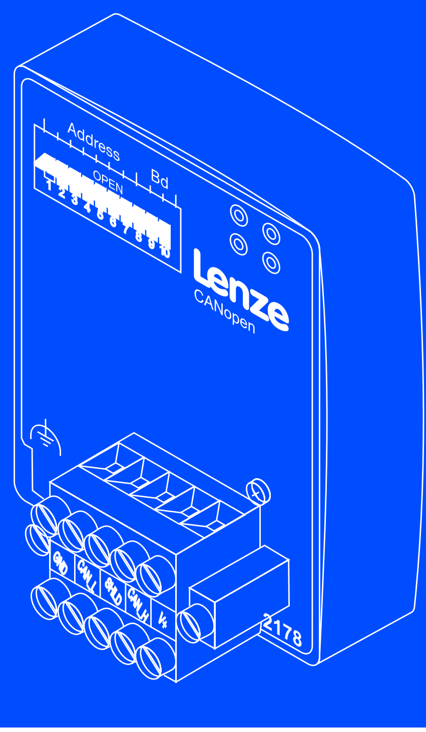
EDSMF2178IB
13437291

L-force *Communication*

Communication Manual

CANopen



EMF2178IB

Communication module

Lenze

Contents

1 About this documentation

Contents

This documentation exclusively contains descriptions of the EMF2178IB communication module (CANopen).

Note!

This documentation supplements the **mounting instructions** supplied with the function/communication module and the **documentation of the used standard device**.

The mounting instructions contain safety instructions which must be observed!

The features and functions of the communication module are described in detail.

Examples illustrate typical applications.

Furthermore this documentation contains the following:

- Safety instructions that must be observed.
- Key technical data relating to the communication module
- Information on versions of Lenze standard devices to be used.
- Notes on troubleshooting and fault elimination

The theoretical correlations are only explained in so far as they are necessary for comprehending the function of the communication module.

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

All brand names mentioned in this manual are trademarks of their respective companies.

Tip!

For further information visit the homepage 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 auxiliary devices related to the Lenze products can be found in the download area at

http://www.Lenze.com

Validity information

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

- ► EMF2178IB communication modules (CANopen) as of version 1x.2x.

1.1 Document history

Material no.	Version			Description
-	1.0	01/2008	TD17	First edition
13127634	2.0	07/2011	TD17	General revision
13437291	3.0	06/2013	TD17	• New chapter "Replacing communication module EMF2172IB (CAN)" (🕮 39) • General updates

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

Type of information	Identification	Examples/notes
Spelling of numbers		
Decimal separator	Point	In general, the decimal point is used. For instance: 1234.56
Decimal	Standard notation	Example: 1234
Hexadecimal	0x[0 ... 9, A ... F]	Example: 0x60F4
Binary ● Nibble	0b[0, 1] Point	Example: '0b0110' Example: '0b0110.0100'
Text		
Program name	» «	PC software For example: »Engineer«, »Global Drive Control« (GDC)
Icons		
Page reference	🕮	Reference to another page with additional information For instance: 🕮 16 = see page 16
Documentation reference	☞	Reference to another documentation with additional information For example: ☞ EDKxxx = see documentation EDKxxx

1.3 Terminology used

Term	Meaning
Standard device	Lenze controllers that can be used with the communication module. 🕮 13
Drive	
»Global Drive Control« / »GDC«	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 speech, the term is usually called "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 code and subcode (e.g. "C00118/3"). In normal speech, the term is also called "Subindex".
Lenze setting	These are settings the device is preconfigured with ex works.
Basic setting	
HW	Hardware
SW	Software
PDO	Process data object
SDO	Service data object

1.4 Notes used

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

Safety instructions

Structure of safety instructions:

> **Danger!**
> (characterises the type and severity of danger)
> **Note**
> (describes the danger and gives information about how to prevent dangerous situations)

Pictograph and 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 and signal word	Meaning
Note!	Important note to ensure troublefree operation
Tip!	Useful tip for simple handling
	Reference to another documentation

2 Safety instructions

Note!

It is absolutely vital that the stated safety measures are implemented in order to prevent serious injury to persons and damage to material assets.

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

2.1 General safety information

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 for the intended purpose.

  ... must never be operated if damaged.

  ... must never be subjected to technical modifications.

  ... must never be operated unless completely assembled.

  ... must never be operated without the covers/guards.

  ... can - depending on their degree of protection - have live, movable or rotating parts during or after operation. Surfaces can be hot.
- For Lenze drive components ...

  ... only use permitted accessories.

  ... only use original manufacturer spare parts.
- All specifications of the corresponding enclosed documentation must be observed.

  This is vital for a safe and trouble-free operation and for achieving the specified product features.

  The procedural notes and circuit details provided in this document are proposals which the user must check for suitability for his application. The manufacturer does not accept any liability for the suitability of the specified procedures and circuit proposals.
- Only qualified skilled personnel are permitted to work with or on Lenze drive and automation components.

  According to IEC 60364 or CENELEC HD 384, these are persons ...

  ... who are familiar with the installation, assembly, commissioning and operation of the product,

  ... possess the appropriate qualifications for their work,

  ... and are acquainted with and can apply all the accident prevent regulations, directives and laws applicable at the place of use.

2.2 Device- and application-specific safety instructions

- During operation, the communication module must be securely connected to the standard device.
- 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 meet the given specifications. (🕮 26)

Documentation of the standard device, control system, and plant/machine
All the other measures prescribed in this documentation must also be implemented. Observe the safety instructions and application notes contained in this manual.

2.3 Residual hazards

Protection of persons

- If controllers are connected to phase-earthed system with a rated mains voltage ≥ 400 V, external measures need to be implemented to provide reliable protection against accidental contact. (see chapter "4.2", 🕮 18)

Device protection

- The communication module contains electronic components that can be damaged or destroyed by electrostatic discharge.

3 Product description

3.1 Application as directed

The communication module ...

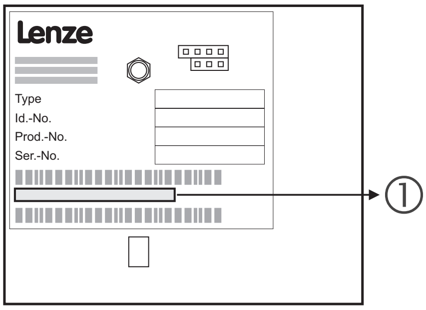
- enables communication with Lenze controllers over the CAN bus with the CANopen communication profile.
- is a device intended for use in industrial power systems.
- can be used in conjunction with the following Lenze controllers:

Series	Device type	Version		Explanation
		HW	SW 1)	
8200 vector	E82xVxxxKxBxxxXX	Vx	1x	8200 vector frequency inverter
9300	EVS9321-xS ... EVS9332-xS	2x	1x	Servo inverter
	EVS9321-xK ... EVS9332-xK	2x	1x	Servo cam profiler
	EVS9321-xP ... EVS9332-xP	2x	1x	Servo position controller
	EVS9321-xR ... EVS9332-xR	2x	1x	Servo register controller
	EVS9321-xI ... EVS9332-xI	2x	8x	9300 servo PLC
	EVS9321-xT ... EVS9332-xT	2x	8x	
9300 vector	EVF9321-xV ... EVF9333-xV	2x	1x	9300 vector frequency inverter
	EVF9335-xV ... EVF9338-xV	1x	0x	
	EVF9381-xV ... EVF9383-xV	1x	0x	
ECS servo system	ECSxSxxxC4xxxxXX	1A	6x	”Speed and Torque”
	ECSxPxxxC4xxxxXX	1A	6x	”Posi and Shaft”
	ECSxMxxxC4xxxxXX	1A	6x	”Motion”
	ECSxAxxxC4xxxxXX	1A	6x	”Application”
Drive PLC	EPL10200-xI ... EPL10203-xI	1x	8x	Drive PLC

1) Operating system software versions of the controllers

Any other use shall be deemed inappropriate!

3.2 Identification



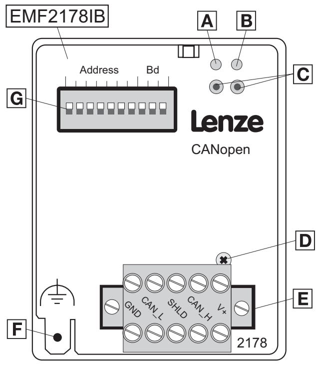
99371BC013

Type code	① →	33.2178IB	1x	2x
Series				
Hardware version				
Software version				

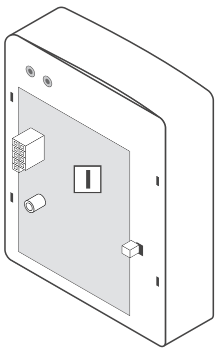
3.3 Product features

- Attachable communication module for the basic Lenze devices 8200 vector, 93XX, 9300 Servo PLC, Drive PLC, ECSXX
- Front DIP switches for setting the ...
  – CAN node address (max. 127 nodes)
  – baud rate (10, 20, 50, 125, 250, 500 and 1000 kbit/s)
- Bus expansion without repeater up to 7450 m
- Topology: Line terminated at both ends (R = 120 Ω)
- Simple connection through plug-on screw terminals
- Together with a 9300 Servo PLC, additional CANopen application profiles can be implemented.

3.4 Connections and interfaces



2178CAN003



2102LEC007

Legend for fold-out page		
Pos.	**Description**	**Detailed information**
A	Connection status to standard device (two-colour LED)	
B	Connection status to fieldbus (two-colour LED)	🕮 89
C	Operating status of standard device (green and red Drive LED)	
D	Fixing screw	
E	Plug connector with double screw connection, 5-pole	🕮 24
F	PE shield cable connection	
G	DIP switches for setting the ● node address (switches 1 ... 7) ● baud rate (switches 8 ... 10)	🕮 33
I	Nameplate	🕮 14

4 Technical data

4.1 General data and operating conditions

Field	Values
Order designation	EMF2178IB
Communication media	DIN ISO 11898
Network topology	Line terminated at both ends (R = 120 Ω)
Communication profile	CANopen, DS301 V4.01
Node addresses	Max. 127
Cable length	Max. 7450 m (depending on the baud rate, 🕮 27)
Baud rate [kbit/s]	10, 20, 50, 125, 250, 500, 1000
Voltage supply	Internal or external supply possible for basic devices: 8200 vector / 93XX / 9300 Servo PLC / Drive PLC / ECSXX (also see 🕮 30)
	External supply via separate power supply unit: V+: V = 24 V DC ± 10 % I = 100 mA GND: Reference potential for external voltage supply

Documentation for Lenze series of devices 8200 vector, 9300 and ECS

Here you can find the **ambient conditions** and the **electromagnetic compatibility (EMC)** specifications applying to the communication module.

4.2 Protective insulation

Danger!

Dangerous electrical voltage

If Lenze controllers are used on a phase earthed mains with a rated mains voltage ≥ 400 V, protection against accidental contact is not ensured without implementing external measures.

Possible consequences:

► Death or serious injury

Protective measures:

► If protection against accidental contact is required for the control terminals of the controller and the connections of the plugged device modules, ...
– a double isolating distance must exist.
– the components to be connected must be provided with the second isolating distance.

Protective insulation between the bus and ...	Type of insulation according to EN 61800-5-1
Reference earth / PE	Functional insulation
External supply	No functional insulation
Power section	
● 8200 vector	Double insulation
● 9300 vector, Servo PLC	Double insulation
● Drive PLC	Double insulation
● ECSXX	Double insulation
Control terminals	
● 8200 vector (with internal supply, 🕮 30)	No functional insulation
● 8200 vector (with external supply, 🕮 30)	Basic insulation
● 9300 vector, Servo PLC	Basic insulation
● Drive PLC	Basic insulation
● ECSXX	Basic insulation

4.3 Communication time

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

The CAN bus communication times depend on ...

- the processing time in the controller (see documentation of the controller)
- Telegram runtime
  – baud rate
  – telegram length
- the data priority
- the bus load

Processing time in the controller

Documentation for the controller
Here you can find information on the processing times in the controller.

Telegram time

The telegram times depend on the baud rate and the telegram length:

Baud rate [kbit/s]	**Telegram runtime** [ms]		
	0 bytes	**2 bytes**	**8 bytes**
10	5.44	7.36	13.12
20	2.72	3.68	6.56
50	1.09	1.47	2.62
125	0.44	0.59	1.05
250	0.22	0.29	0.52
500	0.11	0.15	0.26
1000	0.05	0.07	0.13

4.4 Dimensions

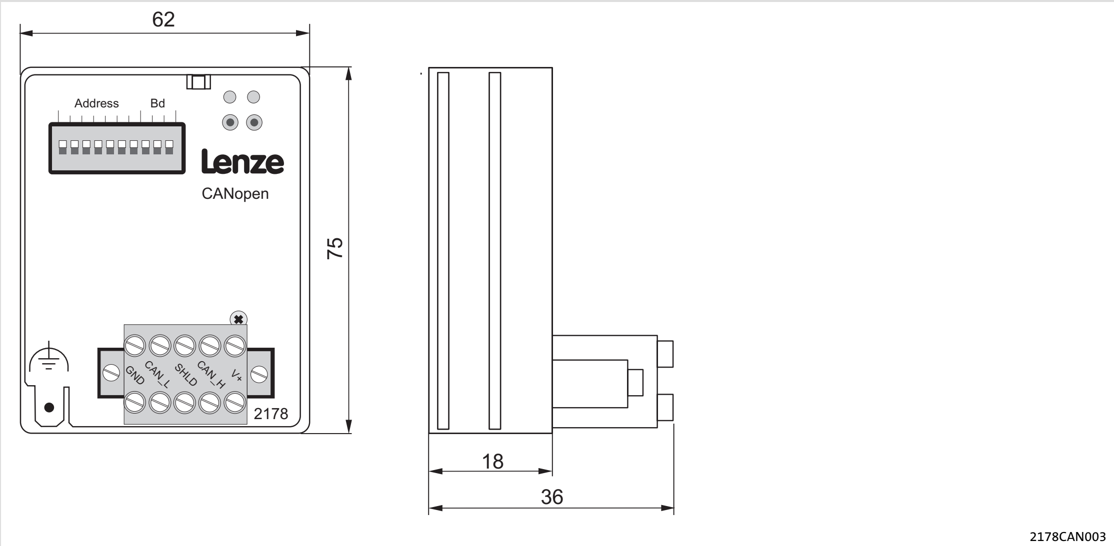


Fig. 4-1 Dimensions of the communication module (all dimensions in mm)

5 Installation

Danger!

Inappropriate handling of the communication module and the standard device can cause serious personal injury and material damage.

Observe the safety instructions and residual hazards described in the documentation for the standard device.

Stop!

The device contains components that can be destroyed by electrostatic discharge!

Before working on the device, the personnel must ensure that they are free of electrostatic charge by using appropriate measures.

5.1 Mechanical installation

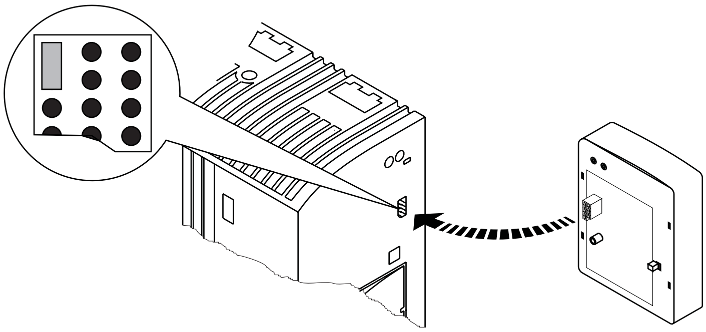


Fig. 5-1 Attaching the communication module

- Plug the communication module onto the standard device (here: 8200 vector).
- Tighten the communication module to the standard device using the fixing screw in order to ensure a good PE connection.

Note!

For the internal supply of the communication module by the 8200 vector frequency inverter the jumper has to be adjusted within the interface opening (see illustration above).

Observe the notes (📖 30).

5.2 Electrical installation

5.2.1 Wiring according to EMC (CE-typical drive system)

For wiring according to EMC requirements observe the following points:

Note!

- Separate control cables/data lines from motor cables.
- Connect the shields of control cables/data lines *at both ends* in the case of digital signals.
- Use an equalizing conductor with a cross-section of at least 16 mm^2 (reference: PE) to avoid potential differences between the bus nodes.
- Observe the other notes concerning EMC-compliant wiring given in the documentation for the standard device.

Procedure for wiring

1. Observe the bus topology, i.e. do not use stubs.
2. Observe notes and wiring instructions in the documents for the control system.
3. Only use cables corresponding to the listed specifications (📖 26).
4. Observe the permissible bus cable length (📖 27).
5. Connect bus terminating resistors of 120 Ω each (scope of supply):
   – only to the physically first and last node
   – between the terminals CAN-LOW and CAN-HIGH

5.2.2 Wiring with a host (master)

Danger!
An additional electrical isolation is required if a safe electrical isolation (reinforced insulation) to EN 61800-5-1 is necessary.

For this purpose for instance an interface module for the master computer with an additional electrical isolation can be used (see respective manufacturer information).

For wiring observe the electrical isolation of the supply voltage. The supply voltage is on the same potential as the data bus.

The 5-pole plug connector with double screw connection serves to

- connect the bus (📖 25);
- effect the external voltage supply (📖 30).

Fig. 5-2 5-pole plug connector with double screw connection

Designation	Explanation
GND	Reference potential for external voltage supply CAN-GND connection
CAN_L	Data line / input for terminating resistor 120 Ω
SHIELD	Shielding
CAN_H	Data line / input for terminating resistor 120 Ω
V+	External voltage supply

5.2.3 Wiring system bus (CAN)

Structure of a CAN bus system (example)

The CAN bus system is designed as a 2 conductor (twisted pair) shielded with additional mass and termination at both ends of the line.

For sending and receiving data the following paths are available:

► Max. three process data channels (PDO = Process Data Object)
  – Process data are sent via the process data channel and are used for high-speed and high-priority control tasks. Typical process data are, for instance, control words, status words, setpoints and actual values of a standard device.

► Two parameter data channels (SDO = Service Data Object)
  – The parameters are transferred at lower priority than the process data and are set or changed e.g. during commissioning or product change.
  – The parameters are accessed via the parameter data channels of the EMF2178IB communication module to the codes of the basic Lenze device or the corresponding CANopen objects.
  – With both parameter data channels, two masters can be connected to a standard device. A PC (e.g. with the Lenze software ”Global Drive Control”) or an operator terminal serve to change parameters directly at the standard device during operation of a system connected to PLC. The second parameter data channel can be reached under the set address (via DIP switch or C0009) with an offset of ”64”. If, e.g., a PLC addresses the standard device with the address ”1” and a second commanding device the address ”65”, always the same standard device is addressed.
  – The second parameter channel is deactivated in the default state.

Note!

► The last telegram determines the parameter when a parameter is accessed by two units (see CANopen objects 1200 and 1201 ”Server SDO Parameters”.(📖 107)).
► Please observe the notes in the chapter 6Commissioning (📖 33), if you do not select the baud rate and address via the front DIP switches.

Fig. 5-3 Connection to the plug connector

5.2.4 Specification of the transmission 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	≤ 70 mΩ/m / 0.25 ... 0.34 mm^2 (AWG22)
	Cable length 301 ... 1000 m	≤ 40 mΩ/m / 0.5 mm^2 (AWG20)
Signal propagation delay		≤ 5 ns/m

5.2.5 Bus cable length

Note!

- ► It is absolutely necessary to comply with the permissible cable lengths.
- ► If the total cable lengths of the CAN nodes differ for the same baud rate, the smaller value must be used to determine the max. cable length.
- ► Observe the reduction of the total cable length due to the signal delay of the repeater.

5.2.5.1 Total cable length

The baud rate determines the total cable length.

Baud rate [kbps]	**Max. bus length** [m]
10	7450
20	3950
50	1550
125	630
250	290
500	120
1000	25

Tab. 5-1 Total cable length

5.2.5.2 Segment cable length

The segment cable length is determined by the cable cross-section used and by the number of nodes. Repeaters divide the total cable length into segments. If no repeaters are used, the segment cable length is identical to the total cable length.

Max. number of nodes per segment	**Cable cross-section** (can be interpolated)			
	0.25 mm² **(AWG24)**	**0.5 mm²** **(AWG21)**	**0.75 mm²** **(AWG19)**	**1.0 mm²** **(AWG18)**
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
100	150 m	270 m	410 m	600 m

Tab. 5-2 Segment cable length

Example: Selection help

Given	
Total cable length to be implemented	200 m
Number of nodes	63

Results	
Max. possible baud rate	250 kbps (derived from Tab. 5-1 Total cable length)
Required cable cross-section (interpolated)	0.30 mm² (AWG23) (derived from Tab. 5-2 Segment cable length)
Cable cross-section of standard CAN cable	0.34 mm² (AWG22) (see specification of the transmission cable 🕮 26)

5.2.5.3 Checking the use of repeaters

Compare the values derived from Tab. 5-1 Total cable length (🕮 27) and Tab. 5-2 Segment cable length (🕮 28).

- If the sum of the segment cable lengths is smaller than the total cable length to be implemented, either repeaters must be used or the cable cross-section must be increased.
- If the use of repeaters reduces the max. possible total cable length so much that it is smaller than the total cable length to be implemented, then the cable cross-section must be increased or less repeaters must be used or the baud rate must be decreased.
- The use of a further repeater is recommended as ...
  - service interface
    Advantage: Trouble-free connection during bus operation is possible.
  - calibration interface
    Advantage: The calibration/programming unit remains electrically isolated.

Example

Given	
Total cable length to be implemented	450 m
Number of nodes	32
Cable cross-section	0.50 mm^2 (AWG 20)
Baud rate	125 kbps
Repeater used	Lenze repeater EMF2176IB
Reduction of the max. total cable length per repeater (EMF2176IB)	30 m

Results	
Max. total cable length	630 m (see Tab. 5-1 Total cable length (🕮 27))
Max. segment cable length	360 m (see Tab. 5-2 Segment cable length (🕮 28))
Comparison	The max. segment cable length is smaller than the total cable length to be implemented.
Conclusion	A repeater must be installed at the determined max. segment cable length of 360 m.

Results with 1 repeater	
Max. total cable length	600 m (Reduction of the total cable length (🕮 27) by 30 m)
Max. segment cable length	720 m
Comparison	Both the possible total cable length and the segment cable lengths are larger than the total cable length to be implemented.
Conclusion	1 repeater is sufficient to implement the total cable length of 450 m.

5.2.6 Voltage supply

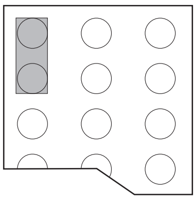
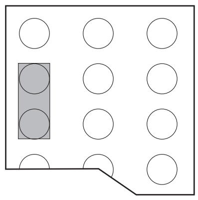
Internal voltage supply

Note!

Internal voltage supply has been selected in the case of standard devices with an extended AIF interface opening (e.g. front of 8200 vector). The area shown on a grey background in the graphic marks the jumper position.

- ► By default, this is **not** supplied internally in the standard device.
- ► For internal voltage supply place the jumper on the position indicated below.

In the case of all other device series (9300, ECS), voltage is always supplied from the standard device.

Lenze setting (Only external voltage supply possible.)	**Internal voltage supply**
	

External voltage supply

Note!

In the case of an external voltage supply and for greater distances between the control cabinets, always use a separate power supply unit (SELV/PELV) that is safely separated in accordance with EN 61800-5-1 in each control cabinet.

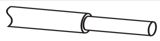
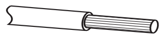
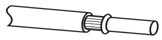
The external voltage supply of the communication module ...

- ► is required if communication via the fieldbus is to be continued in case the supply of the standard device fails.
- ► is carried out via the 2-pole plug connector with screw connection (24 V DC):

Terminal	Description
V+	External voltage supply V = 24 V DC ± 10 % I = 100 mA
GND	Reference potential for external voltage supply

- ► The parameters of a basic device disconnected from the mains cannot be accessed.

Terminal data

Area	Values	
Electrical connection	Plug connector with screw connection	
Possible connections	rigid:	
		1.5 mm^2 (AWG 16)
	flexible:	
		without wire end ferrule 1.5 mm^2 (AWG 16)
		with wire end ferrule, without plastic sleeve 1.5 mm^2 (AWG 16)
		with wire end ferrule, with plastic sleeve 1.5 mm^2 (AWG 16)
Tightening torque	0.5 ... 0.6 Nm (4.4 ... 5.3 lb-in)	
Stripping length	6 mm	

6 Commissioning

During commissioning, system-dependent data as e.g. motor parameters, operating parameters, responses and parameters for fieldbus communication are selected for the controller.

In Lenze devices, this is done via codes. The codes are stored in numerically ascending order in the Lenze controllers and in the plugged-in communication/function modules.

In addition to these configuration codes, there are codes for diagnosing and monitoring the bus devices.

6.1 Before switching on

Stop!

Before you switch on the standard device with the communication module for the first time, check

- ► the entire wiring with regard to completeness, short circuit, and earth fault.
- ► whether the bus system is terminated by a bus terminating resistor at the physically first and last node.

6.2 Installing EDS files

The EDS files serve to implement the Lenze communication modules for the AIF and FIF interfaces into the CANopen configuration software.

The single EDS files describe the implemented CANopen functions of the respective communication module and the "on board" Lenze system bus (CAN).

Tip!

The current EDS file required for configuring the EMF2178IB (CANopen) communication module can be found in the download area on:

www.Lenze.com

6.3 Setting node address and baud rate

The node address and the baud rate can be set via codes or via the DIP switches arranged at the front:

- Node address: Switches **1 ... 7** / code **C1850/C2350**
- Baud rate: Switches **8 ... 10** / code **C1851/C2351**

The Lenze setting of all DIP switches is OFF.

Note!

Settings via codes

- In the Lenze setting – Address switches **1 ... 7 = OFF** –, the values are accepted from the codes **C1850/C2350** (node address) and **C1851/C2351** (baud rate).
  – Node addresses > 99 can only be set via DIP switch.
  – The baud rates 10 kbps and 20 kbps can only be selected via DIP switch.
- Writing the codes (e.g. with GDC via CAN) has a direct effect on the standard device codes C0009 and C0126.
- Acceptance of code changes by:
  – Switching off and then on again the voltage supply;
  – ”Reset node” with C0358 = 1;
  – Network management command ”Reset Communication”;
  – Set C2120 (AIF control byte) = 1.
- The codes are inactive if at least one address switch (1 ... 7) has been set to ON position before a renewed mains connection.

Node address setting

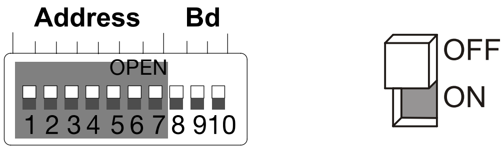


Fig. 6-1 Address assignment via DIP switch

- ► If several devices are connected to the CAN network, the node addresses must differ from each other.
- ► The desired node address results from the sum of the values of switches (1 ... 7) in ON position.

Switch	Value	Example	
		Switch position	**Node address**
1	64	OFF	
2	32	OFF	
3	16	**ON**	
4	8	OFF	16 + 4 + 2 + 1 = **23**
5	4	**ON**	
6	2	**ON**	
7	1	**ON**	

Note!

Switch off the voltage supply of the communication module, and then switch it on again to activate the changed settings.

Baud rate setting

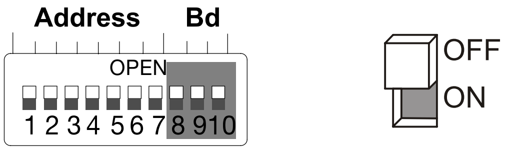


Fig. 6-2 Baud rate setting

- The baud rate must be the same for all CAN nodes.
- The following baud rates can be set:

Baud rate [kbps]	**Switch**		
	8	**9**	**10**
10	**ON**	**ON**	OFF
20	**ON**	OFF	**ON**
50	OFF	**ON**	**ON**
125	OFF	**ON**	OFF
250	OFF	OFF	**ON**
500	OFF	OFF	OFF
1000	**ON**	OFF	OFF

Note!

Switch off the voltage supply of the communication module, and then switch it on again to activate the changed settings.

6.4 Initial switch-on

Note!

Do not change the setting sequence.

1. Switch on the standard device and, if necessary, the external voltage supply of the communication module.
   – The green LED A on the front of the communication module is lit.
   – The status LED of the standard device (Drive LED) C must be lit or blinking. The meaning of the signalling can be found in the standard device documentation.
2. You can now communicate with the drive, i. e.
   – all parameters (SDO) can be read;
   – all writable parameters (SDO) can be overwritten.
3. After a state change (”Operational”) process data can be exchanged with the drive.

Fig. 6-3 LEDs of the communication module

6.5 Enable drive via the communication module

Note!

- ► During operation, the plugging of the communication module onto a different controller may cause undefined operating states.
- ► Observe the information on the CAN configuration and controller enable in the documentation of the corresponding controller.

Enable the 8200 vector via the communication module

Step	Procedure	Comments
1.	Set C0001 from ”0” to ”3”	The Lenze parameter C0001 (operating mode) can be set with the GDC, keypad XT or directly via CANopen. Example for direct setting via CANopen: Write (C0001 = 3) ● Index = 0x5FFE (resulting from: 0x5FFF - $(C0001)_{hex}$;) ● Subindex: 0 ● Value: 30000 (resulting from: C0001 = 3 x 10000)
2.	Assign 28 to HIGH level	The terminal 28 (controller enable) is always active and must be assigned to HIGH level during CANopen operation. Otherwise the controller cannot be enabled via CANopen.
3.	Assign input terminal for QSP to HIGH level	The quick stop function (QSP) is always active. If QSP is configured to an input terminal (Lenze setting: Not assigned), it must be assigned to HIGH level during CANopen operation.
4.	The controller now accepts parameter data and process data.	

Enable 93XX via the communication module

Step	Procedure	Comments
1.	Set C0005 to th value ”xxx3”	The value ”xxx3” of the Lenze parameter C0005 (control of the controller via CANopen) can be set with the GDC, keypad XT or directly via CANopen. Example for the first commissioning with the signal configuration ”1013”: Write (C0005 = 1013) ● Index = 0x5FFA (resulting from: 0x5FFF - $(C0005)_{hex}$) ● Subindex: 0 ● Value: 10130000 (resulting from: C0005 = 1013 x 10000)
2.	Set C0142 = 0	See ”Protection against uncontrolled restart” (📖 38).
3.	Assign 28 to HIGH level	The terminal 28 (controller enable) is always active and must be assigned to HIGH level during CANopen operation. Otherwise the controller cannot be enabled via CANopen.
4.	Assign terminal E1 to HIGH level	If the signal configuration C0005 = 1013, the quick stop function (QSP) is assigned to the digital input terminals E1 and E2 in connection with the right/left change-over and thus always active.
5.	Connect terminal X5/A1 to ● X5/28 and ● X5/E1	Only affects the signal configuration C0005 = xx13 With this signal configuration the terminal A1 is switched as voltage output.
6.	The controller now accepts parameter data and process data.	

Enable ECSXX via the communication module

Step	Procedure	Comments
1.	Select control interface ”AIF” via code.	See documentation of the corresponding ECS controller.
2.	Set C0142 = 0	See ”Protection against uncontrolled restart” (38).
3.	Assign terminals X6/SI1 and X6/SI2 to HIGH level	The terminals X6/SI1 (controller enable/inhibit) and X6/SI2 (pulse enable/inhibit) are always active and must be assigned to HIGH level during CANopen operation. Otherwise, the controller cannot be enabled via CANopen.
4.	The controller now accepts parameter data and process data.	

Protection against uncontrolled restart

Note!

Establishing communication

If communication is to be established via an externally supplied communication module, initially the standard device must also be switched on.

After communication has been established, the externally supplied module is independent of the power on/off state of the standard device.

Protection against uncontrolled restart

After a fault (e.g. short-term mains failure), a restart of the drive is not always wanted and - in some cases - even not allowed.

The restart behaviour of the controller can be set in C0142:

- ► C0142 = 0 (Lenze setting)
  - – The controller remains inhibited (even if the fault is no longer active).
  - – The drive starts up in a controlled manner by explicit controller enable:
    93XX: Set terminal 28 to HIGH level.
    ECSXX: Set terminals X6/SI1 and X6/SI2 to HIGH level.
- ► C0142 = 1
  - – An uncontrolled restart of the drive is possible.

7 Replacing the EMF2172IB communication module (CAN)

Observe the following information when replacing the **EMF2172IB (CAN)** communication module by **EMF2178IB (CANopen)**:

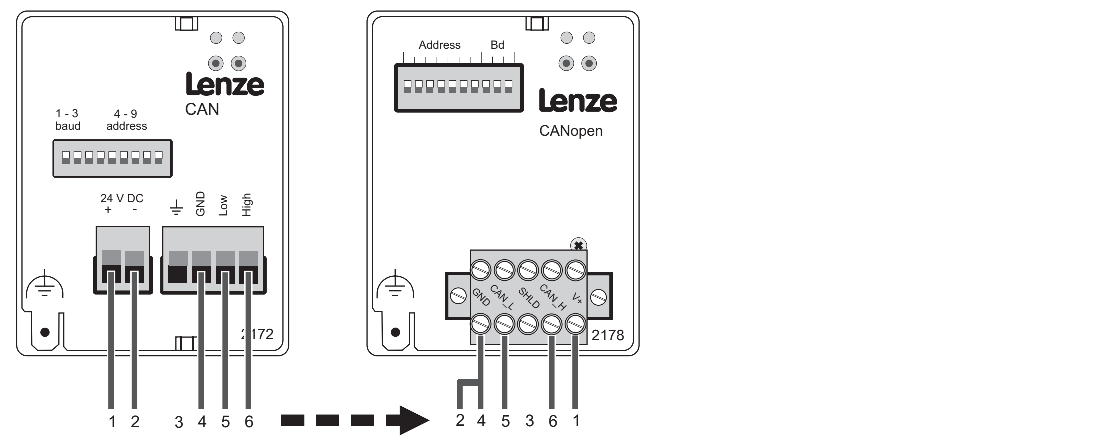
- ► The communication modules feature different plug connectors for connecting the voltage supply and the system bus (CAN).
- ► The DIP switches of the communication modules feature different assignments for setting the CAN node address and the baud rate (🕮 40). Furthermore there are different adjustable CAN address ranges:

Communication module	CAN address range	DIP switch
EMF2172IB (CAN)	1 ... 63	S4 ... S9
EMF2178IB (CANopen)	1 ... 127	S1 ... S7

- ► Depending on the standard device used, the 2. SDO channel is activated via code **C1865/1** or **C2365/1** (🕮 41).

Changing the wiring

The following illustration shows how you must carry out the previous wiring for the EMF2178IB communication module now.

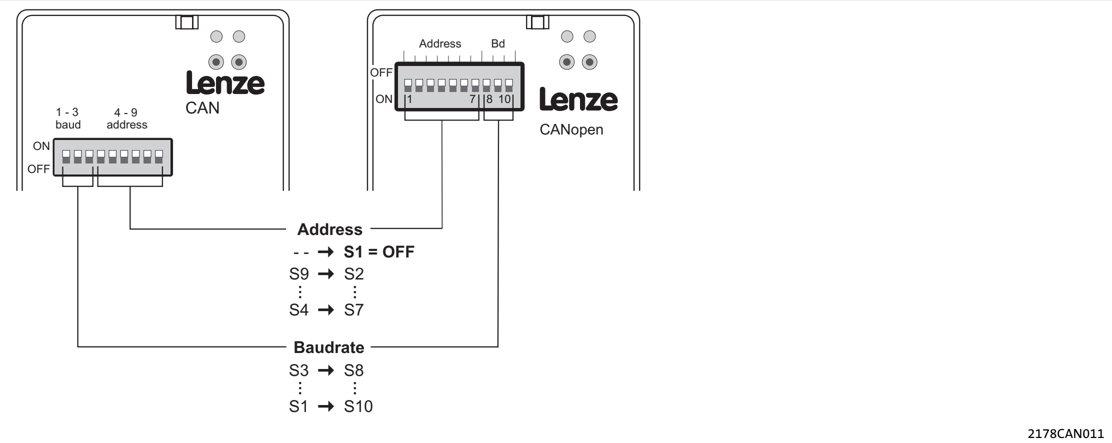


2178CAN010

DIP switch settings

Note!

On the **EMF2178IB** communication module, the switch **S1 = OFF**(Lenze setting) must remain set. (An address setting > 63 was not possible for EMF2172IB.)



Switch		Value	Example	
EMF2172IB	EMF2178IB		Switch position	CAN node address
-	S1	64	OFF	
S9	S2	32	OFF	
S8	S3	16	**ON**	
S7	S4	8	OFF	16 + 4 + 2 + 1 = **23**
S6	S5	4	**ON**	
S5	S6	2	**ON**	
S4	S7	1	**ON**	

Activate the 2. SDO channel

Depending on the standard device used, set code **C1865/1** or **C2365/1** to activate the 2. SDO channel.

Settings via the GDC/XT keypad for these standard devices:	
EVS 93xx-ES **EVS 93xx-EP** **EVS 93xx-EK** **EVS 93xx-ER** **EVF 93xx-EV** **E82EVxxx**	**EVS 93xx-EI** **EVS 93xx-ET** **EPL-10200** **ECSxS/P/M/A**
1. Plug the EMF2178IB communication module onto the standard device.	Use the GDC or XT keypad to set standard device code **C2365/1 = 1** and save it with mains failure protection. After mains connection of one of the standard devices specified above, or after plugging on the EMF2178IB communication module, the contents of C2365 are written to the communication module.
2. Use the GDC to set code **C1865/1 = 1**. The setting is saved with mains failure protection directly in the communication module. When using the standard devices specified above, the setting can also be carried out beforehand.	

8 Data transfer

Master and controller communicate with each other by exchanging data telegrams via the CAN bus.

The user data area of the CAN telegram either contains *network management data*, *process data* or *parameter data* (📖 44).

Different communication channels are assigned to parameter and process data in the controller:

- ► Process data are transferred via the *process data channel*.
- ► Parameter data are transferred via the *parameter data channel*.

8.1 Structure of the CAN telegram

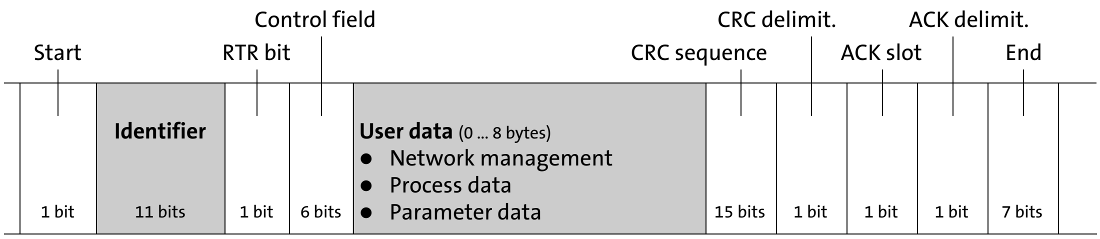


Fig. 8-1 Basic structure of the CAN telegram

Tip!
The identifier and the user data will be explained in more detail in this documentation.
The other signals refer to the transfer characteristics of the CAN telegram which are not described in this documentation.
For more information please refer to the homepage of the CAN user organisation CiA (CAN in Automation):
http://www.can-cia.org

Identifier (COB-ID)

The principle of the CAN communication is based on a message-oriented data exchange between a transmitter and various receivers. Here, all nodes can quasi simultaneously transmit and receive messages. In case of CANopen, a node orientation is achieved by having only one transmitter for each message.

The control which node is to receive a transmitted message is executed via the *identifier*in the CAN telegram, also called *COB-ID*(*Communication Object Identifier*). In addition to the addressing, the identifier contains information on the priority of the message and the type of user data.

With the exception of the network management and the sync telegram, the identifier consists of a basic identifier and the node address of the node to be addressed:

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

The identifier assignment is defined in the CANopen protocol.

Basic identifier

The basic identifier is preset as default with the following values according to the CANopen specification:

Object		Direction		Basic identifier	
		from the drive	to the drive	dec	hex
NMT				0	0x000
Sync				128	0x080
Emergency		X		128	0x080
PDO1 (Process data channel 1)	TPDO1	X		384	0x180
	RPDO1		X	512	0x200
PDO2 (Process data channel 2)	TPDO2	X		640	0x280
	RPDO2		X	768	0x300
PDO3 (process data channel 3)	TPDO3	X		896	0x380
	RPDO3		X	1024	0x400
SDO1 (Parameter data channel 1)		X		1408	0x580
			X	1536	0x600
SDO2 (parameter data channel 2) Lenze setting: not active.		X		1472	0x5C0
			X	1600	0x640
Node guarding / heartbeat		X		1792	0x700

Node address (node ID)

Each node of the CAN network must be assigned with a node address (also called *node ID*) within the valid address range for unambiguous identification.

- A node address must not be assigned more than once within a network.
- The node address of the controller is configured under code **C1850/C2350** or via **DIP switch**(🕮 33).

User data

The user data area of the CAN telegram either contains *network management data*, *process data* or *parameter data*:

- Network management data (NMT data)

  Network management data contain information on the establishment of communication via the CAN network (📖 45).

- Process data (PDO, Process Data Objects)
  - Process data are transferred via the *process data channel* (see also chapter "Process data transfer", 📖 48).
  - Process data serve to control the controller (slave).
  - Process data are transferred between the host and the controllers to ensure a continuous exchange of current input and output data.
  - The host has direct access to process data. In the PLC, the data are, for instance, directly assigned to the I/O area. An exchange between host and controller is required as fast as possible. Small amounts of data can be transferred cyclically.
  - Process data are not saved in the controller.
  - Process data are, for instance, control words, status words, setpoints and actual values.

- Parameter data (SDO, Service Data Objects)
  - Parameter data are transmitted as SDOs via the *parameter data channel* and acknowledged by the receiver, i.e.the sender gets a feedback about the transmission being successful or not (see also chapter "parameter data transfer", 📖 69).
  - The parameter data channel enables access to all Lenze codes and CANopen indices.
  - The parameters for instance are set for the initial system set-up during commissioning or when material is changed on the production machine.
  - In general, the parameter data transfer is not time-critical.
  - Parameter changes are automatically stored in the controller.
  - Parameter data for example are operating parameters, diagnostics information, and motor data.

8.2 CAN communication phases / network management (NMT)

Regarding communication, the controller knows the following statuses:

Status	Description
"Initialisation"	After the controller is switched on, the initialisation phase is run through. During this phase, the controller is not involved in the data exchange on the bus. A part of the initialisation or the complete initialisation can be run through again in every NMT status by transmitting different telegrams (see "state transition"). Here, all parameters are written with their set values. After completing the initialisation, the controller automatically adopts the "Pre-Operational" status.
"Pre-Operational"	The controller can receive parameter data. The process data are ignored.
"Operational"	The controller can receive parameter data and process data.
"Stopped"	Only network management telegrams can be received.

Network management (NMT)

The telegram structure used for the network management contains the identifier and the command included in the user data which consists of the command byte and the node address.

Telegrams with the identifier ”0” and 2 byte user data are used to change between the different communication phases.

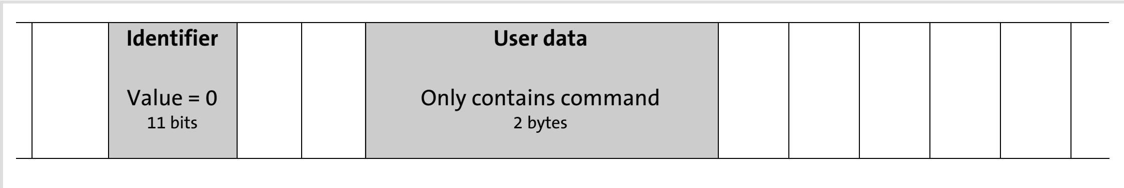


Fig. 8-2 Telegram for switching over the communication phases

A bus node, the network master, carries out the changes between the communication phases for the whole network.

With code **C1852 / C2352** (📖 125) you can set up the communication module for master or slave operation.

After an adjustable boot-up time in **C1856/1 / C2356/1** in master operation, (📖 130) the NMT command *Start_Remote_Node* is transmitted, which puts all nodes into the ”Operational” status.

Note!

Only a change to ”Operational” status enables communication via the process data!

Example:

If all nodes connected to the bus are to be switched from the ”Pre-Operational” communication status to the ”Operational” communication status via the CAN master, the identifier and the user data must have the following values in the transmission telegram:

- ► Identifier: 0x00 (broadcast telegram)
- ► User data: 0x0100

State transitions

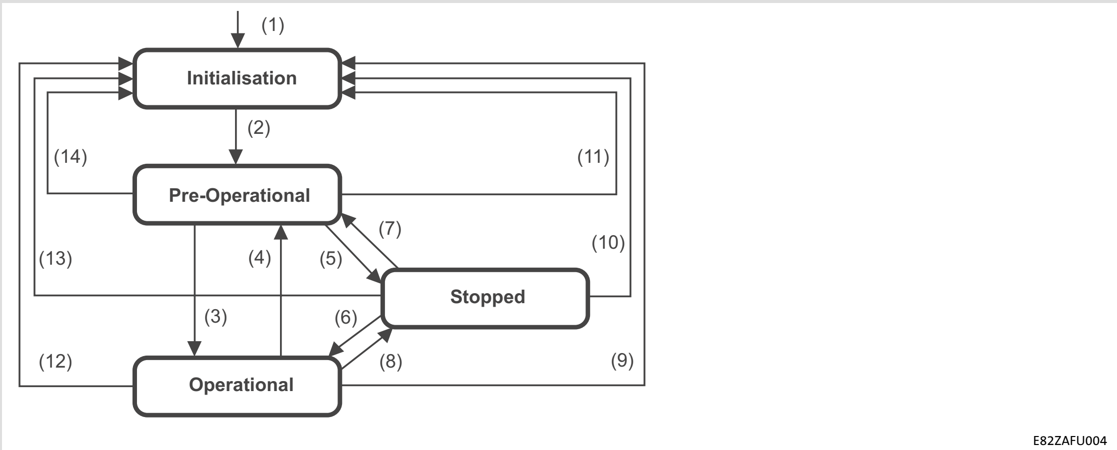


Fig. 8-3 State transitions in CAN network (NMT)

Status transition	Command (hex)	Network status after change	Effects on process and parameter data after the status change
(1)	-	Initialisation	Initialisation starts automatically when the mains is switched on. During initialisation, the drive does not take part in the data transfer. After initialisation has been completed, the node sends a boot-up message with an individual identifier on the CAN bus and changes automatically to the status ”Pre-operational”.
(2)	-	Pre-operational	In this phase, the master determines the way in which the controller takes part in the communication.
From that moment on, the master changes the states for the whole network. A target address, which is part of the command, specifies the receiver(s).			
(3), (6)	0x01yy	Operational	Network management telegrams, sync, emergency, process data (PDOs) and parameter data (SDOs) active (like ”Start remote node”) Optional: When the status is changed, event and time-controlled process data (PDOs) will be sent once.
(4), (7)	0x80yy	Pre-operational	Network management telegrams, sync, emergency and parameter data (SDOs) active (like ”Enter pre-operational state”)
(5), (8)	0x02yy	Stopped	Only network management telegrams can be received.
(9)	0x81yy	Initialisation	Initialises all parameters in the communication module with the stored values (corresponds to ”Reset node”)
(10)			
(11)			
(12)	0x82yy		Initialises communication-relevant parameters (CIA DS 301) in the communication module with the stored values (corresponds to ”Reset communication”)
(13)			
(14)			

yy = 00 In case of this assignment, the telegram addresses all devices connected. The state of all devices can be changed at the same time.

yy = node ID If a node address is given, only the state of the device with the corresponding address will be changed.

9 Process data transfer

Agreements

- Process data telegrams between host (master) and controller (slave) are distinguished as follows with regard to their direction:
  – Process data telegrams **to** the controller
  – Process data telegrams **from** the controller
- In CANopen, the process data objects are named from the node's view:
  – RPDOx: A process data object received by a node
  – TPDOx: A process data object transmitted by a node

9.1 Available process data objects

Depending on the basic device used, up to 3 RPDOs and TPDOs can be available.

Tip!

From the masters's view, the following terms are used for process data transfer with the standard device and the plugged communication module:

- The master sends the process data output words (POWs) as process output data to the standard device
- The master receives the process data input words (PIWs) as process input data from the standard device.

Process data telegram to the controller (RPDO)

The identifier of the process data telegram includes the node address of the controller. The telegram has a maximum user data length of 8 bytes. This chapter describes which user data will be evaluated for the controllers.

The CAN bus is connected to the automation interface X1 via the communication module. X1 is connected to the function block AIF-IN. Here, the user data is transformed into corresponding signal types in order to use them for further function blocks. The control word is especially important for the controller. It contains the drive setpoint in user data bytes 1 and 2.

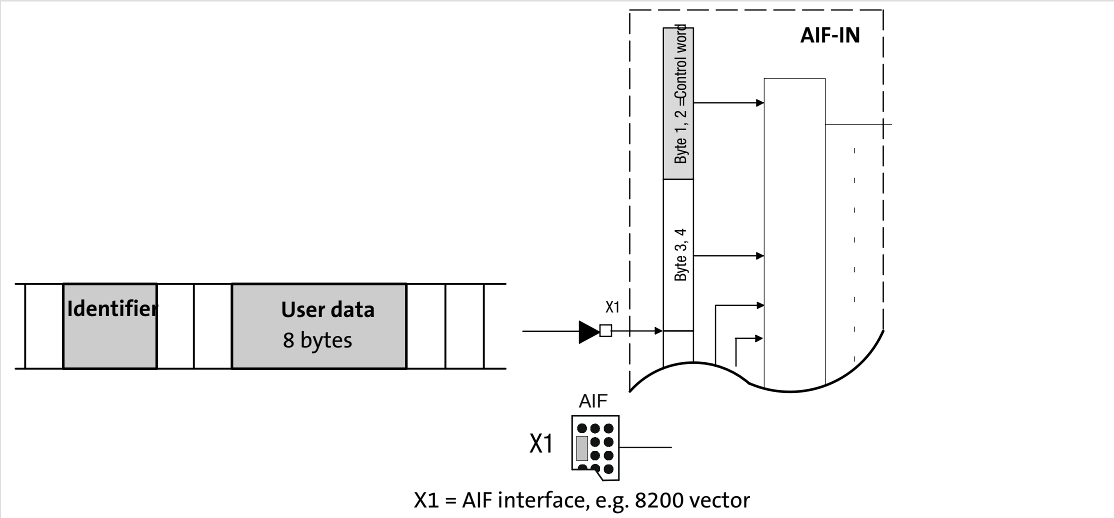


Fig. 9-1 Example: Process data telegram to controller

Process data telegram from the controller (TPDO)

For the cyclic process data telegrams **from** the controller, the function block to be used is called AIF-OUT. The status word (bytes 1 and 2) included in the process data telegram is transmitted on the CAN bus to the master via this function block.

9.2 Configuring process data channel

Selection of the setpoint source

Drive controller 8200 vector

The selection of the setpoint source for these controllers is determined with code C0001 (index: 0x5FFE). For process data evaluation the code C0001 must be set to 3 when operating the controller with the communication module.

The setpoint source is the process channel which overwrites the frequency setpoint (C0046) and the control word (C0135) (see documentation for 8200 vector).

Note!
Please observe that the setpoint source selection (C0001) must be set equally in all parameter sets.

Drive controller 93XX

The 9300 controllers do not offer a setpoint source selection which can be set by one code only. For this purpose, you only need to connect preconfigured function blocks in order to adapt the controller to the drive task without being an expert in programming.

The user himself is able to carry out the interconnection. However, it is recommended to use the preconfigurations provided by Lenze, which are saved in the read-only memory of the controller. The Lenze preconfigurations (code C0005) define which source (terminal, keyboard, communication module) overwrites the frequency setpoint and the control word.

For operation via CAN bus, the value to be set under code C0005 must be set to ”xxx3” (x = wildcard for selected preconfiguration).

9300 system manual
Here, you can find more detailed information.

PLC devices: 9300 Servo PLC / Drive PLC / ECSxA

For communicating via a communication module (e.g. EMF2178IB), it is necessary that the system blocks AIF-IN/OUT1 ... 3 and, if required, the AIF management are integrated into the control configuration of the IEC61131 project.

Axis modules of the ECS series

Operating instructions for the axis modules of the ECS series
Here you can find detailed information on the process data configuration.

9.3 Cyclic process data objects

Synchronisation of cyclic process data

The ”sync telegram” is used to ensure that the process data can be cyclically read by the controller and will be accepted by the controller.

The sync telegram is the trigger point for accepting data in the controller and activates the sending process from the controller. For cyclic process data processing, the sync telegram must be generated accordingly.

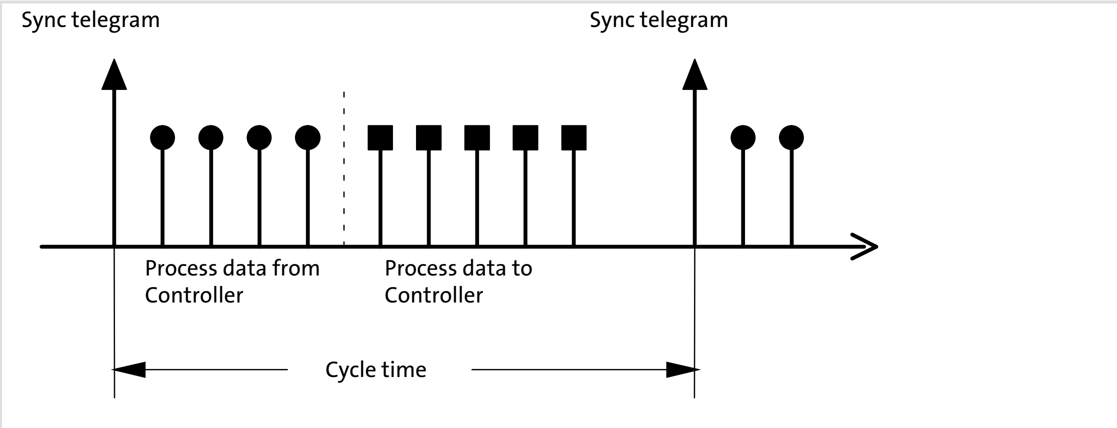


Fig. 9-2 Sync telegram

After the sync telegram has been received, the controllers send the cyclic process data to the master. The master reads them as process input data.

When the sending process has been completed, the controllers receive the process output data (from the master).

All other telegrams (e. g. parameters or event-controlled process data) are acyclically accepted by the controllers after transmission has been completed.

Acyclic data have not been taken into account for the above figure. When selecting the cycle time, they must be considered.

9.3.1 Process data signals of Lenze controllers

9.3.1.1 Process data signals for 8200 vector frequency inverters

A change of code C0001 to 3 preconfigures the process data words in the controller.

Note!
Frequency and speed values are normalised with ±24000 ≡ ±480 Hz.

Process data telegram to drive

User data (up to 8 bytes)							
Byte 1	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**	**Byte 8**
Control word Low byte	**Control word** High byte	**AIF-IN.W1** Low byte	**AIF-IN.W1** High byte	**AIF-IN.W2** Low byte	**AIF-IN.W2** High byte	**xx**	**xx**

Byte description:

Byte	Content	Description
1	C0135 Control word, low byte	Bits 0 ... 7 of the control word under C0135 (📖 145) are entered here.
2	C0135 Control word, high byte	Bits 8 ... 15 of the control word under C0135 (📖 145) are entered here.
3	AIF-IN.W1, low byte	AIF-IN.Wx is parameterised under code C0412.
4	AIF-IN.W1, high byte	
5	AIF-IN.W2, low byte	
6	AIF-IN.W2, high byte	
7 / 8	xx	No evaluation of these data, any content possible

Process data telegram from drive

User data (up to 8 bytes)							
Byte 1	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**	**Byte 8**
Status word Low byte	**Status word** High byte	**AIF-OUT.W1** Low byte	**AIF-OUT.W1** High byte	**AIF-OUT.W2** Low byte	**AIF-OUT.W2** High byte	**xx**	**xx**

Byte description:

Byte	Content	Explanation
1	C0150 Status word, low byte	Bits 0 ... 7 of the status word are entered here under C0150 (📖 145).
2	C0150 Status word, high byte	Bits 8 ... 15 of the status word are entered here under C0150 (📖 145).
3	AIF-OUT-W1 Low byte	AIF-OUT.Wx is parameterised under code C0421.
4	AIF-OUT.W1 High byte	
5	AIF-OUT-W2 Low byte	
6	AIF-OUT.W2 High byte	
7 / 8	xx	No evaluation of these data, any content possible

Device control word AIF-CTRL for 8200 vector (C0135, index 0x5F78)

Bit	Assignment (Lenze setting) C0001 = 3 with C0007 ≤ 51	C0001 = 3 with C0007 > 51	Set under C0410/...
0 / 1	**JOG values** Bit 1 / Bit 0: 0 0 — 00 = C0046 active 0 1 — 01 = NSET1-JOG1 (C0037) active 1 0 — 10 = NSET1-JOG2 (C0038) active 1 1 — 11 = NSET1-JOG3 (C0039) active	**Freely configurable**	1 2
2	**Current direction of rotation** (DCTRL1-CW/CCW) 0 Not active 1 Active	**Freely configurable**	3
3	**Quick stop (QSP)** (AIF-CTRL-QSP) 0 Not active 1 Active	**Quick stop (QSP)** (AIF-CTRL-QSP) 0 Not active 1 Active	4
4	**Stop ramp function generator** (NSET1-RFG1-STOP) 0 Not active 1 Active	**Freely configurable**	5
5	**Ramp function generator input = 0** (NSET1-RFG1-0) 0 Not active 1 Active	**Freely configurable**	6
6	**UP function motor potentiometer** (MPOT1-UP) 0 Not active 1 Active	**Freely configurable**	7
7	**DOWN function motor potentiometer** (MPOT1-DOWN)	**Freely configurable**	8
8	**Freely configurable**	**Freely configurable**	9
9	**Controller inhibit** (AIF-CTRL-CINH) 0 Not active 1 Active	**Controller inhibit** (AIF-CTRL-CINH) 0 Not active 1 Active	10
10	**External fault** (AIF-CTRL-TRIP-SET) 0 Not active 1 Active	**External fault** (AIF-CTRL-TRIP-SET) 0 Not active 1 Active	11
11	**Reset fault** (AIF-CTRL-TRIP-RESET) 0 -> 1 Edge from 0 to 1	**Reset fault** (AIF-CTRL-TRIP-RESET) 0 -> 1 Edge from 0 to 1	12
12	**Change over parameter set** (DCTRL1-PAR2/4) 0 Not active 1 Active	**Freely configurable**	13
13	**Change over parameter set** (DCTRL1-PAR3/4) 0 Not active 1 Active	**Freely configurable**	14
14	**DC injection brake** (MCTRL1-DCB) 0 Not active 1 Active	**Freely configurable**	15
15	**Freely configurable**	**Freely configurable**	16

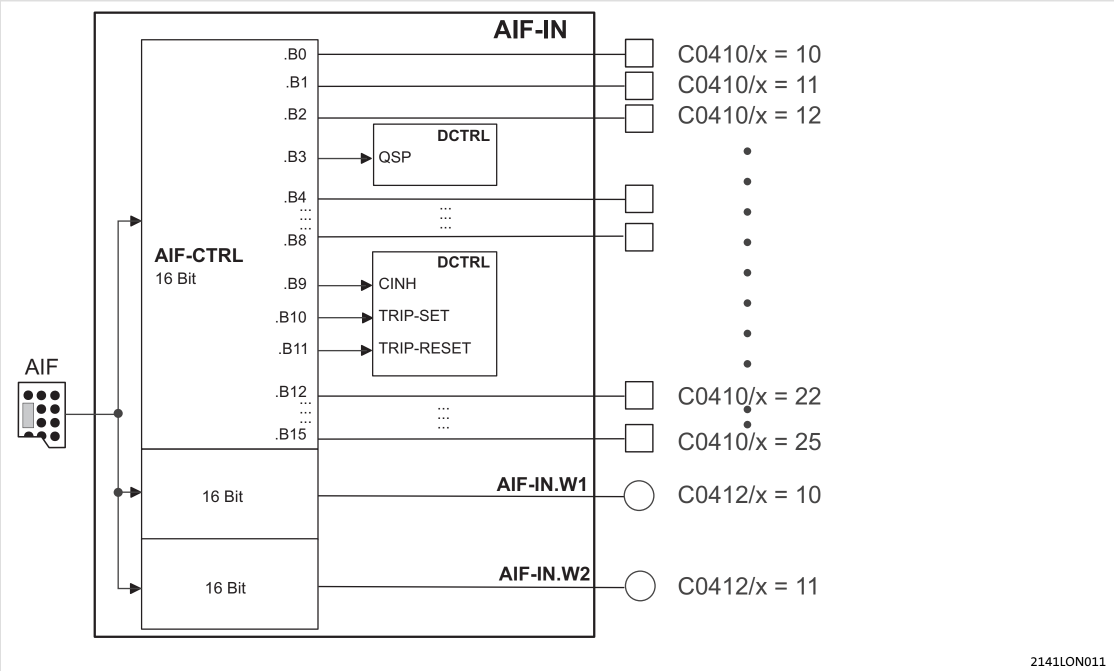


Fig. 9-3 System block AIF-IN in 8200 vector (freely configurable assignment)

Device status word AIF-STAT for 8200 vector (C0150, index 0x5F69)

Bit	Assignment (Lenze setting)	Set under C0417/...
0	**Current parameter set** (DCTRL-PAR-B0)	1
1	**Pulse inhibit** (DCTRL1-IMP)	2
2	**I_{max} limit** (MCTRL1-IMAX)	3
3	**Output frequency = frequency setpoint** (MCTRL1-RFG1=NOUT)	4
4	**Ramp function generator input = ramp function generator output 1** (NSET1-RFG1-I=0)	5
5	**Q_{min} threshold** (PCTRL1-QMIN)	6
6	**Output frequency = 0** (DCTRL1-NOUT=0)	7
7	**Controller inhibit** (DCTRL1-CINH)	8
8 ... 11	**Device status** (DCTRL1-Stat*1 ... STAT*8)	Reserved
12	**Overtemperature warning** (DCTRL1-OH-WARN)	13
13	**DC-bus overvoltage** (DCTRL1-OV)	14
14	**Direction of rotation** (DCTRL1-CCW)	15
15	**Ready for operation** (DCTRL1-RDY)	16

Device status, bits 8 ... 11:

Bit 11	10	9	8	
0	0	0	0	Device initialisation
0	0	1	0	Switch-on inhibit
0	0	1	1	Operation inhibited
0	1	0	0	Flying restart circuit active
0	1	0	1	DC injection brake active
0	1	1	0	Operation enabled
0	1	1	1	Message active
1	0	0	0	Fault active
1	1	1	1	Communication with basic device not possible

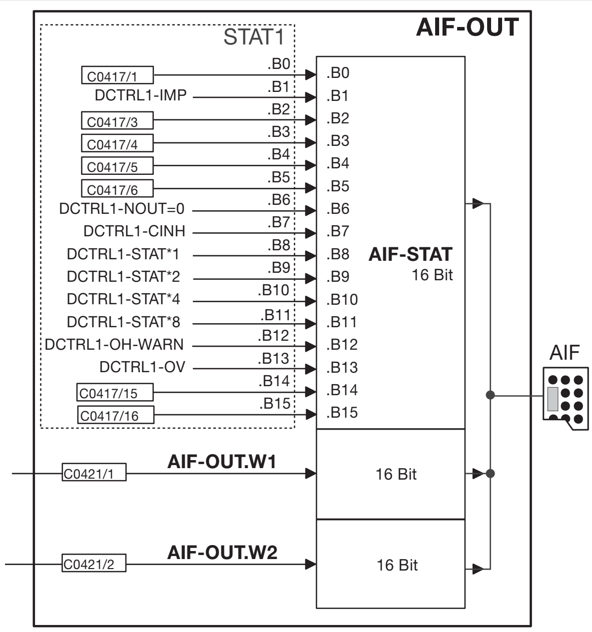


Fig. 9-4 System block AIF-OUT in 8200 vector (freely programmable assignment)

9.3.1.2 Process data signals for 9300 servo inverter

The assignment of the process data for the 93XX controller can be changed by reconfiguring the function blocks AIF-IN and AIF-OUT.

9300 system manuals
Here you can find detailed information on the function blocks and signal configurations.

Function block AIF-IN

The function block AIF-IN determines the input data of the controller as data interface for the EMF2178IB communication module.

Process data telegram to drive

User data (up to 8 bytes)							
Byte 1	2nd byte	3rd byte	Byte 4	Byte 5	Byte 6	7th byte	Byte 8
Control word Low byte	**Control word** High byte	**AIF-IN.W1** Low byte	**AIF-IN.W1** High byte	**AIF-IN.W2** Low byte	**AIF-IN.W2** High byte	**AIF-IN.W3** Low byte	**AIF-IN.W3** High byte
🕮 57							

AIF-IN.W1 to AIF-IN.W3 depend on the signal configuration selected under code C0005. Under code C0005 you can preconfigure the signals of the control word and the status word.

In the controller, other signals can be assigned to AIF-IN.W1 to AIF-IN.W3. For this, the function block configuration described in the 93XX System Manual is used.

Signal configuration (C0005)		**AIF-IN.W1**	**AIF-IN.W2**	**AIF-IN.W3**	**AIF-IN.D1**
Speed control	1003 / 1013 / 1113	NSET-N Speed setpoint 100 % = 16383	Not assigned	Not assigned	Not assigned
Torque control	4003 / 4013 / 4113	MCTRL-MADD Torque setpoint 100 % = 16383			
LF master	5003 / 5013 / 5113	NSET-N Speed setpoint 100 % = 16383			
LF slave rail	6003 / 6013 / 6113	DFSET-A-TRIM Phase trimming	DFSET-N-TRIM Speed trimming		
LF slave cascade	7003 / 7013 / 7113	DFSET-VP-DIV LF factor	DFSET-A-TRIM Phase trimming		
cam profiler	1xxx3	YSET1-FACT	Not assigned		
Positioning	2xxx3	Not assigned			
vector control	1xx3 / 2xx3 / 3xx3 / 5xx3 / 100x3	NLIM-IN1			
vector control	4xx3	NCTRL-MADD			
vector control	6xx3	DFSET-A-TRIM	DFSET-N-TRIM		
vector control	7xx3 / 8xx3 / 9xx3	DFSET-VP-DIV	DFSET-A-TRIM		
vector control	100x3	NLIM-IN1	Not assigned		
vector control	110x3	Not assigned			

Control word for 93XX

9300	9300 servo inverter				9300 position controller	9300 cam profiler	9300 vector		
C0005	1xx3	4xx3	5xx3	6xx3, 7xx3	2xx3	xxx3	1xxx, 2xxx, 3xxx, 5xxx, 10xxx, 11xxx	4xx3	6xx3, 7xx3
0	NSET-JOG*1	Not assigned	NSET-JOG*1	Not assigned	Not assigned	CSEL1-CAM*1	NSET-JOG*1	Not assigned	Not assigned
1	NSET-JOG*2	Not assigned	NSET-JOG*2	Not assigned	Not assigned	CSEL1-CAM*2	NSET-JOG*2	Not assigned	Not assigned
2	NSET-N-INV	NSET-N-INV	NSET-N-INV	NSET-N-INV	Not assigned	CSEL1-CAM*4	NSET-N-INV	Not assigned	Not assigned
3	AIF-CTRL.QSP	AIF-CTRL.QSP	AIF-CTRL.QSP	AIF-CTRL.QSP	AIF-CTRL.QSP	AIF-CTRL.QSP	AIF-CTRL.QSP	AIF-CTRL.QSP	AIF-CTRL.QSP
4	NSET-RFG-STOP	NSET-RFG-STOP	NSET-RFG-STOP	NSET-RFG-STOP	POS-PRG-START	CSEL1-EVENT	NSET-RFG-STOP	NSET-RFG-STOP	Not assigned
5	NSET-RFG-0	NSET-RFG-0	NSET-RFG-0	NSET-RFG-0	POS-PRG-STOP	CDATA-CYCLE	NSET-RFG-0	NSET-RFG-0	Not assigned
6	Not assigned	Not assigned	Not assigned	Not assigned	Not assigned	CSEL1-LOAD	Not assigned	Not assigned	Not assigned
7	Not assigned	Not assigned	Not assigned	Not assigned	POS-PRG-RESET	CSEL1-LOAD	Not assigned	Not assigned	Not assigned
8	Not assigned	Not assigned	Not assigned	Not assigned	Not assigned	Not assigned	Not assigned	Not assigned	Not assigned
9	AIF-CTRL.CINH	AIF-CTRL.CINH	AIF-CTRL.CINH	AIF-CTRL.CINH	AIF-CTRL.CINH	AIF-CTRL.CINH	AIF-CTRL.CINH	AIF-CTRL.CINH	AIF-CTRL.CINH
10	AIF-CTRL.TRIP-SET	AIF-CTRL.TRIP-SET	AIF-CTRL.TRIP-SET	AIF-CTRL.TRIP-SET	AIF-CTRL.TRIP-SET	AIF-CTRL.TRIP-SET	AIF-CTRL.TRIP-SET	AIF-CTRL.TRIP-SET	AIF-CTRL.TRIP-SET
11	AIF-CTRL.TRIP-RESET	AIF-CTRL.TRIP-RESET	AIF-CTRL.TRIP-RESET	AIF-CTRL.TRIP-RESET	AIF-CTRL.TRIP-RESET	AIF-CTRL.TRIP-RESET	AIF-CTRL.TRIP-RESET	AIF-CTRL.TRIP-RESET	AIF-CTRL.TRIP-RESET
12	DCTRL-PAR*1	DCTRL-PAR*1	DCTRL-PAR*1	DCTRL-PAR*1	POS-PS-CANCEL	Not assigned	DCTRL-PAR*1	DCTRL-PAR*1	DCTRL-PAR*1
13	DCTLR-PAR-LOAD	DCTLR-PAR-LOAD	DCTLR-PAR-LOAD	DCTLR-PAR-LOAD	POS-PARAM-RD	Not assigned	DCTLR-PAR-LOAD	DCTLR-PAR-LOAD	DCTLR-PAR-LOAD
14	NSET-Ti*1	NSET-JOG*1	REF-ON	REF-ON	POS-LOOP-ONH	Not assigned	NSET-Ti*1	NSET-JOG*1	Not assigned
15	NSET-Ti*2	NSET-JOG*2	NSET-Ti*1	Not assigned	POS-STBY-STP	Not assigned	NSET-Ti*2	NSET-JOG*2	Not assigned

Note!

The individual bit-control commands of the control word depend on other bit positions.

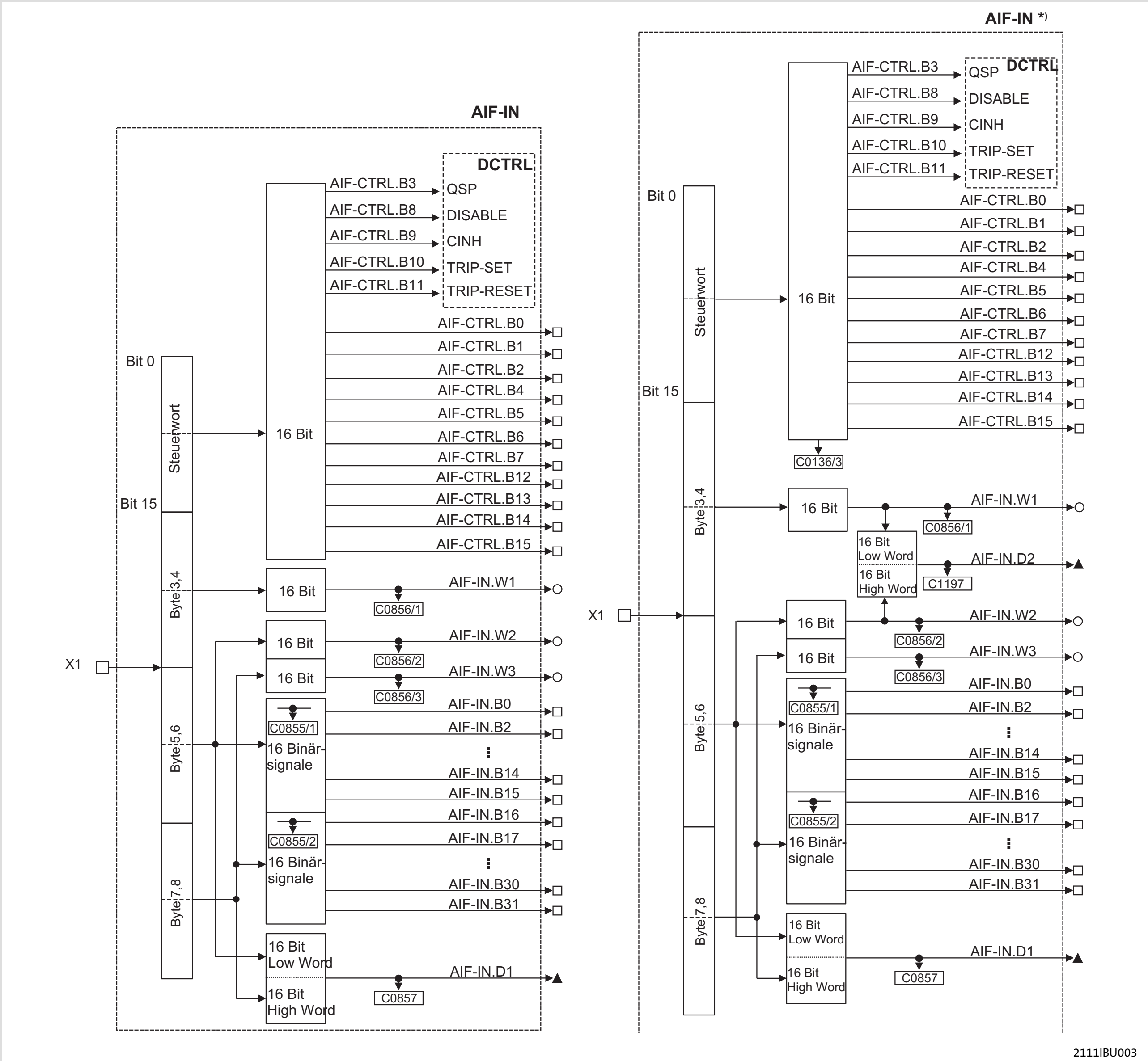


Fig. 9-5 Function blocks AIF-IN and AIF-IN*)
AIF-IN*) is available for the 9300 technology variants servo inverter, positioning controller and cam profiler as of software version 2.0. AIF-IN.D2 is new.

Function block AIF-OUT

The function block AIF-OUT determines the output data of the controller as data interface for the EMF2178IB communication module.

Process data telegram from drive

User data (up to 8 bytes)							
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Status word Low byte	**Status word** High byte	**AIF-OUT.W1** Low byte	**AIF-OUT.W1** High byte	**AIF-OUT.W2** Low byte	**AIF-OUT.W2** High byte	**AIF-OUT.W3** Low byte	**AIF-OUT.W3** High byte
🕮 60							

AIF-OUT.W1 to AIF-OUT.W3 depend on the signal configuration selected under code C0005. Under code C0005 you can preconfigure the signals of the control word and the status word.

In the controller, other signals can be assigned to AIF-OUT.W1 to AIF-OUT.W3. For this, the function block configuration described in the 93XX System Manual is used.

Signal configuration (C0005)		AIF-OUT.W1	AIF-OUT.W2	AIF-OUT.W3	AIF-OUT.D1
Speed control	1003 / 1013 / 1113	MCTRL-NACT Actual speed value 100 % = 16383	MCTRL-MSET2 Torque display 100 % = 16383	MCTRL-NSET2 Speed controller input 100 % = 16383	Not assigned
Torque control	4003 / 4013 / 4113	MCTRL-MSET2 Torque display 100 % = 16383	MCTRL-NACT Actual speed in % 100 % = 16383	MCTRL-NSET2 Speed controller input 100 % = 16383	
LF master	5003 / 5013 / 5113	MCTRL-NACT Actual speed value 100 % = 16383	MCTRL-MSET2 Torque display 100 % = 16383	MCTRL-NSET2 Speed controller input 100 % = 16383	
LF slave rail	6003 / 6013 / 6113	MCTRL-NACT Actual speed value 100 % = 16383	MCTRL-PHI-ACT Angle actual value	MCTRL-MSET2 Torque setpoint in % 100 % = 16383	
LF slave cascade	7003 / 7013 / 7113	MCTRL-NACT Actual speed value 100 % = 16383	MCTRL-PHI-ACT Angle actual value	MCTRL-MSET2 Torque setpoint in % 100 % = 16383	
Cam profiler	1xxx3	MCTRL-NACT Actual speed value 100 % = 16383	Not assigned	Not assigned	
Positioning	2xxx3	MCTRL-NACT Actual speed value 100 % = 16383	Not assigned	Not assigned	
vector control	1xx3 / 4xx3 / 5xx3 / 10xx3	MCTRL-NACT Actual speed value 100 % = 16383	MCTRL-IACT	MCTRL-NSET2 Speed controller input 100 % = 16383	
vector control	6xx3 / 7xx3 / 8xx3 / 9xx3	MCTRL-NACT Actual speed value 100 % = 16383	MCTRL-PHI-ANA	MCTRL-MSET2 Torque setpoint in % 100 % = 16383	
vector control	110x3	Not assigned	Not assigned	Not assigned	

Status word for 93XX

9300	9300 servo inverter				9300 position controller	9300 cam profiler	9300 vector		
C0005	**1xx3**	**4xx3**	**5xx3**	**6xx3, 7xx3**	**2xx3**	**1xx3**	**1xxx, 2xxx, 3xxx, 5xxx, 10xxx, 11xxx**	**4xxx**	**6xxx, 7xxx, 8xxx, 9xxx**
0	DCTRL-PAR1-0	DCTRL-PAR1-0	DCTRL-PAR1-0	DCTRL-PAR1-0	Not assigned	CERR1-ERR	DCTRL-PAR1-0	DCTRL-PAR1-0	DCTRL-PAR1-0
1	DCTRL-IMP	DCTRL-IMP	DCTRL-IMP	DCTRL-IMP	DCTRL-IMP	DCTRL-IMP	DCTRL-IMP	DCTRL-IMP	DCTRL-IMP
2	MCTRL-IMAX	MCTRL-IMAX	REF-OK	REF-OK	POS-REF-OK	MCTRL-IMAX	MCTRL-IMAX	MCTRL-IMAX	MCTRL-IMAX
3	MCTRL-MMAX	Not assigned	MCTRL-MMAX	Not assigned	Not assigned	MCTRL-MMAX	MCTRL-MMAX	MCTRL-IMAX negated	MCTRL-MMAX
4	NSET-RFG-I=0	MCTRL-IMAX negated	NSET-RFG-I=0	MCTRL-IMAX negated	MCTRL-MMAX negated	DCTRL-TRIP	NSET-RFG-I=0	NSET-RFG-I=0	NSET-QSP-OUT
5	QMIN	QMIN	REF-BUSY	REF-BUSY	POS-IN-TARGET	CDATA-X0	QMIN	QMIN	QMIN
6	DCTRL-NACT=0	DCTRL-NACT=0	DCTRL-NACT=0	DCTRL-NACT=0	DCTRL-NACT=0	DCTRL-NACT=0	DCTRL-NACT=0	DCTRL-NACT=0	DCTRL-NACT=0
7	DCTRL-CINH	DCTRL-CINH	DCTRL-CINH	DCTRL-CINH	DCTRL-CINH	DCTRL-CINH	DCTRL-CINH	DCTRL-CINH	DCTRL-CINH
8 ... 11	Device status: 0b0000 = 0 0b0010 = 2 0b0011 = 3 0b0100 = 4 0b0101 = 5 0b0110 = 6 0b0111 = 7 0b1000 = 8 0b1010 = 10	 Device initialisation Switch-on inhibit Operation inhibited Flying restart circuit active DC injection brake active Operation enabled Message active Fault active Fail-QSP (only 9300 servo position controller)							
12	DCTRL-WARN	DCTRL-WARN	DCTRL-WARN	DCTRL-WARN	DCTRL-WARN	DCTRL-WARN	DCTRL-WARN	DCTRL-WARN	DCTRL-WARN
13	DCTRL-MESS	DCTRL-MESS	DCTRL-MESS	DCTRL-MESS	DCTRL-MESS	DCTRL-MESS	DCTRL-MESS	DCTRL-MESS	DCTRL-MESS
14	DCTRL-CW/CCW	DCTRL-CW/CCW	DCTRL-CW/CCW	Not assigned	DCTRL-AIFL-QSP	DCTRL-CW/CCW	DCTRL-CW/CCW	DCTRL-CW/CCW	DCTRL-CW/CCW
15	DCTRL-RDY	DCTRL-RDY	DCTRL-RDY	DCTRL-RDY	DCTRL-RDY	DCTRL-RDY	DCTRL-RDY	DCTRL-RDY	DCTRL-RDY

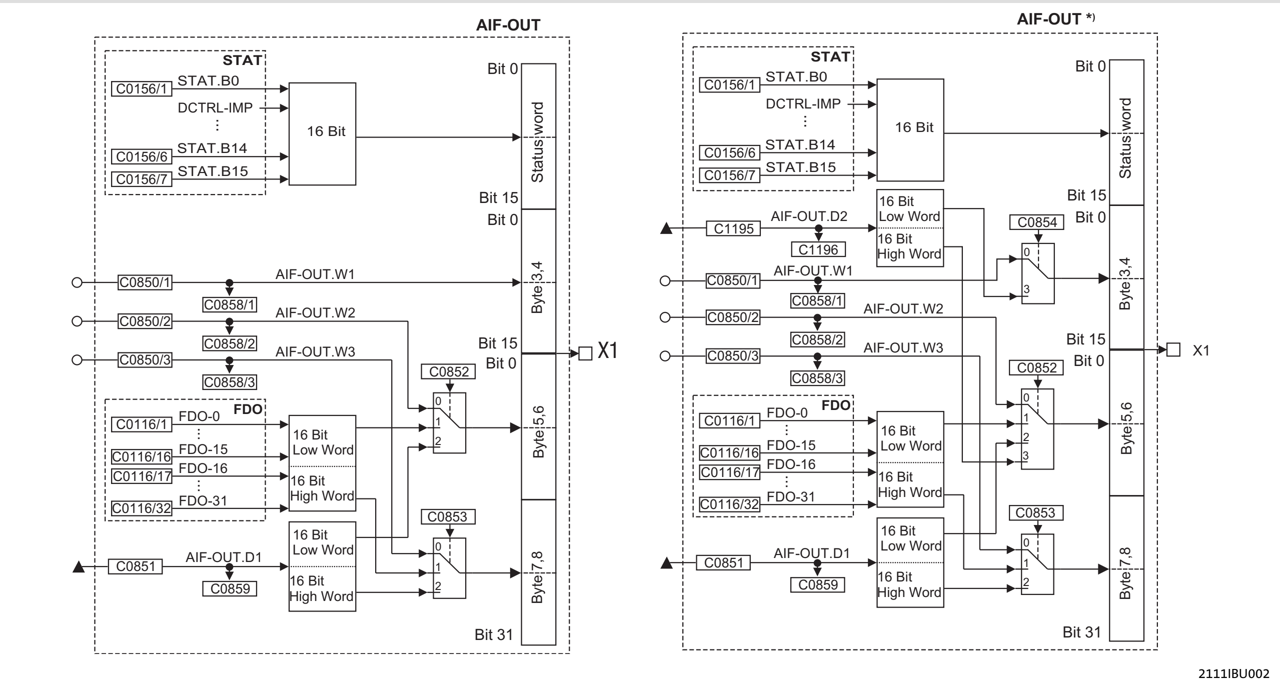


Fig. 9-6 Function blocks AIF-OUT and AIF-OUT*)
AIF-OUT*) is available for the 9300 technology variants servo inverter, positioning controller and cam profiler as of software version 2.0. AIF-OUT.D2 is new.

9.3.1.3 Process data signals for 9300 Servo PLC, Drive PLC, ECSxA

For more information ...

please see the chapter "System blocks, AIF1_IO_AutomationInterface" in the Manuals and Operating Instructions for the 9300 Servo PLC, Drive PLC and ECSxA axis module.

Note!

Only 9300 Servo PLC & ECSxA:

Perform the following linkings in the PLC program of the drive controller:

- AIF1_wDctrlCtrl →DCTRL_wAIF1Ctrl
- DCTRL_wStat → AIF1_wDctrlStat

Process data telegram to drive

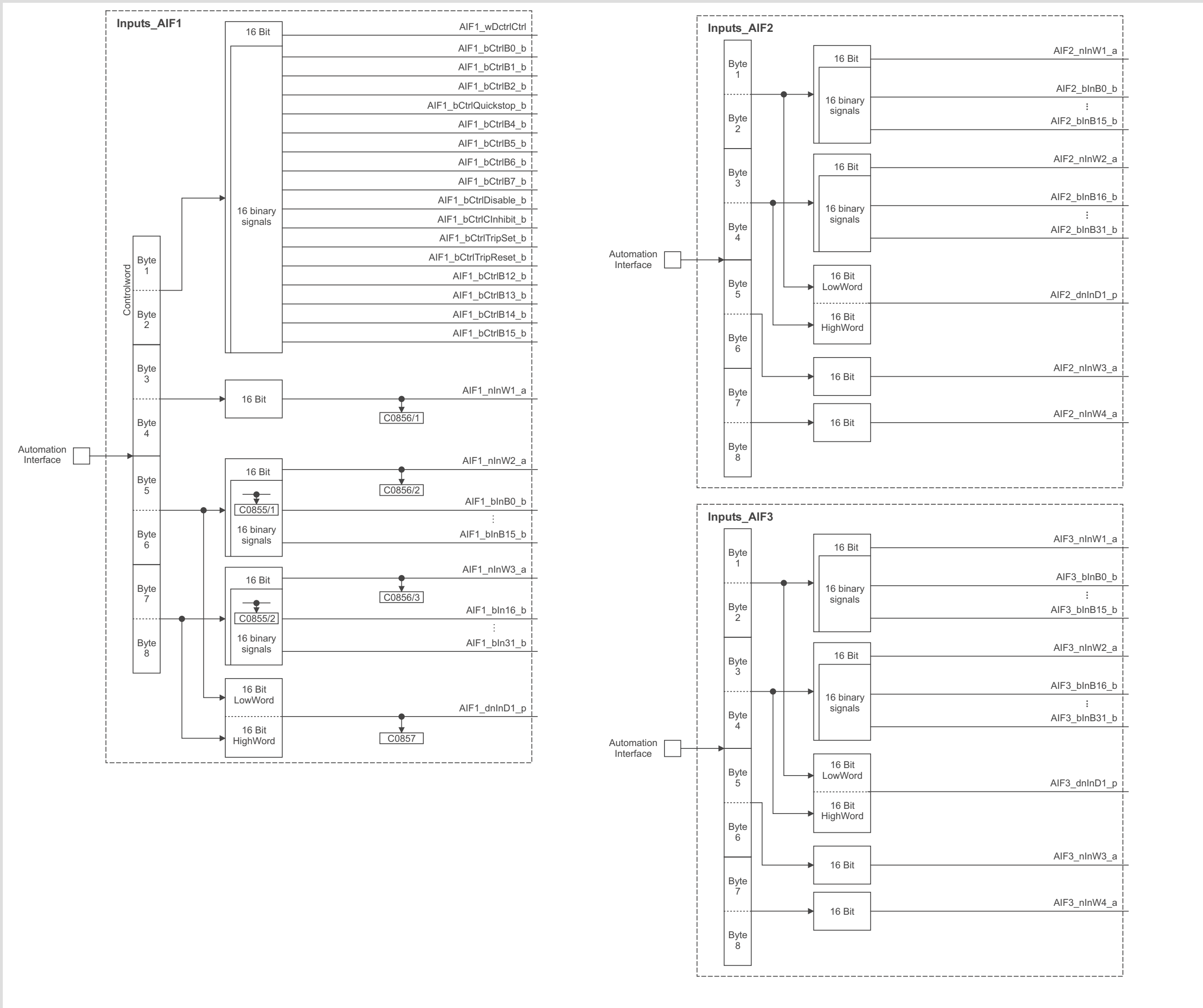


Fig. 9-7 Function blocks AIF1-IN, AIF2-IN and AIF3-IN

Process data telegram from drive

The following data can be assigned to the PIW data (TPDOs):

Designation / variable name	Explanation
Device status word (AIF1_DctrlStat)	AIF word 1
AIF_nOutW1_a	AIF word 2
AIF_nOutW2_a	AIF word 3
AIF_nOutW3_a	AIF word 4
AIF2_nOutW1_a	AIF word 5
AIF2_nOutW2_a	AIF word 6
AIF2_nOutW3_a	AIF word 7
AIF2_nOutW4_a	AIF word 8
AIF3_nOutW1_a	AIF word 9
AIF3_nOutW2_a	AIF word 10
AIF3_nOutW3_a	AIF word 11
AIF3_nOutW4_a	AIF word 12
AIF1_dnOutD1_p	AIF double word 1

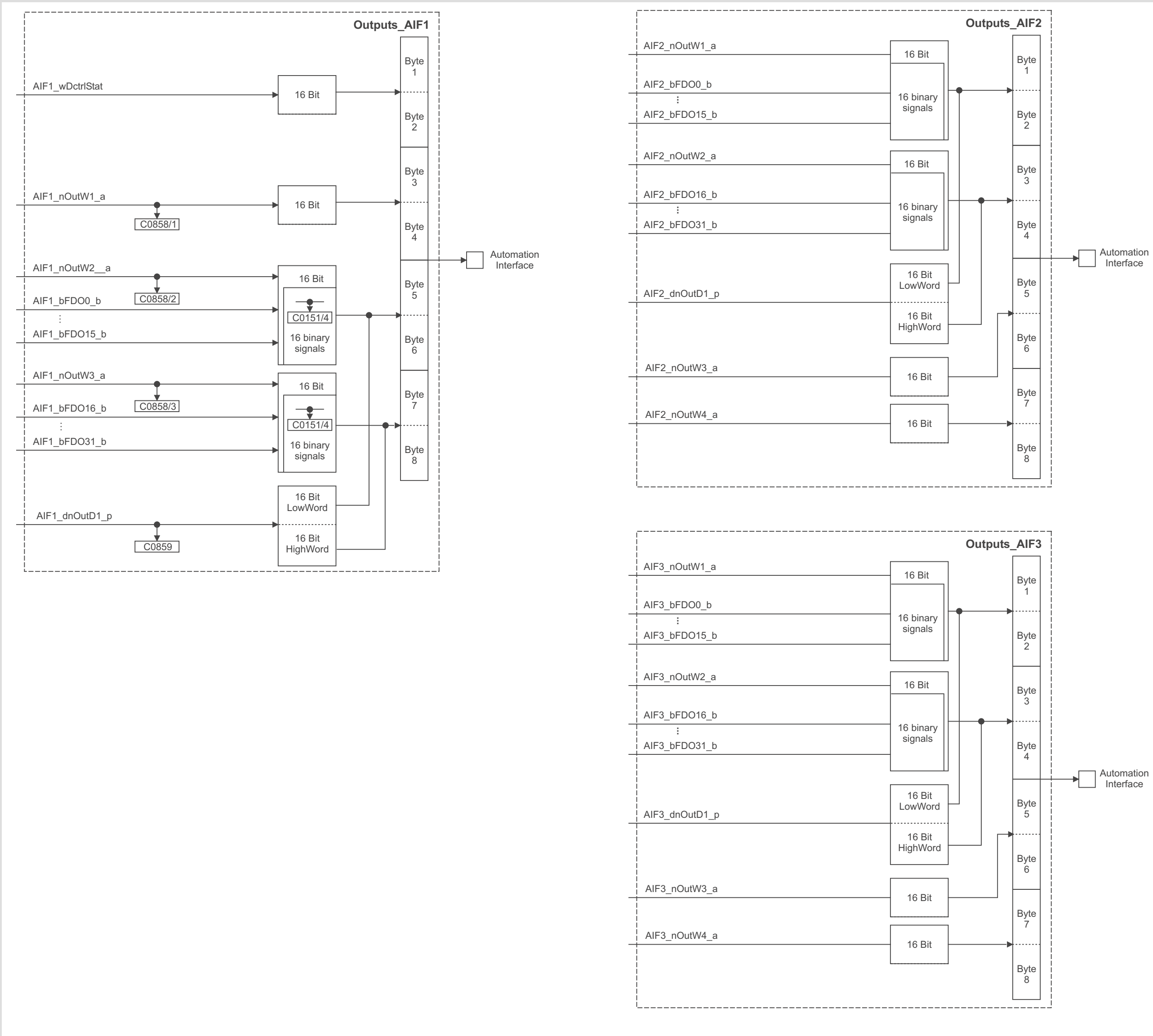


Fig. 9-8 Function blocks AIF1-OUT, AIF2-OUT and AIF3-OUT

Axis modules of the ECS series

Operating instructions for the axis modules of the ECS series
Here you can find detailed information on the process data configuration.

9.3.2 Mapping in CANopen objects (I-160x, I-1A0x)

General

Note!
Mapping is not possible in communication modules used together with old devices of the 8200 series.

The CANopen objects I-160x and I-1A0x are used to assign the CANopen process data to the internal data. In the EMF2178IB communication module an assignment to the process data of the internal AIF interface is possible. The process data volume and type depends on the standard device.

The process data length depends on the AIF mode (□ 68):

Basic device	AIF mode
8200 vector	Mode 3: 3 process data words with variable assignment
93XX	Mode 4: 4 process data words with variable assignment
9300 servo PLC	Mode 5: 12 process data words with variable assignment
ECSXX	depends on the variant (mode 4 or 5)

For mapping in the module, the master enters indices and data lengths.

Mapping structure

MSB								LSB
31	...	16	15	...	8	7	...	0
Index			Subindex			Object length DWord 0 = 0x20 Word = 0x10		

The EDS file for the EMF2178IB communication module supports the mapping.

Tip!
The current EDS file required for configuring the EMF2178IB (CANopen) communication module can be found in the download area on:
www.Lenze.com

AIF data image in codes

The AIF process data images are mapped to the following codes:

AIF process data	Code	Index [hex]
Process input data AIF-IN as 16-bit values	C1822/1 ... 12	0x58E1
Process output data AIF-OUT as 16-bit values	C1823/1 ... 12	0x58E0
Process input data AIF-IN as 32-bit values	C1824/1 ... 11	0x58DF
Process output data AIF-OUT as 32-bit values	C1825/1 ... 11	0x58DE

The codes represent the values at the AIF interface. Converted into indices, mapping refers to these codes.

<table>
<tr><th colspan="2" rowspan="2">Process data AIF-IN</th><th colspan="3">Represented as</th></tr>
<tr><th>16-bit values</th><th colspan="2">32-bit values 1)</th></tr>
<tr><td>Byte 1</td><td rowspan="8">AIF1-IN
(bytes 1 ... 8)</td><td rowspan="2">C1822/1</td><td rowspan="4">[C1824/1]</td><td rowspan="2"></td></tr>
<tr><td>2nd byte</td></tr>
<tr><td>3rd byte</td><td rowspan="2">C1822/2</td><td rowspan="4">C1824/2</td></tr>
<tr><td>Byte 4</td></tr>
<tr><td>Byte 5</td><td rowspan="2">C1822/3</td><td rowspan="4">C1824/3</td></tr>
<tr><td>Byte 6</td></tr>
<tr><td>7th byte</td><td rowspan="2">C1822/4</td><td rowspan="4">[C1824/4]</td></tr>
<tr><td>Byte 8</td></tr>
<tr><td>Byte 9</td><td rowspan="8">AIF2-IN
(bytes 1 ... 8)</td><td rowspan="2">C1822/5</td><td rowspan="4">C1824/5</td></tr>
<tr><td>Byte 10</td></tr>
<tr><td>Byte 11</td><td rowspan="2">C1822/6</td><td rowspan="4">[C1824/6]</td></tr>
<tr><td>Byte 12</td></tr>
<tr><td>Byte 13</td><td rowspan="2">C1822/7</td><td rowspan="4">[C1824/7]</td></tr>
<tr><td>Byte 14</td></tr>
<tr><td>Byte 15</td><td rowspan="2">C1822/8</td><td rowspan="4">[C1824/8]</td></tr>
<tr><td>Byte 16</td></tr>
<tr><td>Byte 17</td><td rowspan="8">AIF3-IN
(bytes 1 ... 8)</td><td rowspan="2">C1822/9</td><td rowspan="4">C1824/9</td></tr>
<tr><td>Byte 18</td></tr>
<tr><td>Byte 19</td><td rowspan="2">C1822/10</td><td rowspan="4">[C1824/10]</td></tr>
<tr><td>Byte 20</td></tr>
<tr><td>Byte 21</td><td rowspan="2">C1822/11</td><td rowspan="4">[C1824/11]</td></tr>
<tr><td>Byte 22</td></tr>
<tr><td>Byte 23</td><td rowspan="2">C1822/12</td><td rowspan="2"></td></tr>
<tr><td>Byte 24</td></tr>
</table>

1) [Cxxx/y] = Codes not relevant for the AIF modes (📖 68)

Process data	AIF-OUT	Represented as		
		16-bit values	32-bit values 1)	
Byte 1	**AIF1-OUT** (bytes 1 ... 8)	C1823/1	[C1825/1]	
2nd byte				
3rd byte		C1823/2		C1825/2
Byte 4				
Byte 5		C1823/3	C1825/3	
Byte 6				
7th byte		C1823/4		[C1825/4]
Byte 8				
Byte 9	**AIF2-OUT** (bytes 1 ... 8)	C1823/5	C1825/5	
Byte 10				
Byte 11		C1823/6		[C1825/6]
Byte 12				
Byte 13		C1823/7	[C1825/7]	
Byte 14				
Byte 15		C1823/8		[C1825/8]
Byte 16				
Byte 17	**AIF3-OUT** (bytes 1 ... 8)	C1823/9	C1825/9	
Byte 18				
Byte 19		C1823/10		[C1825/10]
Byte 20				
Byte 21		C1823/11	[C1825/11]	
Byte 22				
Byte 23		C1823/12		
Byte 24				

1) [Cxxx/y] = Codes not relevant for the AIF modes (📖 68)

CANopen indices for mapping

According to the CANopen specifications, the indices I-160x and I-1A0x are available for mapping the CAN PDOs. During mapping, the process output data of the master are mapped via the indices I-160x to the AIF input data of the basic device. The indices I-160x refer to codes C1822 and C1824. The entry contains the index of the corresponding code and the corresponding subindex for the assignment of a word or double word. The object length must be indicated accordingly.

Under the indices I-1A0x, the indices of code C1823 must be entered for words or the indices of code C1825 for double words.

Device-internal mechanisms

The user must know the AIF interface assignment to configure the mapping accordingly in the communication module. All data of the AIF interface can be placed on any position of the CAN PDOs. The maximum scope of the AIF interface is shown in the codes. The available values depend on the basic device and on the application in the basic device.

AIF interface assignment / AIF modes

Process data		Mode 3	Mode 4			Mode 5	
Byte 1	**AIF1-IN/OUT** (bytes 1 ... 8)	Control word / status word	AIF-CTRL / AIF-STAT			AIF_wDctrlCtrl / AIF_wDctrlStat	
2nd byte							
3rd byte		AIF-IN.W1 / AIF-OUT.W1	AIF-IN.W1 / AIF-OUT.W1		AIF-IN.D2 / AIF-OUT.D2	AIF1_nInW1_a / AIF1_nOutW1_a	
Byte 4							
Byte 5		AIF-IN.W2 / AIF-OUT.W2	AIF-IN.W2 / AIF-OUT.W2	AIF-IN.D1 / AIF-OUT.D1		AIF1_nInW2_a / AIF1_nOutW2_a	AIF1_dnInD1_p / AIF1_dnOutD1_p
Byte 6							
7th byte			AIF-IN.W3 / AIF-OUT.W3			AIF1_nInW3_a / AIF1_nOutW3_a	
Byte 8							
Byte 9	**AIF2-IN/OUT** (bytes 1 ... 8)					AIF2_nInW1_a / AIF2_nOutW1_a	AIF2_dnInD1_p / AIF2_dnOutD1_p
Byte 10							
Byte 11						AIF2_nInW2_a / AIF2_nOutW2_a	
Byte 12							
Byte 13						AIF2_nInW3_a / AIF2_nOutW3_a	
Byte 14							
Byte 15						AIF2_nInW4_a / AIF2_nOutW4_a	
Byte 16							
Byte 17	**AIF3-IN/OUT** (bytes 1 ... 8)					AIF3_nInW1_a / AIF3_nOutW1_a	AIF3_dnInD1_p / AIF3_dnOutD1_p
Byte 18							
Byte 19						AIF3_nInW2_a / AIF3_nOutW2_a	
Byte 20							
Byte 21						AIF3_nInW3_a / AIF3_nOutW3_a	
Byte 22							
Byte 23						AIF3_nInW4_a / AIF3_nOutW4_a	
Byte 24							

- For mode 3 (e.g. for the 8200 vector), an assignment of the first three words is useful. For the mapping, the index of code C1822 should accordingly be used with the subindices 1 to 3.
- For mode 4 (e.g. for the 93XX), an assignment of the first four words or the double words 2 and 3 is possible. Here, the representation of the double words differs from the text description in the basic device.
- For mode 5 (e.g. for the 9300 Servo PLC), all 12 words are possible and the use of double words 3, 5 and 9.
- The user is responsible for the corresponding assignment. The mapping is not checked once again during the entry.
- For a more detailed description of the entries, see indices I-160x (🕮 111) and I-1A0x (🕮 114).

10 Parameter data transfer

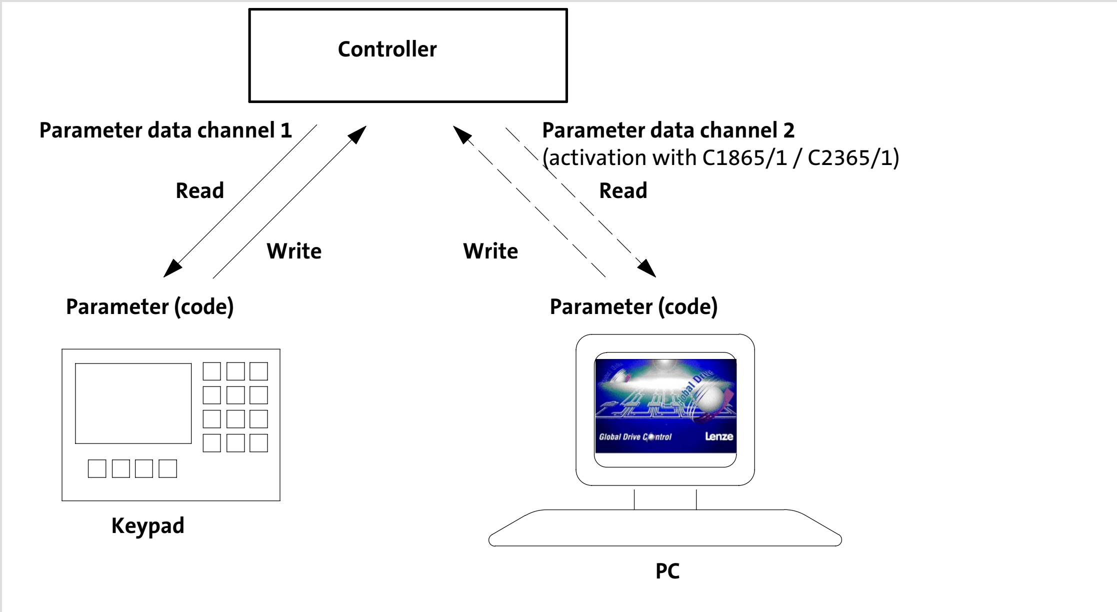


Fig. 10-1 Connection of devices for parameter setting via two parameter data channels

Parameters ...

- are values which are stored under codes in Lenze controllers.
- are, for instance, used for one-off plant settings or a change of material in a machine.
- are transferred with low priority.

Parameter data are transmitted via the system bus as SDOs (Service Data Objects) and acknowledged by the receiver. The SDOs enable read and write access to the object directory.

In the Lenze setting, one parameter data channel is available for parameter setting.

A second parameter data channel can be activated via code C1865/1 / C2365/1 or the implemented CANopen object I-1201 in order to enable the simultaneous connection of various devices. The parameter data channel 2 is **deactivated** as default.

Note!

When the second parameter data channel is activated the possible number of bus nodes is reduced.

10.1 Access to the codes of the controller

When using Lenze communication modules, you can change the features and response characteristics of any controller connected to the network using a higher-level host system(PLC, master).

In Lenze controllers, parameters to be changed are listed under codes.

Controller codes are addressed via the index when accessing the code through the communication module.

The index for Lenze code numbers is between 16576 (0x40C0) and 24575 (0x5FFF).

Documentation for the controller
Here you can find a detailed description of the codes.

Indexing of Lenze codes

Conversion formula	
Index (dec)	**Index (hex)**
24575 - Lenze code	0x5FFF - $(\text{Lenze code})_{hex}$

Example for C0001 (operating mode)	
Index (dec)	**Index (hex)**
24575 - 1 = 24574	0x5FFF - 1 = 0x5FFE

The parameter value is part of the telegram user data (see examples, 🕮 77).

10.2 Lenze parameter sets

Parameter sets are used for separately storing codes based on the need for different configurations of application processes.

Stop!

8200 vector with EMF2178IBcommunication module

- The cyclic writing of parameter data to the EEPROM is not permissible.
- If you still want to change parameter data, code C0003 must be set to ”0” after every mains switching. The parameter data are not stored in the EEPROM but as volatile data.

10.2.1 Parameter sets for 8200 vector controller

The 8200 vector controller has four parameter sets the parameters of which can be directly addressed via the INTERBUS.

Addressing

Addressing is carried out with a code offset:

- Offset ”0” addresses the parameter set 1 with the codes C0000 ... C1999.
- Offset ”2000” addresses the parameter set 2 with the codes C2000 ... C3999.
- Offset ”4000” addresses the parameter set 3 with the codes C4000 ... C5999.
- Offset ”6000” addresses the parameter set 4 with the codes C6000 ... C7999.

If a parameter is only available once (see documentation of the controller), use the code offset ”0”.

Example

Addressing of the code C0011 (maximum field frequency) in different parameter sets:

- C0011 in parameter set 1: Code no. = 11
- C0011 in parameter set 2: Code no. = 2011
- C0011 in parameter set 3: Code no. = 4011
- C0011 in parameter set 4: Code no. = 6011

Note!

Automatic saving of the changed parameter data is activated (Lenze basic setting, can be switched off via C0003).

10.2.2 Parameter sets for controller 93XX

The drive controllers 93XX feature up to four parameter sets for storage in the EEPROM for each technology variant. An additional parameter set is located in the main memory of the drive controller. This parameter set is referred to as "current" parameter set.

Addressing

- Only the current parameter set can be directly addressed via the bus.
- After power-on, parameter set 1 is automatically loaded into the current parameter set.
- Before changing the parameters of parameter sets 2 ... 4, the parameter sets must be activated.

Note!

Changes in the current parameter set are lost after switching off the drive controller.

Code C0003 is used to store the current parameter set (see documentation of the controller).

10.3 Structure of the parameter data telegram

User data (up to 8 bytes)							
Byte 1	**2nd byte**	**3rd byte**	**Byte 4**	**Byte 5**	**Byte 6**	**7th byte**	**Byte 8**
Command	**Index** Low byte	**Index** High byte	**Subindex**	**Data 1**	**Data 2**	**Data 3**	**Data 4**
				Low word		High word	
				Low byte	High byte	Low byte	High byte
				Error code			

Note!

User data are displayed in the left-justified INTEL format.
(Calculation examples: 🕮 77)

Command

The command contains the services for writing and reading parameters and information about the user data length.

Structure of a command:

	Bit 7 MSB	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0 LSB**	
Command	Command Specifier (cs)			0	Length		e	s	**Note**
Write request	0	0	1	0	x	x	1	1	User data length coding in bits 2 and 3: ● 0b00 = 4 bytes ● 0b01 = 3 bytes ● 0b10 = 2 bytes ● 0b11 = 1 byte
Write response	0	1	1	0	x	x	0	0	
Read request	0	1	0	0	x	x	0	0	
Read response	0	1	0	0	x	x	1	1	
Error response	1	0	0	0	0	0	0	0	

The following information is contained/must be entered in the command.

	4-byte data (bytes 5 ... 8)		**2-byte data** (bytes 5 and 6)		**1-byte data** (byte 5)		**Block**	
Command	**hex**	dec	**hex**	dec	**hex**	dec	**hex**	dec
Write request (Send parameters to drive)	**0x23**	35	**0x2B**	43	**0x2F**	47	Writing not possible	
Write response (Controller response to write request (acknowledgement))	**0x60**	96	**0x60**	96	**0x60**	96		
Read request (Request to read a parameter from the controller)	**0x40**	64	**0x40**	64	**0x40**	64	**0x40**	64
Read response (Response to the read request with an actual value)	**0x43**	67	**0x4B**	75	**0x4F**	79	**0x41**	65
Error response (Controller indicates a communication error)	**0x80**	128	**0x80**	128	**0x80**	128	**0x80**	128

Index low byte / Index high byte

Parameters and Lenze codes are selected with these two bytes according to the following formula:

Index = 24575 - (Lenze code number + 2000 (parameter set - 1))

Example	Calculation	Low/high byteindex entries
Code C0012 (acceleration time) in parameter set 1 is to be addressed.	24575 - 12 - 0 = 24563 = 0x5FF3	Index low byte = 0xF3 Index high byte = 0x5F
Code C0012 (acceleration time) in parameter set 2 is to be addressed.	An offset of ”2000” is to be added because of parameter set 2: 24575 - 12 - 2000 = 22563 = 0x5823	Index low byte = 0x23 Index high byte = 0x58

Subindex

Depending on the corresponding Lenze code, further subcodes may exist.

For all array codes, the value of the highest subindex can be read out via subindex 0.

Data (Data 1 ... 4)

<table>
<tr><th colspan="4">Parameter value length depending on the data format
(Data format: See ”table of attributes” in the documentation of the controller)</th></tr>
<tr><td>Parameter value
(length: 1 byte)</td><td>0x00</td><td>0x00</td><td>0x00</td></tr>
<tr><td colspan="2">Parameter value (length: 2 bytes)</td><td rowspan="2">0x00</td><td rowspan="2">0x00</td></tr>
<tr><td>Low byte</td><td>High byte</td></tr>
<tr><td colspan="4">Parameter value (length: 4 bytes)</td></tr>
<tr><td colspan="2">Low word</td><td colspan="2">High word</td></tr>
<tr><td>Low byte</td><td>High byte</td><td>Low byte</td><td>High byte</td></tr>
</table>

Note!

- Lenze parameters are mainly represented as data type FIX32 (32- bit value with sign and four decimal positions). In order to obtain integer values, the desired parameter value must be multiplied by 10000. (See ”Attribute table” in the documentation of the controller.)
- The parameters C0135 and C0150 are transferred as bit code and without factor.

Documentation of the controllers

Here you can find the descriptions of the codes (see there: ”Code table”)

Error messages

User data (up to 8 bytes)							
Byte 1	**2nd byte**	**3rd byte**	**Byte 4**	**Byte 5**	**Byte 6**	**7th byte**	**Byte 8**
Command	Index Low byte	Index High byte	Subindex	**Error code**			

- Byte 1:
  Code **128 or** 0x80 in the **command** byte indicates that an error has occurred.
- Bytes 2 ... 4:
  These bytes contain the **index** (bytes 2 and 3) and the **subindex** (byte 4) of the code in which the error has occurred.
- Bytes 5 ... 8:
  The data bytes 5 ... 8 contain the **error code**. The error code is described in reversed direction compared to the read direction.

Example: Representation of the error code 0x06040041 in bytes 5 ... 8

← Read direction of error code

0x41	**0x00**	**0x04**	**0x06**
Byte 5	Byte 6	7th byte	Byte 8
Low word		High word	
Low byte	High byte	Low byte	High byte

10.4 Error codes

Error code [hex]	Description
0x05030000	Toggle bit not changed
0x05040000	SDO protocol expired
0x05040001	Invalid or unknown client/server command specifier
0x05040002	Invalid block size (only block mode)
0x05040003	Invalid processing number (only block mode)
0x05040004	CRC error (only block mode)
0x05040005	Not enough memory
0x06010000	Object access not supported
0x06010001	Try to read writable object
0x06010002	Try to write readable object
0x06020000	Object not listed in object directory
0x06040041	Object not mapped to PDO
0x06040042	Number and length of objects to be transferred longer than PDO.
0x06040043	General parameter incompatibility
0x06040047	General internal controller incompatibility
0x06060000	Access denied because of hardware error
0x06070010	Inappropriate data type, service parameter length
0x06070012	Inappropriate data type, service parameter length exceeded
0x06070013	Inappropriate data type, service parameters not long enough
0x06090011	Subindex does not exist
0x06090030	Parameter value range exceeded
0x06090031	Parameter values too high
0x06090032	Parameter values too low
0x06090036	Maximum value falls below minimum value
0x08000000	General error
0x08000020	Data cannot be transferred or stored for the application.
0x08000021	Because of local control, data cannot be transferred or stored for the application.
0x08000022	Because of current controller status, data cannot be transferred or stored for the application.
0x08000023	Dynamic generation of object directory not successful or no object directory available (e.g. object directory generated from file, generation not possible because of a file error).

10.5 Examples of parameter data telegram

Reading parameters

The heatsink temperature C0061 (43 °C) is to be read from the controller with node address 5 via the parameter data channel 1.

► Identifier calculation

Identifier from SDO1 to controller	Calculation
1536 + node address	1536 + 5 = 1541

► Command ”Read Request” (request to read a parameter from the controller)

Command	Value [hex]
Read request	**0x40**

► Index calculation

Index	Calculation
24575 - code number	24575 - 61 = 24514 = 0x5FC2

► Subindex: 0

► Telegram to controller

Identifier	User data							
	Command	Index LOW byte	Index HIGH byte	Subindex	Data 1	Data 2	Data 3	Data 4
1541	0x40	0xC2	0x5F	0x00	0x00	0x00	0x00	0x00

► Telegram from controller

Identifier	User data							
	Command	Index LOW byte	Index HIGH byte	Subindex	Data 1	Data 2	Data 3	Data 4
1413	0x43	0xC2	0x5F	0x00	0xB0	0x8F	0x06	0x00

– Command:
”Read Response” (response to the read request) = 0x43
– Identifier:
SDO1 from controller (1408) + node address (5) = 1413
– Index of the read request:
0x5FC2
– Subindex:
0
– Data 1 ... 4:
0x00068FB0 = 430000 → 430000 : 10000 = 43 °C

Writing parameters

The acceleration time C0012 (parameter set 1) of the controller with the node address 1 is to be changed to 20 seconds via the SDO1 (parameter data channel 1).

- Identifier calculation

Identifier from SDO1 to controller	Calculation
1536 + node address	1536 + 1 = 1537

- Command ”Write Request” (transmit parameter to drive)

Command	Value [hex]
Write request	**0x23**

- Index calculation

Index	Calculation
24575 - code number	24575 - 12 = 24563 = 0x5FF3

- Subindex: 0
- Calculation of the acceleration time

Data 1 ... 4	Calculation
Value for acceleration time	20 s · 10000 = 200000 = 0x00030D40

- Telegram to controller

	User data							
Identifier	**Command**	**Index** LOW byte	**Index** HIGH byte	**Subindex**	**Data 1**	**Data 2**	**Data 3**	**Data 4**
1537	0x23	0xF3	0x5F	0x00	0x40	0x0D	0x03	0x00

- Telegram from controller if executed faultlessly

	User data							
Identifier	**Command**	**Index** LOW byte	**Index** HIGH byte	**Subindex**	**Data 1**	**Data 2**	**Data 3**	**Data 4**
1409	0x60	0xF3	0x5F	0x00	0x00	0x00	0x00	0x00

– Command:
”Write Response” (response of the controller (acknowledgement)) = 0x60

– Identifier:
SDO1 from controller (= 1408) + node address (= 1) = 1409

Read block parameters

The software product code (code C0200) of the Lenze 8200 vector is to be read from parameter set 1. The product code has 14 alphanumerical characters. They are transferred as block parameters. The transfer of block parameters uses the entire data width (2nd to 8th byte).

During user data transfer, the command byte (1. Byte) contains the entry 0x40 or 0x41 to ...

– be able to signalise the end of the block transfer;

– request the next block.

► C0200 - request

Command	**Index** Low byte	**Index** High byte	**Subindex**	**Data 1**	**Data 2**	**Data 3**	**Data 4**
0x40	0x37	0x5F	0x00	0x00	0x00	0x00	0x00

Byte 1: 0x40, ”Read request” (requirement to read a parameter from the controller)
Byte 2/3: Index low/high byte: 24575 - 200 - 0 = 24375 (0x5F37)

► Response including the block length (14 characters)

Command	**Index** Low byte	**Index** High byte	**Subindex**	**Data 1**	**Data 2**	**Data 3**	**Data 4**
0x41	0x37	0x5F	0x00	0x0E	0x00	0x00	0x00

Byte 1: 0x41, ”read response”. The entry 0x41 implies that it is a block diagram.
Byte 2/3: Index low/high byte: 24575 - 200 - 0 = 24375 (0x5F37)
Byte 5: 0x0E = data length of 14 characters (ASCII format)

► First data block - request

Note!

The single blocks are toggled*, i.e. at first it is requested with command 0x60 (0b01100000), then with command 0x70 (0b01110000), after his again with 0x6 etc.

The response is sent accordingly. It is alternating because of a toggle bit. The process is completed by command 0x11 (bit 0 is set, see below).

*Toggle-Bit = bit 4 (starting with ”0”)

Command	**Index** Low byte	**Index** High byte	**Subindex**	**Data 1**	**Data 2**	**Data 3**	**Data 4**
0x60	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Byte 1: 0x60, ”Write response” (acknowledgement) with access to bytes 2 ... 8.

► Response

Command	**Index** Low byte	**Index** High byte	**Subindex**	**Data 1**	**Data 2**	**Data 3**	**Data 4**
0x00	0x38	0x32	0x53	0x38	0x32	0x31	0x32

Bytes 2 ... 8, ASCII format: 8 2 S 8 2 1 2

► Second data block - request

Command	Index Low byte	Index High byte	Subindex	Data 1	Data 2	Data 3	Data 4
0x70	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Byte 1: 0x70 (Toggle), ”Write response” (acknowledgement) with access to all 4 data bytes

► Second data block - response with over-detection

Command	Index Low byte	Index High byte	Subindex	Data 1	Data 2	Data 3	Data 4
0x11	0x56	0x5F	0x31	0x34	0x30	0x30	0x30

Byte 1: 0x11, last data block transfer
Bytes 2 ... 8, ASCII format: V _ 1 4 0 0 0

Result of data block transfer: 82S8212V_14000

10.6 Special features for parameter setting of the drive controller

10.6.1 8200 vector controller

The digital and analog input and output signals are freely configurable (see System Manual 8200 vector; code C0410, C0412, C0417 and C0421).

10.6.2 9300 Servo PLC / Drive PLC / ECS

AIF control byte (C2120)/AIF status byte (C2121)

The controller and the EMF2178IB communication module are able to exchange control and status information. For this purpose, a control byte and a status byte are made available in the AIF protocol.

C2120:
AIF control byte

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C2120	-	22455 = 0x57B7	0	See table below	FIX32

Possible settings	
Selection	**Description**
0	No command
1	Update codes 23XX and CAN reinitialisation ≡ reset node
2	Update codes C23XX
10	Reread C2356/1...4
11	Reread C2357
12	Reread C2375
13	Reread C2376 ... C2378
14	Reread C2382
16 [1)]	Update C7999 and C7998 including CAN reinitialisation
17 [1)]	Update C7999, mapping table of profile indices
18 [1)]	Update C7998, error list of emergency messages
32 [1)]	Reset error source, activate emergency messages
33 ... 62 [1)]	Error source 1 ... 30, activate emergency messages

1) Value range is only valid for profile applications in a PLC

With the AIF control byte it is possible to read the codes C23XX saved in the 9300 Servo PLC, Drive PLC and ECS into the communication module.

Execute command

How to accept the values in the communication module:

- Write the table value of the required command into the AIF control byte.
- Change the status of the MSB of the AIF control byte. The command is executed.

Example

By writing the value ”2” into the AIF control byte all codes C23XX are re-read when the MSB changes its status.

For some codes it is necessary to carry out a CAN reinitialisation so that new values and the functions derived from them will become effective.

C2121:
AIF status byte

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C2121	-	22454 = 0x57B6	0	0	[1]	255	U8

The AIF status byte provides the 9300 Servo PLC, Drive PLC and ECS with information of the communication module. With this information, the 9300 Servo PLC, Drive PLC and ECS can monitor the status of the communication module . Depending on the communication module states, it is possible for you to take corresponding countermeasures.

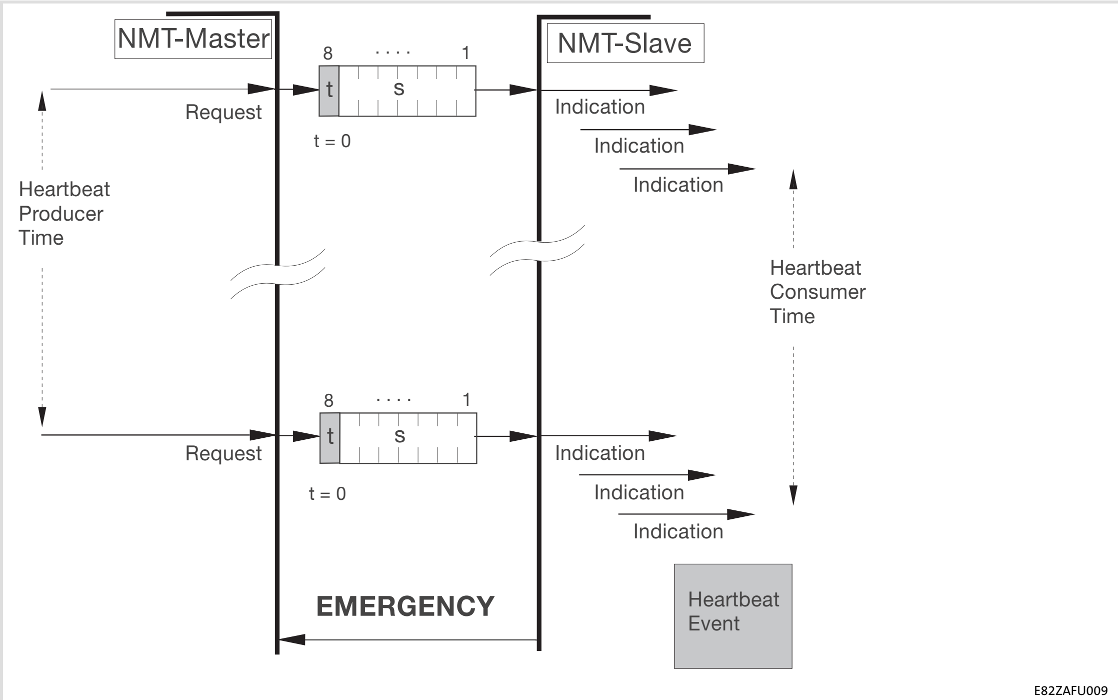
AIF status byte	Description
Bit 0	CE11 error, CAN-IN1 monitoring time exceeded
Bit 1	CE12 error, CAN-IN2 monitoring time exceeded
Bit 2	CE13 error, CAN-IN3 monitoring time exceeded
Bit 3	CE14-error, communication module in BUS-OFF status
Bit 4	Operational status
Bit 5	Pre-operational state
Bit 6	Warning status
Bit 7	Internally assigned

11 Monitoring

11.1 Heartbeat Protocol

Note!

- Only one monitoring function can be active:
  – Heartbeat or node guarding.
- Heartbeat has priority over node guarding:
  – If both functions are configured, the heartbeat settings will be active.



CANopen objects for configuration

CANopen			Lenze		
Index	**Subindex**	**Designation**	**Code**	**Subcode**	**Description**
I-1016	1	Consumer heartbeat time and COB-ID	C1869 C2369		Consumer heartbeat COB-ID
			C1870 C2370	1	Consumer heartbeat time
I-1017	0	Producer heartbeat time	C1870 C2370	2	Producer heartbeat time

The ”heartbeat producer” cyclically sends a heartbeat message to one or several ”heartbeat consumers”.

Heartbeat transmission

The heartbeat transmission is activated by entering a time under in the CANopen object I-1017. The monitoring is activated by setting a time and a node ID in the object I-1016/1.

Telegram structure

NMT identifier plus node ID and a data byte with status information:

Data value (s)	Status
4	Stopped
5	Operational
127	Pre-operational

Identifier

Identifier = Basisadresse (1792) + configurable address (1 ... 127)

Heartbeatmonitoring

The consumer monitors whether the message is received within the ”heartbeat consumer time”. If this is not the case, an emergency telegram with a ”heartbeat event” is sent.

The controller response to a ”heartbeat event” is set under subcode C1882/5 / C2382/5.

If the ”heartbeat event” occurs, the node changes from the ”Operational” state to the ”Pre-operational” state (default setting). By means of the object I-1029 a different response can be set.

The heartbeat monitoring only starts when the first heartbeat telegram of a node monitored has been successfully received and the ”Pre-operational” NMT state has been reached.

Tip!

Detailed information on the objects I-1016, I-1017 and I-1029 can be found in the chapter entitled ”Implemented CANopen objects” (🕮 91).

11.2 Node Guarding Protocol

Note!

- Only one monitoring function can be active:
  – Heartbeat or node guarding.
- Heartbeat has priority over node guarding:
  – If both functions are configured, the heartbeat settings will be active.

The "Node Guarding Protocol" serves to monitor the connection between the NMT master and the NMT slave(s) within a CAN network.

Note!

An NMT master can be a PLC with a CAN interface or a PC card.

The NMT slave function of the "Node Guarding Protocols" (DS301, version 4.01) is supported by the EMF2178IB communication module.

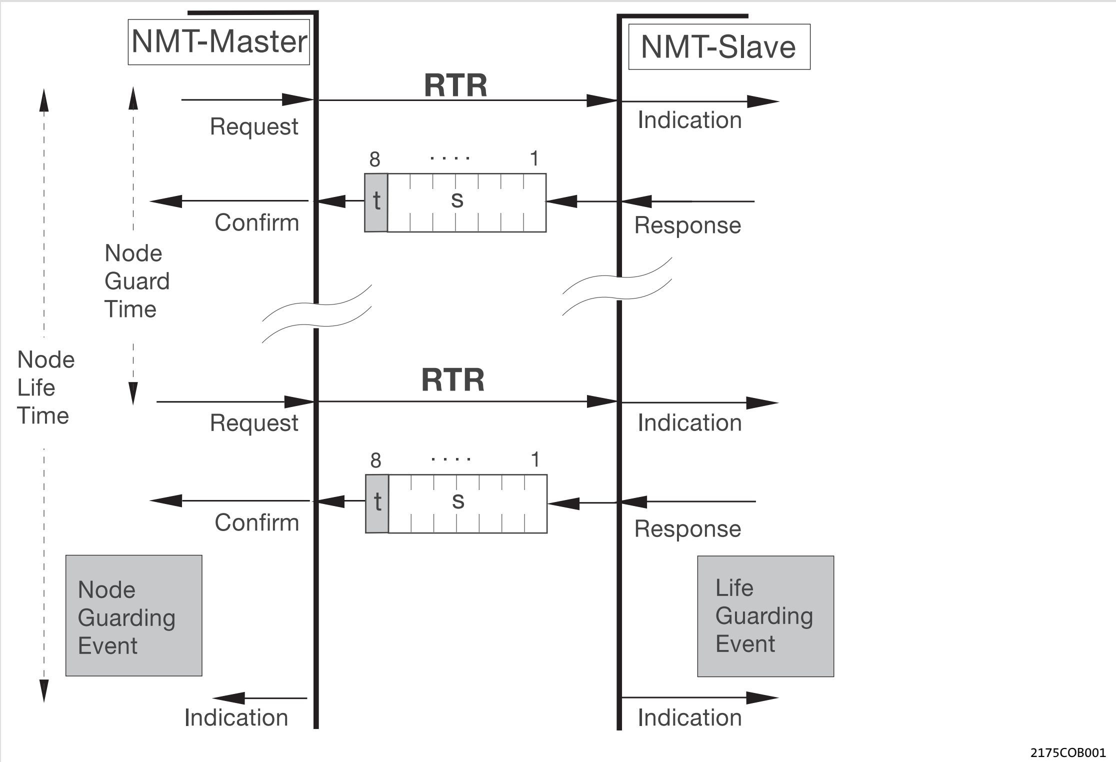


Fig. 11-1 Telegram transfer between NMT master and NMT slave

CANopen objects for configuration

CANopen			Lenze		
Index	Subindex	Designation	Code	Subcode	Description
I-100C	0	Guard time	C1827 C2327	-	Change of ”guard time”
I-100D	0	Life time factor	C1828 C2328	-	Change of ”life time factor”

RTR telegram

The NMT master cyclically (”node guard time”, monitoring time) sends a data telegram called ”Remote Transmit Request” (RTR) to the NMT slave.

The RTR prompts the NMT slave to send its current data.

Response telegram

The NMT slave sends a response telegram with a user data width of 1 byte. The most significant bit of the response telegram is a toggle bit (t).

The data value (s) of the other seven bits indicates the status of the NMT slave:

Data value (s)	Status
4	Stopped
5	Operational
127	Pre-operational

Identifier

Identifier = basic address (1792) + configurable address (1 ... 127)

Node life time

The ”node life time” is the product of ”node guard time” (object I-100C) and ”life time factor” (object I-100D).

Life guarding event

If the NMT slave does not receive an RTR telegram from the NMT master in the ”node life time”, the ”life guarding event” triggers a fault signal in the NMT slave. An emergency telegram with a ”life guarding event” is sent.

The controller response to a ”life guarding event” is set under subcode C1882/5 / C2382/5.

If the ”life guarding event” occurs, the node changes from the ”Operational” state to the ”Pre-operational” state (default setting). By means of the object I-1029 a different response can be set.

Node guarding event

The ”node guarding event” should occur in the NMT master if despite the request by the NMT master no response from the NMT slave is received in the ”node life time” or the toggle bit has not changed in the ”node life time”.

11.3 Emergency telegram

An emergency telegram is sent once to the master if the error status of the communication module changes, i.e.

- if an internal communication module error occurs.
- if an internal communication module error is eliminated.

The emergency telegram sent via the CAN bus is structured as follows:

- Byte 1 + 2: Emergency error code
- Byte 3: Error register object I-1001
- Bytes 4 ... 8: Field for manufacturer-specific error messages
  Contents:
  – Emergency error code 0x1000: Lenze error number
  – All other emergency error codes have the value ”0”.

Emergency error codes [hex]	**Cause**	**Error register entry (I-1001)**
0x0000	One of several errors eliminated	xx
	Single error eliminated (no more errors)	00
0x1000	Basic device in TRIP, message, warning, FAIL, QSP or persistent fault	01
0x3100	Supply voltage of basic device faulty or failed	01
0x6280	Fault in mapping table with operation with a profile implemented in the PLC	01
0x8100	Communication error (warning)	11
0x8130	”Life guard error” or ”heartbeat error”	11
0x8210	PDO length shorter than expected	11
0x8220	PDO length longer than expected	11
0x8700	Sync telegram monitoring	11

12 Diagnostics

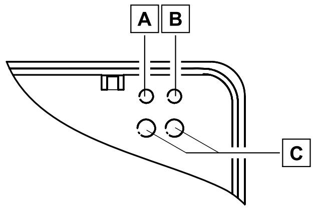
12.1 Measures in case of troubled communication

The following table shows error causes and remedy measures if there is no communication with the controller.

Possible cause of error	Diagnostics	Remedy
Is the controller switched on?	One of the operating status LEDs of the basic device (C) must be on.	Supply controller with voltage (see Operating Instructions/Manual for the basic device)
Is the communication module supplied with voltage?	The green LED A (connection status with controller) is on (remedy 1) or blinking (remedy 2).	• In case of supply from the controller check the connection. With external supply check the 24 V voltage at terminals 39 and 59. A voltage of 24 V +10 % must be applied. • The communication module has not been initialised with the controller yet.
		• **Remedy 1:** Controller not switched on (see fault possibility 1). • **Remedy 2:** Check the connection to the controller.
Does the controller receive telegrams?	The green LED B (connection status with bus) must be blinking when communicating with the host.	Check whether the connection corresponds to the instructions given in the chapter "Electrical installation". Check whether host sends telegrams and uses the appropriate interface.
		Is the existing node address already assigned?

12.2 LED status displays

The LEDs on the front are provided to the communication module for the purpose of fault diagnostics.



2172CAN000D

Fig. 12-1 LEDs of the communication module

Pos.	Status display (LED)		Description
A	Connection status to standard device, two-colour LED (green/red)		
	OFF		● The communication module is not supplied with voltage. ● The external voltage supply is switched off.
	GREEN	Blinking	The communication module is supplied with voltage, but has no connection to the standard device. Cause: The standard device is ... ● switched off; ● in the initialisation phase; ● not available.
		Constantly ON	The communication module is supplied with voltage and is connected to the standard device.
	RED	Constantly ON	CANopen operation impossible.
		Blinking	
		Permanently:	● Parameters are reset to Lenze setting. ● CANopen operation possible.
		1 x blinking:	● An error occurred while saving a value.
		2 x blinking:	● CANopen operation possible. ● The node address/baud rate from C1850/C2350 or C1851/C2351 could not be accepted .

Pos.	Status display (LED)	Description
B	Connection status to fieldbus, two-colour LED (green/red)	
	OFF	No connection to the master
	GREEN	CANopen status ("**Z**")
	RED	CANopen error ("**F**")
	RED constantly on	**Z**: Bus off
	Blinking GREEN every 0.2 seconds	**Z**: Pre-operational, **F**: None
	GREEN every 0.2 seconds 1 x blinking RED, 1 s OFF	**Z**: Pre-operational, **F**: Warning limit reached
	Blinking GREEN every 0.2 seconds 2 x blinking RED, 1 s OFF	**Z**: Pre-operational, **F**: Node guard event
	GREEN constantly on	**Z**: Operational, **F**: None
	GREEN constantly on 1 x blinking RED, 1 s OFF	**Z**: Operational, F: Warning limit reached
	GREEN constantly on 2 x blinking RED, 1 s OFF	**Z**: Operational, **F**: Node guard event
	GREEN constantly on 3 x blinking RED, 1 s OFF	**Z**: Operational, **F**: Sync message error
	Blinking GREEN every second	**Z**: Stopped, **F**: None
	Blinking GREEN every second 1 x blinking RED, 1 s OFF	**Z**: Stopped, **F**: Warning limit reached
	Blinking GREEN every second 2 x blinking RED, 1 s OFF	**Z**: Stopped, **F**: Node guard event

Pos.	Status display (LED)	Description
C	Green and red Drive LED	Operating status of standard device (see standard device documentation)

13 Implemented CANopen objects

Lenze controllers can be parameterised with Lenze codes and manufacturer-independent ”CANopen objects”. A completely CANopen-conform communication can only be achieved by using CANopen objects for parameter setting.

All CANopen objects described in these instructions are defined according to the ”CiA Draft Standard 301/version 4.02”.

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

13.1 Reference between CANopen object and Lenze code

CANopen objects and Lenze codes do not have the same functionalities.

Some CANopen objects have a direct influence on the corresponding codes:

- If a new value is written within an object, the value is also adopted in the corresponding code C18xx or C23xx.
- When reading an object, the values stored under the corresponding code are displayed.

Example

The CANopen object I-1017 (producer heartbeat time) is mapped on the codes C1870/2 and C2370/2:

- Reading the object I-1017:
  – Response: Value under code C1870/C2370, subcode 2.
- Writing a new value into the object I-1017:
  – The new producer heartbeat time is also entered under C1870/2 and C2370/2.

13.2 Overview

CANopen objects			Lenze codes		
Index	Subindex	Designation	Code	Subcode	Description
I-1000	0	Device Type	-	-	-
I-1001	0	Error register	-	-	-
I-1003	0 ... 10 (depending on the error messages)	Error history	-	-	-
I-1005	0	COB-ID SYNC message	C1867 C2367	-	Change of COB-ID Rx
			C1868 C2368	-	Change of COB-ID Tx
			C1856 C2356	5	The time entered depends on bit 30 (sync sending).
			C1857 C2357	6	
I-1006	0	Communication cycle period	C1856 C2356	5	Enter the value corresponding to bit 30 of I-1005$_{hex}$.
			C1857 C2357	6	
I-1008	0	Manufacturer's device name	-	-	-
I-100A	0	Manufacturer's software version	-	-	-
I-100C	0	Guard time	C1827 C2327	-	Change of "guard time"
I-100D	0	Life time factor	C1828 C2328	-	Change of "life time factor"
I-1010	0	Store parameters (U8)	-	-	-
	1...3	Store parameters (U32)			
I-1011	0 ... 7	Restore default parameters	-	-	-
I-1014	0	COB-ID emergency object	C1871 C2371	-	Change of "COB-ID emergency" (setting bit 31 to deactivate the emergency is not directly depend.)
I-1015	0	Emergency inhibit time	C1872 C2372		Change of delay time
I-1016	0, 1	Consumer heartbeat time and COB-ID	C1869 C2369		Change of consumer heartbeat COB-ID
			C1870 C2370	1	Change of consumer heartbeat monitoring time
I-1017	0	Producer heartbeat time	C1870 C2370	2	Change of cycle time for the producer heartbeat telegram
I-1018	0 ... 4	Module device description	-	-	-
I-1029	0, 1	Error behaviour	-	-	-
I-1200	0	Server SDO1 parameters	-	-	-
	1	Identifier client →server (rx)			
	2	Identifier server →client (tx)			
I-1201	0	Server SDO2 parameters	C1865 C2365	1	Change of validity of SDO 2
	1	Identifier client →server (rx)			
	2	Identifier server →client (tx)			

CANopen objects			Lenze codes		
Index	Subindex	Designation	Code	Subcode	Description
I-1400	0	Number of entries	-	-	-
	1	RPDO1 COB-ID	C1853 C2353	1	Change of COB-ID: Set code to CANopen addressing.
			C1855 C2355	1	Enter the new ID under the code.
			C1865 C2365	2	Change of validity of the PDOs
	2	RPDO1 transmission type	C1873 C2373	1	Change of transmission type
	5	RPDO1 event timer	C1857 C2357	1	Change of monitoring time
I-1401	0	Number of entries	-	-	-
	1	RPDO2 COB-ID	C1853 C2353	2	Change of COB-ID: Set code to CANopen addressing.
			C1855 C2355	3	Enter the new ID under the code.
			C1865 C2365	3	Change of validity of the PDOs
	2	RPDO2 transmission type	C1873 C2373	2	Change of transmission type
	5	RPDO2 event timer	C1857 C2357	2	Change of monitoring time
I-1402	0	Number of entries	-	-	-
	1	RPDO3 COB-ID	C1853 C2353	3	Change of COB-ID: Set code to CANopen addressing.
			C1855 C2355	5	Enter the new ID under the code.
			C1865 C2365	4	Change of validity of the PDOs
	2	RPDO3 transmission type	C1873 C2373	3	Change of transmission type
	5	RPDO3 event timer	C1857 C2357	3	Change of monitoring time
I-1600	0	Number of mapped objects in RPDOs	-	-	-
	1	RPDO mapping 1			
	2	RPDO mapping 2			
	3	RPDO mapping 3			
	4	RPDO mapping 4			
I-1601	0	Number of mapped objects in RPDOs	-	-	-
	1	RPDO mapping 1			
	2	RPDO mapping 2			
	3	RPDO mapping 3			
	4	RPDO mapping 4			
I-1602	0	Number of mapped objects in PDOs	-	-	-
	1	RPDO mapping 1			
	2	RPDO mapping 2			
	3	RPDO mapping 3			
	4	RPDO mapping 4			

CANopen objects			Lenze codes		
Index	**Subindex**	**Designation**	**Code**	**Subcode**	**Description**
I-1800	0	Number of entries	-	-	-
	1	TPDO1 COB-ID	C1853 C2353	1	Change of COB-ID: Set code to CANopen addressing.
			C1855 C2355	2	Enter the new ID under the code.
			C1865 C2365	2	Change of validity of the PDOs
	2	TPDO1 transmission type	C1874 C2374	1	Change of transmission type
			C1875 C2375	1	Change of mode
	3	Inhibit time	-	-	-
	5	TPDO1 event timer	C1856 C2356	2	Change of cycle time
I-1801	0	Number of entries	-	-	-
	1	TPDO2 COB-ID	C1853 C2353	2	Change of COB-ID: Set code to CANopen addressing.
			C1855 C2355	4	Enter the new ID under the code.
			C1865 C2365	3	Change of validity of the PDOs
	2	TPDO2 transmission type	C1874 C2374	2	Change of transmission type
			C1875 C2375	2	Change of mode
	3	Inhibit time	-	-	-
	5	TPDO2 event timer	C1856 C2356	3	Change of cycle time
I-1802	0	Number of entries	-	-	-
	1	TPDO3 COB-ID	C1855 C2355	6	Enter the new ID under the code.
			C1865 C2365	4	Change of validity of the PDOs
	2	TPDO3 transmission type	C1874 C2374	3	Change of transmission type
			C1875 C2375	3	Change of mode
	3	Inhibit time	-	-	-
	5	TPDO3 event timer	C1856 C2356	4	Change of cycle time
I-1A00	0	Number of mapped objects in TPDOs	-	-	-
	1	TPDO mapping 1			
	2	TPDO mapping 2			
	3	TPDO mapping 3			
	4	TPDO mapping 4			
I-1A01	0	Number of mapped objects in TPDOs	-	-	-
	1	TPDO mapping 1			
	2	TPDO mapping 2			
	3	TPDO mapping 3			
	4	TPDO mapping 4			

CANopen objects			Lenze codes		
Index	Subindex	Designation	Code	Subcode	Description
I-1A02	0	Number of mapped objects in TPDOs	-	-	-
	1	TPDO mapping 1			
	2	TPDO mapping 2			
	3	TPDO mapping 3			
	4	TPDO mapping 4			

13.2.1 I-1000: Device type

Index (hex) **0x1000**	Name **Device Type**			
Subindex	Default setting	Value range	Access	Data type
0	-	0 ... 2^{32} - 1	ro	U32

The object I-1000 indicates the device profile for this device. It is also possible to include additional information here that is defined in the device profile itself. If a specific device profile is not used, the content is ”0x0000”.

Data telegram assignment

Byte 8	7th byte	Byte 6	Byte 5
U32			
Device profile number		Additional information	

13.2.2 I-1001: Error register

Reading of the error register

Index (hex) **0x1001**	Name **Error register**			
Subindex	Default setting	Value range	Access	Data type
0	-	0 ... 255	ro	U8

Bit assignment in the data byte (U8) for the individual error states

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 fault
0	0	0	0	0	0	0	**1**	Device error message
0	0	0	**1**	0	0	0	**1**	Communication error

13.2.3 I-1003: Error history

Error history

Index (hex)		Name			
0x1003		**Error history**			
Subindex		Default setting	Value range	Access	Data type
0	Number of recorded errors	-	0 ... 255	rw	U8
1 ... 10	Standard error field	-	0 ... 2^{32} - 1	ro	U32

This object shows that errors have occurred in the communication module and in the basic device:

- Subindex 0: Number of saved error messages.
- Subindex 1 ... 10: Error list.

  The error messages consist of a 16-bit error code and a 16-bit manufacturer-specific information field.

Note!

The values in the "standard error field" under subindex 1 ... 10 will be completely deleted if subindex 0 "number of recorded errors" is overwritten with "0".

13.2.4 I-1005: COB-ID SYNC message

Index (hex) **0x1005**	Name **COB-ID SYNC message**			
Subindex	Default setting	Value range	Access	Data type
0	0x80 or 0x80000080	0 ... 2^{32} - 1	rw	U32

This object ensures that sync telegrams can be created for the communication module and that the identifier value can be written.

Creating sync telegrams

Sync telegrams are created when bit 30 (see below) is set to "1".

The time between the syn telegrams can be set using the object I-1006.

Writing identifiers

The default setting for receiving PDOs is "0x80" in the 11-bit identifier (also according to the CANopen specification). This means that all communication modules are default set to the same sync telegram.

If sync telegrams are only to be received by certain communication modules, their identifiers can be entered with values up to "0xFF". The identifiers can only be changed when the communication module does not send sync telegrams (bit 30 = 0).

Data telegram assignment

Byte 8				7th byte		Byte 6		Byte 5	
MSB				U32					LSB
31	30	29	28	...	11	10		...	0
0/1	0/1	0	Bit value: 0			11-bit identifier			

Bits	Value	Description
0 ... 10	X	11-bit identifier
11 ... 29	0	The extended identifier is not supported. Any of these bits must be "0".
30	0	Device does not create sync telegrams.
	1	Device creates sync telegrams.
31	X	optional

13.2.5 I-1006: Communication cycle period

Index (hex) **0x1006**	Name **Communication cycle period**			
Subindex	Default setting	Value range	Access	Data type
0	0	0 ... 2^{32} - 1	rw	U32

This object is used to set a sync telegram cycle time when sync sending is activated (bit 30 in object I-1005).

If sync sending is deactivated and a cycle time is selected, the selected time is used for monitoring the sync telegrams received.

With the default setting of ”0”, sync telegrams are not created or monitored.

Cycle times can be entered as ”1000” or an integer multiple of ”1000”. The unit of the entered time is µs. The maximum value is 65535000 µs.

13.2.6 I-1008: Manufacturer’s device name

Index (hex) **0x1008**	Name **Manufacturer’s device name**			
Subindex	Default setting	Value range	Access	Data type
0	-	Module-specific	const	VS (9 characters)

Display of the manufacturer’s device designation of controller and communication module.

The manufacturer’s device designation comprises a total of 9 characters:

- 1st ... 4th character: The 4th to 7th character is read out of the software ID of the standard device (C0200).
  – e.g. ”8212” from the ID of the 8200 vector frequency inverter
- 5th character: Use of an underscore (”_”)
- 6th ... 9th character: The 4th to 7th character is read out of the software ID of the standard device (C1500).
  – e.g. ”AFU0” from the total ”82ZAFU0B_20000” ID of the E82ZAFUC communication module (CANopen)

13.2.7 I-100A: Manufacturer software version

Index (hex) **0x100A**	Name **Manufacturer's software version**			
Subindex	Default setting	Value range	Access	Data type
0	-	Module-specific	const	VS (11 characters)

Display of the manufacturer's software version of controller and communication module.

The manufacturer's software version consists of a total of 11 characters:

- 1st ... 5th character: The 10th to 14th character is read out of the software ID of the standard device (C0200).
  - 10th character: Major software version
  - 11. character: Minor software version
  - 12th ... 14th character: Power class of the standard device (see documentation of the standard device)
- 6th character: Use of an underscore ("_")
- 7th ... 11th character: The 10th to 14th character is read out of the software ID of the standard device (C1500).
  - e.g. "20000" from the total "82ZAFU0B_20000" ID of the E82ZAFUC communication module (CANopen)

13.2.8 I-100C: Guard time

Index (hex) **0x100C**	Name **Guard time**					
Subindex	Default setting	Value range			Access	Data type
0	0	0	...	65535	rw	U16

The guard time is indicated in milliseconds.

With the default setting of "0", "node guarding" is not supported.

"Node guarding" in the slave is activated if a value > "0" is selected for the "guard time" and the "life time factor". The "guard time" indicates the time within which the RTRs from the master are awaited.

13.2.9 I-100D: Life time factor

Index (hex) **0x100D**	Name **Life time factor**			
Subindex	Default setting	Value range	Access	Data type
0	0	0 ... 255	rw	U8

With the default setting of "0", "node guarding" is not supported.

The product of "guard time" and "life time factor" is important for the monitoring process.

13.2.10 I-1010: Store parameters

Parameter storage in the EEPROM

Index (hex) **0x1010**		Name **Store parameters**			
Subindex		Default setting	Value range	Access	Data type
0	Store parameters	-	0 ... 2^{32} - 1	ro	U8
1 ... 3					U32

Subindices

Subindex	Authorisation	Description	
		Write	**Read**
0	ro	While writing, the 0x06010002 error message occurs.	Supported subindex: 3
1		While writing, the 0x08000020 error message occurs.	Reading of memory functions of all parameters
2			Reading of memory functions of object communication parameters
3			Reading of memory functions of manufacturer-specific parameters (Index 0x6000 ... 0x9FFF)

Bit assignment (reading)

U32				
31	...	2	1	0
0	...	0	0/1	0/1

The following functions are controller-dependent. They are indicated by reading the bit states 0 and 1.

Bits	Value	Meaning of the bit assignment
0	0	No automatic saving of parameters
	1	Automatic saving of parameters
1	0	No saving of parameters on command
	1	Saving of parameters on command
2 ... 31	0	

13.2.11 I-1011: Restore default parameters

Loading of Lenze settings

Note!
For this function, the subindices to be used depend on the controller type.

Index (hex) **0x1011**		Name **Restore default parameters**				
Subindex		Default setting	Value range		Access	Data type
0	Restore default parameters	-	0 ...	2^{32} - 1	ro	U32
1					rw	U32
...						
7						

Subindices

Subindex	Authorisation	Description	
		Write	**Read**
0	ro	While writing, the 0x06010002 error message occurs.	Max. available subindex (depending on standard device): ● 93xx : 3 ● ECSXX: 3 ● 82vector : 7 ● 82xx: 5
1	rw		All parameters can be loaded
2		This function is not supported. While writing, the 0x08000020 error message occurs.	Only object communication parameters can be loaded
3			Only manufacturer-specific parameters can be loaded (Index 0x6000 ... 0x9FFF)
4			Parameter set 1 can be loaded
5			Parameter set 2 can be loaded
6			Parameter set 3 can be loaded
7			Parameter set 4 can be loaded

Bit assignment (reading)

MSB		U32		LSB
31	30	....	1	0
0	0	0	0	0/1

Bit states

Bits	Value	Meaning of the bit assignment
0	0	Parameters cannot be loaded
	1	Parameters can be loaded
1 ... 31	0	

Bit assignment (writing)

The telegram data must include the signature ”**load**” to start the parameter download.

Signature	MSB			LSB
ISO 8859 (ASCII)	**d**	**a**	**o**	**l**
Value (hex)	0x64	0x61	0x6F	0x6C

13.2.12 I-1014: COB-ID emergency object

Index (hex) **0x1014**	Name **COB-ID emergency object**			
Subindex	Default setting	Value range	Access	Data type
0	0x80 + Node ID	0 ... 2^{32} - 1	rw	U32

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

Data telegram assignment

Byte 8			7th byte	Byte 6		Byte 5
MSB			U32			LSB
31	30	29	28 ... 11	10		0
0/1	0	0	Bit value: 0	11-bit identifier		

Bits	Value	Description
0 ... 10	0/1	11-bit identifier
11 ... 29	0	The extended identifier is not supported. Any of these bits must be ”0”.
30	0	Reserved
31	0	Emergency object is valid
	1	Emergency object is invalid

Note!
The COB-ID can only be changed in the ”Emergency object invalid” status (bit 31 = 1).

The emergency telegram sent via the CAN bus is structured as follows:

- Byte 1 + 2: Emergency error code
- Byte 3: Error register object I-1001
- Bytes 4 ... 8: Field for manufacturer-specific error messages
  Contents:
  – Emergency error code 0x1000: Lenze error number
  – All other emergency error codes have the value ”0”.

13.2.13 I-1015: Emergency inhibit time

Index (hex) **0x1015**		Name **Emergency inhibit time**				
Subindex	Default setting	Value range			Access	Data type
0	0	0	...	65535	rw	U16

This object determines the time between two emergency telegrams.

The value entered is multiplied by 0.1, the result is the delay time in milliseconds.

13.2.14 I-1016: Consumer heartbeat time

Index (hex) **0x1016**			Name **Consumer heartbeat time**				
Subindex		Default setting	Value range			Access	Data type
0	Consumer heartbeat time	0	0	...	65535	ro	U32
1						rw	U32

Subindices

Subindex	Explanation
0	Highest subindex
1	Node ID and ”heartbeat time” of node

Bit assignment, subindex 1

U32		
31 ... 24	23 ... 16	15 ... 0
Reserved, value 0	Node ID	Heartbeat time [ms]

The ”consumer heartbeat time” is used to monitor a node.

The ”consumer heartbeat time” must be longer than the ”producer heartbeat time” of the node to be monitored.

If the selected time is exceeded, the communication module will send an emergency telegram including the ”heartbeat error” 0x8130.

Under code C1882/subcode 5, you can set the response of the basic device.

13.2.15 I-1017: Producer heartbeat time

Index (hex) **0x1017**	Name **Producer heartbeat time**			
Subindex	Default setting	Value range	Access	Data type
0	0	0 ... 65535	rw	U32

The heartbeat telegram will be sent automatically if a value > "0" is entered in I-1017. The cycle time corresponds to the entered value in milliseconds.

"Node guarding" is deactivated when the "heartbeat time" is active.

13.2.16 I-1018: Module device description

Entry of vendor ID

Index (hex) **0x1018**		Name **Module device description**			
Subindex		Default setting	Value range	Access	Data type
0 ... 4	Module device description	-	Module-specific	ro	Identity

Subindices

Subindex	Meaning
0	Highest subindex
1	Vendor ID = ID assigned to Lenze by the organisation "CIA"
2	Product code
3	Revision number
4	Serial number

13.2.17 I-1029: Error behaviour

Index (hex) **0x1029**		Name **Error characteristics**			
Subindex		Default setting	Value range	Access	Data type
0	No. of error classes	-	0 ... 255	ro	U8
1	Communication error	-	0 ... 2	rw	U8

Subindex 1

Value of subindex 1	Meaning
0	Status change to "Pre-operational"
1	No status change
2	Status change to "Stopped"

Here, the response to communication errors (node guarding, heartbeat) can be set under subindex 1.

13.2.18 I-1200/I-1201: Server SDO parameters

Server SDOs can be parameterised with two objects:

- I-1200 for the parameter data channel 1 (SDO1)
- I-1201 for the parameter data channel 2 (SDO2)

With I-1201 the identifier can be written in sending and receiving direction, I-1200 has only got reading access. The server SDO parameters are only valid, if bit 31 = ”0” in both transmission directions (subindex 1 and 2).

Index	Subindex	Name	Data type	Value range	Authorisation
I-1200	0	Server SDO1 parameters	U8	0 ... 255	ro
	1	Identifier Client → server (rx)	U32	0 ... (2^{32} - 1)	ro
	2	Identifier Server → client (tx)	U32	0 ... (2^{32} - 1)	ro
I-1201	0	Server SDO2 parameters	U8	0 ... 255	ro
	1	Identifier Client → server (rx)	U32	0 ... (2^{32} - 1)	rw
	2	Identifier Server → client (tx)	U32	0 ... (2^{32} - 1)	rw

Subindices

Subindex	Description
0	Max. supported subindex = 3
1	Specification of the receive identifier
2	Specification of the transmit identifier

Data telegram assignment for subindex 1 and 2

Byte 8				7th byte	Byte 6			Byte 5	
MSB				U32					LSB
31	30	29	28	...	11	10		...	0
0/1	0	0	Bit value: 0			11-bit identifier			

Bits	Value	Description
0 ... 10	X	11-bit identifier
11 ... 29	0	The extended identifier is not supported. Any of these bits must be ”0”.
30	0	Reserved
31	0	SDO valid
	1	SDO invalid

Example:

Parameter data channel 2 (SDO2) of the controller with node address ”4” is to be deactivated. The master must send this command to the communication module via parameter data channel 1 (SDO1).

In sending direction, the basic identifier for SDO2 has the value ”1600” (basic identifier for response: ”1472”).

Identifier = basic identifier + node address = 1604 (0x0644)

For deactivating the parameter data channel (= invalid), set bit 31 in I-1201, subindices 1 and 2 to ”1”. After this, enter the following values under subindices 1 and 2:

I-1201/1: 0x80000000 + 0x644 = 0x80000644

I-1201/2: 0x80000000 + 0x5C4 = 0x800005C4

- User data assignment for I-1201/1:

User data							
Byte 1	**2nd byte**	**3rd byte**	**Byte 4**	**Byte 5**	**Byte 6**	**7th byte**	**Byte 8**
Command	Index Low byte	Index High byte	Subindex	U32			
0x23 (write request)	**0x01**	**0x12**	**0x01** Client → server (rx)	**0x44**	**0x06**	**0x00**	**0x80**

- User data assignment for I-1201/2:

User data							
Byte 1	**2nd byte**	**3rd byte**	**Byte 4**	**Byte 5**	**Byte 6**	**7th byte**	**Byte 8**
Command	Index Low byte	Index High byte	Subindex	U32			
0x23 (write request)	**0x01**	**0x12**	**0x02** Server → client (tx)	**0xC4**	**0x05**	**0x00**	**0x80**

13.2.19 I-1400 ... I-1402: Receive PDO communication parameters

Receipt of PDO communication parameters

Note!
The objects I-1401 and I-1402 are **not** available for 8200 vector and 93XX controllers.

Index	Subindex	Name	Data type	Value range	Authorisation
I-1400	0	Number of entries	U8	0 ... 255	ro
	1	RPDO1 COB-ID	U32	0 ... (2^{32} - 1)	rw
	2	RPDO1 transmission type	U8	0 ... 240, 254	rw
	5	RPDO1 event timer	U32	1 ... 65535	rw
I-1401	0	Number of entries	U8	0 ... 255	ro
	1	RPDO2 COB-ID	U32	0 ... (2^{32} - 1)	rw
	2	RPDO2 transmission type	U8	0 ... 255	rw
	5	RPDO2 event timer	U32	1 ... 65535	rw
I-1402	0	Number of entries	U8	0 ... 255	ro
	1	RPDO3 COB-ID	U32	0 ... (2^{32} - 1)	rw
	2	RPDO3 transmission type	U8	0 ... 255	rw
	5	RPDO3 event timer	U32	1 ... 65535	rw

Subindices

Subindex	Description
0	Max. supported subindices = 5
1	Identifier setting: ● RPDO1: 200_{hex} + node ID ● RPDO2: 300_{hex} + node ID ● RPDO3: 400_{hex} + node ID
2	Transmission type setting (see table above)
5	Monitoring time setting (see table above)

Description of subindex 1

Data telegram assignment

Byte 8				7th byte		Byte 6			
MSB				U32					LSB
31	30	29	28	...	11	10	...		0
0/1	0/1	0	Bit value: 0			11-bit identifier			

Bits	Value	Description
0 ... 10	X	11-bit identifier
11 ... 29	0	The extended identifier is not supported. Any of these bits must be ”0”.
30	0	RTR to RPDO possible (Lenze)
	1	RTR to RPDO not possible (cannot be set)
31	0	RPDO active
	1	RPDO not active

Description of subindex 2

PDO transmission			Transmission type	Description
cyclic	synchronous	event-controlled		
X	X		n = 1 ... 240	By entering value n, this RPDO will be accepted by every n-th sync.
		X	n = 254	Manufacturer-specific, see code C1875 / C2375

Description of subindex 5

The ”monitoring time” describes the period of time in which new process input data must arrive with the CAN-IN1 ... 3 identifiers. If the time entered is exceeded, it is possible to set a corresponding reaction under code C1882.

Note!

- ► The monitoring time starts with the arrival of the first telegram.
- ► Subindex 5 = 0: Monitoring deactivated.

13.2.20 I-1600 ... I-1602: Receive PDO mapping parameters

With these objects, parameter data can be received as PDOs.

Note!
The objects I-1601 and I-1602 are **not** available for 8200 vector and 93XX controllers.

Index	Subindex	Name	Data type	Value range	Authorisation
I-1600	0	Number of mapped objects in RPDOs	U8	0 ... 255	rw
	1	RPDO mapping 1	U32	0 ... (2^{32} - 1)	
	2	RPDO mapping 2			
	3	RPDO mapping 3			
	4	RPDO mapping 4			
I-1601	0	Number of mapped objects in RPDOs	U8	0 ... 255	rw
	1	RPDO mapping 1	U32	0 ... (2^{32} - 1)	
	2	RPDO mapping 2			
	3	RPDO mapping 3			
	4	RPDO mapping 4			
I-1602	0	Number of mapped objects in RPDOs	U8	0 ... 255	rw
	1	RPDO mapping 1	U32	0 ... (2^{32} - 1)	
	2	RPDO mapping 2			
	3	RPDO mapping 3			
	4	RPDO mapping 4			

Subindices

Subindex	Description
0	Number of mapped objects in RPDOs
1 ... 4	This value will be returned in case of a read request for the object.

For mapping in the module, the master enters indices and data lengths.

Mapping structure

MSB								LSB
31	...	16	15	...	8	7	...	0
	Index			Subindex			Object length DWord 0 = 0x20 Word = 0x10	

The EDS file for the EMF2178IB communication module supports the mapping.

Tip!
The current EDS file required for configuring the EMF2178IB (CANopen) communication module can be found in the download area on:
www.Lenze.com

13.2.21 I-1800 ... I-1802: Transmit PDO communication parameters

Sending of PDO communication parameters

Note!
The objects I-1801 and I-1802 are **not** available for 8200 vector and 93XX controllers.

Index	Subindex	Name	Data type	Value range	Authorisatio n
I-1800	0	Number of entries	U8	0 ... 255	ro
	1	TPDO1 COB-ID	U32	0 ... (2^{32} - 1)	rw
	2	TPDO1 transmission type	U8	0 ... 240, 252,254	rw
	3	TPDO1 inhibit time	U32	0 ... 65535	rw
	5	TPDO1 event timer	U32	0 ... 65535	rw
I-1801	0	Number of entries	U8	0 ... 255	ro
	1	TPDO2 COB-ID	U32	0 ... (2^{32} - 1)	rw
	2	TPDO2 transmission type	U8	0 ... 240, 252, 254	rw
	3	TPDO2 inhibit time	U32	0 ... 65535	rw
	5	TPDO2 event timer	U32	0 ... 65535	rw
I-1802	0	Number of entries	U8	0 ... 255	ro
	1	TPDO3 COB-ID	U32	0 ... (2^{32} - 1)	rw
	2	TPDO3 transmission type	U8	0 ... 240, 252, 254	rw
	3	TPDO3 inhibit time	U32	0 ... 65535	rw
	5	TPDO3 event timer	U32	0 ... 65535	rw

Subindices

Subindex	Description
0	Max. supported subindices = 5
1	Identifier setting: ● TPDO1: 0x180 + node ID ● TPDO2: 0x280 + node ID ● TPDO3: 0x380 + node ID
2	Transmission type setting (see table)
3	Inhibit time setting
5	Cycle time setting

Description of subindex 1

Data telegram assignment

Byte 8		7th byte			Byte 6		Byte 5	
MSB				U32				LSB
31	30	29	28	...	11	10	...	0
0/1	0/1	0	Bit value: 0			11-bit identifier		

Bits	Value	Description
0 ... 10	X	11-bit identifier
11 ... 29	0	The extended identifier is not supported. Any of these bits must be ”0”.
30	0	RTR to TPDO possible (Lenze)
	1	RTR to TPDO not possible (cannot be set)
31	0	TPDO active
	1	TPDO not active

Description of subindex 2

PDO transmission			Transmission type	Description
cyclic	synchronous	event-controlled		
X	X		n = 1 ... 240	By entering value n, this TPDO will be accepted by every n-th sync.
	X		n = 252	By sync, TPDO will be filled with new data, but only sent on RTR.
		X	n = 254	Manufacturer-specific, see code C1875 / C2375

Description of subindex 3

Minimum time between two process data telegrams.

Note!
Subindex 3 = 0: Inhibit time deactivated.

Description of subindex 5

With event-controlled/cyclic operation, you can set the cycle time for sending PDOs via the CAN bus.

Note!
Subindex 5 = 0: Cyclic PDO sending deactivated.

13.2.22 I-1A00 ... I-1A02: Transmit PDO mapping parameters

With these objects, parameter data can be sent as PDOs.

Note!
The objects I-1A01 and I-1A02 are **not** available for 8200 vector and 93XX controllers.

Index	Subindex	Name	Data type	Value range	Authorisation
I-1A00	0	Number of mapped objects in TPDOs	U8	0 ... 255	rw
	1	TPDO mapping 1	U32	0 ... (2^{32} - 1)	
	2	TPDO mapping 2			
	3	TPDO mapping 3			
	4	TPDO mapping 4			
I-1A01	0	Number of mapped objects in TPDOs	U8	0 ... 255	rw
	1	TPDO mapping 1	U32	0 ... (2^{32} - 1)	
	2	TPDO mapping 2			
	3	TPDO mapping 3			
	4	TPDO mapping 4			
I-1A02	0	Number of mapped objects in TPDOs	U8	0 ... 255	rw
	1	TPDO mapping 1	U32	0 ... (2^{32} - 1)	
	2	TPDO mapping 2			
	3	TPDO mapping 3			
	4	TPDO mapping 4			

Detailed information on mapping: (🕮 65)

14 Codes

14.1 Overview

Note!

- Some CANopen objects have a direct influence on the corresponding codes. If a new value is written within an index, the value is also adopted in the corresponding code C18xx or C23xx.
- With the 9300 Servo PLC, Drive PLC and ECS, a change of code C23xx will only become effective after one of the following actions:
  – Fresh power-on of communication module.
  – Sending the NMT command ”Reset Node” / ”Reset Communication”.
  – Entering the value ”1” under code C2120 (AIF control byte).
- With all other controllers, changes will become effective immediately.
- Since the codes C18xx are in the module, the corresponding values are directly known. The codes C23xx are in the PLC and therefore must be read by the module first.

Code	Subcode	Index (dec = hex)	Designation	Info
C0001	-	24574 = 0x5FFE	Operating mode for 8200 vector	🕮 144
C0009	-	24566 = 0x5FF6	CAN node address	🕮 144
C0046	-	24529 = 0x5FD1	Display of frequency setpoint	🕮 144
C0125	-	24450 = 0x5F82	Baud rate	🕮 145
C0126	-	24449 = 0x5F81	Behaviour with communication error	🕮 145
C0135	-	24440 = 0x5F78	Controller control word	🕮 145
C0150	-	24425 = 0x5F69	Controller status word	🕮 145
C0161	-	24414 = 0x5F5E	Fault memory 8200 vector: Active fault	🕮 146
C0162	-	24413 = 0x5F5D	Fault memory 8200 vector: Last fault	🕮 146
C0163	-	24412 = 0x5F5C	Fault memory 8200 vector: Last but one fault	🕮 146
C0164	-	24411 = 0x5F5B	Fault memory 8200 vector: Last but two fault	🕮 146
C0168	-	24407 = 0x5F57	Fault memory 9300 / ECSXX	🕮 146
C1810	-	22765 = 0x58ED	Software identification code as string	🕮 120
C1811	-	22764 = 0x58EC	Software creation date as string	🕮 120
C1812	1 ... 4	22763 = 0x58EB	Software identification code in subcodes	🕮 120

Code	Subcode	Index (dec = hex)	Designation	Info
C1813	1 ... 4	22762 = 0x58EA	Software creation date in subcodes	🕮 120
C1822	1 ... 12	22753 = 0x58E1	AIF input words	🕮 121
C1823	1 ... 12	22752 = 0x58E0	AIF output words	🕮 121
C1824	1 ... 11	22751 = 0x58DF	AIF input double words	🕮 121
C1825	1 ... 11	22750 = 0x58DE	AIF output double words	🕮 121
C1827	-	22748 = 0x58DC	Guard time	🕮 122
C1828	-	22747 = 0x58DB	Life time factor	🕮 122
C1850	-	22725 = 0x58C5	CAN node address	🕮 123
C1851	-	22724 = 0x58C4	Baud rate	🕮 124
C1852	-	22723 = 0x58C3	Master / slave operation	🕮 125
C1853	1...3	22722 = 0x58C2	Addressing CAN-INx/CAN-OUTx	🕮 126
C1854	1 ... 6	22721 = 0x58C1	Individual addressing CAN-INx / CAN-OUTx	🕮 129
C1855	1 ... 6	22720 = 0x58C0	Display of resulting identifiers CAN-IN / CAN-OUT	🕮 129
C1856	1 ... 5	22719 = 0x58BF	Boot up and cycle times	🕮 130
C1857	1 ... 6	22718 = 0x58BE	Monitoring times	🕮 131
C1859	-	22716 = 0x58BC	Display of the DIP switch position during initialisation	🕮 132
C1860	-	22715 = 0x58BB	Display of the current DIP switch position	🕮 133
C1861	-	22714 = 0x58BA	Display of the active CAN node address	🕮 133
C1862	-	22713 = 0x58B9	Display of the active baud rate	🕮 133
C1864	-	22711 = 0x58B7	PDO transmission with status change to "Operational"	🕮 134
C1865	1 ... 4	22710 = 0x58B6	Validity of SDO2 and PDOs	🕮 134
C1867	-	22708 = 0x58B4	Sync Rx identifier	🕮 135
C1868	-	22707 = 0x58B3	Sync Tx identifier	🕮 135
C1869	-	22706 = 0x58B2	Consumer heartbeat ID	🕮 136
C1870	1 ... 2	22705 = 0x58B1	Heartbeat time (consumer, producer)	🕮 136
C1871	-	22704 = 0x58B0	Emergency ID	🕮 137
C1872	-	22703 = 0x58AF	Emergency inhibit time	🕮 137

Code	Subcode	Index (dec = hex)	Designation	Info
C1873	1...3	22702 = 0x58AE	Sync rate CAN-IN1 ... CAN IN3	📖 138
C1874	1...3	22701 = 0x58AD	Sync rate CAN-OUT1 ... CAN OUT3	📖 139
C1875	1...3	22700 = 0x58AC	Tx mode CAN-OUT1 ... CAN OUT3	📖 140
C1876	1 ... 4	22699 = 0x58AB	Masks CAN-OUT1	📖 142
C1877	1 ... 4	22698 = 0x58AA	Masks CAN-OUT2	📖 143
C1878	1 ... 4	22697 = 0x58A9	Masks CAN-OUT3	📖 143
C1882	1 ... 6	22693 = 0x58A5	Monitoring responses	📖 143
C2120	-	22455 = 0x57B7	AIF control byte	📖 147
C2121	-	22454 = 0x57B6	AIF status byte	📖 148
C2327	-	22248 = 0x56E8	Guard time	📖 122
C2328	-	22247 = 0x56E7	Life time factor	📖 122
C2350	-	22225 = 0x56D1	CAN node address	📖 123
C2351	-	22224 = 0x56D0	Baud rate	📖 124
C2352	-	22223 = 0x56CF	Master / slave operation	📖 125
C2353	1...3	22222 = 0x56CE	Addressing CAN-INx/CAN-OUTx	📖 126
C2354	1 ... 6	22221 = 0x56CD	Individual addressing CAN-INx / CAN-OUTx	📖 129
C2355	1 ... 6	22220 = 0x56CC	Display of resulting identifiers CAN-IN / CAN-OUT	📖 129
C2356	1 ... 5	22219 = 0x56CB	Boot up and cycle times	📖 130
C2357	1 ... 6	22218 = 0x56CA	Monitoring time	📖 131
C2359	-	22216 = 0x56C8	Display of the DIP switch position during initialisation	📖 132
C2364	-	22211 = 0x56C3	PDO transmission with status change to "Operational"	📖 134
C2365	1 ... 4	22210 = 0x56C2	Validity of SDO2 and PDOs	📖 134
C2367	-	22208 = 0x56C0	Sync Rx identifier	📖 135
C2368	-	22207 = 0x56BF	Sync Tx identifier	📖 135
C2369	-	22206 = 0x56BE	Consumer heartbeat ID	📖 136
C2370	1 ... 2	22205 = 0x56BD	Heartbeat time (consumer, producer)	📖 136
C2371	-	22204 = 0x56BC	Emergency ID	📖 137

Code	Subcode	Index (dec = hex)	Designation	Info
C2372	-	22203 = 0x56BB	Emergency inhibit time	🕮 137
C2373	1...3	22202 = 0x56BA	Sync rate CAN-IN1 ... CAN IN3	🕮 138
C2374	1...3	22201 = 0x56B9	Sync rate CAN-OUT1 ... CAN OUT3	🕮 139
C2375	1...3	22200 = 0x56B8	Tx mode CAN-OUT1 ... CAN OUT3	🕮 140
C2376	1 ... 4	22199 = 0x56B7	Masks CAN-OUT1	🕮 142
C2377	1 ... 4	22198 = 0x56B6	Masks CAN-OUT2	🕮 143
C2378	1 ... 4	22197 = 0x56B5	Masks CAN-OUT3	🕮 143
C2382	1 ... 6	22193 = 0x56B1	Monitoring response	🕮 143

14.2 Communication-relevant Lenze codes

The configuration of the entire system where the individual controller often only represents one of many components in general is defined or altered during the commissioning phase or during service.

For configuring the Lenze controllers, parameters are provided which are stored as so-called "codes" in a numerically ascending order in the memory of the controller.

All codes can be accessed via the CAN bus.

The following codes are available for communication via the CAN bus:

- Codes C1810 ... C1825 and C1860 ... C1862:
  - For all devices.
  - These codes are stored in the EMF2178IB communication module.
- Codes C1827 ... C1882
  - For 8200 vector and 93XX controllers.
  - These codes are stored in the EMF2178IB communication module.
- Codes C2327 ... C2382:
  - For 9300 Servo PLC, Drive PLC and ECS.
  - These codes are stored in the controller.

Communication with the drive requires that the system identifies the drive as node. The identification takes place in the initialisation phase of the communication module.

How to read the tables

Column	Meaning		
Code	(Lenze) code		
Subcode	Subcode		
Index	Information on the addressing of the code		
Lenze	Lenze setting of code		
	Disp	Display code Configuration of this code is not possible.	
Selection	Minimum value	[minimum increment/unit]	maximum value
	For a display code, the displayed values are specified.		
Data type	FIX32	32-bit value with sign; decimal with 4 decimal positions	
	S8	8-bit value, with sign	
	S16	16-bit value, with sign	
	S32	32-bit value, with sign	
	U8	8-bit value, without sign	
	U16	16-bit value, without sign	
	U32	32-bit value, without sign	
	VS	Visible string, character string with defined length	

C1810:
Software ID

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1810	-	22765 = 0x58ED	Disp		VS

Software identification code as string

Only important in the event of service.

C1811:
Software creation date

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1811	-	22764 = 0x58EC	Disp		VS

Software creation date as string

Mainly important in the event of service.

C1812:
Software ID

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1812	/1: ... /4:	22763 = 0x58EB	Disp		U32

Software identification code in subcodes

C1813:
Software creation date

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1813	/1: ... /4:	22762 = 0x58EA	Disp		U32

Software creation date in subcodes

C1822:
AIF-IN all words

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1822	/1: ... /12:	22753 = 0x58E1	Disp		U16

All AIF-IN words as 16-bit values

C1823:
AIF-OUT all words

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1823	/1: ... /12:	22752 = 0x58E0	Disp		U16

All AIF-OUT words as 16-bit values

C1824:
AIF-IN all double words

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1824	/1: ... /11:	22751 = 0x58DF	Disp		U32

All AIF-IN double words as 32-bit values, with an offset of 16 bits each.

(📖 66)

C1825:
AIF-OUT all double words

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1825	/1: ... /11:	22750 = 0x58DE	Disp		U32

All AIF-OUT double words as 32-bit values, with an offset of 16 bits each.

(📖 66)

C1827/C2327:
Guard Time

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1827	-	22748_d = 0x58DC	0	0 [1 ms] 65535	FIX32
C2327	-	22248_d = 0x56E8			

It is possible to select a different ”node life time” for each NMT slave.

The ”node life time” is the product of ”guard time” (object I-100C) and the ”life time factor” (object I-100D). The NMT master must know the two values.

Relation to CANopen

The object I-100C (guard time) has a direct influence on code C1827/C2327. A newly written value in the object is also transferred to code C1827/C2327.

C1828/C2328:
Life time factor

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1828	-	22747 = 0x58DB	0	0 [1] 255	FIX32
C2328	-	22247 = 0x56E7			

It is possible to select a different ”node life time” for each NMT slave.

The ”node life time” is the product of ”guard time” (object I-100C) and the ”life time factor” (object I-100D). The NMT master must know the two values.

Relation to CANopen

The object I-100D (life time factor) has a direct influence on code C1828/C2328. A newly written value in the object is also transferred to code C1828/C2328.

C1850/C2350:
CAN node address

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C1850	-	22725 = 0x58C5	1	1	[1]	99	FIX32
C2350	-	22225 = 0x56D1					

This code serves to set the CAN node address of the communication module.

Note!

- If several CAN nodes are interconnected, their node addresses must differ from each other.
- Node addresses > 99 can only be set via DIP switch(📖 33).
- All DIP switches 1 ... 7 = OFF (Lenze setting):
  - When the device is switched on, the settings under code C1850/C2350 (node address) and C1851/C2351 (baud rate) are active.
  - Code C1850/C2350 is an image of code C0009 of the basic device. Setting C1850/C2350 has a direct effect on C0009.

C1851/C2351:
Baud rate

Code	Subcode	Index	Possible settings			Data type
			Lenze	Selection		
C1851	-	22724 = 0x58C4	0	0 ... 4	0: 500 kbps 1: 250 kbps 2: 125 kbps 3: 50 kbps 4: 1000 kbit/s	FIX32
C2351	-	22224 = 0x56D0				

This code serves to set the baud rate of the communication module.

- ► The baud rate must be the same for all CAN nodes.
- ► The baud rates 10 kbps and 20 kbps can only be selected via DIP switch(🕮 33).
- ► Setting the code (e.g. with GDC via CAN) has a direct effect on the standard device code C0125.

Note!

- ► The baud rate must be the same for all CAN nodes.
- ► The baud rates 10 kbps and 20 kbps can only be selected via DIP switch(🕮 33).
- ► All DIP switches 1 ... 7 = OFF (Lenze setting):
  - – When the device is switched on, the settings under code C1850/C2350 (node address) and C1851/C2351 (baud rate) are active.
  - – Code C1851/C2351 is an image of code C0125 of the basic drive. Setting C1851/C2351 has a direct effect on C0125.

C1852/C2352: Master/slave operation

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1852	-	22723 = 0x58C3	0	0: Slave operation 1: Master operation	FIX32
C2352	-	22223 = 0x56CF			

After switch-on the communication module has the state ”Pre-operational”. In this state only an exchange of parameter data (SDOs) is possible.

In slave operation, the communication module stays in this state until it is put by the NMT command ”Start remote node” into the state ”Operational”.

In the state ”Operational” also process data (PDOs) are exchanged besides parameter data (SDOs).

In master operation, the NMT command ”Start remote node” is transmitted after an adjustable boot-up time (C1856/1, C2356/1), which puts all CAN nodes into the state ”Operational”.

C1853/C2353: Addressing CAN-IN/OUT

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1853	/1: CAN-IN1/CAN-OUT1 /2: CAN-IN2/CAN-OUT2 /3: CAN-IN3/CAN-OUT3	22722 = 0x58C2	0	see table below	FIX32
C2353		22222 = 0x56CE			

Possible settings	
Selection	**Description**
0	Addressing to CANopen (default identifier)
1	Addressing to C1854/C2354 (selective addressing)
2	Addressing to Lenze system bus (CAN)
3	Addressing to CANopen objects I-140X/I-180X

Via this code, the source for the resulting addresses of the CAN-INx/OUTx process data objects (PDOs) on the CAN bus is selected.

Relation to CANopen

The addressing of the corresponding PDP pair or subcode in code C1853/C2353 is switched to the objects I-140X/I-180X (see above: Selection ”3”) if a new value is entered into the objects I-1400 ... I-1402 or I-1800 ... I-1802.

Identifier for addressing to CANopen (default identifier)

This is the Lenze setting of the EMF2178IB communication module. The calculation consists of the basic identifier and the node address. The basic identifier corresponds to the preset value according to DS301 V4.02.

PDO	Identifier	
	C1853/C2353 = 0: Addressing following CANopen (basic identifier)	
CAN-IN1	512 + node address	0x200 + node address
CAN-IN2	768 + node address	0x300 + node address
CAN-IN3	1024 + node address	0x400 + node address
CAN-OUT1	384 + node address	0x180 + node address
CAN-OUT2	640 + node address	0x280 + node address
CAN-OUT3	896 + node address	0x380 + node address

Identifier for selective addressing via C1854/C2354

In case of this addressing, the identifier is the sum of a fixed basic identifier 384 (0x180) + the value of the corresponding subcode of C1854/C2354. Here, the node address has no influence anymore.

This configuration has been created in accordance with 8200 vector controllers with E82ZAFCCxxx function module (system bus (CAN)) and the integrated system bus interface of the 93XX controllers (code C0353/C0354, selective addressing).

PDO	Identifier for addressing via C1854/C2354		
	C1853/C2353 = 1	C1853/C2353 = 1 (with Lenze setting)	
CAN-IN1	384 + C1854/1 or 384 + C2354/1	384 + 129	0x201
CAN-IN2	384 + C1854/3 or 384 + C2354/3	384 + 257	0x281
CAN-IN3	384 + C1854/5 or 384 + C2354/5	384 + 385	0x301
CAN-OUT1	384 + C1854/2 or 384 + C1854/2	384 + 1	0x181
CAN-OUT2	384 + C1854/4 or 384 + C1854/4	384 + 258	0x282
CAN-OUT3	384 + C1854/6 or 384 + C1854/6	384 + 386	0x302

Identifier for addressing to Lenze system bus

This setting corresponds to the preset calculation of the identifiers for 8200 vector controllers in connection with an E82ZAFCCxxx function module (system bus (CAN)) and the integrated system bus interface of 93XX controllers.

The identifier once again consists of a basic identifier and the node address.

PDO	Identifier for addressing via Lenze system bus (CAN)	
	C1853/C2353 = 2	
CAN-IN1	512 + node address	0x200 + node address (hex)
CAN-IN2	640 + node address	0x280 + node address (hex)
CAN-IN3	768 + node address	0x300 + node address (hex)
CAN-OUT1	384 + node address	0x180 + node address (hex)
CAN-OUT2	641 + node address	0x281 + node address (hex)
CAN-OUT3	769 + node address	0x301 + node address (hex)

Identifier for addressing to CANopen projects I-140X/I-180X

If the subcode has the value ”3”, this indicates that the identifiers have been changed via the objects I-140X/I-180X . Now, the identifier is developed directly from the objects.

A change of the codes C1854 / C2354 does not influence the current identifiers.

PDO	Identifier for addressing to CANopen projects I-140X/I-180X
	C1853/C2353 = 3
CAN-IN1	Object I-1400, subindex 1
CAN-IN2	Object I-1401, subindex 1
CAN-IN3	Object I-1402, subindex 1
CAN-OUT1	Object I-1800, subindex 1
CAN-OUT2	Object I-1801, subindex 1
CAN-OUT3	Object I-1802, subindex 1

C1854/C2354:
Individual addressing CAN-IN/OUT

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C1854	/1: CAN-IN1 /2: CAN-OUT1 /3*: CAN-IN2	22721 = 0x58C1	/1: 129 /2: 1 /3: 257*	0	[1]	1663	FIX32
C2354	/4*: CAN-OUT2 /5*: CAN-IN3 /6*: CAN-OUT3	22221 = 0x56CD	/4: 258* /5: 385* /6: 386*				

*) not effective when using 8200 vector or 93XXcontrollers

With this code it is possible to set the addresses of the input and output PDOs individually via six subcodes.

The code becomes effective, if one of the subcodes of code C1853/C2353 contains the value ”1” (selective addressing).

C1855/C2355:
Display of resulting identifiers CAN-IN/OUT

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1855	/1: CAN-IN1 /2: CAN-OUT1 /3*: CAN-IN2	22720 = 0x58C0	Disp		FIX32
C2355	/4*: CAN-OUT2 /5*: CAN-IN3 /6*: CAN-OUT3	22220 = 0x56CC			

*) not effective when using 8200 vector or 93XXcontrollers

In the six subcodes of this code the resulting identifiers of the PDOs can be read out.

Note!

In case of 9300 Servo PLC, Drive PLC and ECS, the value will be transferred to the standard device when the communication module is initialised.

C1856/C2356:
Boot-up- and cycle times

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1856	/1: Boot-up time /2: Cycle time CAN-OUT1 /3: Cycle time CAN-OUT2 /4: Cycle time CAN-OUT3 /5: Sync Tx cycle time	22719 = 0x58BF	0 ms	0 [1 ms] 65535	FIX32
C2356		22219 = 0x56CB			

In the event-controlled/cyclic operation it is possible to define the cycle times with which the single PDOs will be sent via the CAN bus (also see code C1875/C2375, ”Tx mode” for CAN-OUT1 ... 3).

Note!

The value ”0” deactivates the cyclic sending of the PDO.

”Sync Tx cycle time” (C1856/5, C2356/5) describes the interval time (in ms) that is required for sending a sync telegram to the CAN bus.

Relation to CANopen

- The object I-1006 (communication cycle period) has a direct influence on subcode C1856/5 or C2356/5.
  - As the data processing speed of the EMF2178IB communication module amounts to 1000 μs, the entry via the object I-1006 is rounded to an integer multiple of 1000 μs and stored under C1856/5 or C2356/5 (depending on the standard device).
  - When the object I-1006 is read, the content of the corresponding subcode (in μs) is returned as a response.
  - When a value is entered under subcode C1856/5 or C2356/5, bit 30 of the object I-1005 (COB-ID SYNC message) is set automatically and the communication module sends a sync telegram.
- The objects I-1800/5 ... I-1802/5 (TPDO Event Timer) have a direct influence on the subcodes C1856/2 ... 4 or C2356/2 ... 4. Newly entered values in the objects I-1800/5 ... I-1802/5 are also transferred to the subcodes C1856/2 ... 4 or C2356/2 ... 4.

C1857/C2357: Monitoring time

Code	Subcode	Index	Possible settings					Data type
			Lenze	Selection				
C1857	/1: CAN-IN1 /2: CAN-IN2 /3: CAN-IN3 /4: BUS-OFF monitoring time /5: AIF monitoring time /6: Sync Rx monitoring time	22718 = 0x58BE	/1 ... /5: 3000 ms /6: 0 ms	0		[1 ms]	65535	FIX32
C2357		22218 = 0x56CA						

Note!

- The monitoring time starts with the arrival of the first telegram.
- C1857/x = 0: Monitoring deactivated.
- C2357/x = 0: Monitoring deactivated.

CAN-IN1 ... 3 (C1857/1 ... 3, C2357/1 ... 3)

The "monitoring time" describes the period of time in which new process input data must arrive with the CAN-IN1 ... 3 identifiers. If the time entered is exceeded, it is possible to set a corresponding reaction under code C1882/C2382.

BUS-OFF (C1857/4, C2357/4)

Even if the node changes to the state BUS-OFF, it is possible to set a time (in ms) in which a reaction can take place. A monitoring response can be set under code C1882/C2382.

AIF monitoring time (C1857/5, C2357/5)

Time during boot-up for detecting a basic device. If no basic device is detected, an emergency telegram (📖 87) will be sent on the CAN bus after this time.

Sync monitoring time (C1857/6, C2357/6)

Monitoring time for a received, cyclic sync telegram.

Relation to CANopen

- The object I-1006 (Communication Cycle Period) has a direct influence on the subcodes C1857/6 or C2357/6. A new value in the code deletes bit 30 of the object I-1005 (COB-ID SYNC message).
- The objects I-1400/5 ... I-1402/5 (TPDO Event Timer) have a direct influence on the subcodes C1857/1 ... 3 or C2357/1 ... 3. Newly entered values in the objects I-1400/5 ... I-1402/5 are also transferred to the subcodes C1857/1 ... 3 or C2357/1 ... 3.

C1859/C2359: Display of DIP switch position

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1859	-	22716 = 0x58BC	Disp		U16
C2359	-	22216_d = 0x56C8			U32

The DIP switch position is indicated as a hexadecimal number with the initialisation of the communication module.

Note!
In case of 9300 Servo PLC, Drive PLC and ECS, the value will be transferred to the standard device when the communication module is initialised.

DIP switch valencies for hexadecimal calculation

Switch	-	1	2	3	4	5	6	7	8	9	10
Value	-	512	256	128	64	32	16	8	4	2	1
Bit	15 ... 10	9	8	7	6	5	4	3	2	1	0

Example for DIP switch position

- Switches 3, 5, 6, 7 = ON (address 23)
- Switches 8, 9, 10 = OFF (baud rate 500 kbit/s)
  The sum of the corresponding valencies ”0xB8” (184) is displayed when code C1859/C2359 is read.

C1860:
Display of the current DIP switch position

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1860	-	22715 = 0x58BB	Disp		U16

By displaying the current DIP switch position it is possible to find out if the switch position for node address and baud rate has changed since the last initialisation.

DIP switch valencies

Switch	-	1	2	3	4	5	6	7	8	9	10
Value	-	512	256	128	64	32	16	8	4	2	1
Bit	15 ... 10	9	8	7	6	5	4	3	2	1	0

Example for DIP switch position

- Switches 3, 5, 6, 7 = ON (address 23)
- Switches 8, 9, 10 = OFF (baud rate 500 kbit/s)
  The sum of the corresponding valencies ”0xB8” (184) is displayed when code C1859/C2359 is read.

C1861:
Display of the active CAN node address

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1861	-	22714 = 0x58BA	Disp		FIX32

Display of the CAN node address active on the CAN bus.

C1862:
Display of the active baud rate

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1862	-	22713 = 0x58B9	Disp	0: 500 kbps 1: 250 kbps 2: 125 kbps 3: 50 kbps 4: 1000 kbps 5: 20 kbps 6: 10 kbit/s	FIX32

Display of the baud rate active on the CAN bus.

C1864/C2364:
PDP transmission with status change to ”Operational”

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1864	-	22711 = 0x58B7	0	0: Do not send PDOs	FIX32
C2364	-	22211 = 0x56C3		1: Send PDOs	

When the CAN status changes from ”Pre-Operational” to ”Operational”, all PDOs are sent once.

C1865/C2365:
Validity of SDO2 and PDOs

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1865	/1: SDO2 /2: CAN-IN/OUT1 /3*: CAN-IN/OUT2 /4*: CAN-IN/OUT3	22710 = 0x58B6	/1: 0 /2: 1 /3: 1 /4: 1	0 [1] 3	FIX32
C2365	/1: SDO2 /2: CAN-IN/OUT1 /3*: CAN-IN/OUT2 /4*: CAN-IN/OUT3	22210 = 0x56C2	/1: 0 /2: 1 /3: 1 /4: 1		

*) not effective when using 8200 vector or 93XXcontrollers

Validity of service data object 2 (SDO2) and of process data objects

Possible selections C1865/C2365	
Value	**Description**
0	SDO/PDO invalid
1	SDO/PDO valid
2	Receiving direction valid
3	Sending direction valid

C1867/C2367:
Sync Rx identifier

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C1867	-	22708 = 0x58B4	128	0	[1]	2047	FIX32
C2367	-	22208 = 0x56C0					

This code contains the identifier with which the sync telegram is received.

Relation to CANopen

The object I-1005 (COB-ID SYNC message) has a direct influence on this code. Newly written values in the object are also transferred to code C1867/C2367.

C1868/C2368:
Sync Tx identifier

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C1868	-	22707 = 0x58B3	128	0	[1]	2047	FIX32
C2368	-	22207 = 0x56BF					

This code contains the identifier with which the sync telegram is sent.

Relation to CANopen

The object I-1005 (COB-ID SYNC message) has a direct influence on this code. Newly written values in the object are also transferred to code C1868/C2368.

C1869/C2369:
Consumer heartbeat ID

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C1869	-	22706 = 0x58B2	0	0	[1]	255	FIX32
C2369	-	22206 = 0x56BE					

This code contains the identifier with which the heartbeat telegram for monitoring is received.

Relation to CANopen

The object I-1016, subindex 1 (consumer heartbeat) has a direct influence on this code. A value entered into subindex 1 of I-1016 is also transferred to code C1869/C2369.

C1870/C2370:
Heartbeat time (consumer, producer)

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C1870	/1: Consumer	22705 = 0x58B1	0	0	[1 ms]	65535	FIX32
	/2: Producer		0				
C2370	/1: Consumer	22205 = 0x56BD	0				
	/2: Producer		0				

Consumer heartbeat time

Time intervals at which the heartbeat telegram to be monitored is awaited on the CAN bus.

Producer heartbeat time

Time intervals at which the module sends a heartbeat telegram including its status on the CAN bus.

Relation to CANopen

- The object I-1016, subindex 1 (consumer heartbeat time) has a direct influence on this code. A value written into subindex 1 of I-1016 is also transferred to code C1870/1 C23701 .
- The object I-1017 (producer heartbeat time) has a direct influence on this code. A value written into the object is also transferred to code C1870/2 / C2370/2.

C1871/C2371: Emergency ID

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1871	-	22704 = 0x58B0	128	0 [1] 2047	FIX32
C2371	-	22204 = 0x56BC			

This code contains the identifier with which the emergency telegram is sent.

Relation to CANopen

The object I-1014 (COB-ID Emergency Object) has a direct influence on this code. A value written into the object is also transferred to code C1871/C2371.

C1872/C2372: Emergency inhibit time

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1872	-	22703 = 0x58AF	0	0 [1 ms] 65535	FIX32
C2372	-	22203 = 0x56BB			

This code contains the delay time for sending an emergency telegram.

Relation to CANopen

The object I-1015 (Inhibit Time EMCY) has a direct influence on this code. A value written into the object is also transferred to code C1872/C2372.

C1873/C2373:
Sync rate CAN-IN1 ... 3

Code	Subcode	Index	Possible settings					Data type
			Lenze	Selection				
C1873	/1: CAN-IN1 /2*: CAN-IN2 /3*: CAN-IN3	22702 = 0x58AE	1	0	[1]	240		FIX32
C2373		22202 = 0x56BA						

*) not effective when using 8200 vector or 93XXcontrollers

The process input data (CAN-INx) are only transferred to the controller after a certain number of sync telegrams have been received.

The sync rate can be set individually for each input PDO.

Note!

Only sync telegrams are counted that have been received with the identifier set under code C1867/C2367 of the communication module.

Example:

- Selection n = 23:
  Acceptance of input PDO (to CAN-IN1 ... 3) into the controller after arrival of the 23rd sync telegram.

Relation to CANopen

The objects I-1400 ... I-1402 (Receive PDO Communication Parameters), subindex 2 each (transmission type), are mapped on the corresponding subcodes of C1873/C2373 in the value range 1 ... 240.

Exception: Object I-140X, subindex 2 = ”254” is mapped manufacturer-specifically as value ”0” on the corresponding subcodes of C1873/C2373.

Object I-140X, subindex	Mapping to code / subcode
I-140X, subindex 2 = 1 ... 240	C1873, subcode X = 1 ... 240
I-140X, subindex 2 = 254	C1873, subcode X = 0

C1874/C2374:
Sync rate CAN-OUT1...3

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C1874	/1: CAN-OUT1 /2*: CAN-OUT2 /3*: CAN-OUT3	22701 = 0x58AD	1	1	[1]	240	FIX32
C2374		22201 = 0x56B9					

*) not effective when using 8200 vector or 93XXcontrollers

The process output data (CAN-OUTx) are only transferred after a certain number of sync telegrams have been received.

The sync rate can be set individually for each process output data object.

Relation to CANopen

The objects I-1800 ... I-1802 (Transmit PDO Communication Parameters), subindex 2 each (transmission type), are transferred to the corresponding subcodes of C1874/C2374 in the value range 1 ... 240.

C1875/C2375:
Tx mode CAN-OUT1...3

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C1875	/1: CAN-OUT1 /2*: CAN-OUT2 /3*: CAN-OUT3	22700 = 0x58AC	/1: 0 /2: 1 /3: 1	0 [1] 3	FIX32
C2375		22200 = 0x56B8			

*) not effective when using 8200 vector or 93XXcontrollers

This code contains a selection which indicates at which event the TPDOs CAN-OUT1 ... 3 are to be sent. It is possible to make an individual selection for each TPDO by dividing the code in subcodes.

- Value = 0:
  - The TPDOs are sent when a sync telegram has been received.
  - Under code C1874/C2374 you can set that the TPDOs are only to be sent after the n-th sync telegram (n = 1 ... 240).
- Value = 1:
  - TPDOs are not sent.
  - With this selection the transmission of TPDOs is deactivated.
    This is useful for 8200 vector and 9300 drives which are only able to exchange four words as a maximum of control and status information via the AIF. For this purpose, the use of one TPDO is sufficient, TPDOs 2 and 3 are deactivated (Lenze setting). This avoids an unnecessary load of the CAN bus.
- Value = 2:
  - The corresponding TPDO is either transmitted event-controlled or cyclically.
  - The TPDO is transmitted cyclically if a cycle time is defined for the TPDO under code C1856/C2356. If the cycle time is zero, the TPDO is sent in case of an event, i.e. bit change within the TPDO.
- Value = 3:
  - The TPDO is event-controlled and cyclically transmitted. This means that the object is transmitted with the cycle time defined under code C1856/C2356.
  - In addition, the object will be transmitted if one or several bits are changed within the TPDO.

Note!

If the transmission is event-controlled (also with cyclic superimpositions) some bits can be hidden by masking the object using codes C1876 ... C1878 or with the 9300 Servo PLC, Drive PLC and ECS C2376 ... C2378. This means that the TPDO will not be sent when a bit is being changed.

- C1875/C2375, subcodes 1 ... 3 = 0
  The value of code C1874/C2374, subcode 1, 2 or 3 is displayed in the objects I-1800 ... I-1802 (subindex 2 each).
- C1875/C2375, subcodes 1 ... 3 = 1
  The value ”252” is displayed in the objects I-1800 ... I-1802hex.
- C1875/C2375, subcodes 1 ... 3 = 2 or 3
  The value ”254” is displayed in the objects I-1800 ... I-1802hex.

When setting the objects I-1800 ... I-1802, the same relation prevails:

Code C1875/C2375, subcode 1, 2 or 3	**Object I-180X, subindex 2**
0	Code C1874/C2374, subcode 1, 2 or 3 (value = 1 ... 240)
1	252
2 or 3	254

C1876/C2376: Masks CAN-OUT1

This mask is used to skip one or several bits of the output PDO CAN-OUT1.

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C1876	/1: CAN-OUT1.W1 /2: CAN-OUT1.W2 /3: CAN-OUT1.W3 /4: CAN-OUT1.W4	22699 = 0x58AB	65535	0	[1]	65535	FIX32
C2376		22199 = 0x56B7					

The event-controlled transmission of the CAN-OUT object can be e.g. dependent on one bit only. (See also code C1875/C2375, 📖 140.)

Example:

The mask in word 3 of the process data object CAN-OUT1 is set by code C1876/3 with the value ”0x20”.

- ► 1st cycle
  Result after 1st cycle: The PDO is sent.

	CAN-OUT1.W3															
MASK	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0
Data	1	1	1	1	1	1	1	1	0	1	**1**	0	0	0	1	0
Result	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0

- ► 2nd cycle
  Result after 2nd cycle:
  – New data have been written into CAN-OUT1.
  – The PDO is sent due to the bit change.

	CAN-OUT1.W3															
MASK	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0
Data	1	1	1	1	1	1	1	1	0	1	**0**	0	0	0	1	0
Result	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0

C1877/C2377:
Masks CAN-OUT2

This mask is used to skip one or several bits of the output PDO CAN-OUT2.

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C1877	/1: CAN-OUT2.W1 /2: CAN-OUT2.W2 /3: CAN-OUT2.W3 /4: CAN-OUT2.W4	22698 = 0x58AA	65535	0	[1]	65535	FIX32
C2377		22198 = 0x56B6					

For more information, please see the description of code C1876/C2376 (📖 142).

C1878/C2378:
Masks CAN-OUT3

This mask is used to skip one or several bits of the output PDO CAN-OUT3.

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C1878	/1: CAN-OUT3.W1 /2: CAN-OUT3.W2 /3: CAN-OUT3.W3 /4: CAN-OUT3.W4	22697 = 0x58A9	65535	0	[1]	65535	FIX32
C2378		22197 = 0x56B5					

For more information, please see the description of code C1876/C2376 (📖 142).

C1882/C2382:
Monitoring response

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C1882	/1: response CAN-IN1 /2: response CAN-IN2 /3: response CAN-IN3 /4: response BUS-OFF	22693 = 0x58A5	/1:0 /2:0 /3:0 /4:0	0	[1]	2	FIX32
C2382	/5: response life guarding event / heartbeat event /6: response Rx sync monitoring	22193 = 0x56B1	/5:0 /6: 0	0: no response 1: Controller inhibit (CINH) 2: Quick stop (QSP)			

Setting the response in the controller if the monitoring times have been exceeded (see code C1857/C2357, 📖 131.).

14.3 Important controller codes

This section briefly describes the most important codes of the standard device for CAN communication.

Documentation for the basic drive
Here you can find detailed information on the codes of the standard device.

C0001:
Operating mode for 8200 vector

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C0001	-	24574 = 0x5FFE	0	see System Manual for 8200 vector	FIX32

The operating mode defines which source may currently write to which parameters.

The operating unit and CAN are always authorised for parameter setting.

Note!
Please observe that the operating mode C0001 is contained in all parameter sets. Therefore C0001 must be set equally in all parameter sets.

C0009:
CAN node address

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
C0009	-	24566 = 0x5FF6	1	1	[1]	99	FIX32

Node address for unambiguous addressing of the device in a CAN network.

C0046:
Display of frequency setpoint

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C0046	-	24529 = 0x5FD1	Disp	see documentation of the standard device	FIX32

C0125:
Baud rate

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C0125	-	24450 = 0x5F82	0	see documentation of the standard device	FIX32

C0126:
Behaviour with communication error (extract)

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C0126	-	24449 = 0x5F81		see documentation of the standard device	FIX32

Monitoring of the internal communication between communication module and standard device (controller).

C0135:
Controller control word

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C0135	-	24440 = 0x5F78		see documentation for the basic device	I16

C0150:
Controller status word

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C0150	-	24425 = 0x5F69		see documentation of the standard device	I16

C0161 ... C0164: Fault memory 8200 vector

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C0161	-	24414 = 0x5F5E	Disp	active fault	FIX32
C0162		24413 = 0x5F5D		last fault	
C0163		24412 = 0x5F5C		last but one fault	
C0164		24411 = 0x5F5B		last but two fault	

System Manual for 8200 vector
Here you can find detailed information on the fault messages.

C0168: Fault memory 9300 / ECSXX

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C0168	0 ... 8	24407 = 0x5F57	Disp	see documentation of the standard device	FIX32

9300 system manual / ECSXX operating instructions
Here you can find detailed information on the fault messages.

C2120:
AIF control byte

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C2120	-	22455 = 0x57B7	0	See table below	FIX32

Possible settings	
Selection	**Description**
0	No command
1	Update codes 23XX and CAN reinitialisation ≡ reset node
2	Update codes C23XX
10	Reread C2356/1...4
11	Reread C2357
12	Reread C2375
13	Reread C2376 ... C2378
14	Reread C2382
16 [1)]	Update C7999 and C7998 including CAN reinitialisation
17 [1)]	Update C7999, mapping table of profile indices
18 [1)]	Update C7998, error list of emergency messages
32 [1)]	Reset error source, activate emergency messages
33 ... 62 [1)]	Error source 1 ... 30, activate emergency messages

1) Value range is only valid for profile applications in a PLC

With the AIF control byte it is possible to read the codes C23XX saved in the 9300 Servo PLC, Drive PLC and ECS into the communication module.

Execute command

How to accept the values in the communication module:

- ► Write the table value of the required command into the AIF control byte.
- ► Change the status of the MSB of the AIF control byte. The command is executed.

Example

By writing the value ”2” into the AIF control byte all codes C23XX are re-read when the MSB changes its status.

For some codes it is necessary to carry out a CAN reinitialisation so that new values and the functions derived from them will become effective.

C2121: AIF status byte

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
C2121	-	22454 = 0x57B6	0	0 [1] 255	U8

The AIF status byte provides the 9300 Servo PLC, Drive PLC and ECS with information of the communication module. With this information, the 9300 Servo PLC, Drive PLC and ECS can monitor the status of the communication module . Depending on the communication module states, it is possible for you to take corresponding countermeasures.

AIF status byte	Description
Bit 0	CE11 error, CAN-IN1 monitoring time exceeded
Bit 1	CE12 error, CAN-IN2 monitoring time exceeded
Bit 2	CE13 error, CAN-IN3 monitoring time exceeded
Bit 3	CE14-error, communication module in BUS-OFF status
Bit 4	Operational status
Bit 5	Pre-operational state
Bit 6	Warning status
Bit 7	Internally assigned

15 Index

Zahlen

E

F

G

H

I

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

EDSMF2178IB ▪ 13437291 ▪ EN ▪ 3.0 ▪ TD17

10 9 8 7 6 5 4 3 2 1