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 • Fax: 0511/37 29 8-68
Phone: 0511/37 29 80 • International: +49-5 11-37 29 80

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

esd assumes no responsibility for the use of any circuitry other than circuitry which is part of a product of esd gmbh.

esd does not convey to the purchaser of the product described herein any license under the patent rights of esd gmbh nor the rights of others.

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

USA / Canada:
esd electronics Inc.
525 Bernardston Road
Suite 1
Greenfield, MA 01301
USA

Phone:	+1-800-732-8006
Fax:	+1-800-732-8093
E-mail:	us-sales@esd-electronics.com
Internet:	www.esd-electronics.us

Document file:	I:\Texte\Doku\MANUALS\CAN\CAN-EtherCAT\Englisch\CAN-EtherCAT_manual_en_11.odt
Date of print:	2011-05-31
Document type number:	DOC0800

Firmware version:	from Rev. 1.0

Document History

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

Revision	Chapter	Changes versus previous version	Date
1.1	-	First English version	2011-05-31

Technical details are subject to change without further notice.

Safety Instructions

- When working with CAN-EtherCAT follow the instructions below and read the manual carefully to protect yourself from injury and the CAN-EtherCAT from damage.
- Do not open the housing of the CAN-EtherCAT.
  In order to prevent overvoltage damage due to thunder storm, unplug the CAN-EtherCAT from CAN, Ethernet, USB and EtherCAT beforehand.
- 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.
- Do not use damaged or defective cables to connect the CAN-EtherCAT and follow the CAN wiring hints in chapter: "Correctly Wiring Electrically Isolated CAN Networks".
- 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 or PELV) according to EN 60950-1, complies with this conditions.

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 interferences in which case the user may be required to take adequate measures.

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.

Table of contents

1. Overview

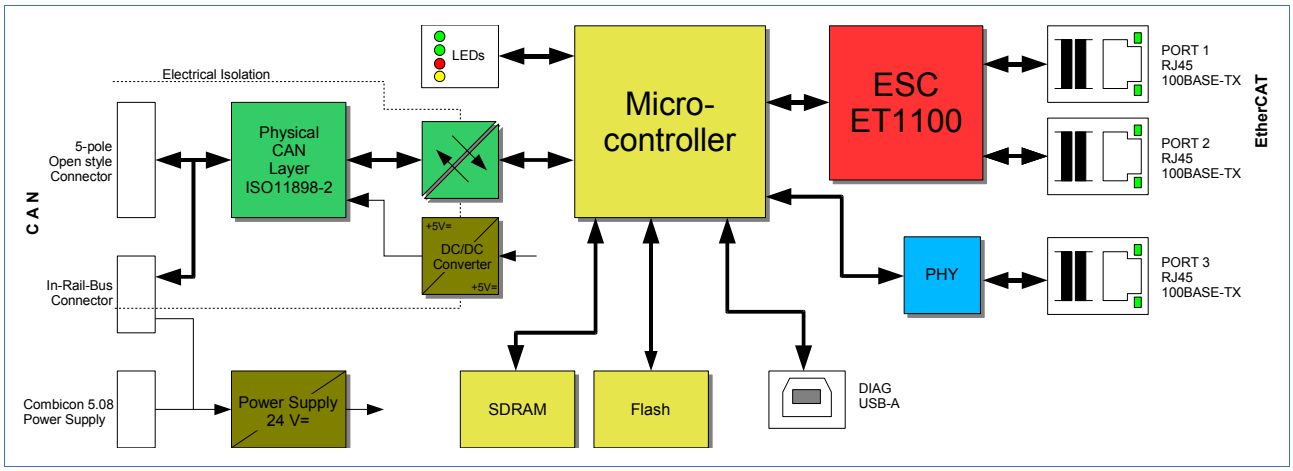


Figure 1: Block circuit diagram

The CAN-EtherCAT device connects an EtherCAT network with one CAN network. In this case the gateway acts as an EtherCAT slave device with a maximum of 8 kBytes of input or output data on the EtherCAT bus.

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 high-speed CAN interface is compliant with ISO 11898-2 and it supports transfer rates up to 1 MByte/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, are electrically isolated.

The configuration of the CAN-EtherCAT is accomplished through the EtherCAT master.
CAN diagnostics and firmware updates are realized via web interface.

2. Hardware Installation

2.1 Connections

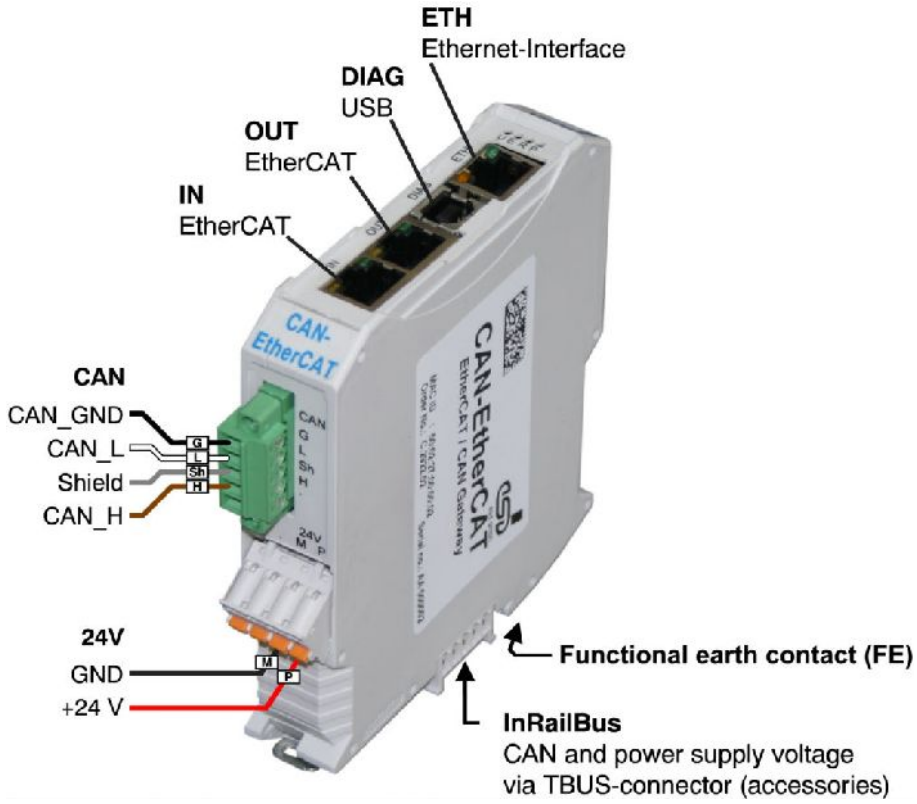


Fig. 2: Connections for operating condition

See also page 56 for signal assignment of the CAN connectors.

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

2.2 Position of the LEDS

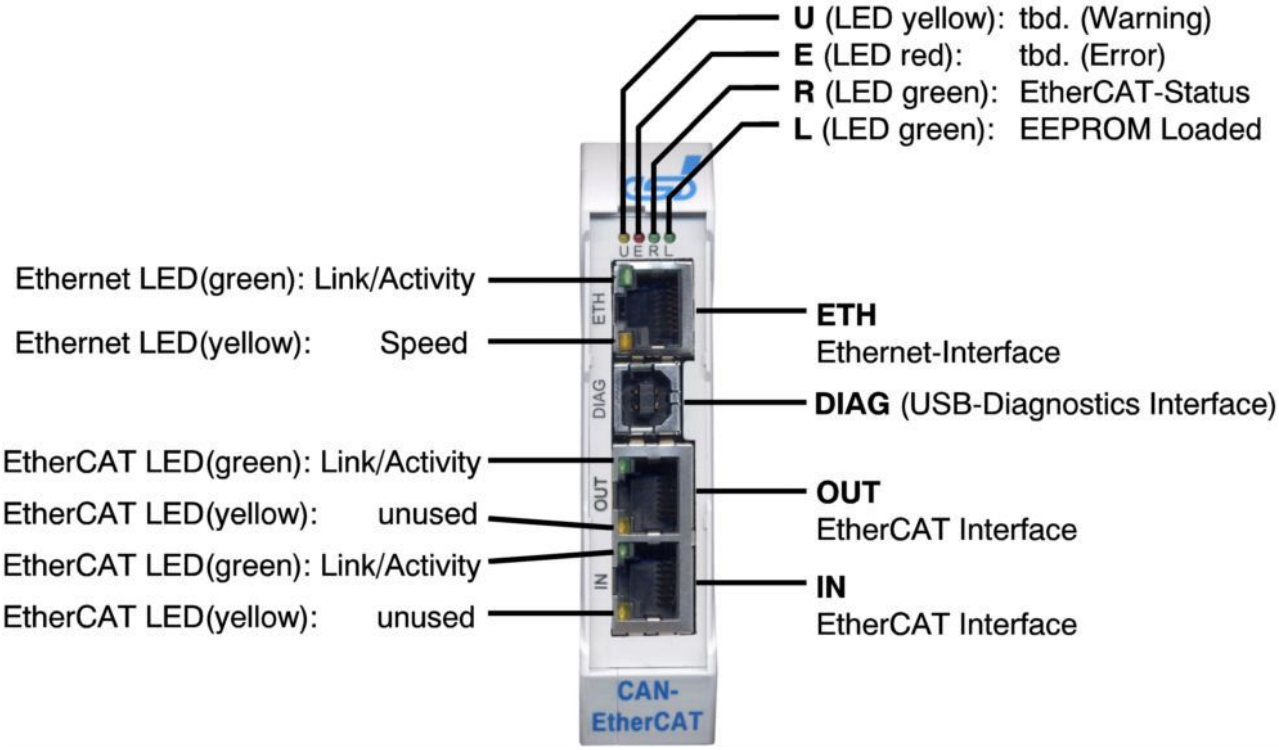


Figure 3: Connectors and LEDs

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	Color	Function	LED Status	Description	Schematic Reference
U	yellow	EtherCAT-Warning	off	currently no function	LED1A
			on		
E	red	EtherCAT-Error	off	currently no function	LED1B
			on		
R	green	EtherCAT-Status	off	Init	LED1C
			slow blinking	Pre-Operational	
			fast blinking	Safe Operation	
			on	Operation	
L	green	EEPROM Loaded	off	unable to retrieve ET1100 configuration from EEPROM	LED1D
			on	successful retrieval of ET1100 configuration from EEPROM	

Table 3: 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!**	4
1.	Mount and connect the CAN-EtherCAT gateway and connect the interfaces (Power supply, CAN bus, EtherCAT, and – if applicable – Ethernet).	8
2.	Please note that the CAN bus has to be terminated at both ends! **esd** offers special T-connectors and termination connectors. Additionally the CAN_GND signal has to be connected to earth at **exactly one** point in the CAN network. All esd termination devices will provide a corresponding contact. Any CAN node that does not support a galvanic isolation represents the equivalent of a Ground (GND) connection.	-
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.	13
5.	Configure the CAN-EtherCAT gateway with the EtherCAT configurator.	12

4. EtherCAT Configurator

4.1 CAN-EtherCAT Gateway Application Example

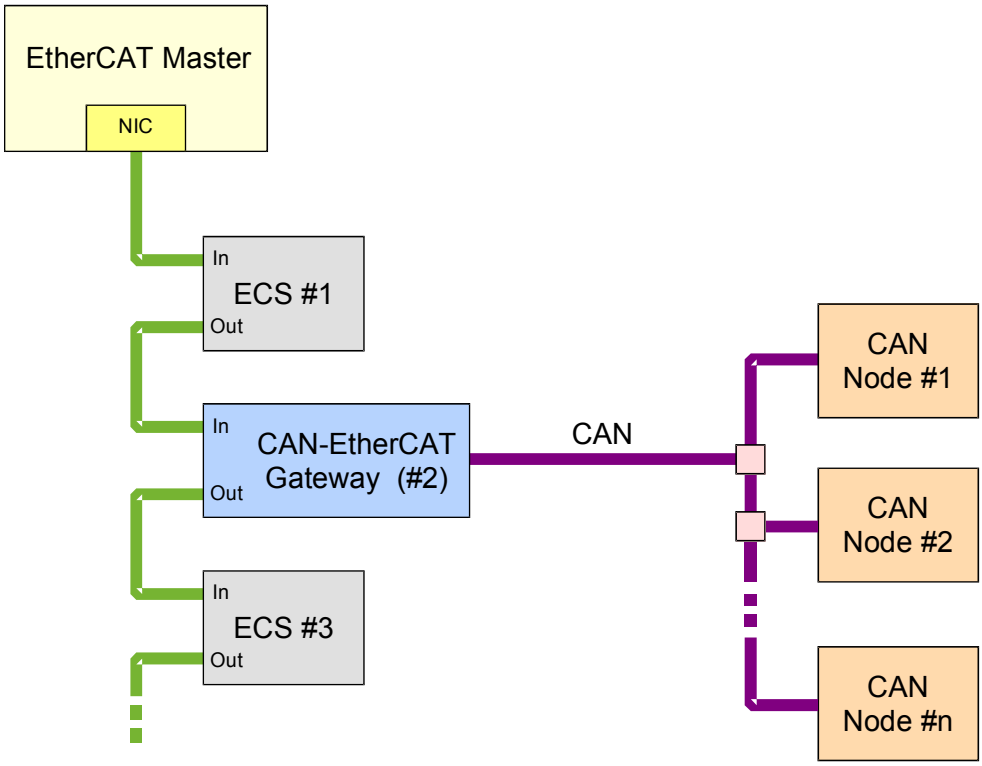


Figure 4: CAN-EtherCAT gateway connection example

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

4.2 Configuration Sequence

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 files (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:

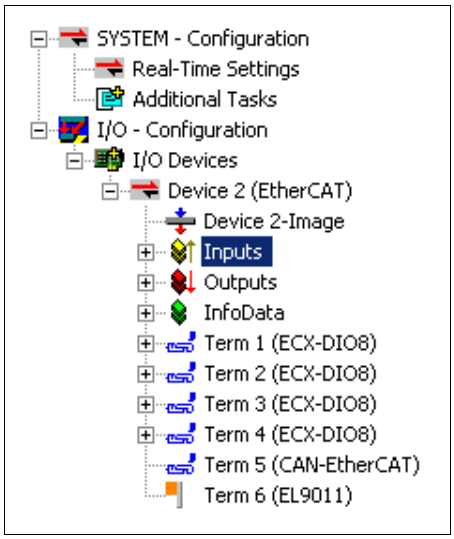


Figure 5: CAN-EtherCAT in device tree view

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

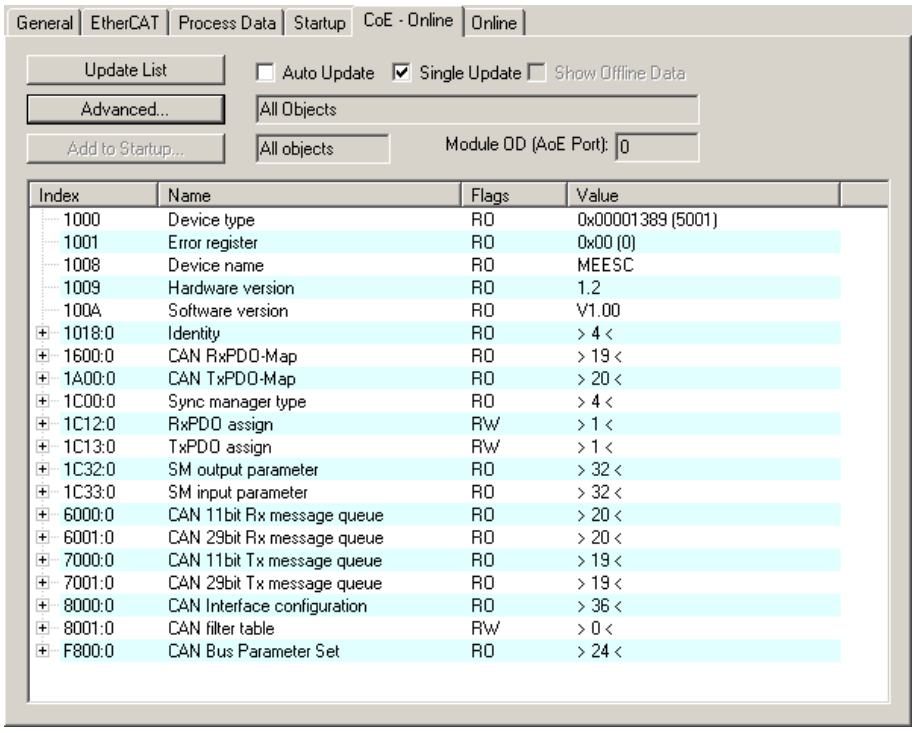


Figure 6: Object dictionary

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

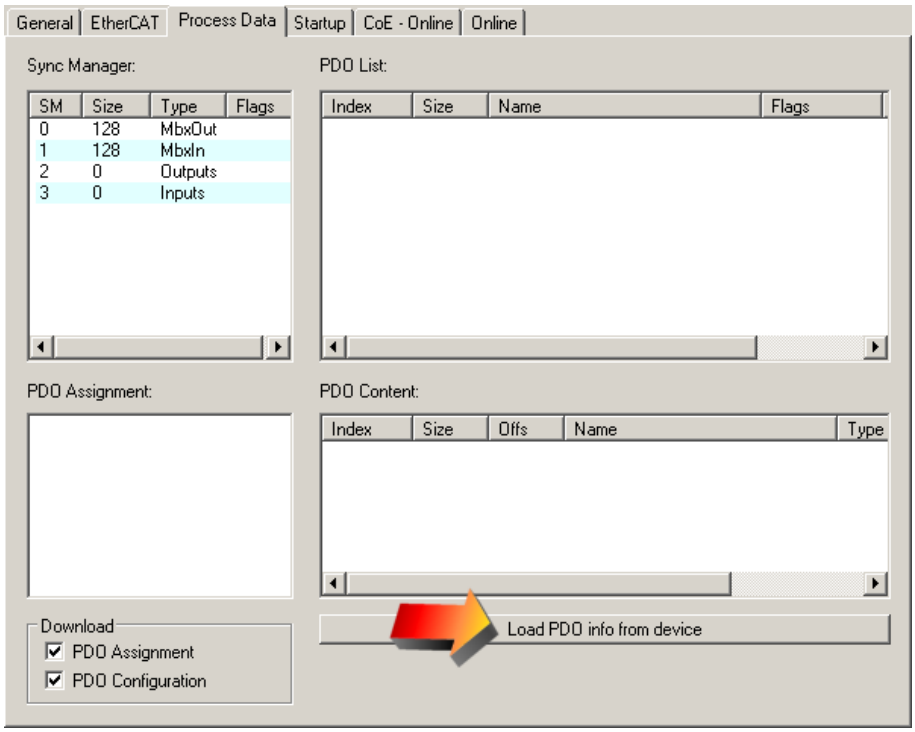


Figure 7: Loading process data

The configurator will now display the process data:

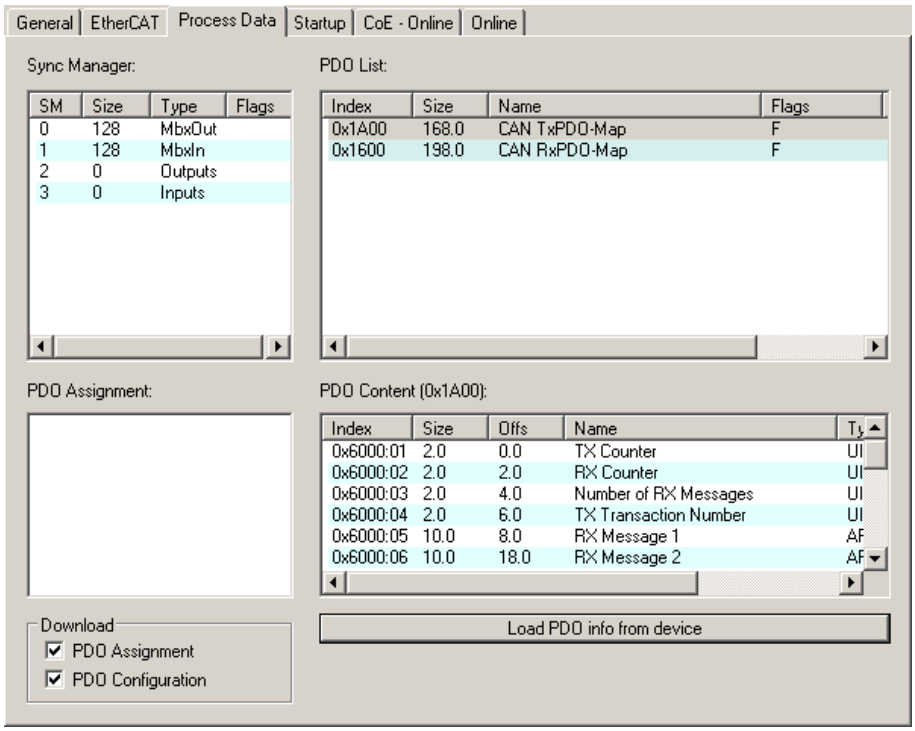


Figure 8: Process data display

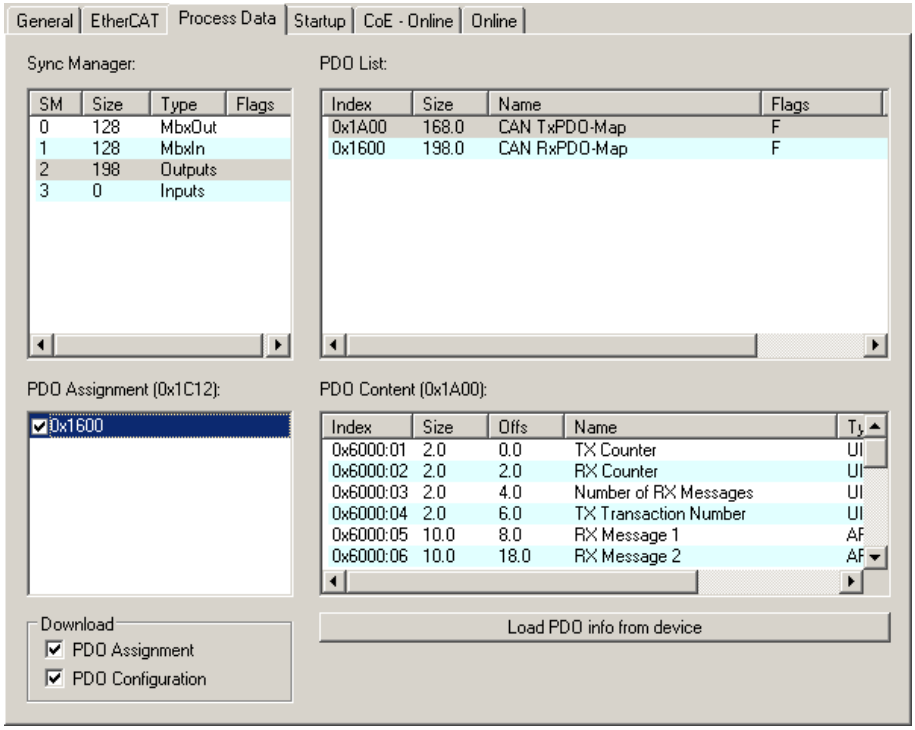


Figure 9: Process data (output PDOs chosen)

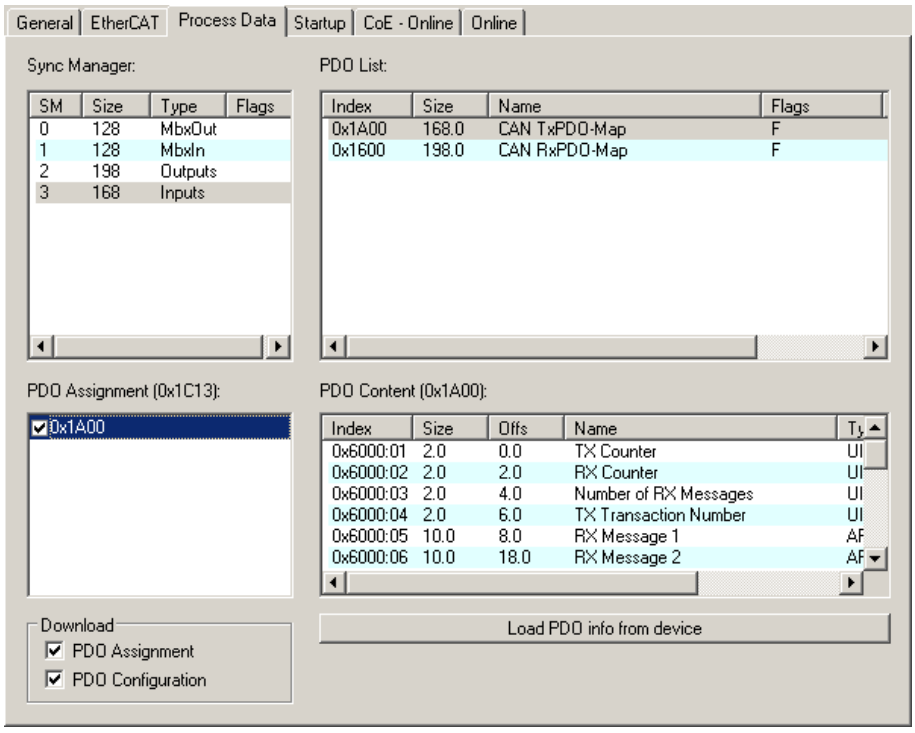


Figure 10: Process data (input PDOs chosen)

The CAN-EtherCAT gateway will only go active on the CAN bus after the baud rate has been set (see chapter „4.5.1.4.7 Object F800h CAN Bus Parameter" from page 39). Consequently, it makes sense to set the baud rate right now.

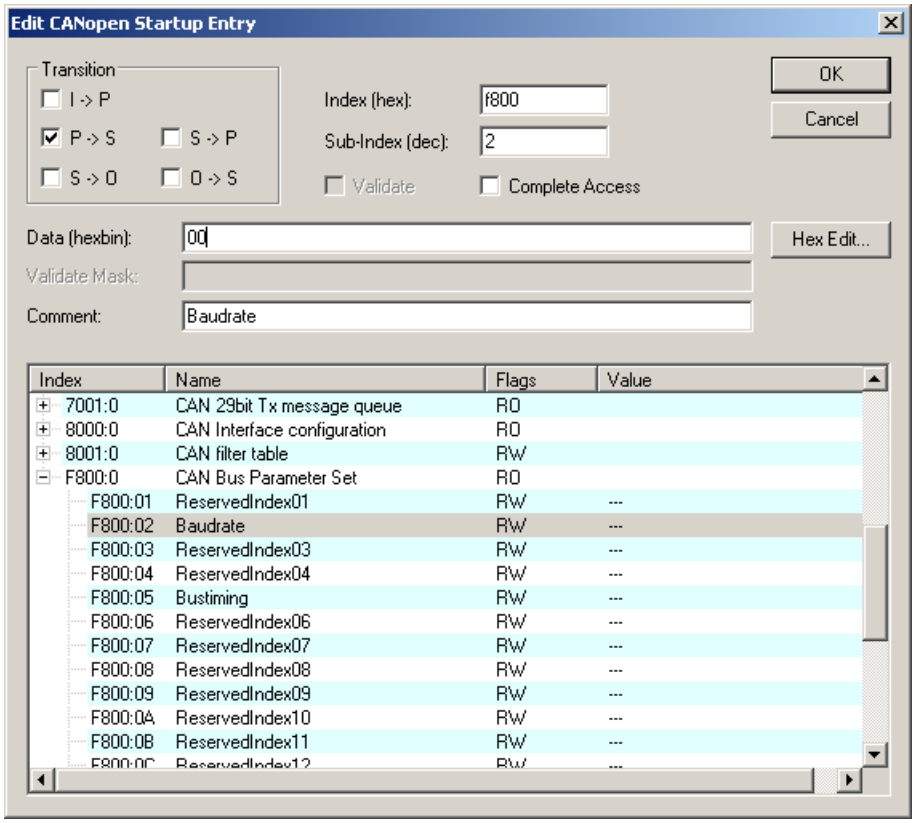


Figure 11: Baud rate setting during startup sequence

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

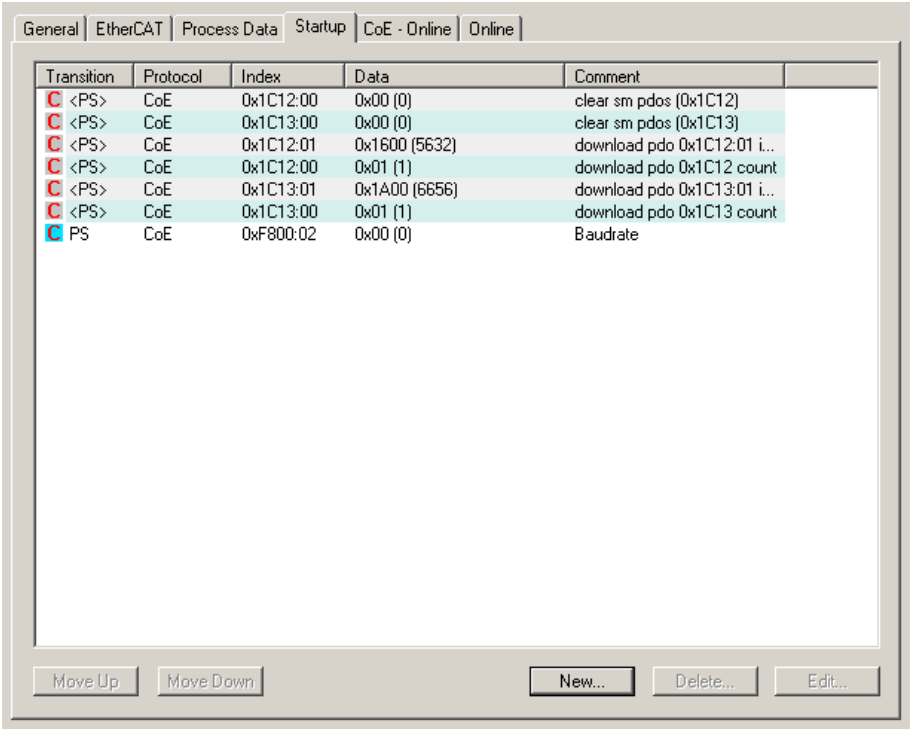


Figure 12: Startup - sequence example

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

Figure 13: Reload Devices

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

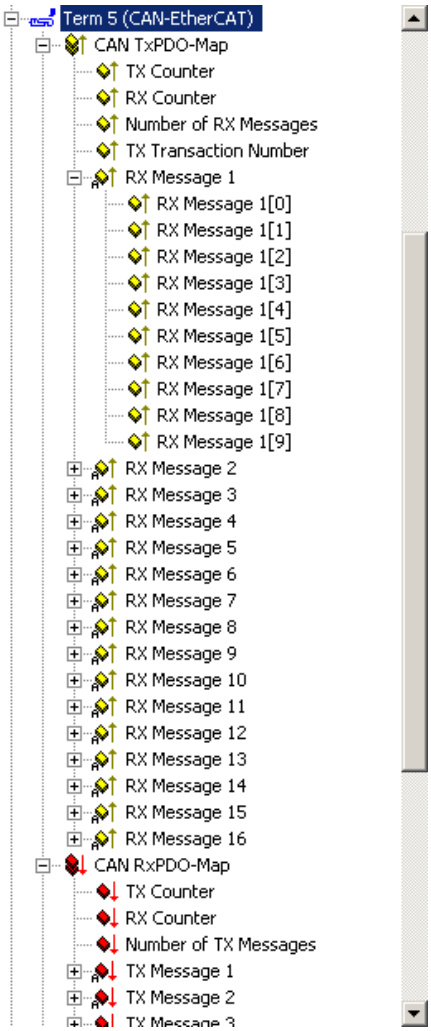


Figure 14: Can interface process image

4.3 Message Structure

4.3.1 Message Structure with 29-Bit Support

- Length (0..8)
  - CobId
    - Bit 0-28: 29-bit identifier
    - Bit 30: RTR
    - Bit 31: 0: Normal message (11-bit identifier), 1: Extended message (29-bit identifier)
- Data[8]

4.3.2 Message Structure without 29-Bit Support

- CobId
  - Bit 0-3: Length (0..8)
  - Bit 4: RTR
  - Bit 5-15: 11-bit identifier
- Data[8]

4.4 Exporting EtherCAT Network Information (ENI)

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

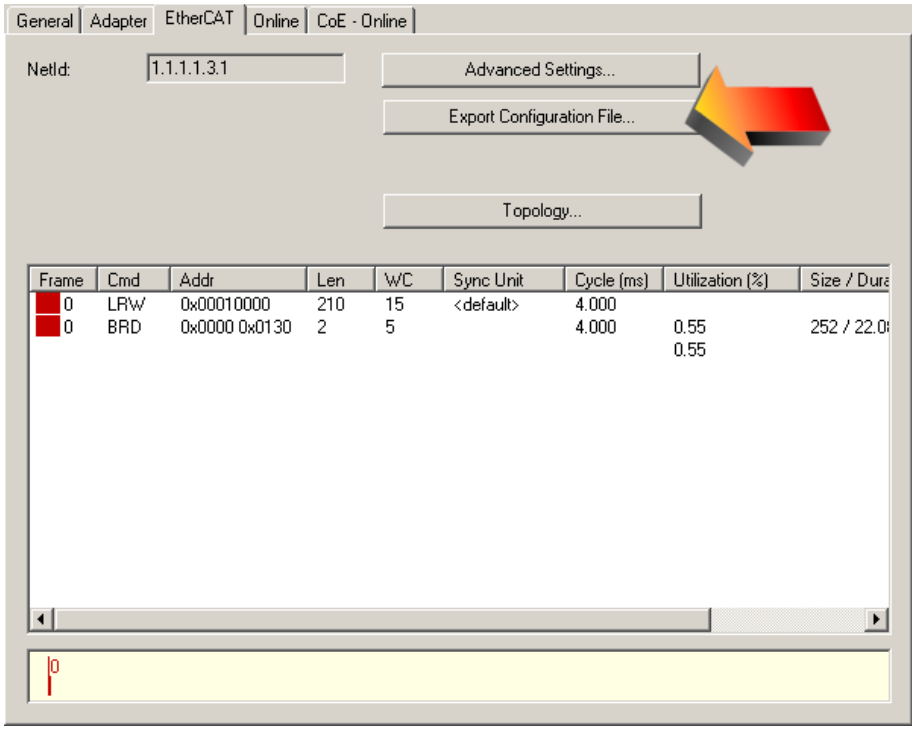


Figure 15: Exporting configuration file

4.5 EtherCAT Communication

4.5.1 CAN Interface

The CAN Interface is based on a modular device profile (Fieldbus Gateway), 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.

4.5.1.1 Object Dictionary Structure

The object dictionary is composed of the following areas:

Index [Hex]	Object Dictionary Areas
0000...0FFF	Data Type Area
1000...1FFF	Communication Area
6000...6FFF	Input Area (CAN Rx message queue)
7000...7FFF	Output Area (CAN Tx message queue)
8000...8FFF	Configuration Area (CAN interface configuration)
F000...FFFF	Device Area

Table 4: 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 RxPDO and TxPDO mapping objects (1600_h and $1A00_h$) will change accordingly.

4.5.1.1.1 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.

4.5.1.1.2 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.

4.5.1.2 Object Dictionary

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

Index [Hex]	Name
1000	Device type
1008	Device name
1009	Hardware version
100A	Software version
1018	Identity
1600	**RxPDO-Map CAN interface**
1A00	**TxPDO-Map CAN interface**
1C00	Sync manager type
1C12	**RxPDO assign**
1C13	**TxPDO assign**
1C32	SM output parameter
1C33	SM input parameter
6000	**CAN interface input (11-bit identifier)**
6001	**CAN interface input (29-bit identifier)**
7000	**CAN interface output (11-bit identifier)**
7001	**CAN interface output (29-bit identifier)**
8000	**CAN interface configuration**
8001	**CAN filter table**
F800	CAN bus parameter

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

4.5.1.3.1 Object 1000_h Device Type

Index [Hex]	Sub-Index	Description	Data Type	RW	Default
1000	0	*Device Type*	UINT32	RO	00001389_h (5001_d)

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.

4.5.1.3.2 Object 1008_h Device Name

Index [Hex]	Sub-Index	Description	Data Type	RW	Default
1008	0	*Device Name*	STRING	RO	„MEESC“

Variable Description

EtherCAT Slave device name

4.5.1.3.3 Object 1009_h Hardware Version

Index [Hex]	Sub-Index	Description	Data Type	RW	Default
1009	0	*Hardware Version*	STRING	RO	

Variable Description

CAN-EtherCAT gateway hardware version

4.5.1.3.4 Object $100A_h$ Software Version

Index [Hex]	Sub-Index	Description	Data Type	RW	Default
100A	0	*Software Version*	STRING	RO	

Variable Description

CAN-EtherCAT gateway software version

4.5.1.3.5 Object 1018_h Identity

Index [Hex]	Sub-Index	Description	Data Type	RW	Default
1018	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	always 0

Variable Description

CAN-EtherCAT gateway identification characteristics.

Vendor ID esd vendor-ID = 23_d

Product code CAN-EtherCAT product code = 2

Revision EtherCAT Slaves revision number
Example: 0100h corresponds to revision = V 1.0

Serial number always 0

4.5.1.3.6 Object 1C00$_h$ Sync Manager Type

Index [Hex]	Sub-Index	Description	Data Type	RW	Default
1C00	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)

4.5.1.3.7 Object 1C32$_h$ Sync-Manager Output Parameters

Index [Hex]	Sub-Index	Description	Data Type	RW	Default
1C32	0	*Number of sub-indexes*	UINT8	RO	32_d
	1	*Sync mode*	UINT16	RW	1
	2	*Cycle time*	UINT32	RW	0
	3	*Shift time*	UINT32	RO	0
	4	*Sync modes supported*	UINT16	RO	4003_h
	5	*Minimum cycle time*	UINT32	RO	0
	6	*Calc and copy time*	UINT32	RO	0
	7	Reserved			
	8	*Command*	UINT16	RW	0
	9	*Delay time*	UINT32	RO	0
	A	Reserved			
	B	*SM event missed counter*	UINT16	RO	0
	C	*Cycle exceeded counter*	UINT16	RO	0
	D	*Shift too short counter*	UINT16	RO	0
	E...1F	Reserved			
	20	*Sync error*	BOOLEAN	RO	0

Parameter Description

This object sets the outputs of the synchronization parameters.

Sync mode Current synchronization mode

Sync mode = 1: Synchronous with SM 2 Event

Cycle time EtherCAT Master cycle time [ns]:

Shift time Not used

Sync modes supported Supported synchronisation modes:
Bit 0 = 1 „Free Run“ supported
Bit 1 = 1 „Synchronous with SM 2 Event“ supported
Bit 2-3 = 00: DC-Mode currently not supported
Bit 14 = 1: Dynamic Times (Measured by writing Object 1C32$_h$:08)

Minimum cycle time Minimum cycle time in [ns]

Calc and copy time Minimum time between SYNC0 and SYNC1 events [ns];
only in DC Mode

Command *Command* = 0: Stops measurement of local cycle time
Command = 1: Starts measurement of local cycle time
Entries 1C32:03, 1C32:05, 1C32:06, 1C32:09, 1C33:03, 1C33:06, C33:09 are updated with maximum measured data.
Data will be reset with new measurement cycle.

Delay time	Time between SYNC1 event and writing of outputs [ns] (only in DC mode)
SM event missed counter	Number of missed SM events in OPERATIONAL (only in DC mode)
Cycle exceeded counter	Number of cycle violations in OPERATIONAL (Cycle did not terminate in time, or next cycle occurred too early)
Shift too short counter	Number of too-short cycles between SYNC0 and SYNC1 events (only in DC mode)
Sync error	Synchronization during the last cycle was incorrect (outputs set too late; only in DC mode)

4.5.1.3.8 Object 1C33$_h$ Sync-Manager Input Parameters

Index [Hex]	Sub-Index	Description	Data Type	RW	Default
1C33	0	*Number of sub-indexes*	UINT8	RO	32_d
	1	*Sync mode*	UINT16	RW	34_h
	2	*Cycle time*	UINT32	RW	0
	3	*Shift time*	UINT32	RO	0
	4	*Sync modes supported*	UINT16	RO	4003_h
	5	*Minimum cycle time*	UINT32	RO	0
	6	*Calc and copy time*	UINT32	RO	0
	7	Reserved			
	8	*Command*	UINT16	RW	0
	9	*Delay time*	UINT32	RO	0
	A	Reserved			
	B	*SM event missed counter*	UINT16	RO	0
	C	*Cycle exceeded counter*	UINT16	RO	0
	D	*Shift too short counter*	UINT16	RO	0
	E...1F	Reserved			
	20	*Sync error*	BOOLEAN	RO	0

Parameter Description

This object sets the inputs of the synchronization parameters.

Sync mode Current Synchronization Mode

Sync mode = 34: Synchronous with SM 2 Event
(Outputs present)

Cycle time EtherCAT Master Cycle Time in [ns]:

Shift time Time between SYNC0 event and reading of inputs ([ns], only in DC Mode)

Sync modes supported Support Synchronization Modes:
Bit 0 = 1 „Free Run“ supported
Bit 1 = 1 “Synchronous with SM 2 Event” supported
(outputs are present)
Bit 14 = 1: Dynamic Times
(measured by writing object 1C32$_h$:08 or 1C33$_h$:08)

Minimum cycle time Minimum cycle time in [ns]

Calc and copy time Time between reading the inputs and availability of inputs to the master ([ns]; only in DC Mode)

Command	*Command* = 0: Stops measurement of local cycle time *Command* = 1: Starts measurement of local cycle time Entries 1C32:03, 1C32:05, 1C32:06, 1C32:09, 1C33:03, 1C33:06, 1C33:09 are updated with maximum measured data. Data will be reset with new measurement cycle.
Delay time	Not supported.
SM event missed counter	Number of missed SM events in OPERATIONAL (only in DC mode).
Cycle exceeded counter	Number of cycle violations in OPERATIONAL (Cycle did not terminate in time, or next cycle occurred too early)
Shift too short counter	Number of too-short cycles between SYNC0 and SYNC1 events (only in DC mode).
Sync error	Synchronization during the last cycle was incorrect (outputs set too late; only in DC mode)

4.5.1.3.9 Object 1600_h RxPDO-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 [Hex]	Sub-Index	Description	Data Type	RW	Default
1600	0	*Number of CAN-Messages+3*	UINT8	RO	
	1	*1. PDO Mapping entry* (object 0x700z (CAN interface output), entry 0x01 (Tx Counter))	UINT32	RO	
	2	*2. PDO Mapping entry* (object 0x700z (CAN interface output), entry 0x02 (Rx Counter))	UINT32	RO	
	3	*3. PDO Mapping entry* (object 0x700z (CAN interface output), entry 0x03 (Number of Tx Messages))	UINT32	RO	
	4	*4. PDO Mapping entry* (object 0x700z (CAN interface output), entry 0x04 (Tx Message 1))	UINT32	RO	
	...	..	...	...	
	m	*m. PDO Mapping entry* (object 0x700z (CAN interface output), entry m (Tx Message m-3))	UINT32	RO	

4.5.1.3.10 Object 1A00$_h$ TxPDO-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 [Hex]	Sub-Index	Description	Data Type	RW	Default
1A00	0	Number of *CAN-Messages+4*	UINT8	RO	
	1	*1. PDO Mapping entry* (object 0x6000 (CAN interface input), entry 0x01 (Tx Counter))	UINT32	RO	
	2	*2. PDO Mapping entry* (object 0x6000 (CAN interface input), entry 0x02 (Rx Counter))	UINT32	RO	
	3	*3. PDO Mapping entry* (object 0x6000 (CAN interface input), entry 0x03 (Number of Rx Messages))	UINT32	RO	
	4	*4. PDO Mapping entry* (object 0x6000 (CAN interface input), entry 0x04 (Tx Transaction Number))	UINT32	RO	
	5	*5. PDO Mapping entry* (object 0x6000 (CAN interface input), entry 0x05 (Rx Message 1))	UINT32	RO	
	...	..		...	
	m	*m. PDO Mapping entry* (object 0x6000 (CAN interface input), entry m (Rx Message m-4))	UINT32	RO	

4.5.1.3.11 Object 1C12$_h$ RxPDO-Assign

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

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

4.5.1.3.12 Object 1C13$_h$ TxPDO-Assign

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

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

4.5.1.4 Profile Specific Objects (6000_h-$FFFF_h$)

The profile specific objects apply for all EtherCAT Slave devices supporting the 5001 profile.

4.5.1.4.1 Object 6000_h CAN-Rx-Message-Queue

Index [Hex]	Sub-Index	Name	Data Type	RW	Default
6000	0	*Number of sub-indexes*	UINT8	RO	
	1	*Tx Counter*	UINT16	RO	
	2	*Rx Counter*	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 — The Tx counter is increased by the CAN interface to indicate that the CAN Tx messages were copied from the output data to the CAN send queue.

Rx Counter — The Rx counter is increased by the CAN interface every time when new CAN Rx data arrived and the Rx counter (6000h:02)is identical with Rx counter (7000h:02). This indicates that new Rx data has been written into the process input data.

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

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

4.5.1.4.2 Object 6001$_h$ CAN-Rx-Extended-Message-Queue

Index [Hex]	Sub-Index	Name	Data Type	RW	Default
6001	0	*Number of sub-indexes*	UINT8	RO	
	1	*Tx Counter*	UINT16	RO	
	2	*Rx Counter*	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.

Parameter Description

Tx Counter The Tx counter is increased by the CAN interface to indicate that the CAN Tx messages were copied from the output data to the CAN send queue.

Rx Counter The Rx counter is increased by the CAN interface every time when new CAN Rx data arrived and the Rx counter (6001h:02) is identical with Rx counter (7001h:02). This indicates that new Rx data has been written into the process input data.

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.

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

4.5.1.4.3 Object 7000$_h$ CAN-Tx-Message-Queue

Index [Hex]	Sub-Index	Name	Data Type	RW	Default
7000	0	*Number of sub-indexes*	UINT8	RO	
	1	*Tx Counter*	UINT16	RW	
	2	*Rx Counter*	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.

Parameter Description

Tx Counter This counter must be increased when or after writing the CAN Tx message to the output data.

Rx Counter 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.

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 16-19: 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

4.5.1.4.4 Object 7001$_h$ CAN-Tx-Extended-Message-Queue

Index [Hex]	Sub-Index	Name	Data Type	RW	Default
7001	0	*Number of sub-indexes*	UINT8	RO	
	1	*Tx Counter*	UINT16	RW	
	2	*Rx Counter*	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.

Parameter Description

Tx Counter This counter must be increased when or after writing the CAN Tx message to the output data.

Rx Counter 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.

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

4.5.1.4.5 Object 8000_h CAN-Interface-Configuration

Index [Hex]	Sub-Index [Hex]	Name	Data Type	RW	Default
8000	0	*Number of sub-indexes*	UINT8	RO	24_h (36_d)
	1	*Node Address*	UINT16	RW	0000_h
	2...19	Reserved for future extensions	-	-	-
	20	*Flags*	UINT16	RW	0000_h
	21	*Rx queue size*	UINT8	RW	10_h (16_d)
	22	*Tx queue size*	UINT8	RW	10_h (16_d)
	23	*Local Rx queue size*	UINT16	RW	$03E8_h$ (1000_d)
	24	*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; currently not implemented

Local Tx queue size Tx queue size of the internal CAN driver; currently not implemented

4.5.1.4.6 Object 8001_h CAN-Rx-Filter-Table

Index [Hex]	Sub-Index	Name	Data Type	RW	Default
8001	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.

Note:
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. 64_h (100_d)

4.5.1.4.7 Object F800$_h$ CAN Bus Parameter

Index [Hex]	Sub-Index	Name	Data Type	RW	Default
F800	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 5: 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 6: 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 8
CAN_BR	x	*CAN baud rate register of ARM9 AT91SAM9263*

Table 7: 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 [hex]	Bit rate [kBit/s]	Constant *1)
0	1000	NTCAN_BAUD_1000
E	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	83.3	-
8	66.6	-
9	50	NTCAN_BAUD_50
A	33.3	not allowed
B	20	not allowed
C	12.5	not allowed
D	10	not allowed

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

Table 8: Pre-defined bit rate table

Constants and special features

Constant	Value [hex]	Function
NTCAN_BAUD_1000	0	Sets baud rate to 1000 kBit/s
NTCAN_BAUD_800	E	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	Gateway cannot receive or transmit any message; stays passive on CAN bus
NTCAN_AUTOBAUD	00FF FFFE	Gateway checks baud rates until it detected the correct one
NTCAN_USER_BAUDRATE	8000 0000	sets the *UBR* bit
NTCAN_USER_BAUDRATE_NUM	2000 0000	*Sets the UBRN bit*
NTCAN_LISTEN_ONLY_MODE	4000 0000	*Sets the LOM bit*

Table 9: 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 52)
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.

Note:
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. Ethernet Interface

5.1 Overview

Note:
The CAN-EtherCAT gateway will be configured through EtherCAT.
The Ethernet interface is only used for diagnosis and firmware updates.

The setup of the Ethernet interface requires the following steps:

1. Configuration of IP address (provided it is not known/cannot be retrieved)
2. Configuration of all other parameters by means of a web browser (see page 47 ff.)

For this purpose the Ethernet interface must be connected to either an Ethernet switch or hub (twisted pair cable) or to a computer handling the configuration (cross twisted pair cable). Make sure the yellow 'Link' LED is on, indicating a working connection.

5.2 IP Address Configuration

In a first step the CAN-EtherCAT needs a valid IP address. An IP address is a unique device address in a TCP/IP network. It is mandatory that each device in the network is assigned a unique address.

Initial IP address configuration:
After power-up a new CAN-EtherCAT gateway waits for max. 2 minutes for a DHCP server to assign an IP address.

If no IP address was assigned the CAN-EtherCAT will set an auto IP address in the range of 169.254.X.X.

In order to change the IP address it is recommended to use the esdcp tool (see page 45). In order for esdcp to find the device, the computer running the esdcp tool must be connected to the same subnet.

If esdcp finds the device, IP address and net mask should be set as required for the later use.

The configuration per web browser can be accomplished following a CAN-EtherCAT reboot (see page 47 ff.).

5.2.1 Configuration via DHCP

For a configuration per DHCP the DHCP server must be connected to the **same subnet** as the CAN-EtherCAT gateway. In some cases, the DHCP server must be specifically configured as well. Please contact your system administrator for more information. The server will assign a valid IP address to the CAN-EtherCAT gateway. In addition it will assign a network mask, gateway address, and the IP address of the name server. The CAN-EtherCAT gateway will use these parameters immediately (without restart).

In some cases it might be necessary to determine the IP address (**<IP Address>**, as assigned to the device) through the help of the DHCP server's logging mechanism. Any following configuration of the network parameters can now be accomplished through a regular web browser, which must be connected to the same subnet at the URL **http://<IP Address>** - See chapter "5.3 Web based Configuration" for more information.

Attention!
Without additional configuration the DHCP server may assign a different IP address after each restart and only for a certain time. Please contact your system administrator in case the CAN-EtherCAT gateway requires a static IP address.

5.2.2 Configuration via esdcp

esdcp is a software tool that locates esd Ethernet interfaces in a LAN and configures these devices. It utilizes a special protocol using UDP.

In order to be able to find devices with Ethernet interface the computer needs to be connected to the same subnet as these devices.

After program start hit the Discover button to search for esd devices. All devices found in the network will be displayed in the *Device List*.

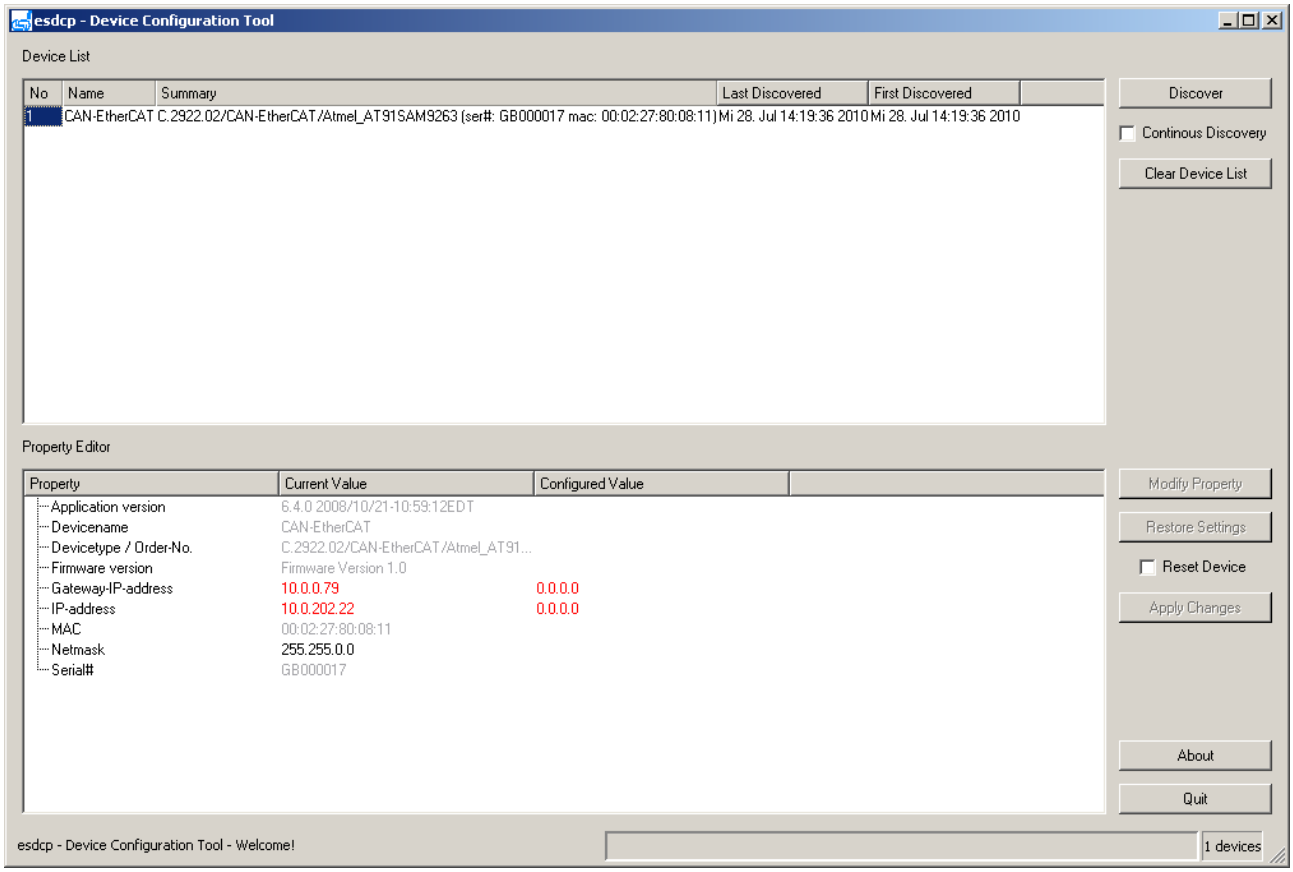


Figure 16: esdcp search result

Checking the *Continuous Discovery* option initiates the continuous search for esd devices. Devices that have been removed from the network will not automatically removed from the device list. Click on the Clear Device List command button to clear the list.

In case no DHCP was found and the CAN-EtherCAT gateway assigned an IP address out of the 169.255.x.x range, the computer must use be set to use the same address (only once to configure the CAN-EtherCAT gateway).

As soon as the esdcd tool has detected the CAN-EtherCAT gateway it is possible to set the IP address and subnet mask.

In the *Property Editor* window double-click on the *IP-address* row to change the IP address or double-click the row *Netmask* to change the subnet mask.
Instead of double-clicking you can also use the Modify Property command button.

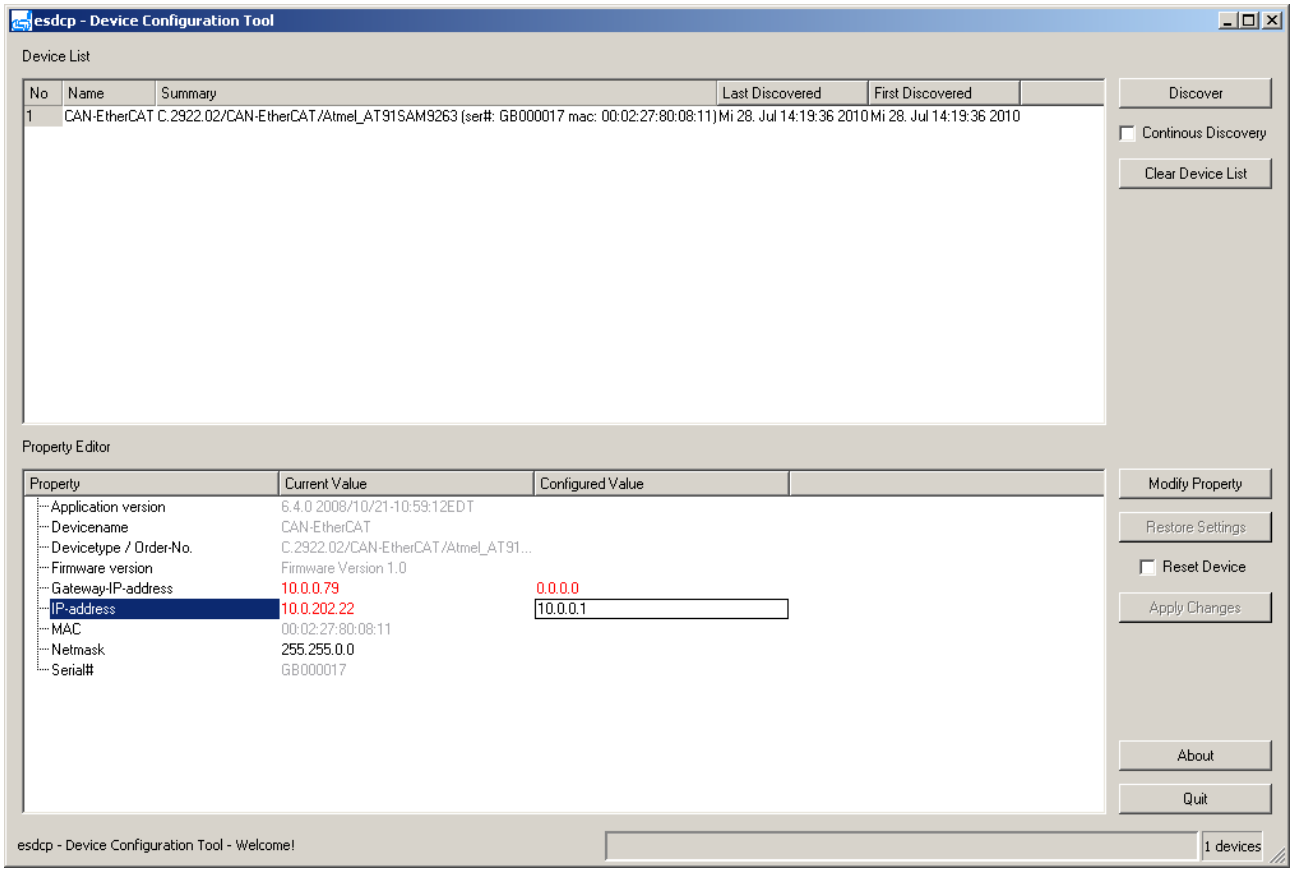


Figure 17: Setting the IP address

Enter the desired IP address in the *Configured Value* edit box. The same procedure applies to entering the subnet mask.

Check *Reset Device* to restart the CAN-EtherCAT gateway. Any changes of the IP address or the subnet mask will only take effect after a restart.
Confirm the new parameters by clicking the Apply Changes command button.

The value entered in *Configured Value* will be accepted after a request of the password. The new value is displayed under *Current Value*.

Clicking the Restore Settings command button allows the clearing of the values entered through *Configured Value*. Values entered through *Current Value* remain unchanged.

The esdcp standard password is: esd

Note:
To turn on the DHCP through the esdcp tool configure the IP address 0.0.0.0.

5.3 Firmware Update and CAN Status via Web Interface

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.

5.3.1 Overview

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

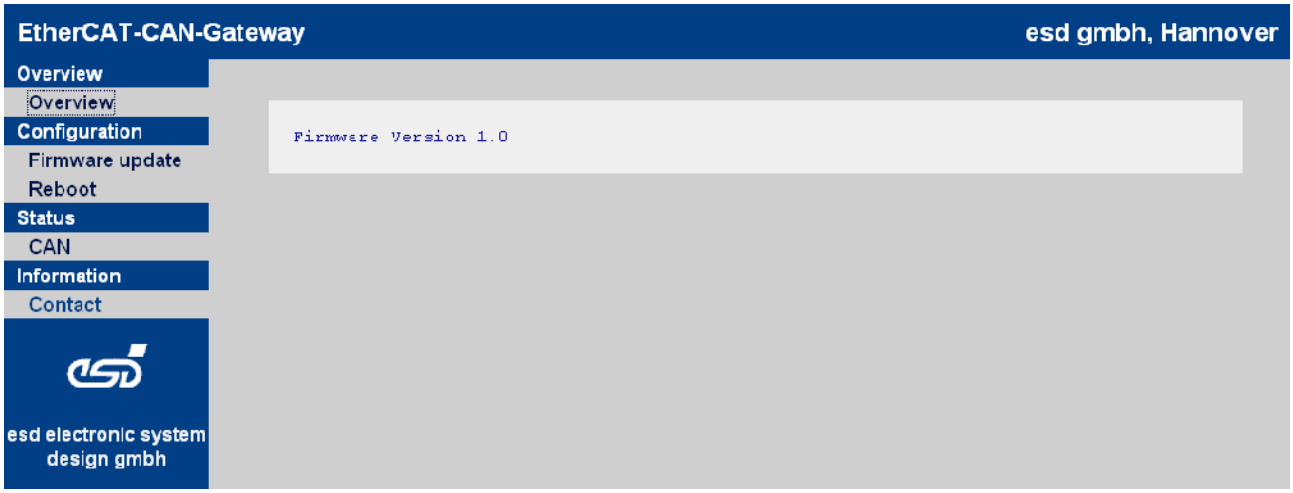


Figure 18: Overview

5.3.1.1 Firmware Update

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

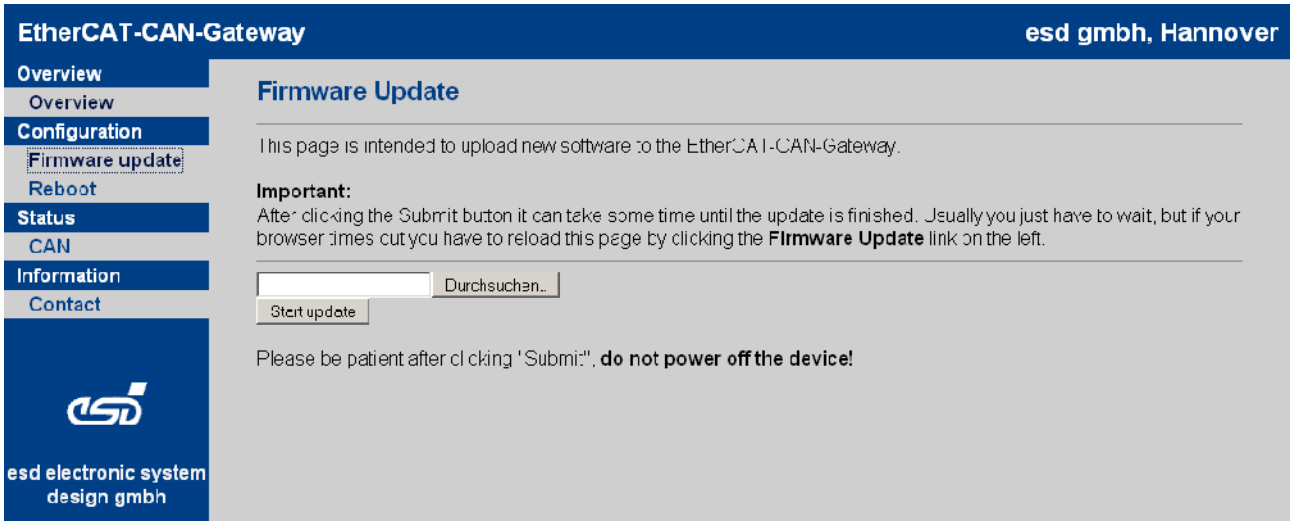


Figure 19: Firmware update

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.

Attention!
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:

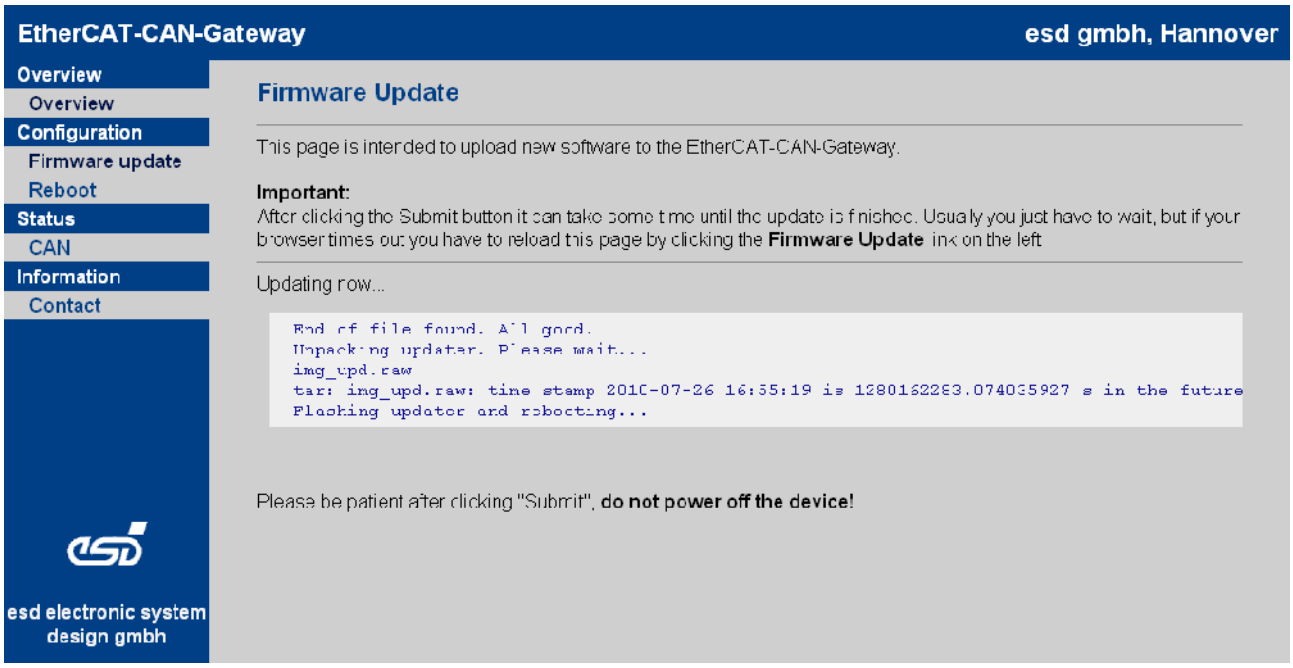


Figure 20: Firmware update output

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

When finished the system will initiate an automatic restart.

5.3.1.2 Reboot

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

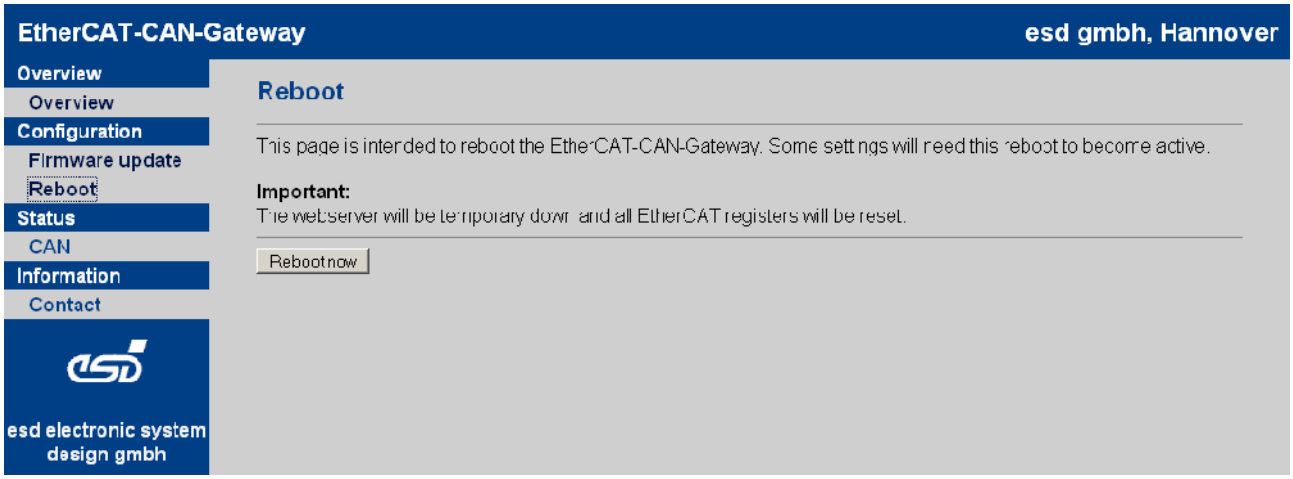


Figure 21: Reboot

5.3.2 Status

5.3.2.1 CAN Statistics

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

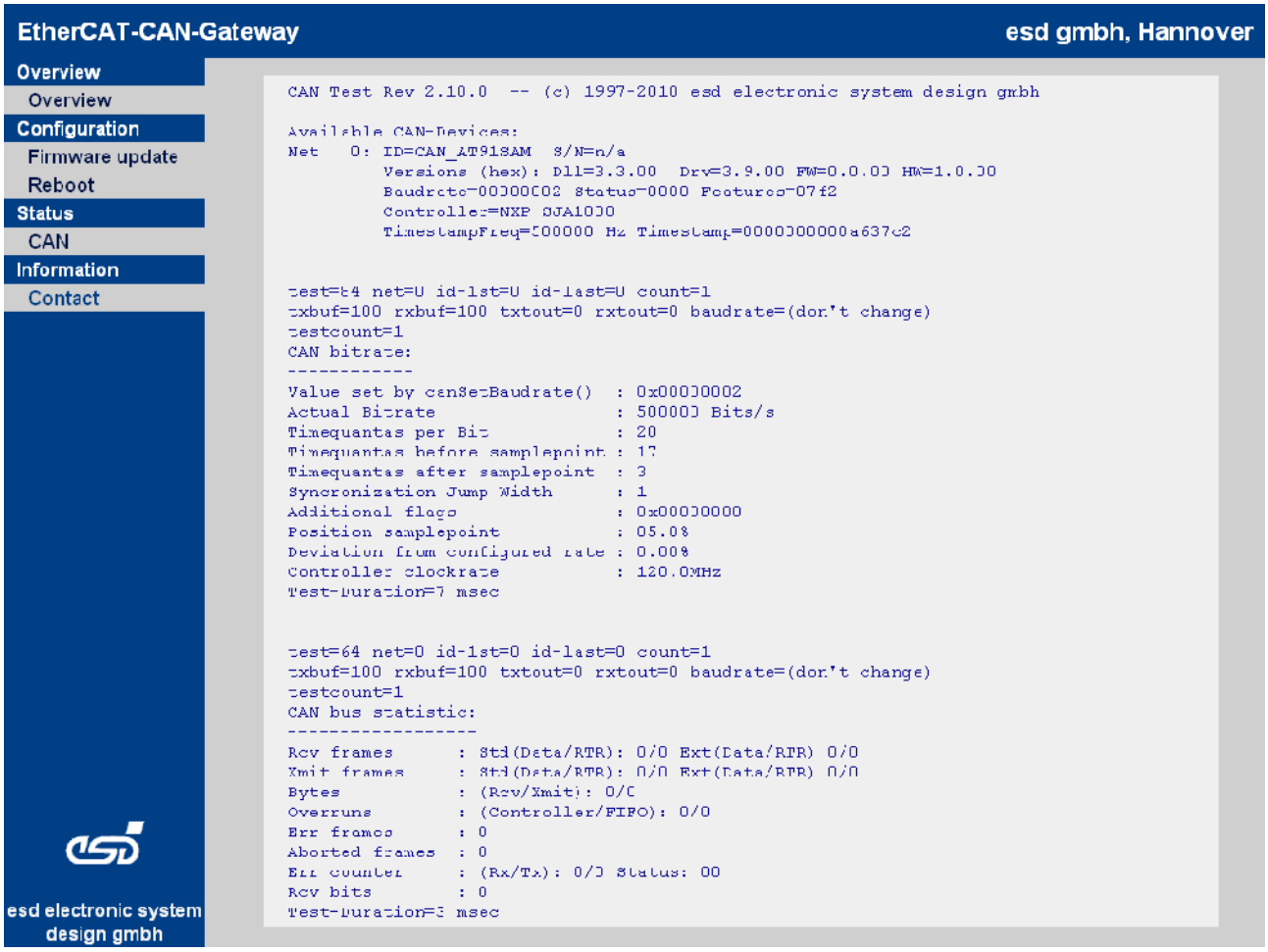


Figure 22: CAN status output

6. Technical Data

6.1 General Technical Data

Power supply voltage	Nominal voltage: typical: 24 V/DC, (min.: 18 V, max.: 32 V) Current consumption: (24 V, 20 °C): tbd.
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 10: General data of the module

6.2 Microprocessor and Memory

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

Table 11: Microprocessor and memory

6.3 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 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 12: Data of the CAN interface

6.4 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 13: Data of the EtherCAT interface

6.5 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 14: Data of the Ethernet interface

6.6 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 15: Data of the USB interface

6.7 Operating System and License Information

Operating system	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.

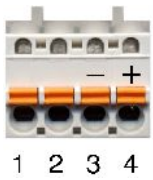
7. Interfaces and Connector Assignments

7.1 24 V-Power Supply Voltage

The power supply voltage can be fed via connector X1 or optional via InRailBus (connector assignment see page 69)

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 8.

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

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

Signal Description:

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

7.2 CAN

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

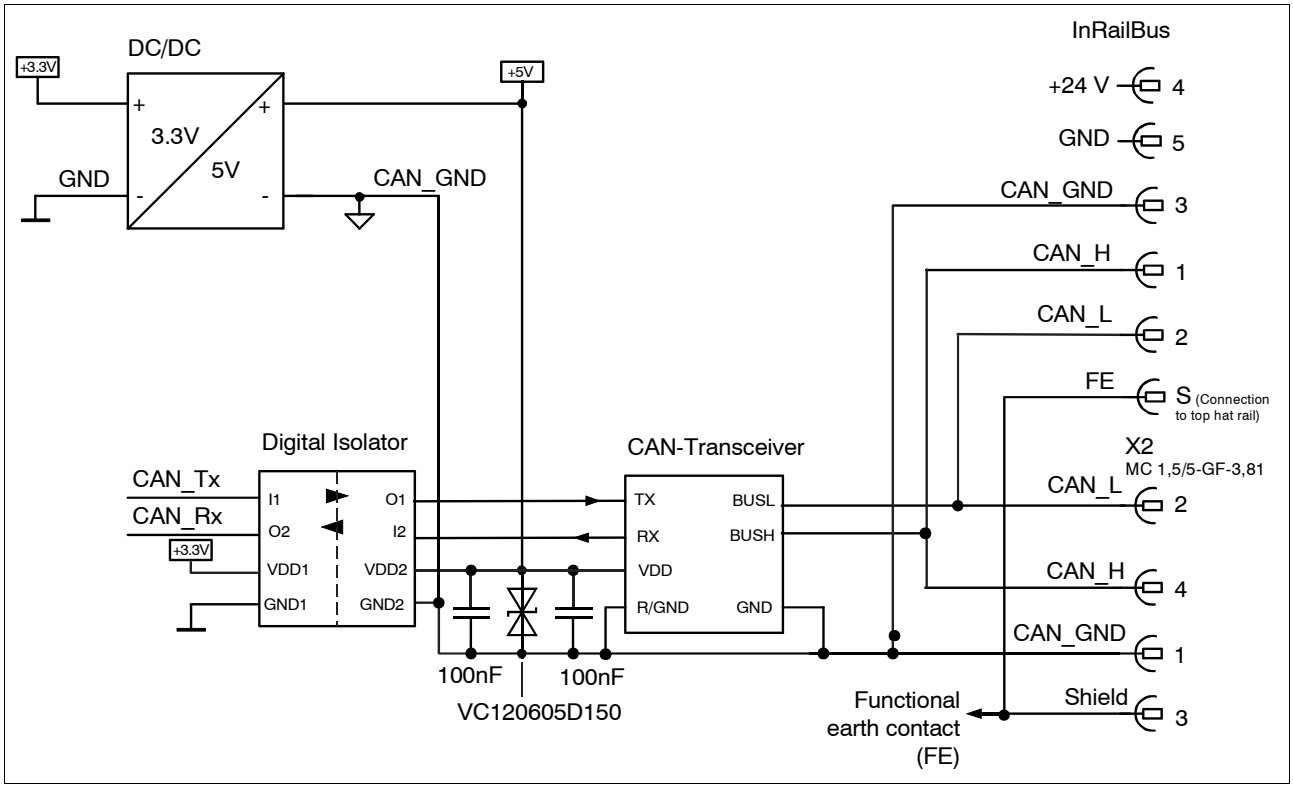


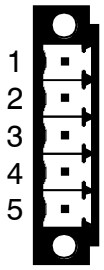
Fig. 23: CAN-Interface

The CAN interface can be connected via CAN connector (X2) or optionally via InRailBus (connector assignment see page 69).

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)



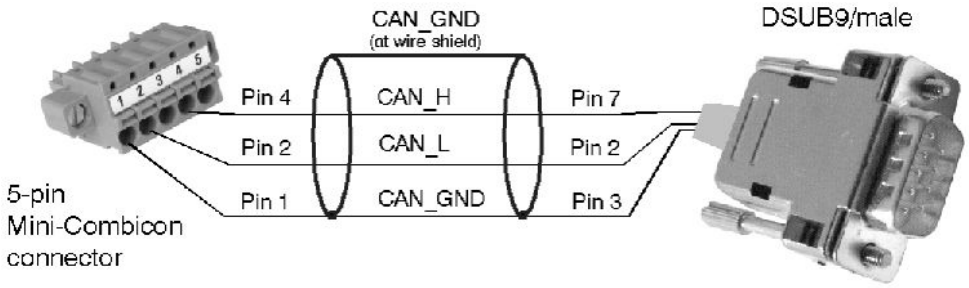
Pin Assignment:

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

7.3 24 V and CAN via InRailBus

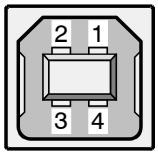
Power supply voltage and CAN can optionally be fed via InRailBus.
Use the mounting-rail bus connector of the CBX-InRailBus for the connection via the InRailBus, see order information (page 75).
Read and follow the instructions for connecting power supply and CAN signals via InRailBus (see page 70)!

7.4 DIAG

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

Note:
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. Correctly Wiring Electrically Isolated CAN Networks

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

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

8.1.1 General Rules

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

1	A cable type with a wave impedance of about 120 Ω ±10% with an adequate wire 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! (Pay attention to the effects that may happen, if multiple earth points are used (ground loops).)
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** at 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.

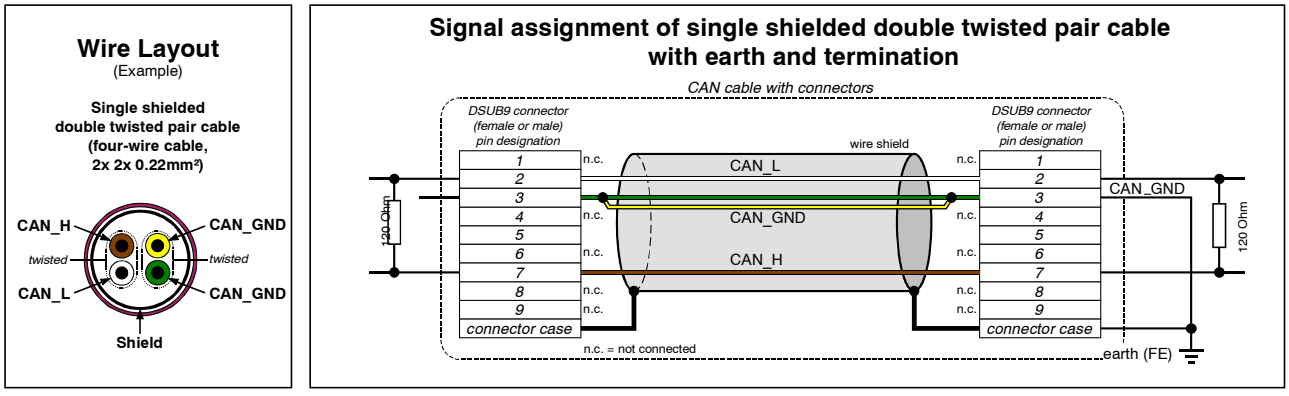


Figure. 24: CAN wiring for heavy industrial environment

8.1.2 Device Cabling

- To connect CAN devices which are equipped with one CAN connector per net, use T-connectors and cable stubs (shorter than 0.3 m).

Attention:
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. from ERNI (ERBIC CAN BUS MAX, order no.:154039).

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

Furthermore, mixed use of single twisted and double twisted cables should be avoided!

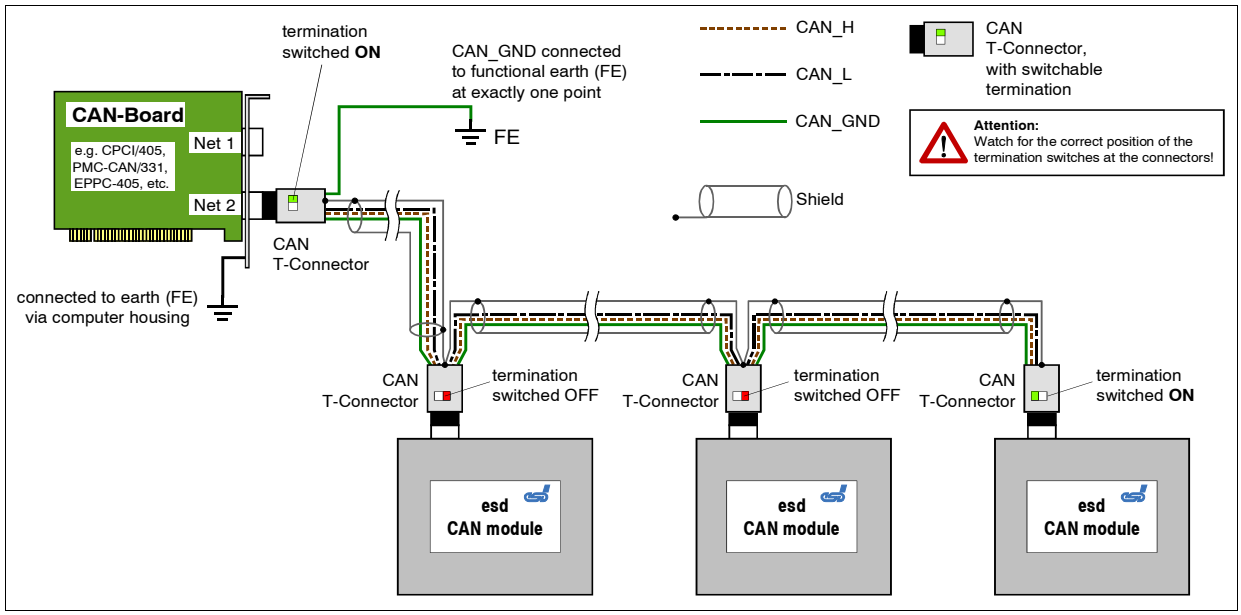


Figure. 25: Example for proper wiring with single shielded double twisted pair cables

8.1.3 Termination

- Use external termination plugs, because they can later be rediscovered more easily than internal terminations within the CAN devices!

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

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

8.2.1 General Rules

Note:
esd grants the EC 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 118982-2 (table 9). (Single shielded *double* twisted pair cable wiring as described in chapter 8.1. ensures the EC Conformity as well.)

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

1	A cable type with a wave impedance of about 120 Ω ±10% with an adequate wire 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 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.

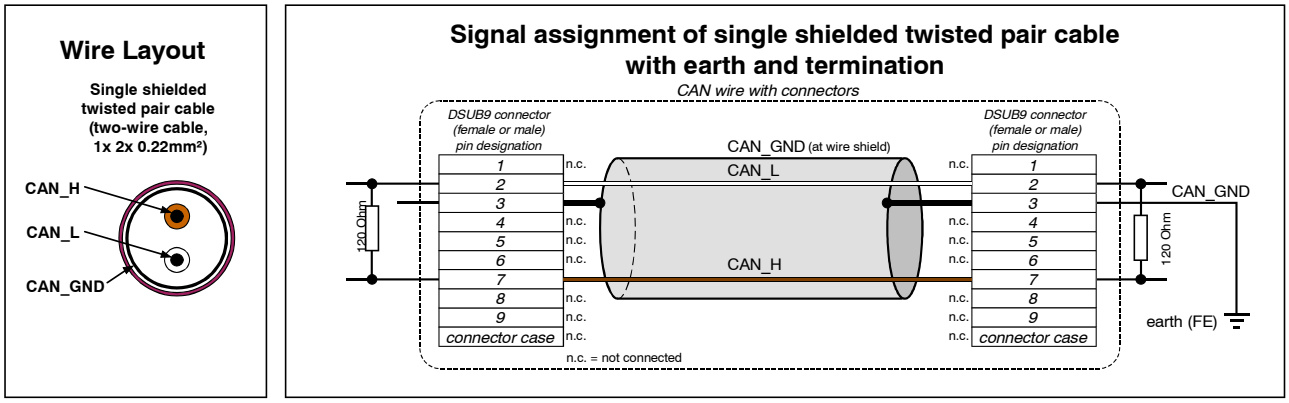


Figure. 26: CAN wiring for light industrial environment

8.2.2 Cabling

- for devices which have only one CAN connector per net use T-connector and stub (shorter than 0.3 m) (available as accessory)

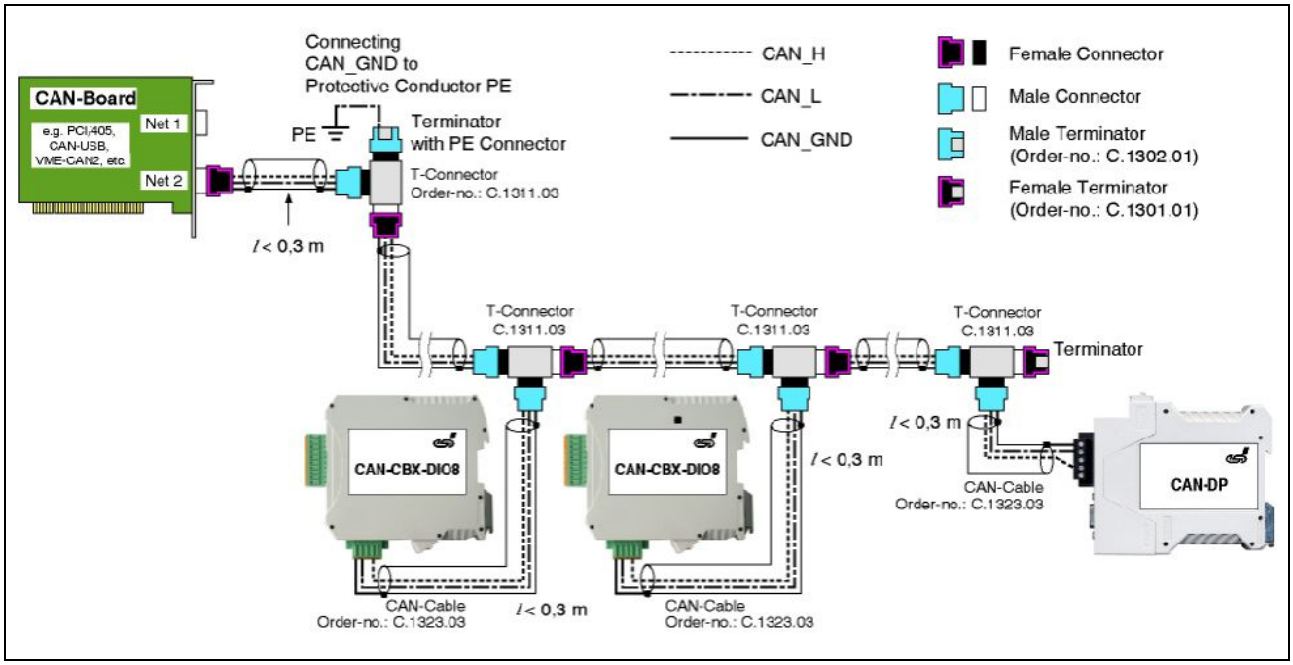


Figure. 27: Example for proper wiring with single shielded single twisted pair wires

8.2.3 Termination

- Use external termination plugs, because they can be rediscovered more easily than internal terminations within the CAN devices!
- 9-pin DSUB-termination connectors with male and female contacts and earth terminal are available as accessories

8.3 Earthing

- CAN_GND has to be connected between the CAN devices, because esd CAN devices are electrically isolated from each other!
- CAN_GND has to be connected to the earth potential (PE) at **exactly one** point of the network!
- Each CAN interface without electrically isolated interface acts as an earthing point. For this reason do not connect more than one CAN device without electrically isolated CAN interface!
- Earthing can e.g. be made at a connector/T-connector.

8.4 Bus Length

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

Bit rate [Kbits/s]	Typical values of reachable wire length **with esd interface** l_{max} [m]	**CiA recommendations** (07/95) for reachable wire lengths l_{min} [m]
1000	37	25
800	59	50
666.6	80	-
500	130	100
333.3	180	-
250	270	250
166	420	-
125	570	500
100	710	650
66.6	1000	-
50	1400	1000
33.3	2000	-
20	3600	2500
12.5	5400	-
10	7300	5000

Table 16: Recommended cable lengths at typical bit rates (with esd-CAN interfaces)

Note:
Please note the recommendations according to ISO 11898 for the selection of the cross section of the wire depending of the wire length.

8.5 Examples for CAN Cables

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

Manufacturer	Cable Type
U.I. LAPP GmbH Schulze-Delitzsch-Straße 25 70565 Stuttgart Germany www.lappkabel.de	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.)

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

Manufacturer	Cable Type
U.I. LAPP GmbH Schulze-Delitzsch-Straße 25 70565 Stuttgart Germany www.lappkabel.de	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.)

Note:
Configured CAN cables can be ordered from **esd**.

9. 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.

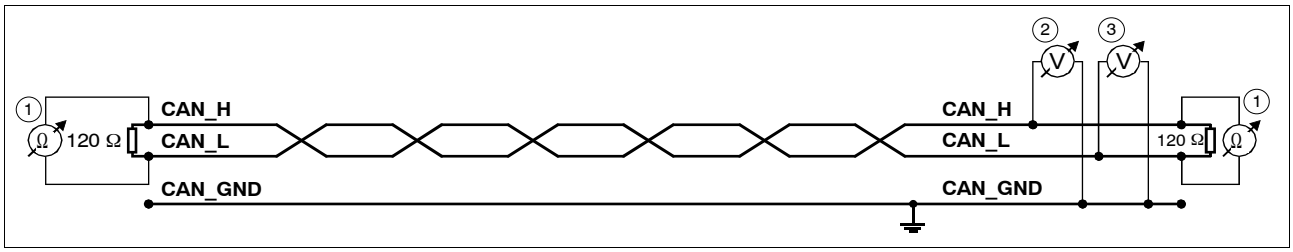


Figure. 28: Simplified diagram of a CAN network

9.1 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 eliminated. 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 the ends of the network ① (see figure above) and at the centre of the network (if the network cable consists of more than one line section).

The measured value should be between 50 Ω and 70 Ω. The measured value should be nearly the same at each point of the network.

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
- 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.

9.2 Ground

The CAN_GND of the CAN network has to be connected to the functional earth potential (FE) at only **one** point. This test will indicate 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. Connect CAN_GND to earth potential.

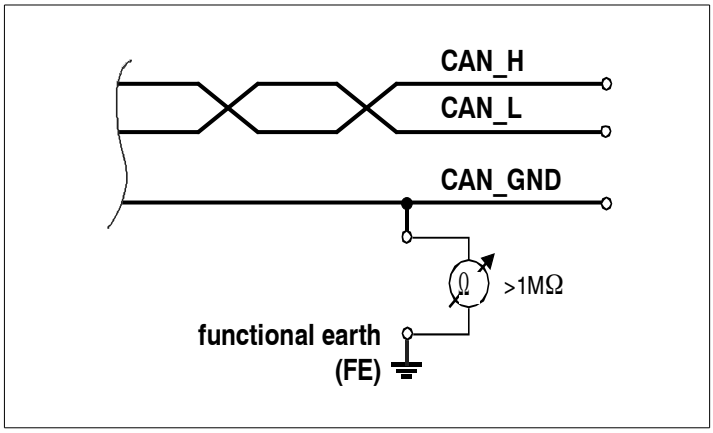


Figure. 29: Simplified schematic diagram of ground test measurement

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

9.3 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 the error rate will increase strongly. Make sure that there is no short circuit between CAN_GND and CAN_L!

9.4 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 volts. 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 GND ② (see figure above).
4. Measure the DC voltage between CAN_L and GND ③ (see figure above).

Normally the voltage should be between 2.0 V and 4.0 V.

If it is lower than 2.0 V or higher than 4.0 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. For a voltage higher than 4.0 V, please check for excessive voltage.

To find the node with a faulty transceiver please test the CAN transceiver resistance (see next page).

9.5 CAN Transceiver Resistance Test

CAN transceivers have one circuit that controls CAN_H and another circuit that controls CAN_L. Experience has shown that electrical damage to one or both of the circuits may increase the leakage current in these circuits.

To measure the current leakage through the CAN circuits, please use an 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 MΩ for each signal. If it is much lower, the CAN transceiver it is probably faulty.
Another sign for a faulty transceiver is a very high deviation between the two measured input resistance (>> 200%).

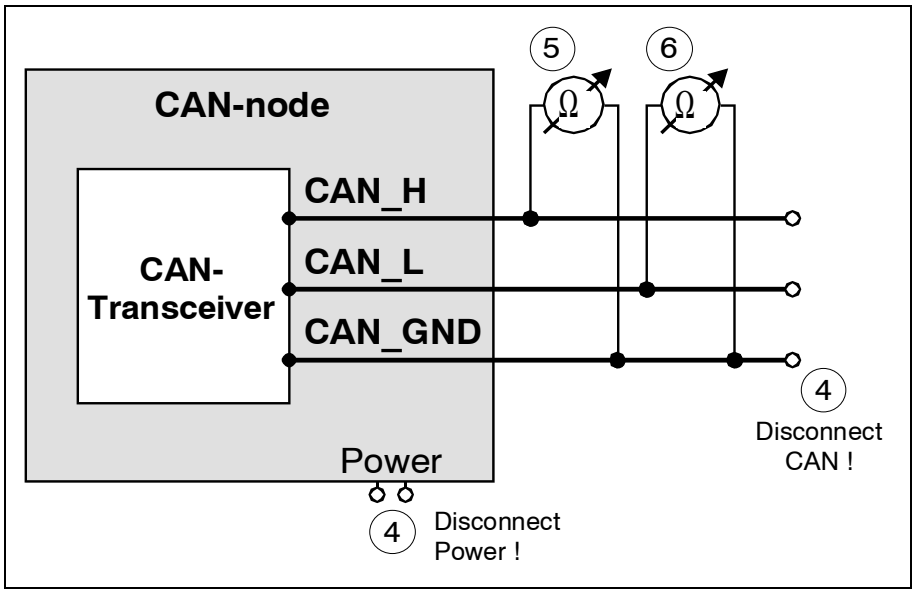


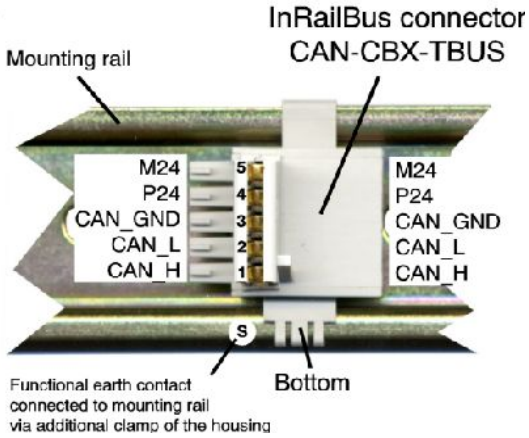
Figure. 30: Measuring the internal resistance of CAN transceivers

10. Option InRailBus

10.1 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)

10.2 Using InRailBus

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

10.2.1 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:

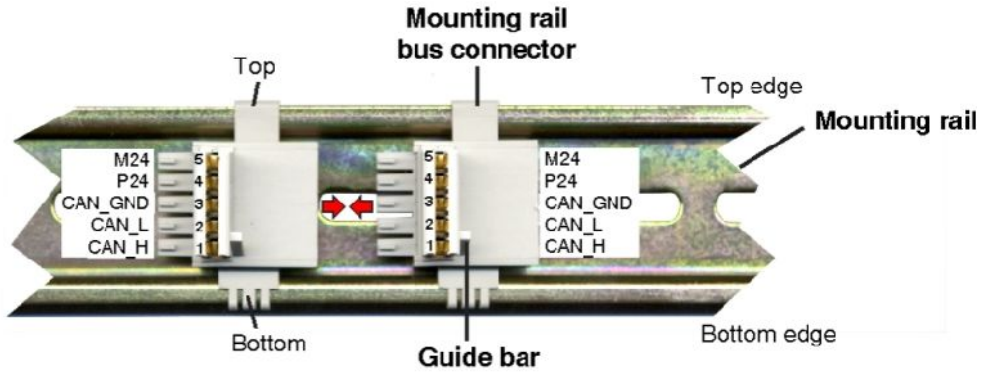


Figure. 31: Mounting rail with bus connector

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. 32: 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 31. 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. 33: Mounted CAN-CBX module

10.2.2 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 75).

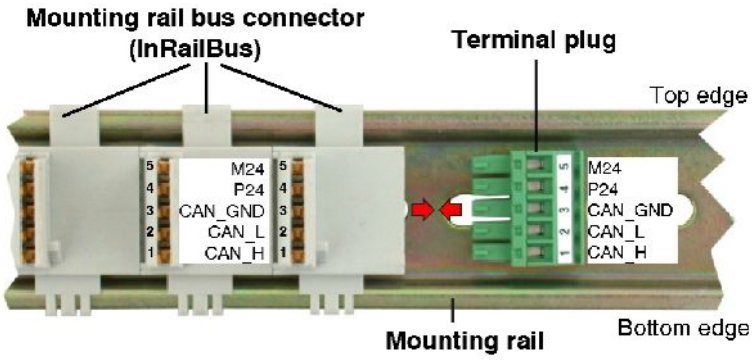


Figure. 34: Mounting rail with InRailBus and terminal plug

Plug the terminal plug into the socket on the right of the mounting-rail bus connector of the InRailBus, as described in Figure 34. Then connect the CAN interface and the power supply voltage via the terminal plug.

10.2.3 Connection of the Power Supply Voltage

Attention!
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.

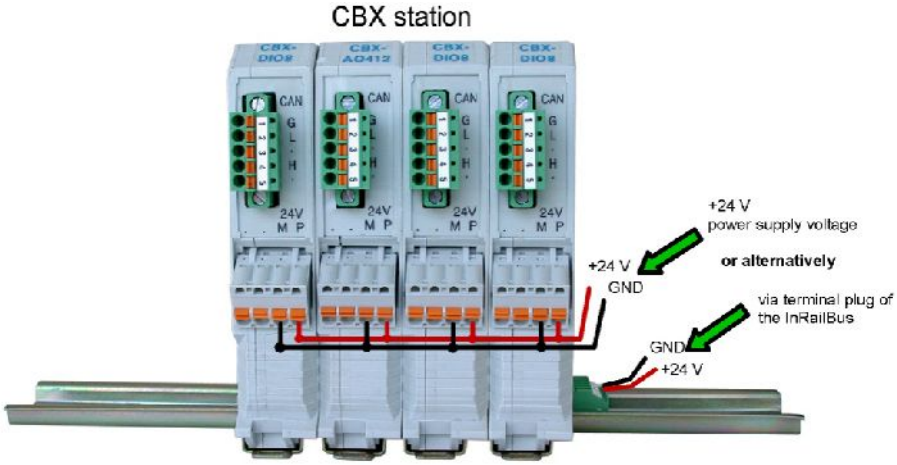


Figure. 35: Connecting the power supply voltage to the CAN-CBX station

10.2.4 Connection of CAN

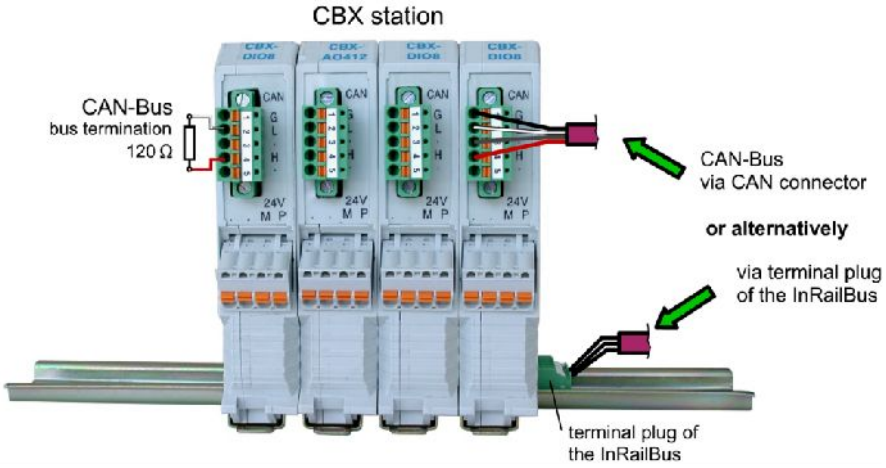


Figure. 36: Connecting the CAN signals to the CAN-CBX station

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. 36), if the CAN bus ends there.

10.3 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. 33) downwards (e.g. with a screwdriver). Now the module is detached from the bottom edge of the mounting rail and can be removed.

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

11. Declaration of Conformity

CE-KONFORMITÄTSERKLÄRUNG
CE DECLARATION OF CONFORMITY

Adresse *Address*	**esd electronic system design gmbh** **Vahrenwalder Str. 207** **30165 Hannover** **Germany**
esd erklärt, dass die Produkte *esd declares, that the products*	**CAN-EtherCAT,** **CANopen-EtherCAT**
Typ, Modell, Artikel-Nr. *Type, Model, Article No.*	**C.2922.02** **C.2922.03**
die Anforderungen der Normen für industrielle Produkte (Klasse A) *fullfills the requirements of the standards* *for industrial products (class A)*	**EN 61000-6-4 (01/2007),** **EN 61000-6-2 (08/2005)**
gemäß folgendem Prüfbericht erfüllt. *according to test certificate.*	**H-K00-0366-09**
Das Produkt entspricht damit den EG-Richtlinien *Therefore the product corresponds to the EU-Directives*	**2004/108/EG (2004-Dez-15)**

Diese Erklärung gilt für alle Exemplare, die das CE-Zeichen tragen und verliert ihre Gültigkeit, wenn Veränderungen am Produkt vorgenommen werden.
This declaration is valid for all units with the CE label on it and it loses its validity if a modification is done on the product.

Name / *Name*	Dr. Ing. Werner Schulze
Funktion / *Title*	Geschäftsführer / *Managing Director*
Datum / *Date*	Hannover, den 2009-Jul-23

Rechtsgültige Unterschrift / *authorized Signature*

.....

I:\Texte\Doku\MANUALS\CAN\EtherCAT\Conform_EtherCAT.wpd

12. Order Information

Type	Properties	Order No.
CAN-EtherCAT	EtherCAT/CAN gateway, documentation and EtherCAT Slave Information (ESI) file on CD	C.2922.02
Accessories (InRailBus Option)		
 **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
Manuals		
CAN-EtherCAT-ME	Manual in English	C.2922.21

Table 17: Order information