EDS84DMOTCAN
13395082

L-force Communication

Communication Manual

8400 motec

E84DGFCCxxx

CANopen communication unit

Lenze

Contents

1 About this documentation

Contents

This documentation exclusively describes the system bus (CAN) and the CANopen-specific functions of the Inverter Drive 8400 motec.

Note!

This documentation supplements the **mounting instructions** and the **hardware manual "Inverter Drives 8400 motec"** supplied with the controller.

The features of the system bus (CAN) and CANopen-specific functions for the Inverter Drive 8400 motec are described in detail.

Typical applications are explained with the help of examples.

This documentation also contains ...

- the most important technical data for CAN communication;
- information on the installation and commissioning of the CAN network;
- information on CAN data transfer, CAN monitoring functions, communication-relevant parameters and implemented CAN objects.

The theoretical concepts are only explained to the level of detail required to understand the function of CAN communication with Inverter Drives 8400 motec.

Depending on the software version of the controller and of the »Engineer« software installed, the screenshots in this documentation may vary from the »Engineer« representation.

This documentation does not describe the software of other manufacturers. No responsibility is taken for corresponding information given in this documentation. Information on how to use the software can be obtained from the documents of the host (master).

All brand names used in this documentation are trademarks of their respective owners.

Tip!

Detailed information about the system bus (CAN) can be found on the website of the CAN user organisation CiA (CAN in Automation):

www.can-cia.org

Target group

This documentation is intended for all persons who plan, install, commission and maintain the networking and remote service of a machine.

Tip!

Information and software updates for Lenze products can be found in the Download area at:

www.Lenze.com

Information regarding the validity

The information given in this documentation is valid for the following devices:

Product series	Type designation	Variant
Inverter Drives 8400 motec **CANopen communication unit**	E84DGFCCxNx	CANopen
	E84DGFCCxJx	CANopen + safety

▸ Product features and variants (📖 15)

1.1 Document history

Version			Description
1.0	09/2010	TD17	First edition
2.0	01/2011	TD17	Update of the ... • Parameters for CANopen communication (📖 76) (version 02.00) • »Engineer« screenshots
3.0	11/2011	TD17	General revision

Your opinion is important to us!

These instructions were created to the best of our knowledge and belief to give you the best possible support for handling our product.

If you have suggestions for improvement, please e-mail us to:

feedback-docu@Lenze.de

Thank you for your support.

Your Lenze documentation team

1.2 Conventions used

This manual uses the following conventions to distinguish between different types of information:

Type of information	Writing	Examples/notes
Numbers		
Decimal	Standard notation	Example: 1234
Hexadecimal	0x[0 ... 9, A ... F]	Example: 0x60F4
Binary • Nibble	In inverted commas Point	Example: '100' Example: '0110.0100'
Decimal separator	Point	The decimal point is always used. Example: 1234.56
Text		
Program name	» «	PC software Example: Lenze »Engineer«
Control element	**Bold**	The **OK** button... / The **Copy** command... / The **Properties** tab... / The **Name** input field...
Hyperlink	Underlined	Optically highlighted reference to another topic. Can be activated with a mouse-click in this documentation.
Symbols		
Page reference	(📖 9)	Optically highlighted reference to another page. Can be activated with a mouse-click in this documentation.
Step-by-step instructions		Step-by-step instructions are indicated by a pictograph.

1.3 Terminology used

Term	Meaning
Controller	Lenze frequency inverter of the "Inverter Drives 8400 motec" product series
Standard device	
Drive unit Communication unit Wiring unit	The 8400 motec controller consists of the following modules: "drive unit", "communication unit", and "wiring unit". • The drive unit is available in various power classes. • The communication unit is available in the following versions: – No fieldbus – AS-i option – CANopen option – PROFIBUS option – PROFINET option – EtherCAT option • The wiring unit provides flexible connection options for an easy integration into the power supply of the machine.
»Engineer«	PC software from Lenze which supports you in "engineering" (parameter setting, diagnosing, and configuring) during the entire life cycle, i.e. from planning to maintenance of the commissioned machine.
Code	Parameter which serves to parameterise and monitor the controller. In normal usage, the term is usually referred to as "Index".
Subcode	If a code contains more than one parameter, these parameters are stored in "subcodes". In this documentation a slash "/" is used as a separator between the code and subcode (e.g. "C00118/3"). In normal usage, the term is also referred to as "Subindex".
Lenze setting	These are settings with which the device is preconfigured ex works.
Basic setting	
HW	Hardware
SW	Software

Note!

Some of the terms used originate from the CANopen protocol.

1.4 Notes used

The following signal words and symbols are used in this documentation to indicate dangers and important information:

Safety instructions

Structure of safety instructions:

Pictograph and signal word!

(characterise the type and severity of danger)

Note

(describes the danger and suggests how to avoid the danger)

Pictograph	Signal word	Meaning
	**Danger!**	**Danger of personal injury through dangerous electrical voltage** Reference to an imminent danger that may result in death or serious personal injury if the corresponding measures are not taken.
	**Danger!**	**Danger of personal injury through a general source of danger** Reference to an imminent danger that may result in death or serious personal injury if the corresponding measures are not taken.
	**Stop!**	**Danger of property damage** Reference to a possible danger that may result in property damage if the corresponding measures are not taken.

Application notes

Pictograph	Signal word	Meaning
	**Note!**	Important note for trouble-free operation
	**Tip!**	Useful tip for simple handling
		Reference to other documents

2 Safety instructions

Note!

Always observe the specified safety measures to avoid severe injury to persons and damage to property!

Always keep this documentation to hand in the vicinity of the product during operation.

2.1 General safety and application instructions

Danger!

Disregarding the following basic safety measures may lead to severe personal injury and damage to material assets.

- Lenze drive and automation components ...
  - must only be used as directed.
    ▸ Application as directed (📖 14)
  - must never be commissioned in the event of visible damage.
  - must never be technically modified.
  - must never be commissioned before they have been completely mounted.
  - must never be operated without the covers required.
  - can - depending on the degree of protection - have live, movable or rotating parts during operation and after operation. Surfaces can be hot.
- For Lenze drive components ...
  - use only the accessories approved.
  - use only original spare parts from the manufacturer.
- Observe all specifications given in the attached and associated documentation.
  - This is the precondition for safe and trouble-free operation and for achieving the specified product features.
    ▸ Product features and variants (📖 15)
  - The procedural notes and circuit details described in this document are only proposals. It is up to the user to check whether they can be adapted to the particular applications. Lenze does not take any responsibility for the suitability of the procedures and circuit proposals described.

- Only qualified personnel may work with and on Lenze drive and automation components. In accordance with IEC 60364 and CENELEC, these are persons ...
  - who are familiar with the installation, assembly, commissioning, and operation of the product.
  - who have the corresponding qualifications for their work.
  - who know all regulations for the prevention of accidents, directives and laws applicable on site and are able to apply them.

2.2 Device and application-specific safety instructions

- During operation, the communication unit must be connected to the wiring unit and the drive unit.
- With external voltage supply, always use a separate power supply unit, safely separated in accordance with EN 61800-5-1 in every control cabinet ("SELV"/"PELV").
- Only use cables that correspond to the given specifications.
  - Specification of the bus cable (📖 26)

Documentation of "Inverter Drives 8400 motec", control system, plant/machine

All other measures prescribed in this documentation must also be implemented. Observe the safety instructions and application notes specified in the documentation.

2.3 Residual hazards

Device protection

- The communication unit contains electronic components that can be damaged or destroyed by electrostatic discharge.
  - Installation (📖 22)

3 Product description

3.1 Application as directed

The CANopen communication unit ...

- is a unit that can only be used in conjunction with the following modules:

Product series	Type designation
Inverter Drives 8400 motec **Drive unit**	E84DGDVxxxxxxxx
Inverter Drives 8400 motec **Wiring unit**	E84DGVNxx

- is a device intended for use in industrial power systems.
- may only be operated under the operating conditions specified in this documentation.
- may only be used in CANopen networks.
- can also be used without being connected to the CANopen network.

Any other use shall be deemed inappropriate!

3.2 Product features and variants

The CANopen communication unit is available in the following versions:

Product series	Type designation	Features				
		Enclosure IP 65	CANopen M12	I/O: Terminal	I/O: M12	Safety
Inverter Drives 8400 motec **CANopen communication unit**	E84DGFCCANP	●	●		●	
	E84DGFCC9NP	●	●	●		
	E84DGFCCAJP	●	●		●	●
	E84DGFCC9JP	●	●	●		●

- The CANopen communication unit ...
  - is mounted on the wiring unit (E84DGVNxx);
  - is exclusively supplied internally by the drive unit (E84DGDVxxxxxxxx).
- The I/O connections can be led into the device via M12 connectors or by means of cable glands.
- In the E84DGFCC9xx version, a maximum of four digital inputs is conducted on M12 connectors (see "Inverter Drives 8400 motec" hardware manual).
- Devices without an integrated safety system (safety option) have no analog input and no relay output.
- In the case of the E84DGFCCxJx communication units, the integrated safety system can be used for the protection of persons on machines.
- Setting of the CAN node address and baud rate is possible via DIP switch or code.
- Communication with the Lenze »Engineer« (access to all Lenze parameters) is preferably carried out via the CAN bus. Furthermore communication can be effected via the diagnostic interface of the drive unit.

"Inverter Drives 8400 motec" hardware manual

Here you'll find detailed information on the integrated safety system (safety option).

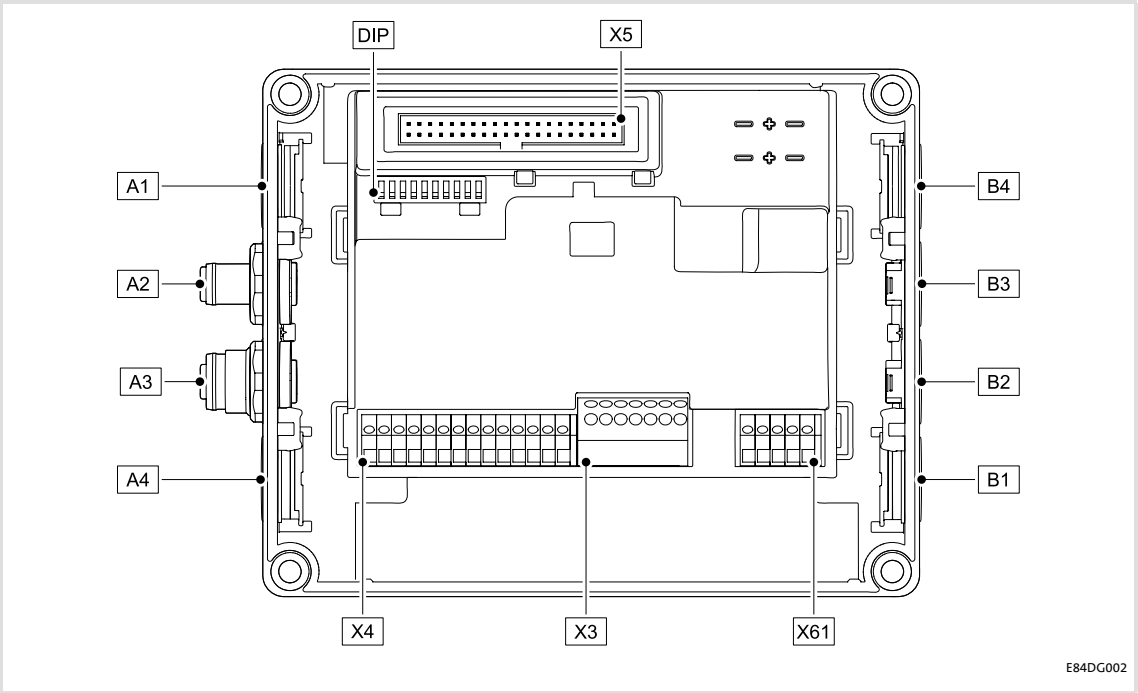
Software manual / »Engineer« online help "Inverter Drives 8400 motec"

Here you will find detailed information on how to configure the safety system (safety option).

The system bus (CANopen) of the Inverter Drives 8400 motec is the advanced version of the system bus (CAN) and includes the following features:

- Full compatibility according to CANopen DS301, V4.02.
- Support of the "Heartbeat" NMT slave function (DS301, V4.02).
- Number of parameterisable server SDO channels:
  - Max. 2 channels with 1 ... 8 bytes
  - Because of the 2 server SDO channels, the address range from 1 ... 63 is available.
- Number of parameterisable PDO channels:
  - Max. 2 transmit PDOs (TPDOs) with 1 ... 8 bytes (adjustable)
  - Max. 2 receive PDOs (RPDOs) with 1 ... 8 bytes (adjustable)
- All PDO channels are functionally equivalent.
- Monitoring of the RPDOs for data reception
- Adjustable error response to ...
  - physical CAN errors (frame, bit, ACK error)
  - bus-stop, bus-working
  - missing PDOs
- Telegram counters for SDOs and PDOs
- Bus status diagnostics
- Boot-up telegram generation
- Emergency telegram generation
- Reset node telegram generation (in the case of master configuration)
- Sync telegram generation and response to sync telegrams:
  - Data transmission/reception
  - Device-internal time base synchronisation
- Abort codes
- Object directory (all mandatory functions, optional functions, indexes)

3.3 Connections and interfaces



[3-1] CANopen communication unit

Pos.	Description
DIP	DIP switch ▸ Possible settings via DIP switch (📖 32)
A2	CANopen input (M12 pins, 5-pole) ▸ CANopen connection (📖 29)
A3	CANopen output (M12 socket, 5-pole) ▸ CANopen connection (📖 29)
A1 / A4 **B1 ... B4**	Positions for further freely designable inputs and outputs: • Digital inputs • Digital output • Analog input (only for E84DGFCCxJx) • Relay output (only for E84DGFCCxJx) • Connection of safety system "Safety Option" (only for E84DGFCCxJx)
X3 / X4 / X61	Terminal strips for wiring the connectors at A1 ... A4 and B1 ... B4
X5	Plug connector for connection to the drive unit

- By default, the CANopen connectors are already pre-assembled and wired with the terminal strip X3.
- The CANopen connections and further connections (e.g. digital inputs) can be freely designed at the positions A1 ... A4 and B1 ... B4..
- The connections can be equipped with 5-pole M12 connectors, or optionally with cable glands (cable cross-section max. 1.0 mm^2, AWG 18).
- The M12 connectors, cable glands and prefabricated system cables can be obtained from various manufacturers.
- Wire the M12 connectors or cable glands used to the corresponding contacts of terminal strips X3, X4, and X61.

"Inverter Drives 8400 motec" hardware manual

Observe the notes and wiring instructions given in the documentation.

4 Technical data

"Inverter Drives 8400 motec" hardware manual

Here you will find the **ambient conditions** and information on the **electromagnetic compatibility (EMC)** that also apply to the communication unit.

4.1 General data and operating conditions of the CANopen

Field	Values
Order designation	• E84DGFCCxNx (CANopen) • E84DGFCCxJx (CANopen + Safety)
Communication profile	CANopen, DS301 V4.02
Communication medium	DIN ISO 11898
Interface	• CANopen input: M12 pins, 5-pole, A-coded • CANopen output: M12 socket, 5-pole, A-coded
Network topology	Line terminated on both sides
Adjustable node address	1 ... 63 (can be set via DIP switch or code C00350)
Max. number of nodes	63
Baud rate [kbps]	20, 50, 125, 250, 500, 800, 1000 kbps, adjustable via DIP switches or code C00351
Process data	• Max. 2 transmit PDOs (TPDOs) with 1 ... 8 bytes (adjustable) • Max. 2 receive PDOs (RPDOs) with 1 ... 8 bytes (adjustable)
Parameter data	Max. 2 server SDO channels with 1 ... 8 bytes
Transmission mode for TPDOs	• With change of data • Time-controlled, 1 to x ms • After the reception of 1 to 240 sync telegrams
Conformities, approvals	• CE • UR / cUR

4.2 Supported protocols

Protocols	
Standard PDO protocols	PDO write PDO read
SDO protocols	SDO download SDO download initiate SDO download segment
	SDO upload SDO upload initiate SDO upload segment
	SDO abort transfer
	SDO block download SDO block download initiate SDO block download end
	SDO block upload SDO block upload initiate SDO block upload end
NMT protocols	Start remote node (master and slave)
	Stop remote node (slave)
	Enter pre-operational (slave)
	Reset node (slave and local device)
	Reset communication protocol (slave)
Monitoring protocols	Heartbeat (heartbeat producer and heartbeat consumer) • 1 Heartbeat Producer can be monitored.
	Emergency telegram (to master)

4.3 Communication time

The communication time is the time between the start of a request and the arrival of the corresponding response.

The communication times in a CANopen network depend on ...

- ▶ the processing time in the controller;
- ▶ the telegram runtime (baud rate / telegram length);
- ▶ the nesting depth of the network.

Processing time in the controller

Data	Processing time
Process data	Approx. 2 ms update cycle + 0 ... 1 ms processing time in the module + 1 ... x ms application task runtime of the technology application used (tolerance)
Parameter data	Approx. 30 ms + 20 ms tolerance (typical) • For some codes, the processing time may be longer (see software manual/»Engineer« online help "Inverter Drives 8400 motec").

There are no interdependencies between parameter data and process data.

5 Installation

Stop!

Electrostatic discharge

Electronic components within the communication unit can be damaged or destroyed by electrostatic discharge.

Possible consequences:

- The communication unit is defective.
- Communication via the fieldbus is not possible or faulty.
- I/O signals are faulty.
- The safety function is faulty.

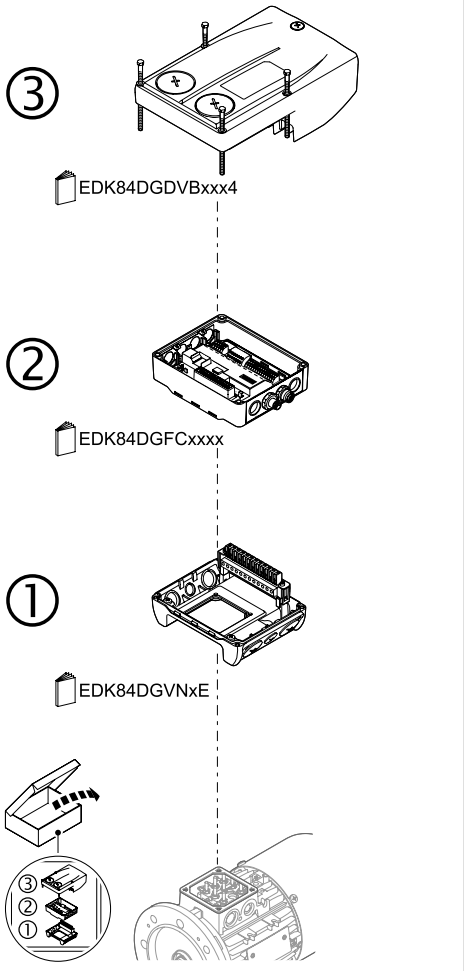
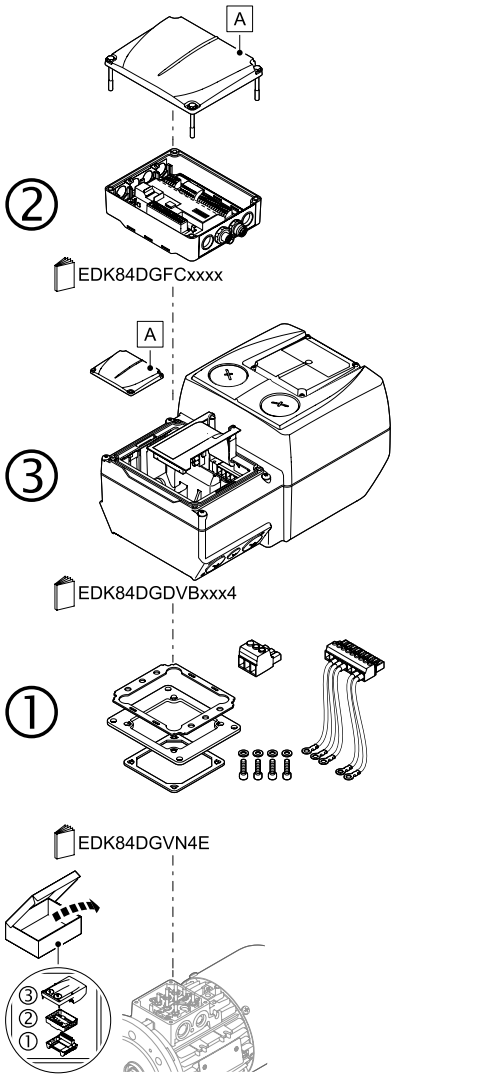
Protective measures

- Discharge electrostatic charges before touching the communication unit.

5.1 Mechanical installation

Mounting instructions for "Inverter Drives 8400 motec"

Here you will find detailed information on the installation.

0.37 ... 3.0 kW	4.0 ... 7.5 kW
	
E84DG023a	E84DG023b

[5-1] Mechanical installation of the 8400 motec components

Legend for Fig. [5-1]	
1	Drive unit
2	Communication unit
3	Wiring unit
A	Cover of the drive unit
EDK84DG...	Mounting instructions for the drive unit, communication unit, wiring unit

5.2 Electrical installation

"Inverter Drives 8400 motec" hardware manual

Here you'll find detailed information on ...

- the digital and analog inputs and outputs;
- the relay output;
- the integrated safety system (safety option);
- the wiring of the connections.

Observe the notes and wiring instructions given in the documentation.

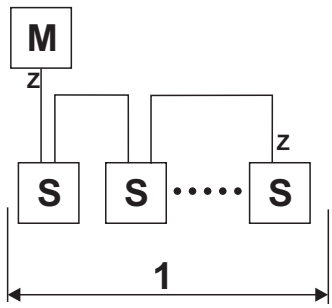
5.2.1 Network topology

The following examples show two simple CAN networks.

Each segment of the network must be terminated at both ends by resistors (120 Ω) between CAN-Low and CAN-High. The bus terminators of the system bus (CAN) are marked with a "Z" in the following examples.

A CAN network consisting of only one segment starts with the CAN master (M) with integrated bus termination, whereas the last CAN node (S) has to be terminated by a bus terminating resistor.

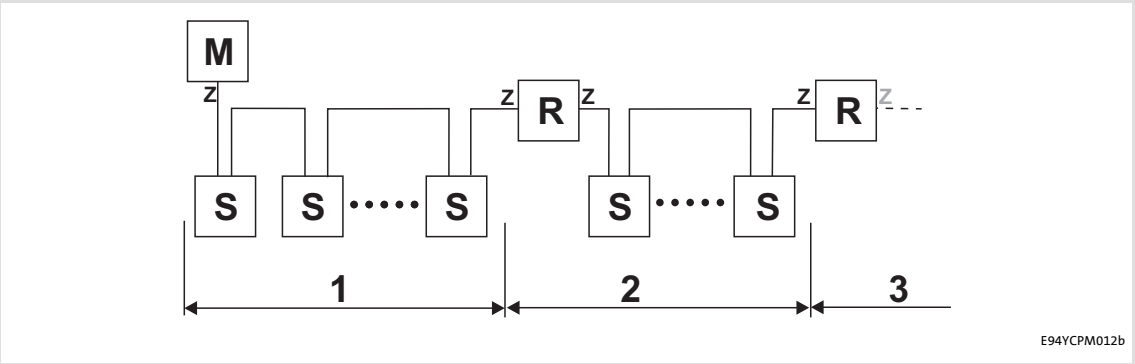
▸ Bus termination (📖 25)



[5-2] CAN network with one segment

A CAN network consisting of several segments contains repeaters (R) for connecting the segments. The repeaters are provided with integrated bus terminations.

▸ Consider the use of repeaters (📖 28)



[5-3] CAN network with repeaters

If no repeater is to be used at the end of the segment, the bus must be terminated by a bus terminating resistor at the last node (S). The bus termination is supplied by the node itself.

5.2.2 Bus termination

The system bus (CANopen) must be terminated through a bus terminating resistor at the first and last physical node (120 Ω).

In the case of the communication unit, the bus terminating resistor can only be installed externally at the M12 connector. This has the advantage that the presence of the resistor can be identified on the closed device.

Note!

- The CANopen terminals (input and output) must be installed so that they are closed. For this purpose either use a connecting cable, a closed terminating resistor plug (M12 pins, 5-pole, A-coded), or a cap.
- The connecting cable and terminating resistor plug can be procured freely from various cable manufacturers (e.g. Lapp or Turck).
- If you want to disconnect individual nodes, ensure that the bus terminations at the cable ends remain active. Otherwise the bus may become instable.
- Observe that the bus terminator is no longer active when the terminating resistor plug has been removed.

5.2.3 Specification of the bus cable

We recommend the use of CAN cables in accordance with ISO 11898-2:

CAN cable in accordance with ISO 11898-2	
Cable type	Paired with shielding
Impedance	120 Ω (95 ... 140 Ω)
Cable resistance/cross-section Cable length ≤ 300 m Cable length 301 ... 1000 m	 ≤ 70 mΩ/m / 0.25... 0.34 mm^2 (AWG22) ≤ 40 mΩ/m / 0.5 mm^2 (AWG20)
Signal propagation delay	≤ 5 ns/m

Observe the notes provided on the Bus cable length (📖 26)**!**

5.2.4 Bus cable length

Note!

- It is absolutely necessary to comply with the permissible cable lengths.
- Please take into account the reduction of the total cable length due to the signal delay of the repeater.
  ▸ Consider the use of repeaters (📖 28)
- Mixed operation refers to different nodes being connected to the same network.
- If the total cable lengths of the nodes are different at the same baud rate, the smaller value must be used to determine the maximum cable length.

Total cable length

1. Check that the total cable length is not exceeded.

   The total cable length is determined by the baud rate.

Baud rate [kbps]	Max. bus length [m]
20	4013
50	1575
125	600
250	275
500	113
800	38
1000	13

[5-1] Total cable length

Segment cable length

2. Check that the segment cable length is not exceeded

   The segment cable length is determined by the number of nodes and the cable cross-section used. Without a repeater, the segment cable length corresponds to the total cable length.

Maximum number of nodes per segment	Cable cross-section			
	0.25 mm²	0.5 mm²	0.75 mm²	1 mm²
2	240 m	430 m	650 m	940 m
5	230 m	420 m	640 m	920 m
10	230 m	410 m	620 m	900 m
20	210 m	390 m	580 m	850 m
32	200 m	360 m	550 m	800 m
63	170 m	310 m	470 m	690 m

[5-2] Segment cable length

3. Compare both values.

   If the value determined from the Segment cable length [5-2] table is smaller than the required total cable length Total cable length [5-1], repeaters must be used. Repeaters divide the total cable length into segments.

Selection example

Given	
• Cable cross-section:	0.5 mm², according to Specification of the bus cable (📖 26)
• Number of nodes:	63
• Repeater:	Lenze repeater, type 2176 (cable reduction: 30 m)

Based on the given specifications, the following cable lengths/number of repeaters result for a maximum of 63 nodes:

Baud rate [kbps]	20	50	125	250	500	800	1000
Max. cable length [m]	4013	1575	600	275	113	38	13
Segment cable length [m]	270	270	270	270	113	38	13
Number of repeaters	15	6	2	1	-	-	-

Consider the use of repeaters

Note!

The use of an additional repeater is recommended as:

- Service interface
  - Advantage: Trouble-free connecting during ongoing bus operation is possible.
- Calibration interface
  - Advantage: Calibration/programming units remain electrically isolated.

Given	
• Baud rate:	125 kbps
• Cable cross-section:	0.5 mm^2
• Number of nodes:	28
• Cable length:	450 m

Steps		Cable length	See
1	Total cable length at 125 kbps:	600 m	Table Total cable length [5-1] (📖 26)
2	Segment cable length for 28 nodes and a cable cross-section of 0.5 mm^2:	360 m	Table Segment cable length [5-2] (📖 27)
3	Comparison: The value determined in step 2 is smaller than the required cable length of 450 m.		

Conclusion:

▶ A cable length of 450 m is not possible without installing a repeater.

▶ After 360 m (step 2), a repeater must be installed.

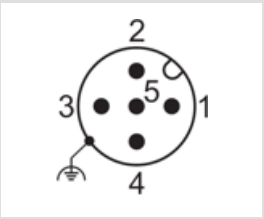
Result:

▶ The Lenze repeater, type 2176 (cable reduction: 30 m), is used

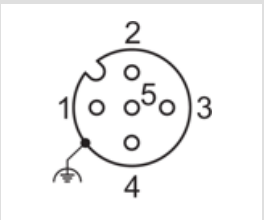
▶ Calculation of the max. cable length:
  – First segment: 360 m
  – Second segment: 360 m (according to the table Segment cable length [5-2] (📖 27)) *minus* 30 m (cable reduction when a repeater is used)

▶ Max. possible cable length with one repeater: 690 m
  – Now the required cable length is possible.

5.2.5 CANopen connection



- Input: M12 pins, 5-pole, A-coded
- Wiring at terminal strip X3



- Output: M12 socket, 5-pole, A-coded
- Wiring at terminal strip X3

CANopen connection		
Pin	**Signal**	**Description**
1	-	Not assigned
2	-	Not assigned
3	CG	CAN GND potential
4	CH	CAN-High data line
5	CL	CAN-Low data line

6 Commissioning

During commissioning, system-related data such as motor parameters, operating parameters, responses, and parameters for fieldbus communication are defined for the controller. For Lenze devices, this is done via the codes.

The codes of the controller and communication are saved non-volatilely as a data set in the memory module.

In addition, other codes are also available for diagnosing and monitoring the nodes.

▸ Parameter reference (📖 75)

6.1 Before initial switch-on

Stop!

Before switching on the device for the first time, please check ...

- the entire wiring for completeness, short circuit, and earth fault.
- whether the bus system is terminated through a bus terminating resistor at the first and last physical node.
  ▸ Bus termination (📖 25)

6.2 Configuring the host (master)

First you have to configure the host (master) for the communication with the controller.

Defining the user data length

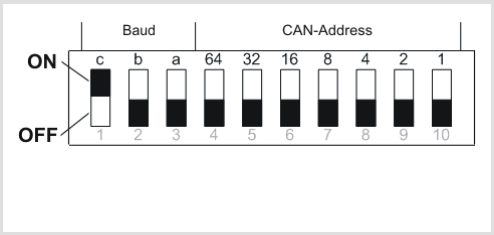
- The CANopen communication unit supports the configuration of max. 8 process data words (max. 64 bytes).
- The user data length is defined during the initialisation phase of the master.
- The user data lengths for process input data and process output data are the same.

Note!

The CANopen process data objects are designated as seen from the node's view:

- Receive PDO (RPDOx): Process data object received by a node
- Transmit PDO (TPDOx): Process data object sent by a node

6.3 Possible settings via DIP switch



[6-1] DIP switch

The DIP switches serve to ...

- ▶ Setting the baud rate (📖 32) (switches: a ... c)
- ▶ Setting the CAN node address (📖 33) (switches: 1 ... 64)

Lenze setting: All switches OFF

Note!

- The DIP switches can only be accessed when the drive unit is detached from the communication unit. Loosen the four fixing screws at the drive unit. **Observe the notes in the mounting instructions.**
- Switch off the voltage supply of the controller and the external supply of the communication unit before starting with the dismounting of the drive unit.
- The DIP switches are only read in when the device is switched on.

6.3.1 Setting the baud rate

The baud rate ...

- ▶ must be the same for all networked CANopen nodes;
- ▶ can be set via the DIP switches **a ... c** or via the »Engineer« (code C00351).

DIP switch position			Baud rate
c	**b**	**a**	
ON	OFF	**ON**	20 kbps
OFF	**ON**	**ON**	50 kbps
OFF	**ON**	OFF	125 kbps
OFF	OFF	**ON**	250 kbps
OFF	OFF	OFF	500 kbps
ON	**ON**	OFF	800 kbps
ON	OFF	OFF	1000 kbps

▸ Settings in the Lenze »Engineer« (📖 34)

6.3.2 Setting the CAN node address

The node addresses must differ from each other in the case of several networked CANopen nodes.

The node address can be set via DIP switches **1 ... 64** or via the »Engineer« with code C00350.

For the setting with C00350 DIP switches **1 ... 64 must be set to OFF**.

Note!

- The valid address range is 0 ... 63.
- If DIP switch 64 = ON (node address > 63), always node address 63 is used.

DIP switch							Node address
64	32	16	8	4	2	1	
OFF	OFF	OFF	OFF	OFF	OFF	OFF	Value from C00350
OFF	OFF	OFF	OFF	OFF	OFF	**ON**	1
OFF	...	...	...	...	...	...	...
OFF	**ON**	**ON**	**ON**	**ON**	**ON**	**ON**	63
ON	...	...	...	...	...	...	

The labelling on the housing corresponds to the values of the individual DIP switches for determining the node address.

DIP switch	**64**	**32**	**16**	**8**	**4**	**2**	**1**
Switch position	OFF	OFF	**ON**	OFF	**ON**	**ON**	**ON**
Value	0	0	16	0	4	2	1
Node address	**= Sum of the values = 16 + 4 + 2 + 1 = 23**						

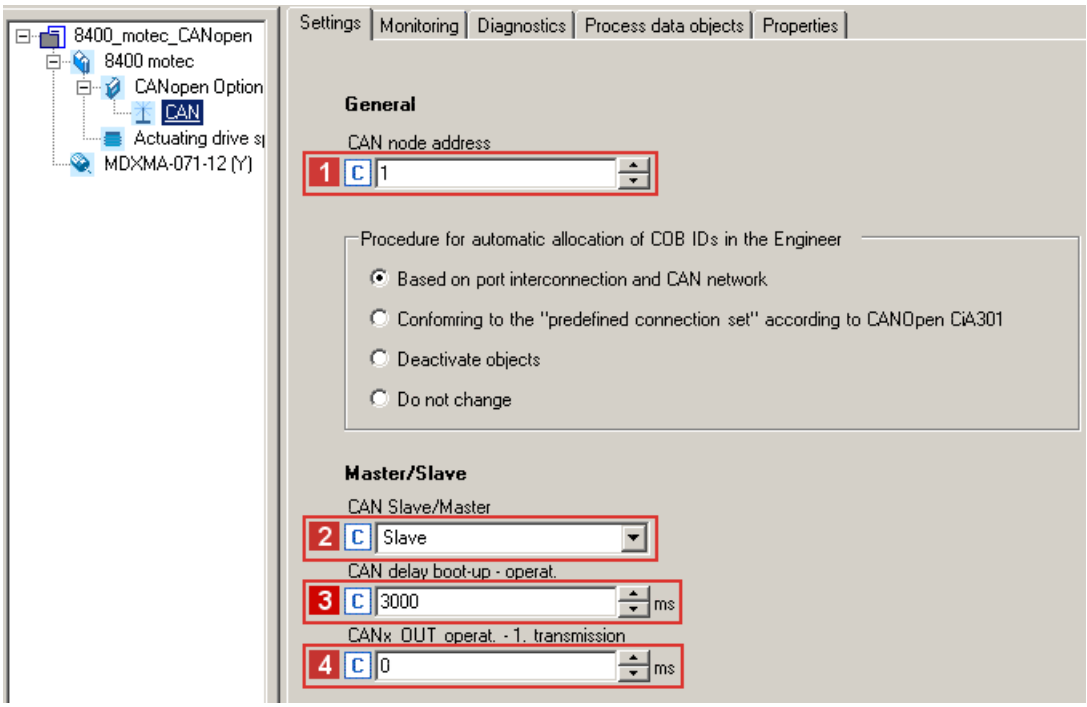
The current address setting of the DIP switches is displayed in C00349.

▸ DIP switch positions for setting the CAN node address (📖 106)
▸ Settings in the Lenze »Engineer« (📖 34)

6.4 Settings in the Lenze »Engineer«

The following settings can be made in the »Engineer« under the **Settings** tab:

- CAN node address 1 (C00350)
  - The node address can only be parameterised if the node address "0" is set via the DIP switches.
  - A change of the node address will only become effective after a CAN reset node.
- CAN node is slave or master 2 (C00352)
- Deceleration during status change from "Boot-up" to "Operational" 3 (C00356/1)
- Time to the first transmission of CANx_OUT in the "Operational" state 4 (C00356/4)



Save changed settings with the device command **C00002/11** (save all parameter sets).

6.5 Initial switch-on

Establishing communication

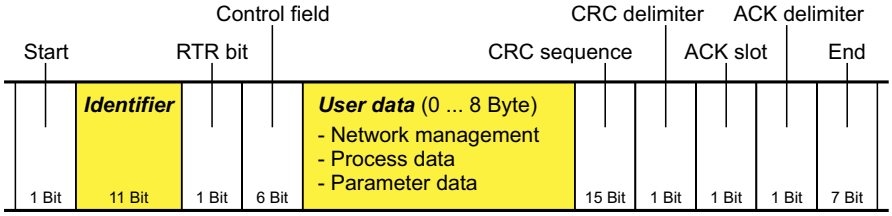
- To establish communication, the controller must be supplied with mains voltage.
- All parameters (codes) and DIP switch settings are read in when the device is switched on.
- If an error occurs, the error message "CE04: MCI communication error" (error no. 01.0127.00002) is output.
- The positions of the DIP switches define whether the CAN node address and the baud rate are selected via the DIP switches or via codes C00350 and C00351.
  - Possible settings via DIP switch (📖 32)

7 Data transfer

Via the system bus interface, for instance process data and parameter values can be exchanged between the nodes. In addition, the interface enables the connection of additional modules such as distributed terminals, keypads and input devices or external control systems and hosts (masters).

The system bus interface transfers CAN objects following the CANopen communication profile (CiA DS301, version 4.02) developed by the umbrella organisation of **CiA** (**C**AN **i**n **A**utomation) in conformity with the **CAL** (**C**AN **A**pplication **L**ayer).

7.1 Structure of the CAN data telegram



[7-1] Basic structure of the CAN telegram

The following subchapters provide a detailed description of the identifier and the user data. The other signals refer to the transfer characteristics of the CAN telegram the description of which is not included in the scope of this documentation.

Tip!

Please visit the homepage of the CAN user organisation CiA (CAN in automation) for further information:
http://www.can-cia.org

7.1.1 Identifier

The principle of CAN communication is based on a message-oriented data exchange between a transmitter and many receivers. All nodes can virtually transmit and receive simultaneously.

The identifier, also called COB-ID (abbr. for communication object identifier), is used to control which node is to receive a transmitted message. In addition to the addressing, the identifier contains information on the priority of the message and the type of user data.

The identifier consists of a basic identifier and the node address of the node to be addressed:

Identifier (COB-ID) = basic identifier + node address (node ID)

Exception: The identifier for process data/heartbeat/emergency objects as well as network management and sync telegrams is freely assigned by the user (either manually or automatically by the network configurator), or is permanently assigned.

Node address (node ID)

For unambiguous identification, a node address (also called node ID) within the valid address range (1 ... 63) must be assigned to every node of the system bus network.

- A node address may not be assigned more than once within a network.
- The own node address can be configured via the DIP switches or via code C00350.

▸ Setting the CAN node address (📖 33)

Identifier assignment

The system bus is message-oriented instead of node-oriented. Every message has an unambiguous identification, the identifier. For CANopen, node-oriented transfer is achieved by the fact that every message has only one transmitter.

- The basic identifiers for network management (NMT) and sync as well as the basic SDO channel (SDO1) are defined in the CANopen protocol and cannot be changed.
- In the Lenze setting, the basic identifiers of the PDOs are preset according to the "Predefined connection set" of DS301, V4.02 and can be changed via parameters/ indexes, if required.

▸ Identifiers of the process data objects (📖 51)

Object		Direction		Lenze-Base-ID		CANopen-Base-ID	
		from device	to device	dec	hex	dec	hex
Network management (NMT)				0	0	0	0
Sync [1)]				128	80	128	80
Emergency [1)]		●		128	80	128	80
PDO1 (Process data channel 1)	TPDO1	●		384	180	384	180
	RPDO1		●	512	200	512	200
PDO2 (Process data channel 2)	TPDO2	●		640	280	640	280
	RPDO2		●	641	281	768	300
SDO1 (Parameter data channel 1)	TSDO1	●		1408	580	1408	580
	RSDO1		●	1536	600	1536	600
SDO2 (Parameter data channel 2)	TSDO2	●		1472	5C0	1472	5C0
	RSDO2		●	1600	640	1600	640
Heartbeat		●		1792	700	1792	700
Boot-up		●		1792	700	1792	700

1) If you set the sync transmit/receive identifier manually, observe the use of the emergency telegram, since it has the same COB-ID.

7.1.2 User data

All nodes communicate by exchanging data telegrams via the system bus. The user data area of the CAN telegram either contains network management data, or parameter data, or process data:

Network management data

(NMT data)

- Control information on start, stop, reset, etc. of communication to specific nodes or to all nodes of the CAN network.

Process data

(PDOs – process data objects)

- Process data are transferred via the process data channel.
- Process data can be used to control the controller.
- Process data are not saved to the controller.
- Process data are transmitted between the host (master) and controllers (slaves) to ensure a continuous exchange of current input and output data.
- Process data usually are unscaled/scalable raw data.
- Process data are, for instance, setpoints and actual values.
- The exact meaning of the PDO file contents is determined via the function block editor (FB Editor) in the I/O level or via the PDO mapping.

Parameter data

(SDOs – service data objects)

- Parameter data are the CANopen indexes or, in the case of Lenze devices, the codes.
- Parameters are, for instance, used for one-off plant setting during commissioning or when the material on a production machine is changed.
- Parameter data are transmitted as SDOs via the parameter data channel. They are acknowledged by the receiver, i.e. the sender gets a feedback about whether the transmission was successful or not.
- The parameter data channel enables access to all Lenze codes and CANopen indexes.
- Parameter changes are automatically saved to the controller until mains switching.
- Generally the parameter transfer is not time-critical.
- Parameter data are, for instance, operating parameters, diagnostic information, and motor data.

7.2 Communication phases/network management

With regard to communication via the system bus, the controller distinguishes between the following states:

Status	Explanation
"Initialisation" (Initialisation)	After switch-on, an initialisation run is carried out. • During this phase, the controller is not involved in the data exchange via the bus. • The standard values are re-written to all CAN-relevant parameters. • After initialisation is completed, the controller is automatically set to the "Pre-operational" status.
"Pre-operational" (before being ready for operation)	Parameter data can be received, process data are ignored.
"Operational" (ready for operation)	Parameter data and process data can be received!
"Stopped" (stopped)	Only network management telegrams can be received.

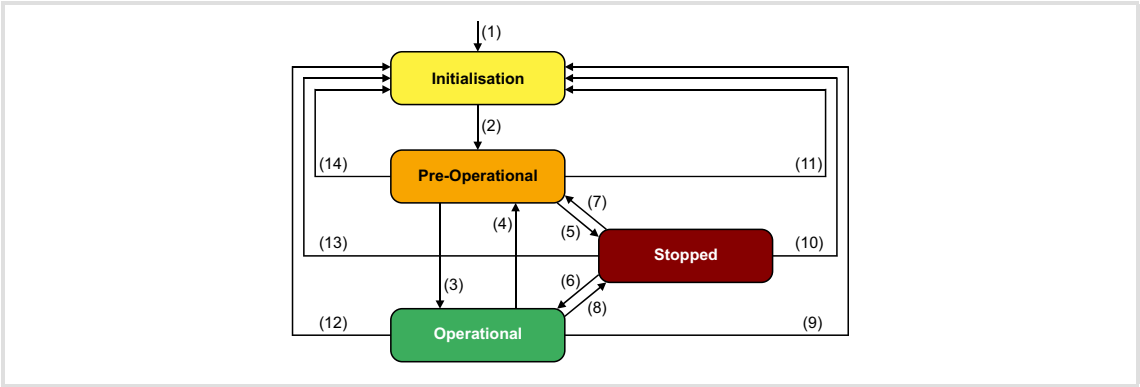
Communication object	Initialisation	Pre-operational	Operational	Stopped
PDO			●	
SDO		●	●	
Sync		●	●	
Emergency		●	●	
Boot-up	●			
Network management (NMT)		●	●	●

Code C00359 serves to display the status of the CAN bus.

Tip!

Part of the initialisation or the entire initialisation can be carried out again in every status by transmitting the corresponding network management telegrams.

7.2.1 State transitions

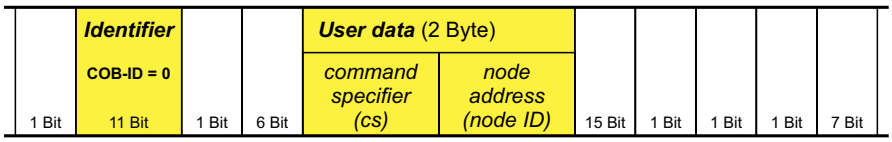


[7-2] NMT state transitions in the CAN network

Transition	NMT command	Status after change	Effects on process/parameter data after status change
(1)	-	Initialisation	Initialisation starts automatically at mains connection. • During initialisation, the controller is not involved in the data exchange. • After the initialisation is completed, the node sends a boot-up message with an individual identifier and automatically changes to the "Pre-operational" status.
(2)	-	Pre-operational	In this phase, the master determines the way in which the node(s) takes/take part in communication.
	From here, the master changes the states for the entire network. • A target address included in the NMT command defines the receiver(s). • If the controller is configured as CAN master, the status is automatically changed to "Operational" after a waiting time has expired (C00356/1) and the command 0x0100 NMT ("Start Remote Node") is transmitted to all nodes. • Data can only be exchanged via process data objects if the status is "Operational"!		
(3), (6)	0x01 xx Start remote node	Operational	Network management/sync/emergency telegrams as well as process data (PDO) and parameter data (SDO) are active. Optional: When the status is changed, event- and time-controlled process data (PDOs) are transmitted once.
(4), (7)	0x80 xx Enter Pre-operational	Pre-operational	Network management/sync/emergency telegrams and parameter data (SDO) are active.
(5), (8)	0x02 xx Stop remote node	Stopped	Only network management telegrams can be received.
(9), (10), (11)	0x81 xx Reset node	Initialisation	All CAN-relevant parameters (CiA DS 301) are initialised with the saved values.
(12), (13), (14)	0x82 xx Reset communication		All CAN-relevant parameters (CiA DS 301) are initialised with the saved values.
	Meaning of the node address in the NMT command: • xx = 0x00: If this assignment is selected, the telegram addresses all nodes (broadcast telegram). The status of all nodes can be changed at the same time. • xx = Node ID: If a node address is specified, only the status of the node with the corresponding address changes.		

7.2.2 Network management telegram (NMT)

The telegram for the network management contains the identifier "0" and the command included in the user data, which consists of the command byte and the node address:



[7-3] Network management telegram for changing over the communication phases

Command specifier (cs)		NMT command
dec	hex	
1	0x01	Start remote node
2	0x02	Stop remote node
128	0x80	Enter Pre-operational
129	0x81	Reset node
130	0x82	Reset communication

The change-over of the communication phases for the entire network is carried out by one node, the CAN master. The function of the CAN master can also be carried out by the controller.

▸ Parameterising the Inverter Drives 8400 motec as CAN master (📖 43)

Example:

Data can only be exchanged via process data objects if the status is "Operational". If the CAN master is supposed to switch all nodes connected to the bus from the "Pre-operational" communication status to the "Operational" communication status, the identifier and user data in the transmission telegram must be set as follows:

- Identifier: 0x00 (network management)
- User data: 0x0100 ("Start remote node" NMT command to all nodes)

7.2.3 Parameterising the Inverter Drives 8400 motec as CAN master

If the initialisation of the system bus and the associated status change from "Pre-operational" to "Operational" is not effected by a higher-level host, the Inverter Drive 8400 motec can instead be defined to be a "quasi" master to execute this task.

The controller is configured as CAN master in C00352.

- Being the CAN master, the controller sets all nodes connected to the bus (broadcast telegram) to the "Operational" communication status with the "Start remote node" NMT telegram. Only in this communication status data can be exchanged via process data objects.
- A delay time can be set in C00356/1, which must expire after mains switching before the controller transmits the "Start remote node" NMT telegram.

Parameter	Info	Lenze setting	
		Value	Unit
C00352	CAN slave/master	Slave	
C00356/1	CAN delay boot-up - Operational	3000	ms

Note!

The changes of the master/slave operation in C00352 will not be activated until

- another mains switching of the controller

or

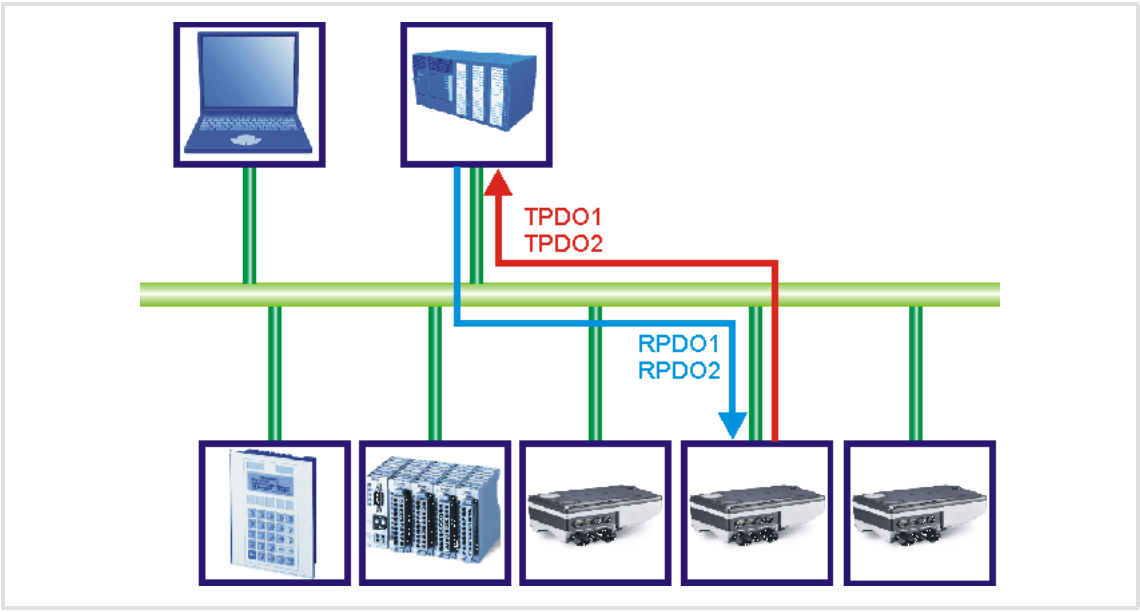
- the "Reset node" or "Reset communication" NMT telegram has been transmitted to the controller.

The "CAN reset node" device command (C00002/26) is provided as an alternative to the "Reset node" NMT telegram for the reinitialisation of the CAN-specific device parameters.

Tip!

Master functionality is only required during the initialisation phase of the drive system.

8 Process data transfer



[8-1] PDO data transfer from / to the higher-level host (master)

The CANopen communication unit is provided with two separate process data channels (PDO1 and PDO2) for transmitting process data. Each process data channel can transmit up to four words (8 bytes) at a maximum.

The system bus (CANopen) transmits parameter data, configuration data, diagnostic data, alarm messages and process data between the host (master) and the controllers (slaves) participating on the fieldbus. Depending on their time-critical nature, the data are transmitted via different communication channels.

- Process data are transmitted via the process data channel.
- The process data serve to control the controller.
- Transferring process data is time-critical.
- Process data are transferred cyclically between the master and the slaves participating on the fieldbus (continuous exchange of current input and output data).
- The master can directly access the process data. In the PLC, for instance, the data are directly assigned to the I/O area.
- Process data are not saved in the controller.
- Process data are, for instance, setpoints, actual values, control words and status words.

Definitions

- Process data telegrams between the host (master) and the controllers (slaves) are distinguished in terms of direction as follows:
  - Process data telegrams to the device (RPDO)
  - Process data telegrams from the device (TPDO)
- The CANopen process data objects are designated as seen from the node's view:
  - Receive PDOs (RPDOx): Process data object received by a node
  - Transmit PDOs (TPDOx): Process data object sent by a node

Note!

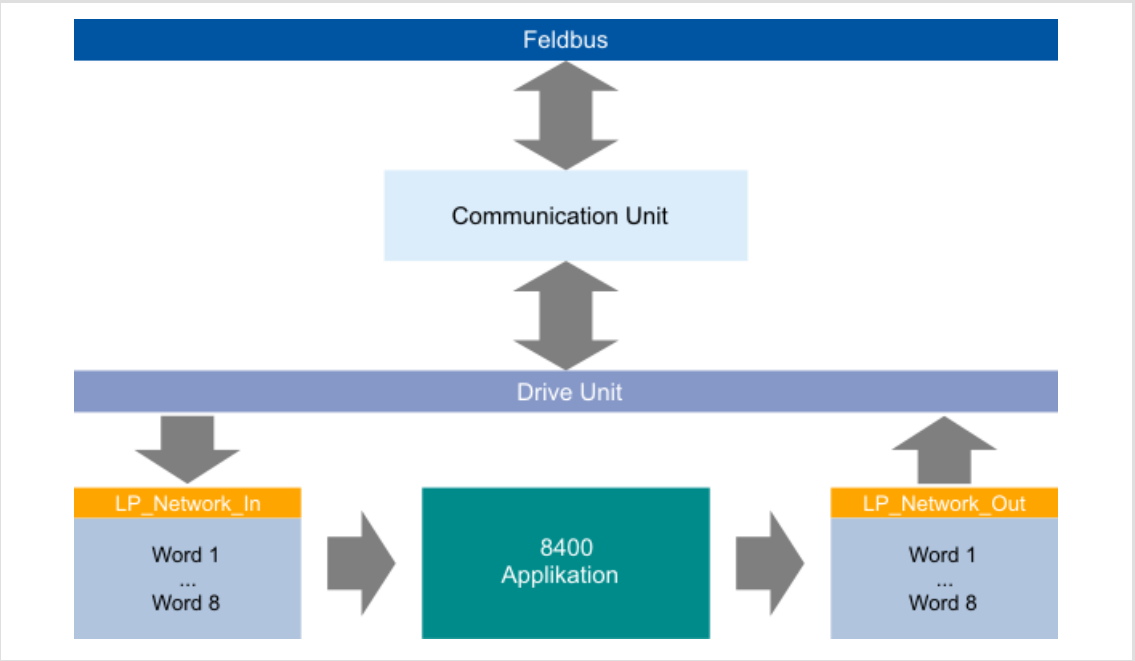
Data can only be exchanged via process data objects if the status is "Operational"!

▸ Communication phases/network management (📖 40)

8.1 Access to process data / PDO mapping

The process data are transferred via the MCI/CAN interface.

- Max. 8 words (16 bits/word) per direction can be exchanged.
  - 2 x 4 words via the input ports CAN1_IN and CAN2_IN
  - 2 x 4 words via the output ports CAN1_OUT and CAN2_OUT
- The process data are accessed via the port blocks **LP_Network_In** and **LP_Network_Out**. These port blocks are also called process data channels.
- The port/function block interconnection of the process data objects (PDO) takes place via the Lenze »Engineer«.



[8-2] External and internal data transfer between the bus system, controller, and application

Software manual / »Engineer« online help "Inverter Drives 8400 motec"

Here you will find detailed information on port blocks and port/function block interconnection in the »Engineer«.

8.2 Port interconnection of the process data objects (PDO)

Note!

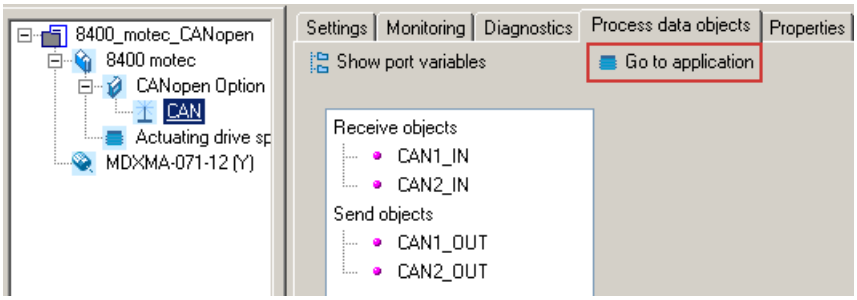
The »Engineer« screenshots shown on the following pages are only examples for the setting sequence and the resulting screens.

Depending on the software version of the controller and of the »Engineer« software installed, the screenshots may deviate from your »Engineer« representation.

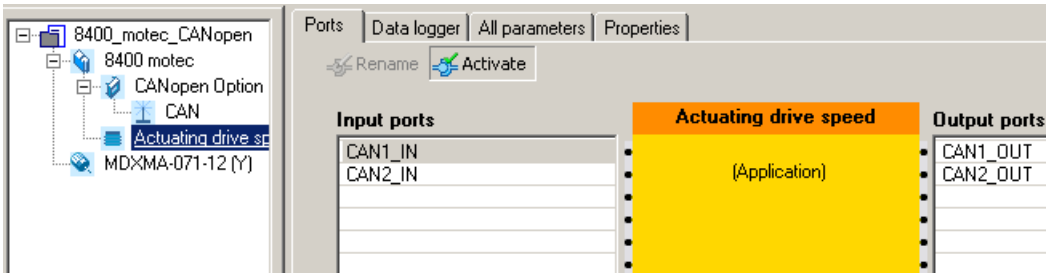
The preconfigured port interconnection of the process data objects is activated by setting code **C00007 = 40: Network (MCI/CAN)**.

How to freely configure the port interconnection in the »Engineer«:

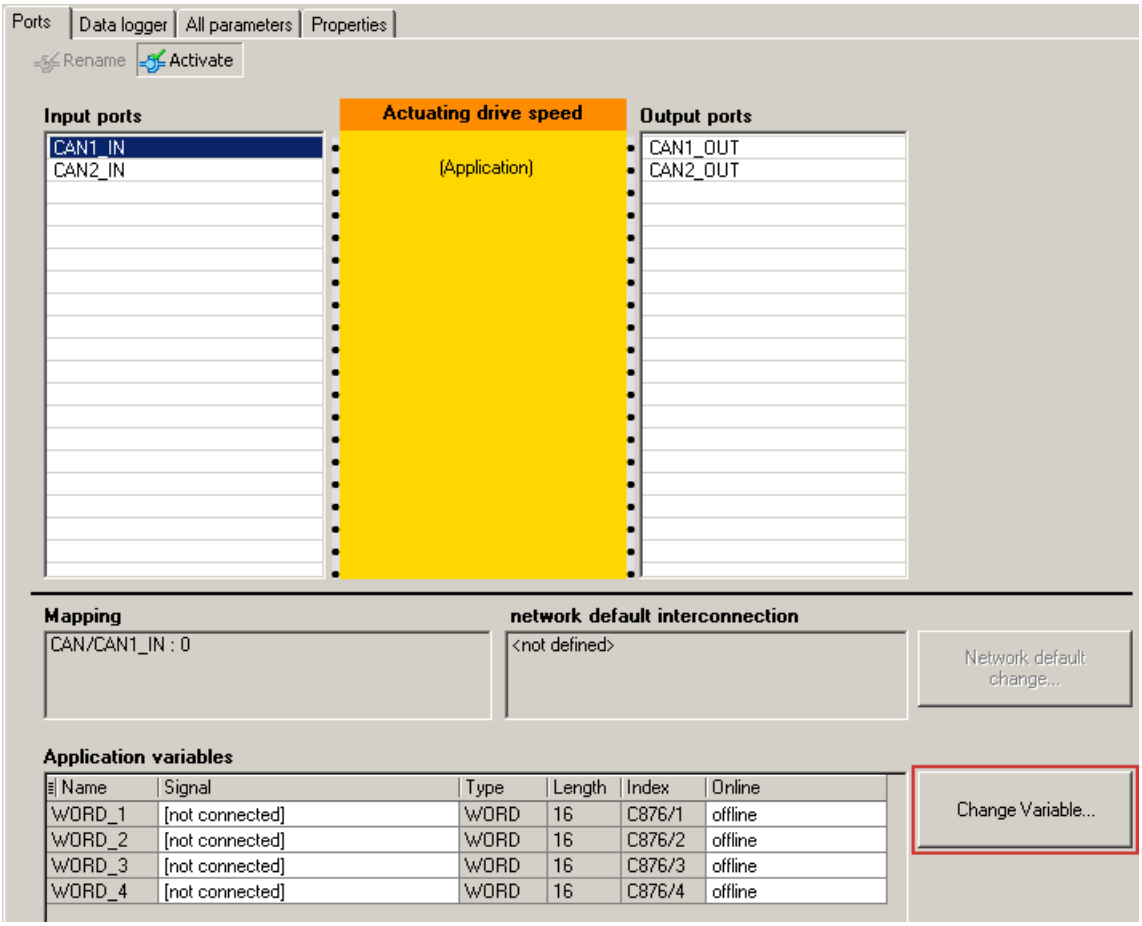
1. Under the **Process data object** tab, click the **Go to application** button.



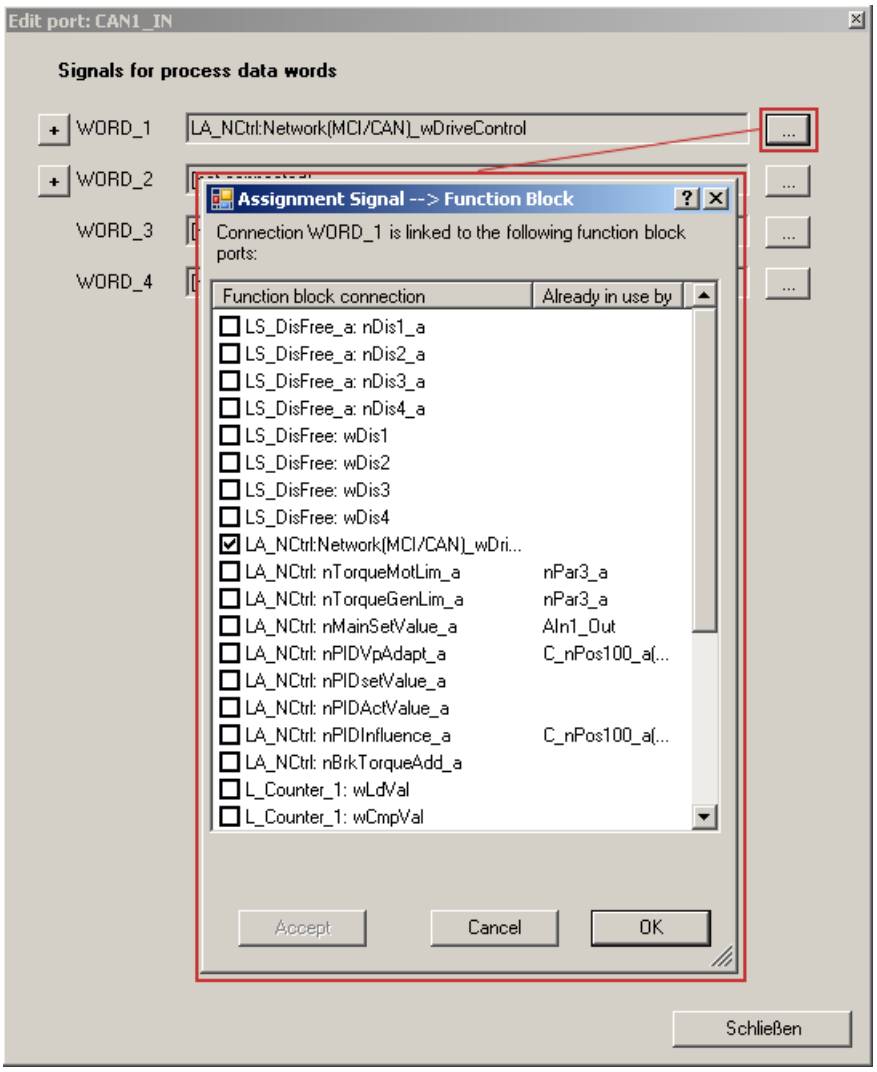
2. The **Ports** tab displays the port blocks CAN1_IN/CAN2_IN and CAN1_OUT/ CAN2_OUT.



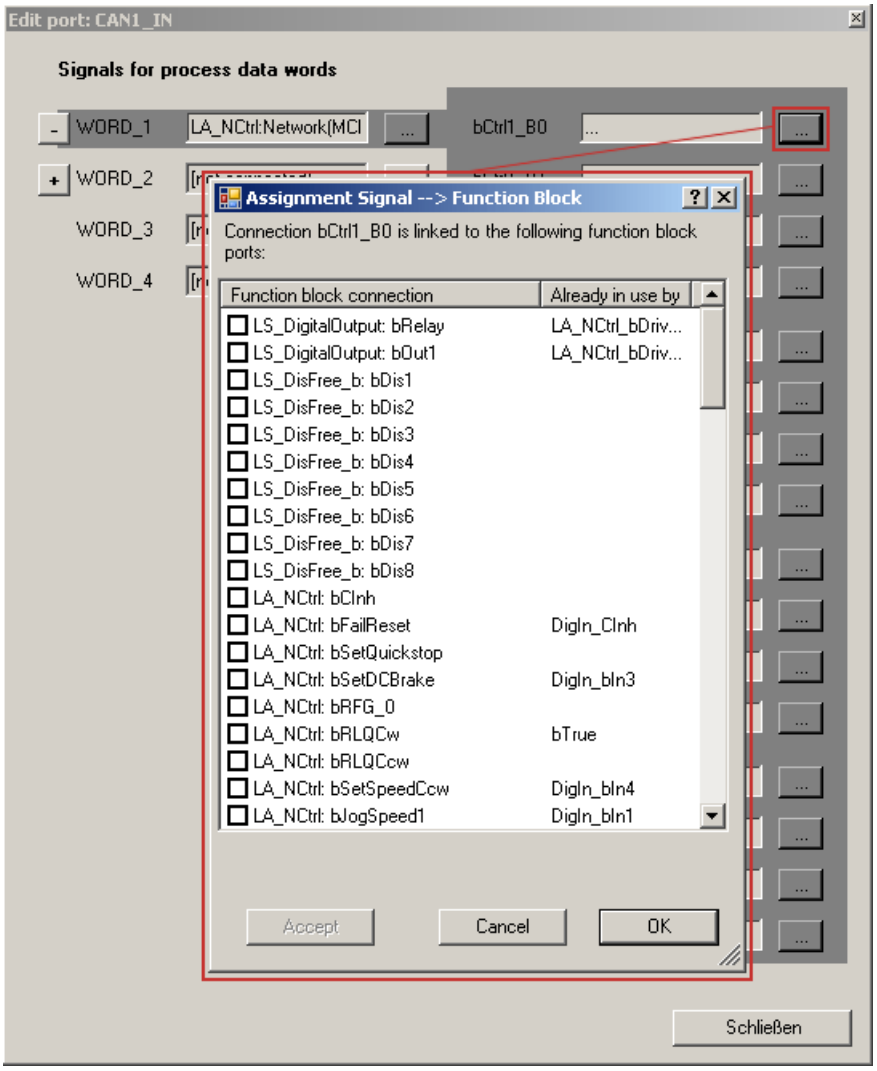
3. Click the port to be configured and press the **Edit port ...** button.



4. Via the [...] button, you can assign signals to the process data words in the *Signal assignment --> Function Block* dialog box.
   ➞ Select the signals and then confirm the selection with the **OK** button.



For some process data words, you can also assign signals to the individual bits via the [+] and [...] buttons.
➞ Select the signals and then confirm the selection with **OK**.



The current interconnection is only displayed if the following has been set for the control mode in code **C00007 = 40: Network (MCI/CAN)**.

8.3 Identifiers of the process data objects

In the Lenze setting, the identifier for the process data objects PDO1 and PDO2 consists of a basic identifier (CANBaseID) and the node address set in C00350:

Identifier (COB-ID) = basic identifier + node address (node ID)

- The basic identifiers of the PDOs comply with the "Predefined connection set" of DS301, V4.02.
- Alternatively, define via code C00353 that the identifiers of the PDOs are to be assigned according to Lenze definition or that individual settings are to be made.
  - If C00353 = "2: COBID = C0354/x", the identifiers of the PDOs can be set individually via the Lenze codes and CANopen indexes listed in the table below. That way, identifiers independent of the node address can be set for specific PDOs.
  - If identifiers are assigned individually, all PDOs must have basic identifier values in the range of 385 ... 1407.

Process data object	Basic identifier		Individual setting	
	dec	hex	Lenze code	CANopen index
PDO1				
RPDO1	512	0x200	C00354/1	I-1400/1
TPDO1	384	0x180	C00354/2	I-1800/1
PDO2				
RPDO2	768	0x300	C00354/3	I-1401/1
TPDO2	640	0x280	C00354/4	I-1801/1

Note!

After a node address change (C00350) and a subsequent CAN reset node, the subcodes of C00354 automatically resume the values which result from the respective basic identifier and the node address set.

Short overview: Parameters for setting the identifiers

Parameter	Info	Lenze setting	
		Value	Unit
C00353/1	COBID source CAN1_IN/OUT	0: COBID = C0350 + CANBaseID	
C00353/2	COBID source CAN2_IN/OUT	0: COBID = C0350 + CANBaseID	
C00354/1	COBID CAN1_IN	0x00000201	
C00354/2	COBID CAN1_OUT	0x00000181	
C00354/3	COBID CAN2_IN	0x00000301	
C00354/4	COBID CAN2_OUT	0x00000281	

8.4 Transmission type

Process data objects can be transmitted in an event-controlled or time-controlled manner. The below table shows that it is possible to combine the different methods by means of logic operations (AND, OR):

- Event-controlled
  The PDO is sent when a certain device-internal event has occurred, e.g. when the data contents of the TPDO have changed or when a transmission cycle time has elapsed
- Synchronous transmission
  A TPDO (or RPDO) is sent (or received) after the device has received a sync telegram (COB-ID 0x80).
- Cyclic transmission
  The cyclic transmission of PDOs takes place when the transmission cycle time has elapsed.
- Polled via RTR
  A TPDO is sent when another device requests it by means of a data request telegram (RTR remote transmit request). For this purpose, the data requester (e.g. the master) sends the data request telegram with the COB-ID of the TPDO requested to be sent. The receiver recognises the RTR and transmits the corresponding PDO.

Transmission type	PDO transmission			Logic combination of different transmission types
	cyclic	synchronous	event-controlled	
0		●	●	AND
1 ... 240	●	●		AND
254	●		●	OR

Transmission type	Description
0	Synchronous and acyclic: The PDO is transmitted on an event-controlled basis with every sync (e.g. when a bit change occurs in the PDO).
1 ... 240	Synchronous and cyclic (sync-controlled with response): • Selection n = 1: The PDO is transmitted with every sync. • Selection 1 < n ≤ 240: The PDO is transmitted with every n-th sync.
241 ... 251	Reserved
252, 253	RTR-controlled manner is not permissible.
254	Event-controlled with cyclic transmission: If this value is entered, the PDO is transferred in an event-controlled or cyclic manner. (The values "254" and "255" are equivalent). For a cyclic transmission, a cycle time must be set for the respective PDO. In this case, cyclic transmission takes place in addition to event-controlled transmission.
255	Not permissible

The communication parameters such as the transmission mode and cycle time can be set freely for every PDO and independently of the settings of other PDOs:

Parameter	Info	Lenze setting	
		Value	Unit
C00322/1	Transmission mode CAN1 OUT	254	
C00322/2	Transmission mode CAN2 OUT	254	
C00323/1	Transmission mode CAN1 IN	254	
C00323/2	Transmission mode CAN2 IN	254	
C00324/1	Inhibit time for emergency telegrams	0	ms
C00324/2	CAN1_OUT inhibit time	0	ms
C00324/3	CAN2_OUT inhibit time	0	ms
C00356/5	CAN1_OUT cycle time	0	ms
C00356/2	CAN2_OUT cycle time	0	ms

Tip!

The setting can also be carried out via the following CANopen objects:

- I-1400 / I-1401: Communication parameter for RPDO1 and RPDO2
- I-1800 / I-1801: Communication parameter for TPDO1 and TPDO2

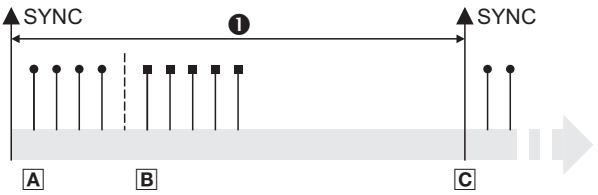
8.5 PDO synchronisation via sync telegram

During cyclic transmission, one or more PDOs are transmitted/received in fixed time intervals. An additional specific telegram, the so-called sync telegram, is used for synchronising cyclic process data.

- The sync telegram is the trigger point for the transmission of process data from the slaves to the master and for the acceptance of process data from the master in the slaves.
- For sync-controlled process data processing, the sync telegram must be generated accordingly.
- The response to a sync telegram is determined by the selected transmission type.

Transmission type (📖 52)

Basic workflow



❶ Sync cycle time

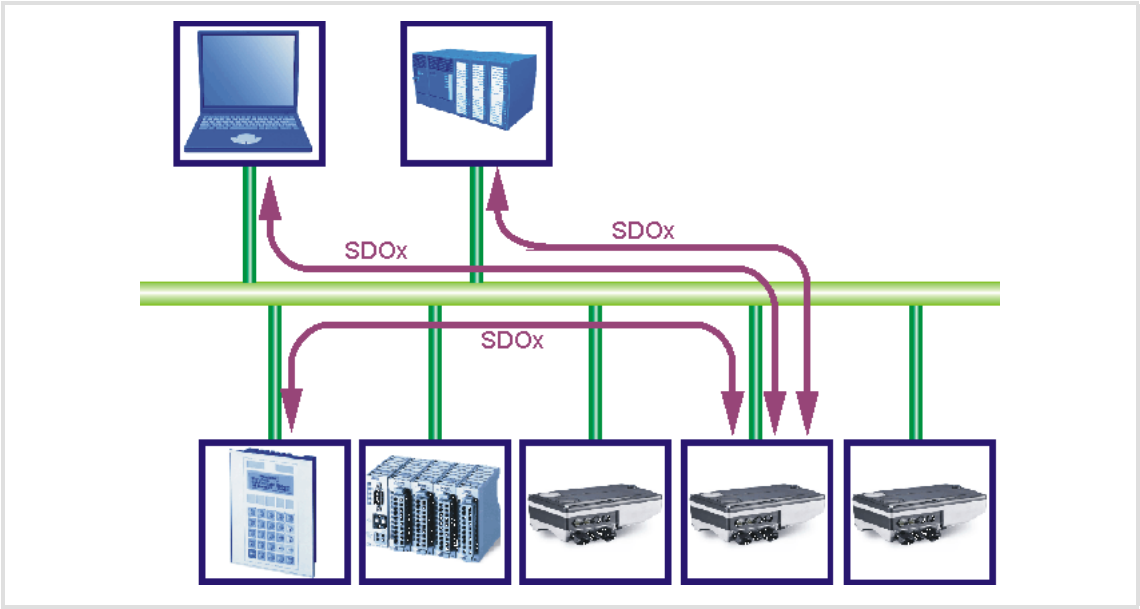
[8-3] Sync telegram

A. After the sync telegram has been received, the slaves transmit the synchronous process data to the master (TPDOs). The master reads them as process input data.
B. When the transmission process is completed, the slaves receive (RPDOs) the process output data (of the master).
   – All other telegrams (e.g. parameters or event-controlled process data) are accepted acyclically by the slaves after the transmission is completed.
   – Illustration [8-3] does not include acyclic data. However, they need to be considered when dimensioning the cycle time.
C. The data are accepted in the slave with the next sync telegram if the Rx mode is set to 1 ... 240. If the Rx mode is 254 or 255, the data are accepted in the next device cycle, irrespective of the sync telegram.

Short overview: Parameters for the synchronisation via sync telegram

Parameter	Info	Lenze setting		Assignment	
		Value	Unit	Sync master	Sync slave
C00367	CAN sync Rx identifier	128			●
C00368	CAN sync Tx identifier	128		●	
C00369	CAN sync transmission cycle time	0	ms	●	

9 Parameter data transfer



[9-1] Parameter data transfer via the available parameter data channels

Parameters are values stored in codes on Lenze controllers.

Two parameter data channels are available for parameter setting, enabling the simultaneous connection of different devices for configuration purposes.

Parameter data are transmitted via the system bus as SDOs (*Service Data Objects*) via the system bus (CANopen) and are acknowledged by the receiver. The SDO enables read and write access to all device parameters and to the CANopen object directory integrated in the device. Indexes (e.g. 0x1000) enable access to device parameters and functions included in the object directory. To transfer SDOs, the information contained in the user data must comply with the CAN-SDO protocol.

9.1 Identifiers of the parameter data objects

In the Lenze setting, the basic identifiers of the SDOs are preset according to the "Predefined Connection Set".

The identifiers for the parameter data objects SDO1 and SDO2 are generated from the basic identifier and the node address set in code C00350:

Identifier = basic identifier + node address

Object		Direction		Lenze-Base-ID		CANopen-Base-ID	
		from device	to device	dec	hex	dec	hex
SDO1 (Parameter data channel 1)	TSDO1	●		1408	580	1408	580
	RSDO1		●	1536	600	1536	600
SDO2 (Parameter data channel 2)	TSDO2	●		1472	5C0	1472	5C0
	RSDO2		●	1600	640	1600	640
Heartbeat		●		1792	700	1792	700
Boot-up		●		1792	700	1792	700

Note!

Please observe that the parameter data channels 1 and 2 are active in the factory setting.

9.2 User data

Structure of the user data of the parameter data telegram

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
	Low byte	High byte		Low word		High word	
				Low byte	High byte	Low byte	High byte

Note!

For the user data, the Motorola format is used.

▸ Parameter data telegram examples (📖 62)

The following subchapters provide detailed information on user data.

9.2.1 Command

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
	Low byte	High byte		Low word		High word	
				Low byte	High byte	Low byte	High byte

The following commands can be transmitted or received for writing and reading the parameters:

Command	1st byte		Data length	Info
	hex	dec		
Write request	0x23	35	4 bytes	Writing of a parameter to the controller.
	0x2B	43	2 bytes	
	0x2F	47	1 byte	
	0x21	33	Block	
Write response	0x60	96	4 bytes	Controller acknowledges a write request.
Read request	0x40	64	4 bytes	Reading of a parameter from the controller.
Read response	0x43	67	4 bytes	Controller's response to a read request with the current parameter value.
	0x4B	75	2 bytes	
	0x4F	79	1 byte	
	0x41	65	Block	
Error response	0x80	128	4 bytes	Response of the controller if the write/read request could not be executed correctly. ▸ Error messages (📖 60)

More precisely, the command byte comprises the following information:

Command	1st byte							
	Command specifier (cs)			Toggle (t)	Length*		e	s
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Write request	0	0	1	0	0/1	0/1	1	1
Write response	0	1	1	0	0	0	0	0
Read request	0	1	0	0	0	0	0	0
Read response	0	1	0	0	0/1	0/1	1	1
Error response	1	0	0	0	0	0	0	0

*Bit coding of the length: 00 = 4 bytes, 01 = 3 bytes, 10 = 2 bytes, 11 = 1 byte
e: expedited (shortened block service)
s: segmented (normal block service)

Tip!

More commands are defined in CANopen specification DS301, V4.02 (e.g. segmented transfer).

9.2.2 Addressing by means of index and subindex

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
	Low byte	High byte		Low word		High word	
				Low byte	High byte	Low byte	High byte

A parameter (a Lenze code) is addressed as per the following formula:
Index = 24575 - (Lenze code number)

Example

The C00011 parameter (motor reference speed) is to be addressed.

Calculation:

- Index:
  - Decimal: 24575 - 11 = 24564
  - Hexadecimal: 0x5FFF - 0xB = 0x5FF4
- Subindex: 0x00 (subindex 0 since the parameter does not have any subcodes)

Entries:

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
	0xF4	0x5F	0x00				

9.2.3 Data 1 ... data 4

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
	Low byte	High byte		Low word		High word	
				Low byte	High byte	Low byte	High byte

Maximally 4 bytes are available for parameter value entries. Depending on the data format, they are assigned as follows:

5th byte	6th byte	7th byte	8th byte
Parameter value (1 byte)	0x00	0x00	0x00
Parameter value (2 bytes)		0x00	0x00
Low byte	High byte		
Parameter value (4 bytes)			
Low word		High word	
Low byte	High byte	Low byte	High byte

Note!

The "Factor" column of the Table of attributes (📖 87) contains a scaling factor for all Lenze parameters. The scaling factor is relevant to the transfer of parameter values which have one or more decimal positions in the parameter list.

If the scaling factor is > 1, the value must be multiplied by the indicated scaling factor prior to transmission to be able to transfer the value as an integer. At the SDO client end, the integer must be divided by the scaling factor to obtain the original value including decimal positions again.

Example

A value of "123.45" is to be transmitted for a code, unit: "%" (e.g. C00039/1: "Fixed setpoint-JOG1").

Parameters with the "%" unit have two decimal positions and hence a scaling factor of "100".

Calculation:

- Value to be transmitted = scaling factor x value
- Data $_{(1 \ldots 4)}$ = 100 x 123.45 = 12345 (0x00 00 30 39)

Entries:

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
				0x39	0x30	0x00	0x00

9.2.4 Error messages

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Command	**Index**		**Subindex**	**Error code**			
0x80 (128)	Low byte	High byte		Low word		High word	
				Low byte	High byte	Low byte	High byte

In the event of an error, the node addressed generates a telegram with the "Error response" (0x80) command.

- The telegram includes the index and subindex of the code where the error occurred.
- The error code is entered in bytes 5 ... 8.
  - The error codes are standardised according to DS301, V4.02.
  - The representation of the error codes is provided in reverse read direction (see example below).

Example

Representation of error code "0x06 04 00 41" in bytes 5 ... 8:

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Command	**Index**		**Subindex**	**Error code**			
				0x41	**0x00**	**0x04**	**0x06**

Meaning of the error codes

The error codes are standardised according to DS301, V4.02.

Error code	Explanation
0x0503 0000	Toggle bit not changed
0x0504 0000	SDO protocol expired
0x0504 0001	Invalid or unknown client/server command specifier
0x0504 0002	Invalid block size (only block mode)
0x0504 0003	Invalid sequence number (only block mode)
0x0504 0004	CRC error (only block mode)
0x0504 0005	Not sufficient memory
0x0601 0000	Object access not supported
0x0601 0001	Attempt to read a write-only object
0x0601 0002	Attempt to write to a read-only object
0x0602 0000	Object not listed in object directory
0x0604 0041	Object not mapped to PDO
0x0604 0042	Number and length of objects to be transferred longer than PDO length.
0x0604 0043	General parameter incompatibility
0x0604 0047	General internal device incompatibility
0x0606 0000	Access denied because of hardware error
0x0607 0010	Unsuitable data type, unsuitable service parameter length
0x0607 0012	Unsuitable data type, service parameter length exceeded
0x0607 0013	Unsuitable data type, service parameter length not long enough
0x0609 0011	Subindex does not exist
0x0609 0030	Parameter value range exceeded
0x0609 0031	Parameter values too high
0x0609 0032	Parameter values too low
0x0609 0036	Maximum value falls below minimum value
0x0800 0000	General error
0x0800 0020	Data cannot be transferred/saved for application.
0x0800 0021	Data cannot be transferred/saved for application due to local control.
0x0800 0022	Data cannot be transferred/saved for application due to current device status.
0x0800 0023	Dynamic generation of object directory failed or no object directory available (e.g. object directory generated from file, generation not possible because of a file error).

9.3 Parameter data telegram examples

9.3.1 Reading parameters

Task: The heatsink temperature of 43 °C (code: C00061, data format: INTEGER32, scaling factor: 1) of the controller with node address "5" is to be read.

Telegram to the drive

Identifier	User data							
	1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
	Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
0x0605	0x40	0xC2	0x5F	0x00	0x00	0x00	0x00	0x00

Explanations on the telegram to the drive	
Identifier	= 1536 + node address = 1536 + 5 = 1541 = 0x0605 (1536 = SDO1 basic identifier to the controller)
Command	= 0x40 = "Read request" (read request of a parameter from the controller)
Index	= 24575 - code number = 24575 - 61 = 24514 = 0x5FC2
Subindex	= 0 (code C00061 does not have any subcodes)

Response message from drive (if data have been transmitted correctly)

Identifier	User data							
	1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
	Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
0x0585	0x43	0xC2	0x5F	0x00	0x2B	0x00	0x00	0x00

Explanations on the telegram from the drive	
Identifier	= 1408 + node address = 1408 + 5 = 1413 = 0x0585 (1408 = SDO1 basic identifier from the controller)
Command	= 0x43 = "Read response" (response to the read request with current value)
Index	As in telegram to the drive
Subindex	
Data 1 ... 4	= 0x0000002B = 43 [°C]

9.3.2 Write parameters

Task: The rated current of the connected motor is to be entered with I_N = 10.20 A (code C00088) into the controller with node address "2".

Data 1 ... 4	Calculation
Value for motor current, (data type U16; display factor 1/100)	10.20 x 100 = 1020 (0x03 FC)

Telegram to the drive

Identifier	User data							
	1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
	Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
0x0602	0x23	0xA7	0x5F	0x00	0xFC	0x03	0x00	0x00

Explanations on the telegram to the drive	
Identifier	= 1536 + node address = 1536 + 2 = 1538 = 0x0602 (1536 = SDO1 basic identifier to the controller)
Command	= 0x23 = "Write request" (write request of a parameter to the controller)
Index	= 24575 - code number = 24575 - 88 = 24487 = 0x5FA7
Subindex	= 0 (code C00088 does not have any subcodes)
Data 1 ... 4	= 10.20 x 100 = 1020 = 0x000003FC (motor current value; data type U32; display factor 1/100)

Response message from drive (if data have been transmitted correctly)

Identifier	User data							
	1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
	Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
0x0582	0x60	0xA7	0x5F	0x00	0x00	0x00	0x00	0x00

Explanations on the telegram from the drive	
Identifier	= 1408 + node address = 1408 + 2 = 1410 = 0x0582 (1408 = SDO1 basic identifier from the controller)
Command	= 0x60 = "Write response" (acknowledgement of the write access from the controller)
Index	As in telegram to the drive
Subindex	

9.3.3 Reading block parameters

Task: The firmware version (code C00099) is to be read from the parameter set of the controller with node address "12". The firmware version has a length of 11 ASCII characters which are transmitted as a block parameter. Depending on the block, the data width from the 2nd to 8th byte is assigned within the user data.

Telegram 1 to the drive: Read request

Identifier	User data							
	1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
	Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
0x060C	0x40	0x9C	0x5F	0x00	0x00	0x00	0x00	0x00

Explanations on the telegram to the drive	
Identifier	= 1536 + node address = 1536 + 12 = 1548 = 0x060C (1536 = SDO1 basic identifier to the controller)
Command	= 0x40 = "Read request" (read request of a parameter from the controller)
Index	= 24575 - code number = 24575 - 99 = 24476 = 0x5F9C
Subindex	= 0 (code C00099 does not have any subcodes)

Response message 1 from the drive: Indication of the block length (11 characters)

Identifier	User data							
	1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
	Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
0x058C	0x41	0x9C	0x5F	0x00	0x0B	0x00	0x00	0x00

Explanations on the telegram from the drive	
Identifier	= 1408 + node address = 1408 + 12 = 1420 = 0x058C (1408 = SDO1 basic identifier from the controller)
Command	= 0x41 = "Read response" (response is block telegram)
Index	As in telegram to the drive
Subindex	
Data 1 ... 4	= 0x0000000B = data length of 11 characters in the ASCII format

Telegram 2 to the drive: Request of the 1st data block

Identifier	User data							
	1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
	Command	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7
0x060C	0x60	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Explanations on the telegram to the drive	
Command	= 0x60 = "Read segment request" (request: read data block) • Bit 4 = 0 (toggle bit)
	Influence of the toggle bit on the request command The blocks are toggled one after another, i.e. the request ist made with the "0x60" (= 011**0***0000_{bin}) command, then with the "0x70" (= 011**1***0000_{bin}) command, and then again with the "0x60" command, etc. * Toggle bit

Response message 2 from the drive: Transmission of the 1st data block

Identifier	User data							
	1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
	Command	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7
0x058C	0x00	0x30	0x31	0x2E	0x30	0x30	0x2E	0x30
		0_{asc}	1_{asc}	$._{asc}$	0_{asc}	0_{asc}	$._{asc}$	0_{asc}

Explanations on the telegram to the drive	
Command	= 0x00 = 00000000_{bin} • Bit 4 = 0 (toggle bit)
	Influence of the toggle bit on the transmission command • The 1st response of the controller in the command byte is "0x000**0***0000_{bin}" if bytes 2 ... 8 are completely filled with data and other telegrams are following. • The 2nd response of the controller in the command byte is "0x000**1***0000_{bin}" if bytes 2 ... 8 are completely filled with data and other telegrams are following, etc. * Toggle bit
Data 1 ... 7	= "01.00.0" (ASCII representation)

Telegram 3 to the drive: Request of the 2nd data block

Identifier	User data							
	1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
	Command	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7
0x060C	0x70	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Explanations on telegram 3 to the drive	
Command	= 0x70 = "Read segment request" (request: read data block) • Bit 4 = 1 (toggle bit)

Response message 3 from the drive: Transmission of the 2nd data block including end identifier

Identifier	User data							
	1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
	Command	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7
0x058C	0x17	0x30	0x2E	0x30	0x30	0x00	0x00	0x00
		0_{asc}	$\cdot_{asc}$	0_{asc}	0_{asc}	-	-	-

Explanations on telegram 3 from the drive	
Command	= 0x17 = 00010111_{bin}: • Bit 0 = 1 (end of transmission) • Bit 1 ... bit 3 = 011_{bin} (3 bytes do not contain any data) • Bit 4 = 1 (toggle bit)
	Influence of the final bit and the residual data length on the transmission command • The end of transmission is signalled via the set final bit 0. • Bits 1 ... 3 reveal the number of bytes that do not contain data anymore. * Toggle bit
Data 1 ... 7	= "0.00" (ASCII representation) The result of the data block transmission is: "01.00.00.00"

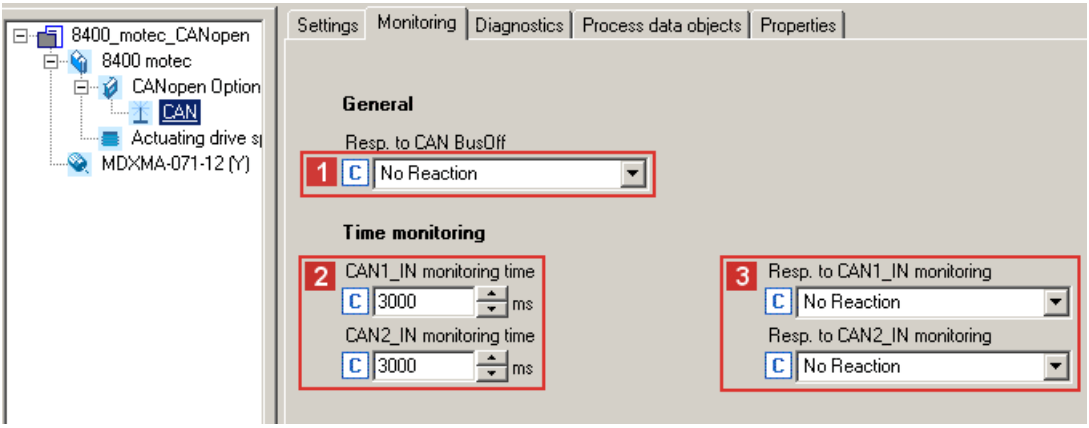
10 Monitoring

10.1 Monitoring of the RPDOs for data reception

RPDO1 and RPDO2 have a parameterisable monitoring time within which the RPDO must arrive.

The following settings can be made in the »Engineer« under the **Monitoring** tab:

- Response to "BusOff" (bus system switched off), 1 C00592/2
- CAN1_IN monitoring time, 2 C00357/1
- CAN2_IN monitoring time, 2 C00357/2
- Response to CAN1_IN monitoring, 3 C00593/1
- Response to CAN2_IN monitoring, 3 C00593/2



If a monitoring time > 0 ms 2 (C00357/1...2) is entered for CAN1_IN/CAN2_IN, the RPDO is expected after the set time has expired.

If the RPDO is not received within the monitoring time or with the configured sync, the response set for the respective RPDO is effected 3 (C00593/1...2).

A monitoring time = 0 ms deactivates the monitoring function.

10.2 Integrated error detection

If a node detects an error, it rejects the CAN telegram bits received so far and transmits an error flag. The error flag consists of 6 consecutive bits with the same logic value.

The following errors are detected:

Bit error

The sending node monitors the bus and interrupts the transmission if it receives a different logic value than the value transmitted. With the next bit, the sending node starts the transmission of an error flag.

In the arbitration phase, the sender only detects a bit error if a dominantly sent bit is received as a recessive bit. In the ACK slot as well, the dominant overwriting of a recessive bit is not indicated as a bit error.

Stuff-bit error

If more than 5 consecutive bits before the ACK delimiter in the CAN telegram have the same logic value, the previously transmitted telegram will be rejected and an error flag will be sent with the next bit.

CRC error

If the CRC checksum received does not correspond to the checksum calculated in the CAN chip, the CAN controller sends an error flag after the ACK delimiter, and the previously transmitted telegram is invalidated.

Acknowledgement error

If the ACK slot which is sent recessively by the transmitting node is not overwritten dominantly by a receiver, the transmitting node aborts the transmission. The transmitting node invalidates the telegram transmitted and sends an error flag with the next bit.

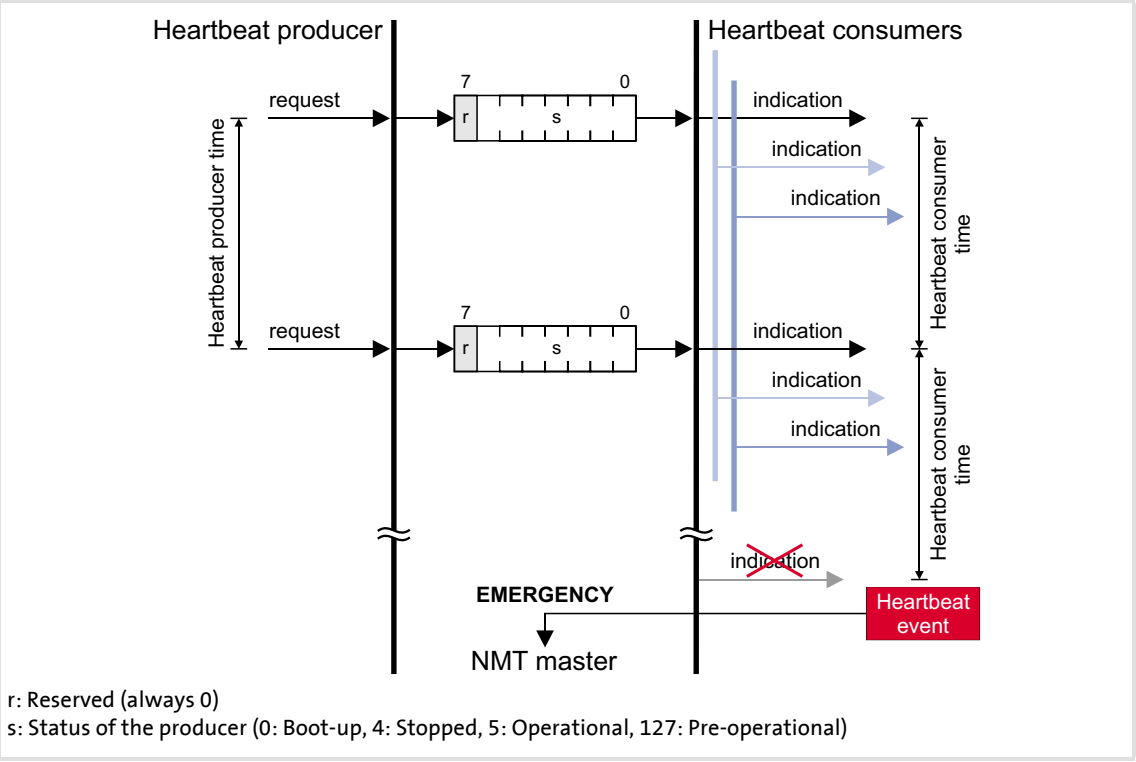
Format error

If a dominant bit is detected in the CRC delimiter, in the ACK delimiter or in the first 6 bits of the EOF field, the telegram received is rejected and an error flag is sent with the next bit.

10.3 Heartbeat protocol

The heartbeat protocol can be used for node monitoring purposes within a CAN network.

Basic workflow



r: Reserved (always 0)
s: Status of the producer (0: Boot-up, 4: Stopped, 5: Operational, 127: Pre-operational)

[10-1] Heartbeat protocol

1. A heartbeat producer cyclically transmits a so-called heartbeat telegram to one or more consumers.
2. The consumer(s) monitor(s) the heartbeat telegram for arrival on a regular basis.

10.3.1 Telegram structure

- The heartbeat telegram of the producer has the following identifier:
  Identifier (COB-ID) = 1792 + producer's node address
- The user data (1 byte) contain the status (s) of the producer:

Heartbeat producer status		Data							
Communication status	Decimal value (s)	(r)	Producer status (s)						
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Boot-up	0	0	0	0	0	0	0	0	0
Stopped	4	0	0	0	0	0	1	0	0
Operational	5	0	0	0	0	0	1	0	1
Pre-operational	127	0	1	1	1	1	1	1	1

10.3.2 Parameter setting

Short overview of the parameters for the "Heartbeat" monitoring function:

Parameter	Info	Lenze setting		Assignment	
		Value	Unit	Consumer	Producer
C00347/1...n	CAN status of heartbeat producer 1	-		●	
C00381	Heartbeat producer time	0	ms		●
C00385/1...n	CAN node address of heartbeat producer 1	0		●	
C00386/1...n	Heartbeat consumer time for heartbeat producer 1	0	ms	●	
C00592/5	Resp. to heartbeat event	No response		●	
Highlighted in grey = display parameter					

Heartbeat producer time

Time interval for the transmission of the heartbeat telegram to the consumer(s).

- Parameterisable in C00381 or via object I-1017. The parameterised time is rounded down to an integer multiple of 5 ms.
- The heartbeat telegram is sent automatically as soon as a time > 0 ms is set.

Heartbeat consumer time

Monitoring time for the nodes (producers) to be monitored.

- Parameterisable in C00386/1...n or via object I-1016.
- The parameterised time is rounded down to an integer multiple of 5 ms and must have a greater value than the heartbeat producer time of the node to be monitored.
- 1 Heartbeat Producer can be monitored.
- The node address(es) of the nodes to be monitored is/are set in C00385/1...n or via object I-1016, too.

Heartbeat event

The "Heartbeat event" is activated in the consumer if it does not receive any heartbeat telegram from the producer within the heartbeat consumer time:

- The consumer changes from the "Operational" communication status to the "Pre-operational" communication status.
- The NMT master receives an emergency telegram containing emergengy error code 0x8130.
- The response parameterised in C00592/5 is activated (Lenze setting: "No response").

Note!

The heartbeat monitoring will not start until the first heartbeat telegram of a monitored producer has been received successfully and the "Pre-operational" NMT status has been achieved.

The boot-up telegram counts as the first heartbeat telegram.

10.3.3 Commissioning example

Task

A controller (node 2) which is configured as heartbeat consumer is to monitor another controller (heartbeat producer, node 1).

- The heartbeat producer is to transmit a heartbeat telegram to the heartbeat consumer every 10 ms.
- The heartbeat consumer monitors the heartbeat telegram for arrival. A response is to be activated in the event of an error.

Parameterising the heartbeat producer (node 1)

1. Set the heartbeat producer time (C00381) to 10 ms.

Parameterising the heartbeat consumer (node 2)

1. Set the CAN node address of the producer in C00385/1.
2. Set the heartbeat consumer time in C00386/1.
   - Note: The heartbeat consumer time must be greater than the heartbeat producer time of the node to be monitored set in C00381.
3. Set the desired response in C00592/5 which is to be activated if a heartbeat event in the consumer occurs.

Tip!

C00347/1...n displays the heartbeat status of the nodes monitored.

Heartbeat telegram

- The heartbeat telegram from the producer has the following identifier:
  Identifier (COB-ID) = 1792 + producer node address = 1792 + 1 = 1793 = 0x701

10.4 Emergency telegram

If the error status changes because an internal device error occurs or has been eliminated, the NMT master once receives an emergency telegram with the following structure:

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Emergency error code		Error register	Manufacturer-specific error message				
Low byte	High byte	I-1001	0x00 Reserved	Low word		High word	
				Low byte	High byte	Low byte	High byte
See table below			• For emergency error code 0xF000: Lenze error number (value displayed in C00168) • All other emergency error codes have a value of "0" here.				

Emergency error code	Error register	Cause
0x0000	0xXX	One of several errors eliminated
	0x00	One error has been eliminated (error-free status afterwards)
0x3100	0x01	Supply voltage of standard device faulty or failed
0x8100	0x11	Communication error (warning)
0x8130	0x11	Life guarding error or heartbeat error
0x8150	0x11	Collision of identifiers (COB-IDs): An identifier parameterised for reception is also used for transmission.
0x8210	0x11	PDO length shorter than expected
0x8220	0x11	PDO length greater than expected
0x8700	0x11	Monitoring of the sync telegram
0xF000	0x01	Generic error • An error with a "Fault", "Trouble", "TroubleQSP", "Warning", or "SystemFault" error response occurred in the standard device. • Error message is the Lenze error number (C00168).

More emergency error codes are listed in the short overview of the error messages of the operating system in the software manual/»Engineer« online help "Inverter Drives 8400 motec".

Example

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Emergency error code		Error register	Manufacturer-specific error message				
0x00	0xF0	0x01	0x00 Reserved	Lenze error number			
Generic error				Corresponding error-free message: Value "0x00000000"			

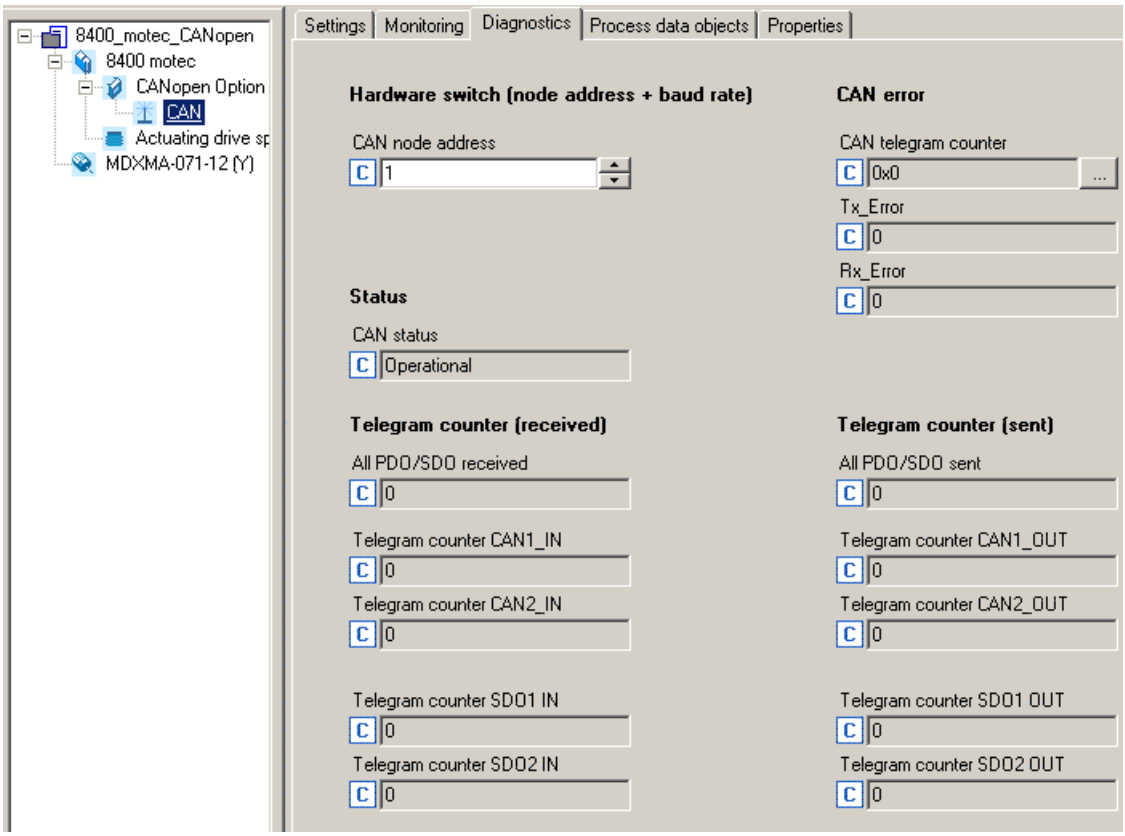
Tip!

A detalied description can be found in CAN specification DS301, V4.02.

11 Diagnostics

Diagnostics with the »Engineer«

In the »Engineer« under the **Diagnostics** tab, various system bus (CANopen) diagnostics information is displayed.



12 Parameter reference

This chapter complements the parameter list and table of attributes in the software manual and the »Engineer« online help for the Inverter Drive 8400 motec by the parameters for CANopen communication.

Software manual/»Engineer« online help "Inverter Drives 8400 motec"

Here you can find general information on parameters.

12.1 Communication-relevant parameters of the operating system

This chapter lists communication-relevant parameters of the 8400 motec operating system in numerically ascending order.

C01501

Parameter | Name: **C01501 | Resp. to communication error with MCI**		Data type: UNSIGNED_8 Index: $23074_d = 5A22_h$
Configuration of monitoring modes for the communication unit		
Selection list		
0	No Reaction	
1	Fault	
4	WarningLocked	
Subcodes	**Lenze setting**	**Info**
C01501/1	1: Fault	Resp. to MCI error 1 • Response to a communication error.
C01501/2	1: Fault	Resp. to MCI error 2 • Response to an incompatible communication unit.

☑ Read access ☑ Write access ☐ CINH ☐ PLC STOP ☐ No transfer ☐ COM ☐ MOT Scaling factor: 1

C01503

Parameter | Name: **C01503 | MCI timeout**			Data type: UNSIGNED_16 Index: $23072_d = 5A20_h$
Setting range (min. value | unit | max. value)			
0	ms	1000	
Subcodes	**Lenze setting**		**Info**
C01503/1	200 ms		MCI timeout

☑ Read access ☑ Write access ☐ CINH ☐ PLC STOP ☐ No transfer ☐ COM ☐ MOT Scaling factor: 1

12.2 Parameters for CANopen communication

This chapter lists the CANopen parameters of the communication unit in numerically ascending order.

C00322

Parameter | Name:
C00322 | Transmission mode CAN TxPDOs

Data type: UNSIGNED_8
Index: $24253_d = 5EBD_h$

TPDO transmission type according to DS301 V4.02
- The following transmission modes are supported:
  – 0: Synchronous and acyclic
  – 1 ... 240: Synchronous and cyclic
  – 252: Synchronous - only RTR
  – 253: Asynchronous - only RTR
  – 254: Asynchronous - manufacturer-specific
  – 255: Asynchronous - device profile-specific
- The basic setting for all PDOs is the "asychronous - manufacturer-specific" setting (254).
- Mapping of the CANopen objects I-1800/2 and I-1801/2 (see DS301 V4.02).

Setting range (min. value | unit | max. value)		
0		255

Subcodes	Lenze setting	Info
C00322/1	254	Transmission mode CAN1 OUT
C00322/2	254	Transmission mode CAN2 OUT

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00323

Parameter | Name:
C00323 | Transmission mode CAN Rx PDOs

Data type: UNSIGNED_8
Index: $24252_d = 5EBC_h$

RPDO transmission type according to DS301 V4.02
- For the RPDO, it serves as monitoring setting in the case of sync-controlled PDOs.
- The following transmission modes are supported:
  – 0: Synchronous and acyclic
  – 1 ... 240: Synchronous and cyclic
  – 252: Synchronous - only RTR
  – 253: Asynchronous - only RTR
  – 254: Asynchronous - manufacturer-specific
  – 255: Asynchronous - device profile-specific
- The basic setting for all PDOs is the "asychronous - manufacturer-specific" setting (254).
- Mapping of the CANopen objects I-1400/2 and I-1401/2 (see DS301 V4.02).

Setting range (min. value | unit | max. value)		
0		255

Subcodes	Lenze setting	Info
C00323/1	254	Transmission mode CAN1 IN
C00323/2	254	Transmission mode CAN2 IN

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00324

Parameter | Name:
C00324 | CAN Tx inhibit time

Data type: UNSIGNED_16
Index: $24251_d = 5EBB_h$

Inhibit time for the transmission of the emergency telegram and the process data

Note:
If the "Asynchronous - manufacturer-specific/device profile-specific" transmission time is set, the transmission cycle timer is reset to 0 if the transmission has been triggered in an event-controlled manner.
Example: Cycle time (C00356/x) = 500 ms, inhibit time = 100 ms, data change sporadically:

- In the case of a sporadical data change < 500 ms, due to the inhibit time set, transmission takes place as quickly as possible every 100 ms (event-controlled transmission).
- In the case of a sporadical data change > 500 ms, due to the cycle time set, transmission takes place every 500 ms (cyclic transmission).
- Mapping of the CANopen objects I-1800/3 and I-1801/3 (see DS301 V4.02).

Setting range (min. value | unit | max. value)		
0	ms	6500

Subcodes	Lenze setting	Info
C00324/1	0 ms	Inhibit time for emergency telegrams
C00324/2	0 ms	CAN1_OUT inhibit time
C00324/3	0 ms	CAN2_OUT inhibit time

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00345

Parameter | Name:
C00345 | CAN error status

Data type: UNSIGNED_8
Index: $24230_d = 5EA6_h$

Display of the CAN error status

Selection list (read only)	
0	No Error
1	Warning ErrActive
2	Warning ErrPassive
3	Bus off
4	Reserved
5	Reserved

☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☑ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☐ COM ☐ MOT

C00347

Parameter | Name:
C00347 | CAN status HeartBeat producer

Data type: UNSIGNED_8
Index: $24228_d = 5EA4_h$

Display of the heartbeat producer's CAN status
▸ Heartbeat protocol (📖 69)

Selection list	
0	Boot-up
4	Stopped
5	Operational
127	Pre-operational
250	Failed
255	NoResponse

Subcodes	Info
C00347/1	Status node 1

☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☑ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00349

Parameter | Name:
C00349 | CAN setting, DIP switch

Data type: UNSIGNED_16
Index: $24226_d = 5EA2_h$

Display of the DIP switch setting at the last mains connection
▸ Possible settings via DIP switch (📖 32)

Display area (min. hex value | max. hex value)		
0x0000		0xFFFF

Value is bit-coded:	
Bit 0	Node address 1
Bit 1	Node address 2
Bit 2	Node address 4
Bit 3	Node address 8
Bit 4	Node address 16
Bit 5	Node address 32
Bit 6	Node address 64
Bit 7	Baud rate 1
Bit 8	Baud rate 2
Bit 9	Baud rate 4
Bit 10	Reserved
...	...
Bit 14	Reserved
Bit 15	Accept DIP switch at 24V-ON

☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☑ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☐ COM ☐ MOT

C00350

Parameter | Name:
C00350 | CAN node address

Data type: UNSIGNED_8
Index: $24225_d = 5EA1_h$

Setting of the node address via parameters
- The node address can only be parameterised if the node address "0" is set via the DIP switches.
- A change of the node address will only become effective after a CAN reset node.

▸ Setting the CAN node address (📖 33)

Setting range (min. value | unit | max. value)			Lenze setting
1		63	**1**

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00351

Parameter | Name:
C00351 | CAN baud rate

Data type: UNSIGNED_8
Index: $24224_d = 5EA0_h$

Setting of the baud rate via parameters
- The baud rate can only be parameterised if the baud rate "0" is set via the DIP switches.
- A change of the baud rate will only become effective after a CAN reset node.

▸ Setting the baud rate (📖 32)

Selection list (Lenze setting printed in bold)	
0	**500 kbps**
1	250 kbps
2	125 kbps
3	50 kbps
4	1000 kbps
5	20 kbps
14	800 kbps

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00352

Parameter | Name:
C00352 | CAN Slave/Master

Data type: UNSIGNED_8
Index: $24223_d = 5E9F_h$

The drive starts as CAN master after mains switching if the value "1" has been entered and saved here.

Selection list (Lenze setting printed in bold)	
0	**Slave**
1	Master

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00353

Parameter | Name:
C00353 | CAN IN/OUT COBID source

Data type: UNSIGNED_8
Index: $24222_d = 5E9E_h$

Identifier allocation procedure for the CANx_IN/OUT process data

Selection list		Info
0	COBID = C0350 + LenzeBaseID	COBID = device address + LenzeBaseID
1	COBID = C0350 + CANBaseID	COBID = device address + CANBaseID (C00354/x)
2	COBID = C0354/x	COBID = direct setting from C00354/x

Subcodes	Lenze setting	Info
C00353/1	1	COBID source CAN1_IN/OUT
C00353/2	1	COBID source CAN2_IN/OUT

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00354

Parameter | Name:
C00354 | COBID

Data type: UNSIGNED_32
Index: $24221_d = 5E9D_h$

Setting of the default COBID according to CANopen
- A change of the COBID will only become effective after a CAN reset node.

▸ Identifiers of the process data objects (📖 51)

Value is bit-coded:	
Bit 0	COBID Bit0
...	...
Bit 10	COBID Bit10
Bit 11	Reserved
...	...
Bit 30	Reserved
Bit 31	PDO invalid

Subcodes	Lenze setting	Info
C00354/1	513 (0x00000201)	COBID CAN1_IN
C00354/2	385 (0x00000181)	COBID CAN1_OUT
C00354/3	769 (0x00000301)	COBID CAN2_IN
C00354/4	641 (0x00000281)	COBID CAN2_OUT

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00355

Parameter | Name:
C00355 | Active COBID

Data type: UNSIGNED_16
Index: $24220_d = 5E9C_h$

Display of the COBID of the PDOs that is active in the CAN stack

▸ Identifiers of the process data objects (📖 51)

Display area (min. value | unit | max. value)		
0		2047

Subcodes	Info
C00355/1	Active COBID CAN1_IN
C00355/2	Active COBID CAN1_OUT
C00355/3	Active COBID CAN2_IN
C00355/4	Active COBID CAN2_OUT

☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☑ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00356

Parameter | Name:
C00356 | CAN time settings

Data type: UNSIGNED_16
Index: $24219_d = 5E9B_h$

Different time settings for the CAN interface

Setting range (min. value | unit | max. value)		
0	ms	65000

Subcodes	Lenze setting	Info
C00356/1	3000 ms	CAN delay during status change from "Boot-up" to "Operational"
C00356/2	0 ms	CAN2_OUT cycle time
C00356/3	0 ms	Reserved
C00356/4	0 ms	CANx_OUT time "Operational" to "First transmission"
C00356/5	0 ms	CAN1_OUT cycle time

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00357

Parameter | Name:
C00357 | CAN monitoring times

Data type: UNSIGNED_16
Index: $24218_d = 5E9A_h$

Mapping of the RPDO event time (see DS301 V4.02)
- If a non-zero value is entered, the RPDO is expected after the time set has elapsed.
- If the RPDO is not received within the expected time, the response set in C00593/1...2 is effected.

Setting range (min. value | unit | max. value)		
0	ms	65000

Subcodes	Lenze setting	Info
C00357/1	3000 ms	CAN1_IN monitoring time
C00357/2	3000 ms	CAN2_IN monitoring time

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00359

Parameter | Name:
C00359 | CAN status

Data type: UNSIGNED_8
Index: $24216_d = 5E98_h$

Display of the CAN status
▸ Communication phases/network management (📖 40)

Selection list (read only)	
0	Operational
1	Pre-operational
2	Reserved
3	Reserved
4	BootUp
5	Stopped
6	Reserved
7	Reset

☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☑ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00360

Parameter | Name: **C00360 | CAN telegram counter**			Data type: UNSIGNED_16 Index: 24215_d = $5E97_h$
Number of received and sent CAN telegrams			
Display area (min. value | unit | max. value)			
0		65535	

Subcodes	Info
C00360/1	All PDOs/SDOs sent
C00360/2	All PDOs/SDOs received
C00360/3	Telegram counter CAN1_OUT
C00360/4	Telegram counter CAN2_OUT
C00360/5	Reserved
C00360/6	Telegram counter SDO1 OUT
C00360/7	Telegram counter SDO2 OUT
C00360/8	Telegram counter CAN1_IN
C00360/9	Telegram counter CAN2_IN
C00360/10	Reserved
C00360/11	Telegram counter SDO1 IN
C00360/12	Telegram counter SDO2 IN

☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☑ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00364

Parameter | Name: **C00364 | CAN MessageError**		Data type: UNSIGNED_8 Index: 24211_d = $5E93_h$
Value is bit-coded:		
Bit 0	No Error	
Bit 1	StuffError	
Bit 2	FormError	
Bit 3	AckError	
Bit 4	Bit1Error	
Bit 5	Bit0Error	
Bit 6	CRCError	
Bit 7	Reserved	

☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☑ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00366

Parameter | Name:
C00366 | Number of CAN SDO channels

Data type: UNSIGNED_8
Index: $24209_d = 5E91_h$

Available from firmware version 02.00.

Selection of the number of active parameter data channels

- In the Lenze setting in accordance with CANopen, only parameter data channel 1 is activated. To activate both parameter data channels, set the selection "2 SDO Lenze".
- Mapping of the CANopen object I-1201 (see DS301 V4.02)

Selection list (Lenze setting printed in bold)		Info
0	**1 SDO CANopen**	I-1201 • Subindex1.Bit31 = 1 (client -> server (rx)) • Subindex2.Bit31 = 1 (server -> client (tx)) • Bit 31 = 1 (SDO invalid/not available)
1	2 SDO Lenze	I-1201 • Subindex1.Bit31 = 0 (client -> server (rx)) • Subindex2.Bit31 = 0 (server -> client (tx)) • Bit 31 = 1 (SDO valid/available)

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00367

Parameter | Name:
C00367 | CAN sync-Rx identifier

Data type: UNSIGNED_16
Index: $24208_d = 5E90_h$

Identifier by means of which the sync slave is to receive sync telegrams.

- Mapping of the CANopen object I-1005 (see DS301 V4.02).

▸ PDO synchronisation via sync telegram (📖 54)

Setting range (min. value | unit | max. value)			Lenze setting
128		255	**128**

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00368

Parameter | Name:
C00368 | CAN sync-Tx identifier

Data type: UNSIGNED_16
Index: $24207_d = 5E8F_h$

Identifier by means of which the sync master is to transmit sync telegrams.

- Mapping of the CANopen object I-1005 (see DS301 V4.02).

▸ PDO synchronisation via sync telegram (📖 54)

Setting range (min. value | unit | max. value)			Lenze setting
128		255	**128**

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00369

Parameter | Name:
C00369 | CAN sync transmission cycle time

Data type: UNSIGNED_16
Index: $24206_d = 5E8E_h$

Cycle during which the sync master is to transmit sync telegrams.

- If "0 ms" is set (Lenze setting), no sync telegrams are generated.
- Mapping of the CANopen object I-1006 (see DS301 V4.02).

▸ PDO synchronisation via sync telegram (📖 54)

Setting range (min. value | unit | max. value)			Lenze setting
0	ms	65000	**0 ms**

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00372

Parameter | Name:
C00372 | CAN_Tx_Rx_Error

Data type: UNSIGNED_8
Index: $24203_d = 5E8B_h$

Display of CAN transmission and reception errors

Display area (min. value | unit | max. value)		
0	ms	255

Subcodes	Info
C00372/1	Transmission error (Tx_Error)
C00372/2	Receipt error (Rx_Error)

☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☑ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00381

Parameter | Name:
C00381 | CAN Heartbeat Producer Time

Data type: UNSIGNED_16
Index: $24194_d = 5E82_h$

Time interval for the transmission of the heartbeat telegram to the consumer(s).
- The heartbeat telegram is sent automatically as soon as a time > 0 ms is set.
- Mapping of the CANopen object I-1017 (see DS301 V4.02).

▸ Heartbeat protocol (📖 69)

Setting range (min. value | unit | max. value)			Lenze setting
0	ms	65535	**0 ms**

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00385

Parameter | Name:
C00385 | CAN NodeID heartbeat producer

Data type: UNSIGNED_8
Index: $24190_d = 5E7E_h$

Subcode 1 represents the node which is to be monitored via heartbeat.

▸ Heartbeat protocol (📖 69)

Setting range (min. value | unit | max. value)		
0		127

Subcodes	Lenze setting	Info
C00385/1	0	CAN NodeID heartbeat producer 1

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☑ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00386

Parameter | Name:
C00386 | ConsumerTime HeartBeat producer

Data type: UNSIGNED_16
Index: $24189_d = 5E7D_h$

Monitoring time for the nodes to be monitored
- Mapping of the CANopen object I-1016 (see DS301 V4.02).

▸ Heartbeat protocol (📖 69)

Setting range (min. value | unit | max. value)		
0	ms	60000

Subcodes	Lenze setting	Info
C00386/1	0 ms	ConsumerTime HeartBeat Producer 1

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00389

Parameter | Name:
C00389 | PDO valid / not valid

Data type: UNSIGNED_8
Index: $24186_d = 5E7A_h$

Validity of the PDOs

Selection list (Lenze setting printed in bold)	
0	**PDO available/valid**
1	PDO not available/invalid

Subcodes	Lenze setting	Info
C00389/1	0	PDO valid / invalid CAN1_IN
C00389/2	0	PDO valid / invalid CAN1_OUT
C00389/3	0	PDO valid / invalid CAN2_IN
C00389/4	0	PDO valid / invalid CAN2_OUT

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☑ COM ☐ MOT

C00409

Parameter | Name:
C00409 | LP_CanIn Mapping

Data type: UNSIGNED_16
Index: $24166_d = 5E66_h$

Mapping for the port blocks LP_CanIn1...2
- Mapping of the CANopen objects I-1600 ... I-1601 (see DS301 V4.02)

Setting range (min. value | unit | max. value)		
0		65535

Subcodes	Lenze setting	Info
C00409/1	0	LP_CanIn1_wIn1(wCtrl)
C00409/2	0	LP_CanIn1_wIn2
C00409/3	0	LP_CanIn1_wIn3
C00409/4	0	LP_CanIn1_wIn4
C00409/5	0	LP_CanIn2_wIn1
C00409/6	0	LP_CanIn2_wIn2
C00409/7	0	LP_CanIn2_wIn3
C00409/8	0	LP_CanIn2_wIn4

☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☑ PDO_MAP_RX ☐ PDO_MAP_TX ☐ COM ☐ MOT

C00592

Parameter | Name: **C00592 | Resp. to CAN bus connection**		Data type: UNSIGNED_8 Index: $23983_d = 5DAF_h$
Configuration of monitoring of the CAN interface		
Selection list		
0	No Reaction	
1	Fault	
2	Trouble	
4	WarningLocked	
Subcodes	**Lenze setting**	**Info**
C00592/1	0: No Reaction	Response to an incorrect telegram during CAN communication
C00592/2	0: No Reaction	Response to "BusOff" (bus system switched off)
C00592/3	0: No Reaction	Response to warnings of the CAN controller
C00592/4	0: No Reaction	Response to communication stop of a CAN bus node
C00592/5	0: No Reaction	Response to an event in the case of monitoring via heartbeat protocol
☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☐ COM ☐ MOT		

C00593

Parameter | Name: **C00593 | Resp. to CANx_IN monitoring**		Data type: UNSIGNED_8 Index: $23982_d = 5DAE_h$
Configuration of monitoring for the reception of PDOs CAN1_IN and CAN2_IN		
Selection list		
0	No Reaction	
1	Fault	
2	Trouble	
4	WarningLocked	
Subcodes	**Lenze setting**	**Info**
C00593/1	0: No Reaction	Response if the monitoring time set in C00357/1 for the reception of the PDO CAN1_IN is exceeded.
C00593/2	0: No Reaction	Response if the monitoring time set in C00357/2 for the reception of the PDO CAN2_IN is exceeded.
☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer ☐ PDO_MAP_RX ☐ PDO_MAP_TX ☐ COM ☐ MOT		

12.3 Table of attributes

How to read the table of attributes:

Column		Meaning	Entry	
Code		Parameter name	Cxxxxx	
Name		Parameter short text (display text)	*Text*	
Index	dec	Index under which the parameter is addressed. The subindex for array variables corresponds to the Lenze subcode number.	24575 - Lenze code number	Is only required for access via a bus system.
	hex		$5FFF_h$ - Lenze code number	
Data	DS	Data structure	E	Single variable (only one parameter element)
			A	Array variable (several parameter elements)
	DA	Number of array elements (subcodes)	*Number*	
	DT	Data type	BITFIELD_8	1 byte, bit-coded
			BITFIELD_16	2 bytes bit-coded
			BITFIELD_32	4 bytes, bit-coded
			INTEGER_8	1 byte with sign
			INTEGER_16	2 bytes with sign
			INTEGER_32	4 bytes with sign
			UNSIGNED_8	1 byte without sign
			UNSIGNED_16	2 bytes without sign
			UNSIGNED_32	4 bytes, without sign
			VISIBLE_STRING	ASCII string
			OCTET_STRING	
	Factor	Factor for data transmission via a bus system, depending on the number of decimal positions	*Factor*	1 = no decimal positions 10 = 1 decimal position 100 = 2 decimal positions 1000 = 3 decimal positions
Access	R	Read access	☑ Reading permitted	
	W	Write access	☑ Writing permitted	
	CINH	Controller inhibit required	☑ Writing is only possible if controller inhibit is set	

Table of attributes

Code	Name	Index		Data				Access		
		dec	hex	DS	DA	Data type	Factor	R	W	CINH
C00322	Transmission mode CAN TxPDOs	24253	5EBD	A	2	UNSIGNED_8	1	☑	☑	
C00323	Transmission mode CAN Rx PDOs	24252	5EBC	A	2	UNSIGNED_8	1	☑	☑	
C00324	CAN Tx inhibit time	24251	5EBB	A	3	UNSIGNED_16	1	☑	☑	
C00345	CAN error status	24230	5EA6	E	1	UNSIGNED_8	1	☑		
C00347	CAN status HeartBeat producer	24228	5EA4	A	1	UNSIGNED_8	1	☑		
C00349	CAN setting - DIP switch	24226	5EA2	E	1	UNSIGNED_16		☑		
C00350	CAN node address	24225	5EA1	E	1	UNSIGNED_8	1	☑	☑	
C00351	CAN baud rate	24224	5EA0	E	1	UNSIGNED_8	1	☑	☑	
C00352	CAN slave/master	24223	5E9F	E	1	UNSIGNED_8	1	☑	☑	
C00353	CAN IN/OUT COBID source	24222	5E9E	A	2	UNSIGNED_8	1	☑	☑	
C00354	COBID	24221	5E9D	A	4	UNSIGNED_32		☑	☑	
C00355	Active COBID	24220	5E9C	A	4	UNSIGNED_16	1	☑		
C00356	CAN time settings	24219	5E9B	A	5	UNSIGNED_16	1	☑	☑	
C00357	CAN monitoring times	24218	5E9A	A	2	UNSIGNED_16	1	☑	☑	
C00359	CAN status	24216	5E98	E	1	UNSIGNED_8	1	☑		
C00360	CAN telegram counter	24215	5E97	A	12	UNSIGNED_16	1	☑		
C00364	CAN MessageError	24211	5E93	E	1	UNSIGNED_8		☑		
C00366	Number of CAN SDO channels	24209	5E91	E	1	UNSIGNED_8	1	☑	☑	
C00367	CAN sync Rx identifier	24208	5E90	E	1	UNSIGNED_16	1	☑	☑	
C00368	CAN sync Tx identifier	24207	5E8F	E	1	UNSIGNED_16	1	☑	☑	
C00369	CAN sync transmission cycle time	24206	5E8E	E	1	UNSIGNED_16	1	☑	☑	
C00372	CAN_Tx_Rx_Error	24203	5E8B	A	2	UNSIGNED_8	1	☑		
C00381	CAN heartbeat producer time	24194	5E82	E	1	UNSIGNED_16	1	☑	☑	
C00385	CAN NodeID Heartbeat producer	24190	5E7E	A	1	UNSIGNED_8	1	☑	☑	
C00386	ConsumerTime HeartBeat Producer	24189	5E7D	A	1	UNSIGNED_16	1	☑	☑	
C00389	PDO valid / invalid	24186	5E7A	A	4	UNSIGNED_8	1	☑	☑	
C00409	LP_CanIn mapping	24166	5E66	A	8	UNSIGNED_16	1	☑	☑	
C00592	Resp. to CAN bus connection	23983	5DAF	A	5	UNSIGNED_8	1	☑	☑	
C00593	Resp. to CANx_IN monitoring	23982	5DAE	A	2	UNSIGNED_8	1	☑	☑	

13 Implemented CANopen objects

Lenze devices can be parameterised with both Lenze codes and manufacturer-independent "CANopen objects". Completely CANopen-compliant communication can only be achieved by solely using CANopen objects for parameter setting. The CANopen objects described in this chapter are defined in the CAN specification DS301 V4.02.

Many CANopen objects can be mapped to Lenze codes. The "Relationship to Lenze code" column of the following table lists the Lenze codes used.

Note!

Some of the terms used here derive from the CANopen protocol.

Overview of CANopen indexes and their relationship to Lenze codes

CANopen object			Relationship to Lenze code
Index	**Subindex**	**Name**	
I-1000	0	Device type	-
I-1001	0	Error register	-
I-1003	Predefined error field		
	0	Number of errors	-
	1 ... 10	Standard error field	-
I-1005	0	COB-ID SYNC message	C00367
			C00368
I-1006	0	Communication cycle period	C00369
I-1014	0	COB-ID EMCY	-
I-1016	Consumer heartbeat time		
	0	Highest subindex supported	-
	1	Consumer heartbeat time	C00385/1...n C00386/1...n
I-1017	0	Producer heartbeat time	C00381
I-1018	Identity object		
	0	Highest subindex supported	-
	1	Vendor ID	-
	2	Product code	-
	3	Revision number	-
	4	Serial number	-
I-1200	SDO1 server parameter		
	0	Highest subindex supported	-
	1	COB-ID client –> server (rx)	-
	2	COB-ID server –> client (tx)	-
I-1201	SDO2 server parameter		C00366
	0	Highest subindex supported	
	1	COB-ID client –> server (rx)	
	2	COB-ID server –> client (tx)	

CANopen object			Relationship to Lenze code
Index	**Subindex**	**Name**	
I-1400	RPDO1 communication parameter		
	0	Highest subindex supported	-
	1	COB-ID used by RPDO	C00355/1
	2	Transmission type	C00323/1
I-1401	RPDO2 communication parameter		
	0	Highest subindex supported	-
	1	COB-ID used by RPDO	C00355/3
	2	Transmission type	C00323/2
I-1600	RPDO1 mapping parameter		
	0	Number of mapped application objects in PDO	-
	1 ... 4	Application object 1 ... 4	C00409/1...4
I-1601	RPDO2 mapping parameter		
	0	Number of mapped application objects in PDO	-
	1 ... 4	Application object 1 ... 4	C00409/5...8
I-1800	TPDO1 communication parameter		
	0	Highest subindex supported	-
	1	COB-ID used by TPDO	C00355/2
	2	Transmission type	C00322/1
	3	Inhibit time	-
	5	Event timer	C00356/5 C00369
I-1801	TPDO2 communication parameter		
	0	Highest subindex supported	-
	1	COB-ID used by TPDO	C00355/4
	2	Transmission type	C00322/2
	3	Inhibit time	-
	5	Event timer	C00356/2 C00369
I-1A00	TPDO1 mapping parameter		
	0	Number of mapped application objects in PDO	-
	1 ... 4	Application object 1 ... 4	-
I-1A01	TPDO2 mapping parameter		
	0	Number of mapped application objects in PDO	-
	1 ... 4	Application object 1 ... 4	-

I-1000 - Device type

Index **I-1000**	Name: **Device type**					
Subindex	Default setting	Display range (min. value | unit | max. value)			Access	Data type
0: Device type	0	0		4294967295	ro	U32

The CANopen index I-1000 specifies the profile for this device. Furthermore, additional information defined in the device profile itself can be stored here.

8th byte	7th byte	6th byte	5th byte
Data 4	**Data 3**	**Data 2**	**Data 1**
High word		Low word	
High byte	Low byte	High byte	Low byte
Additional information		**Device profile number**	

[13-1] Data telegram assignment

In case of 8400 series controllers, the four bytes contain the following values:

- 5th and 6th byte: The data contents are 0x0000, i.e. no profile definition.
- 7th byte: The data content specifies the device type: Here the value is 0x00 for controllers.
- 8th byte: The data contents are 0x00.

The data content for the 8400 controller thus is: 00 00 00 00

I-1001 - Error register

Index: **I-1001**	Name: **Error register**					
Subindex	Default setting	Display range (min. value | unit | max. value)			Access	Data type
0: Error register	-	0		255	ro	U8

Error register

The error status in the data byte (U8) is bit-coded. The following error states are coded in the data byte (U8):

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Error status
0	0	0	0	0	0	0	0	No error
0	0	0	0	0	0	0	1	Device error message
0	0	0	1	0	0	0	1	Communication error

I-1003 - Pre-defined error field

Index: **I-1003**	Name: **Predefined error field**					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Number of errors	0	0		255	rw	U8
1 ... 10: Standard error field	-	0		4294967295	ro	U32

Error history

This object indicates that an error has occurred in the module and in the standard device.

Subindex	Meaning
0	Number of saved error messages
1 ... 10	Display of the error list The error messages (U32) consist of a 16-bit error code and a manufacturer-specific information field comprising 16 bits.

Note!

The values of the "Standard error field" in subindex 1 ... 10 will be deleted if the "Number of recorded errors" subindex is overwritten with a value of "0".

Emergency Error code	Cause	Entry in the Error register (I-1001)
0x0000	One of several errors eliminated	0xXX
	Elimination of one single error (afterwards no more errors)	0x00
0x1000	Standard device is in error status (error response "fault", "message", "warning", "error", "quick stop by trouble", or "system error")	0x01
0x3100	Supply voltage of standard device faulty or failed	0x01
0x8100	Communication error (warning)	0x11
0x8130	Life guard error or heartbeat error	0x11
0x8150	Collision of COB IDs: An ID parameterised for reception is also used for transmission.	0x11
0x8210	PDO length shorter than expected	0x11
0x8220	PDO length greater than expected	0x11
0x8700	Monitoring of the sync telegram	0x11

I-1005 - COB-ID SYNC message

Index: I-1005	Name: COB-ID SYNC message					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: COB-ID SYNC message	0x0000 0080 or 0x8000 0080	0		4294967295	rw	U32

This object can be used to activate the generation of sync telegrams and to write the identifier value.

This object relates to codes C00367 and C00368.

Creating sync telegrams

Sync telegrams are generated by setting bit 30 (see below) to a value of "1". The time interval between the sync telegrams can be set using object I-1006.

Writing identifiers

For the reception of PDOs, the value 0x80 is entered in the Lenze setting (and according to CANopen specification) into the 11 bit identifier. This means that all modules are set to the same sync telegram by default.

- If sync telegrams are only to be received by certain communication modules, their identifiers can be entered with values up to and including 0x07FF.
- The identifier may only be changed when the communication module does not send any sync telegrams (bit 30 = "0").
- How to change the identifier:
  - Deactivate identifier (set bit 30 to "0").
  - Change identifier.
  - Activate identifier (set bit 30 to "1").

8th byte		7th byte	6th byte	5th byte
Data 4		Data 3	Data 2	Data 1
Bit 31	Bit 30	Bit 29 ... bit 11		Bit 10 ... bit 0
x	**0/1**	**Extended identifier***		**11-bit identifier**
* The extended identifier is not supported - bit 11 ... bit 29 must be set to "0".				

[13-2] Data telegram assignment

I-1006 - Communication cycle period

Index: **I-1006**	Name: **Communication cycle period**					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Communication cycle period	0 µs	0	µs	65535000	rw	U32

Setting the sync telegram cycle time.

- The cycle time can be selected as "1000" or as an integer multiple of it.
- If "0 µs" is set (Lenze setting), no sync telegrams are generated.
- This object relates to code C00369.

I-1014 - COB-ID EMCY

Index: **I-1014**	Name: **COB-ID EMCY**					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: COB-ID EMCY	0x80 + node ID	0		4294967295	rw	U32

When communication errors occur and are acknowledged or when internal errors occur in the communication module or controller (e.g. "fault"), an error message is sent on the system bus. The telegram is sent once for every error. This function can be activated or deactivated with bit 31.

8th byte		7th byte	6th byte	5th byte
Data 4		**Data 3**	**Data 2**	**Data 1**
Bit 31	Bit 30	Bit 29 ... bit 11		Bit 10 ... bit 0
0/1	**0**	**Extended identifier***		**11-bit identifier**
* The extended identifier is not supported - bit 11 ... bit 29 must be set to "0".				

[13-3] Data telegram assignment

Bit	Setting	
Bit 31	0	Emergency object is valid.
	1	Emergency object is invalid.

Note!

The identifier can only be changed in the "emergency object invalid" status (bit 31 = 1).

I-1016 - Consumer heartbeat time

Index: I-1016	Name: Consumer heartbeat time					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Highest subindex supported	1	- (read access only)			ro	U16
1 ... n: Consumer heartbeat time	0	0		65535	rw	U16

Monitoring time for the nodes to be monitored via heartbeat.

▸ Heartbeat protocol (📖 69)

The parameterised time is rounded down to an integer multiple of 5 ms and must have a greater value than the heartbeat producer time of the node to be monitored.

Subindex	Meaning	Lenze code
0	Number of nodes to be monitored	
1 ... n	Node ID and heartbeat time of the node to be monitored	Node ID: C00385/x Heartbeat time: C00386/x

8th byte	7th byte	6th byte	5th byte
Data 4	**Data 3**	**Data 2**	**Data 1**
Bit 31 ... bit 24	Bit 23 ... bit 16	Bit 15 ... bit 0	
0 Reserved	**Node ID**	**Heartbeat time** in [ms]	

[13-4] Data telegram assignment

I-1017 - Producer heartbeat time

Index: I-1017	Name: Producer heartbeat time					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Producer heartbeat time	0	0	ms	65535	rw	U16

Time interval for the transmission of the heartbeat telegram to the consumer(s).

▸ Heartbeat protocol (📖 69)

- The parameterised time is rounded down to an integer multiple of 5 ms.
- The heartbeat telegram is sent automatically as soon as a time > 0 ms is set. In this case the monitoring function "Node Guarding" is deactivated.
- This object relates to code C00381.

I-1018 - Identity object

Index: **I-1018**	Name: **Identity object**					
Subindex	Default setting	Display range (min. value | unit | max. value)			Access	Data type
0: Highest subindex supported	See below	0		4294967295	ro	U32
1: Vendor ID						
2: Product code						
3: Revision number						
4: Serial number						

Subindex	Meaning
1	Manufacturer's identification number • The identification number allocated to Lenze by the organisation "CAN in Automation e. V." is "0x0000003B".
2	Product code
3	Main version and subversion of firmware
4	Serial number

I-1200 - SDO1 server parameter

Index: **I-1200**	Name: **SDO1 server parameter**					
Subindex	Default setting	Display range (min. value | unit | max. value)			Access	Data type
0: Highest subindex supported	2	2		2	ro	U8
1: COB-ID client -> server (rx)	Node ID + 0x600	0		4294967295	ro	U32
2: COB-ID server -> client (tx)	Node ID + 0x580	0		4294967295	ro	U32

Identifiers for SDO server channel 1 (basic SDO channel).

According to DS301 V4.02, the basic SDO channel can neither be changed nor deactivated.

Subindex	Meaning
1	Specification of the receive identifier • For SDO server channel 1: node address (C00350) + 0x600
2	Specification of the transmit identifier • For SDO server channel 1: node address (C00350) + 0x580

8th byte		7th byte	6th byte	5th byte
Data 4		Data 3	Data 2	Data 1
Bit 31	Bit 30	Bit 29 ... bit 11		Bit 10 ... bit 0
0	**0**	**Extended identifier***		**11-bit identifier**
* The extended identifier is not supported - bit 11 ... bit 29 must be set to "0".				

[13-5] Data telegram assignment

I-1201 - SDO2 server parameter

Index: I-1201	Name: SDO2 server parameter					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Highest subindex supported	3	- (read access only)			ro	U8
1: COB-ID client -> server (rx)	0x80000000	0		4294967295	rw	U32
2: COB-ID server -> client (tx)	0x80000000	0		4294967295	rw	U32

Setting of the identifiers for SDO server channel 2.

- The server SDO parameter is only valid if bit 31 is set to "0" for both transmission directions (subindex 1 and 2).
- In the Lenze setting, SDO server channel 2 is deactivated (bit 31 = "1").
- The identifier may only be changed when the SDO is invalid (bit 31 = "1").

Subindex	Meaning
1	Specification of the receive identifier
2	Specification of the transmit identifier

8th byte		7th byte	6th byte	5th byte
Data 4		Data 3	Data 2	Data 1
Bit 31	Bit 30	Bit 29 ... bit 11		Bit 10 ... bit 0
0/1	**0**	**Extended identifier***		**11-bit identifier**
* The extended identifier is not supported - bit 11 ... bit 29 must be set to "0".				

[13-6] Data telegram assignment

Bit	Setting
Bit 31	0 SDO is valid.
	1 SDO is invalid.

How to change the identifier:

1. Deactivate identifier (set bit 31 to "1").
2. Change identifier.
3. Activate identifier (set bit 31 to "0").

Example Parameter data channel 2 of the controller with node address 4 is to be activated.

- For this, bit 31 must be set to the value "0" (SDO is valid) in subindexes 1 and 2 of object I-1201.
- The master must send the two "write request" commands to the nodes via the basic SDO channel.

Identifier calculation

- Identifier (COB-ID) = basic identifier + node address (node ID)
- Basic identifier SDO2 from master to drive: 1600 (0x640)
  → identifier = 0x640 + 0x4 = 0x644
- Basic identifier SDO2 from drive to master: 1472 (0x5C0)
  → identifier = 0x5C0 + 0x4 = 0x5C4

Resulting data (data 1 ... data 4)

8th byte		7th byte	6th byte	5th byte
Data 4		Data 3	Data 2	Data 1
Bit 31	Bit 30	Bit 29 ... bit 11		Bit 10 ... bit 0
0	0	Extended identifier = 0		11-bit identifier = 0x644
0x00		0x00	0x06	0x44

[13-7] Data telegram assignment for subindex 1

8th byte		7th byte	6th byte	5th byte
Data 4		Data 3	Data 2	Data 1
Bit 31	Bit 30	Bit 29 ... bit 11		Bit 10 ... bit 0
0	0	Extended identifier = 0		11-bit identifier = 0x5C4
0x00		0x00	0x05	0xC4

[13-8] Data telegram assignment for subindex 2

User data assignment

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
0x23	0x01	0x12	0x01	0x44	0x06	0x00	0x00

[13-9] User data assignment for writing to subindex 1

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Command	Index		Subindex	Data 1	Data 2	Data 3	Data 4
0x23	0x01	0x12	0x02	0xC4	0x05	0x00	0x00

[13-10] User data assignment for writing to subindex 2

I-1400 - RPDO1 communication parameter

Index: I-1400	Name: RPDO1 communication parameter					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Highest subindex supported	5	- (read access only)			ro	U8
1: COB-ID used by RPDO	0x200 + node ID	0		4294967295	rw	U32
2: Transmission type	254	0		255	rw	U8
3: Inhibit time	-	- (not used for RPDOs)			rw	U16
4: Compatibility entry	-	- (reserved, read or write access results in error message 0x06090011)			rw	U8
5: Event timer	-	- (not used for RPDOs)			rw	U16

Communication parameters for receiving process data via RPDO1

Subindex	Meaning	Code
0	The value 5 is permanently set. • Max. 5 subindexes are supported.	-
1	RPDO1 identifier • According to the "Predefined Connection Set", the basic setting is: identifier = 0x200 + node ID	C00354/1
2	RPDO transmission type according to DS301 V4.02 ▸ Transmission type (📖 52)	C00323/1

8th byte		7th byte	6th byte	5th byte
Data 4		Data 3	Data 2	Data 1
Bit 31	Bit 30	Bit 29 ... bit 11		Bit 10 ... bit 0
0/1	**0/1**	**Extended identifier***		**11-bit identifier**
* The extended identifier is not supported - bit 11 ... bit 29 must be set to "0".				

[13-11] Data telegram assignment

How to change the identifier:

1. Deactivate identifier (set bit 31 to "1").
2. Change identifier.
3. Activate identifier (set bit 31 to "0").

Description of subindex 1

Bit no.	Value	Explanation
0 ... 10	0/1	11-bit identifier
(11 ... 28)*	0	*) The extended identifier (29 bits) is not supported. Any of these bits must be "0".
29*	0	
30	0	RTR to this PDO permissible (cannot be set)
	1	RTR to this PDO not permissible (Lenze)
31	0	PDO active
	1	PDO not active

[13-12] I-1400 / I-1401, subindex 1

Description of subindex 2

PDO transmission			Transmission type	Explanation
cyclic	synchronous	event-controlled		
●	●		n = 1 ... 240	When a value n is entered, this PDO will be accepted with every nth sync.
		●	n = 254	PDO will be accepted immediately.

[13-13] I-1400 / I-1401, subindex 2

I-1401 - RPDO2 communication parameter

Index: **I-1401**	Name: **RPDO2 communication parameter**					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Highest subindex supported	5	- (read access only)			ro	U8
1: COB-ID used by RPDO	0x300 + node ID	0		4294967295	rw	U32
2: Transmission type	254	0		255	rw	U8
3: Inhibit time	-	- (not used for RPDOs)			rw	U16
4: Compatibility entry	-	- (reserved, read or write access results in error message 0x06090011)			rw	U8
5: Event timer	-	- (not used for RPDOs)			rw	U16

Communication parameters for receiving process data via RPDO2

Subindex	Meaning	Code
0	The value 5 is permanently set. • Max. 5 subindexes are supported.	-
1	RPDO2 identifier • According to the "Predefined Connection Set", the basic setting is: identifier = 0x300 + node ID	C00354/3
2	RPDO transmission type according to DS301 V4.02 ▸ Transmission type (📖 52)	C00323/2

For data telegram assignment and description of subindexes 1 and 2, see object I-1400.

How to change the identifier:

1. Deactivate identifier (set bit 31 to "1").
2. Change identifier.
3. Activate identifier (set bit 31 to "0").

I-1600 - RPDO1 mapping parameter

Index: I-1600	Name: RPDO1 mapping parameter					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Number of mapped application objects in PDO	0	0		8	rw	U8
1 ... 4: Application object 1 ... 4	0	0		4294967295	rw	U32

Object I-1600 serves to receive parameter data as RPDO1.

This object relates to code C00409/1...4.

Subindex	Meaning
0	Number of mapped objects
1 ... 4	Mapping entries 1 ... 4 for RPDO1 • The 4th mapping entry is used for the static mapping. For this, no value is available.

8th byte	7th byte	6th byte	5th byte
Data 4	**Data 3**	**Data 2**	**Data 1**
Bit 31 ... bit 16		Bit 15 ... bit 8	Bit 7 ... bit 0
Index		**Subindex**	**Length**

[13-14] Data telegram assignment

IEC 61131 process data words are mapped. Only whole bytes can be mapped (1 byte/ mapping entry).

I-1601 - RPDO2 mapping parameter

Index: I-1601	Name: RPDO2 mapping parameter					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Number of mapped application objects in PDO	0	0		8	rw	U8
1 ... 4: Application object 1 ... 4	0	0		4294967295	rw	U32

Object I-1601 serves to receive parameter data as RPDO2.

This object relates to code C00409/5...8.

Subindex	Meaning
0	Number of mapped objects
1 ... 4	Mapping entries 1 ... 4 for RPDO2 • The 4th mapping entry is used for the static mapping. For this, no value is available.

For data telegram assignment, see object I-1600.

I-1800 - TPDO1 communication parameter

Index: **I-1800**	Name: **TPDO1 communication parameter**					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Highest subindex supported	5	- (read access only)			ro	U8
1: COB-ID used by TPDO	0x180 + node ID	0		4294967295	rw	U32
2: Transmission type	254	0		255	rw	U8
3: Inhibit time	0 ms	0	0.1 ms	65535	rw	U16
4: Reserved	-	- (reserved, read or write access results in error message 0x06090011)			rw	U8
5: Event timer	0 ms	0	ms	65535	rw	U16

Communication parameters for sending process data via TPDO1

Subindex	Meaning	Code
0	The value 5 is permanently set. • Max. 5 subindexes are supported.	-
1	TPDO1 identifier • According to the "Predefined Connection Set", the basic setting is: identifier = 0x180 + node ID	C00354/2
2	TPDO transmission type according to DS301 V4.02 ▸ Transmission type (📖 52)	C00322/1
3	Minimum time between sending two identical TPDOs (see DS301 V4.02).	-
5	Cycle time for PDO transmission with transmission type "254".	C00356/5 C00369

8th byte		7th byte	6th byte	5th byte
Data 4		Data 3	Data 2	Data 1
Bit 31	Bit 30	Bit 29 ... bit 11		Bit 10 ... bit 0
0/1	**0/1**	**Extended identifier***		**11-bit identifier**

* The extended identifier is not supported - bit 11 ... bit 29 must be set to "0".

[13-15] Data telegram assignment

Bit	Setting	
Bit 30	0	RTR to this PDO permissible (Lenze).
	1	RTR to this PDO not permissible (cannot be set).
Bit 31	0	PDO active
	1	PDO inactive

How to change the identifier:

1. Deactivate identifier (set bit 31 to "1").
2. Change identifier.
3. Activate identifier (set bit 31 to "0").

Subindex 2 - transmission type

PDO transmission			Transmission type	Explanation
cyclic	synchronous	event-controlled		
●	●		n = 1 ... 240	When a value n is entered, this PDO will be accepted with every nth sync.
●		●	n = 254	Event-controlled or cyclic

Subindex 3 - inhibit time

Note!

The delay time can only be changed when the PDO is not active (see subindex 1, bit 31 = 1).

The entered value multiplied by 0.1 gives the delay time in [ms]. Only integers will be considered, i.e. fractional numbers will be **rounded down** to integers.

Example:

- Entered value: 26
- Calculated time = 26 x 0.1 [ms] = 2.6 [ms] → delay time = 2 [ms]

Subindex 5 - event timer

For cyclic operation (transmission type 254), the cycle time for sending the process data object on the CAN bus can be set here:

The value entered corresponds to the time in [ms].

I-1801 - TPDO2 communication parameter

Index: **I-1801**	Name: **TPDO2 communication parameter**					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Highest subindex supported	5	- (read access only)			ro	U8
1: COB-ID used by TPDO	0x280 + node ID	0		4294967295	rw	U32
2: Transmission type	254	0		255	rw	U8
3: Inhibit time	0 ms	0	0.1 ms	65535	rw	U16
4: Reserved	-	- (reserved, read or write access results in error message 0x06090011)			rw	U8
5: Event timer	0 ms	0	ms	65535	rw	U16

Communication parameters for sending process data via TPDO2

Subindex	Meaning	Code
0	The value 5 is permanently set. • Max. 5 subindexes are supported.	-
1	TPDO2 identifier • According to the "Predefined Connection Set", the basic setting is: identifier = 0x280 + node ID	C00354/4
2	TPDO transmission type according to DS301 V4.02 ▸ Transmission type (📖 52)	C00322/2
3	Minimum time between sending two identical TPDOs (see DS301 V4.02).	-
5	Cycle time for PDO transmission with transmission type "254".	C00356/2 C00369

For data telegram assignment and description of subindexes, see object I-1800.

How to change the identifier:

1. Deactivate identifier (set bit 31 to "1").
2. Change identifier.
3. Activate identifier (set bit 31 to "0").

I-1A00 - TPDO1 mapping parameter

Index: I-1A00	Name: TPDO1 mapping parameter					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Number of mapped application objects in PDO	0	0		8	rw	U8
1 ... 4: Application object 1 ... 4	0	0		4294967295	rw	U32

Object I-1A00 serves to send parameter data as TPDO1.

Subindex	Meaning
0	Number of mapped objects
1 ... 4	Mapping entries 1 ... 4 for TPDO1 • The 4th mapping entry is used for the static mapping. For this, no value is available.

8th byte	7th byte	6th byte	5th byte
Data 4	Data 3	Data 2	Data 1
Bit 31 ... bit 16		Bit 15 ... bit 8	Bit 7 ... bit 0
Index		Subindex	Length

[13-16] Data telegram assignment

IEC 61131 process data words are mapped. Only whole bytes can be mapped (1 byte/ mapping entry).

I-1A01 - TPDO2 mapping parameter

Index: I-1A01	Name: TPDO2 mapping parameter					
Subindex	Default setting	Setting range (min. value | unit | max. value)			Access	Data type
0: Number of mapped application objects in PDO	0	0		8	rw	U8
1 ... 4: Application object 1 ... 4	0	0		4294967295	rw	U32

Object I-1A01 serves to send parameter data as TPDO2.

Subindex	Meaning
0	Number of mapped objects
1 ... 4	Mapping entries 1 ... 4 for TPDO2 • The 4th mapping entry is used for the static mapping. For this, no value is available.

For data telegram assignment, see object I-1A00.

14 DIP switch positions for setting the CAN node address

The node address results from the sum of the binary values of switches 1 ... 64.

The following table shows the switch positions for the valid address range of 1 ... 63.

▸ Setting the CAN node address (📖 33)

Station address	DIP switch						
	64	32	16	8	4	2	1
1	OFF	OFF	OFF	OFF	OFF	OFF	**ON**
2	OFF	OFF	OFF	OFF	OFF	**ON**	OFF
3	OFF	OFF	OFF	OFF	OFF	**ON**	**ON**
4	OFF	OFF	OFF	OFF	**ON**	OFF	OFF
5	OFF	OFF	OFF	OFF	**ON**	OFF	**ON**
6	OFF	OFF	OFF	OFF	**ON**	**ON**	OFF
7	OFF	OFF	OFF	OFF	**ON**	**ON**	**ON**
8	OFF	OFF	OFF	**ON**	OFF	OFF	OFF
9	OFF	OFF	OFF	**ON**	OFF	OFF	**ON**
10	OFF	OFF	OFF	**ON**	OFF	**ON**	OFF
11	OFF	OFF	OFF	**ON**	OFF	**ON**	**ON**
12	OFF	OFF	OFF	**ON**	**ON**	OFF	OFF
13	OFF	OFF	OFF	**ON**	**ON**	OFF	**ON**
14	OFF	OFF	OFF	**ON**	**ON**	**ON**	OFF
15	OFF	OFF	OFF	**ON**	**ON**	**ON**	**ON**
16	OFF	OFF	**ON**	OFF	OFF	OFF	OFF
17	OFF	OFF	**ON**	OFF	OFF	OFF	**ON**
18	OFF	OFF	**ON**	OFF	OFF	**ON**	OFF
19	OFF	OFF	**ON**	OFF	OFF	**ON**	**ON**
20	OFF	OFF	**ON**	OFF	**ON**	OFF	OFF
21	OFF	OFF	**ON**	OFF	**ON**	OFF	**ON**
22	OFF	OFF	**ON**	OFF	**ON**	**ON**	OFF
23	OFF	OFF	**ON**	OFF	**ON**	**ON**	**ON**
24	OFF	OFF	**ON**	**ON**	OFF	OFF	OFF
25	OFF	OFF	**ON**	**ON**	OFF	OFF	**ON**
26	OFF	OFF	**ON**	**ON**	OFF	**ON**	OFF
27	OFF	OFF	**ON**	**ON**	OFF	**ON**	**ON**
28	OFF	OFF	**ON**	**ON**	**ON**	OFF	OFF
29	OFF	OFF	**ON**	**ON**	**ON**	OFF	**ON**
30	OFF	OFF	**ON**	**ON**	**ON**	**ON**	OFF
31	OFF	OFF	**ON**	**ON**	**ON**	**ON**	**ON**
32	OFF	**ON**	OFF	OFF	OFF	OFF	OFF
33	OFF	**ON**	OFF	OFF	OFF	OFF	**ON**
34	OFF	**ON**	OFF	OFF	OFF	**ON**	OFF
35	OFF	**ON**	OFF	OFF	OFF	**ON**	**ON**
36	OFF	**ON**	OFF	OFF	**ON**	OFF	OFF
37	OFF	**ON**	OFF	OFF	**ON**	OFF	**ON**
38	OFF	**ON**	OFF	OFF	**ON**	**ON**	OFF

Station address	DIP switch						
	64	32	16	8	4	2	1
39	OFF	**ON**	OFF	OFF	**ON**	**ON**	**ON**
40	OFF	**ON**	OFF	**ON**	OFF	OFF	OFF
41	OFF	**ON**	OFF	**ON**	OFF	OFF	**ON**
42	OFF	**ON**	OFF	**ON**	OFF	**ON**	OFF
43	OFF	**ON**	OFF	**ON**	OFF	**ON**	**ON**
44	OFF	**ON**	OFF	**ON**	**ON**	OFF	OFF
45	OFF	**ON**	OFF	**ON**	**ON**	OFF	**ON**
46	OFF	**ON**	OFF	**ON**	**ON**	**ON**	OFF
47	OFF	**ON**	OFF	**ON**	**ON**	**ON**	**ON**
48	OFF	**ON**	**ON**	OFF	OFF	OFF	OFF
49	OFF	**ON**	**ON**	OFF	OFF	OFF	**ON**
50	OFF	**ON**	**ON**	OFF	OFF	**ON**	OFF
51	OFF	**ON**	**ON**	OFF	OFF	**ON**	**ON**
52	OFF	**ON**	**ON**	OFF	**ON**	OFF	OFF
53	OFF	**ON**	**ON**	OFF	**ON**	OFF	**ON**
54	OFF	**ON**	**ON**	OFF	**ON**	**ON**	OFF
55	OFF	**ON**	**ON**	OFF	**ON**	**ON**	**ON**
56	OFF	**ON**	**ON**	**ON**	OFF	OFF	OFF
57	OFF	**ON**	**ON**	**ON**	OFF	OFF	**ON**
58	OFF	**ON**	**ON**	**ON**	OFF	**ON**	OFF
59	OFF	**ON**	**ON**	**ON**	OFF	**ON**	**ON**
60	OFF	**ON**	**ON**	**ON**	**ON**	OFF	OFF
61	OFF	**ON**	**ON**	**ON**	**ON**	OFF	**ON**
62	OFF	**ON**	**ON**	**ON**	**ON**	**ON**	OFF
63	OFF	**ON**	**ON**	**ON**	**ON**	**ON**	**ON**

15 Index

R

S

T

U

V

Lenze Drives GmbH
Postfach 10 13 52
D-31763 Hameln
Germany

+49 (0)51 54 / 82-0

+49 (0)51 54 / 82-28 00

Lenze@Lenze.de

www.Lenze.com

Service Lenze Service GmbH
Breslauer Straße 3
D-32699 Extertal
Germany

00 80 00 / 24 4 68 77 (24 h helpline)

+49 (0)51 54 / 82-11 12

Service@Lenze.de

EDS84DMOTCAN ▪ 13395082 ▪ EN 3.0 ▪ TD17

10 9 8 7 6 5 4 3 2 1