CAN-EtherCAT you just configured click “Export master configuration file (ENI)” in the “File” menu: (Or press “Ctrl+M” or use the “Export ENI” button in the Workbench's toolbar – they all do the same)

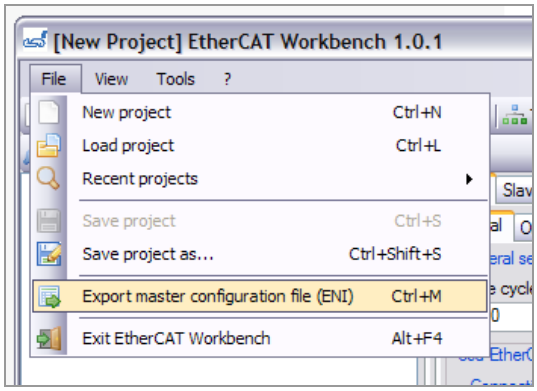


Figure 11: Exporting the ENI

4.3 Configuration Sequence, Beckhoff Configurator

The following chapter describes the CAN interface configuration of the CAN-EtherCAT gateway for example by means of the Beckhoff EtherCAT configurator.

First, the enclosed EtherCAT Slave information file (ESI)

ESD CAN-EtherCAT.xml

must be copied to the corresponding folder.

Using the EtherCAT configurator the folder may be, for example:
“C:\Program Files\EtherCAT Configurator\EtherCAT”.

As soon as the EtherCAT configurator has recognized the CAN-EtherCAT, it will display it in the device tree view:

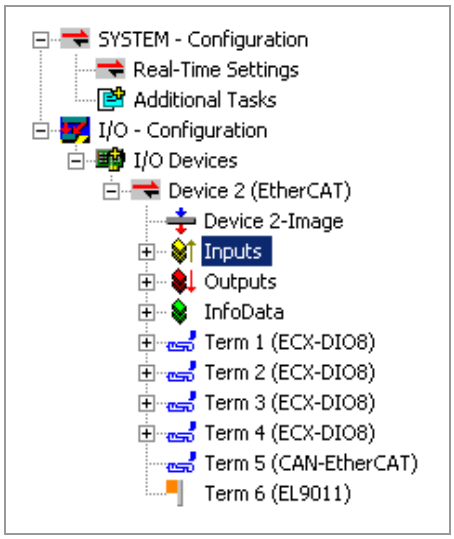


Figure 12: CAN-EtherCAT in device tree view

Use the table CoE-Online to display the object dictionary:

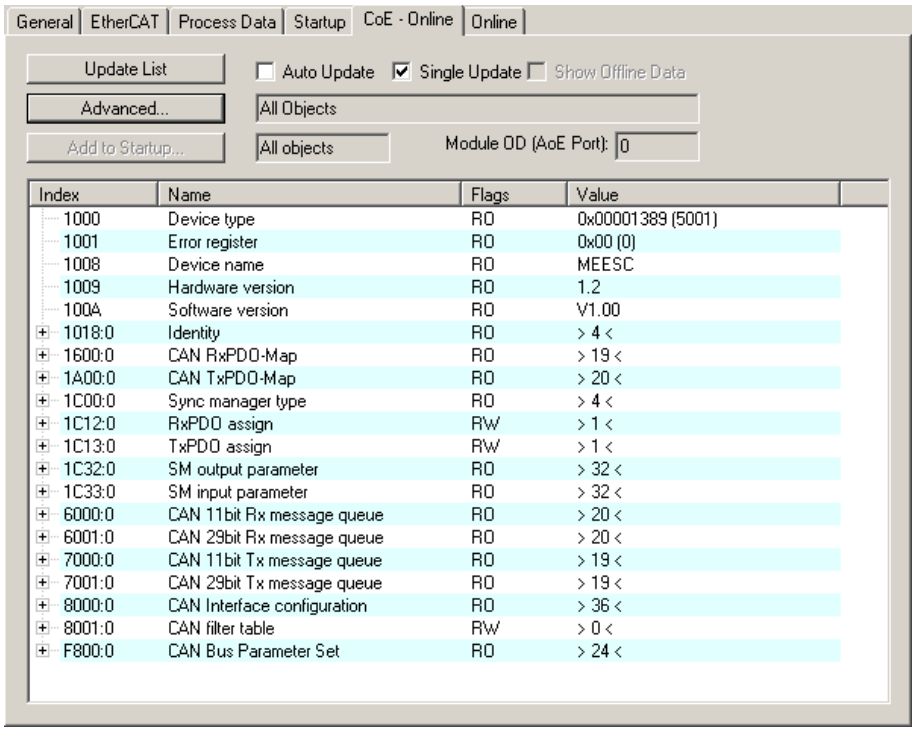


Figure 13: Object dictionary

The *Process Data* section will be initially empty. Click the Load PDO info from device button to read the data:

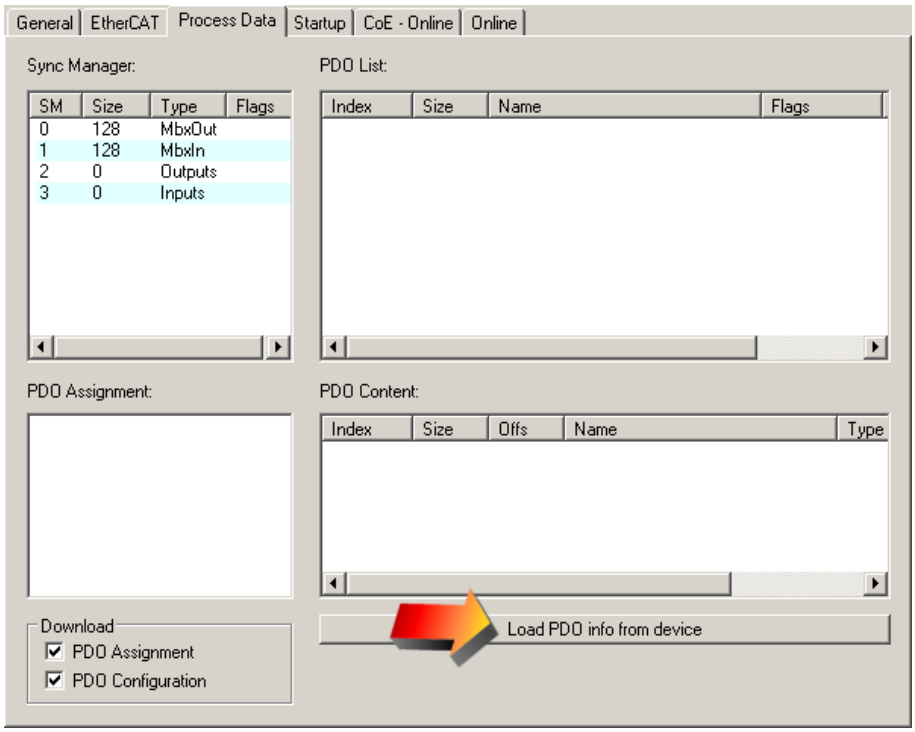


Figure 14: Loading process data

The configurator will now display the process data:

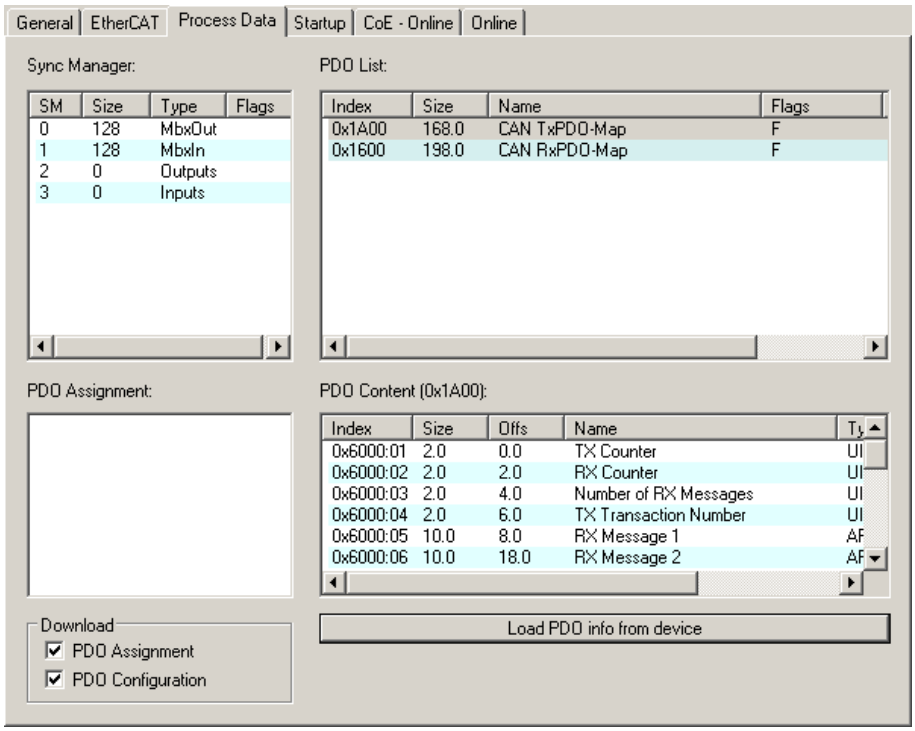


Figure 15: Process data display

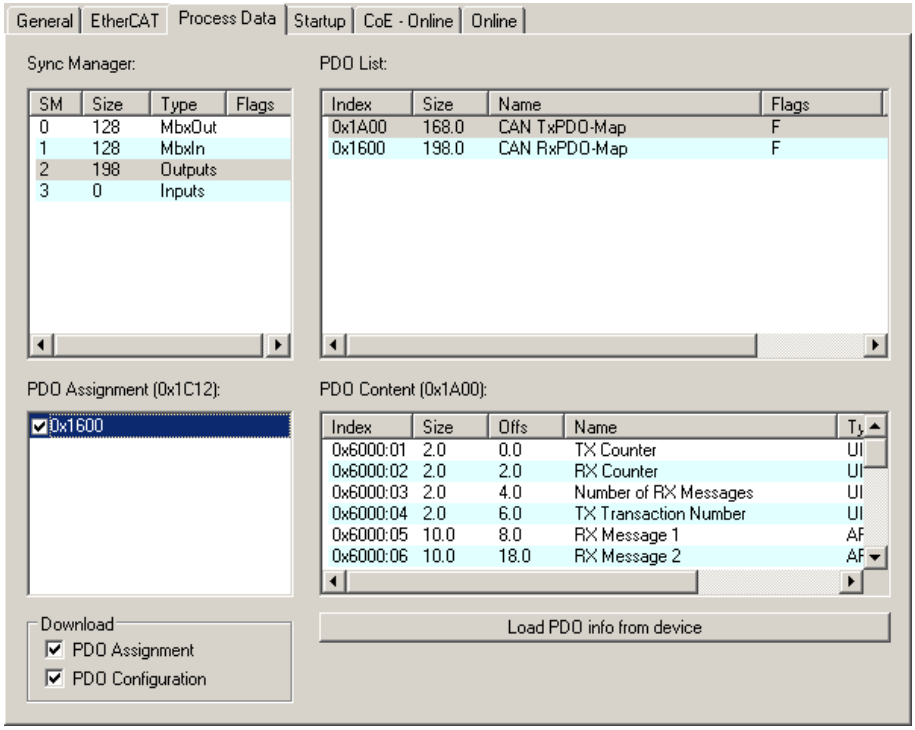


Figure 16: Process data (output PDOs chosen)

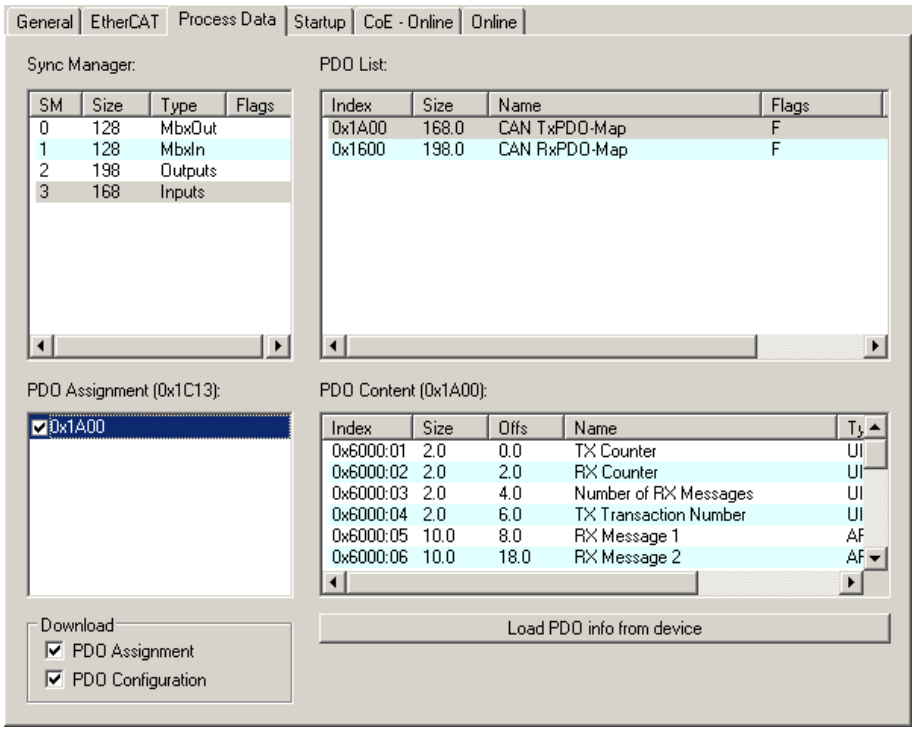


Figure 17: Process data (input PDOs chosen)

The CAN-EtherCAT gateway will only go active on the CAN bus after the baud rate has been set (see chapter “Object F800h CAN Bus Parameter” from page 49). Consequently, it makes sense to set the baud rate right now.

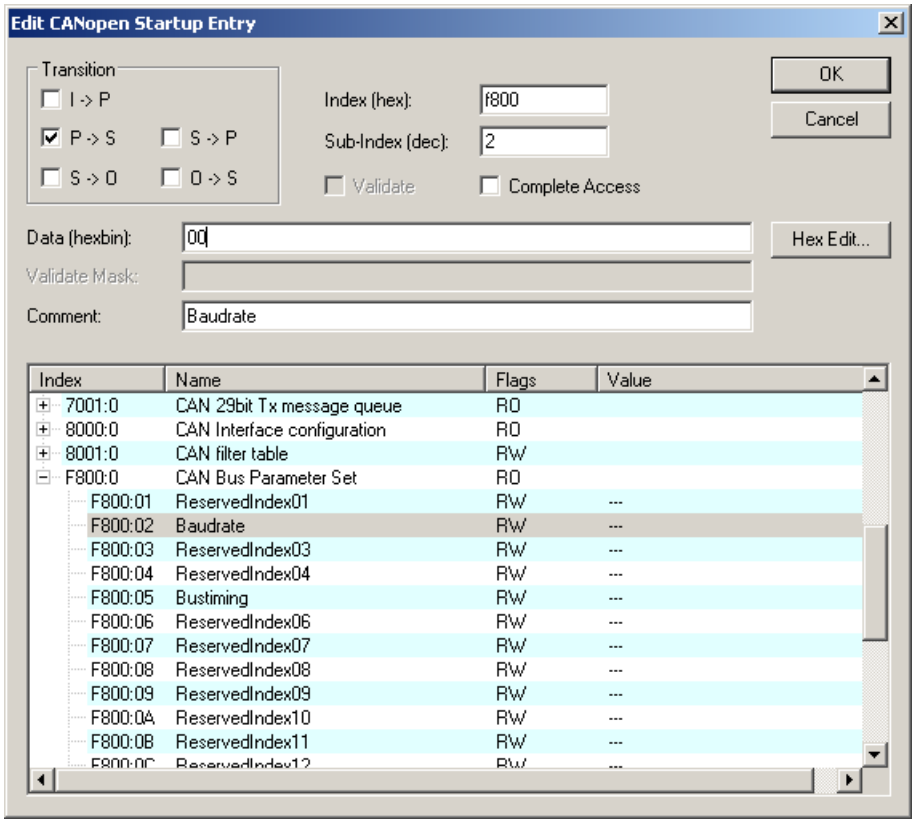


Figure 18: Baud rate setting during startup sequence

The final startup sequence may look like shown in the following example:

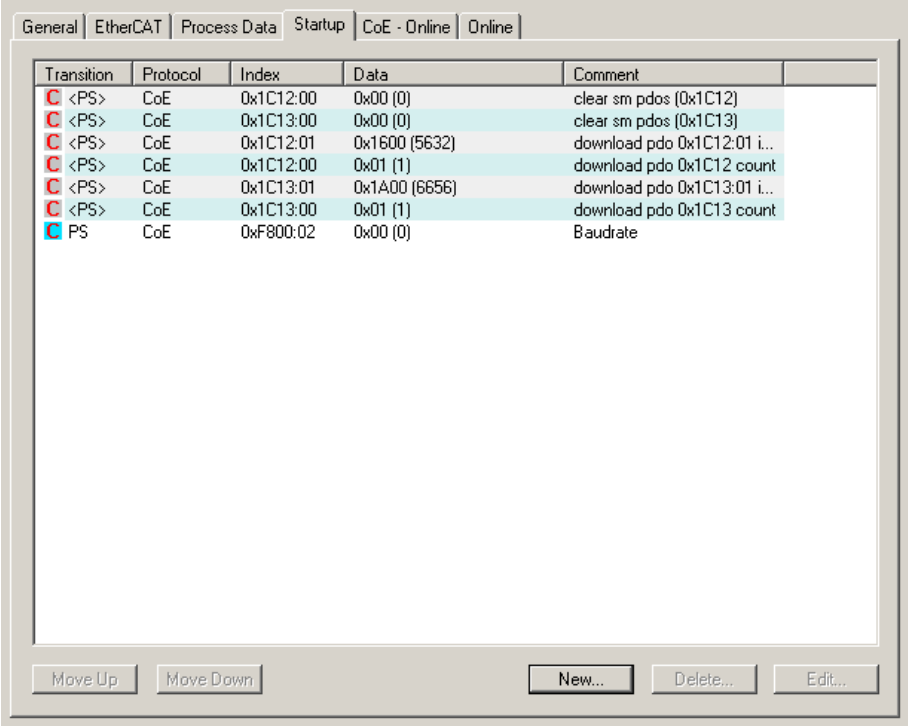


Figure 19: Startup - sequence example

Following the startup sequence, the EtherCAT network can be set active by calling the Reload Devices function by clicking *Main Menu/Actions*:

Figure 20: Reload Devices

The CAN interface's process image will look as follows:

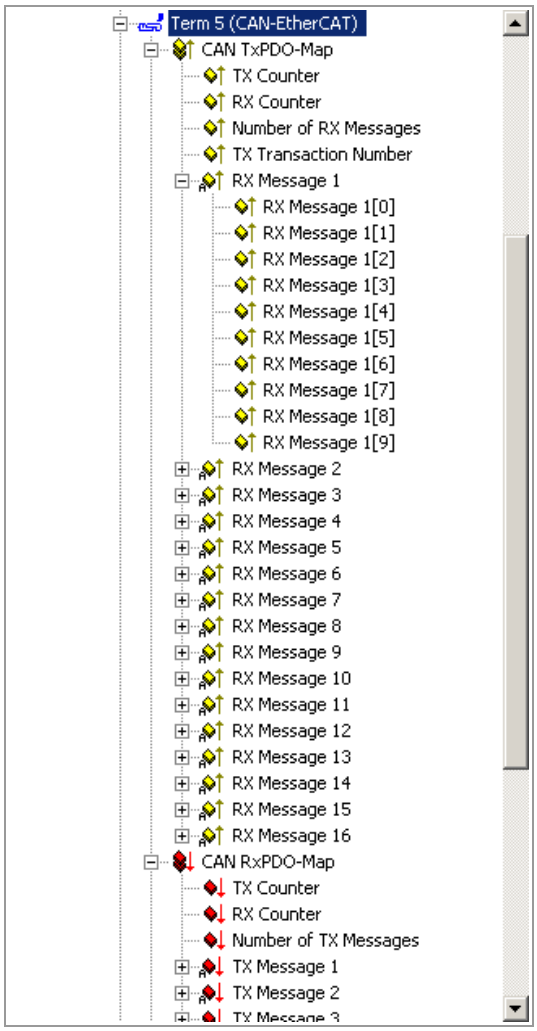


Figure 21: Can interface process image

4.3.1 Exporting EtherCAT Network Information (ENI)

To export the configuration file for an EtherCAT Master choose “Export Configuration File…”:

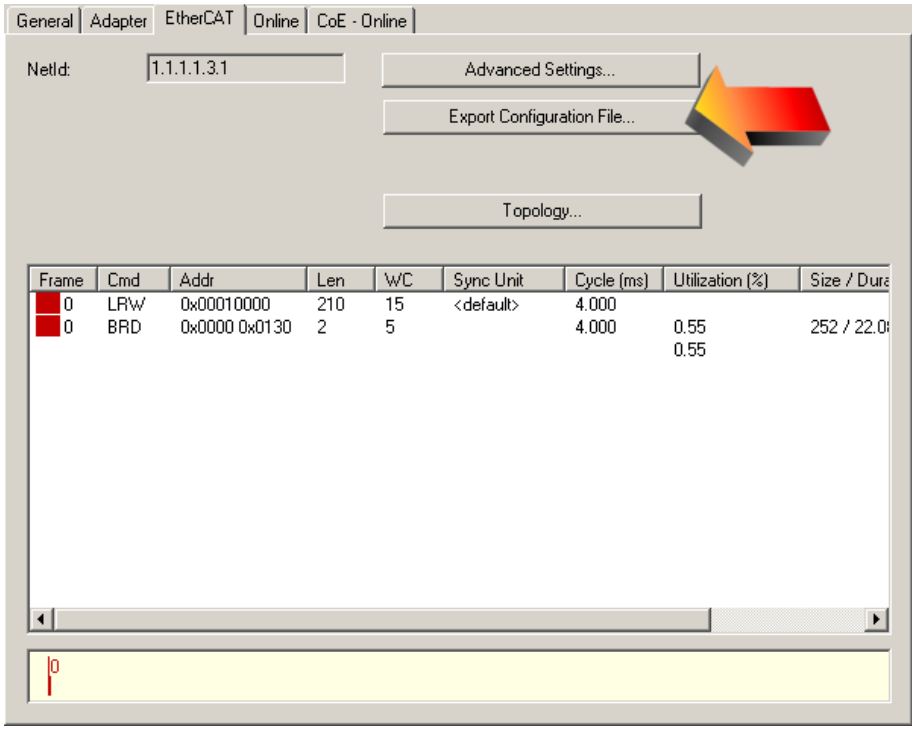


Figure 22: Exporting configuration file

5. EtherCAT Communication

5.1 CAN Interface

The CAN Interface is based on a modular device profile (Fieldbus Gateway, Profile No. 5000), and it supports one CAN module. This module includes one CAN Tx message queue in the output area and one CAN Rx message queue in the input area.

5.1.1 Object Dictionary Structure

The object dictionary is composed of the following areas:

Index	Object Dictionary Areas
0000_h...$0FFF_h$	Data Type Area
1000_h...$1FFF_h$	Communication Area
2000_h..$5FFF_h$	Vendor Specific Area
6000_h...$6FFF_h$	Input Area (CAN Rx message queue)
7000_h...$7FFF_h$	Output Area (CAN Tx message queue)
8000_h...$8FFF_h$	Configuration Area (CAN interface configuration)
$F000_h$...$FFFF_h$	Device Area

Table 5: Object dictionary structure

The following explains the definition of a standard and an extended CAN message queue. For proper operation one of both CAN message queues must be chosen. This can be accomplished by writing the CAN interface settings object (8000_h). The RPDO and TPDO mapping objects (1600_h and $1A00_h$) will change accordingly.

5.1.1.1 Output Data

The CAN interface output data include the Tx message queue plus the control data for the Rx and TX message queues. The CAN interface output data is always required.

5.1.1.2 Input Data

The CAN interface input data include the Rx message queue plus the status information for the Rx and Tx message queues. The CAN interface input data is always required.

5.1.2 Object Dictionary

The CAN-EtherCAT gateway layer 2 implementation supports the following objects:

Index	Name
1000_h	Device type
1008_h	Device name
1009_h	Hardware version
$100A_h$	Software version
1018_h	Identity
1600_h	**RPDO-Map CAN interface**
$1A00_h$	**TPDO-Map CAN interface**
$1A85_h$	**CAN Status PDO**
$1C00_h$	Sync manager type
$1C12_h$	**RPDO assign**
$1C13_h$	**TPDO assign**
2000_h	Other Settings
2010_h	Statistics
6000_h	**CAN interface input (11-bit identifier)**
6001_h	**CAN interface input (29-bit identifier)**
7000_h	**CAN interface output (11-bit identifier)**
7001_h	**CAN interface output (29-bit identifier)**
8000_h	**CAN interface configuration**
8001_h	**CAN filter table**
$F000_h$	Modular Device Profile
$F108_h$	CAN Status
$F800_h$	CAN bus parameter

5.1.3 Standard Objects (1000_h...$1FFF_h$)

5.1.3.1 Object 1000_h Device Type

Index	Sub-Index	Description	Data Type	RW	Default
1000_h	0	*Device Type*	UINT32	RO	13881389_h

Variable Description

EtherCAT Slave device type:
The low word contains the used CoE profile (5001_d). The high word contains the module profile according to the modular device profile: 5000_d.

5.1.3.2 Object 1008_h Device Name

Index	Sub-Index	Description	Data Type	RW	Default
1008_h	0	*Device Name*	STRING	RO	“MEESC”

Variable Description

EtherCAT Slave device name.

5.1.3.3 Object 1009_h Hardware Version

Index	Sub-Index	Description	Data Type	RW	Default
1009_h	0	*Hardware Version*	STRING	RO	

Variable Description

CAN-EtherCAT gateway hardware version.

5.1.3.4 Object $100A_h$ Software Version

Index	Sub-Index	Description	Data Type	RW	Default
$100A_h$	0	*Software Version*	STRING	RO	

Variable Description

CAN-EtherCAT gateway software version.

5.1.3.5 Object 1018_h Identity

Index	Sub-Index	Description	Data Type	RW	Default
1018_h	0	*Number of sub-indexes*	UINT8	RO	4
	1	*Vendor ID*	UINT32	RO	17_h (23_d)
	2	*Product code*	UINT32	RO	2
	3	*Revision*	UINT32	RO	see below
	4	*Serial number*	UINT32	RO	see below

Variable Description

CAN-EtherCAT gateway identification characteristics.

Vendor ID esd vendor-ID = 23_d

Product code CAN-EtherCAT product code = 2

Revision CAN-EtherCAT ESI revision number
Corresponds to the slave revision number stored in its EEPROM ESI – used to determine which .xml ESI the configuration tool (e.g. the esd EtherCAT Workbench) shall use.

(Exception: With firmware version 1.0 this does not match the EEPROM revision number – it's 100_h / 256_d there)

Serial number Serial number. (Always 0 with Firmware Version 1.X)

5.1.3.6 Object 1C00$_h$ Sync Manager Type

Index	Sub-Index	Description	Data Type	RW	Default
1C00$_h$	0	*Number of sub-indexes*	UINT8	RO	4
	1	*Sync-Manager Type Channel 1: Mailbox Write*	UINT8	RO	1
	2	*Sync-Manager Type Channel 2: Mailbox Read*	UINT8	RO	2
	3	*Sync-Manager Type Channel 3: Process Data Write (Outputs)*	UINT8	RO	3
	4	*Sync-Manager Type Channel 4: Process Data Read (Inputs)*	UINT8	RO	4

Parameter Description

Sync-Manager Type:

Sync-Manager Type Channel 1: Mailbox Write

Sync-Manager Type Channel 2: Mailbox Read

Sync-Manager Type Channel 3: Process Data Write (Outputs)

Sync-Manager Type Channel 4: Process Data Read (Inputs)

5.1.3.7 Object 1600_h RPDO-Map CAN-Interface

This object defines the CAN interface mapping into the EtherCAT input data.

The first three sub-indexes contain the size of the Tx and Rx counters plus the number of Tx messages. The size of the CAN Rx message queue is configured through object 8000_h.

Object 8000_h is also used to define the CAN message ID mode, either 11-bit (Object 7000_h) or 29-bit (7001_h). Depending on the settings the contents of objects 7000_h and 7001_h are mapped in object 1600_h.

Object 1600_h is always required and must be defined in the PDO Assign Object $1C12_h$, sub-index 1.

Index	Sub-Index	Description	Data Type	RW	Default
1600_h	0	*Number of CAN-Messages+3*	UINT8	RO	
	1	*1. PDO Mapping entry* (object $700z_h$ (CAN interface output), entry 01_h (Tx Counter))	UINT32	RO	
	2	*2. PDO Mapping entry* (object $700z_h$ (CAN interface output), entry 02_h (Rx Counter))	UINT32	RO	
	3	*3. PDO Mapping entry* (object $700z_h$ (CAN interface output), entry 03_h (Number of Tx Messages))	UINT32	RO	
	4	*4. PDO Mapping entry* (object $700z_h$ (CAN interface output), entry 04_h (Tx Message 1))	UINT32	RO	
	...	..	...	...	
	m	*m. PDO Mapping entry* (object $700z_h$ (CAN interface output), entry m (Tx Message m-3))	UINT32	RO	

5.1.3.8 Object 1A00$_h$ TPDO-Map CAN-Interface

This object defines the CAN interface mapping into the EtherCAT output data.

The first three sub-indexes contain the size of the Tx and Rx counters plus the number of Tx messages. The size of the CAN Tx message queue is configured through object 8000$_h$.

Object 8000$_h$ is also used to define the CAN message ID mode, either 11-bit (object 7000$_h$) or 29 Bit (object 7001$_h$). Depending on the settings the contents of objects 6000$_h$ and 6001$_h$ are mapped in object 1A00$_h$.

Object 1A00$_h$ is always required and must be defined in the PDO Assign Object 1C13$_h$, sub-index 1.

Index	Sub-Index	Description	Data Type	RW	Default
1A00$_h$	0	Number of *CAN-Messages+4*	UINT8	RO	
	1	*1. PDO Mapping entry* (object 6000$_h$ (CAN interface input), entry 01$_h$ (Tx Counter))	UINT32	RO	
	2	*2. PDO Mapping entry* (object 6000$_h$ (CAN interface input), entry 02$_h$ (Rx Counter))	UINT32	RO	
	3	*3. PDO Mapping entry* (object 6000$_h$ (CAN interface input), entry 03$_h$ (Number of Rx Messages))	UINT32	RO	
	4	*4. PDO Mapping entry* (object 6000$_h$ (CAN interface input), entry 04$_h$ (Tx Transaction Number))	UINT32	RO	
	5	*5. PDO Mapping entry* (object 6000$_h$ (CAN interface input), entry 05$_h$ (Rx Message 1))	UINT32	RO	
	...	..		...	
	m	*m. PDO Mapping entry* (object 6000$_h$ (CAN interface input), entry m (Rx Message m-4))	UINT32	RO	

5.1.3.9 Object 1A85$_h$ CAN Status PDO

This object allows to map the CAN Status entries from object F108$_h$. See 5.1.5.8 for details about the mapped entries.

Index	Sub-Index	Description	Data Type	RW	Default
1A85$_h$	0	*Max Subitem*	UINT8	RO	14$_d$
	1	Object F108$_h$ sub index 01$_h$	UINT32	RO	
	2	Object F108$_h$ sub index 02$_h$	UINT32	RO	
	3	Object F108$_h$ sub index 03$_h$	UINT32	RO	
	4	*Padding (1 Bit)*	UINT32	RO	
	5	Object F108$_h$ sub index 05$_h$	UINT32	RO	
	6	Object F108$_h$ sub index 06$_h$	UINT32	RO	
	7	*Padding (10 Bit)*	UINT32	RO	
	8	Object F108$_h$ sub index 11$_h$	UINT32	RO	
	9	Object F108$_h$ sub index 12$_h$	UINT32	RO	
	10	Object F108$_h$ sub index 13$_h$	UINT32	RO	
	11	Object F108$_h$ sub index 14$_h$	UINT32	RO	
	12	*Padding (12 Bit)*	UINT32	RO	
	13	Object F108$_h$ sub index 21$_h$	UINT32	RO	
	14	Object F108$_h$ sub index 22$_h$	UINT32	RO	

5.1.3.10 Object 1C12$_h$ RPDO-Assign

Object 1C12$_h$ assigns the mapping of the CAN interface RPDOs.

Index	Sub-Index	Description	Data Type	RW	Default
1C12$_h$	0	*Number of sub-indexes*	UINT8	RW	1
	1	*CAN-Interface RPDO*	UINT16	RW	1600$_h$

5.1.3.11 Object 1C13$_h$ TPDO-Assign

Object 1C13$_h$ assigns the mapping of the CAN interface TPDOs.

Index	Sub-Index	Description	Data Type	RW	Default
1C13$_h$	0	*Number of sub-indexes*	UINT8	RW	1
	1	*CAN-Interface TPDO*	UINT16	RW	1A00$_h$

5.1.4 Manufacturer Specific Objects (2000_h-$5FFF_h$)

5.1.4.1 Object 2000_h Other Settings

NOTICE
The following CoE object (Index, Sub-Index) can only be written in *Pre-Operational* state:
Index: 2000_h, Sub-Index: 1

Index	Sub-Index	Name	Data Type	RW	Default
2000_h	0	*Number of sub-indexes*	UINT8	RO	2
	1	*EoE IP Port local mode*	BOOL	RW	0
	2	*Custom LED state*	UINT8	RW	0

Parameter Description

EoE IP Port local mode
Set to TRUE when Local IP Port Mode (section 5.2.3) shall be enabled

Custom LED state
Used to overwrite the state of the “Universal” LED (section 2.2.1)
Values / State:
0: Off
1..4: Flash x1..Flash x4
13: Blink
14: Flicker
15: On
Reading this value reflects only the value that was last written – not the actual LED state. The state set by writing this object is overwritten whenever the CAN-EtherCAT itself sets a state for the LED (i.e. the LED is usually turned off when the EtherCAT State changes)

5.1.4.2 Object 2010_h Statistics

Index	Sub-Index	Name	Data Type	RW	Default
2010_h	0	*Number of sub-indexes*	UINT8	RO	33
	1	*Reset*	UINT32	RW	
	2	*Cyclic handler time (min.)*	UINT32	RO	
	3	*Cyclic handler time (max.)*	UINT32	RO	
	4	*Cyclic handler time (avg.)*	UINT32	RO	
	5	*CAN handler time (min.)*	UINT32	RO	
	6	*CAN handler time (max.)*	UINT32	RO	
	7	*CAN handler time (avg.)*	UINT32	RO	
	8	*Watchdog triggered*	UINT32	RO	
	10_h	*EoE Frames EtherCAT Rx*	UINT32	RO	
	11_h	*EoE Frames EtherCAT Tx*	UINT32	RO	
	12_h	*EoE Frames EtherCAT Tx Error*	UINT32	RO	
	13_h	*EoE Frames EtherCAT Tx Overrun*	UINT32	RO	
	14_h	*EoE Frames Local Rx*	UINT32	RO	
	15_h	*EoE Frames Local Tx*	UINT32	RO	
	16_h	*EoE Frames Local Tx Error*	UINT32	RO	
	20_h	*App. CPU Usage (User)*	UINT8	RO	
	21_h	*App. CPU Usage (System)*	UINT8	RO	
	30_h	*CAN Frames TX Requested*	UINT32	RO	
	31_h	*CAN Frames TX*	UINT32	RO	
	32_h	*CAN Frames RX*	UINT32	RO	
	34_h	*CANDriver Controller overrun*	UINT32	RO	
	35_h	*CANDriver FIFO overrun*	UINT32	RO	
	36_h	*CANDriver Error Frames*	UINT32	RO	
	37_h	*CANDriver Aborted Frames*	UINT32	RO	
	38_h	*CANDriver RX Frames*	UINT32	RO	
	39_h	*CANDriver RX RTR Frames*	UINT32	RO	
	$3A_h$	*CANDriver RX Frames Ext.*	UINT32	RO	
	$3B_h$	*CANDriver RX RTR Frames Ext.*	UINT32	RO	
	$3C_h$	*CANDriver TX Frames*	UINT32	RO	
	$3D_h$	*CANDriver TX RTR Frames*	UINT32	RO	
	$3E_h$	*CANDriver TX Frames Ext.*	UINT32	RO	
	$3F_h$	*CANDriver TX RTR Frames Ext.*	UINT32	RO	

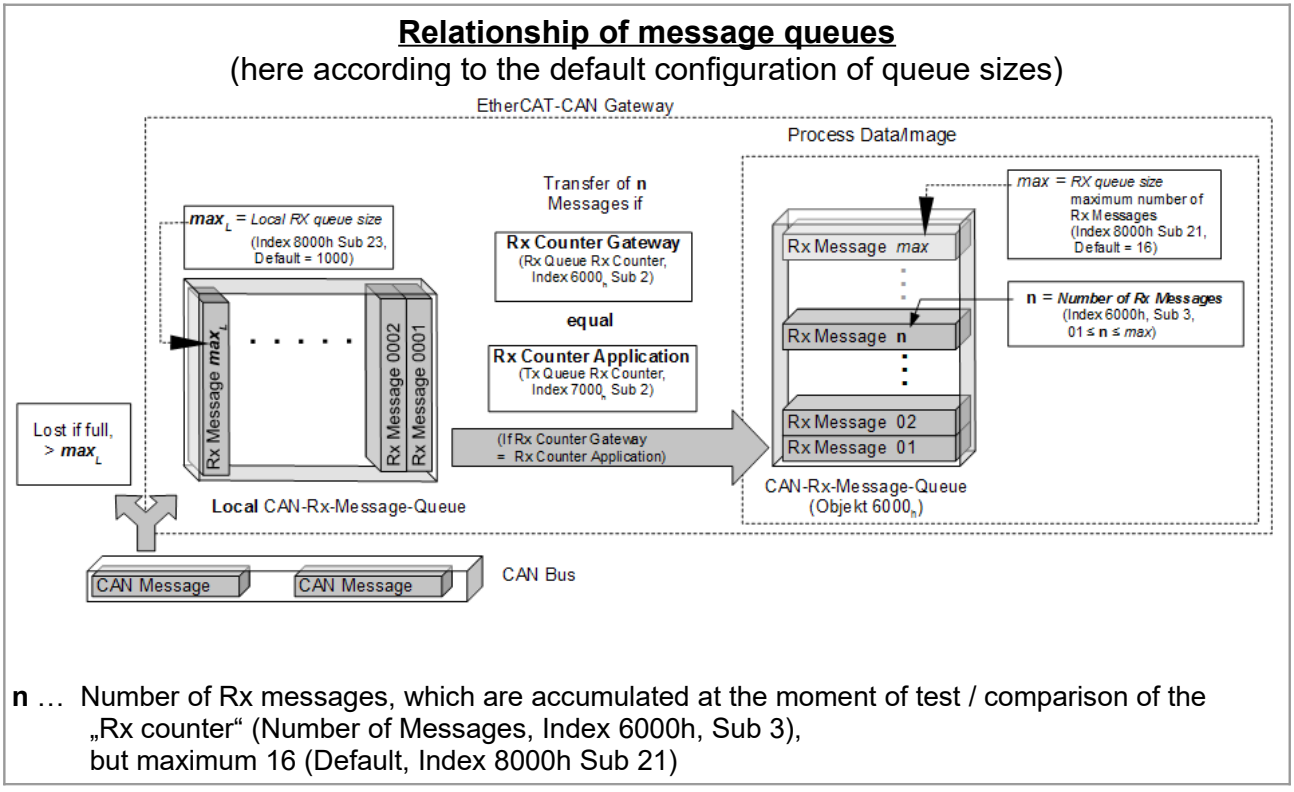
Parameter Description

Reset	When this object is written the statistics are reset. (Reading this value shows the time stamp of the last reset – milliseconds since device start up)
Cyclic handler time (min.)	For debugging purposes only. (Minimum time in application's cyclic handler within its last 10000 calls, in us)
Cyclic handler time (max.)	For debugging purposes only. (Maximum time in application's cyclic handler within its last 10000 calls, in us)
Cyclic handler time (avg.)	For debugging purposes only. (Average time in application's cyclic handler within its last 10000 calls, in us)
CAN handler time (min.)	For debugging purposes only. (Minimum time in application's CAN handler within its last 1000 calls, in us)
CAN handler time (max.)	For debugging purposes only. (Maximum time in application's CAN handler within its last 1000 calls, in us)
CAN handler time (avg.)	For debugging purposes only. (Average time in application's CAN handler within its last 1000 calls, in us)
Watchdog triggered	Times application watchdog was triggered due to missing process data (Outputs). (Watchdog value is calculated by ESC registers 0x0400 and 0x0420)
EoE Frames EtherCAT Rx	No. of Ethernet frames received from EtherCAT
EoE Frames EtherCAT Tx	No. of Ethernet frames sent to EtherCAT
EoE Frames EtherCAT Tx Error	No. of Ethernet frames that could not be sent to EtherCAT due to an error
EoE Frames EtherCAT Tx Overrun	No. of Ethernet frames that could not be sent to EtherCAT due to Tx buffer overrun (This kind of frame loss is likely to happen – higher level protocols on Ethernet side, such as TCP/IP, will handle this)
EoE Frames Local Rx	No. of Ethernet frames received on local Ethernet interface
EoE Frames Local Tx	No. of Ethernet frames sent to local Ethernet interface
EoE Frames Local Tx Error	No. of Ethernet frames that could not be sent to local Ethernet interface
App. CPU Usage (User/System)	For debugging purposes only. (Application's average CPU usage since last reading one of these two items, in percent)
CAN Frames TX Requested	Total No. of CAN frames that should have been sent – according to write accesses to “Tx Counter” objects. etc.
CAN Frames TX	Number of CAN frames that were successfully forwarded to the CAN driver
CAN Frames RX	Total No. of CAN frame successfully copied to the RX objects
CANDriver Controller overrun	CAN Driver statistics: No. of CAN Controller overruns
CANDriver FIFO overrun	CAN Driver statistics: No. of FIFO overruns
CANDriver Error Frames	CAN Driver statistics: No. of error frames
CANDriver Aborted Frames	CAN Driver statistics: No. of aborted frames
CANDriver RX Frames	CAN Driver statistics: No. of standard frames received
CANDriver RX RTR Frames	CAN Driver statistics: No. of standard RTR frames received
CANDriver RX Frames Ext.	CAN Driver statistics: No. of extended frames received
CANDriver RX RTR Frames Ext.	CAN Driver statistics: No. of extended RTR frames received
CANDriver TX Frames	CAN Driver statistics: No. of standard frames sent
CANDriver TX RTR Frames	CAN Driver statistics: No. of standard RTR frames sent
CANDriver TX Frames Ext.	CAN Driver statistics: No. of extended frames sent
CANDriver TX RTR Frames Ext.	CAN Driver statistics: No. of extended RTR frames sent

5.1.5 Profile Specific Objects (6000$_h$-FFFF$_h$)

These objects are identical for all EtherCAT Slave devices supporting the profile number 5000 (“CAN Interface”).

5.1.5.1 Object 6000$_h$ CAN Rx Message Queue



n … Number of Rx messages, which are accumulated at the moment of test / comparison of the „Rx counter“ (Number of Messages, Index 6000h, Sub 3), but maximum 16 (Default, Index 8000h Sub 21)

Figure 23: Relationship of the CAN Rx message queues

The number of transmitted Rx messages (**n**) is written in *Number of Rx Messages* (Sub-Index 3) and must not be changed until the "Rx Counters" are equal again.
For the chronological sequence see the example in Figure 25 on page 40.

Index	Sub-Index	Name	Data Type	RW	Default
6000$_h$	0	*Number of sub-indexes*	UINT8	RO	
	1	*Tx Counter Gateway*	UINT16	RO	
	2	*Rx Counter Gateway*	UINT16	RO	
	3	*Number of Rx Messages*	UINT16	RO	
	4	*Tx Transaction Number*	UINT16	RO	
	5	*Rx Message 1*	OCTET-STRING[10]	RO	
	...	...	...	...	
	m	*Rx Message m-4*	OCTET-STRING[10]	RO	

This object contains the CAN interface input messages with 11 Bit ID.

Parameter Description

Tx Counter Gateway

The Tx counter is increased by the Gateway to indicate that the CAN Tx messages were copied from the output data to the local CAN send queue (see Figure 26).

Rx Counter Gateway

The Rx counter is increased by the Gateway every time when new CAN Rx data arrived and the *Rx Counter Gateway* (6000_h, sub-index 02) is identical with *Rx Counter Application* (7000_h, sub-index02). This indicates that new Rx data has been written into the process input data (see Figure 23).

Number of Rx Messages

Contains the number of CAN Rx messages in the following input data when the RX Counter was increased (1…m-4).

Tx Transaction Number

Contains the transaction number of the last sent Tx (see Figure 24)

Rx Message 1...(m-4)

1. to (m-4). CAN Rx message
The message is composed of the following components:

Bit 0-3: CAN message length (0…8 bytes)

Bit 4: RTR Bit

Bit 5-15: CAN Identifier (11-bit CAN ID)

Bit 16-79: CAN-Rx data

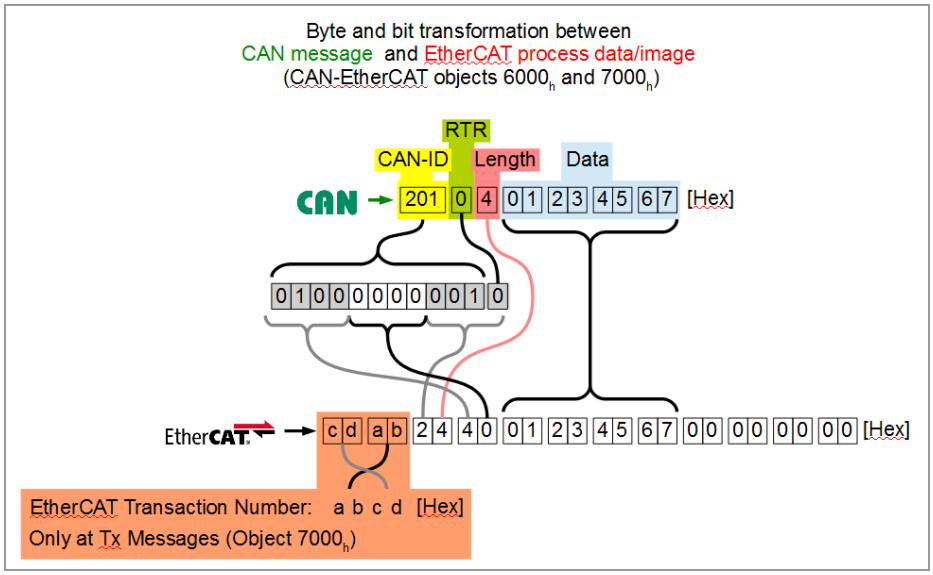


Figure 24: Formatting of the CAN data in the EtherCAT process image (11-bit Rx and Tx)

Example Sequence Rx-Counter

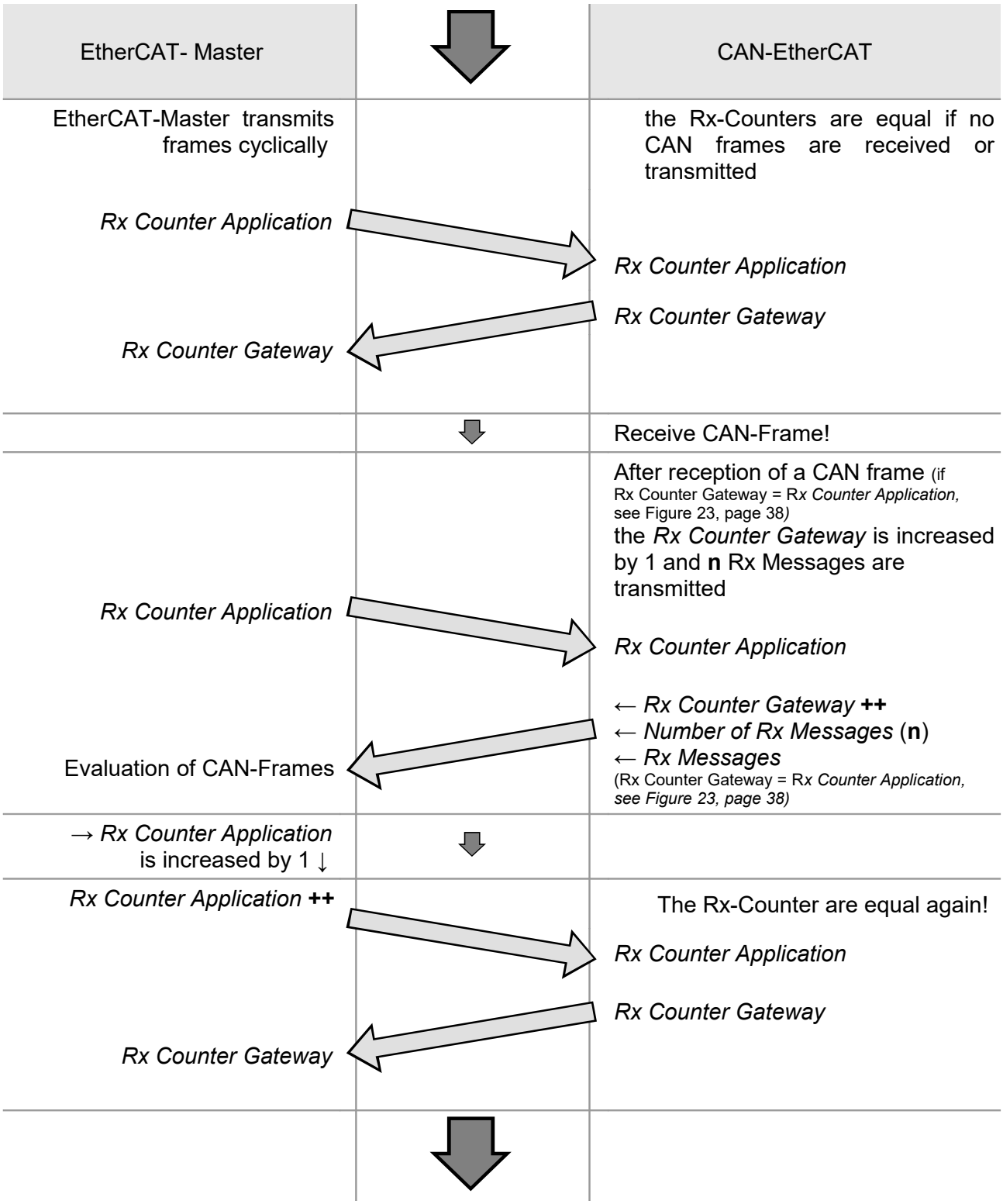


Figure 25: Chronological sequence of the Rx-Counters

5.1.5.2 Object 6001_h CAN Rx Extended Message Queue

Index	Sub-Index	Name	Data Type	RW	Default
6001_h	0	*Number of sub-indexes*	UINT8	RO	
	1	*Tx Counter Gateway*	UINT16	RO	
	2	*Rx Counter Gateway*	UINT16	RO	
	3	*Number of Rx Messages*	UINT16	RO	
	4	*Tx Transaction Number*	UINT16	RO	
	5	*Rx Message 1*	OCTET-STRING[14]	RO	
	...				
	m	*Rx Message m-4*	OCTET-STRING[14]	RO	

This object contains the CAN interface input messages with 29-bit ID.
See Figure 23 for the relationship of the CAN-Rx-Message-Queues. For 29-Bit-Identifiers objects 6001h and 7001h are used instead of objects 6000h and 7000h accordingly.
For the chronological sequence see the example in Figure 25 on page 40.

Parameter Description

Tx Counter Gateway — The Tx counter is increased by the Gateway to indicate that the CAN Tx messages were copied from the output data to the CAN send queue (see Figure 26).

Rx Counter Gateway — The Rx counter is increased by the Gateway every time when new CAN Rx data arrived and the *Rx Counter Gateway* (6001_h, sub-index 02) is identical with *Rx Counter Application* (7001_h, sub-index 02). This indicates that new Rx data has been written into the process input data (see Figure 23).

Number of Rx Messages — Contains the number of CAN Rx messages in the following input data when the Rx counter was increased (1…m-4).

Tx Transaction Number — Contains the transaction number of the last sent Tx message (see Figure 24).

Rx Message 1...(m-4) — 1. to (m-4). CAN Rx message
The message is composed of the following components:

Bit 0-3: CAN-Rx message length (0...8 byte)

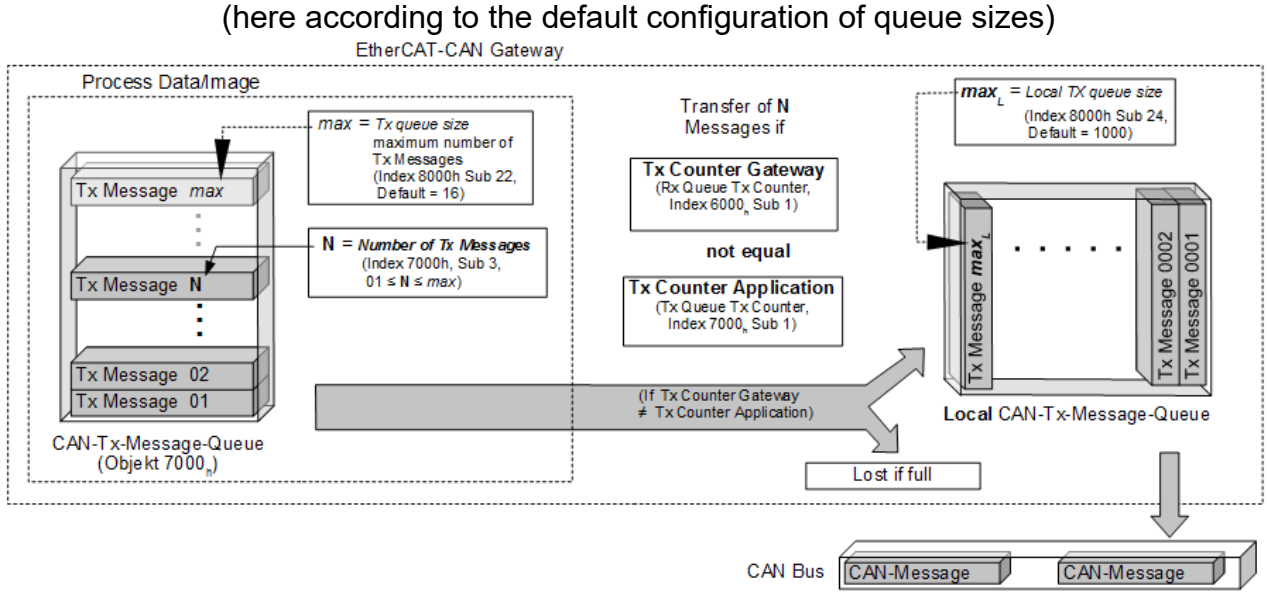
Bit 5-15: reserved
Bit 16-44: CAN Identifier (11- or 29-bit CAN identifier)

Bit 46: RTR bit

Bit 47: 0 = 11-bit CAN identifier
1 = 29-bit CAN identifier

Bit 48-111: CAN Rx data

5.1.5.3 Object 7000$_h$ CAN Tx Message Queue



N … Number of Tx messages, which are accumulated at the moment of test / comparison of the „Tx counter“ (*Number of Tx Messages*, Index 7000h, Sub 3), but maximum 16 (Default, *Tx queue size* Index 8000h Sub 22).

Figure 26: Relationship of the CAN Tx message queues

The number of effectively transmitted Tx messages (**N**) is written in *Number of Tx Messages* (Sub-Index 3) and will not be changed until the "Tx Counters" are equal again.
For the chronological sequence see the example in Figure 27 on page 44.

NOTICE
The following CoE objects (Index, Sub-Index) can only be written in *Pre-Operational* state:
Index: 7000$_h$, Sub-Index: 4...m$_h$

Index	Sub-Index	Name	Data Type	RW	Default
7000$_h$	0	*Number of sub-indexes*	UINT8	RO	
	1	*Tx Counter Application*	UINT16	RW	
	2	*Rx Counter Application*	UINT16	RW	
	3	*Number of Tx Messages*	UINT16	RW	
	4	*Tx Message 1*	OCTET-STRING[12]	RW	
	...	...	...	...	
	m	*Tx Message m-3*	OCTET-STRING[12]	RW	

This object contains the CAN interface output messages with 11-bit ID.

The maximum value of the sub-index, and thus the number of Tx messages, is defined in the RxPDO-Mapping-Object (object 1600$_h$).

Parameter Description

Tx Counter Application

This counter must be increased when or after writing the CAN Tx message to the output data (see Figure 26).

Rx Counter Application

This counter must be increased by the EtherCAT Master application for each CAN Rx message list it has received and read. This indicates that the received Rx messages have been read (see Figure 23).

Number of Tx Messages

Contains the number of CAN Tx messages, which are transmitted with every increase of the Tx counter (N = 1…m-3).

Tx Message 1...(m-3)

CAN Tx messages which are transmitted with every increase of the Tx counter.
The message is composed of the following components:

Bit 0-15: Transaction Number
The transaction number of the last transmitted CAN Tx message; readable in the input data.

Bit 16-19: CAN message length (0…8 bytes)

Bit 20: RTR bit

Bit 21-31: CAN identifier (11-bit CAN ID)

Bit 32-95: CAN Tx data

See Figure 24 on page 39.

Example Chronological Sequence Tx-Counter

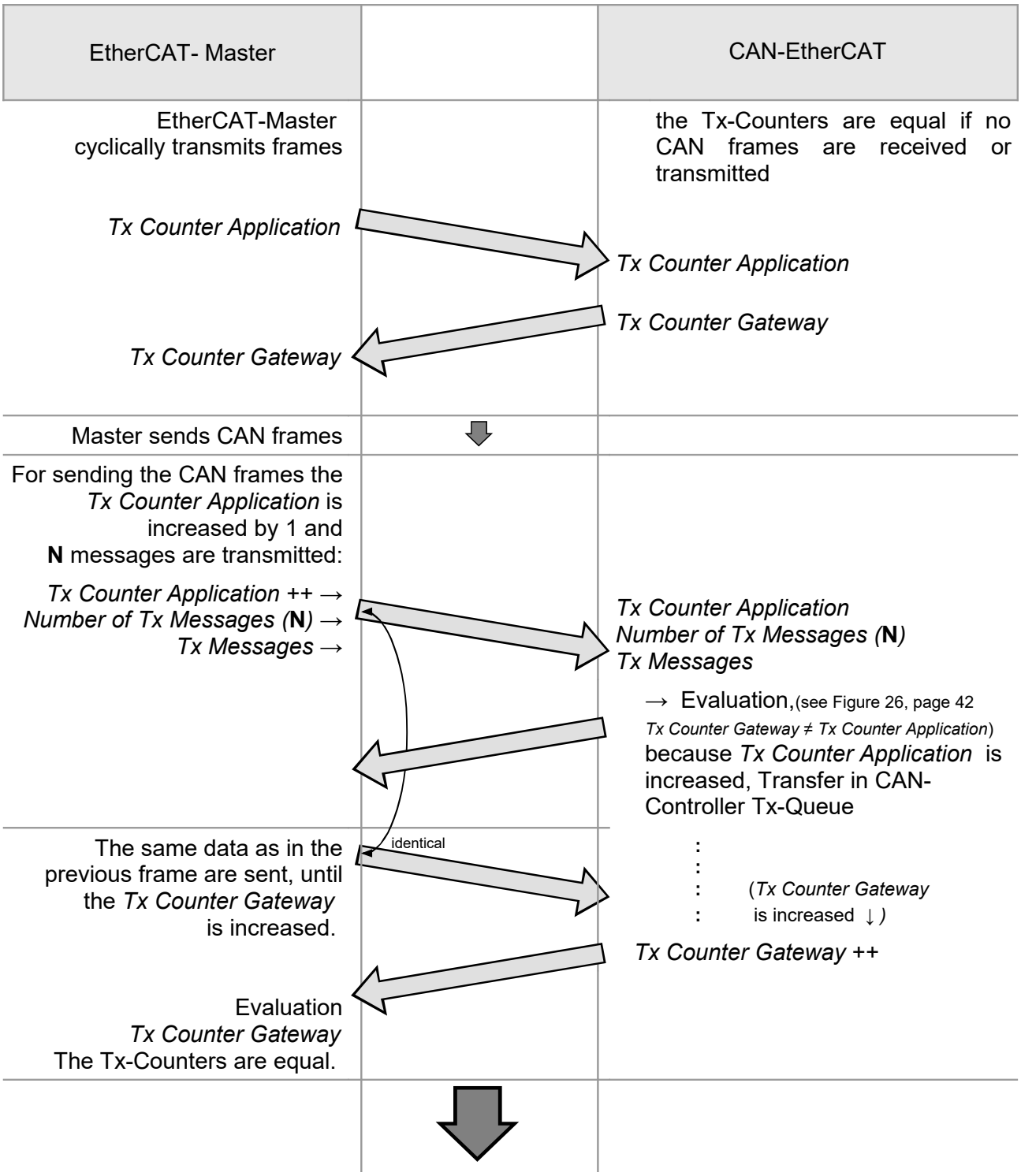


Figure 27: Chronological Sequence Tx-Counter

5.1.5.4 Object 7001_h CAN Tx Extended Message Queue

NOTICE
The following CoE objects (Index, Sub-Index) can only be written in *Pre-Operational* state:
Index: 7001_h, Sub-Index $4...m_h$

Index	Sub-Index	Name	Data Type	RW	Default
7001_h	0	*Number of sub-indexes*	UINT8	RO	
	1	*Tx Counter Application*	UINT16	RW	
	2	*Rx Counter Application*	UINT16	RW	
	3	*Number of Tx Messages*	UINT16	RW	
	4	*Tx Message 1*	OCTET-STRING[16]	RW	
	...	...	...	...	
	m	*Tx Message m-3*	OCTET-STRING[16]	RW	

This object contains the CAN interface input messages with 29-Bit ID.
See Figure 26 for the relationship of the CAN-Tx-Message-Queues. For 29-Bit-Identifiers objects 6001h and 7001h are used instead of objects 6000h and 7000h accordingly.
For the chronological sequence see the example in Figure 27 on page 44.

Parameter Description

Tx Counter Application
This counter must be increased when or after writing the CAN Tx message to the output data (see Figure 26).

Rx Counter Application
This counter must be increased by the EtherCAT Master application for each CAN Rx message list it has received and read. This indicates that the received Rx messages have been read (see Figure 23).

Number of Tx Messages
Contains the number of CAN Tx messages which are transmitted with every increase of the Tx counter (1…m-3).

Tx Message 1...(m-3)
CAN Tx messages which are transmitted with every increase of the Tx counter.
The message is composed of the following components:

Bit 0-15: Transaction Number
The transaction number of the last transmitted CAN Tx message; readable in the input data.

Bit 16-31: CAN message length (0...8 byte)

Bit 32-60: CAN Identifier (11- or 29-bit CAN ID)

Bit 62: RTR bit

Bit 63: 0 = 11-bit CAN identifier
1 = 29-bit CAN identifier

Bit 64-127: CAN Tx data

5.1.5.5 Object 8000_h CAN-Interface-Configuration

NOTICE
The following CoE objects (Index, Sub-Index) can only be written in *Pre-Operational* state:
Index: 8000_h, Sub-Index: 20_h, 21_h, 22_h

Index	Sub-Index	Name	Data Type	RW	Default
8000_h	0	*Number of sub-indexes*	UINT8	RO	24_h (36_d)
	1	*Node Address*	UINT16	RW	0000_h
	$2...19_h$	Reserved for future extensions	-	-	-
	20_h	*Flags*	UINT16	RW	0000_h
	21_h	*Rx queue size*	UINT8	RW	10_h (16_d)
	22_h	*Tx queue size*	UINT8	RW	10_h (16_d)
	23_h	*Local Rx queue size*	UINT16	RW	$03E8_h$ (1000_d)
	24_h	*Local Tx queue size*	UINT16	RW	$03E8_h$ (1000_d)

The CAN interface can be configured with this object.

Parameter Description

Node Address Must be set to 0

Flags Bit 0-2: Reserved for future extensions; must be 0

Bit 3: 0 = Standard Queue (11-bit identifier),
1 = Extended Queue (29-bit identifier)

Bit 4-14: Reserved for future extensions; must be 0

Rx queue size Number of Rx messages; max. 250_d*

Tx queue size Number of Tx messages; max. 250_d*

Local Rx queue size Rx queue size of the internal CAN driver

Local Tx queue size Tx queue size of the internal CAN driver

** Also limited by SM size/configuration*

5.1.5.6 Object 8001_h CAN-Rx-Filter-Table

Index	Sub-Index	Name	Data Type	RW	Default
8001_h	0	*Number of sub-indexes*	UINT8	RW	
	1	*Identifier Area 1*	UINT64	RW	
	...	...	...	...	
	m	*Identifier Area m*	UINT64	RW	

This object assigns the CAN identifier areas, which are filled into the RX queue and are transmitted with the EtherCAT input data.

INFORMATION
For 29-Bit CAN identifiers bit 31 and 63 must be set!

In case this object is not configured, all received CAN messages will be assigned to the Rx queue and transmitted through the EtherCAT input data.

Parameter Description

Identifier Area 1 Byte 0-3: First identifier to be assigned to the Rx queue*

Byte 4-7: Last identifier to be assigned to the Rx queue*

Identifier Area m Byte 0-3: First identifier to be assigned to the Rx queue*

Byte 4-7: Last identifier to be assigned to the Rx queue*

m = max. FF_h (255_d)

*Firmware version 1.0: “First” and “Last” are exchanged, i.e. “First identifier” is in Byte 4-7 and “Last identifier” is in Byte 0-3.

5.1.5.7 Object $F000_h$ Modular Device Profile

Usually only needed by configuration tools, e.g. esd EtherCAT Workbench. See ETG.5100 documents for details.

Index	Sub-Index	Name	Data Type	RW	Default
$F000_h$	0	Max. sub-index	UINT8	RO	3
	1	Index distance	UINT	RO	16_d
	2	Maximum number of modules	UINT	RO	1
	3	General configuration	UDINT	RO	1

5.1.5.8 Object $F108_h$ CAN Status

Index	Sub-Index	Name	Data Type	RW	Default
$F108_h$	0	Max. sub-index	UINT8	RO	22_h
	1	Bus OFF (Read from CAN controller status byte)	BOOL	RO	false
	2	Warning Limit reached (Read from CAN controller status byte)	BOOL	RO	false
	3	Rx overflow (Read from CAN controller overrun counter)	BOOL	RO	false
	4	Reserved	BIT1	RO	0
	5	Tx overflow (Not served, always false)	BOOL	RO	false
	6	Ack error (Not served, always false)	BOOL	RO	false
	7	Reserved	BIT2	RO	0
	8	Reserved	BIT8	RO	0
	11_h	Reserved (by esd)	BIT1	RO	0
	12_h	Reserved (by esd)	BIT1	RO	0
	13_h	Reserved (by esd)	BIT1	RO	0
	14_h	Reserved (by esd)	BIT1	RO	0
	21_h	Rx error counter (Read from CAN controller Rx error counter byte)	USINT	RO	0
	22_h	Tx error counter (Read from CAN controller Tx error counter byte)	USINT	RO	0

5.1.5.9 Object F800$_h$ CAN Bus Parameter

Index	Sub-Index	Name	Data Type	RW	Default
F800$_h$	0	*Number of sub-indexes*	UINT8	RW	
	1	*Reserved for future extensions*			
	2	*Baud rate*	UINT8	RW	FF$_h$
	3, 4	*Reserved for future extensions*			
	5	*API-baud rate*	UINT32	RW	7FFFFFFF$_h$
	6...24	*Reserved for future extensions*			

This object contains the baud rate.

Parameter Description

Baud rate CAN bit rate according to table below:

Parameter *Baud rate* [decimal]	CAN Bit rate [kBaud]
0	1000
1	800
2	500
3	250
4	125
5	100
6	50
7	not allowed
8	not allowed
255	Baud rate as defined in parameter “API-baud rate” sub-index 5

Table 6: Parameters *Baud rate*

API-baud rate The structure of the 32-bit parameter “API-baud rate” depends on the UBR and UBRN values as shown in the following:

31 *UBR*	30 *LOM*	29 *UBRN*	28... ...24	23... ...16	15... ...8	7... ...0
0	*LOM*	0	Reserved		*Table index*	
0	*LOM*	1	Reserved	*Numerical Value*		
1	*LOM*	0	Reserved	*CAN_BR (of ARM9)*		

A combination of UBR = UBRN = 1 is not allowed!

Table 7: Parameter *API-baud rate*

Bit(s)	Value	Description
UBR	0	Use the pre-defined bit rate table (Table Index)(in combination with UBRN)
	1	Set the CAN controller bit rate register directly (BTR0/BTR1)
LOM	0	Configure the bit rate in 'active' mode (normal operation)
	1	Configure the bit rate in 'Listen-Only' mode
UBRN	0	Use the pre-defined bit rate table (in combination with UBR)
	1	Set bit rate to numerical value
Table index	x	Use the bit rate in pre-defined Table 9
CAN_BR	x	*CAN baud rate register of ARM9 AT91SAM9263*

Table 8: Bits of parameter API-baud rate

When 'User Bit Rate' (UBR) and 'User Bit Rate Numerical' (UBRN) are set to 0, bits 0…15 are interpreted as an index to a pre-defined bit rate table. This allows the setting of CAN bit rates without detailed knowledge of the CAN controller hardware.

Table index	Bit rate [kBit/s]	Constant *1)
0	1000	NTCAN_BAUD_1000
E_h	800	NTCAN_BAUD_800
1	666.6	-
2	500	NTCAN_BAUD_500
3	333.3	-
4	250	NTCAN_BAUD_250
5	166	-
6	125	NTCAN_BAUD_125
7	100	NTCAN_BAUD_100
10_h	83.3	-
8	66.6	-
9	50	NTCAN_BAUD_50
A_h	33.3	not allowed
B_h	20	not allowed
C_h	12.5	not allowed
D_h	10	not allowed

*1) The constants follow the CiA (CAN in Automation) recommendations.

Table 9: Pre-defined bit rate table

Constants and special features

Constant	Value	Function
NTCAN_BAUD_1000	0	Sets baud rate to 1000 kBit/s
NTCAN_BAUD_800	E_h	Sets baud rate to 800 kBit/s
NTCAN_BAUD_500	2	Sets baud rate to 500 kBit/s
NTCAN_BAUD_250	4	Sets baud rate to 250 kBit/s
NTCAN_BAUD_125	6	Sets baud rate to 125 kBit/s
NTCAN_BAUD_100	7	Sets baud rate to 100 kBit/s
NTCAN_BAUD_50	9	Sets baud rate to 50 kBit/s
NTCAN_NO_BAUDRATE	7FFF FFFF$_h$	Gateway cannot receive or transmit any message; stays passive on CAN bus
NTCAN_AUTOBAUD	00FF FFFE$_h$	Gateway checks baud rates until it detected the correct one
NTCAN_USER_BAUDRATE	8000 0000$_h$	sets the *UBR* bit
NTCAN_USER_BAUDRATE_NUM	2000 0000$_h$	*Sets the UBRN bit*
NTCAN_LISTEN_ONLY_MODE	4000 0000$_h$	*Sets the LOM bit*

Table 10: Constant

Leaving the CAN Bus
The special constant NTCAN_NO_BAUDRATE can be used as an argument for the Parameter *API-baud rate* to force the hardware to leave the CAN bus and return to the Boot-Up condition (or to start it).

Automatic Baud Rate Detection
The CAN-EtherCAT gateway is capable of detecting the CAN baud rate and initiating bus communication without effecting the CAN bus operation. This is only possible with the default bit rates from the esd bit rate table supporting the CiA bit timing requirements.

The automatic baud rate detection requires at least two other CAN nodes communicating with each other. The CAN-EtherCAT gateway will initially act as ‘Listen-Only’.

Use the special constant NTCAN_AUTOBAUD as an argument for the parameter API baud rate to initiate the automatic baud rate detection.

The driver will cease the automatic baud rate detection as soon as a valid baud rate is recognized, which is reported to the application through a so-called baud rate change event, or when a tangible baud rate was set through object F800$_h$.

With the UBR flag set to '1' and the URBN flag set to '0' the bits 0…24 are used to configure the CAN controller’s bit rate register directly using the predefined values.

In order to set the bit rate register directly the following information will be necessary:
CPU: ARM9 (see technical data from page 64)
CPU Master Clock: 120 MHz
CPU Manual: http://www.atmel.com -> CAN -> CAN Baud Rate Register

When the UBR flag is set to '0' and the UBRN flag is set to '1' the bits 0… 23 represent the baud rate as a numerical value in bits per second.

INFORMATION
When using the UBRN flag the BTR values are generated and may deviate from the values in the baud rate table.

UBR and UBRN cannot be set at the same time!

Listen Only Mode
This mode was developed for the purpose of CAN bus monitoring without effecting other CAN nodes. Combined with the baud rate setting it serves the implementation of 'hot plugging'.

With the Listen Only Mode (LOM) flag set to '0' the CAN controller works in regular active mode using the bit rate, which implies that messages can be received and transmitted.

Setting the LOM flag to '1' causes the CAN controller to operate in Listen Only Mode using the bit rate and can only receive messages.

5.2 EoE

The CAN-EtherCAT supports three different EoE modes, it is selected by the master sending the EoE configuration. (Fig. 36 and Fig. Fehler: Referenz nicht gefunden show configuration tool samples)

5.2.1 Switch Port Mode

This is the default mode, it is enabled when no EoE configuration is received. (i.e. usually configuration tools don't send a configuration when this mode is selected) This mode is also set when an invalid EoE configuration is received.

Every Ethernet frame received from EtherCAT will be sent to Ethernet interface and vice versa.

5.2.2 IP Port Mode

This mode is enabled when an EoE configuration is received. The EoE configuration must contain the IP address, subnet mask and default gateway. The MAC-Id value is ignored, the name server is optional.

In this mode the CAN-EtherCAT also acts as DHCP server for a single DHCP client on its Ethernet interface: The received IP configuration is offered to the DHCP client. (Therefore only one client must be connected)

5.2.3 Local IP Port Mode

This mode is enabled when an EoE configuration is received and object 0x2000.1 (see 5.1.4.1) was set to TRUE. As for the IP Port Mode the EoE configuration must contain the IP address, subnet mask and default gateway. The MAC-Id value is ignored, the name server is optional.

In this mode the CAN-EtherCAT uses the received IP configuration itself – allowing access to its

Webserver Interface, see chapter 6.

(As Ethernet–EtherCAT forwarding is still active this configuration should match the LAN settings the EtherCAT network is in, i.e. no duplicate IP addresses must exist, etc.)

5.2.4 Disabling EoE

The CAN-EtherCAT reads the “EoE enabled” bit from its EEPROM (cat. “General”, as defined in ETG.1000.6 documents). When this is set to 0 (checked at first change to PreOp after device start up), all EoE activity will be disabled and all EoE configuration options are ignored.

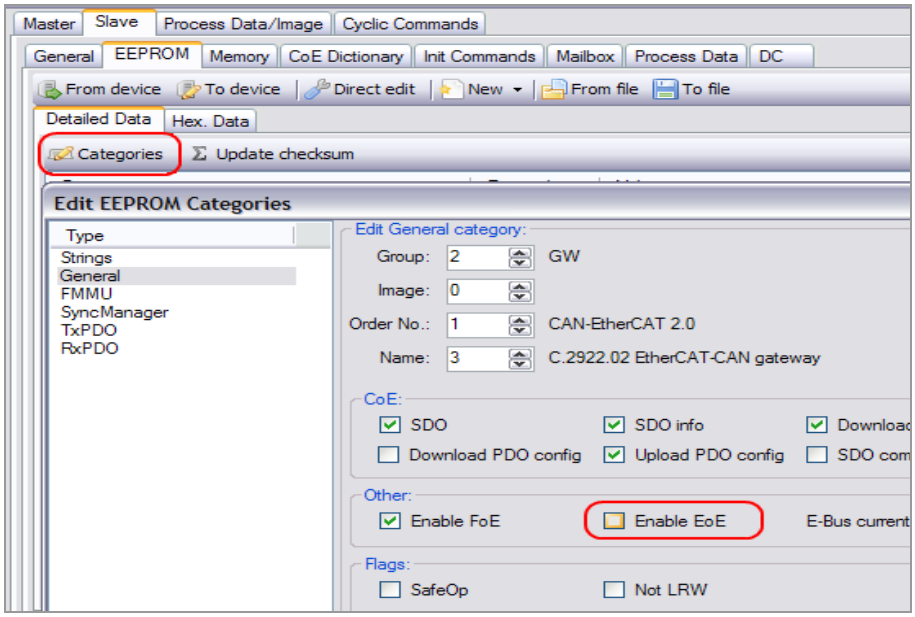


Figure 28: esd EtherCAT Workbench: Where to en/disable EoE

5.3 FoE

5.3.1 Firmware Update with the esd Workbench

NOTICE
Do not interrupt the CAN-EtherCAT gateway power supply during a firmware update as this might result in unforeseeable operating conditions.

1. Make sure slave is connected, etc.
2. Set the slave in the state *Bootstrap.* Therefore choose the tab *Slave* and than *General. Now* click on the button *Bootstr* as described in Figure 29. The *Current state:* of the CAN-EtherCAT gateway is switched to *Bootstrap.*

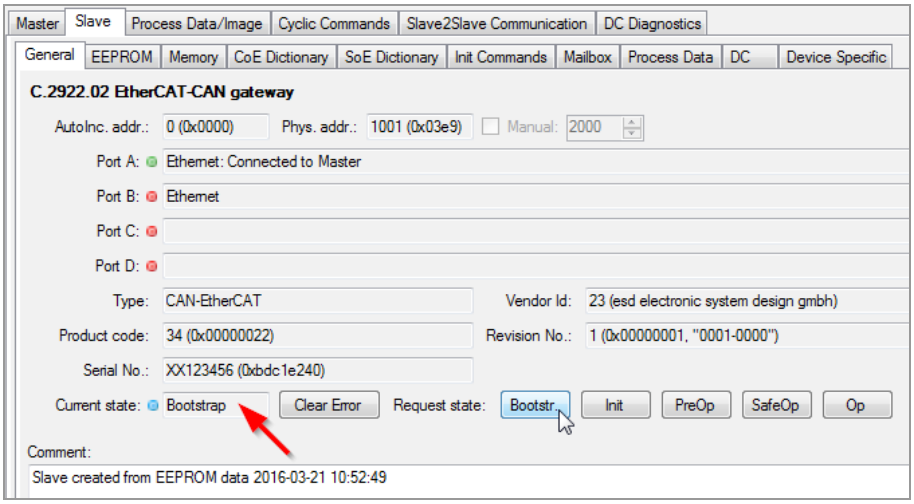


Figure 29: Firmware update via FoE

3. Select the tab *Slave*/*Mailbox* and choose the tab *FoE.*
   Enter „firmwareUpdate“ as name of the file in the input field *FoE filename:.*
   The *FoE password* has to be set to „0“.

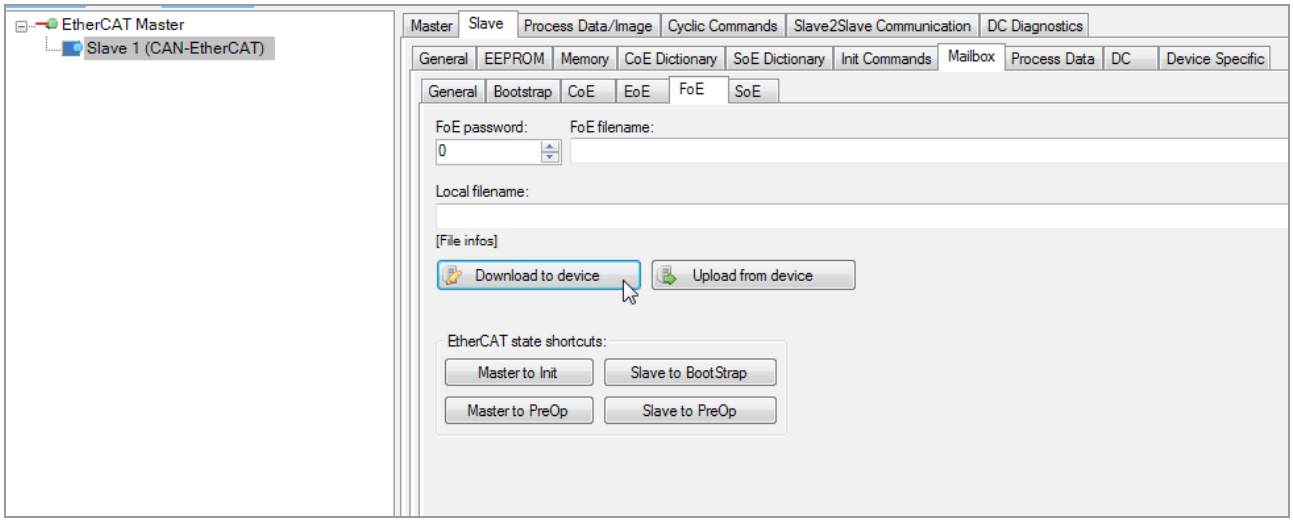


Figure 30: FoE file transfer dialog

4. Click the button *Download to device* (see Figure 30) and select the firmware file in the Windows selection dialog that appears.

Confirm your settings with the OK button and wait until the file is transferred (a progress bar will appear and the yellow LED is flickering while the transfer is in progress)

5. Wait until the update procedure is completed (approx. 3 minutes)

6. Change to the tab *CoE Dictionary* under *Slave*.

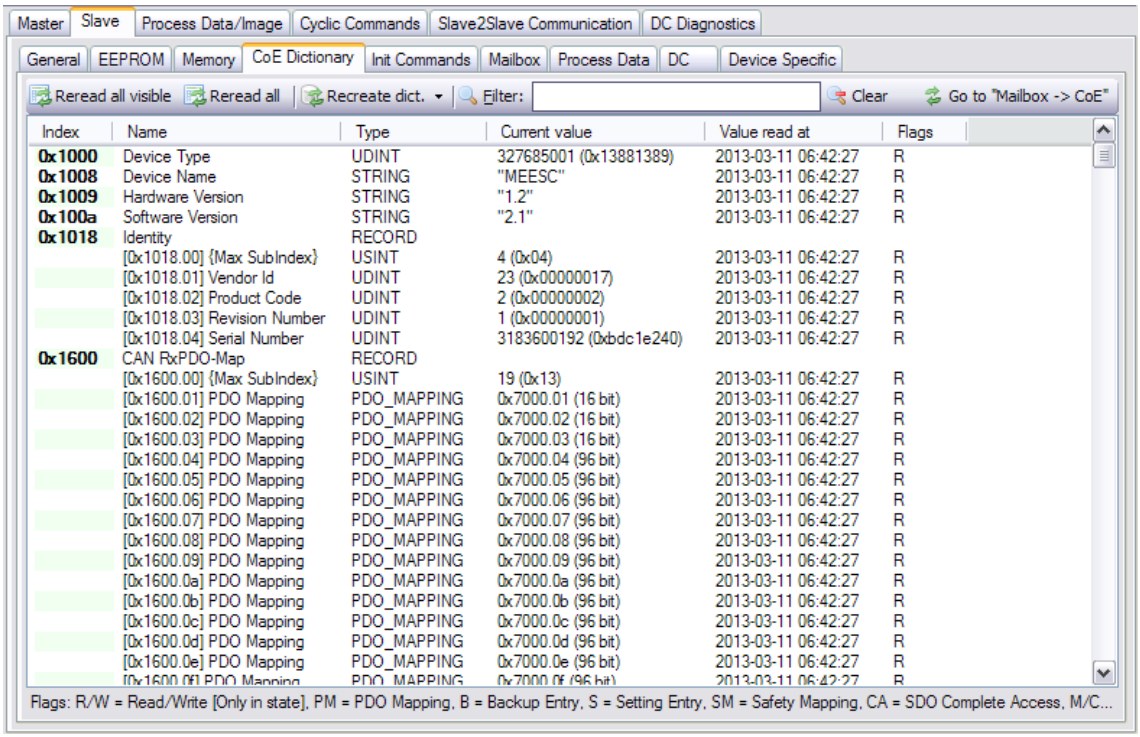


Figure 31: CAN-EtherCAT CoE Dictionary

7. Click on the button *Reread all* to ensure that the current objects are displayed.
8. Verify the current software version in object $100A_h$, see Figure 31.

5.3.2 Firmware update with Beckhoff EtherCAT Configurator

NOTICE
Do not interrupt the CAN-EtherCAT gateway power supply during a firmware update as this might result in unforeseeable operating conditions.

1. Make sure slave is connected, etc.
2. Set slave to “Bootstrap”, by button Bootstrap (Fig. 32)

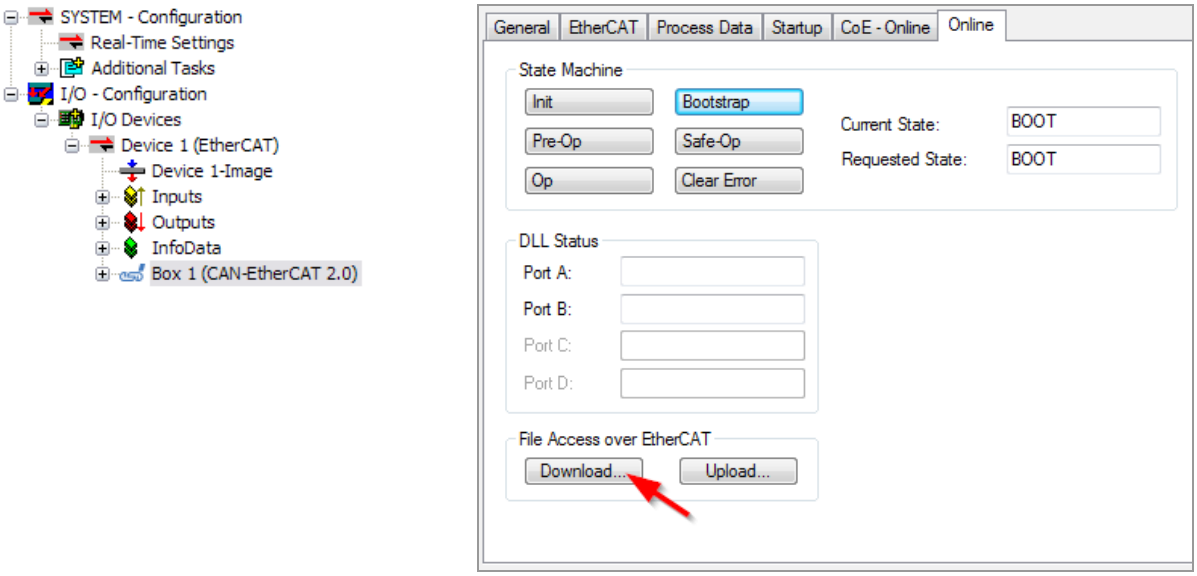


Figure 32: Firmware update by FoE

3. Click the Download... button and select the firmware file you received in the Windows file selection dialog that appears
4. Now a file transfer dialog (Fig. 33) will appear: Set the file name string to “firmwareUpdate” and leave the password at “00000000”

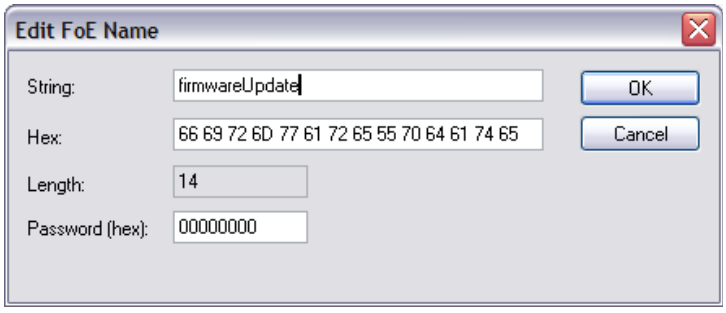


Figure 33: FoE file transfer dialog

5. Click OK and wait until the file is transferred (a progress bar will appear and the yellow LED is flickering while the transfer is in progress)
6. Wait until the update is applied, approx. 3 minutes
7. Now click “Reload I/O Devices” in the Configurator toolbar (activating “Free run” is not needed) and switch to the “CoE - Online” tab page.

8. Verify the current version in object $100A_h$, Fig. 34
   (Make sure you're actually seeing online data: uncheck “Show Offline Data”, and perhaps do “Reload I/O Devices” again, etc.)

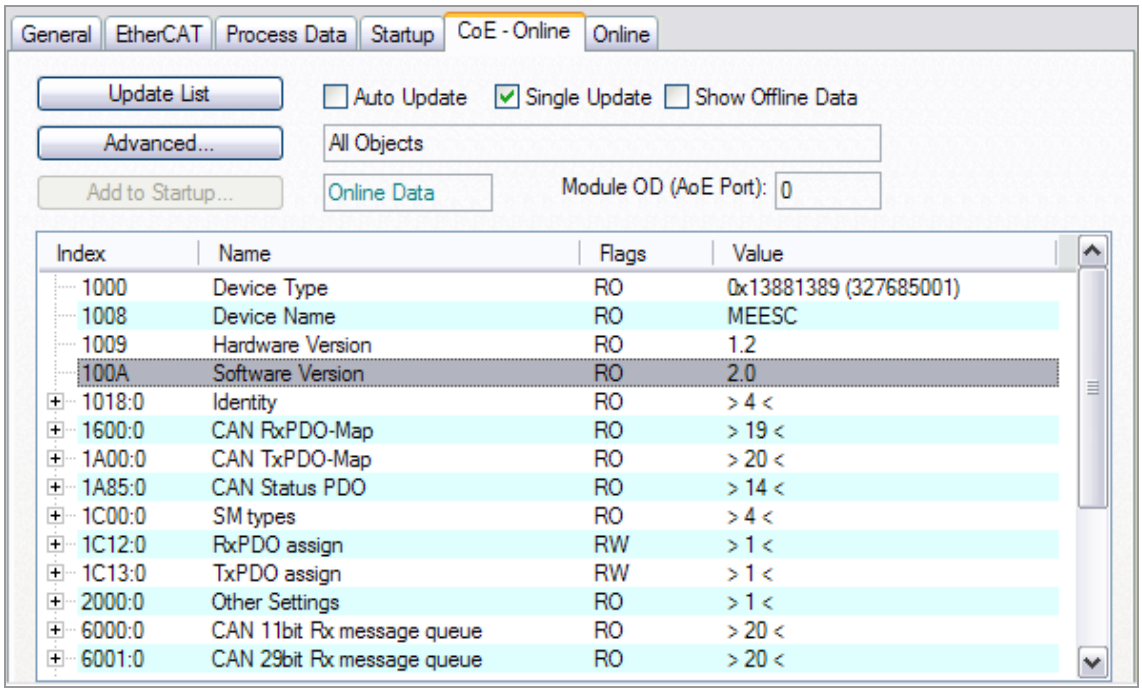


Figure 34: CAN-EtherCAT CoE dict., Software/Firmware version selected

6. Webserver Interface

To access the Webserver the CAN-EtherCAT has to be in “Local IP Port Mode“, see section 5.2.3. (EoE itself requires the EtherCAT device state “Operational”)

Make sure the IP settings assigned to the CAN-EtherCAT match the settings that are used by the system that shall access it, e.g. no IP address conflicts must occur etc.

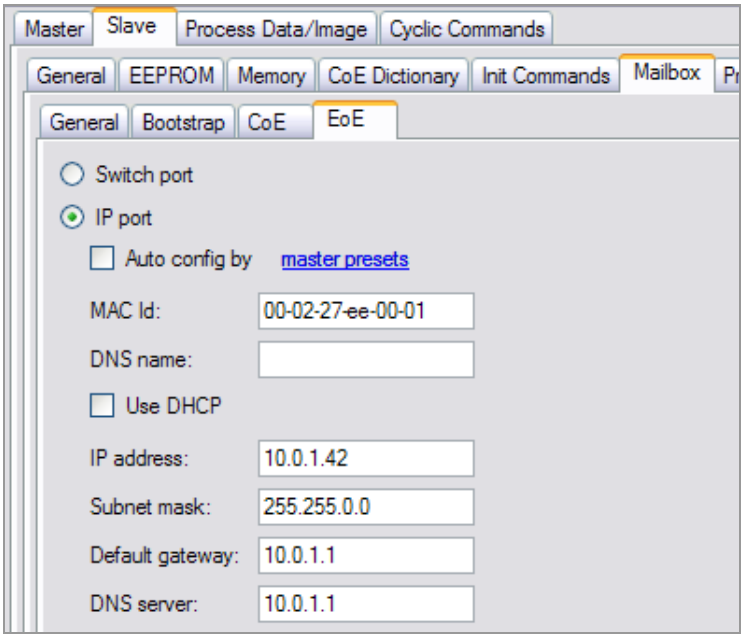


Figure 35: esd EtherCAT Workbench: Sample IP settings for the CAN-EtherCAT

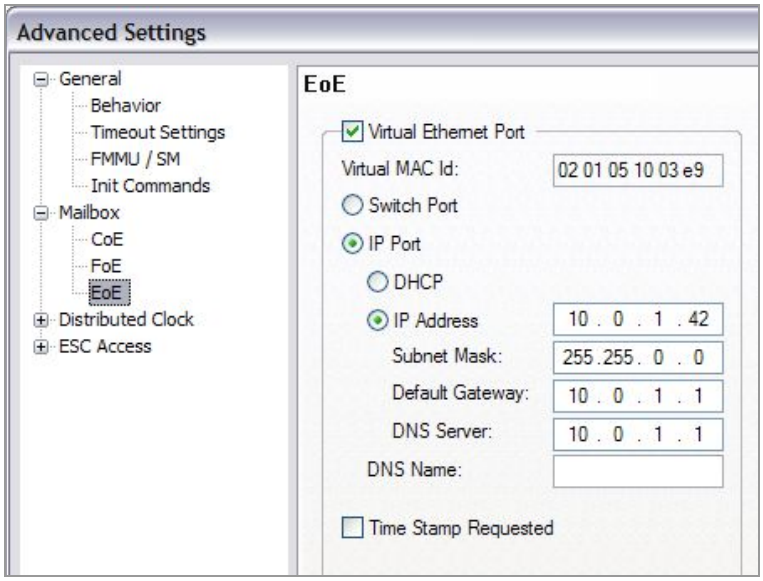


Figure 36: Beckhoff Configurator: Sample IP settings for the CAN-EtherCAT

6.1 Firmware Update

The CAN-EtherCAT gateway uses an internal HTTP server. Through means of a standard web browser it allows firmware updates and the display of CAN status information.

Just enter the IP address that was assigned to the CAN-EtherCAT in the web browser at the device that is connected to the CAN-EtherCAT. (e.g. `http://10.0.1.42` for the sample screen shots above)

6.1.1 Overview

The browser window provides a menu on the left hand side of the screen.

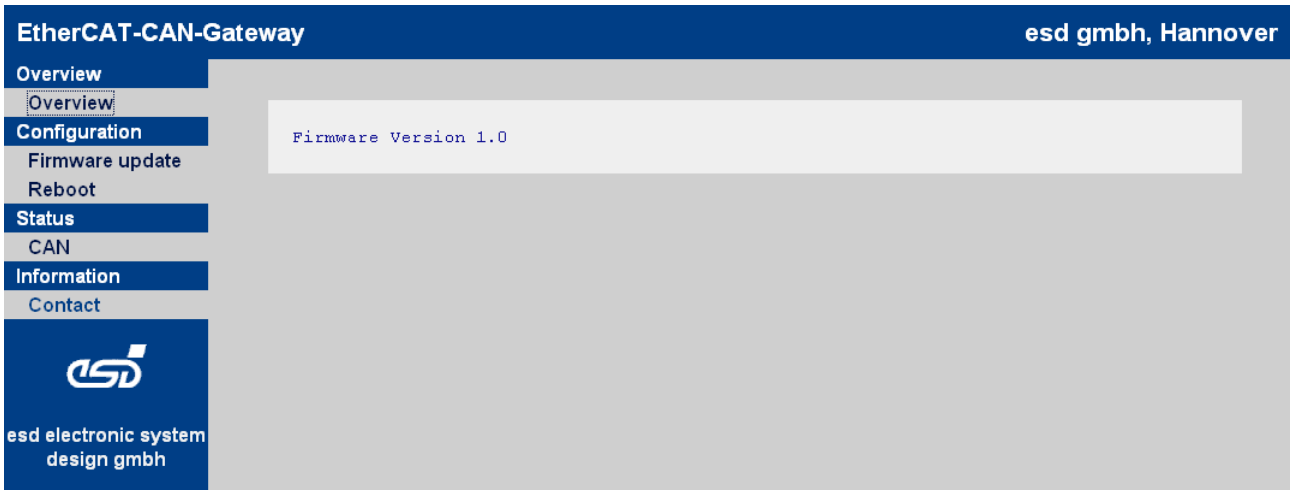


Figure 37: Overview

6.1.2 Firmware Update

In order to initiate a firmware update click the corresponding menu item *Firmware Update*.

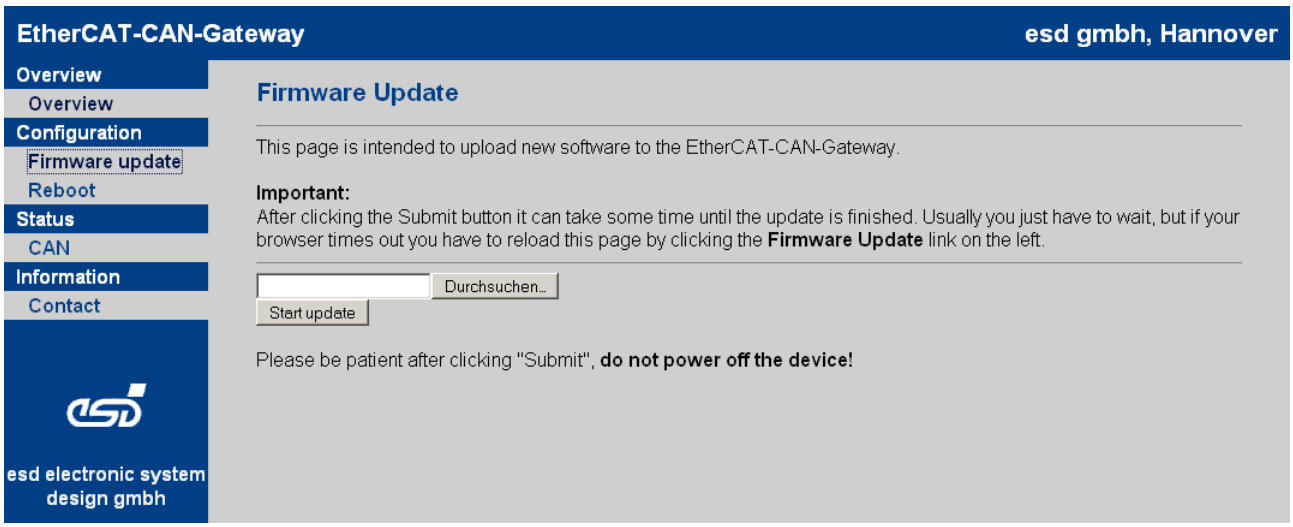


Figure 38: Firmware update

The upload of the file is handled through the web browser. Enter the file name or click the Choose... command button to select a file name.

The firmware update starts after confirmation of the entry with the command button Submit. This procedure will take some time. The progress of the update is recorded.

NOTICE
Do not interrupt the CAN-EtherCAT gateway power supply during a firmware update as this might result in unforeseeable operating conditions.

Example of a firmware update:

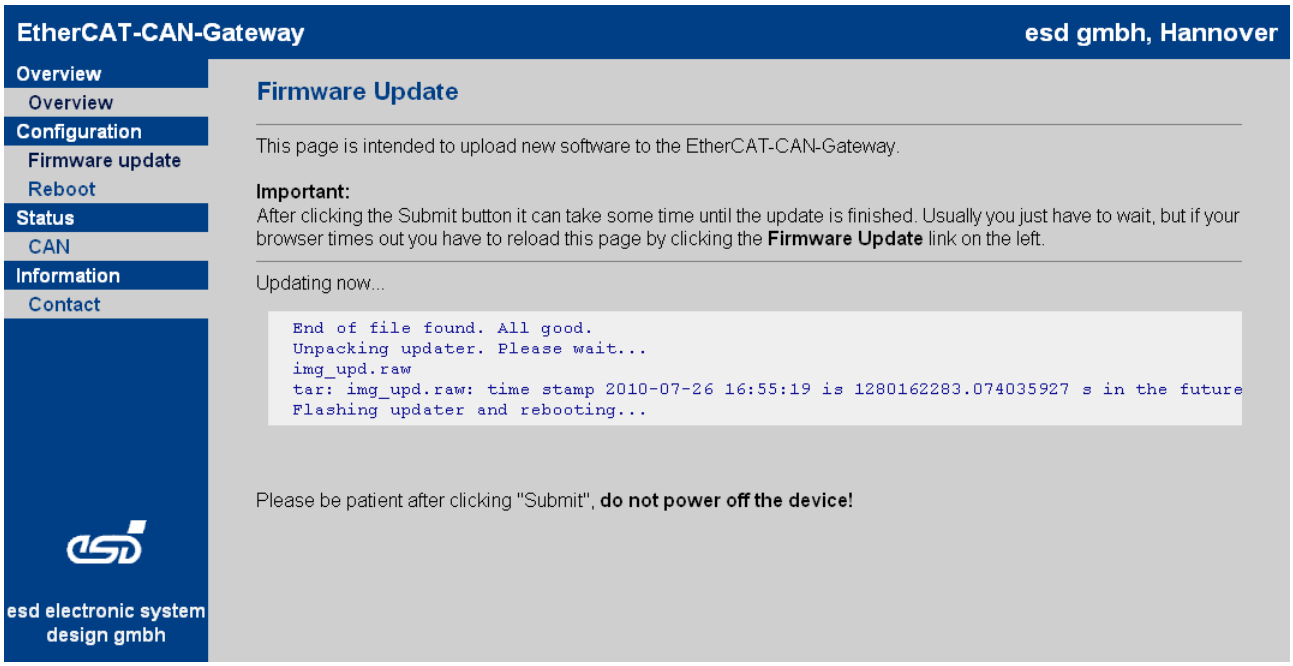


Figure 39: Firmware update output

Please wait until the firmware update is finished, which may take several minutes.

When finished the system will initiate an automatic restart.

6.1.3 Reboot

To initiate a system restart choose the menu item *Reboot*; then click the Reboot now command button.

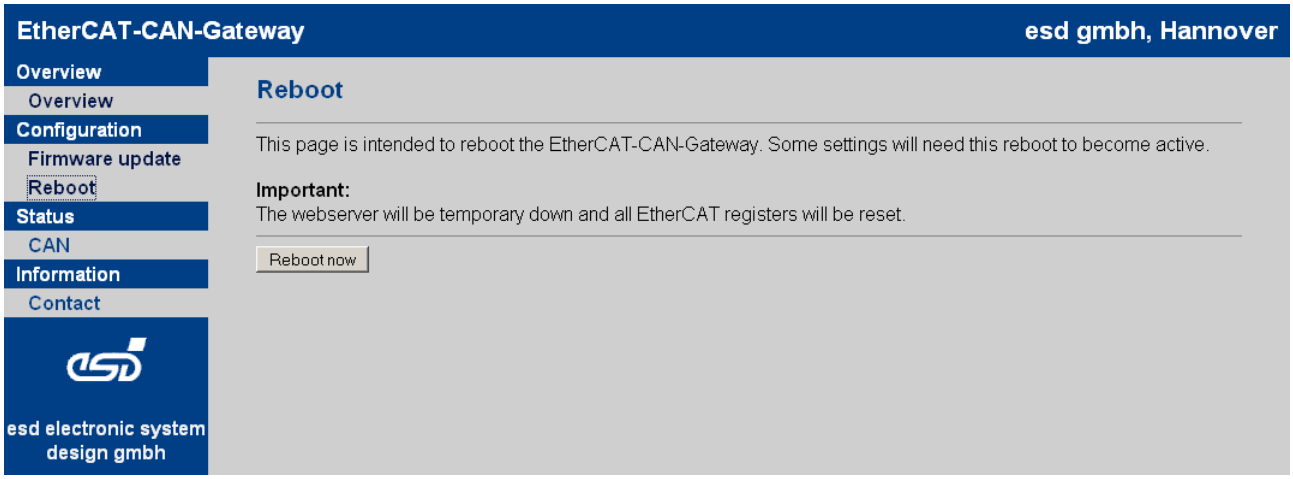


Figure 40: Reboot

6.2 Status

6.2.1 CAN Statistics

Click the *CAN* menu item to access the CAN bus statistics.

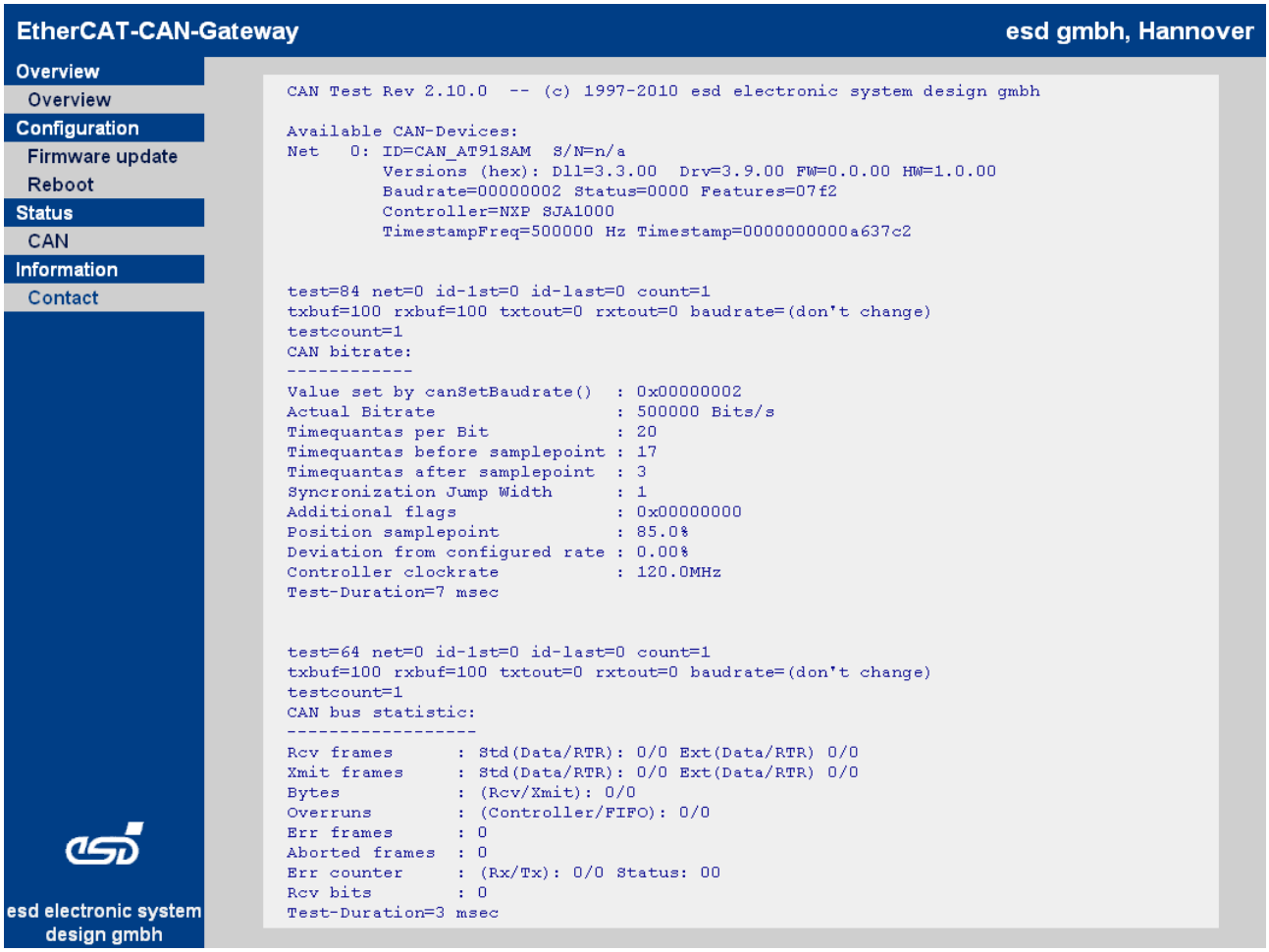


Figure 41: CAN status output

7. Technical Data

7.1 General Technical Data

Power supply voltage	Nominal voltage: typical: 24 V/DC, (min.: 18 V, max.: 32 V) Current consumption: (24 V, 20 °C): typ. 150 mA
Connectors	24V 24 V-power supply voltage (X1, 4-pin. COMBICON-connector with spring-cage connection) CAN CAN Bus interface (X2, 5-pin Phoenix Contact MC 1,5/5-GF-3,81) IN/OUT EtherCAT interface (X3A/B, 2x RJ45 socket) ETH Ethernet interface (X5, 8-pin. RJ45 socket) InRailBus CAN Bus interface and power supply voltage via InRailBus (X6, 5-pin TBUS-connector, accessory) Only for manufacturing purposes: DIAG DIAG interface (X4, USB connector type-B)
Temperature range	0 °C ... 50 °C ambient temperature
Humidity	max. 90%, non-condensing
Pollution degree	maximum permissible according to DIN EN 61131-2: Pollution Degree 2
Dimensions	Width: 22.5 mm, Height: 114.5 mm, Depth: 99 mm
Weight	130 g

Table 11: General data of the module

7.2 Microprocessor and Memory

CPU	ARM9-Prozessor, 240 MHz, AT91SAM9263
Data Flash	1 MB
NAND Flash	256 MB
SDRAM	32 MB

Table 12: Microprocessor and memory

7.3 CAN Interface

Number of CAN interfaces	1x CAN
CAN controller	integrated in CPU
CAN protocol	according to ISO 11898-1
Physical layer	High-speed CAN interface according to ISO 11898-2, bit rate from 50 kBit/s up to 1 Mbit/s
Electrical isolation	Isolation voltage U: 500 V (= withstand-impulse voltage according to DIN EN 60664-1)
Bus termination	terminating resistor has to be set externally, if required
Connector	CAN, 5-pin COMBICON (X2)

Table 13: Data of the CAN interface

7.4 EtherCAT Interface

Number of interfaces	1
Controller	Beckhoff ET1100
Bit rate	100BASE-TX, 100 Mbit/s
Connection	Twisted Pair (compatible to IEEE 802.3), 100BASE-TX
Electrical isolation	via transformer
Connector	2x RJ-45-socket with integrated LEDs in the front panel IN (X3B), OUT (X3A)

Table 14: Data of the EtherCAT interface

7.5 Ethernet Interface

Number of Ethernet interfaces	1
Bit rate	10BASE-T, 100BASE-TX, 10/100 Mbit/s
Connection	Twisted Pair (compatible to IEEE 802.3), 100BASE-TX,
Electrical isolation	via transformer
Connector	RJ-45-socket with integrated LEDs in the front panel (X5)

Table 15: Data of the Ethernet interface

7.6 DIAG, USB Interface

Design	USB, for manufacturing purposes only
USB interface	USB 2.0, Full-Speed, 12 Mbit/s
Connector	DIAG (X4), USB type B connector

Table 16: Data of the USB interface

7.7 Operating System and License Information

Operating system	QNX 6.5 (Firmware 1.0: QNX 6.4)

Bootloader	U-Boot
License information	This product uses the open source-bootloader "Das U-Boot". The U-Boot-source code is released under the terms of the GNU Public License (GPL). The complete text of the license is contained in the esd-document "3rd Party Licensor Notice" as part of the product documentation. esd provides the complete bootloader-source code on request. esd strives to restore all changes on the bootloader into the official sources. The homepage of the U-Boot project is: http://www.denx.de/wiki/U-Boot .

HTTP server	thttpd - tiny/turbo/throttling HTTP server
Copyright Information	Copyright (C) 1995,1998,1999,2000,2001 by Jef Poskanzer <jef@mail.acme.com>. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

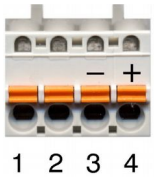
8. Interfaces and Connector Assignments

8.1 24 V-Power Supply Voltage

The power supply voltage can be fed via connector X1 or optional via InRailBus (connector assignment see page 86)

Device socket: Phoenix Contact MSTBO 2,5/4-G1L-KMGY
Line connector: Phoenix Contact FKCT 2,5/4-ST, 5.0 mm pitch,
spring cage connection,
Phoenix Contact order No.: 19 21 90 0 (included in the scope of delivery)

Pin Position:



Pin Assignment:

Labelling of the CAN-EtherCAT			24V	
	.	**.**	M	P
Connector label	(free)	(free)	-	+

Pin-Nr.	1	2	3	4
Signal	P24 (+ 24 V)	M24 (GND)	M24 (GND)	P24 (+ 24 V)

Please refer to the connecting diagram page 11.

The pins 1 and 4 are connected internally.
The pins 2 and 3 are connected internally.

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 the connector X1 and to supply the power supply voltage to another CAN module station!

Signal Description:

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

8.2 CAN

8.2.1 CAN Interface

The CAN bus signals are electrically isolated from the other signals via digital isolator and DC/DC-converter.

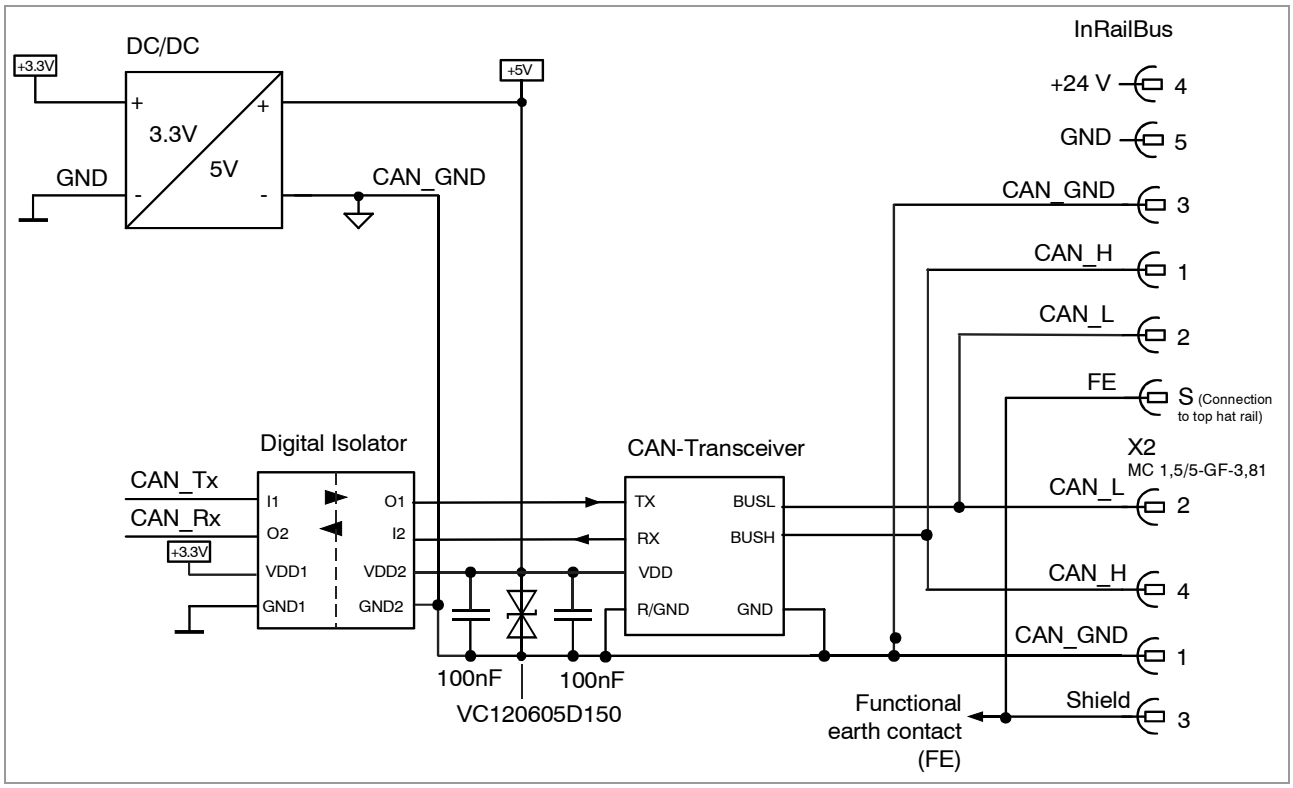


Figure 42: CAN-Interface

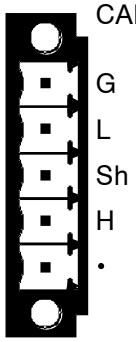
The CAN interface can be connected via CAN connector (X2) or optionally via InRailBus (connector assignment see page 86).

8.2.2 CAN Connector

Device connector : Phoenix Contact MC 1,5/5-GF-3,81
Line connector: Phoenix Contact FK-MCP 1,5/5-STF-3,81, spring-cage connection
Phoenix Contact Order No.: 1851261 (included in delivery)

Pin Position:

(device connector view)



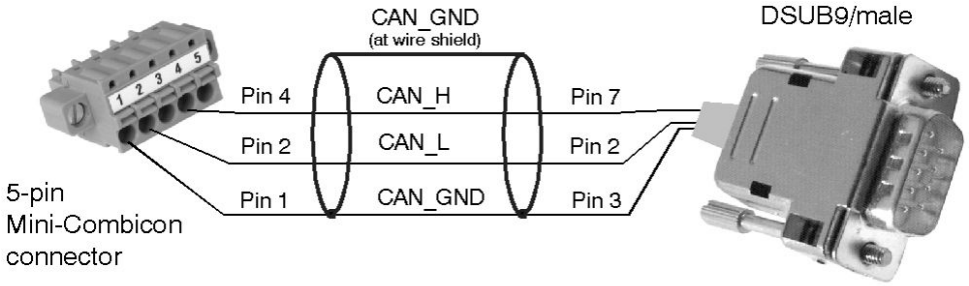
Pin Assignment:

Pin	Signal
1	CAN_GND
2	CAN_L
3	Shield
4	CAN_H
5	-

Signal description:

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

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



The assignment of the 9-pin DSUB-connector is designed according to CiA DS-102.

The assignment of the 5-pin Mini- COMBICON is designed according to CiA DR-303 Part 1

8.3 24 V and CAN via InRailBus

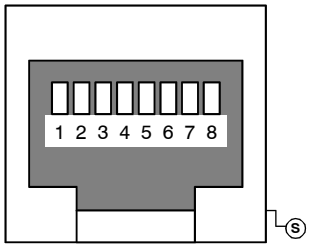
Power supply voltage and CAN can optionally be fed via InRailBus.
Use the mounting-rail bus connector of the CBX-InRailBus for the connection via the InRailBus, see order information (page 92).
Read and follow the instructions for connecting power supply and CAN signals via InRailBus (see page 87)!

8.4 Ethernet 100BASE-TX (IEEE 802.3)

Device connector: RJ45 socket, 8-pin, according to IEEE 802.3-2008, Table 25-3 'UTP MDI contact assignment'

The ports have an identical pin assignment.

Pin Position:



Pin Assignment:

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

S	Shield	

Pin 1 to 8 are connected to a line termination.

Signal Description:

MDI0+/-, MDI0+/-, MDI1+/-, MDI1+/- ...	EtherCAT data lines
- ...	reserved for future applications, do not connect!
Shield...	line shield connection (using hat rail mounting direct contact to the mounting rail potential)

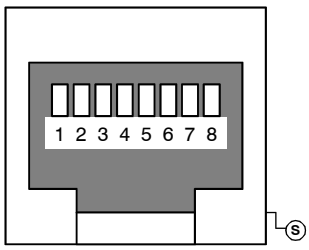
NOTICE
Permissible cables: To ensure function in networks with up to 100 MBit/s cables of Cat. 5e or better have to be used. To ensure the EC Conformity cables with shielding SF/UTP or better have to be used.

8.5 EtherCAT

Device connector: RJ45 socket, 8-pin, according to IEEE 802.3-2008, Table 25-3 'UTP MDI contact assignment'

The ports have an identical pin assignment.

Pin Position:



Pin Assignment:

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

S	Shield	

Pin 1 to 8 are connected to a line termination.

Signal Description:

MDI0+/-, MDI0+/-, MDI1+/-, MDI1+/- ...	EtherCAT data lines
- ...	reserved for future applications, do not connect!
Shield...	line shield connection (using hat rail mounting direct contact to the mounting rail potential)

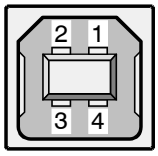
NOTICE
Permissible cables: To ensure function in networks with up to 100 MBit/s cables of Cat. 5e or better have to be used. To ensure the EC Conformity cables with shielding SF/UTP or better have to be used.

8.6 DIAG

The USB interface DIAG does not fulfill a function and is only used for manufacturing purposes.

NOTICE
The CAN-EtherCAT may only be operated with USB nets with USB interfaces with versions 1.1 or 2.0!
Operability can only be guaranteed for these USB interfaces.

Pin Position:



Pin Assignment:

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

USB socket (type B)

8.7 Conductor Connection/Conductor Cross Sections

The following table contains an extract of the technical data of the cable plugs.

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

1 Technical Data from Phoenix Contact website, printed circuit board connector, plug component

9. Correct Wiring of Electrically Isolated CAN Networks

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

9.1 Standards concerning CAN Wiring

The flexibility in CAN network design is one of the key strengths of the various extensions and additional standards like e.g. CANopen, ARINC825, DeviceNet and NMEA2000 that have been built on the original ISO 11898-2 CAN standard. In using this flexibility comes the responsibility of good network design and balancing these tradeoffs.

Many CAN organizations and standards have scaled the use of CAN for applications outside the original ISO 11898. They have made system level tradeoffs for data rate, cable length, and parasitic loading of the bus.

However for CAN network design margin must be given for signal loss across the complete system and cabling, parasitic loadings, network imbalances, ground offsets against earth potential and signal integrity. **Therefore the practical maximum number of nodes, bus length and stub length are typically much lower.**

esd has concentrated her recommendations concerning CAN wiring on the specifications of the ISO 11898-2. Thus this wiring hints forgoes to describe the special features of the derived standards CANopen, ARINC825, DeviceNet and NMEA2000.

The consistent compliance to ISO 11898-2 offers significant advantages:

- Durable operation due to well proven design specifications
- Minimizing potential failures due to sufficient margin to physical limits
- Trouble-free maintenance during future network modifications or during fault diagnostics due to lack of exceptions

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

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

9.2.1 General Rules

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

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

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

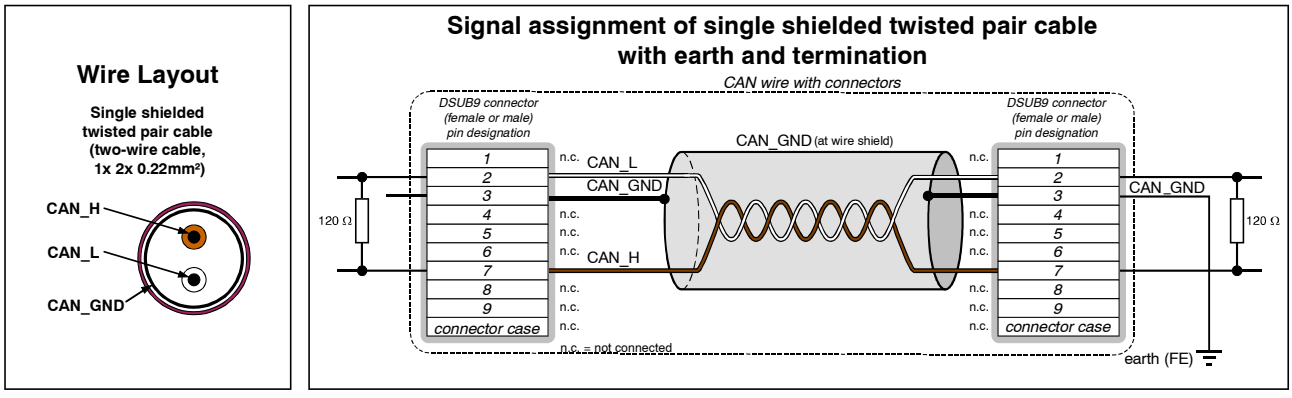


Figure 43: CAN wiring for light industrial environment

9.2.2 Cabling

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

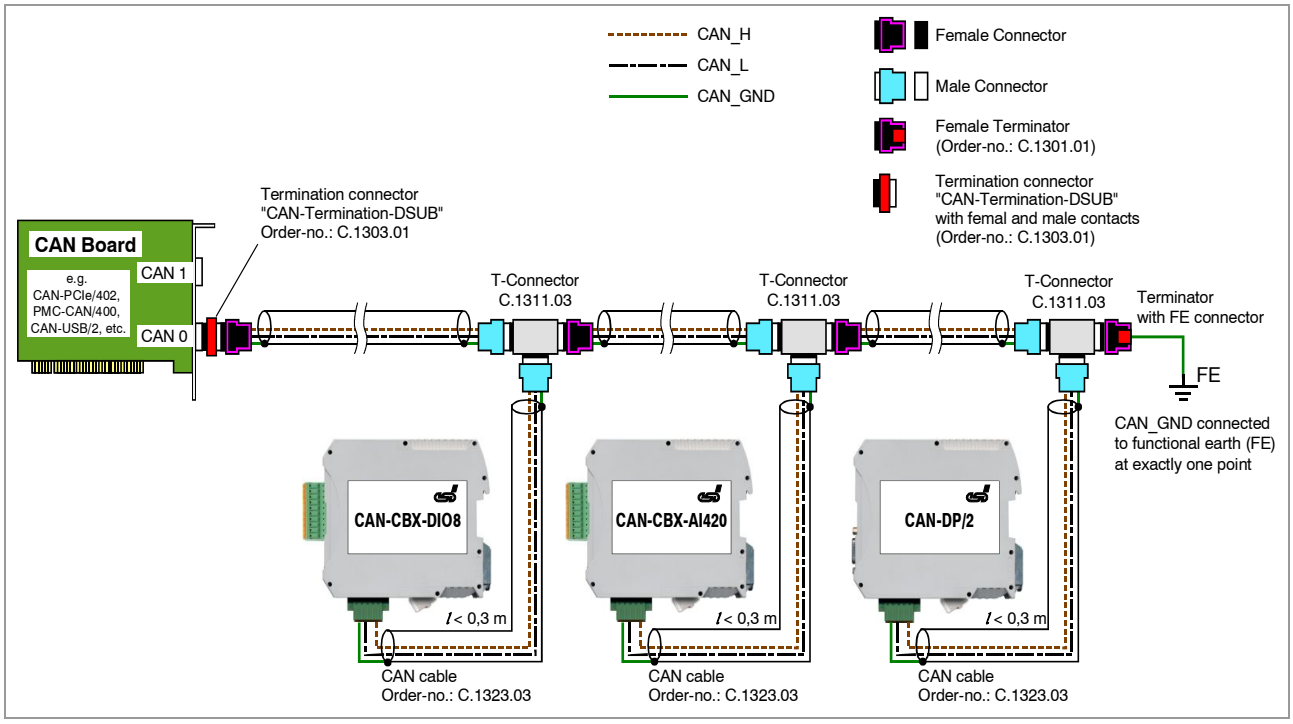


Figure 44: Example for proper wiring with single shielded single twisted pair wires

9.2.3 Termination

- A termination resistor has to be connected at both ends of the CAN bus.
  If an integrated CAN termination resistor which is equipped at the CAN interface at the end of the bus is connected, this one has to be used for termination instead of an external CAN termination plug.

- 9-pin DSUB-termination connectors with integrated termination resistor and male and female contacts are available from esd (order no. C.1303.01).

- DSUB termination connectors with male contacts (order no. C.1302.01) or female contacts (order no. C.1301.01) and additional functional earth contact are available, if CAN termination and grounding of CAN_GND is required.

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

9.3.1 General Rules

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

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

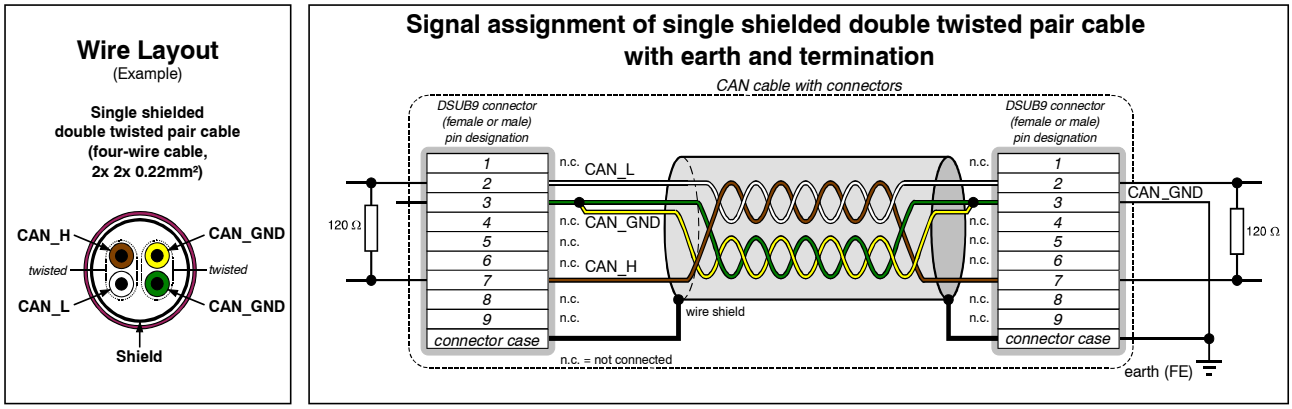


Figure 45: CAN wiring for heavy industrial environment

9.3.2 Device Cabling

NOTICE
If single shielded *double* twisted pair cables are used, realize the T-connections by means of connectors that support connection of two CAN cables at one connector where the cable's shield is looped through e.g. DSUB9 connector from ERNI (ERBIC CAN BUS MAX, order no.:154039).

The usage of esd's T-connector type C.1311.03 is not recommended for single shielded *double* twisted pair cables because the shield potential of the conductive DSUB housing is not looped through this T-connector type.

If a mixed application of single twisted and double twisted cables is unavoidable, take care that the CAN_GND line is not interrupted!

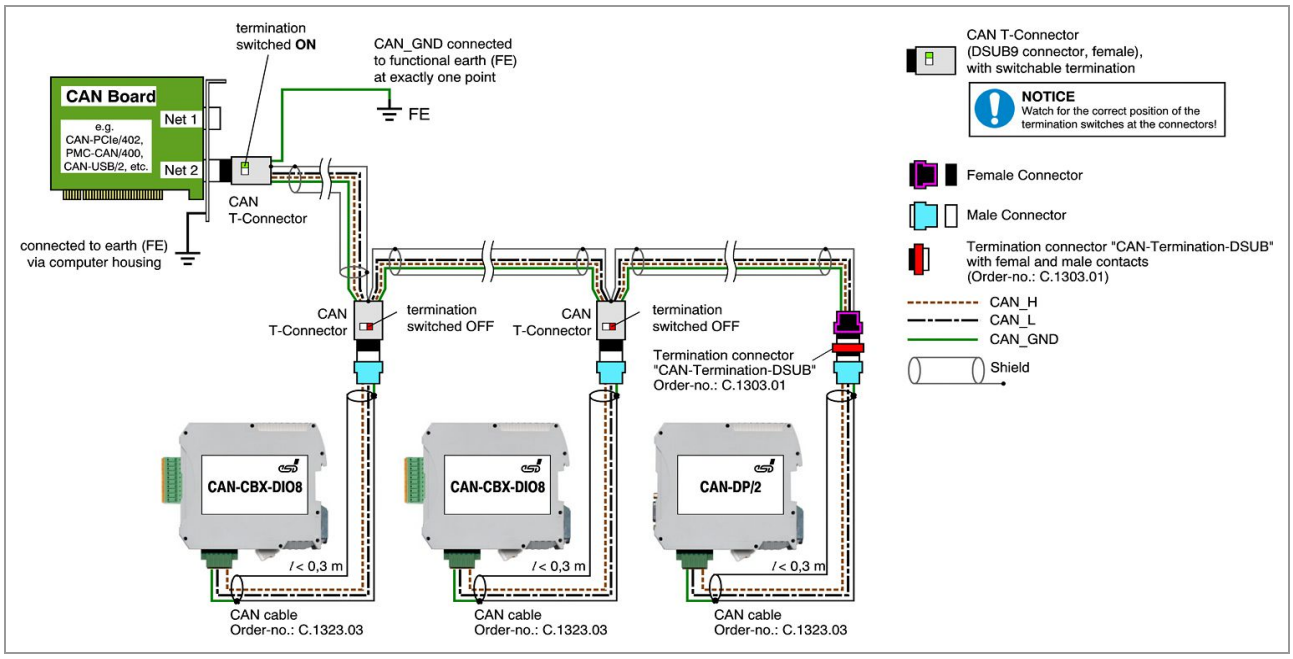


Figure 46: Example of proper wiring with single shielded double twisted pair cables

9.3.3 Termination

- A termination resistor has to be connected at both ends of the CAN bus.
  If an integrated CAN termination resistor which is equipped at the CAN interface at the end of the bus is connected, this one has to be used for termination instead of an external CAN termination plug.

- 9-pin DSUB-termination connectors with integrated termination resistor and male and female contacts are available from esd (order no. C.1303.01).

- 9-pin DSUB-connectors with integrated switchable termination resistor can be ordered e.g. from ERNI (ERBIC CAN BUS MAX, female contacts, order no.:154039).

9.4 Electrical Grounding

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

9.5 Bus Length

NOTICE
Please note that the cables, connectors and termination resistors used in CANopen networks shall meet the requirements defined in ISO11898-2.
In addition, further recommendations of the CiA, like standard values of the cross section, depending on the cable length, are described in the CiA recommendation CiA 303-1 (see CiA 303 CANopen Recommendation - Part 1: „Cabling and connector pin assignment“, Version 1.8.0, Table 2).

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

Table 17: Recommended cable lengths at typical bit rates (with esd-CAN interfaces)

- Optical couplers are delaying the CAN signals. esd modules typically reach a wire length of 37 m at 1 Mbit/s within a proper terminated CAN network without impedance disturbances like e.g. caused by cable stubs > 0.3 m.

9.6 Examples for CAN Cables

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

9.6.1 Cable for light industrial Environment Applications (Two-Wire)

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

9.6.2 Cable for heavy industrial Environment Applications (Four-Wire)

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

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

10. CAN Troubleshooting Guide

The CAN Troubleshooting Guide is a guide to find and eliminate the most frequent hardware-error causes in the wiring of CAN networks.

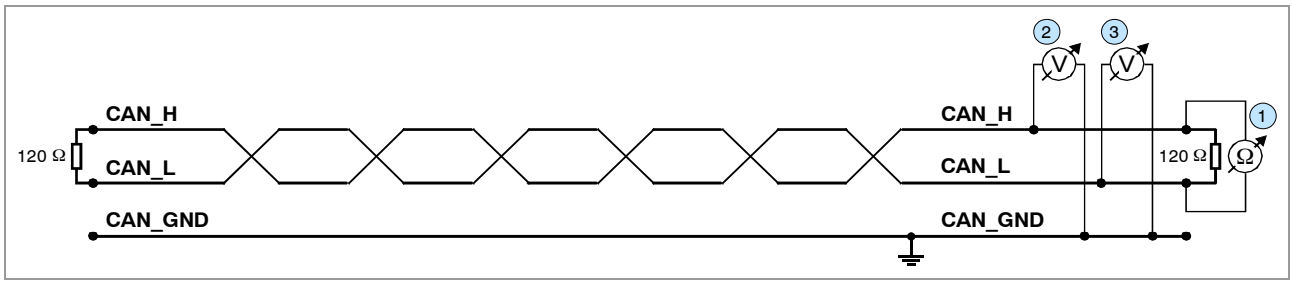


Figure 47: Simplified diagram of a CAN network

10.1 Termination

The termination is used to match impedance of a node to the impedance of the transmission line being used. When impedance is mismatched, the transmitted signal is not completely absorbed by the load and a portion is reflected back into the transmission line. If the source, transmission line and load impedance are equal these reflections are avoided. This test measures the series resistance of the CAN data pair conductors and the attached terminating resistors.

To test it ,please

1. Turn off all power supplies of the attached CAN nodes.
2. Measure the DC resistance between CAN_H and CAN_L at one end of the network ① (see figure above).

The measured value should be between 50 Ω and 70 Ω.

If the value is below 50 Ω, please make sure that:
- there is no **short circuit** between CAN_H and CAN_L wiring
- there are **not more than two** terminating resistors connected
- the nodes do not have faulty transceivers.

If the value is higher than 70 Ω, please make sure that:
- there are no open circuits in CAN_H or CAN_L wiring
- your bus system has two terminating resistors (one at each end) and that they are 120 Ω each.

10.2 Electrical Grounding

The CAN_GND of the CAN network should be connected to the functional earth potential (FE) at only **one** point. This test will check if the CAN_GND is grounded in several places.
To test it, please

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).
3. Reconnect CAN_GND to earth potential.

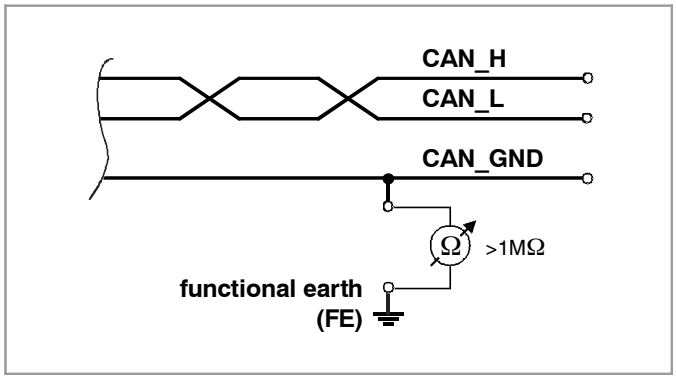


Figure 48: Simplified schematic diagram of ground test measurement

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

10.3 Short Circuit in CAN Wiring

A CAN bus might possibly still be able to transmit data if there is a short circuit between CAN_GND and CAN_L, but generally the error rate will increase strongly. Make sure that there is no short circuit between CAN_GND and CAN_L!

10.4 CAN_H/CAN_L-Voltage

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. Faulty transceivers can cause the idle voltages to vary and disrupt network communication.

To test for faulty transceivers, please

1. Turn on all supplies.
2. Stop all network communication.
3. Measure the DC voltage between CAN_H and CAN_GND (2) (see figure at previous page).
4. Measure the DC voltage between CAN_L and CAN_GND (3) (see figure at previous page).

Normally the voltage should be between 2.0 V and 3.0 V.

If it is lower than 2.0 V or higher than 3.0 V, it is possible that one or more nodes have faulty transceivers. For a voltage lower than 2.0 V please check CAN_H and CAN_L conductors for continuity.

To find the node with a faulty transceiver within a network please test the CAN transceiver resistance (see below) of the nodes.

10.5 CAN Transceiver Resistance Test

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

To measure the current leakage through the CAN circuits, please use a resistance measuring device and:

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

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

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

The measured resistance has to be about 500 kΩ for each signal. If it is much lower, the CAN transceiver it is probably faulty.
Another indication for a faulty transceiver is a very high deviation between the two measured input resistances (>> 200 %).

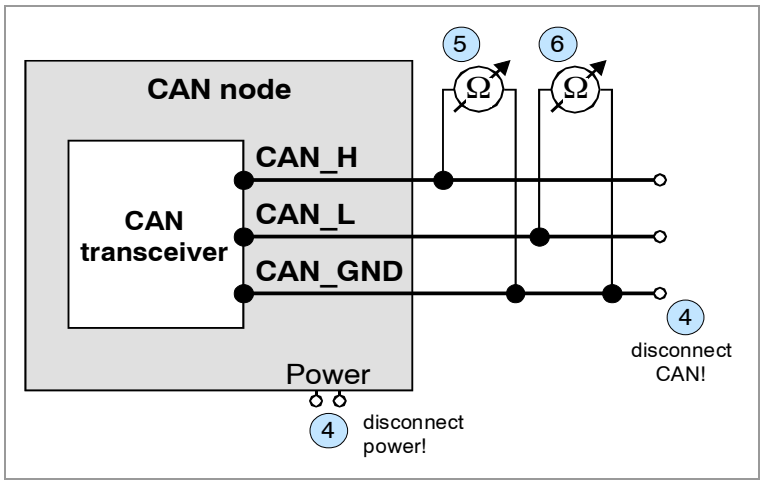


Figure 49: Measuring the internal resistance of CAN transceivers

10.6 Support by esd

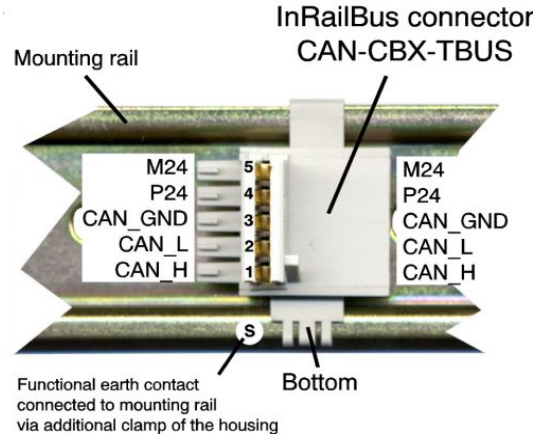
If you have executed the fault diagnostic steps of this troubleshooting guide and you even can not find a solution for your problem our support department will be able to assist.
Please contact our support via email at **support@esd.eu** or by phone **+40-511-37298-130**.

11. Option InRailBus

11.1 Connector Assignment 24V and CAN via InRailBus

Connector type: InRailBus PCB direct plug-in mount
CAN-CBX-TBUS
(Phoenix Contact ME 22,5 TBUS 1,5/5-ST-3,81 KMGY)

Connector View:



Pin Assignment:

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

S	FE (PE_GND)

Signal Description:

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

11.2 Using InRailBus

INFORMATION
This chapter describes the installation when using the InRailBus for CAN-CBX-modules. For the CAN-EtherCAT gateway the following chapters apply accordingly.

11.2.1 Installation of the Module Using InRailBus Connector

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

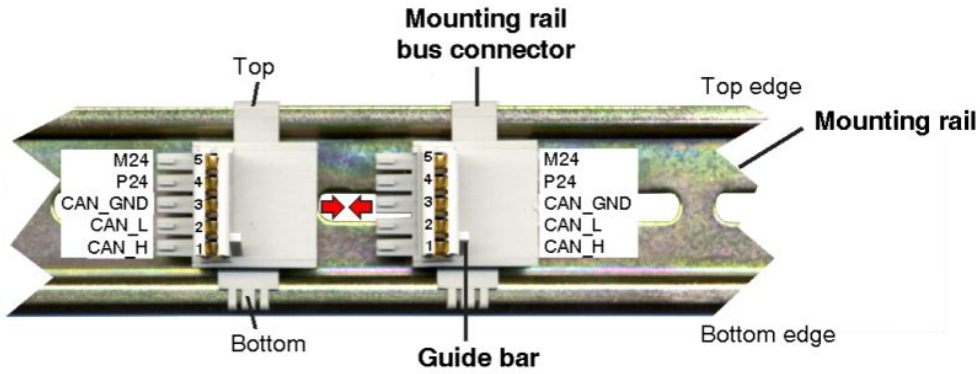


Figure. 50: Mounting rail with bus connector

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

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

Figure. 51: Mounting CAN-CBX modules

3. Swivel the CAN-CBX module onto the mounting rail in pressing the module downwards according to the arrow as shown in figure 51. The housing is mechanically guided by the DIN rail bus connector.

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

Figure. 52: Mounted CAN-CBX module

11.2.2 Connecting Power Supply and CAN Signals to CBX-InRailBus

To connect the power supply and the CAN-signals via the InRailBus, a terminal plug is needed. The terminal plug is not included in delivery and must be ordered separately (order no.: C.3000.02, see order information for InRailBus Accessories, page 92).

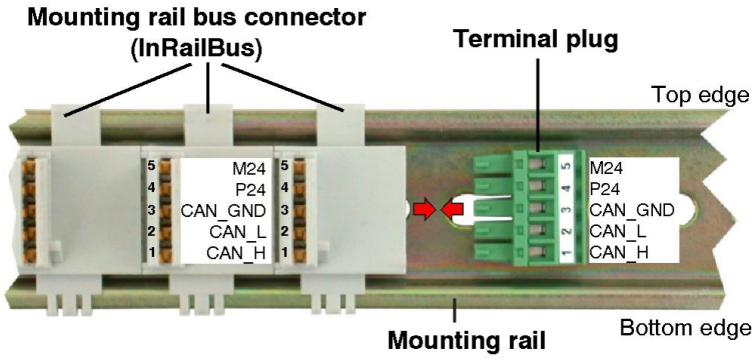


Figure. 53: Mounting rail with InRailBus and terminal plug

Plug the terminal plug into the socket on the right of the mounting-rail bus connector of the InRailBus, as described in Figure 53. Then connect the CAN interface and the power supply voltage via the terminal plug.

11.2.3 Connection of the Power Supply Voltage

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

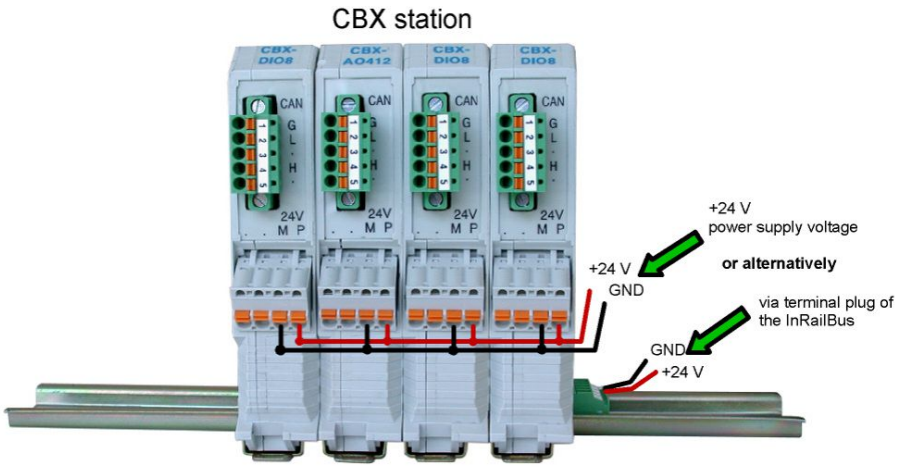


Figure. 54: Connecting the power supply voltage to the CAN-CBX station

11.2.4 Connection of CAN

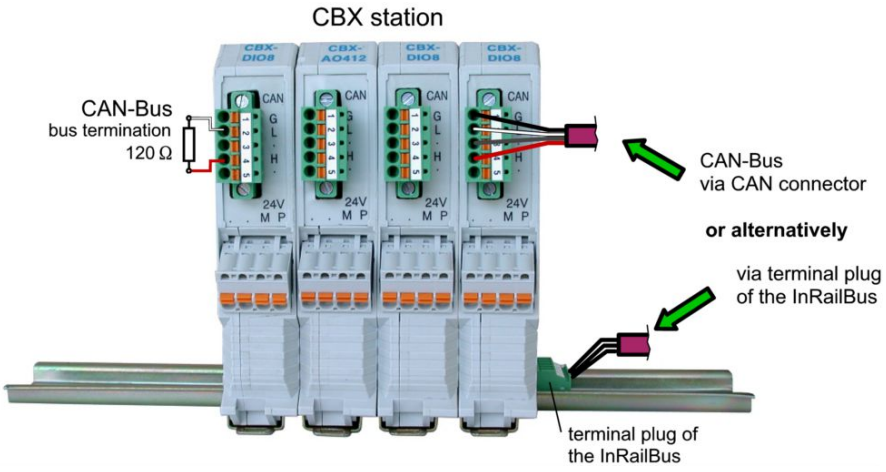


Figure. 55: Connecting the CAN signals to the CAN-CBX station

Generally the CAN signals can be fed via the CAN connector of the first CAN-CBX module of the CBX station. The signals are then connected through the CAN-CBX station via the InRailBus. To lead through the CAN signals the CAN bus connector of the last CAN-CBX module of the CAN-CBX station has to be used. The CAN connectors of the CAN-CBX modules which are not at the ends of the CAN-CBX station must not be connected to the CAN bus, because this would cause incorrect branching.
A bus termination must be connected to the CAN connector of the CAN-CBX module at the end of the CBX-InRailBus (see Fig. 55), if the CAN bus ends there.

11.3 Remove the CAN-CBX Module from InRailBus

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

Release the module from the mounting rail in moving the foot catch (see Fig. 52) downwards (e.g. with a screwdriver). Now the module is detached from the bottom edge of the mounting rail and can be removed.

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

12. Declaration of Conformity

EU-KONFORMITÄTSERKLÄRUNG
EU DECLARATION OF CONFORMITY

Adresse / *Address*
esd electronic system design gmbh
Vahrenwalder Str. 207
30165 Hannover
Germany

esd erklärt, dass das Produkt *esd declares, that the product*	Typ, Modell, Artikel-Nr. *Type, Model, Article No.*
CAN-EtherCAT	**C.2922.02**
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.*	**H-K00-0336-09,** **H-Z01-0336-14**
Das Produkt entspricht damit der EU-Richtlinie „EMV" *Therefore the product conforms to the EU Directive 'EMC'*	**2014/30/EU**
Das Produkt entspricht der EU-Richtlinie „RoHS" *The product conforms to the EU Directive 'RoHS'*	**2011/65/EU**

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

Name / *Name*	T. Ramm
Funktion / *Title*	CE-Koordinator / *CE Coordinator*
Datum / *Date*	Hannover, 2015-02-12

Thorsten Ramm

Rechtsgültige Unterschrift / *authorized signature*

I:\Texte\Doku\MANUALS\CAN\CAN-EtherCAT\Konformitätserklärung\CAN-EtherCAT_EU-Konformitaetserklaerung_2015-02-12.odt

13. Order Information

Type	Properties	Order No.
CAN-EtherCAT	EtherCAT/CAN gateway, documentation and EtherCAT Slave Information (ESI) file on CD	C.2922.02
Accessories		
 **CAN-CBX-TBUS**	Mounting-rail bus connector of the CBX-InRailBus for CAN-CBX modules (order separately)	C.3000.01
 **CAN-CBX-TBUS-Connector**	Terminal plug of the CBX-InRailBus for the connection of the +24V power supply voltage and the CAN interface Female type	C.3000.02
 **CAN-CBX-TBUS-Connection adapter**	Terminal plug of the CBX-InRailBus for the connection of the +24V power supply voltage and the CAN-Interface Male type	C.3000.03

Table 18: Order information

PDF Manuals
Manuals are available in English and usually in German as well. Available manuals are listed in the following table
Please download the manuals as PDF documents from our esd website www.esd.eu for free.

Manuals		Order No.
CAN-EtherCAT-MD	Manual in German	C.2922.20
CAN-EtherCAT-ME	Manual in English	C.2922.21

Table 19: 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.