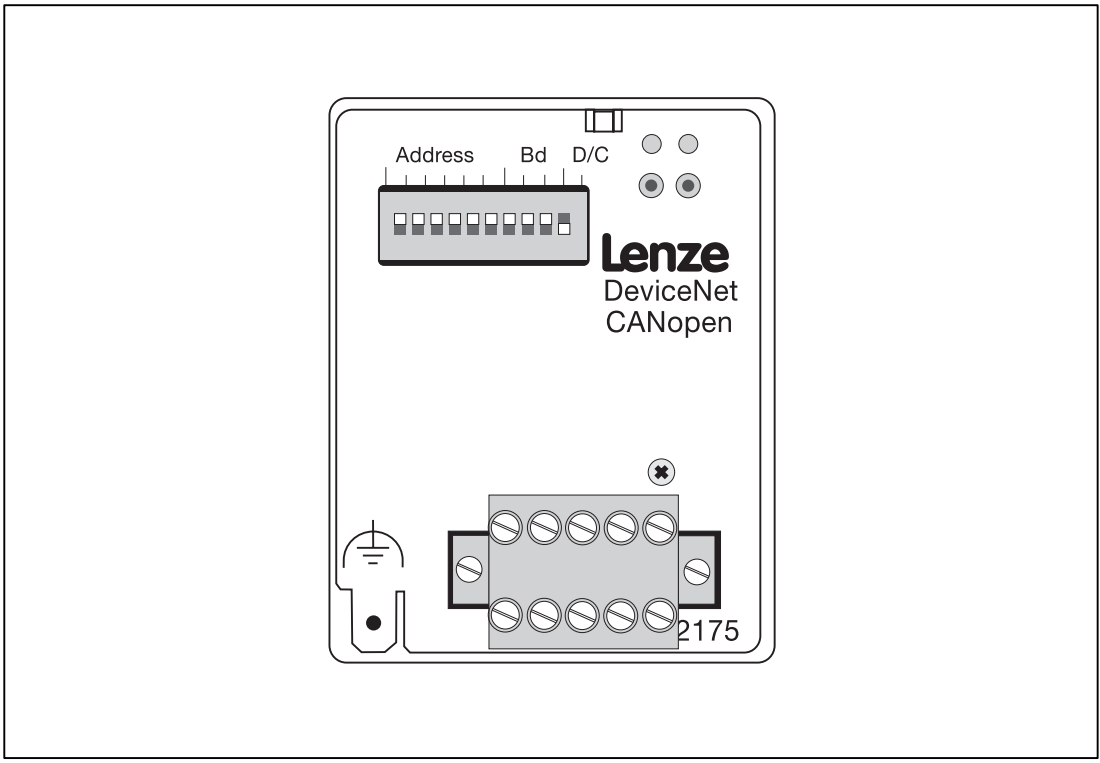
EDB2175EN
00423117

Lenze

Operating Instructions



Fieldbus module type 2175

DeviceNet / CANopen

These Operating Instructions are valid for fieldbus modules with the following nameplates:

2175IB Vx. 0x

In connection with the unit series as from nameplate data:

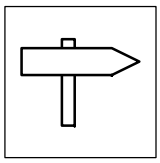
820X	E.	2x.	1x.		(8201 - 8204)
820X	E./C.	2x.	1x.	Vxxx	(8201 - 8204)
821X	E.	2x.	2x.		(8211 - 8218)
821X	E./C.	2x.	2x.	Vxxx	(8211 - 8218)
822X	E.	1x.	1x.		(8221 - 8225)
822X	E.	1x.	1x.	Vxxx	(8221 - 8227)
824X	E.	1x.	1x.		(8241 - 8246)
824X	E./C.	1x.	1x.	Vxxx	(8241 - 8246)
82EVxxxxxBxxxXX	Vx	1x			8200 vector
82CVxxxxxBxxxXX	Vx	1x			8200 vector, cold plate
93XX	E./C.	2x	1x		(9321 - 9332)
93XX	Ex	2x	1x		(Servo PLC 9300)

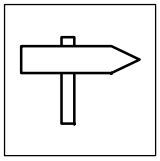
Important:
These Operating Instructions are only valid together with the corresponding Instructions for 82XX, 82XX; 8200 vector, 93XX and 9300 Servo PLC controllers. C

What is new / what has been changed ?

Material no.	Edition	Important	Contents
423117	2.0 07/01 TD02	1st edition	

Contents





Contents

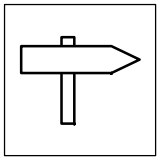
CANopen

Contents

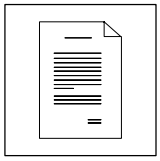
Contents

Contents

DeviceNet



Contents



1 Preface and general information

1.1 How to use these Operating Instructions

- These Operating Instructions are intended for safety-relevant operation on and with the module. They contain safety information which must be observed.
- All personnel working on and with the module must have these Operating Instructions available and observe the information and notes relevant for them.
- These Instructions are only valid in combination with the Operating Instructions of the corresponding controller. They must always be complete and in a perfectly readable state.
- These Operating Instructions describe the communication profiles CANopen and DeviceNet. The chapter Preface and General Information and the Safety Instructions apply to both communication profiles. Please see the header of each page.

1.1.1 Terminology used

Controller	In the following, the term ”controller” is used for ”93XX servo inverters” or ”82XX frequency inverters”.
Drive system	In the following the term ”drive system” is used for drive systems with fieldbus modules and other Lenze drive components.
Fieldbus module	In the following text, the term ”fieldbus module” is used for the fieldbus module type 2175 DeviceNet/CANopen.
L-Cxxxx	Lenze code
Cxxxx/y	Subcode y of code Cxxxx (e.g. C0410/3 = subcode 3 of code C0410)
Xk/y	Terminal strip Xk/terminal y (e.g. X3/28 = terminal 28 on terminal strip X3)
(📖xx-yyy)	Cross reference (chapter - page)

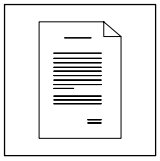
1.1.2 Conventions used

The following conventions are used to differentiate between different types of information:

Type of information	Printing	Example
Names of dialog boxes, input fields or selection lists	*italics*	Dialog box *Options*
Buttons	**bold**	Click **OK** , to...
Menu commands	**bold**	Use the command **Messages** to...
		If the execution of a function requires several commands, the commands are separated from each other by arrows: Select **File →Open** , to...

1.2 Items supplied

Items supplied	Important
• 1 2175 fieldbus module with housing (type of protection IP20) • 1 M3 fixing screw • 1 5-pole connector for DeviceNet and CANopen • 1 Mounting Instructions • 1 floppy	After the delivery has received, check immediately whether the items supplied match the accompanying papers. Lenze does not accept any liability for deficiencies claimed subsequently. Claim • visible transport damage immediately to the forwarder • visible deficiencies/incompleteness immediately to your Lenze representative.



1.2.1 Legal regulations

Labelling	**Nameplate**	**CE-identification**	**Manufacturer**
	Lenze 2175 fieldbus modules are unambiguously identified by their nameplates.	Conforms to the EC Low-Voltage Directive	Lenze GmbH & Co KG Postfach 101352 D-31763 Hameln
Application as directed	**2175 fieldbus module** • Operate the fieldbus module only under the conditions prescribed in these Operating Instructions. • The fieldbus module is an accessory module and can be optionally attached to the following Lenze controllers: 820X, 821X, 822X, 8200 vector, 9300 servo 9300 servo PLC. The 2175 fieldbus module links Lenze controllers with the fast serial communication system CAN. • The fieldbus module must be attached and electrically connected so that it complies with its function and does not cause any hazards when attached and operated as instructed. • Observe all notes given in chapter „Safety Instructions“ (📖 2-1)). • Please observe all information given in these Operating Instructions. This means: – Read these Operating Instructions carefully before you start to work with the system. – These Operating Instructions must always be available during operation of the fieldbus module. **Any other use shall be deemed inappropriate!**		
Liability	• The information, data, and notes in these Instructions met the state of the art at the time of printing. Claims referring to drive systems which have already been supplied cannot be derived from information, illustrations, and descriptions given in these Operating Instructions. • The specifications, processes, and circuitry described in these Instructions are for guidance only and must be adapted to your own specific application. Lenze does not take responsibility for the suitability of the process and circuit proposals. • The indications given in these Operating Instructions describe the features of the product without warranting them. • Lenze does not accept any liability for damage and operating interference caused by: – disregarding these Instructions – unauthorized modifications to the controller – operating faults – improper working on and with the controller		
Warranty	• Warranty conditions: see Sales and Delivery Conditions of Lenze GmbH & Co KG. • Warranty claims must be made to Lenze immediately after detecting the deficiency or fault. • The warranty is void in all cases where liability claims cannot be made.		
Disposal	**Material**	**recycle**	**dispose**
	Metal	●	-
	Plastic	●	-
	Assembled PCBs	-	●
	Short Instructions/Operating Instructions	●	

2 Safety information

2.1 Persons responsible for safety

Operator

- An operator is any natural or legal person who uses the drive system or on behalf of whom the drive system is used.
- The operator or his safety personnel is obliged
  - to ensure the compliance with all relevant regulations, instructions and legislation.
  - to ensure that only skilled personnel works on and with the2102IB fieldbus module.
  - to ensure that the personnel has the Operating Instructions available for all corresponding works.
  - to ensure that all unqualified personnel are prohibited from working on and with the drive system.

Qualified personnel

Qualified personnel are persons who - because of their education, experience, instructions, and knowledge about corresponding standards and regulations, rules for the prevention of accidents, and operating conditions - are authorized by the person responsible for the safety of the plant to perform the required actions and who are able to recognize potential hazards.
(Definition for qualified personnel to VDE 105 or IEC 364)

2.2 General safety information

- These safety notes do claim to be complete. In case of questions and problems please contact your Lenze representative.
- At the time of delivery the fieldbus module meets the state of the art and ensures basically safe operation.
- The indications given in these Operating Instructions refer to the stated hardware and software versions of the fieldbus modules.
- The fieldbus module is hazardous if:
  - unqualified personnel works on and with the fieldbus module.
  - the fieldbus module is used inappropriately.
- The processing notes and circuit sections shown in these Operating Instructions are proposals which cannot be transferred to other applications without being tested and checked.
- Ensure by appropriate measures that neither personal injury nor damage to property may occur in the event of failure of the fieldbus module.
- The drive system must only be operated when no faults occur.
- Retrofittings, modifications, or redesigns are basically prohibited.Lenze must be contacted in all cases.
- The fieldbus module is electrical equipment intended for use in industrial high-power plants. The fieldbus module must be tightly screwed to the corresponding controller during operation. In addition, all measures described in the Operating Instructions of the used controller must be taken. Example: Fasten covers to ensure protection against contact.

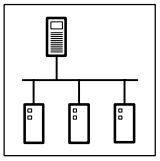
2.3 Layout of the safety information

- All safety information has a uniform layout:
  – The icon characterizes the type of danger.
  – The signal word characterizes the severity of danger.
  – The note text describes the danger and gives information how to prevent dangerous situations.

Signal word

Note

	Icons used		Signal words	
Warning of damage to persons		Warning of hazardous electrical voltage	**Danger!**	Warns of **impending danger**. Consequences if disregarded: Death or severe injuries.
		Warning of a general danger	**Warning!**	Warns of **potential, very hazardous situations**. Possible consequences if disregarded: Death or severe injuries.
			Caution!	Warns of **potential, hazardous situations**. Possible consequences if disregarded: Light or minor injuries.
Warning of damage to material			**Stop!**	Warns of **potential damage to material**. Possible consequences if disregarded: Damage of the controller/drive system or its environment.
Other notes			**Tip!**	This note designates general, useful notes. If you observe it, handling of the controller/drive system is made easier.



3 Technical data

3.1 General information

The internationally standardized CAN bus protocol, which had been developed for the European Automobile Industry, is mainly characterized by:

- its resistance against interference and extreme temperatures
- short transfer times
- low connection expenses

These advantages have made CAN products interesting for other industries too.

The CANopen communication profiles is based on CAN technology.

The CANopen protocol has been specified by drive, control, and sensor manufacturers.

The protocol has been developed by the **CiA** (**C**ON **i**n **A**utomation) in conformity with the **CAL** (**C**ON **A**pplication **L**ayer). All mandatory parts of the CiA DS301 protocol, version 4.01 have been implemented in the 2175 bus module.

3.2 Features

- Attachable module for the following Lenze controller series: 82XX, 8200 vector, 93XX 9300 servo PLC.
- The front DIP switch enables easy setting of
  – Communication profile DeviceNet or CANopen
  – Baud rate of 10, 20, 50, 125, 250 , 500 and 1000 kbit/s (depending on the communication profile)
  – Node address (max. 63 participants)
- Bus extension up to max. 5000m
- Topology: Line terminated at both ends (R = 120 Ohm)
- Easy connection because of pluggable screw terminals

3.3 General data and application conditions

For	Values						
Order name	33.2175IB						
Communication media	DIN ISO 11898						
Baud rate [kbit/s]	10	20	50	125	250	500	1000
Maximum cable length [m]	5000	2500	1000	550	250	100	25
Ambient temperature	during operation:		-20 °C	to	60 °C		
	during transport:		-25 °C	to	70 °C		
	during storage		-25 °C	to	60 °C		
Permissible humidity	Class 3K3 to EN 50178 (without condensation, average relative humidity 85%)						
24-V-DC-Voltage supply	• external supply only: 820X • internal or external supply: 9300 servo PLC / 93XX / 821X / 822X / 8200 vector (see chapter 4.3.2)						

3.4 Rated data

For	Values		
Voltage supply	24 V DC ± 10 %; max. 100 mA		
Communication medium	ISO 11898		
Insulation voltages for bus systems		**Rated insulation voltage**	**Type of insulation**
	• PE	50VAC	Electrical isolation
	• External supply (term. 39/59)	-	No electrical isolation
	• Power stage		
	– 820X / 821X	270 V AC	Basic insulation
	– 822X / 8200 vector	270 V AC	Double insulation
	– 93XX	270 V AC	Double insulation
	• Control terminals:		
	– 820X / 8200 vector (with internal supply)	-	No electrical isolation
	– 8200 vector (with external supply)	100 V AC	Basic insulation
	– 821X	50 V AC	Electrical isolation
	– 822X	270 V AC	Basic insulation
	– 93XX	270 V AC	Basic insulation
	• External bus systems	0 V AC	No electrical isolation
Pollution degree	VDE0110, part 2, pollution degree 2		

3.5 Dimensions

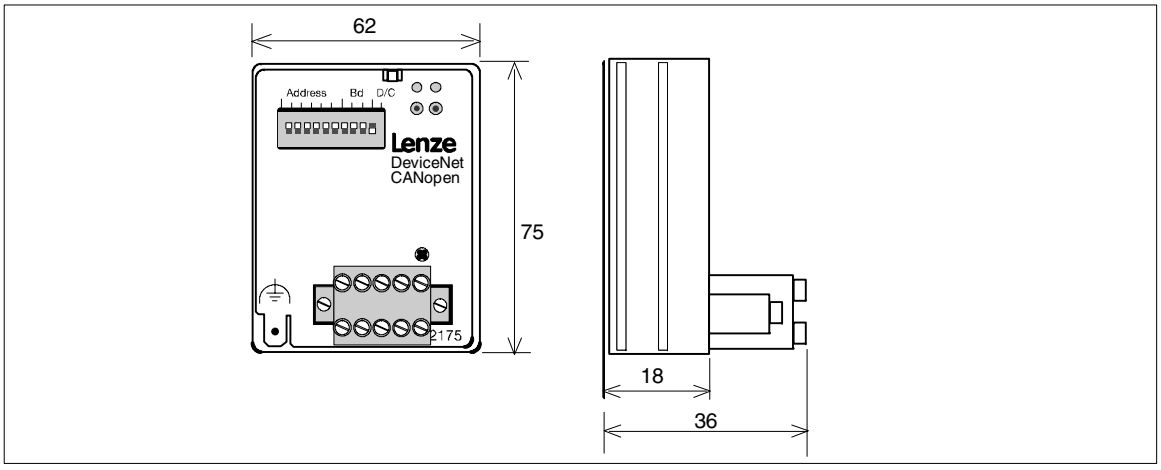
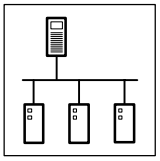


Fig. 3-1 Dimensions: 2175 fieldbus module (all dimensions in mm)



3.5.1 Communication times

Note!

The CAN bus communication times depend on

- Processing time in the controller
- Baud rate
- Data priority
- Bus load

More information about bus access control can be obtained from corresponding literature specialised on Controller Area Networks.

3.5.1.1 Processing times in the controller

Processing times of 820X controllers

The processing times for 8200 controllers differ from the times for the 821X/822X/8200 vector series.

In opposite to the 821X/822X/824X series, which have parallel process data processing, the 8200 series process process and parameter data sequentially. Therefore the time needed to respond process data depends on previous actions.

The processing time needed for telegrams also depends on the actual value conditioning (process data from controller). If these data (status word, actual frequency) are not required, they can be deactivated with the control word "Bit 15" (PE inhibit).

The individual telegram times are:

Telegram	Processing time	
	PE-inhibit = 0	PE-inhibit = 1
Parameters	62...140 ms	62...70 ms
Change of a process data value to controller (*)	27...105 ms	27...35 ms
Change of both process data values to controller *	62...140 ms	4...70 ms
Process data from controller *	108...140 ms	not possible

Processing times of 821X/8200 vector/822X controllers:

Parameters 30...50 ms

Process data, 3...5 ms (*)

* The process data processing times refer to the sync telegram (📖 6-13)

Processing time of 93XX controllers

Parameter data and process data are independent of each other.

- Parameter data: approx. 30 ms + 20 ms tolerance (typically)
  – Some codes require longer processing times (see the 9300 Manual).
- Process data: approx. 3 ms + 2 ms tolerance

3.5.1.2 Telegram time

Telegram times depend on the baud rate and telegram length:

Baud rate [kbit/s]	**Data length** [byte]		
	0	**2**	**8**
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

Tab. 3-1 Maximum telegram time in [ms]

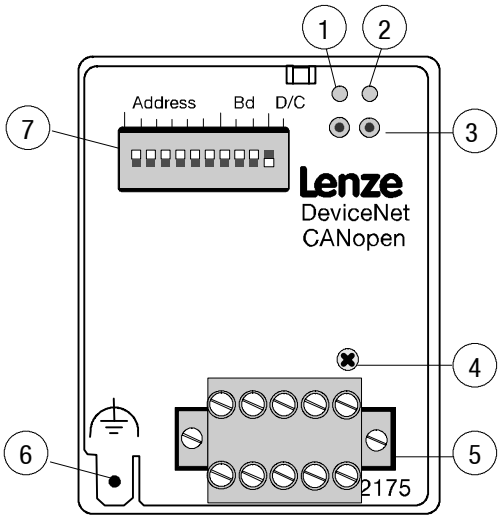
The telegram times indicated in the table above are calculated according to the following equation. This equation allows to calculate any intermediate value t_{Tmax} if necessary.

$$t_T \leq \frac{54.4 + 9.6 \cdot L_D}{d_{\ddot{U}}}$$

t_T = telegram time [ms]
L_D = telegram length [byte]
$d_{\ddot{U}}$ = baud rate [kbit/s]

4 Installation

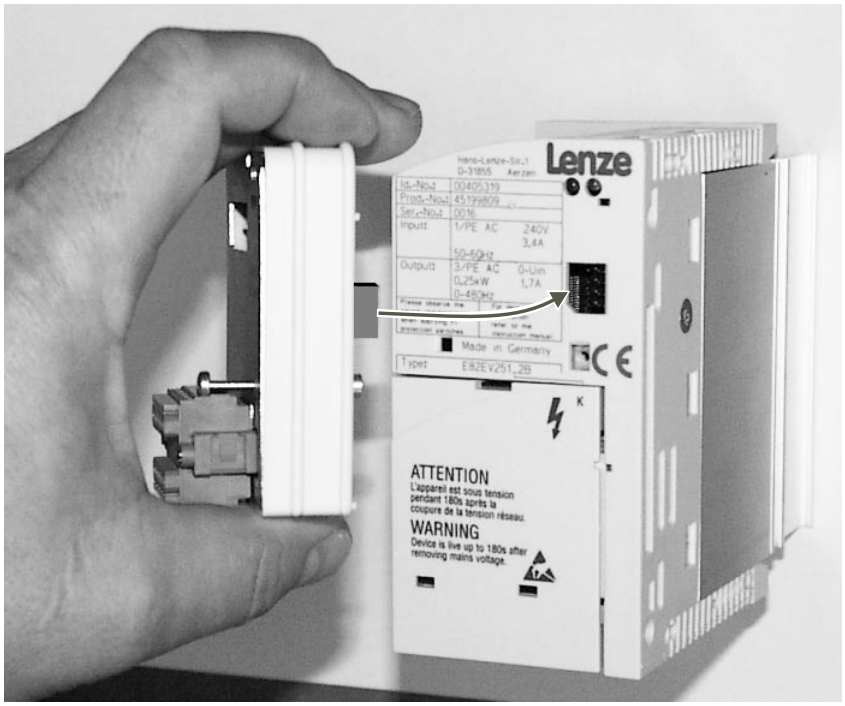
4.1 Components of the fieldbus module



Pos.	Designation	Meaning		Notes
1	**Controller connection status** (two-colour LED)	OFF		2175 fieldbus module is not supplied with voltage; controller or external voltage supply is switched off.
		GREEN	BLINKING	2175 fieldbus module is supplied with voltage but is not connected to the controller (controller is switched off, in initialisation or not available).
			Constantly ON	2175 fieldbus module is supplied with voltage and connected to the controller.
2	**Bus connection status** (two-colour LED)	OFF		• No communication with the fieldbus module • Fieldbus module is not supplied with voltage
		GREEN	BLINKING	Communication via the fieldbus has been set up
		RED	ON	Internal fault of the fieldbus module
3	**Green and red drive LEDs (drive)**	Operating status of the following controllers: 82XX, 8200 vector, 93XX and servo PLC 9300 (see Operating Instructions for the controller)		
4	**Fixing screw**			
5	**5-pole plug-in connector**			
6	**Connection PE shield cable**			Only for 820X and 821X: If necessary use an additional PE shield cable which avoids EMC-related communication interference in especially noisy environments.
7	**DIP switch**	For settings see chapter 5		

4.2 Mechanical installation

- Plug the fieldbus module onto the controller (here: 8200 vector)



2175DeN020

- Tighten the fixing screw (📖 4-1) Pos. 4)

2175DeN021

4.3 Electrical installation

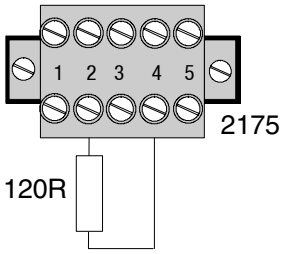
Note!

The communication of 820X and 821X controllers can be interfered by electromagnetic interferences.

If necessary, use an additional PE shield cable at position 6 (📖 4-1)

4.3.1 Assignment of the plug/socket connector

The 2175 fieldbus module is connected to the bus through a 5 pole plug/socket connector.



Terminal	Designation	Explanation
1	V-	GND; reference for external voltage supply
2	CAN_L	Data cable / input for terminating resistance of 120 Ohm
3	SHIELD	Shield
4	CAN_H	Data cable / input for terminating resistance of 120 Ohm
5	V+	External voltage supply; see notes in chapter 10.3.3

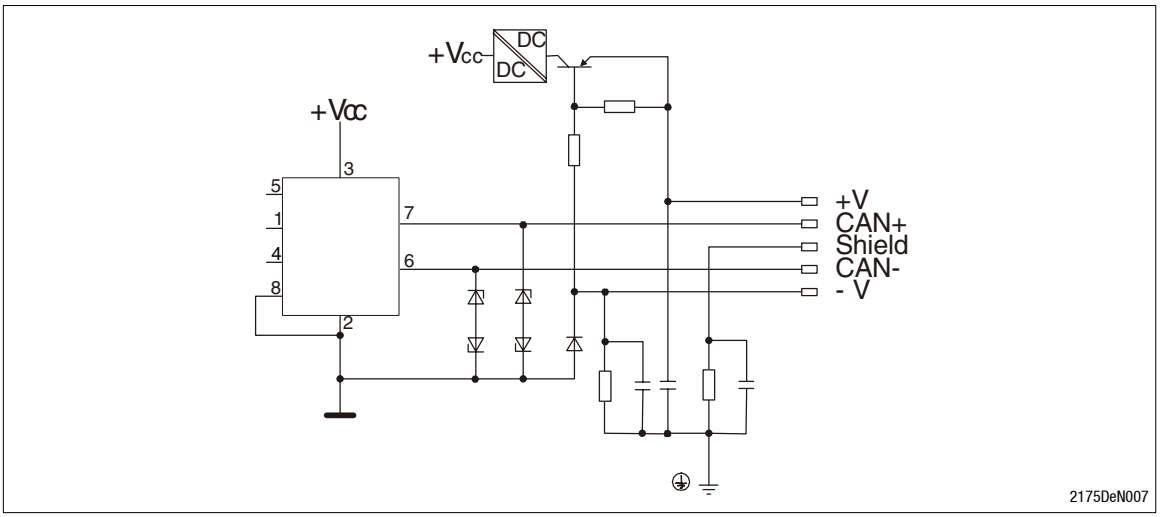


Fig. 4-1 Terminal assignment

4.3.2 Voltage supply

If necessary, supply the 2175 fieldbus module via the plug-in contacts V+/V- (📖 4-3) using a separate voltage supply of 24 V DC ± 10 %.

820X controllers always require a separate voltage supply.

Use a separate supply unit in every control cabinet if the distance between the control cabinets is larger than normal.

Controller	External voltage supply
820X	always required
821X / 822X / 824X and 93XX	Only necessary if the mains which supply the corresponding controllers is to be switched off but communication must not be interrupted.
8200 vector	see below

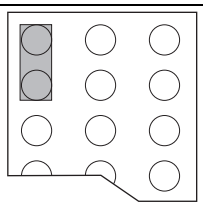
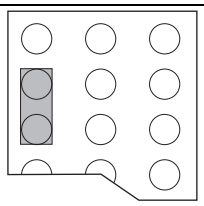
Chapter 4.4 describes how to connect the 2175 fieldbus module to the bus system.

Note!

Internal voltage supply of the fieldbus module connected to a 8200 vector
(only applies to controllers as of nameplate data 82EV 1x 1x)

Controllers with an extended AIF interface (front of the 8200 vector) can be internally supplied. The part of the drawing highlighted in grey shows the jumper position.

With Lenze setting, the fieldbus module is not internally supplied.
For internal voltage supply, put the jumper in the position indicated below.

Lenze setting (only external voltage supply)	Internal voltage supply
	

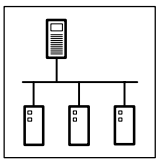
4.3.3 Wiring to a host

Warning!

An additional electrical isolation is required if

- a 820X, 821X or 8200 vector controller is connected to a host
- a safe electrical isolation (double basic insulation) to VDE 0160 is required.

For this, you can use an interface module for the host with an additional electrical isolation (see the corresponding manufacturer's information).
For wiring, the electrical isolation of the supply voltage must be taken into account. The supply voltage is assigned to the same potential as the data bus.



4.4 Structure of a CAN bus system (Example)

The CAN bus system is designed as 2 conductor type (twisted pair) shielded with additional mass and termination at both ends of the line (see (📖 4-6)).

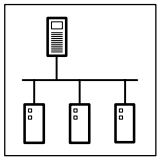
Data can be send and received as follows:

- Max. 3 process data channels (PDO = Process Data Object)
  – Process data are send via the process data channel and are used for high-speed and high-priority control tasks. Typical process data are, for instance, setpoints and actual values for a controller.
- 2 parameter data channels(SDO = Service Data Object)
  – Parameters are transferred at lower priority than process data. Parameters are set or changed during, for instance, commissioning or changing the product.
  – Parameters are accessed via the parameter data channel of the 2175 fieldbus module to the Lenze controller codes or corresponding CANopen objects (detailed description in chapter CANopen parameter setting).
  – 2 masters can be connected to the controllers because of the 2 parameter channels. Thus, parameters can be changed directly at the controller during operation of a machine or system networked via PLC, using a PC (e.g. with the user software Global Drive Control) or a keypad. The second parameter data channel can be accessed with an offset of 64 via the address set (switch or L-C0009). For instance, if a PLC addresses the controller with address 1 and a second commanding unit address 65, the same controller will be addressed. Please note that the last telegram determines a parameter when it is accessed by two units (see “Server SDO Parameters” (📖 6-35)).

Note!

Please note that the last telegram determines a parameter when it is accessed by two units.

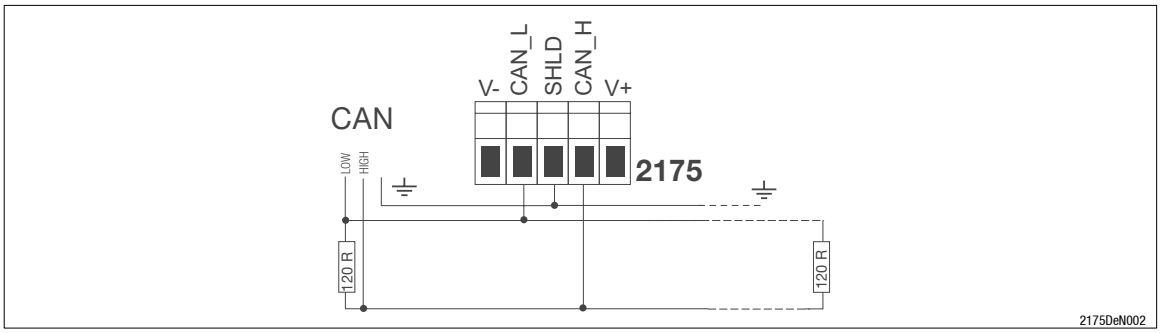
See chapter 5, if you do not use the front switch to enter baud rate and address.



CANopen

Installation

4.4.1 CAN bus wiring



Please observe our recommendations for signal cables:

Total length up to 300 m	
Cable type	LIYCY 2 x 2 x 0.5 mm^2 (twisted in pairs with shield)
Cable resistance	≤ 40 Ω/km
Capacitance per unit length	≤ 130 nF/km

Total length up to 1000 m	
Cable type	CYPIMF 2 x 2 x 0.5 mm^2 (twisted in pairs with shield)
Cable resistance	≤ 40 Ω/km
Capacitance per unit length	≤ 60 nF/km

5 Commissioning

Tip!

Settings using GDC or a keypad

Controller address and baud rate can be set using GDC or a keypad. For this, the DIP-switches S1 to S6 must be set to OFF.

Codes L-C0009 (controller address) and L-C0125 (baud rate) are not active if one or several switches have been set to ON before a restart.

Please see the descriptions for

- Node address L-C1850/2350 (📖 6-46)
- Baud rate L-C1851/2351 (📖 6-47).

The DIP switch on the front of the 2175 fieldbus module can be used for the following settings:

- Controller address S1 - S6
- Baud rate S7 - S9
- Communication profile S10

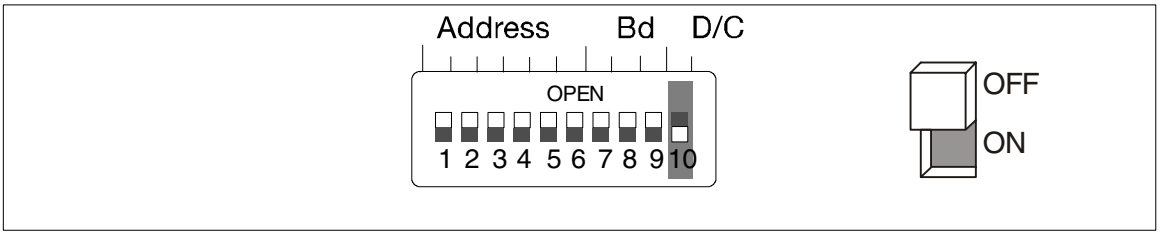
Tip!

In default setting all switches are OFF.

The controller address and baud rate set using DIP switches will only be active after a restart.

Only the combinations shown in the following tables are defined:

5.1 Communication profile setting

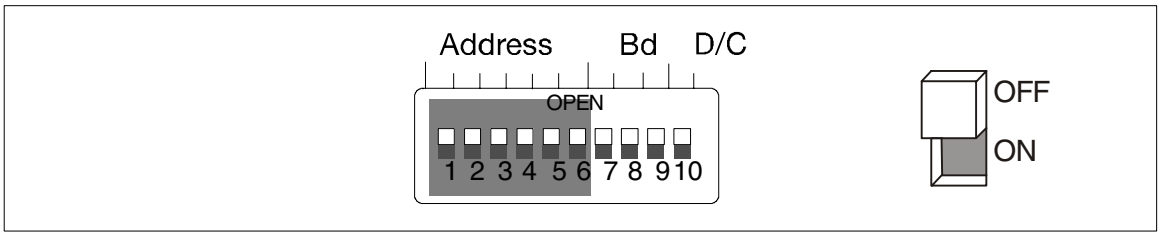


Communication profile	S10
DeviceNet	OFF
CANopen	ON

5.2 Controller address setting

Tip!

Please ensure that the addresses are not the same when using several controllers.



$$Address_{dec} = S_6 \cdot 2^0 + S_5 \cdot 2^1 + S_4 \cdot 2^2 + S_3 \cdot 2^3 + S_2 \cdot 2^4 + S_1 \cdot 2^5$$

The address calculation (decimal number) is based on the positions of switches S1 ... S6 (’0’ = OFF and ’1’ = ON). The numbers must be entered into the equation above.

The equation also indicates the valency of a switch. The sum of valencies results in the controller address to be set:

Switch valencies:

Switch	S1	S2	S3	S4	S5	S6
Valency	32	16	8	4	2	1

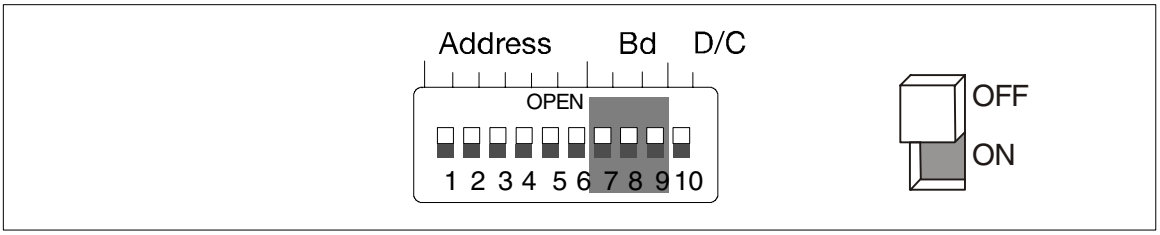
Example:

Switch	S1	S2	S3	S4	S5	S6
Switch position	ON	ON	ON	OFF	OFF	OFF
Address (= 56)	32	16	8	0	0	0

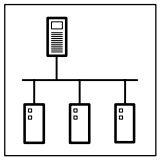
5.3 Baud rate setting

Tip!

Please ensure that the baud rate is the same for all controllers and the host.



Baud rate [kbit/s]	S7	S8	S9
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



5.4 Initial switch-on

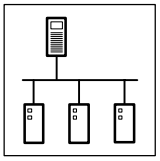
Please do not change the switch-on sequence explained below!

Stop!

Prior to initial switch-on of the controller, check the wiring for

- completeness
- short-circuit
- earth fault

1. Switch on the controller and, if necessary, the external supply of the 2175 fieldbus module.
   – One of the operating status LEDs of the controller (📖 4-1) , pos. 3 must come on or blink. If this is not the case, see chapter ”Troubleshooting” (📖 7-1)
   – The green LED (”Controller connection status”) must be on, too (📖 4-1) (pos. 1). If this is not the case, see chapter ”Troubleshooting” (📖 7-1) .
2. It should now be possible to communicate with the drive:
   – All parameters can be read and written
   – All SDO parameters except process data, such as frequency setpoint or control word, can be overwritten.
   – For more information about the communication phases of the CAN network see (📖 6-3) .



5.5 Drive enable via 2175 fieldbus module

Tip!

If the 2175 module is plugged onto a different controller during operation, an undefined operating status might occur.

82XX / 8200 vector	1. Set the Lenze parameter Operating Mode (L-C0001) from 0 to 3 to enable the driva via the 2175 fieldbus module. The parameter can be set using the keypad or directly via CANopen. Examples for write (L-C0001=3): – Index = $5FFE_{hex}$ (results from $5FFF_{hex}$ – $(L\text{-}C0001)_{hex}$; see Lenze Codes, Addressing, chapter 6.2 📖 6-5) – Subindex: 0 – Value: 30000_{dec} (results from: L-C0001 = 3 x 10000)
	2. Terminal 28 (controller enable) is always active and must be set to HIGH during CANopen operation (see the corresponding Operating Instructions for the controller). The controller can also be enabled via CANopen. – With 821X, 8200vector and 822X, the function QSP (quick stop) is always active. If QSP is assigned to an input terminal (factory setting: not assigned), this terminal must be at HIGH level during CANopen operation (see the corresponding Operating Instructions). The controller now accepts parameter and process data.
93XX	1. For drive control via CANopen set the Lenze parameter Signal Configuration (L-C0005) to xxx3. These changes can be made using the 9371BB keypad or directly via CANopen. For first commissioning, select signal configuration 1013. Examples for write (L-C0005=1013): – Index = $5FFA_{hex}$ (results from $5FFF_{hex}$ – $(L\text{-}C0005)_{hex}$; see chapter 6.2 📖 6-5) – Subindex: 0 – Value: 10130000_{dec} (results from: L-C0005 = 1013 x 10000) 2. Set the parameter L-C0142 to 0. Please see chapter “Protection against uncontrolled restart“. 3. Terminal 28 (RFR = controller enable) is always active and must be set to HIGH during CANopen operation (see the Operating Instructions for 93XX). The controller can also be enabled via CANopen. – With the signal configuration L-C0005=1013, the function QSP (quick stop) and the CW/CCW changeover are assigned to the digital input terminals E1 and E2 and thus they are always active. For CANopen operation E1 must be set to HIGH level (see Operating Instructions for 93XX).
	With the signal configuration L-C0005=xx13, terminal A1 is switched as voltage output. Connect the following terminals: • X5.A1 with X5.28 (ctrl. enable) • X5.A1 with X5.E1 (CW/QSP)
	The controller now accepts parameter and process data.

5.5.1 Protection against uncontrolled restart

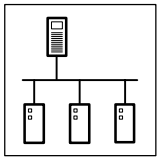
Tip!

After a fault (e.g. short-term mains failure) a restart of the drive is not always wanted.

- By setting L-C0142 = 0, the drive can be inhibited if
  – the corresponding controller sets a “Message“ fault
  – the fault is active for more than 0.5 s

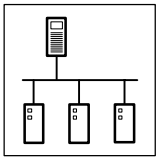
Parameter function:

- L-C0142 = 0
  – Controller remains inhibited *(even if the fault is no longer active)* and
  – The drive restarts in a controlled mode: LOW-HIGH edge at one of the inputs for “Controller inhibit“ (CINH, e.g. at terminal X5/28)
- L-C0142 = 1
  – Uncontrolled restart of the controller possible



CANopen

Commissioning



6 Parameter setting

Master and slave communicate with each other by sending data telegrams via the CAN bus. The user data range of a data telegram contains either network management data, parameter data or process data..

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

Telegram type		Communication channel	
Parameter data (SDO, Service Data Objects)	These are, for instance, • operating parameters • diagnostics information • motor data In general, the parameter transfer is not as time-critical as the tranfer of process data.	**Parameter data channel** (Chapter 6.2)	• Enables access to all Lenze codes and the CANopen index. • Parameter changes are automatically stored in the controller (take L-C0003 into account).
Process data, (PDO, Process Data Objects)	These are, for instance, • Setpoints • Actual values Exchange between host and controller required as fast as possible. Small amounts of data which can be transferred cyclically.	**Process data channel** (chapter 6.3)	• The controller can be controlled using process data. • The host has direct access to process data. In the PLC the data are, for instance, directly assigned to the I/O area. • Process data are – not stored in the controller. – cyclically transferred between host and controller to ensure a continuous exchange of current input and output data.

Tab. 6-1 Division of parameter data and process data in different communication channels

The following regulations for a communication protocol only describe what is related to Lenze controller networkds.

6.1 Structure of a CAN data telegram

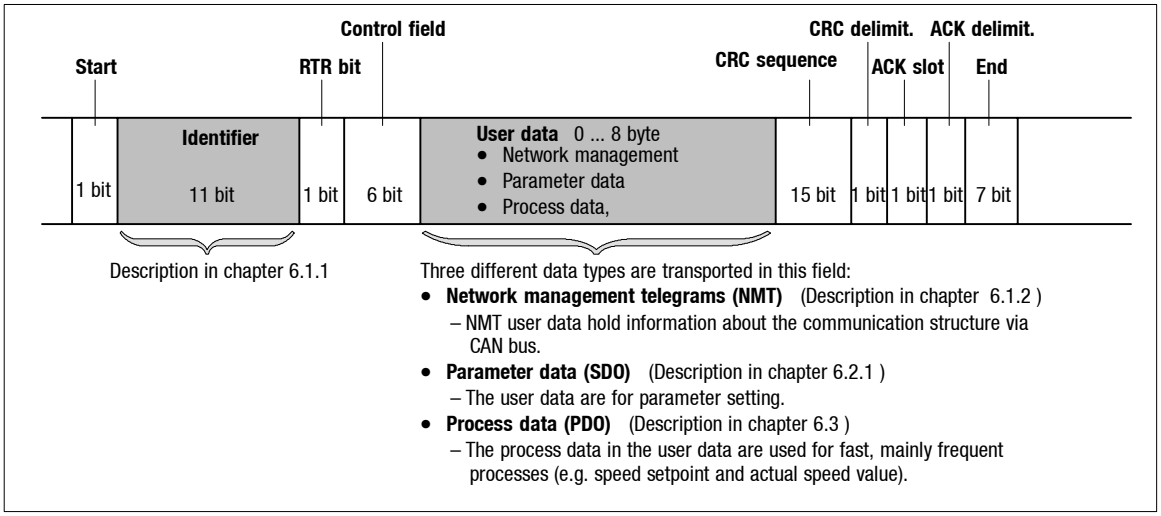


Fig. 6-1 Structure of a CAN data telegram

The data relevant for programming the bus module (identifiers and user data) will be described in detail in the corresponding chapters.

All other signals refer to the transfer features of the CAN telegram. These Instructions do not describe them in detail. For more information please see the homepage of “CAN in Automation (CiA)”: www.can-cia.de.

6.1.1 Identifiers

The CAN bus system is message-oriented. One of the most important components of a data telegram is the identifier. Except for the network management (see chapter 6.1.2) and the sync telegram (see chapter 6.3) the identier contains the controller address.

Identifier = basic identifier + controller address

With CANopen the controller address is used for a participant-oriented message addressing.

The identifier assignment is determined in the CANopen protocol. The basic identifier is default set as described in the following:

	Direction		Basic identifier		+ Controller address
	from the controller	to the controller	dec	hex	
Network manager (NMT)			0	0	No
Sync telegram			128	80	
Process data channel 1	X		384	180	yes
		X	512	200	
Process data channel 2	X		640	280	
		X	768	300	
Process data channel 3	X		896	380	
		X	1024	400	
Parameter data channel 1	X		1408	580	
		X	1536	600	
Parameter data channel 2	X		1472	5C0	
		X	1600	640	
Node guarding			1792	700	

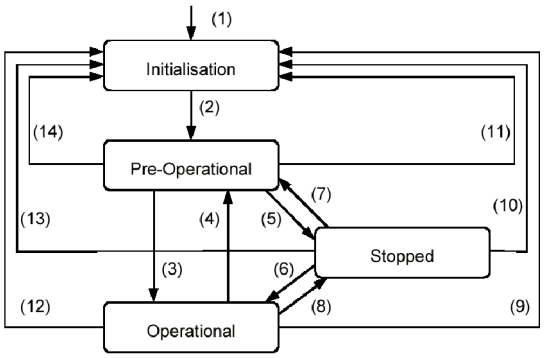
6.1.2 Network management (NMT)

The telegram used for network management contains an identifier (see chapter 6.1.1) and the command which is part of the user data and consists of command byte and controller address.

Identifier		User data (2 byte)
0000000000		1. byte: command 2. byte: controller address

User data

1. byte: command	2. byte: controller address
01, 02, 80, 81 or 82	**Controller address: xx** "xx" in the table means: • **xx = 00_{hex}** With this assignment, all controllers connected are addressed by the telegram. All controllers can change their status at the same time. • **xx = Controller address** If a certain address is indicated, the status will only be changed for the controller addressed.



Initialisation (1):
Initialisation starts automatically when the mains is switched ON or the hardware is reset (1).
The drive does not take part in the data transfer. Various initialisation activities are carried out. This status is reached after the controller has been switched on. Furthermore it is possible to restart the entire initialisation phase or parts of it by transferring different telegrams. All parameters already set are overwritten with their standard values. After initialisation (2) has been completed the participant is automatically set to the status "Pre-operational". From that moment on the master determines the controller communication. The master changes a status for the whole network. A target address, which is part of the command, selects the slaves.

Status transfer	Command	Network status after change	Effect on process and parameter data
(1)	Initialisation		For description see above
(3), (6)	01 xx	Operational	Process and parameter data active
(2), (4), (7)	80 xx	Preoperational	Only parameter data active
(5), (8)	02 xx	Stopped	Parameter and process data cannot be received. Network management telegrams can be received.
(9)	81 xx	Initialisation	Resets the drive, all parameters are overwritten with standard values (like "Reset node")
(10)			
(11)			
(12)	82 xx		Resets the drive, only communication-relevant parameters are reset
(13)			
(14)			

6.1.2.1 Node/life guarding

If you multiply guard time (chapter 6.4.1.8) and life time factor (chapter 6.4.1.9) the result is a time. This time must not be exceeded

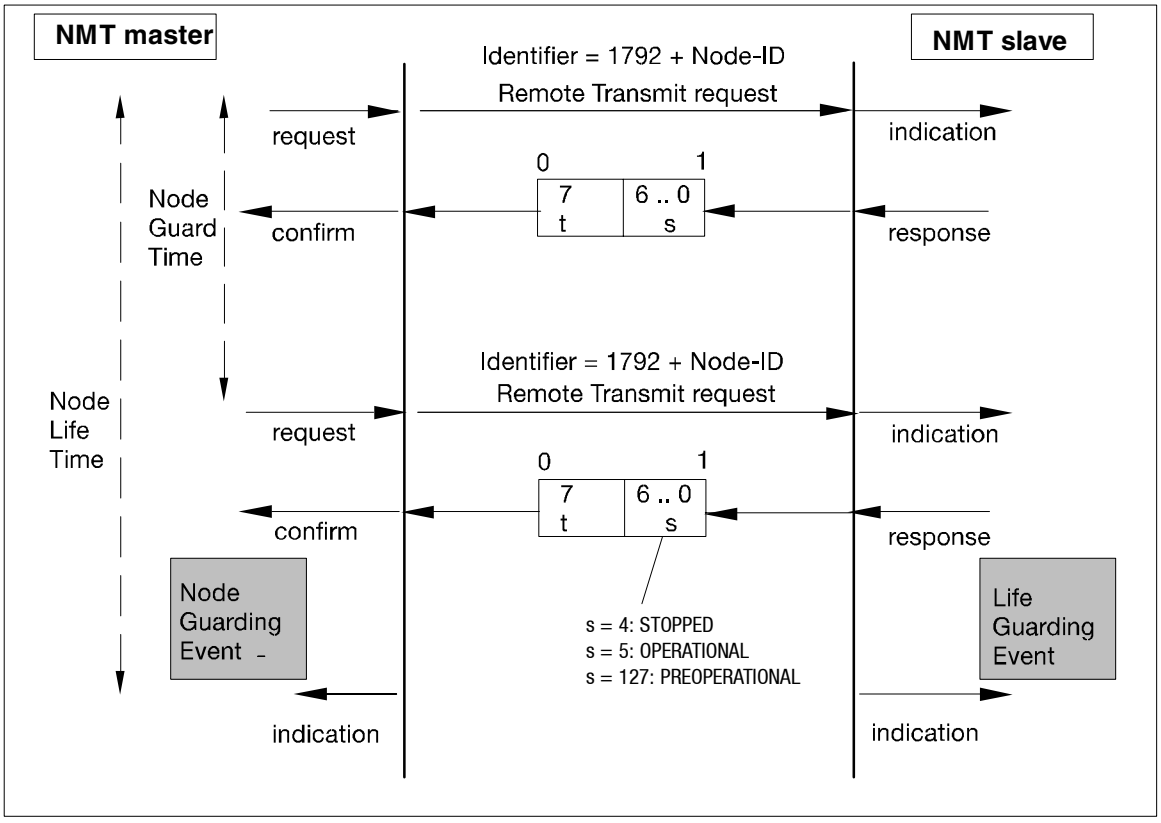
- when a request is sent to a slave
- a slave sends a response to a request from the master

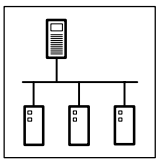
If the master does not send a request, a “life guarding event” will be initialised.

If the slave does not send a response to a request from the master, a “node guarding event” will be initialised.

Tip!

The response to a “life guarding event” is set under code L-C1882 / L-C2382.





6.2 Parameter data channel

Note!

- Lenze codes have a L in front of the code „L-Cxxxx“ so that they cannot be mixed up with the CANopen index.
  – Example: ’L-C0001’ stands for Lenze code C0001.
- Please obtain the value range for Lenze codes from the corresponding Operating Instructions (see ’Code list’).

Access to controller codes

When using fieldbus modules, a host (e.g. a PLC) can change the features and response characteristics of any controller connected to the network.

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

Controller codes are addressed via the index when accessing the code through a 2175 fieldbus module (see chapter 6.2.1).

The index for Lenze code numbers is between 16576 ($40C0_{hex}$) and 24575 ($5FFF_{hex}$).

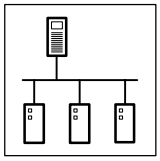
Conversion formula:

Index[*dec*] = 24575 − *Lenze code number*

Example:

Lenze codes		dec	hex
	• Addressing of Lenze codes via offset:	Index = 24575 - LENZE CODENO	$Index_{hex}$ = $5FFF_{hex}$ - $LENZECODENO_{hex}$
	– Example for operating mode L-C0001	Index = 24574 (= 24575 - 1)	$Index_{hex}$ = $5FFE_{hex}$ (= $5FFF_{hex}$ - 1)

The parameter value is part of the telegram user data (see expamples (📖 6-10)).



CANopen

Parameter setting

Lenze parameter sets

Parameter sets are for special code saving which is necessary because of different configurations for different application processes.

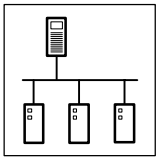
The following table informs about number and addressing of parameter sets for your controller:

<table>
<tr><th>82XX</th><th>8200 vector</th><th>93XX</th></tr>
<tr><td colspan="2">The 82XX and 8200 vector have 2 and 4 parameter sets. The parameters can be directly addressed via CAN.
They are addressed by means of a code-digit offset:</td><td rowspan="6">93XX controllers provide up to 4 parameter sets per technology variant to save data in the EEPROM. Another parameter set is in the user memory of the controller. This is the current parameter set. Only the current parameter set can be directly addressed via CAN. For the codes, see the Operating Instructions or the Manual for 93XX. Changes of the current parameter set will be lost after switching off the controller. Code C0003 is for saving the current parameter set. After switching on the controller, parameter set 1 is automatically loaded into the current parameter set.
Parameter sets 2 - 4 must have been activated before the parameters can be changed.</td></tr>
<tr><td colspan="2">• Offset 0 addresses parameter set 1 with Lenze codes L-C0000 to L-C1999
• Offset 2000 addresses parameter set 2 with Lenze codes L-C2000 to L-C3999</td></tr>
<tr><td>No additional parameter sets available.</td><td>• Offset 4000 addresses parameter set 3 with Lenze codes L-C4000 to L-C5999
• Offset 6000 addresses parameter set 4 with Lenze codes L-C6000 to L-C7999</td></tr>
<tr><td colspan="2">If a parameter is available only once (see Operating Instructions for 82XX or 8200 vector), use code digit offset 0.</td></tr>
<tr><td colspan="2">Example for L-C0011 (maximum field frequency):
L-C0011 in parameter set 1: Lenze code = 11
L-C0011 in parameter set 2: Lenze code = 2011</td></tr>
<tr><td>-</td><td>L-C0011 in parameter set 3: Lenze code = 4011
L-C0011 in parameter set 4: Lenze code = 6011</td></tr>
<tr><td colspan="2">Parameter changes:
82XX: Automatic saving in the controller
8200 vector: Automatic saving as default setting (can be switched off with L-C0003)
Process data changes:
82XX, 8200 vector: no automatic saving</td></tr>
</table>

Stop! (only for 8200 vector and 82XX)

Please take into account that cyclic writing of parameter data to the EEPROM is not permitted.

Only 8200 vector:
Configure code C0003 = 0 after every mains switching if you want to change parameter data cyclically.



6.2.1 Structure of a parameter data telegram

User data (up to 8 byte)							
1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Command	**Index** Low byte	**Index** High byte	**Subindex**	**Data 1**	**Data 2**	**Data 3**	**Data4**
				Low Word		High Word	
				Low byte	High byte	Low byte	High byte
				Error message			

Parameters can be set via two separate software channels which are selected by means of the controller address (chapter 6.1.1).

Tip!

User data are displayed in the left-justified INTEL format.

For calculation examples see chapter 6.2.2.

Details about the parameter data telegram:	1st byte
	Command

The command contains the follwing information which must be entered if not already indicated:

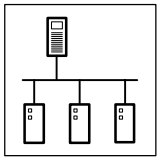
Command	**Access to Data 1 - Data 4**			**Block** (chapter 6.2.4)
	4 byte data (5th - 8th byte)	**2 byte data** (5th + 6th byte)	**1 byte data** (5th byte)	
	hex	hex	hex	hex
Write Request (Send parameters to drive)	23	2B	2F	Writing not possible
Write Response (Controller response to write request (acknowledgement))	60	60	60	
Read Request (Request to read a parameter from the drive)	40	40	40	40
Read Response (Response to the read request with an actual value)	43	4B	4F	41
Error response (Controller indicates communication error)	80	80	80	80

Details about the parameter data telegram:	2nd byte	3rd byte
	Index Low byte	**Index** High byte

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

$$Index = 24575 - (Lenze\text{-}code + 2000 \cdot (parameter\ set - 1))$$

Example	**Calculation**	**Index low/high byte**
Code L-C0012 (acceleration time) in parameter set 1 is to be addressed.	24575 - 12 - 0 = 24563 = $5FF3_{hex}$	According to the left-justified Intel data format the entries are (see chapter 6.2.1): Index low byte = $F3_{hex}$ Index high byte = $5F_{hex}$
Code L-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 = 5823_{hex}	According to the left-justified Intel data format the entries are (see chapter 6.2.1): Index low byte = 23_{hex} Index high byte = 58_{hex}



CANopen

Parameter setting

Details about the parameter data telegram:	4th byte
	Subindex

Table position of a parameter value using the index.

Example:
L-C0465 (function block processing list)
Entry of subcode: 1 ... 50 (dez) bzw. 1 ... 32 (hex)

Explanation for the parameter data telegram	5th byte	6th byte	7th byte	8th byte
	Parameter value (length 1)	00	00	00

Explanation for the parameter data telegram	5th byte	6th byte	7th byte	8th byte
	Parameter value (length 2)		**00**	**00**
	Low byte	High byte		

Explanation for the parameter data telegram	5th byte	6th byte	7th byte	8th byte
	Parameter value (length 4)			
	Low Word		High Word	
	Low byte	High byte	Low byte	High byte

Depending on the data format (see 'Attribute list' in the Manual) the parameter value needs 1 to 4 bytes.

Tip!

Lenze parameters are mainly FIX32 data (32 bit value with signal, decimal with four decimal codes, see the Attribute Table in the corresponding Manual). Integers are the result of a multiplication of a parameter value by 10000.

Parameters C0135 and C0150 must be transferred as bit code and without factor.

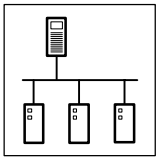
- Error messages in the parameter data telegram

Explanation for the parameter data telegram	1st byte	2nd byte	3rd byte	4th byte	**5th byte**	**6th byte**	**7th byte**	**8th byte**
	Command	Index Low byte	Index High byte	Subindex	**Error code**			

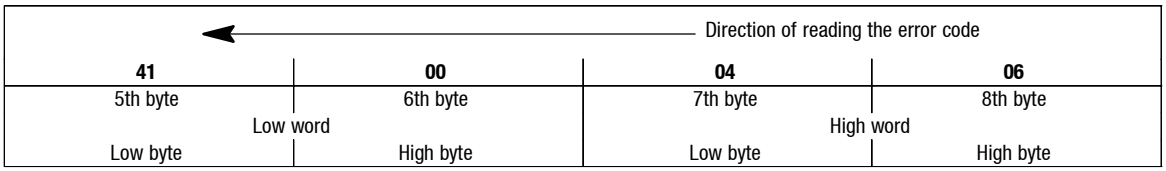
Byte 1:
The **Command** byte shows in code $\mathbf{128_{dec}}$ **or** $\mathbf{80_{hex}}$ that an error has occured.

Byte 2, 3 and 4:
The **index byte** and the **subindex** contain index and subindex of the faulty code.

Byte 5 - 8:
In data bytes 5. - 8. the **error code** will be entered.

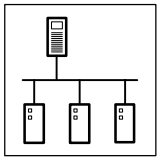


The error code is displayed inverse to the direction of reading.
Example: Error code 06 04 00 41_{hex} and representation of the error code:



The following table lists explanations for the error numbers:

Error code (hex)	**Explanation**
0503 0000	Toggle bit not changed
0504 0000	SDO protocol expired
0504 0001	Invalid or unknown client/server command specifier
0504 0002	Invalid block size (only block mode)
0504 0003	Invalid processing number (only block mode)
0504 0004	CRC error (only block mode)
0504 0005	Not enough memory
0601 0000	Object access not supported
0601 0001	Try to read writeable object
0601 0002	Try to write readable object
0602 0000	Object not listed in object organiser
0604 0041	Object not transferrable to PDO
0604 0042	Number and length of objects to be transferred longer than PDO
0604 0043	General parameter incompatibility
0604 0047	General internal controller incompatibility
0606 0000	Access denied because of hardware error
0607 0010	Inappropriate data type, service parameter length
0607 0012	Inapproriate data type, service parameter length exceeded
0607 0013	Inapproriate data type, service parameters not long enough
0609 0011	Subindex does not exist
0609 0030	Parameter value range exceeded
0609 0031	Parameter values too high
0609 0032	Parameter values too low
0609 0036	Maximum value falls below minimum value
0800 0000	General error
0800 0020	Data cannot be transferred/stored for application
0800 0021	Because of local control data cannot be transferred/stored for application
0800 0022	Becaue of current controller status data cannot be transferred/stored for application
0800 0023	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)



6.2.2 Examples

6.2.2.1 Parameter reading

The heatsink temperature (43 °C) C061 is to be read from the controller using address 5 and parameter channel 1.

- Identifier calculation

Identifier parameter channel 1 to controller	= 1536 + controller address
Identifier	= 1536 + 5 = 1541

- Command read request (request to read a parameter from the drive

Command	= 40_{hex}

- Index calculation

Index = 24575 - code number - 2000 (PS - 1)	Index = 24575 - 61 - 2000 · 0 = 24514 = $5FC2_{hex}$

Telegram to drive:

Identifier	Command	Index Low Byte	Index High Byte	Subindex	Data 1	Data 2	Data 3	Data4
1541	40_{hex}	$C2_{hex}$	$5F_{hex}$	00	00	00	00	00

Telegram from drive

Identifier:
Parameter channel 1 of controller (=1408) + controller address = 1413

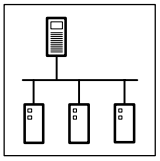
Command:
Read response to read request with the current value of 43_{hex}

Index of read request
$5FC2_{hex}$

Subindex:
0

Data 1 to Data 4:
43 °C • 10.000 = 430.000 = 00 06 8F B0

Identifier	Command	Index Low Byte	Index High Byte	Subindex	Data 1	Data 2	Data 3	Data4
1413	43_{hex}	$C2_{hex}$	$5F_{hex}$	00	$B0_{hex}$	$8F_{hex}$	06_{hex}	00



6.2.3 Write parameter

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

- Identifier calculation

Identifier parameter channel 1 to controller	= 1536 + controller address
Identifier	= 1536 + 1 = 1537

- Command write request (send parameter to drive)

Command	= 23_{hex}

- Index calculation

Index = 24575 - code number - 2000 (PS - 1)	Index = 24575 - 12 - 2000 V 0 = 24563 = $5FF3_{hex}$

- Subindex: 0
- Calculation of the acceleration time

Acceleration-time value	20 s · 10.000 = 200.000 = 00 03 0D 40_{hex}

- Telegram to drive

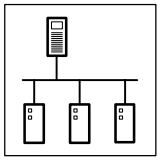
Identifier	Command	Index Low Byte	Index High Byte	Subindex	Data 1	Data 2	Data 3	Data4
1537	23_{hex}	$F3_{hex}$	$5F_{hex}$	00	40_{hex}	$0D_{hex}$	03_{hex}	00

Response of the controller when no fault occurs

Identifier	Command	Index Low Byte	Index High Byte	Subindex	Data 1	Data 2	Data 3	Data4
1409	60_{hex}	$F3_{hex}$	$5F_{hex}$	00	00	00	00	00

Identifier parameter channel 1 from controller = 1408 + controller address = 1409

Command = write response (controller response (acknowledgement)) = 60_{hex}



6.2.4 Read block parameters

The software product code (code L-C0200) of a Lenze product 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 - 8th byte).

During transfer, the command byte (1st byte) contains entries $_{hex}$ or 41_{hex}) to

– indicate the end of a block transfer

– request the next block

- Code L-C0200 - request

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
40_{hex}	37_{hex}	$5F_{hex}$	00	00	00	00	00

1st byte: 40 read request (request to read a parameter from the controller)
2nd/3rd byte: Index Low/High Byte: 24575 - 200 - 0 = 24375 = $5F37_{hex}$

- Response including the block length (14 characters)

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
41_{hex}	37_{hex}	$5F_{hex}$	00	$0E_{hex}$	00	00	00

1st byte: 41 read response. The entry 41_{hex} implies that it is a block telegram.
2nd/3rd byte: see above
5th byte: 0E (=$14_{dec.}$) data length 14 characters (ASCII format)

- First data block - request

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
60_{hex}	00	00	00	00	00	00	00

1st byte: 60_{hex}
Write response (acknowledgement) with access to bytes 2 - 8.
Note:
The individual blocks are toggled one after the other*, i.e. the request with command 60 $_{hex}$ (=0110 0000_{bin}) is before command 70_{hex} (=0111 0000_{bin}) and followed by 60_{hex} again, etc. The response is sent accordingly. It is alternating because of a toggle bit. This process is stopped by command 11_{hex} (bit 0 is set, see below).
*Toggle bit = bit 4 (counting starts with 0)

- Response

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
00	33_{hex}	33_{hex}	53_{hex}	39_{hex}	33_{hex}	30_{hex}	30_{hex}

2nd - 8th byte, ASCII format: 3 3 S 9 3 0 0

- Second data block - request

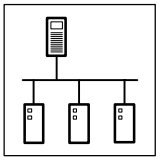
1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
70_{hex}	00	00	00	00	00	00	00

1st byte: 70_{hex} (toggle) write response (acknowledgement) with access to all 4 data bytes

- Second data block - response with over-detection

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
11_{hex}	$4D_{hex}$	$5F_{hex}$	32_{hex}	30_{hex}	30_{hex}	30_{hex}	33_{hex}

1st byte: 11 last transfer of the data block
2nd - 8the byte: M _ 2 0 0 0 3
Result of data block transfer: 33S9300M_20003



6.3 Process data channel

6.3.1 Setpoint source selection

82XX / 8200 vector controllers

The setpoint source is selected under code L-C0001 ($5FFE_{hex}$).

When using AIF fieldbus modules, code L-C0001 (setpoint source selection) must be set to 3 to evaluate process data.

With L-C0001 = 3 the process data channel which describes the frequency setpoint (L-C0046) and the control word (L-C0135) is used as setpoint source (see the corresponding Operating Instructions).

Setpoint source	L-C0001	Read and write parameters	Parameter L-C0046	Process data,
Process data channel of an AIF fieldbus module	3	yes	no	yes

Note!

- Please ensure that the setpoint source (L-C0001) must be set identically for all parameter sets.

93XX controllers

With 93XX controllers it is not possible to select a setpoint source which can be set using just one code. On the other hand, these controllers can be adapted to drive tasks without special programming knowledge, since it uses preconfigured function blocks.

The function blocks can be combined and connected by the user himself. It is however safer to use the preconfiguration provided by Lenze which is stored in the controller. This preconfiguration (code C0005) determines the source (terminal, keyboard, fieldbus module) for frequency setpoint and control word.

For operation with CAN bus code C0005 must be set to “xxx3” (x = space for selected preconfiguration).

More information can be obtained from the corresponding Manual and Operating Instructions.

6.3.2 Process data transfer

Process data telegrams between host and controllers are distinguished as follows:

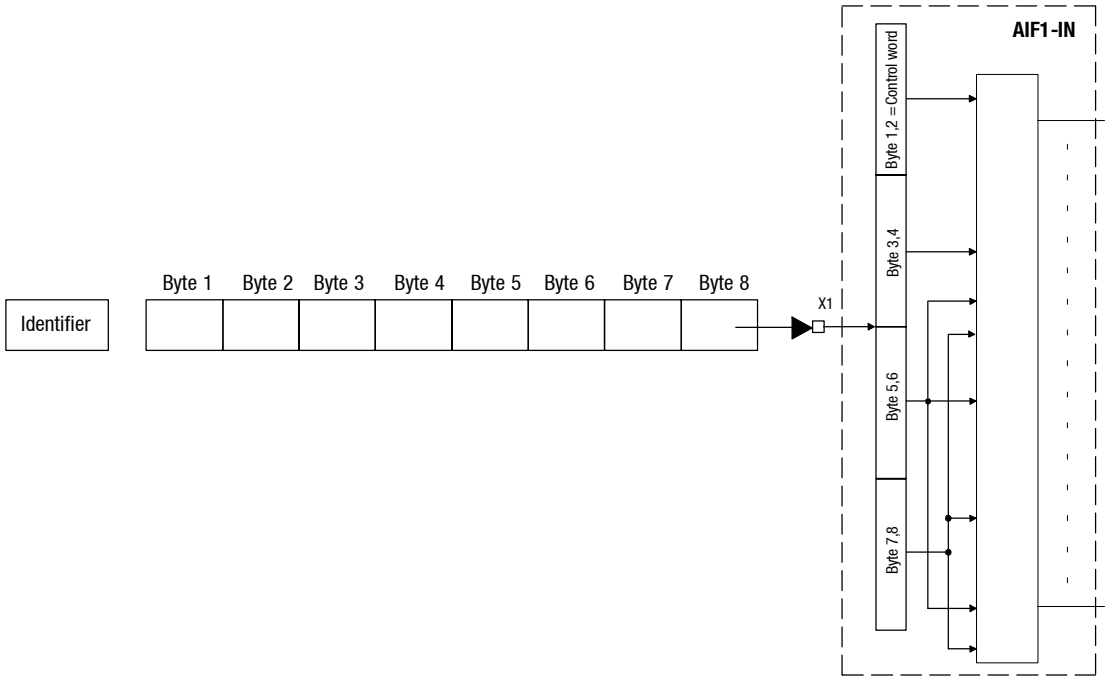
- Process data telegrams **to** drive
- Process data telegrams **from** drive

Process data telegram to drive

The process data telegram includes an indentifier which holds the address (93XX series). This telegram has a user data length of 8 byte (see example below).

The CAN bus is connected to the automation interface X1.

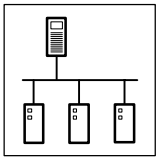
Function AIF-IN is connected to X1. Here the user data for more function blocks are converted into corresponding signal types. The control word is especially important for the drive. Byte 1 and byte 2 of the user data contain the controller setpoint.



Tab. 6-2 Example: 93XX series

Process data telegram from drive

The function block to be used for cyclic process data telegrams from the drive is called AIF-OUT. The status word (byte 1 and byte 2) of the process data telegram is transferred to the CAN bus via this function block and then sent to the master (see also chapter 6.3.5).



To ensure that the process data **to** the controller are accepted, a special telegram, the sync telegram, is required (see chapter 6.1.1).

For cyclic data processing, the sync telegram must be generated accordingly.

Process data synchronisation

The sync telegra is the trigger point for

- accepting data by the controller
- starting the sending process from the controller

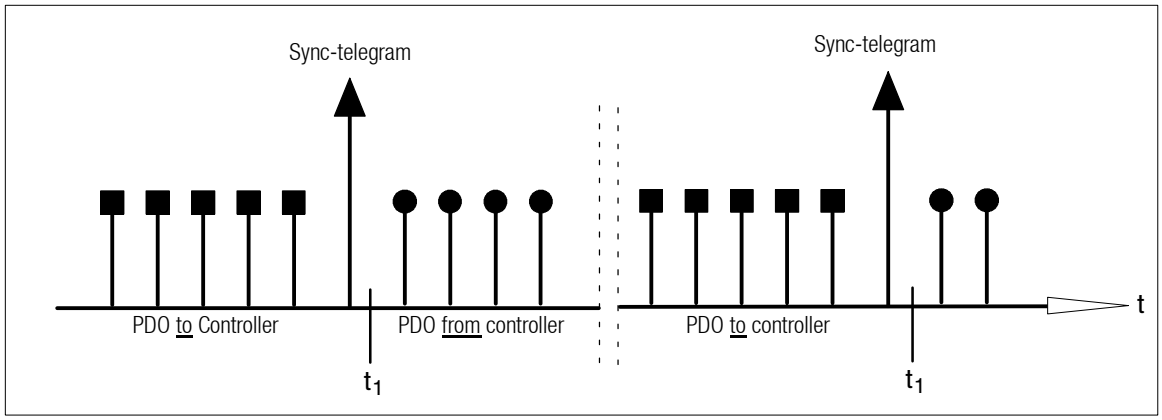


Fig. 6-2 Synchronisation of cyclic PDOs (seen from the bus participant's view)

Note for Fig. 6-2: At t_1 all PDOs accept process data as soon as a sync telegram is received.

Tip!

SDOs or event-controlled PDOs are asynchronously accepted by controllers, i.e. after transmission has been completed.

Asynchronous data have not been taken into account for the figure above.

6.3.3 Process-data assignments for 82XX

Process data telegram to controller

User data length: 8 byte

	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Identifier	Control word L-C0135 Low byte	Control word L-C0135 High byte	Setpoint L-C0046 Low byte	Setpoint L-C0046 High byte	xx	xx	xx	xx

More information about process data Telegram	Byte 1	Bits 0 to 7 of the control word under C0135 are entered here
	Byte 2	Bits 8 to 15 of the control word under C0135 are entered here (see chapter 6.3.4.1). The bit description can be obtained from the Code Table.
	Byte 3	The frequency setpoint, which can also be written as parameter under C046, is entered here as process data word.
	Byte 4	The normalisation differs from the setting under C046. It is a signed value, 24000 = 480 Hz.
	Byte 5	No evaluation of these data, anything possible
	Byte 6	
	Byte 7	
	Byte 8	

Control word: see chapter 6.3.4.1.

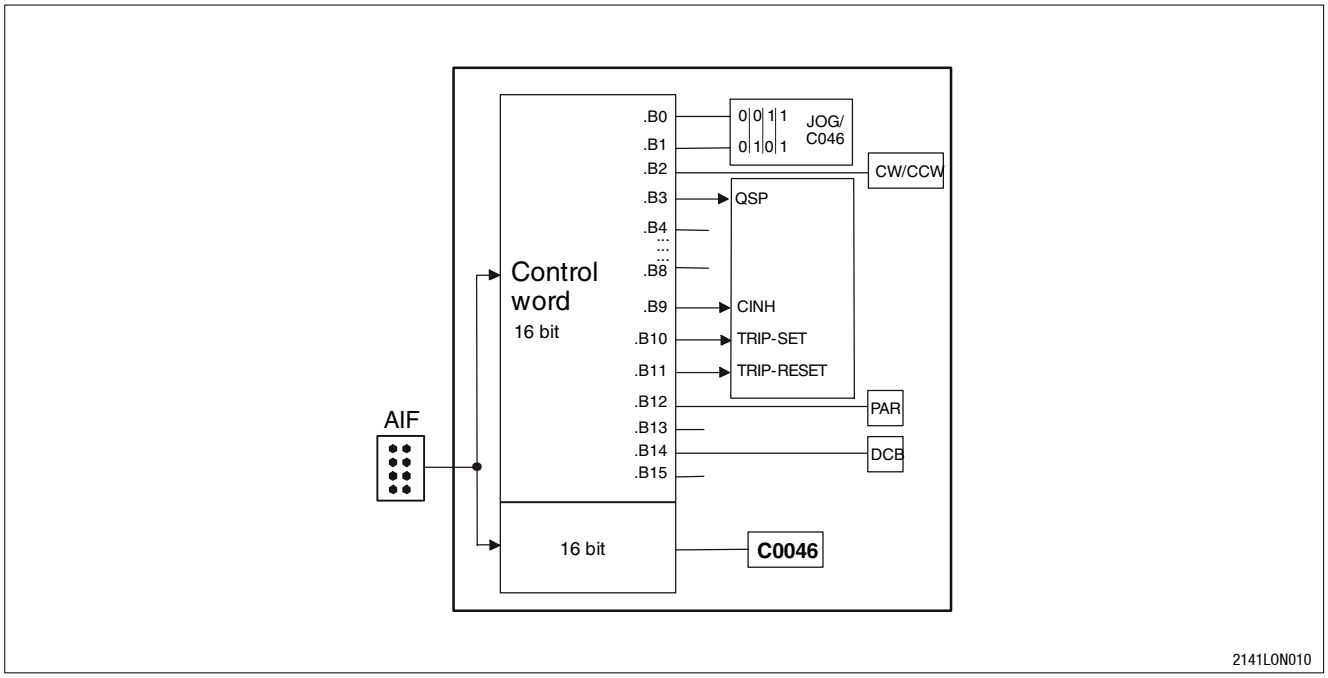


Fig. 6-3 Access to control word and frequency setpoint in 82XX (fixed, see (📖 6-20))

Process data telegram from controller

User data length: 8 byte

	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Identifier	Status word L-C0150 Low byte	Status word L-C0150 High byte	Actual value L-C0050 Low byte	Actual value L-C0050 High byte	xx	xx	xx	xx

More information about the process data telegram:	**Byte 1**	Bits 0 to 7 of the status word under L-C0150 are entered here
	Byte 2	Bits 8 to 15 of the status word under L-C0150 are entered here (seen chapter 6.3.4.2). The bit description can be obtained from the Code Table.
	Byte 3	Actual frequency value with signed normalisation (L-C0050) 24000 = 480 Hz is provided here.
	Byte 4	
	Byte 5	No evaluation of these data, anything possible
	Byte 6	
	Byte 7	
	Byte 8	

Status word: see chapter 6.3.4.2.

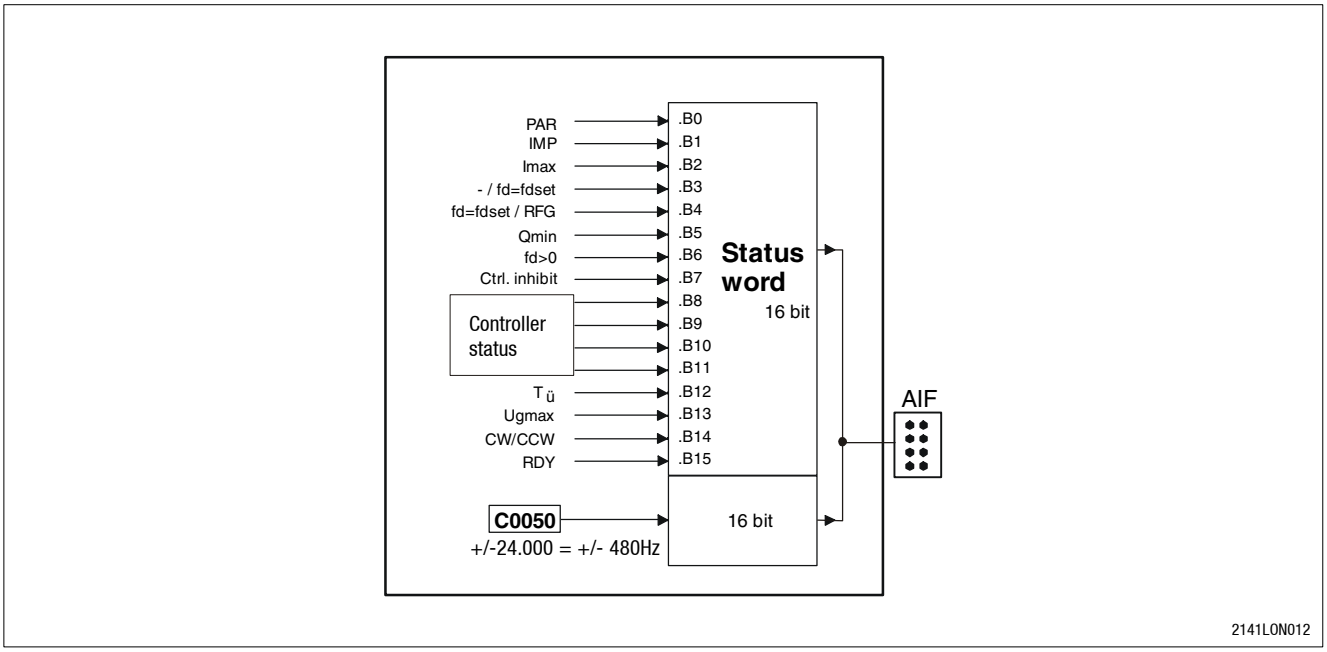


Fig. 6-4 Read access to status word and actual frequency value in 82XX (fixed, see (📖 6-21))

6.3.4 Process-data assignment for 8200 vector

A change of code L-C0001 to 3 starts the preconfiguration of process data words in the controller (see chapter 6.3.1). .

Tip!

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

Process data telegram to drive

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

Control word: see chapter 6.3.4.1.

AIF-IN.Wx is parameterised under code L-C0412.

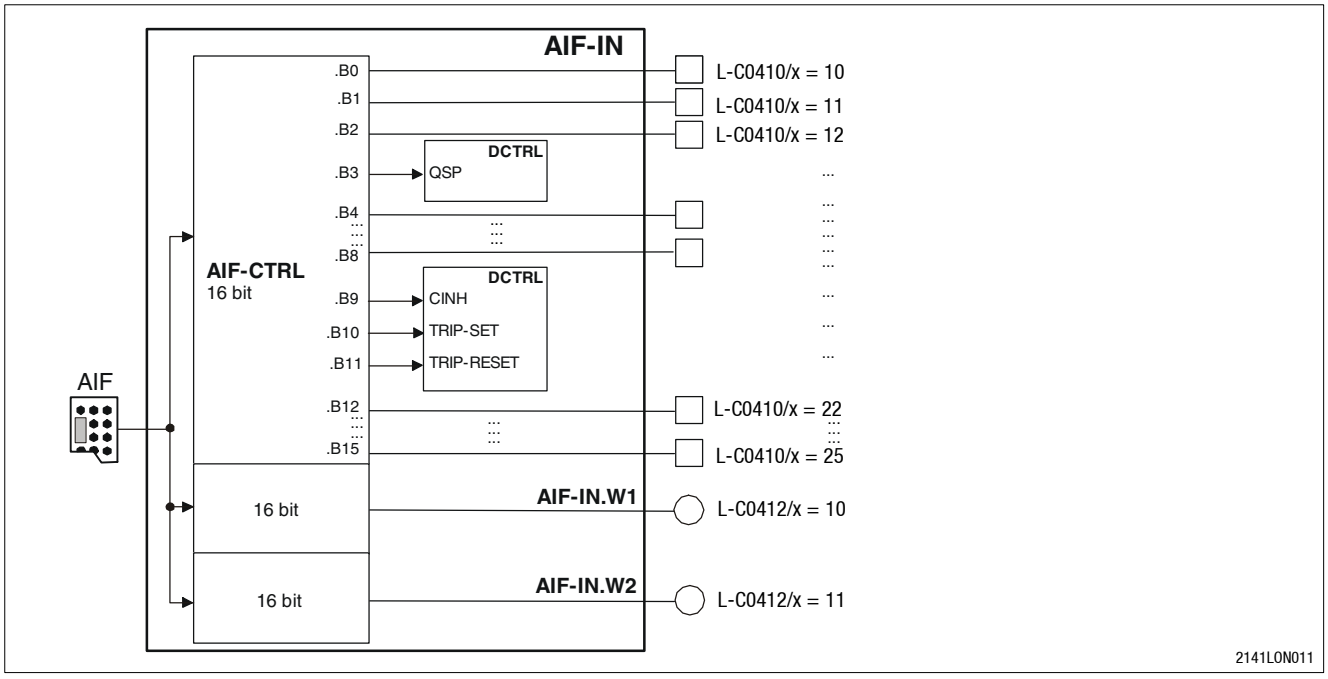


Fig. 6-5 Function block AIF-IN in 8200 vector (freely programmable assignment, for default setting see (📖 6-20))
Note:
A subcode ("x" in figure) determines the bit or word meaning (see 8200 vector Operating Instructions)

Process data telegram from drive

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

Status word: see chapter 6.3.4.2.

AIF-OUT.Wx is parameterised under code L-C0421.

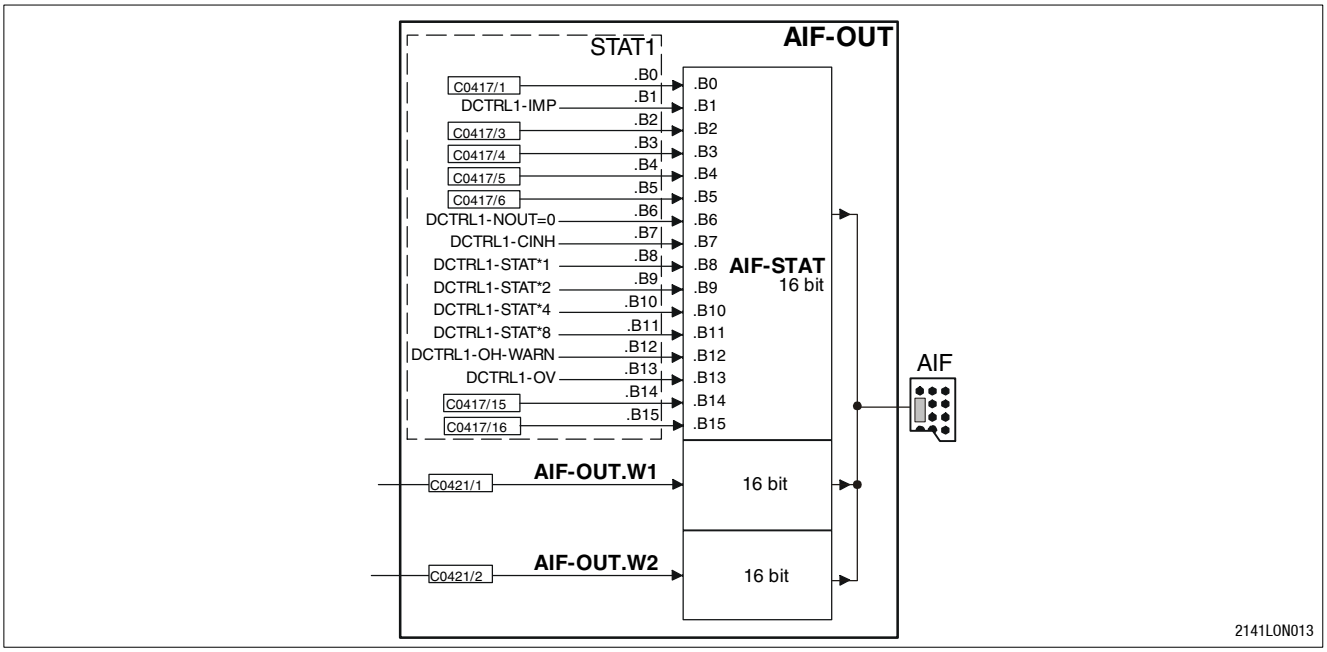
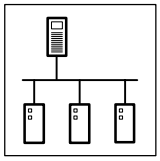
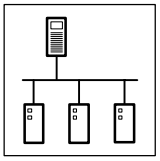


Fig. 6-6 Function block AIF-OUT in 8200 vector (freely programmable assignment, for default setting see (📖 6-21))



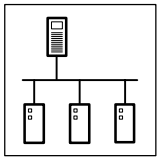
6.3.4.1 Control word for 82XX and 8200 vector

AIF-CTRL .Bxx	820X	821x,822x	8200vector	
			Default setting: C0001=3 if C0007 < 52	**Default setting: C0001=3 if C0007 > 51**
00, 01	00 = C0046 active 01 = JOG1 active in C0037 10 = JOG2 active in C0038 11 = JOG3 active in C0039	00 = C0046 active 01 = JOG1 active in C0037 10 = JOG2 active in C0038 11 = JOG3 active in C0039	00 = C0046 active 01 = NSET1-JOG1 (C0037) active 10 = NSET1-JOG2 (C0038) active 11 = NSET1-JOG3 (C0039) active	Freely configurable by user
02	CW/CCW (CW rotation/CCW rotation) 0 = CW rotation 1 = CCW rotation	CW/CCW (CW rotation/CCW rotation) 0 = CW rotation 1 = CCW rotation	DCTRL1-CW/CCW 0 = not active 1 = active	
03	QSP (quick stop) 0 = QSP not active 1 = QSP active	QSP (quick stop) 0 = QSP not active 1 = QSP active	AIF-CTRL-QSP 0 = not active 1 = active	
04	Reserved	RFG stop (stop of the ramp function generator) 0 = RFG stop not active 1 = RFG stop active	NSET1-RFG1-STOP 0 = not active 1 = active	Freely configurable by user
05	Reserved	RFG zero (deceleration along T_{if} ramp C0013) 0 = RFG zero not active 1 = RFG zero active	NSET1-RFG1-0 0 = not active 1 = active	
06	Reserved	UP function for motor potentiometer 0 = UP not active 1 = UP active	MPOT1-UP 0 = not active 1 = active	
07	Reserved	DOWN function for motor potentiometer 0 = DOWN not active 1 = DOWN active	MPOT1-DOWN 0 = not active 1 = active	
08	Reserved	Reserved	Freely configurable by user	
09	Ctrl. inhibit (controller inhibit) 0 = controller not inhibited 1 = controller inhibited	Ctrl. inhibit (controller inhibit) 0 = controller not inhibited 1 = controller inhibited	AIF-CTRL-CINH 0 = not active 1 = active	
10	Reserved	Reserved	AIF-CTRL-TRIP-SET 0 = not active 1 = active	
11	Reserved	TRIP reset 0 -> 1 = Edge from 0 to 1	AIF-CTRL-TRIP-RESET 0 -> 1 = Edge from 0 to 1	
12	PAR1 (Parameter set changeover) 0 -> 1 = Parameter set 1 -> 0 = Parameter set	PAR1 (Parameter set changeover) 0 -> 1 = Parameter set 1 -> 0 = Parameter set	DCTRL1-PAR2/4 0 = not active 1 = active	Freely configurable by user
13	Reserved	Reserved	DCTRL1-PAR3/4 0 = not active 1 = active	
14	DC brake (DC injection brake) 0 = DC brake not active 1 = DC brake active	DC brake (DC injection brake) 0 = DC brake not active 1 = DC brake active	MCTRL1-DCB 0 = not active 1 = active	
15	Reserved	Reserved	Freely configurable by user	



6.3.4.2 Status word for 82XX and 8200 vector

	820X	821x,822x	8200vector default setting
0	Actual parameter set 0 = Parameter set 1 or 3 active 1 = Parameter set 2 or 4 active	Actual parameter set 0 = Parameter set 1 or 3 active 1 = Parameter set 2 or 4 active	DCTRL-PAR-B0
1	IMP (pulse inhibit) 0 = Pulses for power stage enabled 1 = Pulses for power stage inhibited	IMP (pulse inhibit) 0 = Pulses for power stage enabled 1 = Pulses for power stage inhibited	DCTRL1-IMP
2	I_{max} (current limit reached) 0 = Current limit not reached 1 = current limit reached	I_{max} (current limit reached) 0 = Current limit not reached 1 = current limit reached	MCTRL1-IMAX
3	not assigned	$f_d = f_{dset}$ $0 = f_d \neq f_{dset}$ $1 = f_d = f_{dset}$	MCTRL1-RFG1=NOUT
4	$f_d = f_{dset}$ $0 = f_d \neq f_{dset}$ $1 = f_d = f_{dset}$	RFG on = RFG off 0 = RFG on ≠ RFG off 1 = RFG on = RFG out	NSET1-RFG1-I=0
5	Qmin ($f_d \leq f_{dQmin}$) 0 = Qmin not active 1 = Qmin active	Qmin ($f_d \leq f_{dQmin}$) 0 = Qmin not active 1 = Qmin active	PCTRL1-QMIN
6	f_d + 0 (act. frequency = 0) $0 = f_d \neq 0$ $1 = f_d$ + 0	f_d + 0 (act. frequency = 0) $0 = f_d \neq 0$ $1 = f_d$ + 0	DCTRL1-NOUT=0
7	Ctrl. inhibit (controller inhibit) 0 = controller not inhibited 1 = controller inhibited	Ctrl. inhibit (controller inhibit) 0 = controller not inhibited 1 = controller inhibited	DCTRL1-CINH
8...11	!controller status 0 = Controller initialization 8 = Error active	!controller status 0 = Controller initialization 1 = Switch on inhibit 3 = Operation inhibited 4 = Flying-restart circuit active 5 = DC brake active 6 = Operation enabled 7 = Message active 8 = Error active	!controller status 0 = Controller initialization 1 = Switch on inhibit 3 = Operation inhibited 4 = Flying-restart circuit active 5 = DC brake active 6 = Operation enabled 7 = Message active 8 = Error active
12	Overtemperature warning 0 = No warning 1 = Warning	Overtemperature warning 0 = No warning 1 = Warning	DCTRL1-OH-WARN
13	U_{Gmax} (DC-bus overvoltage) 0 = No overvoltage 1 = overvoltage	U_{Gmax} (DC-bus overvoltage) 0 = No overvoltage 1 = overvoltage	DCTRL1-OV
14	Direction of rotation 0 = CW rotation 1 = CCW rotation	Direction of rotation 0 = CW rotation 1 = CCW rotation	DCTRL1-CCW
15	Ready 0 = not ready for operation 1 = ready for operation	Ready 0 = not ready for operation 1 = ready for operation	DCTRL1-RDY



6.3.5 Process-data assignment for 93XX

With 93XX controllers, the process data assignment can be changed by reconfiguring the function blocks AIF-IN and AIF-OUT.

Process data telegram to drive

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	AIF-IN.W3 Low byte	AIF-IN.W3 High byte

Control word: see chapter 6.3.5.1.

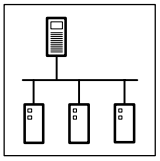
AIF-IN.W1 to AIF-IN.W3 depend on the signal configuration selected under L-C0005.

For detailed description of the 93XX signal configuration see the Operating Instructions for 93XX (only the main configurations: 1000, 4000, 5000, etc.) or the Manual for 93XX.

Other controller signals can be assigned to AIF-IN.W1 to AIF-IN.W3. For this, the function-block configuration - described in the 93XX Manual - is used. The AIF-IN function block determines the input data of the controller as data interface for the 2175 fieldbus module.

For more detailed information about the AIF-IN function block, see the 93XX Manual.

Signal configuration (L-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			
DF master	5003 / 5013 / 5113	NSET-N Speed setpoint 100 % = 16383			
DF slave bus	6003 / 6013 / 6113	DFSET-A-TRIM Phase trimming	DFSET-N-TRIM Speed trimming		
DF slave cascade	7003 / 7013 / 7113	DFSET-VP-DIV DF 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			



6.3.5.1 Control word for 93XX

9300	9300 Servo				9300 Positioning controller	9300 Cam profiler	9300 Vector		
C0005	1xx3	4xx3	5xx3	6xx3,7xx3	2xxx3	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

Tip!

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

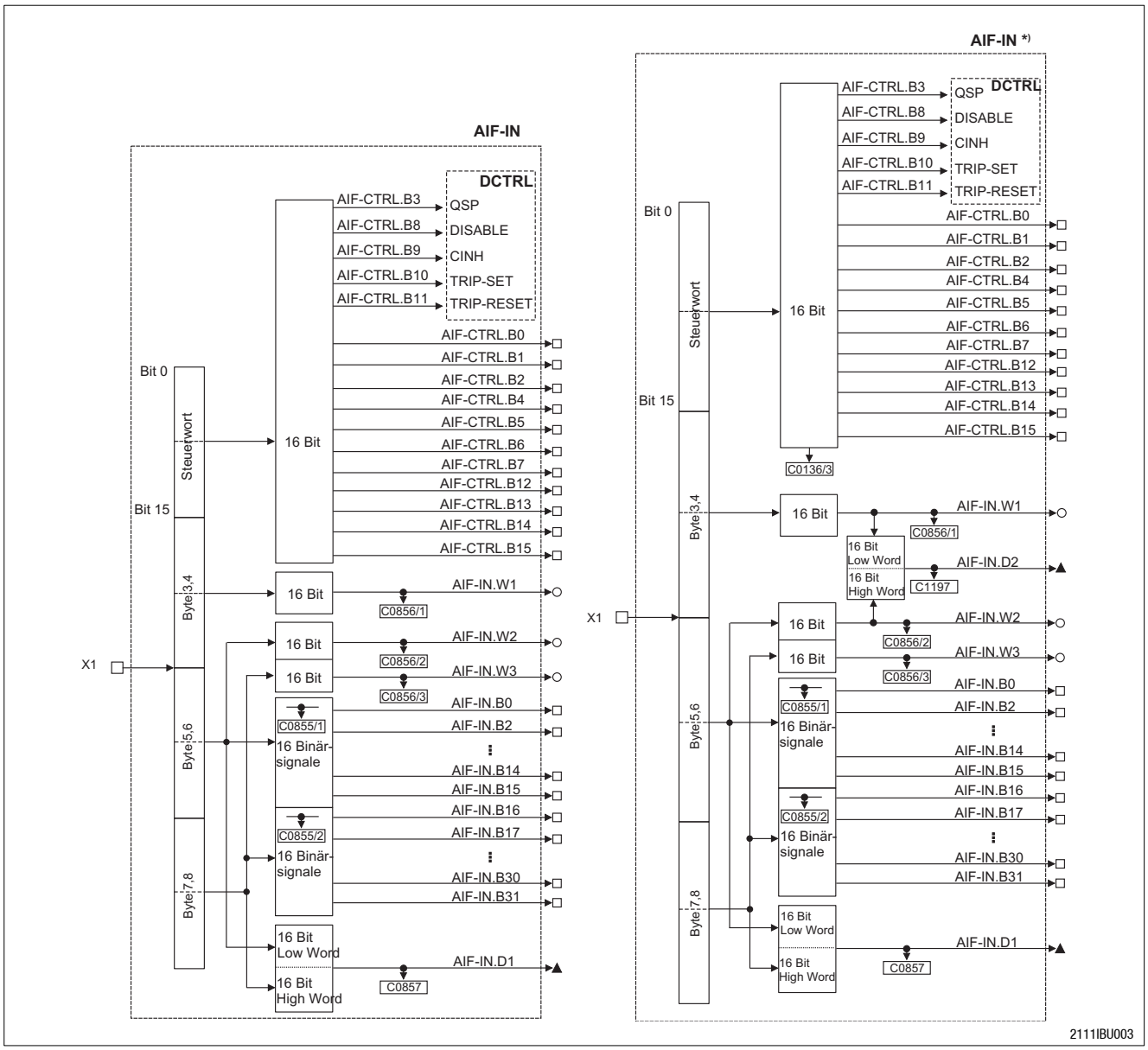
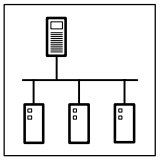


Fig. 6-7 AIF-IN and AIF-IN function blocks*)
AIF-IN*) is available for the following controllers: 9300 servo, positioning controller and cam profiler as of software version 2.0. AIF-IN.D2 is new.



Process data telegram from drive

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Status word High byte	Status word Low byte	AIF-OUT.W1 High byte	AIF-OUT.W1 Low byte	AIF-OUT.W2 High byte	AIF-OUT.W2 Low byte	AIF-OUT.W3 High byte	AIF-OUT.W3 Low byte

Status word: see chapter 6.3.5.2.

AIF-OUT.W1 to AIF-OUT.W3 depend on the signal configuration selected under L-C0005.

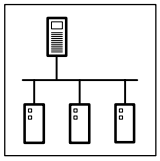
For detailed description of the 93XX signal configuration see the Operating Instructions for 93XX (only the main configurations: 1000, 4000, 5000, etc.) or the Manual for 93XX.

Other controller signals can be assigned to AIF-OUT.W1 to AIF-OUT.W3. For this, the function-block configuration - described in the 93XX Manual - is used. The AIF-OUT function block determines the controller output data as data interface for the 2175 fieldbus module.

For more detailed information about the AIF-OUT function block, see the 93XX Manual .

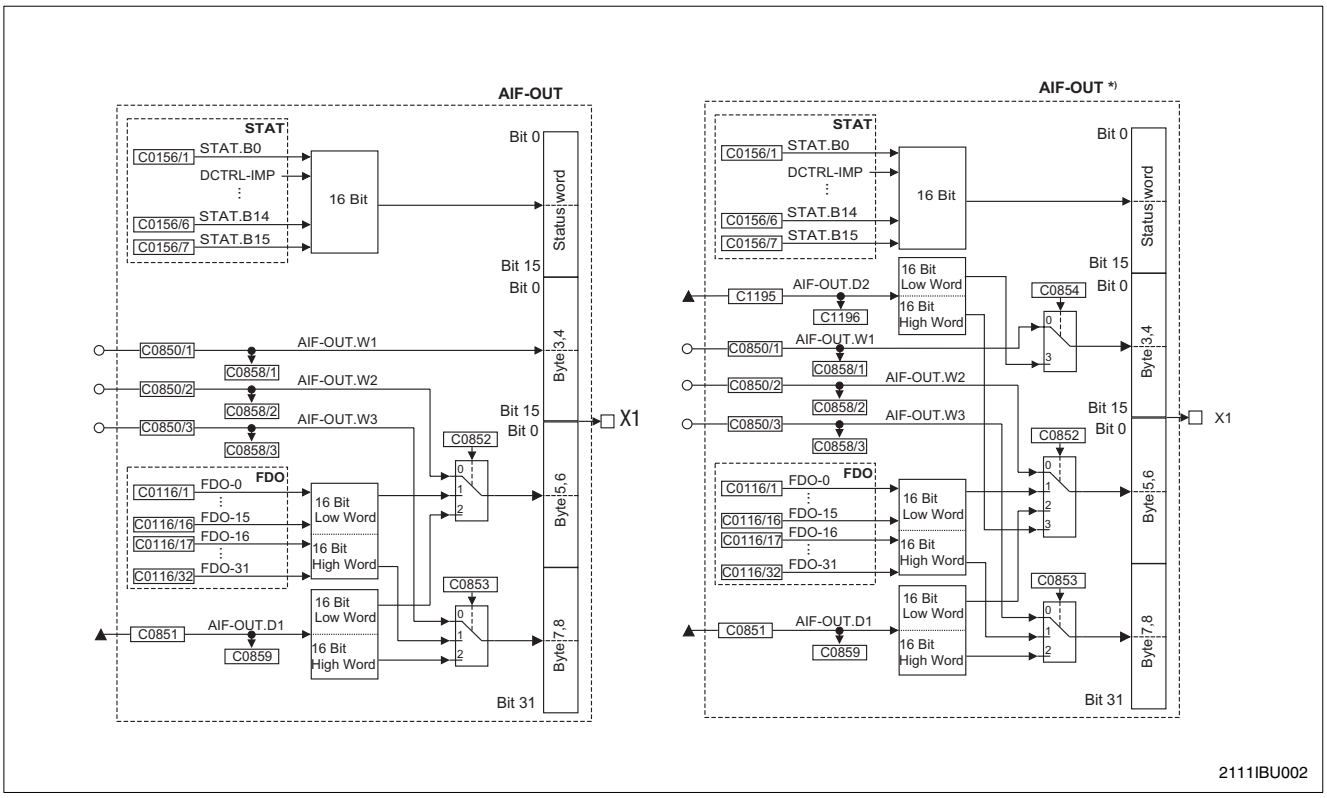
Signal configuration (L-C0005)		AIF-OUT.W1	AIF-OUT.W2	AIF-OUT.W3	AIF-OUT.D1
Speed control	1003/1013/1113	MCTRL-NACT Actual speed 100 % = 16383	MCTRL-MSET2 Torque display 100 % = 16383	MCTRL-NSET2 Speed controller input 100 % = 16383	not assigned
Torque control	4003/4013/4113	MCTRL-MSET2 Torque display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

For more detailed information about the AIF-OUT function block, see the 93XX Manual .



6.3.5.2 Status word for 93XX

9300	Servo				Servo positioning controller	Servo cam profiler	vector		
C0005	1xx3	4xx3	5xx3	6xx3,7xx3	2xxx3	1xxx3	xxx, 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	Controller status: 0 = Unit initialisation 1 = Switch-on inhibit 3 = Operation inhibited 4 = Flying-restart circuit active 5 = DC injection braking active 6 = Operation enabled 7 = Message active 8 = Fault active 10 = Fail-QSP (only 9300 servo positioning 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



2111IBU002

Fig. 6-8 AIF-OUT and AIF-OUT function blocks *)
AIF-OUT*) is available for the following controllers: 9300 servo, positioning controller and cam profiler as of software version 2.0. AIF-OUT.D2 is new.

6.4 Implemented CANopen objects

Lenze controllers can be parameterised with Lenze codes (see (📖 6-43)) 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 4.01".

All CANopen objects can also be mapped by Lenze codes. The section **"CANopen relation"** describes how a change of CANopen objects influences Lenze codes.

Tip!

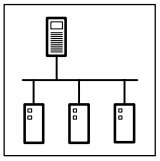
Some of the terminology used here derives from the CANopen protocol.

Abbreviations

Authorisation	
rw	Write-read authorisation
wo	Write-only authorisation
ro	Read-only authorisation
const	Read-only authorisation, constant value
Data type (see below)	
U32	Unsigned integer 4 byte (= 32 bit)
U16	Unsigned integer 2 byte (= 16 bit)
U8	Unsigned integer 1 byte (= 8 bit)
Visible string {length}	Character string with indicated {length}

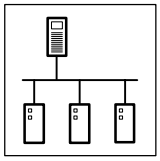
Overview

Index [hex]	**Subindex** [hex]	**Name**	**Data type**	**Authorisation**
1000	0	Device type	U32	ro
1001	0	Error register	U8	ro
1003	0	Recording numbers of error	U8	rw
	1	Standard error field	U32	r0
1005	0	Identifier sync message	U32	rw
1006	0	Communication cycle period	U32	rw
1008	0	Manufacturer device name	Visible string {9 characters}	const
100A	0	Manufacturer software version	Vis. string {11 characters}	const
100C	0	Guard time	U16	rw
100D	0	Life time factor	U8	rw
1010	0 ... 3	Store parameters	U32	ro/rw
1011	0 ... 3	Restore default parameters	U32	rw/ro
1014	0	COB-ID emergency object	U32	rw
1015	0	Inhibit time emergency	U16	rw
1018	0 ... 3	Identity object	Identity	ro
1200/1201	0 1 2	Server SDO parameter Identifier client → Server (rx) Identifier server → Client (tx)	SDO param	ro
1400		Receive PDO1 communication parameter	PDO comm.	
	0	Number of entries		ro
	1	COB-ID used by PDO		rw
	2	Transmission type		rw
1401		Receive PDO2* parameter	PDO comm.	
	0	Number of entries		ro
	1	COB-ID used by PDO		rw
	2	Transmission type		rw



Index [hex]	Subindex [hex]	Name	Data type	Authorisation
1402		Receive PDO3* parameter	PDO comm.	
	0	Number of entries		ro
	1	COB-ID used by PDO		rw
	2	Transmission type		rw
1600		Receive PDO1 mapping parameter	PDO mapping	
	0	Number of mapped objects in PDOs		ro
	1	PDO mapping 1		
	2	PDO mapping 2		
	3	PDO mapping 3		
	4	PDO mapping 4		
1601		Receive PDO2* mapping parameter	PDO mapping	
	0	Number of mapped objects in PDOs		ro
	1	PDO mapping 1		
	2	PDO mapping 2		
	3	PDO mapping 3		
	4	PDO mapping 4		
1602		Receive PDO3* mapping parameter	PDO mapping	
	0	Number of mapped objects in PDOs		ro
	1	PDO mapping 1		
	2	PDO mapping 2		
	3	PDO mapping 3		
	4	PDO mapping 4		
1800		Transmit PDO1 parameter	PDO comm.	
	0	Number of supported subindexes		ro
	1	PDO1 identifier		rw
	2	Transmission type		rw
1801		Transmit PDO2* parameter	PDO comm.	
	0	Number of supported subindexes		ro
	1	PDO2* identifier		rw
	2	Transmission type		rw
1802		Transmit PDO3* parameter	PDO comm.	
	0	Number of supported subindexes		ro
	1	PDO3* identifier		rw
	2	Transmission type		rw
1A00		Transmit PDO1 mapping parameter	PDO mapping	
	0	Number of mapped objects in PDOs		ro
	1	PDO mapping 1		
	2	PDO mapping 2		
	3	PDO mapping 3		
	4	PDO mapping 4		
1A01		Transmit PDO2* mapping parameter	PDO mapping	
	0	Number of mapped objects in PDOs		ro
	1	PDO mapping 1		
	2	PDO mapping 2		
	3	PDO mapping 3		
	4	PDO mapping 4		
1A02		Transmit PDO3* mapping parameter	PDO mapping	
	0	Number of mapped objects in PDOs		ro
	1	PDO mapping 1		
	2	PDO mapping 2		
	3	PDO mapping 3		
	4	PDO mapping 4		

*) not effective when using 82XX, 8200 vector and 93XX controllers



6.4.1 Description of implemented objects

6.4.1.1 1000_{hex}: Device type

Reading of the device type and its functionality

Index [hex]	Subindex	Name	Data type	Authorisation
1000	0	Device type	U32	ro

Bit assignment in telegram data

5th byte	6th byte	7th byte	8th byte
U32			
LSB			MSB
Controller profile number		Additional information	

6.4.1.2 1001_{hex}: Error register

Reading of the error register

Index [hex]	Subindex	Name	Data type	Authorisation
1001	0	Error register	U8	ro

Bit assignment in telegram data byte (U8)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	0	0	0	0: No TRIP 1: Controller TRIP/ error in CANopen module For error analysis see L-C0161 in corresponding Operating Instructions

6.4.1.3 1003_{hex}: Pre-defined error field

Error history

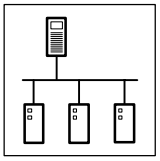
Index [hex]	Subindex	Name	Data type	Authorisation
1003	0	Recording numbers of error	U8	rw
	1	Standard error field	U32	r0

This object shows that errors occurred in the module and the basic unit.

- Subindex 0: Number of saved error messages
- Subindex 1: Error list
  Error messages (U32) consis of a 16-bis error code and a 16-bit manufacturer-specific information field.

Tip!

The “standard error field” (subindex 0) will be completely deleted if it is overwritten with “0”.



6.4.1.4 1005_{hex}: Identifier sync message

This object ensures that

- sync telegrams can be created for a module.
- the identifier value can be written.

Index [hex]	Subindex	Name	Data type	Authorisation
1005	0	Identifier sync message	U32	rw

Sync telegram creation

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

The time between to sync telegrams can be set using a different object (index 1006_{hex}, see chapter 6.4.1.5).

Identifier writing

The default setting for receiving PDOs is 80_{hex} in bit 11 (also according to CANopen). This means that all modules are default set to the same sync telegram.
If sync telegrams are only to be received by certain modules, their identifiers can be entered with values up to $7FF_{hex}$.

5th byte				6th byte				7th byte				8th byte								
U32																				
0										10	11 - 28									29	30	31
11 bit identifier	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1	X

Bit no.	Value	Explanation
0 - 10	X	Identifier (see chapter 6.1.1)
(11 - 28)*	0	*) The extended identifier (29 bit) is not supported for the 2175 fieldbus. Any of these bits must be 0.
29*	0	
30	0 1	Device does not create SYNC telegrams Device creates SYNC telegrams
31	X	Any

6.4.1.5 1006_{hex}: Communication cycle period

Setting of sync telegram cycle time

Index [hex]	Subindex	Name	Data type	Authorisation
1006	0	Communication cycle period	U32	rw

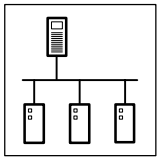
With the default setting of t = 0, sync telegrams are not created.

Cycle times can be entered as 1000 or an integer multiple of 1000. The unit of the entered time is [µs].

6.4.1.6 1008_{hex}: Manufacturer's device name

Controller and module name and code

Index [hex]	Subindex	Name	Data type	Authorisation
1008	0	Manufacturer device name	Visible string {9 characters}	const



6.4.1.7 $100A_{hex}$: Manufacturer's software version

Controller and module software version

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**
100A	0	Manufacturer software version	Vis. string {11 characters}	const

6.4.1.8 $100C_{hex}$: Guard time

Monitoring time

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**
100C	0	Guard time	U16	rw

The guard time is indicated in [ms].

If guarding is not to be supported, the default setting of 0 should not be changed.

6.4.1.9 $100D_{hex}$: Life time factor

Index [hex]	**Subindex** [hex]	**Name**	**Data type**	**Authorisation**
100D	0	Life time factor	U8	rw

If the guarding function, which results from guard time and life time factor, is not to be supported, the default setting of 0 should not be changed. For further information see chapter 6.1.2.1.

6.4.1.10 1010_{hex}: Store parameters

Parameter storage in the EEPROM.

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**
1010	0	Store parameters	U32	ro/rw
	1 ... 3*			

The subindexes 1, 2 and 3 are presently not supported.

Parameter storage

Error messages occur when

- storage is incorrect (in bytes 5 ...8): $0606\ 0000_{hex}$
- the signature is wrong: $0800\ 0020_{hex}$

Tip!

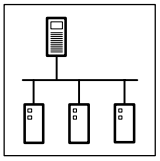
For module parameter storing, the signature "**save**" must be in the telegram data.

Assignment of telegram data words for parameter storing

Signature	MSB			LSB
ISO 8859 (ASCII)	**e**	**v**	**a**	**s**
hex	65	76	61	73

Bit assignment for reading right

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



Subindex	Rights	Explanation		
		Write	Read	
0	ro	• The following error message occurs while writing: 0601 0002	Supported subindex = 3	
1	rw	• This function is not supported yet • The following error message occurs while writing: 0800 0020	Reading of memory functions of all parameters	The following functions are possible (depending on the controller) and represented by reading the values of bit positions 0 and 1: Value 0: No saving Value 1: Saving on command Value 2: Automatic saving Value 3: Automatic saving and saving on command
2			Reading of memory functions of communication parameters (objects of the 2175 fieldbus module)	
3			Reading of memory functions only of manufacturer-specific parameters (range: 6000_{hex} - $9FFF_{hex}$)	

6.4.1.11 1011_{hex}: Restore default parameters

Download of default settings.

Tip!

With this function the subindex used depends on the controller type.

Index [hex]	Subindex	Name	Data type	Authorisation
1011	0 ... 3	Restore default parameters	U32	rw/ro

In addition to index and subindex, telgram data must include the signature“ **load** “ to start the parameter download (see table).

Signature	MSB			LSB
ISO 8859 (ASCII)	d	a	o	l
hex	64	61	6F	6C

Bit assignment for reading right

U32								
0	1 - 31							
0: Download not possible 1: Download possible	0	0	0	..	0	0	0	

Subindex	Rights	Write	Read			
0	ro	• The following error message occurs while writing: $0601\ 0002_{hex}$	Max. available subindex depends on controller type: • 7: 8200 vector frequency inverter • 5: 82XX frequency inverter • 3: 93XX servo inverter			
1	rw	• This function is not supported yet • The following error message occurs while writing: $0800\ 0020_{hex}$	All parameters can be downloaded	Function not available for		
				8200 vector	**82XX**	**93XX**
2			Download of communication parameters only (objects of the 2175 fieldbus module)	X	X	X
3			Download of manufacturer-specific parameters only (index 6000_{hex} - $9FFF_{hex}$)	X	X	X
4			Download of parameter set 1 possible	X	X	
5			Download of parameter set 2 possible	X	X	
6			Download of parameter set 3 possible	X		
7			Download of parameter set 4 possible	X		

6.4.1.12 1014_{hex}: COB-ID emergency object

When errors occur in the 2175 fieldbus module or controller (e.g. TRIP), the CAN bus sends an error message. The telegram is sent once for every fault.

Index [hex]	**Subindex**	**Name**	**Data type**	**Authrorisation**
1014	0	COB-ID emergency object	U32	rw

Data telegram assignment

5th byte								6th byte								7th byte								8th byte							
U32																															
0							10	11 - 28																						29	30	31
11 bit identifier								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reserved	

Explanation

Bit no.	**Value**	**Explanation**
0 - 10	0/1	Identifier (see chapter 6.1.1)
(11 - 28)*	0	*) The extended identifier (29 bit) is not supported for the 2175 fieldbus. Any of these bits must be 0.
29*	0	
30	0	Reserved
31	0	

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

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte
Emergency error code LENZE: Error code “10XX”		Error register Object 1001_{hex}	Field for manufacturer-specific error messages				

6.4.1.13 1015_{hex}: Inhibit time emergency

This object determines the time between an internal module error and the sending of the error message via the bus (“COB-ID emergency object”, code: 1014_{hex}).

Only integer multiples of ten are processed further. The value entered is multiplied by 100, the result is a time in [µs].

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**
1015	0	Inhibit time emergency	U16	rw

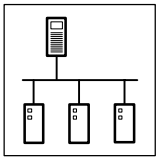
6.4.1.14 1018_{hex}: Identity object

Entry of vendor-ID

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**
1018	0 ... 3	Identity Object	Identity	ro

The identification number for this object has been determined by “Organisation CAN in Automation e. V.” and can be read out using this object:

Subindex	**Meaning**
0	Highest subindex
1	Vendor ID
2	Product code
3	Version



6.4.1.15 1200_{hex} / 1201_{hex}: Server SDO parameters

Server SDOs can be parameterised with two objects (CAN parameter data channel 1 = 1200_{hex} and CAN parameter data channel 2 = 1201_{hex}).

With index 1201 the identifier can be written in sending and receiving direction, index 1200 has only got reading access.

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**	**Explanation**
1200	0 1 2	Server SDO Parameter Identifier client → Server (rx) Identifier server → Client (tx)	SDO Param	ro	0: Max. supported subindex = 2 1: Receive identifier specification 2: Send identifier specification
1201	0 1 2	Server SDO parameter Identifier client → Server (rx) Identifier server → Client (tx)	SDO Param	ro rw rw	

Data telegram assignment

5th byte								6th byte								7th byte								8th byte							
U32																															
0							10	11 - 28																					29	30	31
11 bit identifier								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1

Explanation

Bit no.	**Value**	**Explanation**
0 - 10	0/1	Identifier (see chapter 6.1.1)
(11 - 28)*	0	*) The extended identifier (29 bit) is not supported for the 2175 fieldbus. Any of these bits must be 0.
29*	0	
30	0	Reserved
31	0/1	0: SDO valid 1: SDO invalid

6.4.1.16 1400_{hex}: Receive PDO1 communication parameter

Receipt of PDO1 communication parameters

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**	**Explanation**
1400	0	Number of entries	PDO comm.	ro	Subindex 0: Max. supported subindex = 2 Subindex 1: Identifier setting for this PDO (200_{hex} + node ID) 📖 6-2 Subindex 2: Transfer type setting (see table)
	1	COB-ID used by PDO		rw	
	2	Transmission type		rw	

Data telegram assignment

U32																					
0	10	11 - 28																	29	30	31
11 bit identifier		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1

Explanation

Bit no.	**Value**	**Meaning**
10 - 0 (LSB)	X	Contains identifier (basic + controller address)
(11 - 28)*	0	*) The extended identifier (29 bit) is not supported for the 2175 fieldbus. Every bit must become 0!
29*	0	*) The extended identifier (29 bit) is not supported for the 2175 fieldbus. Every bit must become 0!
30	0 1	RTR to this PDO possible (Lenze) RTR to this PDO not possible (cannot be set) RTR = remote transmission request
31 (MSB)	0 1	0: PDO active 1: PDO not active

PDO transmission			**Acceptance as of signal No. n**	**Explanation**
cyclic	**synchronous**	**event-controlled**		
X	X		n= 1 ... 240	By entering value n, this PDO will be accepted by every n-th SYNC.

6.4.1.17 1401_{hex}: Receive PDO2* communication parameter

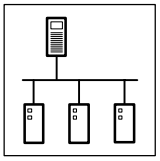
Tip!

*) Object not available for 82XX, 8200 vector and 93XX controllers.

Receipt of PDO 2 communication parameters

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**	**Explanation**
1401	0	Number of entries	PDO comm.	ro	Subindex 0: Max. supported subindex = 2 Subindex 1: Identifier setting for this PDO (200_{hex} + node ID) 📖 6-2 Subindex 2: Transfer type setting (see table)
	1	COB-ID used by PDO		rw	
	2	Transmission type		rw	

The further description of this object is the same as in chapter 6.4.1.16



6.4.1.18 1402$_{hex}$: Receive PDO3* communication parameter

Receipt of PDO 3 communication parameters

Tip!

*) Object not available for 82XX, 8200 vector and 93XX controllers.

Index [hex]	**Subindex**	**Name**	**Data type**	**Rights**	**Explanation**
1402	0	Number of entries	PDO comm.	ro	Subindex 0: Max. supported subindex = 2
	1	COB-ID used by PDO		rw	Subindex 1: Identifier setting for this PDO (200_{hex} + node ID) 📖 6-2
	2	Transmission type		rw	Subindex 2: Transfer type setting (see table)

The further description of this object is the same as in chapter 6.4.1.16

6.4.1.19 1600_{hex}: Receive PDO1 mapping parameter

With this object, parameter data can be received as PDO1.

Tip!

At present this functionality is not available. Depending on the subindex, the following values are returned when reading the object:

Subind. 0 : value 4_{hex}

Subind. 1 - 4: value 10_{hex}

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**	**Explanation**
1600	0	Number of mapped objects in PDOs	PDO mapping	ro	Subindex 0: Max. supported subindex = 4
	1	PDO mapping 1			Read request of this object are responded by the value 10_{hex} .
	2	PDO mapping 2			
	3	PDO mapping 3			
	4	PDO mapping 4			

6.4.1.20 1601_{hex}: Receive PDO2* mapping parameter

With this object, parameter data can be received as PDO2.

Tip!

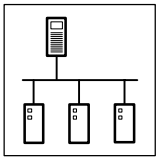
At present this functionality is not available. Depending on the subindex, the following values are returned when reading the object:

Subind. 0 : value 4_{hex}

Subind. 1 - 4: value 10_{hex}

*) Object not available for 82XX, 8200 vector and 93XX controllers.

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**	**Explanation**
1600	0	Number of mapped objects in PDOs	PDO mapping	ro	Subindex 0: Max. supported subindex = 4
	1	PDO mapping 1			Read request of this object are responded by the value 10_{hex} .
	2	PDO mapping 2			
	3	PDO mapping 3			
	4	PDO mapping 4			



6.4.1.21 1602_{hex}: Receive PDO3* mapping parameter

With this object, parameter data can be received as PDO3.

Tip!

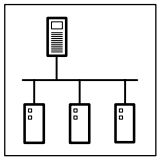
At present this functionality is not available. Depending on the subindex, the following values are returned when reading the object:

Subind. 0 : value 4_{hex}

Subind. 1 - 4: value 10_{hex}

*) Object not available for 82XX, 8200 vector and 93XX controllers.

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**	**Explanation**
1600	0	Number of mapped objects in PDOs	PDO mapping	ro	Subindex 0: Max. supported subindex = 4
	1	PDO mapping 1			Read request of this object are responded by the value 10_{hex} .
	2	PDO mapping 2			
	3	PDO mapping 3			
	4	PDO mapping 4			



6.4.1.22 1800$_{hex}$: Transmit PDO1 parameter

Sending of process data

Index [hex]	Subindex	Name	Data type	Authorisation	Explanation
1800	0	Number of supported subindexes	PDO Comm.	ro	Max. supported subindex = 2
	1	PDO identifier		rw	Identifier setting for this PDO (180$_{hex}$ + node ID)
	2	Transfer type		rw	Transfer type setting (see table)

Transfer type	PDO transfer			Explanation
	cyclic	synchronous	event-controlled	
1 - 240	X	X		With the entry of value n, this PDO is sent at every n SYNC
254			X	

6.4.1.23 1801$_{hex}$: Transmit PDO2* parameter

Sending of process data

Tip!

*) Object not available for 82XX, 8200 vector and 93XX controllers.

Index [hex]	Subindex	Name	Data type	Authorisation	Explanation
1801	0	Number of supported subindexes	PDO Comm.	ro	Max. supported subindex = 2
	1	PDO identifier		rw	Identifier setting for this PDO (280$_{hex}$ + node ID)
	2	Transfer type		rw	Transfer type setting (see table)

Transfer type	PDO transfer			Explanation
	cyclic	synchronous	event-controlled	
1 - 240	X	X		With the entry of value n, this PDO is sent at every n SYNC
254			X	

6.4.1.24 1802$_{hex}$: Transmit PDO3* parameter

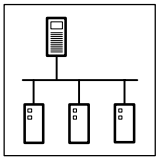
Sending of process data

Tip!

*) Object not available for 82XX, 8200 vector and 93XX controllers.

Index [hex]	Subindex	Name	Data type	Authorisation	Explanation
1802	0	Number of supported subindexes	PDO Comm.	ro	Max. supported subindex = 2
	1	PDO identifier		rw	Identifier setting for this PDO (380$_{hex}$ + node ID)
	2	Transfer type		rw	Transfer type setting (see table)

Transfer type	PDO transfer			Explanation
	cyclic	synchronous	event-controlled	
1 - 240	X	X		With the entry of value n, this PDO is sent at every n SYNC
254			X	



6.4.1.25 1A00$_{hex}$: Transmit PDO1 mapping parameter

With this object, parameter data can be sent as PDO1.

Tip!

At present this functionality is not available. Depending on the subindex, the following values are returned when reading the object:

Subind. 0 : value 4_{hex} (max. supported subindex)

Subind. 1 - 4: Read requests of this object are responded by 10_{hex} .

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**
1A00	0	Number of mapped objects in PDOs	PDO mapping	ro
	1	PDO mapping 1		
	2	PDO mapping 2		
	3	PDO mapping 3		
	4	PDO mapping 4		

6.4.1.26 1A01$_{hex}$: Transmit PDO2* mapping parameter

With this object, parameter data can be sent as PDO2.

Tip!

At present this functionality is not available. Depending on the subindex, the following values are returned when reading the object:

Subind. 0 : value 4_{hex} (max. supported subindex)

Subind. 1 - 4: Read requests of this object are responded by 10_{hex} .

*) Object not available for 82XX, 8200 vector and 93XX controllers.

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**
1A01	0	Number of mapped objects in PDOs	PDO mapping	ro
	1	PDO mapping 1		
	2	PDO mapping 2		
	3	PDO mapping 3		
	4	PDO mapping 4		

6.4.1.27 $1A02_{hex}$: Transmit PDO3* mapping parameter

With this object, parameter data can be sent as PDO3.

Tip!

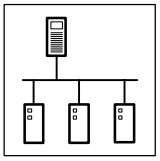
At present this functionality is not available. Depending on the subindex, the following values are returned when reading the object:

Subind. 0 : value 4_{hex} (max. supported subindex)

Subind. 1 - 4: Read requests of this object are responded by 10_{hex} .

*) Object not available for 82XX, 8200 vector and 93XX controllers.

Index [hex]	**Subindex**	**Name**	**Data type**	**Authorisation**
1A02	0	Number of mapped objects in PDOs	PDO mapping	ro
	1	PDO mapping 1		
	2	PDO mapping 2		
	3	PDO mapping 3		
	4	PDO mapping 4		



6.5 Communication-relevant Lenze codes

The behaviour of servo inverters and frequency inverters is determined by their parameters. Lenze controllers can be parameterised by codes. These (Lenze) codes are exchanged between master and 2175 fieldbus module as part of a telegram and via a CAN bus.

Depending on the Lenze inverter, the following codes are available for communication via CAN bus:

- Codes ≥ L-C2350
  For controllers with PLC operating system according to IEC1131 (e.g. 9300 servo PLC).
  These codes are saved in the controller .
- Module codes ≥ L-C1850
  For all other inverters (82XX, 8200 vector and 93XX).
  These codes are saved in the 2175 fieldbus module .

Communication with the drive is only possible when the controller is know as system device. The devices are detected while the modules are initialised.

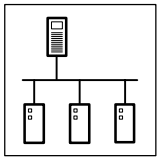
Address and baud rate can be set in different ways (see (📖 5-1)):

- Front switch 1 - 6 ≠ OFF
  Address and baud rate are determined by switch position.
  The 2175 fieldbus module writes the front switch position to codes L-C1859 and L-C2359.
- Front switch 1 - 6 = OFF
  Address and baud rate must be set under codes L-C0009 (address) and L-C0125 (baud rate).

6.5.1 Overview

How to read the table

Column	Abbreviation	Meaning
Code	L-C1853	(Lenze) code C1853
Subcode	1	Subcode 1 of code C1853
	2 .. 3	Subcodes 2 to 4 of code C1853
Index	-	Incidated as hexadecimal/decimal value
Lenze		Code default setting
Selection	1 [1 %] 99	Minimum value [smallest step/unit] maximum value
Data type	-	VS: Visible string, indicated length FIX32: 4 byte (= 32 bit) fixed value U16: 2 byte (= 16 bit) unsigned integer



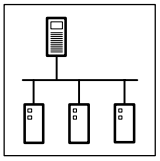
CANopen

Parameter setting

L-C18xx (for 82xx, 8200 vector and 93XX)

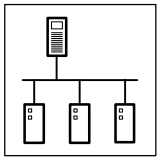
Code	Subcode	Index	Possible settings		Data type	Name
			Lenze	Selection		
L-C1810	-	$22765_d = 58ED_h$	-	-	VS	Software product code
L-C1811	-	$22764_d = 58EC_h$			VS	Software date
L-C1850	-	$58C5_h = 22725_d$	1	1 [1] 63	FIX32	Node address
L-C1851	-	$58C4_h = 22724_d$	0	0 = 500 kbit/s, 1 = 250 kbit/s 2 = 125 kbit/s, 3 = 50 kbit/s 4 = 1000 kbit/s, 5 = 20 kbit/s 6 = 10 kbit/s	FIX32	Baud rate
L-C1852	-	$58C3_h = 22723_d$	0	0 = Slave operation 1 = Master operation	FIX32	Master / slave operation
L-C1853	/1 ... /3	$58C2_h = 22722_d$	0	0 = Addressing to CANopen 1 = Addressing to L-C1854/L-C2354 2 = Addressing to LENZE system bus 3 = Addressing to CANopen index $14Xx_h/18XX_h$	FIX32	CAN-INx/CAN-OUTx addressing
L-C1854	/1 ... /2 /3* ... /6*	$58C1_h = 22721_d$	/1: 129 /2: 1 /3: 257* /4: 258* /5: 385* /6: 386*	0 [1] 1663	FIX32	CAN-IN/CAN-OUT selective addressing
L-C1855	/1 ... /2 /3* ... /6*	$58C0_h = 22720_d$	0	0 [1] 2047	FIX32	Resulting identifier display
L-C1856	/1 ... /5	$58BF_h = 22719_d$	/1: 3000 ms /2 .. /5: 0 ms	0 [1 ms] 65535	FIX32	Boot-up and cycle times
L-C1857	/1 ... /4	$58BE_h = 22718_d$	3000 ms	0 [1 ms] 65535	FIX32	Monitoring time
L-C1859	-	$58BC_h = 22716_d$	-	0 [1] 1023	U16	DIP-switch position display
L-C1860	-	$58BB_h = 22715_d$	-	0 [1] 1023	U16	Current DIP switch position display
L-C1867	-	$58B4_h = 22708_d$	128	0 [1] 2047	FIX32	Sync Rx identifier
L-C1868	-	$58B3_h = 22707_d$	128	0 [1] 2047	FIX32	Sync Tx identifier
L-C1873	/1 /2*, /3*	$58AE_h = 22702_d$	1	1 [1] 240	FIX32	CAN-IN1 ... CAN-IN3 sync rate
L-C1874	/1 /2*, /3*	$58AD_h = 22701_d$	1	0 [1] 240	FIX32	CAN-OUT1 ... CAN-OUT3 sync rate
L-C1875	/1 /2*, /3*	$58AC_h = 22700_d$	/1: 0 /2: 1* /3: 1*	0 [1] 3	FIX32	CAN-OUT1 ... CAN-OUT3 Tx mode
L-C1876	/1 ... /4	$58AB_h = 22699_d$	65535	0 [1] 65535	FIX32	CAN-OUT1 HMIs
L-C1877	/1 ... /4	$58AA_h = 22698_d$	65535	0 [1] 65535	FIX32	CAN-OUT2 masks
L-C1878	/1 ... /4	$58A9_h = 22697_d$	65535	0 [1] 65535	FIX32	CAN-OUT3 masks
L-C1882	/1 ... /5	$58A5_h = 22693_d$	0	0 [1] 2 0: No response 1: Controller inhibit 2: Quick stop	FIX32	Monitoring responses

*) not effective when using 82XX, 8200 vector and 93XX controllers



L-C23xx (for 9300 servo PLC)

Code	Subcode	Index	Possible settings		Data type	Name
			Lenze	Selection		
L-C1810	-	$22765_d = 58ED_h$	-	-	VS	Software product code
L-C1811	-	$22764_d = 58EC_h$			VS	Software date
L-C2120	-	$22455_d = 57B7_h$	0	0 = No command 1 = Code L-23XX update and CAN re-initialisation ≡ Reset node 2 = Code L-C23XX update 10 = Reading of L-C2356/1...4 11 = Reading of L-C2357 12 = Reading of L-C2375 13 = Reading of L-C2376 ... L-C2378 14 = Reading of L-C2382 15 = Not assigned	FIX32	AIF control byte
L-C2121	-	$22454_d = 57B6_h$	0	0 [1] 255	FIX32	AIF status byte
L-C2350	-	$56D1_h = 22225_d$	1	1 [1] 63	FIX32	Node address
L-C2351	-	$56D0_h = 22224_d$	0	0 = 500 kbit/s, 1 = 250 kbit/s 2 = 125 kbit/s, 3 = 50 kbit/s 4 = 1000 kbit/s, 5 = 20 kbit/s 6 = 10 kbit/s	FIX32	Baud rate
L-C2352	-	$56CF_h = 22223_d$	0	0 = Slave operation 1 = Master operation	FIX32	Master / slave operation
L-C2353	/1 ... /3	$56CE_h = 22222_d$	0	0 = Addressing to CANopen 1 = Addressing to L-C1854/L-C2354 2 = Addressing to LENZE system bus 3 = Addressing to CANopen index $14Xx_h/18XX_h$	FIX32	CAN-INx/CAN-OUTx addressing
L-C2354	/1 ... /6	$56CD_h = 22221_d$	/1: 129 /2: 1 /3: 257 /4: 258 /5: 385 /6: 386	0 [1] 1663	FIX32	CAN-IN/CAN-OUT selective addressing
L-C2355	/1 ... /6	$56CC_h = 22220_d$	-	0 [1] 2047	FIX32	Resulting identifier display
L-C2356	/1 ... /5	$56CB_h = 22219_d$	1: 3000 ms 2 .. 5: 0 ms	0 [1 ms] 65535	FIX32	Boot-up and cycle times
L-C2357	/1 ... /4	$56CA_h = 22218_d$	3000 ms	0 [1 ms] 65535	FIX32	Monitoring time
L-C2359	-	$56C8_h = 22216_d$	-	0 [1] 1023	U16	DIP-switch position display
L-C2367	-	$56C0_h = 22208_d$	128	0 [1] 2047	FIX32	Sync Rx identifier
L-C2368	-	$56BF_h = 22207_d$	128	0 [1] 2047	FIX32	Sync Tx identifier
L-C2373	/1 ... /3	$56BA_h = 22202_d$	1	1 [1] 240	FIX32	CAN-IN1 ... CAN-IN3 sync rate
L-C2374	/1 ... /3	$56B9_h = 22201_d$	1	1 [1] 240	FIX32	CAN-OUT1 ... CAN-OUT3 sync rate
L-C2375	/1 ... /3	$56B8_h = 22200_d$	/1: 0 /2: 1 /3: 1	0 [1] 3	FIX32	CAN-OUT1 ... CAN-OUT3 Tx mode
L-C2376	/1 ... /4	$56B7_h = 22199_d$	65535	0 [1] 65535	FIX32	CAN-OUT1 masks
L-C2377	/1 ... /4	$56B6_h = 22198_d$	65535	0 [1] 65535	FIX32	CAN-OUT2 masks
L-C2378	/1 ... /4	$56B5_h = 22197_d$	65535	0 [1] 65535	FIX32	CAN-OUT3 masks
L-C2382	/1 ... /5	$56B1_h = 22193_d$	0	0 [1] 2 0: No response 1: Controller inhibit 2: Quick stop	FIX32	Monitoring responses



6.5.2 Description of communication-relevant Lenze codes

6.5.2.1 L-C1810: Software product code

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
L-C1810	-	$22765_d = 58ED_h$	-	-	VS

During module initialisation the bus devices are defined by means of the product code.

6.5.2.2 L-C1811: Software date

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
L-C1811	-	$22764_d = 58EC_h$			VS

This date is mainly needed for service.

6.5.2.3 L-C1850/L-C2350: Node address

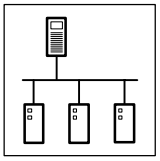
Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1850	-	$58C5_h = 22725_d$	1	1	[1]	63	FIX32
L-C2350	-	$56D1_h = 22225_d$	1	1	[1]	63	FIX32

This code can be used to set the 2175 module address via the CAN bus.

Code L-C1850 maps code L-C0009 of the basic unit. Writing to L-C1850 has a direct effect on L-C0009.

Note!

This code is only effective if the DIP switches S1-S6 have been switched OFF before mains switching.
A node address modification will only become active after power-on of the 2175 module or sending of the network management command *Reset_node* or *Reset_communication* via the CAN bus.
With 9300 servo PLCs it can also be modified by setting code C2120 (AIF control byte) to 1.



6.5.2.4 L-C1851/L-C2351: Baud rate

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
L-C1851	-	$58C4_h = 22724_d$	0	0 = 500 kbit/s 1 = 250 kbit/s 2 = 125 kbit/s 3 = 50 kbit/s 4 = 1000 kbit/s 5 = 20 kbit/s 6 = 10 kbit/s	FIX32
L-C2351	-	$56D0_h = 22224_d$			

This code can be used to set the baud rate for 2175 modules.

Note!

This code is only effective if the DIP switches S1-S6 have been switched OFF before mains switching.

A baud rate modification will only become active after power-on of the 2175 module or sending of the network management command *Reset_node* or *Reset_communication* via the CAN bus.

With 9300 servo PLCs it can also be modified by setting code C2120 (AIF control byte) to 1.

Code L-C1851 maps code L-C0125 of the basic unit, i.e. writing to L-C1851 has a direct effect on L-C0125.

6.5.2.5 L-C1852/L-C2352: Master/slave operation

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
L-C1852	-	$58C3_h = 22723_d$	0	0 = Slave operation 1 = Master operation	FIX32
L-C2352	-	$56CF_h = 22223_d$			

After power-on, the module sets PRE-OPERATIONAL status. With this status parameter data (SDOs) can only be exchanged.

In slave operation, the module remains in this status until the network management command *Start_remote_node* is put into OPERATIONAL status.

With OPERATIONAL status, parameter data (SDOs) and process data (PDOs) can be exchanged.

In master operation, the network management command *Start_remote_node* is sent after the boot-up time set under L-C1856/1 and L-C2356/1). This command puts all nodes into OPERATIONAL status.

Note:
The network management command *Start_remote_node* is a “broadcast” telegram addressing **all** other nodes.

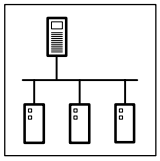
CANopen relation

Bit 30 of the CANopen index 1005_h “COB-ID SYNC message” has direct influence on this code.

Bit 30 = 1: Master operation

Bit 30 = 0: Slave operation

A modification of this code also changes the CANopen index 1005_h.



6.5.2.6 L-C1853/L-C2353: CAN-INx/CAN-OUTx addressing

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
L-C1853	/1 CAN-IN1/OUT1 /2 CAN-IN2/OUT2 /3 CAN-IN3/OUT3	$58C2_h = 22722_d$	0	0 = Addressing to CANopen 1 = Addressing to L-C1854/L-C2354 2 = Addressing to LENZE system bus 3 = Addressing to CANopen index $14Xx_h/18XX_h$	FIX32
L-C2353		$56CE_h = 22222_d$			FIX32

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

Note!

A source address modification in one or several subcodes will only become active after power-on of the module or sending of the network managment command *Reset_Node* or *Reset_Communication* via the CAN bus.

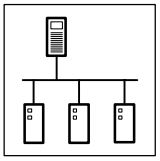
With 9300 servo PLCs it can also be modified by setting code C2120 (AIF control byte) to 1.

CANopen relation

The addressing of the corresponding PDO pair and the subcode under code L-C1853 / L-C2353 is changed to CANopen index $14XX_h/18XX_h$ (see above: value 3), if CANopen index 1400_h, 1401_h, 1402_h, 1800_h, 1801_h or 1802_h get new values.

Identifier calculation

PDO	to CANopen	via code L-1854/L-2354		via Lenze system bus	to CANopen index
			Default setting		
CAN-IN1	512 + node address	384 + C1854/1 and 384 + C2354/1	384 + 129	512 + node address	Index 1400_h, subindex 1
CAN-IN2	768 + node address	384 + C1854/3 and 384 + C2354/3	384 + 257	640 + node address	Index 1401_h, subindex 1
CAN-IN3	1024 + node address	384 + C1854/5 and 384 + C2354/5	384 + 385	768 + node address	Index 1402_h, subindex 1
CAN-OUT1	384 + node address	384 + C1854/2 and 384 + C1854/2	384 + 1	384 + node address	Index 1800_h, subindex 1
CAN-OUT2	640 + node address	384 + C1854/4 and 384 + C1854/4	384 + 258	641 + node address	Index 1801_h, subindex 1
CAN-OUT3	896 + node address	384 + C1854/6 and 384 + C1854/6	384 + 386	769 + node address	Index 1802_h, subindex 1



6.5.2.7 L-C1854/L-C2354: Selective CAN-IN/CAN-OUT addressing

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1854	/1 CAN-IN1 /2 CAN-OUT1 /3* CAN-IN2	$58C1_h = 22721_d$	/1: 129 /2: 1 /3: 257*	0	[1]	1663	FIX32
L-C2354	/4* CAN-OUT2 /5* CAN-IN3 /6* CAN-OUT3	$56CD_h = 22221_d$	/4: 258* /5: 385* /6: 386*				

*) not effective when using 82XX, 8200 vector and 93XX controllers

Input and output PDO addresses can be set individually via 6 subcodes under L-C1854 (see previous chaper).

The code becomes effective when 1 appears as subcode for code L-C1853/L-C2353 (selective addressing).

Note!

An address modification in one or several subcodes becomes effective

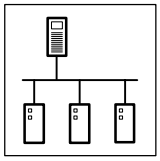
- when the module is switched on again or
- when a network management command is sent to the module via the CAN bus
  – *Reset_node* and L-C2120 = 1 or
  – *Reset_communication*

6.5.2.8 L-C1855/L-C2355: Display of resulting identifiers

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1855	/1 CAN-IN1 /2 CAN-OUT1 /3* CAN-IN2	$58C0_h = 22720_d$	-	0	[1]	2047	FIX32
L-C2355	/4* CAN-OUT2 /5* CAN-IN3 /6* CAN-OUT3	$56CC_h = 22220_d$					

*) not effective when using 82XX, 8200 vector and 93XX controllers

The resulting PDO identifier can be read from the 6 subcodes of this code.



6.5.2.9 L-C1856/L-2356: Boot-up and cycle times

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1856	/1 Boot-up time /2 CAN/OUT1 cycle times	$58BF_h = 22719_d$	1: 3000 ms 2 .. 5: 0 ms	0	[1 ms]	65535	FIX32
L-C2356	/3 CAN/OUT2 cycle times /4 CAN/OUT3 cycle times /5 Sync-Tx cycle time	$56CB_h = 22219_d$					FIX32

The cycle time which indicates when PDOs are sent via the CAN bus (see L-C1875/L-C2375, “Tx mode” for CAN-OUT1..3) can be indicated in event-controlled/cyclic operation.

Note!

A cycle time change becomes effective immediately.

0 deactivates cyclic PDO sending.

Sync-Tx cycle time (C1856/5 and C2356/5)

The “Sync-Tx cycle time” is the interval time (time basis: ms) for SYNC telegram sending.

Note!

A sync-TX cycle time change becomes effective immediately.

0 in L-C1856/5 and L-C2356/5 deactivates cyclic sync telegram sending.

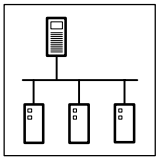
CANopen relation

The CANopen index 1006_h “communication cycle period” corresponds to the settings under codes L-C1856/5 and L-C2356/5 (Zeitbasis: µs).

Since the data processing rate of the 2175 module is 1000 µs,the input via CANopen index 1006_h is an integer multiple of 1000 µs gerundet und unter L-C1856/5 oder L-C2356/5 (abhängig vom Antrieb) gespeichert. Wird der CANopen-Index 1006_h gelesen, so wird der Inhalt dieser Codestelle in [µs] als Antwort zurückgesendet.

Furthermore, writing of code (L-C1856/5) automatically sets bit 30 of the CANopen index 1005_h (COB-ID Sync message).

With 9300 servo PLCs bit 30 of the CANopen index 1005_h must be set additionally to start cyclic sending of syncs.



6.5.2.10 L-C1857/L-C2357: Monitoring time

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1857	/1 CAN-IN1 /2 CAN-IN2 /3 CAN-IN3 /4 BUS-OFF monitoring time	$58BE_h = 22718_d$	3000 ms	0	[1 ms]	65535	FIX32
L-C2357		$56CA_h = 22218_d$					FIX32

Note!

0 deactivates the monitoring process.

A change of the monitoring time becomes effective immediately.The monitoring time starts when the first telegram arrives.

Montoring time is the time when new process input data must arrive together with CAN-IN1..3 identifiers.If the time set is exceeded, a corresponding response can be set under L-C1882.

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

A response time can be set even if the device changes into a BUS-OFF status.
Monitoring response: see codes L-C1882 and L-C2382.

6.5.2.11 L-C1859/L-C2359: Display of DIP-switch position

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1859	-	$58BC_h = 22716_d$	-	0	[1]	1023	U16
L-C2359	-	$56C8_h = 22216_d$					U16

Indicated is the DIP-switch position when initialising the module.

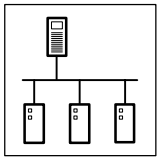
Valency table:

Switch	---	---	---	---	---	---	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
Valency							512	256	128	64	32	16	8	4	2	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Example for DIP-switch position:

- S10 = ON (CANopen communication profile)
- S4, S5 = ON (address 6)
- S7, 8, 9 = OFF (baud rate 500kBit/s)

The table above helps to calculate the sum of valencies: 61_h (97_d) indicated while reading codes L-C1859 and L-C2359.



6.5.2.12 L-C1860: Display of current DIP-switch position

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1860	-	$58BB_h = 22715_d$	-	0	[1]	1023	U16

By indicating the current DIP-switch position it is made easy to find out whether the switch positions for address, baud rate and the communication profile setting have been changed since the last initialisation. For valencies see L-C1859.

6.5.2.13 L-C1867/L-C2367: Sync Rx identifier

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1867	-	$58B4_h = 22708_d$	128	0	[1]	2047	FIX32
L-C2367	-	$56C0_h = 22208_d$					FIX32

The code contains the identifier used for receiving the telegram.

The module can send its process data objects to the CAN bus by sending sync telegrams. See L-C1875 / L-C2375.

Note!

An indentifier modification becomes effective immediately.

CANopen relation

The CANopen index 1005_h “COB-ID SYNC message” has direct influence on this code. The identifier of new value written in index 1005_hwill also be accepted by code L-C1867/L-C2367.

The value saved in L-C1868/L-C2368 will be displayed while reading the CANopen index.

6.5.2.14 L-C1868/L-C2368: Sync Tx identifier

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1868	-	$58B3_h = 22707_d$	128	0	[1]	2047	FIX32
L-C2368	-	$56BF_h = 22207_d$					FIX32

The code contains the identifier used for sending the sync telegram.

The sync is sent to the CAN bus with the identifier set under L-C1868/L-C2368 (see L-C1856 and L-C2356, subcode 5).

Note!

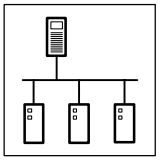
An indentifier modification becomes effective immediately.

CANopen relation

The CANopen index 1005_h “COB-ID SYNC message” has direct influence on this code. The identifier of new value written in index 1005_hwill also be accepted by code L-C1868/L-C2368. The value will be indicated when reading the index 1005_h.

6.5.2.15 L-C1873/L-C2373: Sync rate CAN-IN1 ... CAN-IN3

The input process data (CAN-INx) will only be sent to the controller when a certain number of SYNC telegrams has received.



Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1873	/1 CAN-IN1 /2* CAN-IN2 /3* CAN-IN3	$58AE_h = 22702_d$	1	1	[1]	240	FIX32
L-C2373		$56BA_h = 22202_d$					FIX32

*) not effective when using 82XX, 8200 vector and 93XX controllers

Example:

Selection n = 23. Acceptance of input PDOs (to CAN-IN1... CAN-IN3) by the controller after the 23rd sync telegram has been received.

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

Note!

Only sync telegrams with the identifier set under L-C1867 / L-C2367 count as received telegrams.

A sync rate change becomes effective immediately.

CANopen relation

The CANopen index 1400_h, 1401_h and 1402_h (receive PDO communication parameter) with subindex 2 (transmission type) are directly mapped to the subcodes of code L-C1873 / L-C2373.

Index 1400_h, subindex 2	=	Code L-C1873 and L-C2373, subcode 1
Index 1401_h, subindex 2		Code L-C1873 and L-C2373, subcode 2
Index 1402_h, subindex 2		Code L-C1873 and L-C2373, subcode 3

If, for instance, index 1402_h is read, the response value comes from code L-C1873 or L-C2373 subcode 3. If a new value is written to index 1401_h also code L-C1873 or L-C2373, subcode 2 will be overwritten with this value.

6.5.2.16 L-C1874/L-C2374: Sync rate CAN-OUT1 ... CAN-OUT3

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1874	/1 = CAN-OUT1 **/2* = CAN-OUT2** /3* = CAN-OUT3	$58AD_h = 22701_d$	1	1	[1]	240	FIX32
L-C2374		$56B9_h = 22201_d$					FIX32

*) not effective when using 82XX, 8200 vector and 93XX controllers

The output process data (CAN-OUTx) will only be sent to the controller when a certain number of SYNC telegrams has received.

The sync rate can be individually set for every output PDO.

Note!

Only sync telegrams with the identifier set under L-C1867 / L-C2367 count as received telegrams.

A sync rate change becomes effective immediately.

CANopen relation

The CANopen index “transmit PDO communication parameter” 1800_h, 1801_h and 1802_h (with subindex 2, “transmission type”) are directly written to the subcodes of codes L-C1874 and L-C2374.

Index 1800_h, subindex 2		Code L-C1874 and L-C2374, subcode 1
Index 1801_h, subindex 2	=	Code L-C1874 and L-C2374, subcode 2
Index 1802_h, subindex 2		Code L-C1874 and L-C2374, subcode 3

If, for instance, index 1802_h is read, the response value comes from code L-C1874 / L-C2374, subcode 3. If a new value is written to index 1801_h also code L-C1874, subcode 2 will be overwritten with this value.

Exception:
If value "254" is written via CANopen index 1800_h, 1801_h or 1802_h the sync rate under l-C1874 and L-C2374 of the corresponding subcode will be set zero. In addition the Tx modue under code L-C1875 / L-C2375 of the corresponding subcode will be set to 2 (event control or cyclic control). If "254" is read via index 1800_h, 1801_h or 1802_h, the settings under codes L-C1874 and L-C1875 or L-C2374 and L-C2375, subcode 1, 2 or 3 apply.

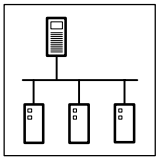
6.5.2.17 L-C1875/L-C2375: Tx mode CAN-OUT1 ... CAN-OUT3

This code contains a selection which indicates when CAN-OUT1 .. CAN-OUT3 PDO are to be sent. The selection can be made for every output PDO by subdivision into subcodes.

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1875	/1 CAN-OUT1 /2* CAN-OUT2 /3* CAN-OUT3	$58AC_h = 22700_d$	/1: 0 /2: 1 /3: 1	0	[1]	3	FIX32
L-C2375		$56B8_h = 22200_d$	/1: 0 /2: 0 /3: 0				FIX32

*) not effective when using 82XX, 8200 vector and 93XX controllers

- Value = 0
  - – Output PDOs are sent when a sync telegram is received.
  - – Only sync telegrams with the identifier set under L-C1867 / L-C2367 count as received telegrams. L-C1874 / L-C2374 can also be used for setting a number of sync telegrams (n = 1 ... 240) after which output PDO are to be sent.
- Value = 1
  - – Output PDOs are not sent.
  - – This selection deactivates the sending of CAN-OUT1..CAN-OUT3.
    This deactivation should be used for 82XX, 8200 vector and 9300 controllers which can exchange a max. of 4 words of control and status information via the AIF. Use just one input and output PDO because every PDO contains 4 information words (default setting).
    Thus unnecessary CAN load can be avoided.
- Value = 2
  - – The output PDO is either event-controlled or cyclically sent if this value is entered in one of the three available subcodes.
  - – The output PDO will be cyclically sent if a cycle time is input for the same CAN-OUT1..3 under code L-C1856 or L-C2356 (depending on the drive). If the cycle time is zero, it will be sent when an event occurs, i.e. bit change in CAN-OUT object.
- Value = 3
  - – The output PDO is sent either event-controlled or cyclically , which means that the object is sent according to the time set under L-C1856 and L-C2356.
  - – The object is also sent when one or several bits in the CAN-OUT object change.



Note!

A Tx mode modification becomes active immediately.

With event-controlled sending (even with cyclic overlay), the objects can be masked using codes L-C1876 to L-C1878 or with 9300 servo PLC certain bits can be skipped using codes L-C2376 to L-C2378 and achieve that the CAN-OUT object will **not** be sent when a bit is changed.

6.5.2.18 L-C1876/L-C2376: CAN-OUT1 masks

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

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1876	/1 CAN-OUT1.W1 /2 CAN-OUT1.W2	$58AB_h = 22699_d$	65535	0	[1]	65535	FIX32
L-C2376	/3 CAN-OUT1.W3 /4 CAN-OUT1.W4	$56B7_h = 22199_d$					FIX32

Event-controlled sending of a CAN-OUT object can depend on a single bit, if necessary. See code L-C1875 / L-C2375.

Example:

The mask in word 3 of the CAN-OUT 1 process data object is set to 20 hex under code L-C1876/3 (see “MASK“). Please observe the field highlighted in grey.

1st cycle

Result after 1st cycle: PDO is sent

	CAN-OUT 1.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

New data is written to CAN-OUT1 after the 2nd cycle.

Result after 2nd cycle: PDO not sent because of bit change

	CAN-OUT 1.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**

Note!

A mask change for the CAN-OUT1 output PDO becomes active immediately.

6.5.2.19 L-C1877/L-C2377: CAN-OUT2 masks

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

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C1877	/1 CAN-OUT2.W1 /2 CAN-OUT2.W2	$58AA_h = 22698_d$	65535	0	[1]	65535	FIX32
L-C2377	/3 CAN-OUT2.W3 /4 CAN-OUT2.W4	$56B6_h = 22198_d$					

See L-C1876 / L-C2376.

6.5.2.20 L-C1878/L-C2378: CAN-OUT3 masks

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

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
L-C1878	/1 CAN-OUT3.W1 /2 CAN-OUT3.W2	$58A9_h = 22697_d$	65535	0 [1] 65535	FIX32
L-C2378	/3 CAN-OUT3.W3 /4 CAN-OUT3.W4	$56B5_h = 22197_d$			

See L-C1876 / L-C2376.

6.5.2.21 L-C1882/L-C2382: Monitoring response

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
L-C1882	/1 CAN-IN1 response /2 CAN-IN2 response /3 CAN-IN3 response	$58A5_h = 22693_d$	0	0 [1] 2 0: No response 1: Controller inhibit 2: Quick stop	FIX32
L-C2382	/4 BUS-OFF response /5 Life guarding event response	$56B1_h = 22193_d$			FIX32

Setting of the response after the monitoring time (see L-C1857/L-C2357) has been exceeded.

Note!

A monitoring response change becomes effective immediately.

CANopen relation

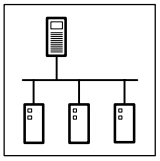
The CANopen indexes $100C_h$ “guard time” and $100D_h$ “life time factor” can be used to set a time for the node guarding p. The node guarding protocol has been developed for monitoring the master/slave connection. A time in milliseconds can be entered in CANopen index $100C_h$“guard time”. Index $100D_h$ “life time factor” saves a factor. The monitoring time (time in which the master sends a certain telegram to the slave/2175 IB) results from multiplying the two indexes. If one of the indexes is set to zero, the monitoring time is zero too and thus deactivated. The slave sends a telegram with its current NMT status to the master. The status can be preoperational, operational or stopped.

If the monitoring time is exceeded, the slave responds with a life guarding event and the master with a node guarding event.

The node guarding protocol must be programmed and started in the master. The 2175IB module supports the node guarding protocol, a response can only be entered under this code.

6.5.2.22 L-C2120: AIF control byte

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
L-C2120	-	$22455_d = 57B7_h$	0	0 = No command 1 = Code L-23XX update and CAN re-initialisation ≡ Reset node 2 = Code L-C23XX update 10 = Reading of L-C2356/1...4 11 = Reading of L-C2357 12 = Reading of L-C2375 13 = Reading of L-C2376 ... L-C2378 14 = Reading of L-C2382 15 = Not assigned	FIX32



The AIF control byte is used to read the codes saved in a 9300 servo PLC by the 2175 fieldbus module. This process can be started by writing a value which is indicated in the table in the AIF control byte.

6.5.2.23 L-C2121: AIF status byte

Code	Subcode	Index	Possible settings				Data type
			Lenze	Selection			
L-C2121	-	$22454_d = 57B6_h$	0	0	1	255	FIX32

The AIF status byte provides information from the 2175 fieldbus module for the 9300 servo PLC. The status of the 2175 fieldbus module can be monitored from the 9300 servo PLC by reading the status byte. The user can take measures against this process.

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, module in BUS-OFF status
Bit 4	Operational status
Bit 5	Preoperational status
Bit 6	Warning status
Bit 7	Internally assigned

Note!

Writing of value 2 to the AIF control byte however re-reads all codes L-C23XX, but L-C2350 ... L-C2355 need a new CAN initialisation to activate new value and their functions.

6.6 Notes to be observed for paramater setting

6.6.1 8200 controllers

The following applies to the 8200 inverter series:

Caution!

Parameter setting (codes except C046, C0135) is only possible when the controller is inhibited. Parameters are accepted when the controller is enabled, but they are not saved.
After having set a parameter, the controller must not be addressed for approx. 50 ms; otherwise the command will be ignored.

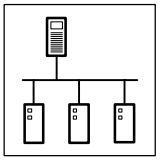
After parameter setting, the controller needs up to approx. 70 ms to set the status 'enabled' (terminal, C040, C0135).

The TRIP reset function is activated by inhibiting the controller and enabling it again under C040 or C0135.

The TRIP reset function initializes the 8200 inverter and the 2175 fieldbus module. Therefore the TRIP reset command is not acknowledged for the master.

6.6.2 8200 vector controllers

Digital and analog input and output signals can be freely configured (see Operating Instructions; codes C0410, C0412, C0417 and C0421)



6.6.3 9300 servo PLC

AIF control / status byte

Controller and fieldbus module can exchange control and status information. For this, the AIF protocol provides so-called control bytes and status bytes.

Thanks to “Drive PLC Developer Studio” (DDS) the user can directly access the status byte via the control configuration of the AIF module. The 2175 fieldbus module writes its error messages to this byte.

The PLC programmer must ensure that these error messages are indicate by means of an error-warning-message firmware module.

Assignment of the C2121 AIF status byte for the 2175 fieldbus module

AIF status bit	Function
0	CE11 error
1	CE12 error
2	CE13 error
3	CE14 error
4	Operational
5	Pre-operational
6	Warning
7	Internally assigned

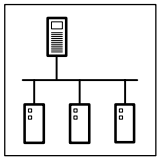
The control byte is used to send messages and commands from the controller to the 2175 fieldbus module.

The control byte can be accessed via code C2120. The commands are defined as numbers. Some of the command numbers apply to all fieldbus modules, others only apply to certain modules. Altogether up to 16 commands can be used.

Order number in C2120, bit0..3	Function	R/W code
0	No command	
1	New initialisation of optional module	all
2	Update of all relevant codes for the current optional module. No new init.	all
10	XCAN-OUT cycle times or boot-up time accepted	C2356
11	Monitoring times accepted	C2357
12	TX mode accepted	C2375
13	HMIs accepted	C2376-C2378
14	CE error configuration accepted	C2382

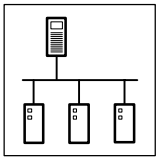
AIF control byte assignment for the 2175 fieldbus module

C2120 is automatically reset to 0 after the command has been transferred to the optional module, i.e. toggle bits are not required. The command itself is written to the 4 last bits of the control byte, i.e. 4 bits are available for future entries.



CANopen

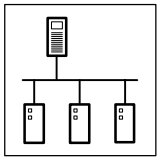
Parameter setting



7 Troubleshooting and fault elimination

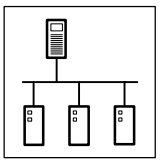
7.1 No communication with the controller

Possible causes	Diagnostics	Remedy
Is the controller switched on?	The operation status LED of the basic unit must be on 📖 4-1 Point 3.	Supply controller with voltage (see Operating Instructions for the basic unit)
Is the fieldbus module supplied with voltage?	The green LED for “Controller connection status” on the fieldbus module 📖 4-1 must be on (Remedy 1) or blinking (Remedy 2)	With supply from the basic unit check the connection. With external supply check the 24 V voltage at terminals 39 and 59. A voltage between 24 V +10 % must be applied.
		The fieldbus module has not been initialised with the controller yet. Possibility 1: Controller not switched on (see fault possibility 1). Possibility 2: Check the connection to the controller
Does the controller receive telegrams	The LED “Bus connection status“ on the fieldbus 📖 4-1 must be blinking in green when the device is communicating with the host.	Check whether the connection corresponds to the instructions given in chapter ”CAN bus wiring“, page. 📖 4-1 . Check whether host sends telegrams and uses the appropriate interface.
		CAN controller address (L-C0009 / DIP switch) and baud rate (L-C0125 / DIP switch) can be set differently for controller and host. Ensure that the addresses are identical.
		The controller address (L-C0009) must be different for all connected devices. Check whether some addresses are used twice and correct them if necessary. Check the wiring to your host.

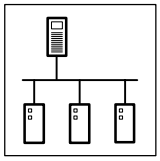


CANopen

Troubleshooting and fault elimination

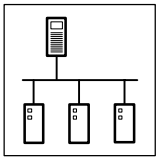


8 Table of keywords



CANopen

Table of keywords



9 Technical data

9.1 General information

The internationally standardized CAN bus protocol, which had been developed for the European Automobile Industry, is mainly characterized by:

- its resistance against interference and extreme temperatures
- short transfer times
- low expenditure for connection

These advantages have made CAN products interesting for other industries too.

The DeviceNet communication profile is based on CAN technology.

A DeviceNet network enables communication between control systems and industry components such as sensors (e.g. initiators) and actuators (e.g. electromagnetically actuated pneumatic valves), frequency inverters and servos.

The units are directly connected to the network. This means reduced costs for installation, commissioning, maintenance, tests and upgrading of the system.

The 2175 fieldbus module with the communication profile DeviceNet is a 'ONLY-SERVER' module of group 2.

9.2 Features

- Attachable module for the following Lenze controller series: 820X, 821X, 82X, 8200 vector, 93XX and 9300 servo PLC.
- The front DIP switch enables easy setting of
  – Communication profile DeviceNet or CANopen
  – Baud rates of 125, 250 and 500 kbit/s
  – Node address (max. 63 participants)
- Bus extension up to max. 500m
- Topology: Line terminated at both ends (R = 120 Ohm)
- Easy connection because of pluggable screw terminals

9.3 General data and application conditions

For	Values
Order name	33.2175IB
Communication media	DIN ISO 11898
Baud rate [kBit/s]	125, 250, 500
Ambient temperature	during operation: -20 °C to 60 °C during transport: -25 °C to 70 °C during storage -25 °C to 60 °C
Permissible humidity	Class 3K3 to EN 50178 (without condensation, average relative humidity 85%)
24-V-DC-Voltage supply	• external supply only: 820X • internal or external supply: 93XX / 9300 servo PLC / 821X / 822X / 8200 vector (see chapter 4.3.2)

9.4 Rated data

For	Values		
Voltage supply	24 V DC ± 10 %; max. 100 mA		
Communication medium	ISO 11898		
Insulation voltages for bus systems		**Rated insulation voltage**	**Type of insulation**
	• PE	50VAC	Electrical isolation
	• External supply (term. 39/59)	-	No electrical isolation
	• Power stage		
	– 820X / 821X	270 V AC	Basic insulation
	– 822X / 8200 vector	270 V AC	Double insulation
	– 93XX	270 V AC	Double insulation
	• Control terminals:		
	– 820X / 8200 vector (with internal supply)	-	No electrical isolation
	– 8200 vector (with external supply)	100 V AC	Basic insulation
	– 821X	50 V AC	Electrical isolation
	– 822X	270 V AC	Basic insulation
	– 93XX	270 V AC	Basic insulation
	• External bus systems	0 V AC	No electrical isolation
Pollution degree	VDE0110, part 2, pollution degree 2		

9.5 Dimensions

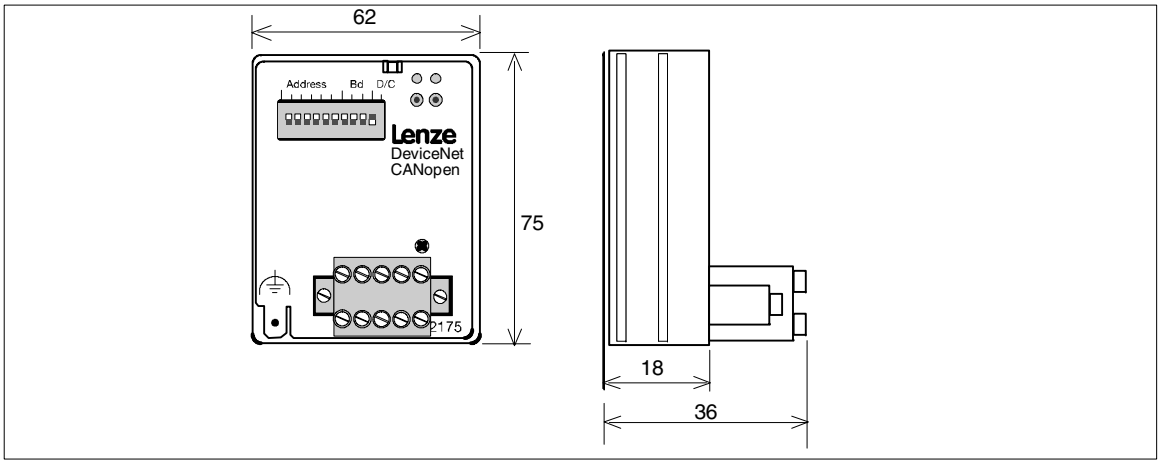
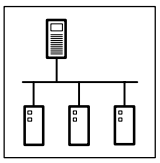


Fig. 9-1 Dimensions: 2175 fieldbus module (all dimensions in mm)



9.5.1 Communication times

Note!

The CAN bus communication times depend on the following:

- Processing time in the controller
- Baud rate
- Data priority
- Bus load

More information about bus access control can be obtained from corresponding literature specialised on Controller Area Networks.

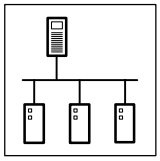
9.5.1.1 Processing times in the controller

Processing times for 820X controllers

The processing times for 8200 controllers differ from the times for the 821X/822X/8200 vector series.

In opposite to the 821X/822X/824X series, which have parallel process data processing, the 8200 series process process and parameter data sequentially. Therefore the time needed to respond process data depends on previous actions.

The processing time needed for telegrams also depends on the actual value conditioning (process data from controller). If these data (status word, actual frequency) are not required, they can be deactivated with the control word ”Bit 15” (PE inhibit).



DeviceNet

Technical data

The individual telegram times are:

Telegram	Processing time	
	PE-inhibit = 0	PE-inhibit = 1
Parameters	62...140 ms	62...70 ms
Change of a process data value to controller (*)	27...105 ms	27...35 ms
Change of both process data values to controller *	62...140 ms	4...70 ms
Process data from controller *	108...140 ms	not possible

* The processing times for process data refer to the sync telegram (chapter 6.3 (📖 6-13))

Processing times for 821X/8200 vector/822X controllers:

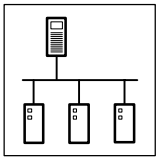
Parameters 30...50 ms

Process data, 3...5 ms (*)

Processing time for 93XX

Parameter data and process data are independent of each other.

- Parameter data: approx. 30 ms + 20 ms tolerance (typically)
  – Some codes require longer processing times (see the 9300 Manual).
- Process data: approx. 3 ms + 2 ms tolerance



9.5.1.2 Telegram time

Telegram times depend on baud rate and telegram length:

Baud rate [kbit/s]	**Data length** [byte]		
	0	**2**	**8**
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

Tab. 9-1 Maximum telegram time in [ms]

The telegram times indicated in the table above are calculated according to the following equation. This equation allows to calculate any intermediate value t_{Tmax} if necessary.

$$t_T \leq \frac{54.4 + 9.6 \cdot L_D}{d_{\ddot{U}}}$$

t_T = telegram time [ms]
L_D = telegram length [byte]
$d_{\ddot{U}}$ = baud rate [kbit/s]

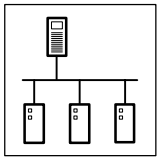
9.5.1.3 Maximum bus length

Depending on the baud rate and the cable used the following bus lengths are possible:

Baud rate [Kbit/s]	**Thin cable**	**Thick cable**
125	100 m	500 m
250		250 m
500		100 m

When using both, thick and thin cable types (see (📖 10-6) and (📖 10-7)) the maximum cable lengths are to be selected according to the baud rate:

- 125 kbit/s: $L_{max} = L_{thick} + 5 \cdot L_{thin}$
- 250 kbit/s: $L_{max} = L_{thick} + 2.5 \cdot L_{thin}$
- 500 kbit/s: $L_{max} = L_{thick} + L_{thin}$

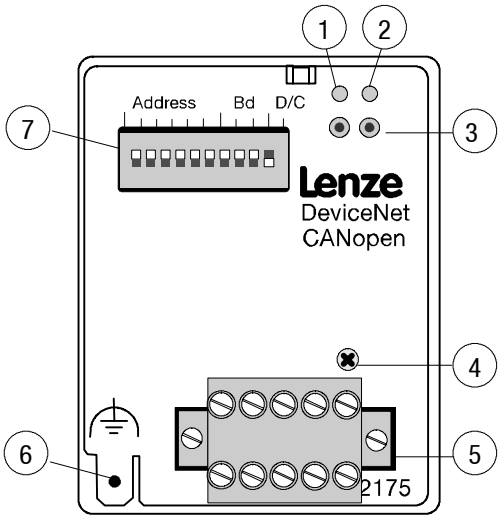


DeviceNet

Technical data

10 Installation

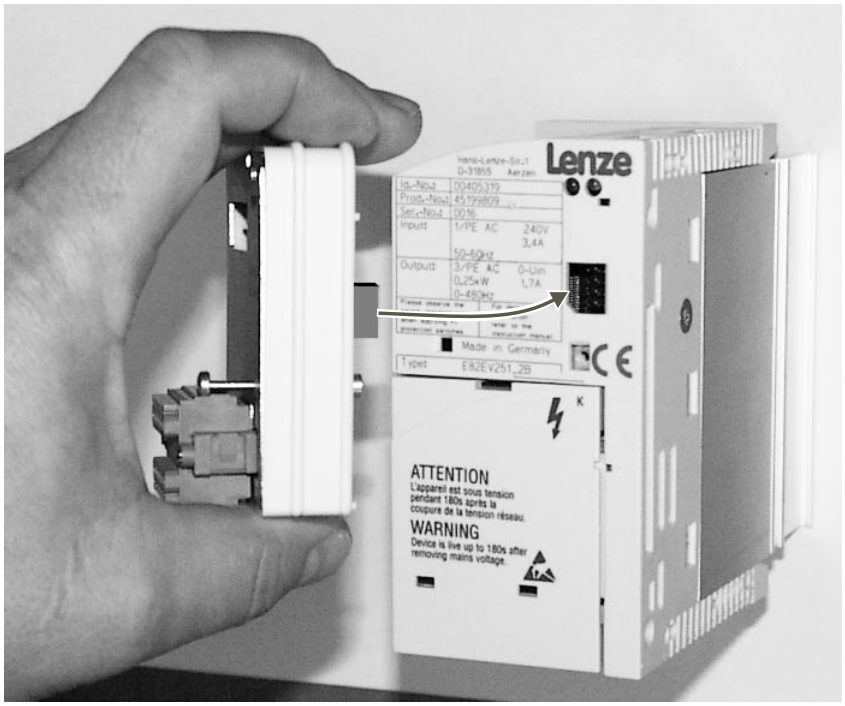
10.1 Components of the fieldbus module



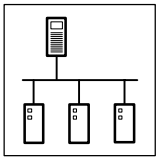
Pos.	Designation	Meaning		Notes
1	**Status of the controller connection** (two-colour LED)	OFF		2175 fieldbus module is not supplied with voltage; controller or external voltage supply is switched off.
		GREEN	BLINKING	2175 fieldbus module is supplied with voltage but is not connected to the controller (controller is switched off, in initialization or not available).
			Constantly ON	2175 fieldbus module is supplied with voltage and is connected to the controller.
2	**Connection status** (two-colour LED)	OFF		• No connection with the master • Fieldbus module is not supplied with voltage
		GREEN	BLINKING	Dup_Mac_ID testing. Still no connection to the master.
		GREEN	ON	DeviceNet connection built up.
		RED	BLINKING	No communication because time limit exceeded
		RED	ON	Internal fault of the fieldbus module
3	**Green and red drive LEDs (drive)**	Operating status of the following controllers: 82XX, 8200 vector, 93XX and servo PLC 9300 (see Operating Instructions for the controller)		
4	**Fixing screw**			
5	**5-pole plug-in connector** (see chapter 10.3.2)			
6	**Connection PE shield cable**			Only for 820X and 821X: If necessary use an additional PE shield cable which avoids EMC-related communication interference in especially noisy environments.
7	**DIP switch**	For settings see chapter 11		

10.2 Mechanical installation

- Plug the fieldbus module onto the controller (here: 8200 vector)



- Tighten the fixing screw (📖 10-1) Pos. 4)



10.3 Electrical installation

10.3.1 Selection of the transmission cable

The devices are connected to the bus by means of a fieldbus cable which must comply with the DeviceNet™ specification (release 2.0). Companies like Belden Wire & Cable, Olflex Wire & Cable, C&M Corp. and Madison Cable manufacture DeviceNet™ “thick”- and “thin” cables.

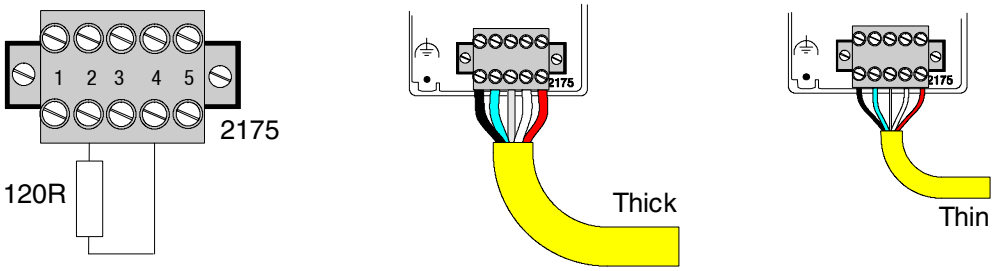
Stop!

If you do not want to use “thick” or “thin” cables, the cables used must comply with the requirements of the DeviceNet Specification (see chapter 10.410.5).
Any other cable is not permitted and must not be used.

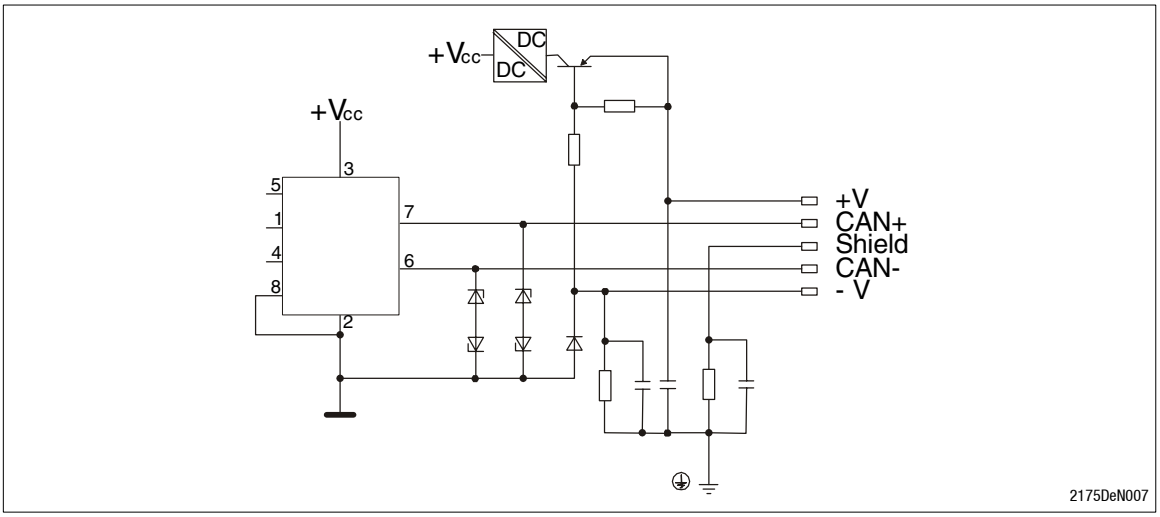
10.3.2 Assignment of the plug/socket connector

The 2175 fieldbus module is connected to the bus through a 5 pole plug/socket connector.

The assignment of the plug/socket connector and the cable colour used according to the DeviceNet Specification is listed in the table.



Terminal	Designation	Cable colour	Explanation
1	V-	black	GND; reference for external voltage supply
2	CAN_L	blue	Data cable / input for terminating resistance of 120 Ohm
3	SHIELD		Shield
4	CAN_H	white	Data cable / input for terminating resistance of 120 Ohm
5	V+	red	External voltage supply; see notes in chapter 10.3.3



Tab. 10-1 Terminal assignment

10.3.3 Voltage supply

External supply of the fieldbus module

Stop!

The Specifications for the communication profile DeviceNet describe the voltage supply in detail. These values must be observed!
The corresponding Specifications can be obtained from, for instance, Allen-Bradley.

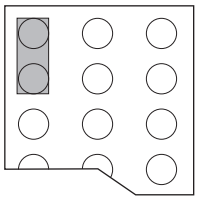
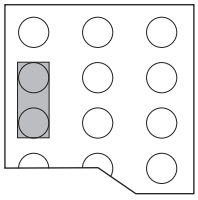
These Operating Instructions do not inform about technical data.

Note!

Internal voltage supply of the fieldbus module connected to a 8200 vector
(only applies to controllers as of nameplate data 82EV 1x 1x)

Controllers with an extended AIF interface (front of the 8200 vector) can be internally supplied. The part of the drawing highlighted in grey shows the jumper position.

With Lenze setting, the fieldbus module is not internally supplied.
For internal voltage supply, put the jumper in the position indicated below.

Lenze setting (only external voltage supply)	**Internal voltage supply**
	

10.3.4 Wiring to a host

Warning!

An additional electrical isolation is required if

- a 820X or 821X controller is connected to a host and
- a safe electrical isolation (double basic insulation) to VDE 0160 is required.

For this, you can use an interface module for the host with an additional electrical isolation (see the corresponding manufacturer's information).
For wiring, the electrical isolation of the supply voltage must be taken into account. The supply voltage is assigned to the same potential as the data bus.

A CAN line can have max. 63 devices. Devices are

- connected controllers
- masters
- all components taking part in the communication

The following diagram shows a network:
By means of this network 8200 vector controllers with attached 2175 fieldbus modules [A] can communicate with the DeviceNet master.

For easy configuration of the fieldbus modules a PC with “DeviceNet Manager” software [B] can be used.

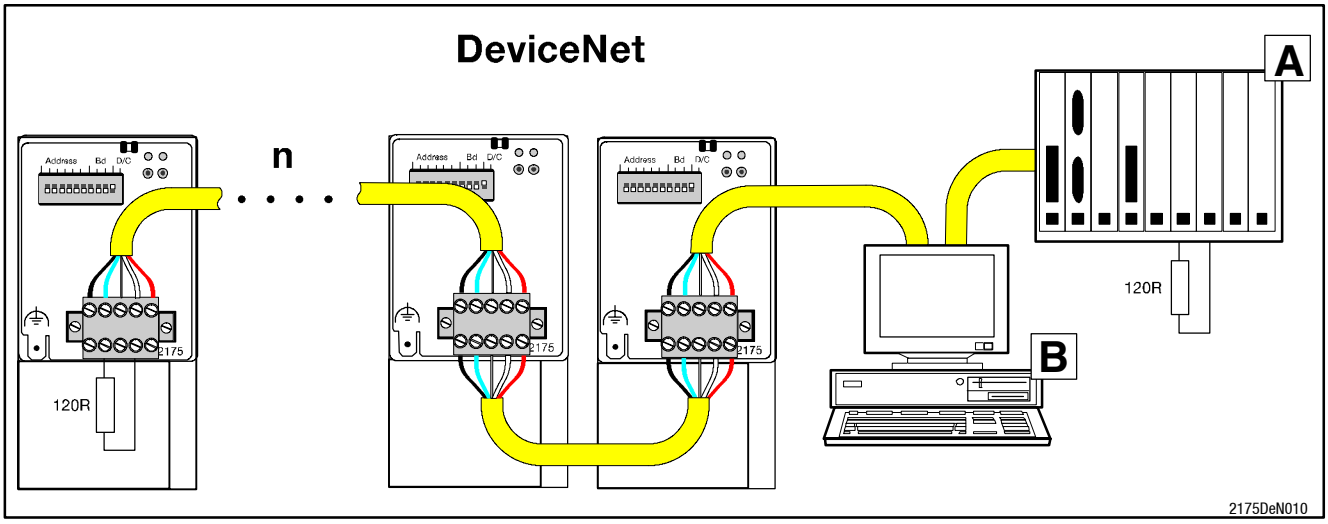


Fig. 10-1 DeviceNet line

The assignment of the plug/socket connector is described in chapter 10.3.2.

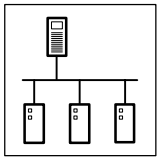
Chapter 9.5.1.3 informs about maximum cable lengths.

Not shown is the voltage supply required for the fieldbus modules.

Note!

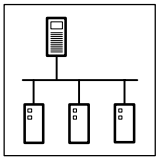
Please take into consideration that

- the shield of the DeviceNet cable is connected with each side (plug/socket connector, connection: Shield).
- a bus termination of 120 Ohm is available on both ends of the network.
- the shield is connected to GND on the master side.



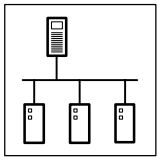
10.4 Features of the ‘thick”cable (according to DeviceNet Specification)

General features	Two shielded pairs	Common axis with drain wire in center.
	Total shielding	65% coverage 36 AWG or 0.12mm Cu braid (individually tinned)
	Drain wire	#18 copper min.; 19 AWG min. (individually tinned)
	Outside diameter	10.41 mm (min) to 12.45 mm (max.) roundness - radius delta to be within 15% of 0.5 O.D.
	Jacket marking	Vendor name & part # and additional markings
	DCR (braid+tape+drain)	5.74 Ohm/1000 m (nom. @ 20ºC)
	Certifications (U.S. and Canada)	NEC (UL) type, CL2/CL3 (min.)
	Bend readius installation / fixed	20 x diameter / 7 x diameter
	Operating ambient temperature	-20 to +60ºC @ 8 Ampere; de-rate current linearly to zero @ 80ºC
	Storage temperature	-40 to +85ºC
	Pull tension	845.5 N. max.
Features of the data cables	Conductor pair size	#18 copper (minimum); 19 AWG min (individually tinned)
	Insulation diameter	3.81 mm (nominal)
	Colours	libht blue blue, white
	Pair twist / m	approx. 10
	Tape shield over pair	2 mil / 1 mil, Al / Mylar. Al side outw/shorting fold (pull-on applied)
	Impedance	120 Ohm +/- 10% (at 1 MHz)
	Capacitance between conductors	39.37 pF / m at 1 kHz (nominal)
	Capacitance between a conductor and the conductor connected to shield.	78.74 pF / m at 1 kHz (nominal)
	Capacitive unbalance	3937 pF/1000 m at 1 kHz (nominal)
	DCR - @ 20ºC	22.64 Ohm/1000 m (maximum)
	Attenuation:	0.43 dB/100 m @ 125 kHz (maximum) 0.82 dB/100 m @ 500 kHz (maximum) 1.31 dB/100 m @1.00MHz (maximum)
Features of the DC voltage cables	Conductor pair size	#15 copper (minimum); 19 AWG minimum (individually tinned)
	Insulation diameter	2.49 mm (nominal)
	Colours	Red / black
	Pair twist/m	approx. 10
	Tape shield over pair	1.0 mil/ 1 mil, Al/Mylar. Al side out w/shorting fold (pull-on applied)
	DCR - @ 20ºC	11.81 Ohm/1000 m (maximum)



10.5 Features of the “thin” cable (according to DeviceNet Specification)

General features	Two shielded pairs	Common axis with drain wire in center.
	Total shielding	65% coverage 36 AWG or 0.12mm Cu braid (individually tinned)
	Drain wire	#22 copper min.; 19 AWG min. (individually tinned)
	Outside diameter	6.096 mm (min) to 7.112 mm (max.) roundness - radius delta to be within 20% of 0.5 O.D.
	Jacket marking	Vendor name & part # and additional markings
	DCR (braid+tape+drain)	10.5 Ohm/1000 m (nom. @ 20ºC)
	General features	Specification
	Certifications (U.S. and Canada)	NEC (UL) type CL2(min.)
	Bend readius installation / fixed	20 x diameter / 7 x diameter
	Oprating ambient temperature	-20 to +70ºC @ 1,5 Ampere; de-rate current linearly to zero @ 80ºC
	Storage temperature	-40 to +85ºC
	Pull tension	289.23 N. max.
Features of the data cable	Insulation diameter	1.96 mm (nominal)
	Conductor pair size	#24 copper (minimum); 19 AWG min (individually tinned)
	Colours	libht blue blue, white
	Pair twist / m	approx. 16
	Tape shield over pair	1 mil / 1 mil, Al / Mylar. Al side out w/shorting fold (pull-on applied)
	Impedance	120 Ohm +/- 10% (at 1 MHz)
	Propagation delay	4.46 ns/m (maximum)
	Capacitance between conductors	39.37 pF / m at 1 kHz (nominal)
	Capacitance between a conductor and the conductor connected to shield.	78.74 pF / m at 1 kHz (nominal)
	Capacitive unbalance	3.94 pF/1000 m at 1 kHz (maximum)
	DCR - @ 20ºC	91.86 Ohm/1000 m (maximum)
	Attenuation:	0.95 dB/100 m @ 125 kHz (maximum) 1.64 dB/100 m @ 500 kHz (maximum) 2.30 dB/100 m @1.00MHz (maximum)
Features of the voltage cable	Conductor pair size	#22 copper (Minimum); 19 AWG Minimum (individually tinned)
	Insulation diameter	1.4 mm (nominal)
	Colours	Red Black
	Pair twist/m	approx. 16
	Tape shield over pair	1 mil / 1 mil, Al / Mylar. Al side out w/shorting fold (pull-on applied)
	DCR - @ 20ºC	57.41 Ohm/1000 m (maximum)



DeviceNet

Installation

11 Commissioning

Note!

Settings using GDC, keypad and DeviceNet manager

Controller address and baud rate can also be set using GDC, the keypad or the DeviceNet Manager. For this the DIP switches S1 to S6 must be in OFF position.

Codes L-C0009 (controller address) and L-C0125 (baud rate) are not active if one or several switches have been set to ON before a restart.

Please see the descriptions for

- L-C1850/2350 node address
- L-C1851/2351 baud rate

The DIP switch on the front of the 2175 fieldbus module can be used for the following settings:

- Controller address S1 - S6
- Baud rate S7 - S9
- Communication profile S10

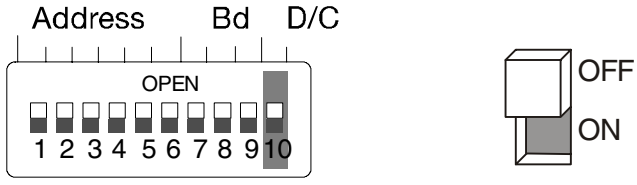
Note!

In default setting all switches are OFF.

The controller address and baud rate set using DIP switches will only be active after a restart.

Only the combinations shown in the following tables are defined:

11.1 Communication profile setting

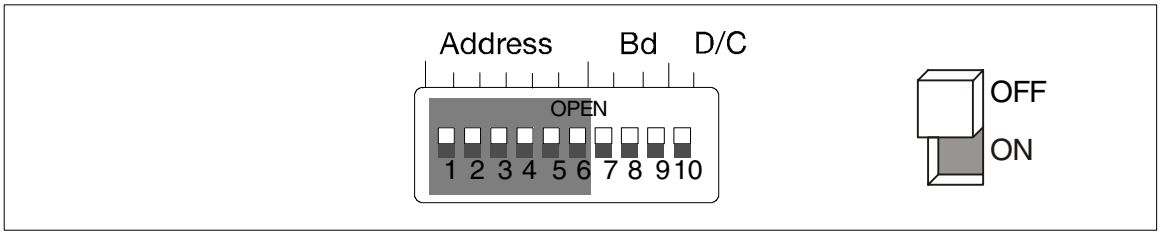


Communication profile	S10
DeviceNet	OFF
CANopen	ON

11.2 Controller address setting

Note!

Please ensure that the addresses are not the same when using several controllers.



$$Address_{dec} = S_6 \cdot 2^0 + S_5 \cdot 2^1 + S_4 \cdot 2^2 + S_3 \cdot 2^3 + S_2 \cdot 2^4 + S_1 \cdot 2^5$$

The address calculation (decimal number) is based on the positions of switches S1 ... S6 ('0' = OFF and '1' = ON). The numbers must be entered into the equation above.

The equation also indicates the valency of a switch. The sum of valencies results in the controller addresses to be set (see examples 1 and 2):

Switch valencies:

Switch	S1	S2	S3	S4	S5	S6
Valency	32	16	8	4	2	1

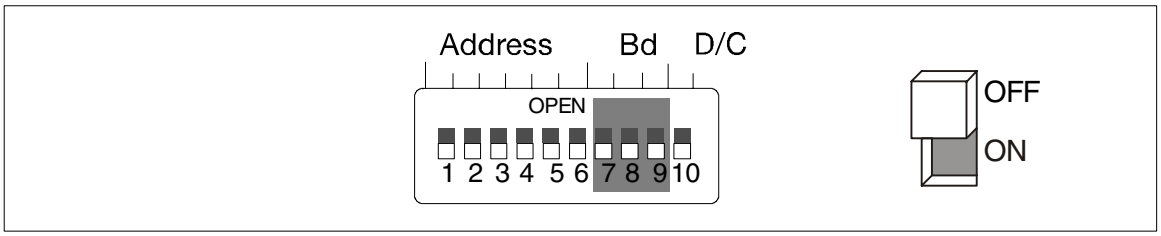
Example:

Switch	S1	S2	S3	S4	S5	S6
Switch position	ON	ON	ON	OFF	OFF	OFF
Address (= 56)	32	16	8	0	0	0

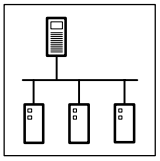
11.3 Baud rate setting

Note!

Please ensure that the baud rate is the same for all controllers and the host.



Baud rate [kbit/s]	**S7**	**S8**	**S9**
125	OFF	OFF	OFF
250	OFF	OFF	ON
500	OFF	ON	OFF



11.4 Initial switch-on

Please do not change the switch-on sequence explained below!

Stop!

Prior to initial switch-on of the controller, check the wiring for

- completeness
- short-circuit
- earth fault

1. Switch on the controller and, if necessary, the external supply of the 2175 fieldbus module.
   – A controller status LED ((📖 10-1) , pos. 3) must now be on or blinking. If this is not the case, see chapter ”Troubleshooting” (📖 13-1)
   – The green LED (”Controller connection status”) must be on, too (📖 10-1) (pos. 1). If this is not the case, see chapter ”Troubleshooting” (📖 13-1) .
2. It should now be possible to communicate with the drive:
   – All parameters (SDO) can be read and written
   – All SDO parameters except process data, such as frequency setpoint or control word, can be overwritten.
   – For more information on the communication phases of the CAN network see chapter 6.1.2 (📖 6-3) .

11.5 Drive enable via 2175 fieldbus module

82XX / 8200 vector	1. Set the Lenze parameter Operating Mode (L-C0001) from 0 to 3 to enable the drive via the 2175 fieldbus module. The parameter can be set using the keypad or directly via CAN. Examples for write (L-C0001=3): – Index = $5FFE_{hex}$ (results from $5FFF_{hex}$ – $(L\text{-}C0001)_{hex}$; see Lenze Codes, Addressing, chapter 6.2 📖 6-5) – Subindex: 0 – Value: 30000_{dec} (results from $3x10^4$)
	2. Terminal 28 (controller enable) is always active and must be set to HIGH during DeviceNet operation (see the corresponding Operating Instructions for the controller). – Otherwise, the controller cannot be enabled. – With 821X, 8200vector and 822X, the function QSP (quick stop) is always active. If QSP is assigned to an input terminal (default setting: not assigned), this terminal must be HIGH during DeviceNet operation (see the corresponding Operating Instructions). The controller now accepts parameter and process data.
93XX	1. For drive control via DeviceNet set the Lenze parameter Signal Configuration (L-C0005) to xxx3. These changes can be made using the 9371BB keypad or directly via DeviceNet. For first commissioning, select signal configuration 1013. Examples for write (L-C0005=1013): – Index = $5FFA_{hex}$ (results from $5FFF_{hex}$ – $(L\text{-}C0005)_{hex}$; see chapter 6.2 📖 6-5) – Subindex: 0 – Value: 10 130 000_{dec} (results from $1013x10^4$) 2. Set the parameter L-C0142 to 0 3. Terminal 28 (controller enable) is always active and must be set to HIGH during DeviceNet operation (see the corresponding Operating Instructions for the controller). Otherwise, the controller cannot be enabled. – With the signal configuration L-C0005=1013, the function QSP (quick stop) and the CW/CCW changeover are assigned to the digital input terminals E1 and E2 and thus they are always active. For DeviceNet operation E1 must be set to HIGH (see Operating Instructions for 93XX).
	With signal configuration L-C0005=xx13, terminal A1 is switched as voltage output. Therefore, only the following terminals must be connected via cables: • X5.A1 with X5.28 (ctrl. enable) • X5.A1 with X5.E1 (CW/QSP)
	The controller now accepts parameter and process data.

11.5.1 Protection against uncontrolled restart

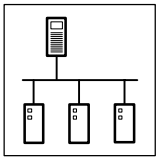
Note!

After a fault (e.g. short-term mains failure) a restart of the drive is not always wanted.

- By setting L-C0142 = 0, the drive can be inhibited if
  – the corresponding controller sets a “Message“ fault
  – the fault is active for more than 0.5 s

Parameter function:

- L-C0142 = 0
  – Controller remains inhibited *(even if the fault is no longer active)* and
  – The drive restarts in a controlled mode: LOW-HIGH edge at one of the inputs for “Controller inhibit“ (CINH, e.g. at terminal X5/28)
- L-C0142 = 1
  – Uncontrolled restart of the controller possible



12 Parameter setting

General information

Ensure to use the terminology defined for the DeviceNet communication profile.

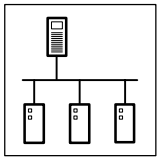
Two different telegram types are transferred between host and controller(s):

- Parameter data (explicite messages)
- Process data (I/O messages)

As indicated in the table, these telegram types are subdivided into communication channels according to their time-critical response.

Communication channel	Telegram type
→ Parameter data channel (chapter 6.2) – Enables the access to all Lenze codes. – Parameter changes are normally saved in the controller (observe C0003). – If the parameter channel is active, it assigns 4 words of the input and output process data. 🕮 6-5)	→ Parameter data (explicite messages) – operating parameters – diagnostics information – motor data In general, the parameter transfer is not as time-critical as the tranfer of process data.
→ Process data channel (chapter 6.3) – The controller can be controlled using process data (🕮 6-13). – The host has direct access to the process data. In the PLC, for instance, the data are directly assigned to the I/O area. – Process data are not stored in the controller and cyclically exchanged between host and controller (continuous exchange of current input and output data).	→ Process data Process data are, for instance, setpoints and actual values. Process data must be exchanged between host and controller as quickly as possible. Process data are usually small amounts of cyclically transferred data. • I/O polled messages (polled) – The poll command sent by the master contains output data for the slave. The slave then sends its input data to the master. The Poll response can also be used as receive message. • Cyclic I/O – With cyclic I/O master and slave generate their data independently of each other. The data is sent as set through the timer. The timer value is to be entered by the user. In default setting the cyclic I/O message is to be acknowledge by the consumer. If the timer cycle is set to a high value, the user can suppress the acknowledgements. • Change of state (COS) – This type of I/O message is a special type of cyclic message. COS nodes send their data when the data status changes or the heartbeat interval set by the user has expired. The heartbeat interval is set by means of the cyclic timer. NOTE: The default message type for 2175 DeviceNet modules is the I/O polled message. Other message types can be enabled using the DeviceNet Manager.

Tab. 12-1 Distribution of parameter data and process data to different communication channels



DeviceNet

Parameter setting

List of classes, instances and attributes

In DeviceNet, process and parameter data are addressed by classes, instances and attributes.

Because of their size, attributes are read individually by “GET single” and written by “SET single”.

The following instances/attributes are assigned to classes:

Identity class (Class1) (GET / --) INSTANCE 1

Attribute 1	Attribute 2	Attribute 3	Attribute 4	Attribute 5	Attribute 6	Attribute 7	Attribute 8
Vendor-ID: 445 (445 = LENZE)	**Device profile** 0 = Generic	**Product code** 0x01** (see below)	**Maj./minor rev.**	**Status**	**Serial number** Unambiguous number of a module	**Product name** 2175 IB	**CFG consistency**

The product code (attribute 3) of the identity class contains “01” in the HIGH byte for the 2175 fieldbus module. The LOW byte contains the number of process data words.

The information given in the identity class are unambiguous for every module. Attribute 6 contains the default setting for every module.

Message router class (class 2) (GET/---) INSTANCE 1

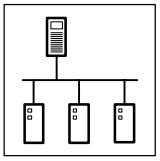
The message router class supports the settings for the communication features.

Attribute 1	Attribute 2	Attribute 3	Attribute 4	Attribute 5	Attribute 6	Attribute 7	Attribute 8
Class List	**Max. # of cnct**	**Curr. # of cnct**	**CAN IDs**				

Assembly class (class 4) (GET/---) INSTANCE 1

Assembly class specifies the relation between data and objects.

Attribute 1	Attribute 2	Attribute 3	Attribute 4	Attribute 5	Attribute 6	Attribute 7	Attribute 8
Max. instance	**No. of members in list**	**Member list**	**Data**				



Connection class (class 5) (GET/---)

The connection class determines process and parameter data features.

Attribute 1	Attribute 2	Attribute 3	Attribute 4	Attribute 5	Attribute 6	Attribute 7	Attribute 8
Instance 1							
State	**Instance type** Explicit messaging (0)	**Transport class**	**Produced Connection ID**	**Consumed Connection ID**	**Init. comm. char.**	SDO length (see TIP) **Prod. connection Size**	**Cons. connection Size**
Instance 2							
State	**Instance type:** (1)	**Transport class**	**Produced connection ID**	**Consumed Connection ID**	**Init. comm. char.**	PDO length (see TIP) **Prod. connection Size**	**Cons. connection Size**
Instance 3							
State	**Instance type:** (1)	**Transport class**	**Produced Connection ID**	**Consumed Connection ID**	**Init. comm. char.**	**Prod. connection Size**	**Cons. connection Size**

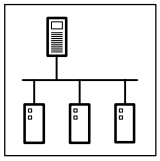
Tip!

The PDO length depends on the controller used and the settings in class 102

- 820X: 4 byte
- 8200 vector: 6 byte
- 8210/8220: 6 byte
- 93XX: 8 byte (all technology variants except 9300 servo PLC)
- 9300 servo PLC 9300: 24 byte

Acknowledge handler class (class 43)

The acknowledge handler object is needed to handle acknowledgements when using COS/cyclic messages.



DeviceNet

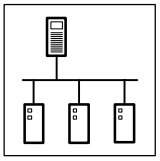
Parameter setting

Vendor-specific class (class 100) (GET/SET)

Byte	Instance 1		
	• Attribute 1: Parameter data		
	Request	**Response**	
1	Code Low	Code Low	
2	Code High	Code High	
3	Subcode	Subcode	
4	Reserve	Status	• Byte value = 0: No errors • Byte value ≠ 0: Incorrect message The byte value indicates the error number. Its meaning is explained in the table below.
5	Data_1 Low	Data_1 Low	
..	Data_1 High	Data_1 High	
..	Data_2 Low	Data_2 Low	
	Data_2 High	Data_2 High	
	.	.	
	.	.	
	.	.	
	.	.	
	Data_22 Low	Data_22 Low	
	Data_22 High	Data_22 High	

The following table lists the explanations for the error numbers:

Error code (hex)	**Explanation**
03	Invalid data type
04	Invalid subcode no.
05	Invalid code no.
07	No access during operation
08	No access because of operating mode
09	No access because parameters can only be read
0A	No access authorisation
0B	Data block too long
0C	Collision with other value ranges
0D	Quit value range
0E	General value range error
2X	Error in AIF interface
jFF	General error



Vendor-specific class (class 101) (GET/SET)

Instance 1
• Attribute 1: Process data monitoring code (2 byte) 0: No response (default) 1: Controller inhibit 2: Quick stop This setting is saved as non-volatile value in the EEPROM.

Vendor-specific class (class 102) (GET/SET)

Instance 1
• VSC attribute 1: Process data length (2 byte)
0: Default value according to EDS file 1: 2 byte ... 12: 24 byte Note: When changing the process data length, the process data length in the master must be adapted under ”produced / consumed data size”.

12.1 Communication-relevant Lenze codes

Tip!

If the 2175 module is plugged onto a different controller during operation, an undefined operating status might occur.

The behaviour of servo inverters and frequency inverters is determined by their parameters. Lenze controllers can be parameterised by codes. These (Lenze) codes are exchanged between master and 2175 fieldbus module as part of a telegram and via a CAN bus.

Depending on the Lenze inverter, two codes are available for communication via CAN bus: L-C1810 and L-C1811.

Communication with the drive is only possible when the controller is know as system device. The devices are detected while the modules are initialised.

Address and baud rate can only be set using the front switch:

- Switches 1 - 9 ≠ OFF (see chapter 10)
- Switch 10 = OFF: DeviceNet communication profile

12.1.1 Description of communication-relevant Lenze codes

How to read the table

Column	Meaning
Code	(Lenze) code
Subcode	Subcode
Index	Incidated as hexadecimal/decimal value
Lenze	Code default setting
Selection	Minimum value [smallest step/unit] maximum value
Data type	VS: Visible string, indicated length

12.1.1.1 L-C1810: Software product code

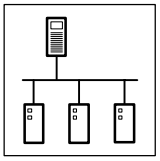
Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
L-C1810	-	$22765_d = 58ED_h$	-	-	Visible string

During module initialisation the bus devices are defined by means of the product code.

12.1.1.2 L-C1811: Software date

Code	Subcode	Index	Possible settings		Data type
			Lenze	Selection	
L-C1811	-	$22764_d = 58EC_h$	-	-	Visible string

This date is mainly needed for service.



12.2 Notes to be observed for paramater setting

12.2.1 8200 controllers

The following applies to the 8200 inverter series:

Caution!

Parameter setting (codes except C046, C0135) is only possible when the controller is inhibited. Parameters are accepted when the controller is enabled, but they are not saved.
After having set a parameter, the controller must not be addressed for approx. 50 ms; otherwise the command will be ignored.

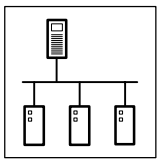
After parameter setting, the controller needs up to approx. 70 ms to set the status 'enabled' (terminal, C040, C0135).

The TRIP reset function is activated by inhibiting the controller and enabling it again under C040 or C0135.

The TRIP reset function initializes the 8200 inverter and the 2175 fieldbus module. Therefore the TRIP reset command is not acknowledged for the master.

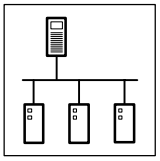
12.2.2 8200 vector controllers

Digital and analog input and output signals can be freely configured (see Operating Instructions; codes C0410, C0412, C0417 and C0421)



DeviceNet

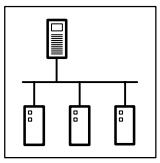
Parameter setting



13 Troubleshooting and fault elimination

13.1 No communication with the controller.

Possible causes	Diagnostics	Remedy
Is the controller switched on?	The operation status LED of the basic unit must be on 📖 10-1 Point 3.	Supply controller with voltage (see Operating Instructions for the basic unit)
Is the fieldbus module supplied with voltage?	The green LED for “Controller connection status” on the fieldbus module 📖 10-1 must be on (Remedy 1) or blinking (Remedy 2)	With supply from the basic unit check the connection. With external supply check the 24 V voltage at terminals 39 and 59. A voltage between 24 V +10 % must be applied.
		The fieldbus module has not been initialised with the controller yet. Possibility 1: Controller not switched on (see fault possibility 1). Possibility 2: Check the connection to the controller
Does the controller receive telegrams	The LED “Bus connection status“ on the fieldbus 📖 10-1 must be blinking in green when the device is communicating with the host.	Check whether the connection corresponds to the instructions given in chapter ”CAN bus wiring“, page. 📖 10-1 . Check whether host sends telegrams and uses the appropriate interface.
		CAN controller address (L-C0009 / DIP switch) and baud rate (L-C0125 / DIP switch) can be set differently for controller and host. Ensure that the addresses are identical.
		The controller address (L-C0009) must be different for all connected devices. Check whether some addresses are used twice and correct them if necessary. Check the wiring to your host.

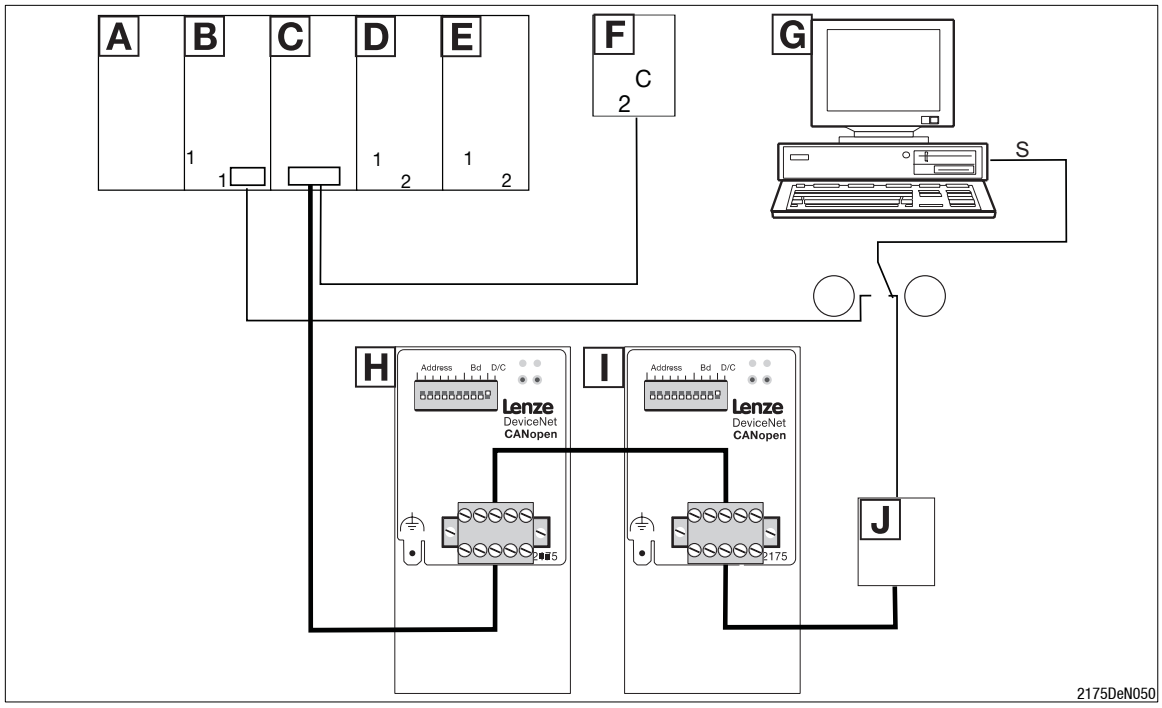


DeviceNet

Troubleshooting and fault elimination

14 Appendix

14.1 DeviceNet commissioning example



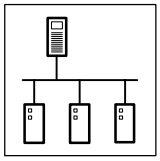
Tab. 14-1 Overview over the hardware and software used

Position	Module	
A	Power supply, type: SLC 500 modular chassis (1746-A4)	
B	SLC 5/04 CPU type: SLC 5/04 CPU (1747-L541)	
C	DeviceNet scanner type: DeviceNet scanner (1747-SDN scanner module)	
D	Digital input type: 32 digital input (1746-IB32)	
E	Digital output typ: 32 digital output (1746-OB32)	
F	External voltage supply for DeviceNet	
G	PC with Allen-Bradley software: • RSLinx (revision 2.20.02) • RSNetworx (revision 3.00) • RSLogix (revision 4.10.01	
H	Controller type: 8200vector, node 28	Setpoint selection via process data channel of the 2175 fieldbus module: C0001 = 3
I	Controller type: 9300 servo inverter, node 34	Setpoint selection via process data channel of the 2175 fieldbus module C0005 = xxx3
J	RS232 interface module type, 1770-KFD	

Parameter setting for communication via DeviceNet:

①	Program transfer to CPU
②	DeviceNet configuration

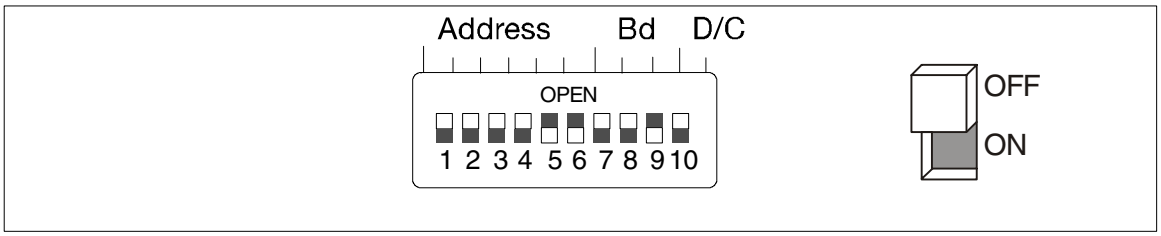
According to the DeviceNet specification, thin cables (📖 10-7) have been used for wiring.



DeviceNet

Appendix

Hardware adjustment on the front of the 2175 fieldbus module



The DIP switch positions correspond to the settings:

– Address = 3
– Baud rate = 250 kbit/s
– Communication profile: DeviceNet

14.1.1 Integration of LENZE-EDS files into “RSNetWorx”

EDS file handling

A floppy is included in the delivery package of the 2175 fieldbus module.
The floppy contains several electronic data sheets (EDS files **E**lectronic **D**ata **S**heets) for the following LENZE controllers:

– 2175ib02.eds: 8200 frequency inverter
– 2175ib03.eds: 8210, 8220, 8200 vector frequency inverter
– 2175ib04.eds: 9300 servo inverter
– 2175ib12.eds: 9300 servo PLC

EDS files are text files describing controllers according to the DeviceNet specification. The corresponding files are archived in the work station and called up by the installation software when being needed (see example “RSNetWorx”, Allen Bradley).

The file name includes the fieldbus (’2175’), two letters (’ib’, interface module) and a two-digit number which indicates the number of supported process data words (16 bit/word).

Tip!

You can read EDS files by using an ASCII editor.

We recommend not to change anything. Every change can have a negative effect on the module!

EDS file download

EDS files can be downloaded from the Internet. See the Lenze homepage: **http://www.Lenze.de** (**Service→Download→DeviceNet**).

Enter the following for saving the self-decompressing file on your hard disk (the subdirectory “LENZE” must have been created earlier).

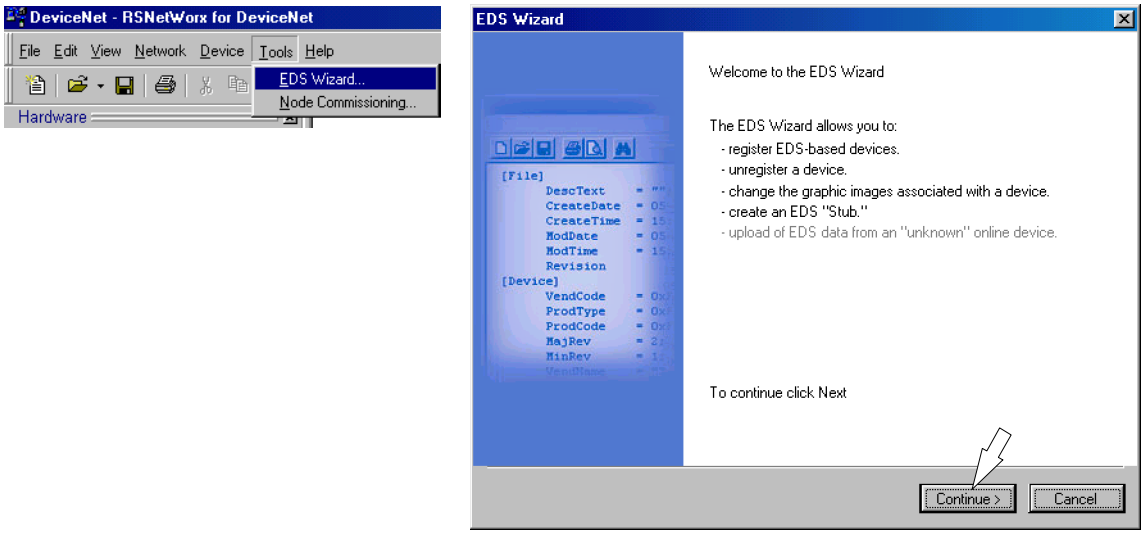
```
RockwellSoftware\RSCommon\EDS\LENZE
```

After decompression, the LENZE-EDS files are available for use.

Integration into “RSNetWorx”

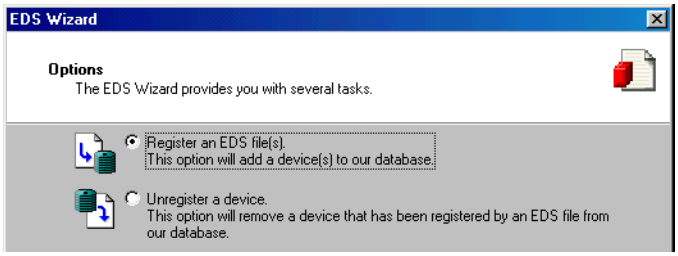
EDS files must be integrated using the installation program “RSNetWorx” :

1. Start “RSNetWorx”
2. Start EDS wizard under **Tools→EDS wizard**



2175DeN051/052

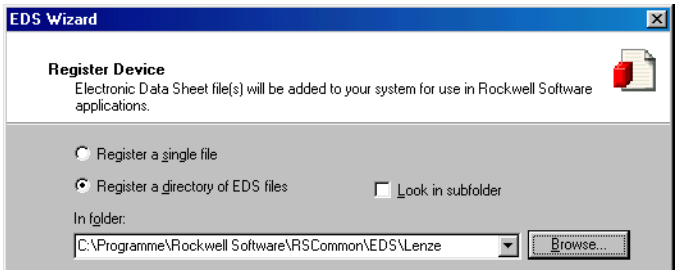
3. Select **Register an EDS file**



2175DeN053

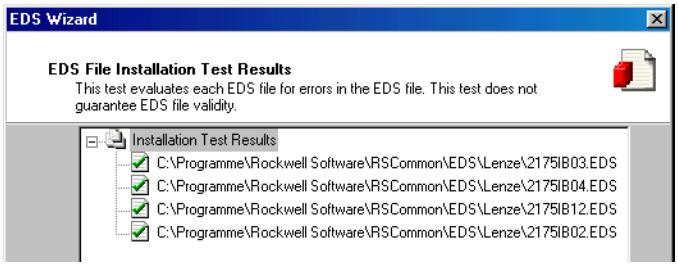
4. Select **Register a directory of EDS files**

A group of LENZE-EDS files and their path (**Browse**) must be registered.



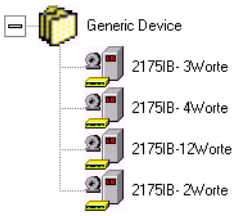
2175DeN054

5. Wizard finds the LENZE-EDS files (”2175IBxx.EDS”) and indicates the test result.



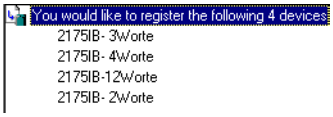
2175DeN055

6. All assigned symbols are displayed.



2175DeN056

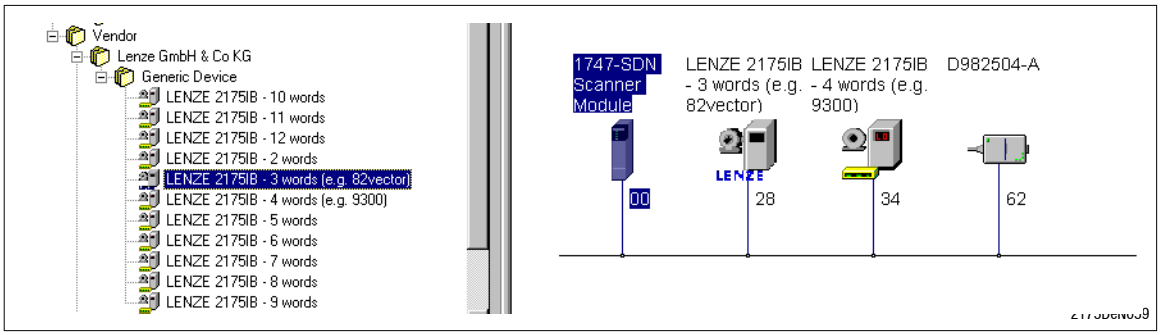
7. The registration query will be completed when pressing **Complete** .



2175DeN057

All inserted data are now available for the hardware catalog. The path is:

`DeviceNet/Vendor/Lenze GmbH&CoKG/Generic Device`



The process data length depends on the controller used.

Select according to the controller:

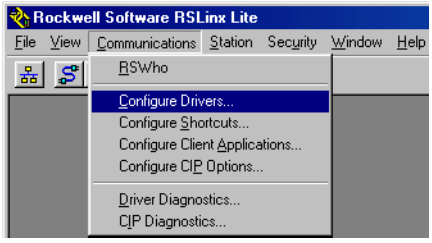
- 820X: 4byte = 2 words
- 821X/822X: 4byte = 2 words
- 8200vector: 6byte = 3 words
- 93XX: 8byte = 4 words
- 9300 servo PLC: 24byte = 12 words

Tip!

"RSNetWorx" automatically assigns the EDS files if the controller is connected online.

14.1.2 Communication between PC and DeviceNet via “RSLinx”

1. Start “RSLinx”
2. Select **Communication →Configure Drivers**

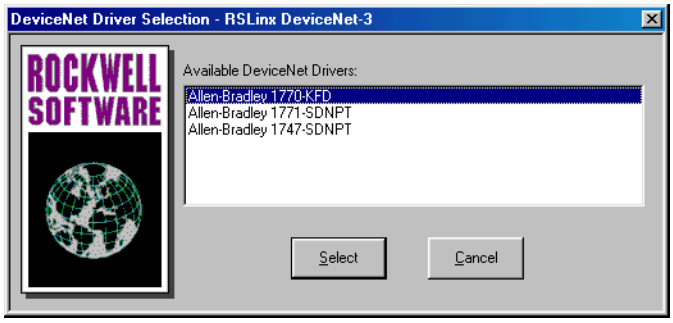


2175DeN086

3. Select *DeviceNet Drivers*. Confirm with **Add new**.

2175DeN060

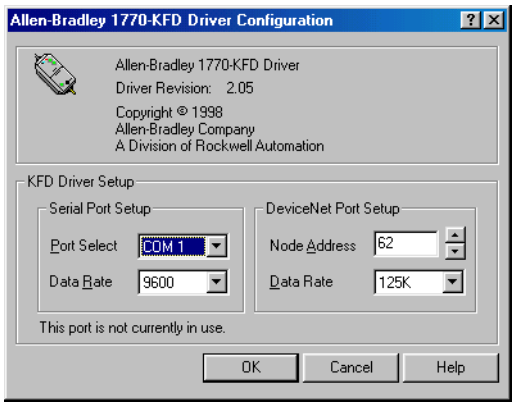
4. Select *Interface 1770-KFD*. Confirm with **Select** .



2175DeN061

5. Configuration of PC interface data in KFD driver setup.

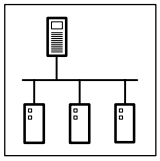
Press **OK** to check the configuration and initialise the KFD box.



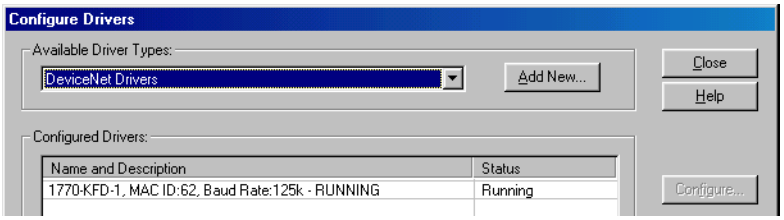
2175DeN062

Tip!

A second name can be assigned to the interface to make identification easier. Here it is “1770-KFD-1”. This name has been suggested by RSLinx and has not been changed.



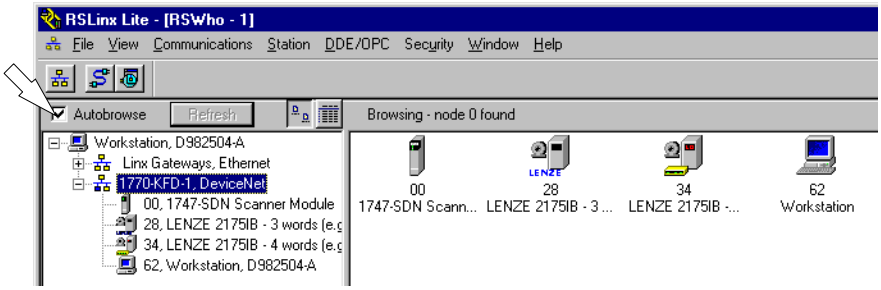
The interface configuration is now completed and indicates ”Running” as current status.



2175DeN063

6. Close the window with **Close** .

After the configuration of the “1770-KFD-1” interface has been completed, the bus structure and assigned node addresses will be output if the function *Autobrowse* (arrow) has been activated.



2175DeN064

This example uses the following components:

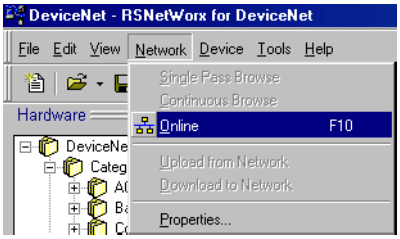
Address	Component
00	Scanner
28	Addressed 2175 module
34	Addressed 2175 module
62	PC workstation / 1770 KFD box interface

Tip!

- The “RSLinx” software tool must run in the background to coordinate PC and KFD box.
- If node addresses are marked with question marks, Lenze components cannot be identified because the corresponding EDS files are missing. Chapter 14.1.1 describes how to proceed.

14.1.3 Online connection between DeviceNet and “RSNetWorx“

1. Select **Network→Online**.

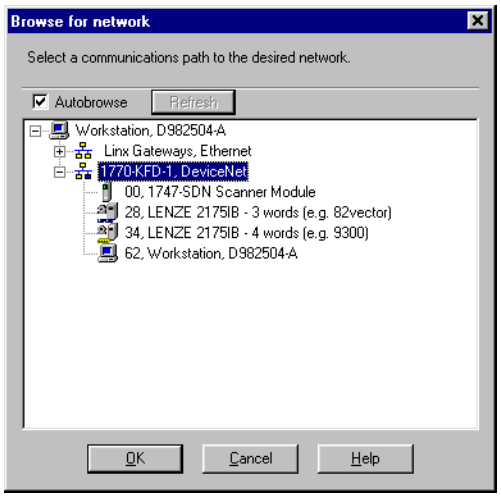


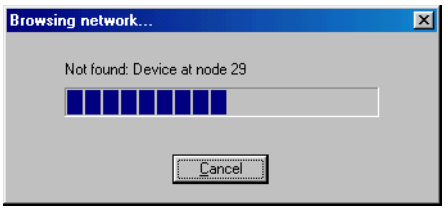
2175DeN065

The menu for scanning DeviceNet will be opened. *1770-KFD-1* lists all bus nodes.

2. Select **Autobrowse** or **Refresh**. Confirm with **OK**.

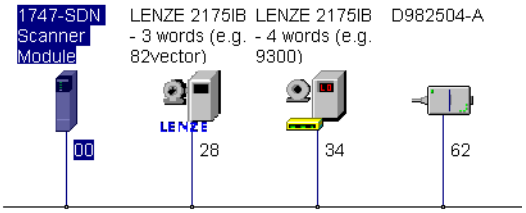
The bus is searched for connected devices.





2175DeN066/067

The components found are indicated as follows:

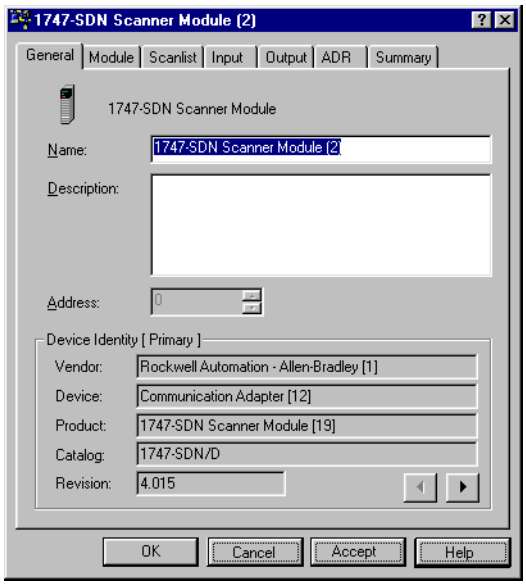


2175DeN068

Processing assignment (scanner properties)

Set the I/O address assignment between CPU and DeviceNet devices under Properties of the DeviceNet scanner using “RSNetWorx”.

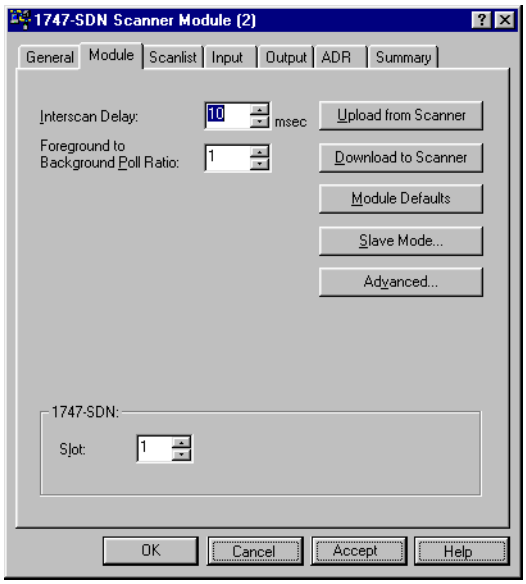
1. General scanner data are indicated when the scanner symbol is double clicked.



2175DeN069

2. Select the register card **Module**

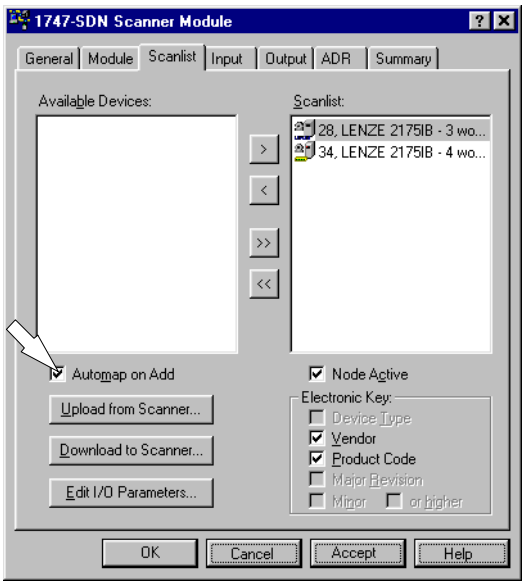
You will be asked whether you want to upload/download the components connected to the DeviceNet before the modules can be displayed. The window **Module** shows the scanner configuration.



2175DeN071

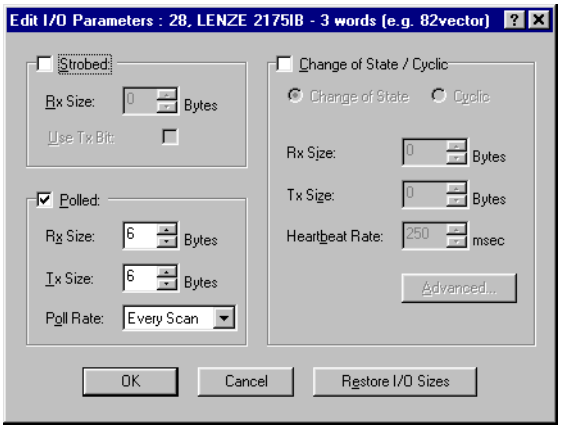
3. Register card **Scan list**

Under *Available devices* you will find a list of all components which can be shifted into the *Scan list*. Use **Automap on add** (arrow) to automatically enter all devices into the *Scan list*. The DeviceNet devices listed here will be processed by the scanner.



2175DeN072

- The selected module can be configured via **Edit I/O Parameters**.



2175DeN073

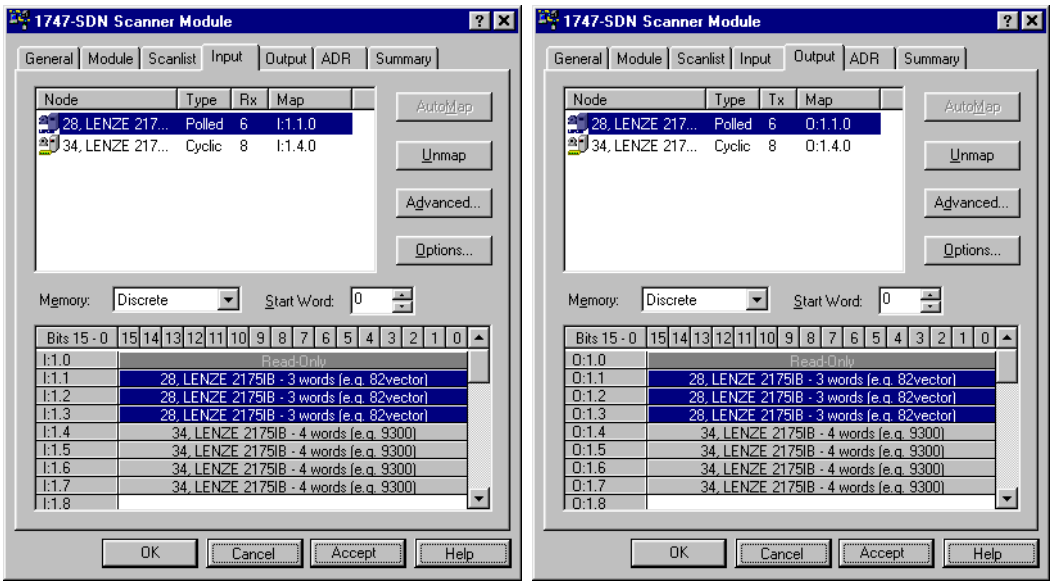
Use **Edit I/O Parameters** to select the corresponding telegram type for a node address.

The different protocols are described in the chapter about DeviceNet parameter setting.

I/O data of the following message types can be processed:

- Strobed
- Polled
- Change of state
- Cylic

4. Register card **Input** / register card **Output**

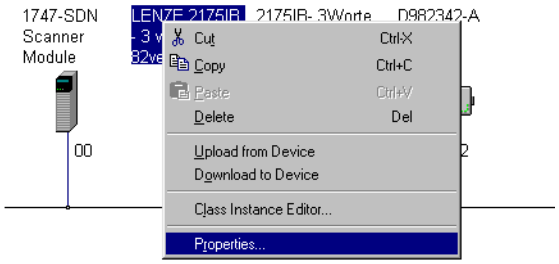


- Use **Input** to enter the peripheral input address of every device.
  – The function **Automap** automatically assigns the input address in the DeviceNet scanner. Select **Unmap** to remove this assignment.
  – The input addresses determined here must be taken into account for the project under RSLogix
- Use **Output** to enter the output peripheral address of every device.
  – The function **Automap** automatically assigns output addresses in the DeviceNet scanner. Select **Unmap** to remove this assignment.
  – The output addresses determined here must be taken into account for the project under RSLogix

The configuration for DeviceNet communication is completed now. The controllers are saved in the scanner under the node and I/O peripheral address.

Processing assignment (DeviceNet slave properties)

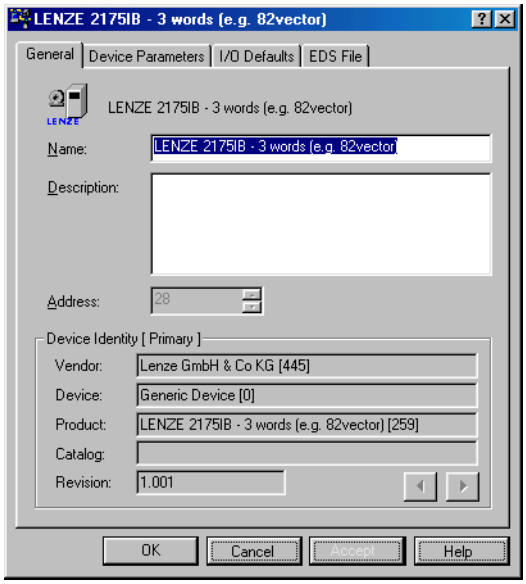
1. Select a controller (right mouse key)
2. Select *Properties*



2175DeN078

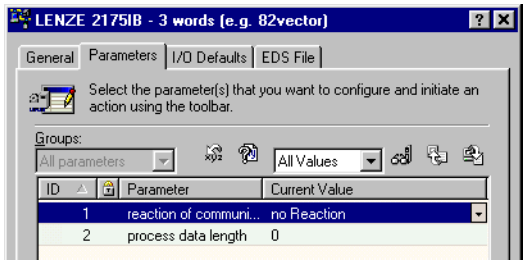
- Register card **General**

Here you will find general data of the EDS file for a certain device.

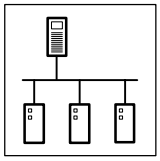


2175DeN079

- Register card **Parameters**
  - Here the controller response to interruption of the process data transfer and communication errors is parameterised. (The module must be supplied with voltage)
  - The length of process data must be entered if the process data size is less than set in default setting (setting = 0).



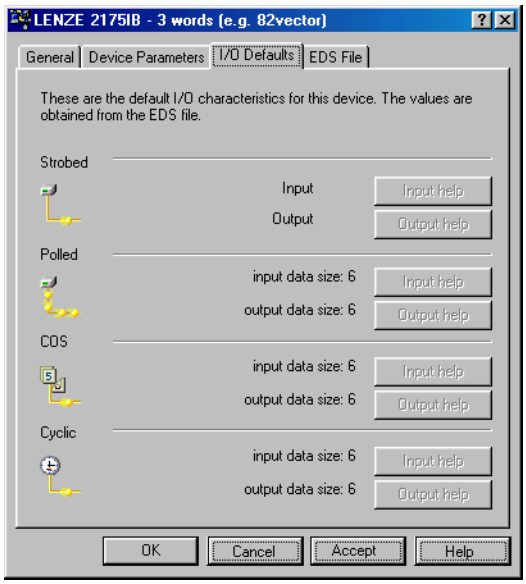
2175DeN080



- Register card **I/O defaults**

Tip!

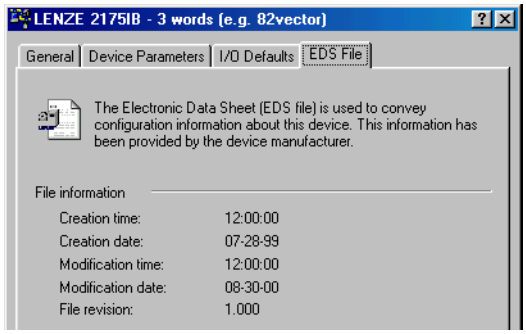
Modifications cannot be made since the communication profile has been defined in the scanner.



2175DeN081

- Register card **EDS file**

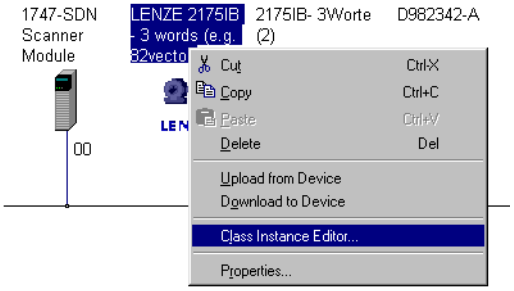
This card shows the identification data of and EDS file.



2175DeN082

Class Instance Editor

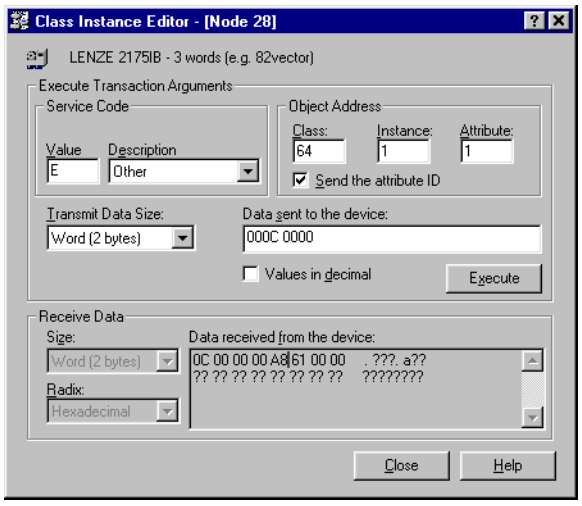
The class instance editor can be used for checking communication or modifying controller parameters.



2175DeN083

Confirm the query with **YES**

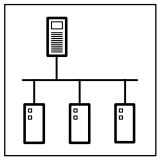
Go to **Service class instance attribute editor** and *Object address* to access controller data.



2175DeN084

Proceed as follows to read, e.g., the acceleration time (code L-C0012) from the controller:

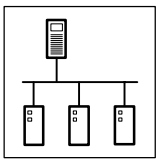
Input HMI	Input	Explanation
• Description	Other	
• Value	I_{hex} (get single attribute), value = 10_{hex} (set single attribute)	
• Class	64_{hex} (always with Lenze controllers)	
• Instance	1 (always with Lenze controllers)	
• Attribute	1 (always with Lenze controllers)	
• Data sent to the device	000C 0000	
• Data received from the device	Response: 0C 00 00 00 A8 61	Because of the left-justified Intel data format the response sequence is LOW byte/HIGH byte.
1st word	Lenze code L-C0012	0C 00_{hex}
2nd word	00 + subcode 00	00 00
3rd word	A8 61	Enter the following for an acceleration time (L-C0012) of 2.5 s:
4the word	Not used	Value * 10000 = 2.5 * 10000 = 25000 = $61A8_{hex}$



14.1.4 Set scanner status

The scanner is equipped with two status LEDs and a 7-segment display (3 digits):

- LED MODULE: Status display for the scanner module (should be green)
- LED NET: Status disply for the network (DeviceNet) (should be green)
- 7-segment display: Every second the display changes from the DeviceNet node address to the current diagnostics status and back.
  – Set status for program mode (no PLC program active)
    node address 00 (scanner) = 80 (scanner in idle running)
  – Set status for ”Run mode” (control active)
    always 00



14.1.5 Communication between PC and PLC

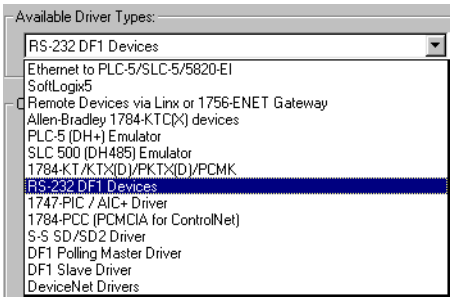
PC and PLC communicate via the RS232 interface of the PC and the CPU of the PLC (ALLEN-BRADLEY) (see Tab. 14-1, ①).

Communication is controlled by the "RSLinx" program. It must be configured and activated before online programming starts.

Tip!

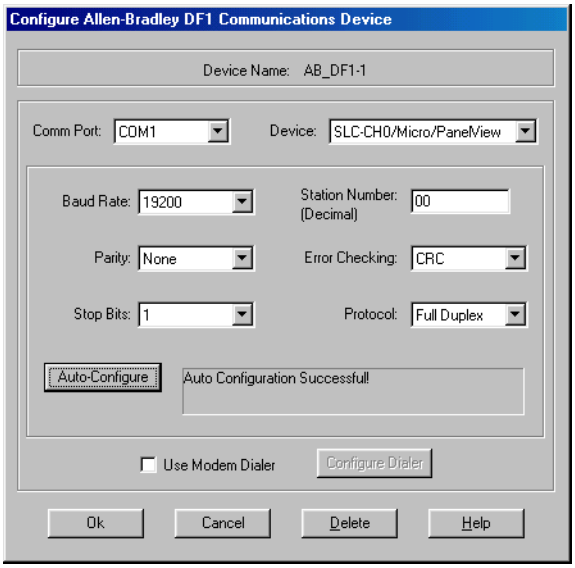
"RS Linx" must not be closed, but minimised (the bottom task bar must show the symbol to indicate that communication is possible).

1. Start "RS Linx"
2. Select **Communication / Configure drivers**
3. Select *RS-232 DF1 devices*. Confirm with **Add new**. The connection is automatically configured. Find a name with max. 15 characters.



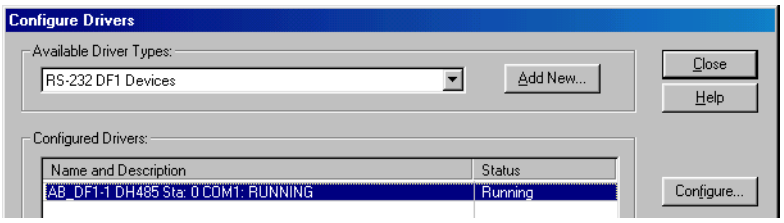
2175DeN087

4. **Auto-Configure** must be confirmed. **Auto configuration successful!** will be displayed after the process has been completed.



2175DeN088

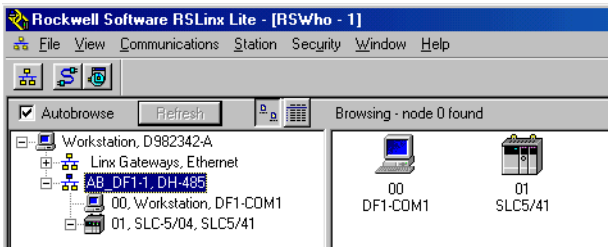
5. The next window will be displayed. Close the window with **Close** .



2175DeN089

With **Autobrowse** you can look at the communication in the start window of “RSLinx”:

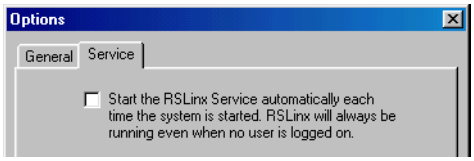
- Address 00: PC
- Address 01: SLC5/04



2175DeN065

Note!

Select **View/ Options/ Service** to determine whether “RS Linx” is activated automatically at every PC start.

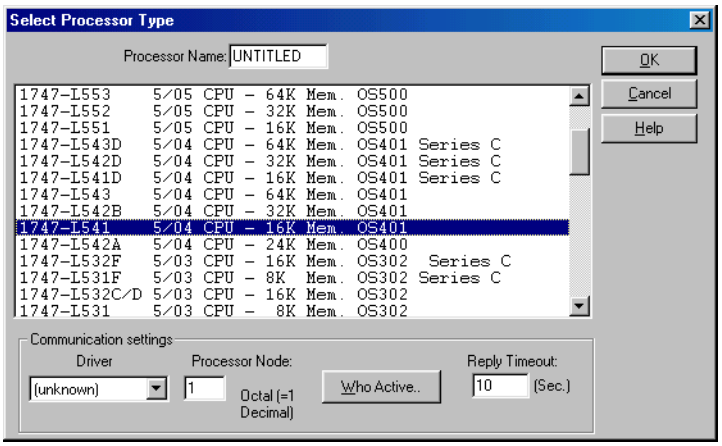


2175DeN091

14.1.6 Starting of the “RSLogix 500 English” PLC program

1. Select **File new** to create a new project.
2. Select “SLC5/04 CPU”

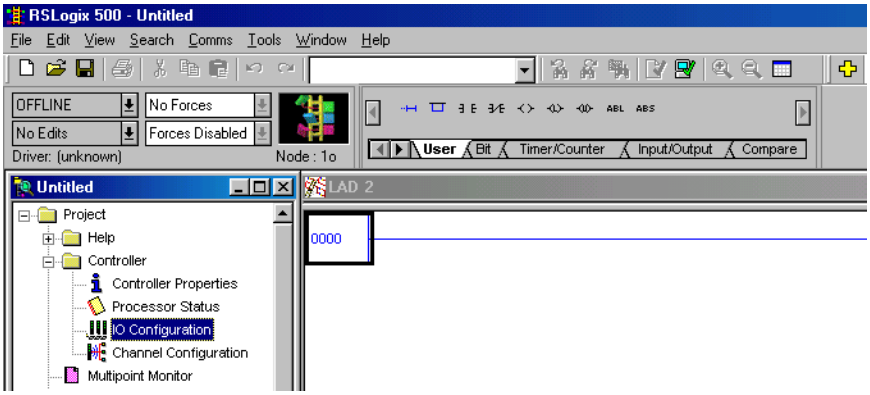
(Rack with four slots; exact name on CPU flap with “1747-L541”)



2175DeN092

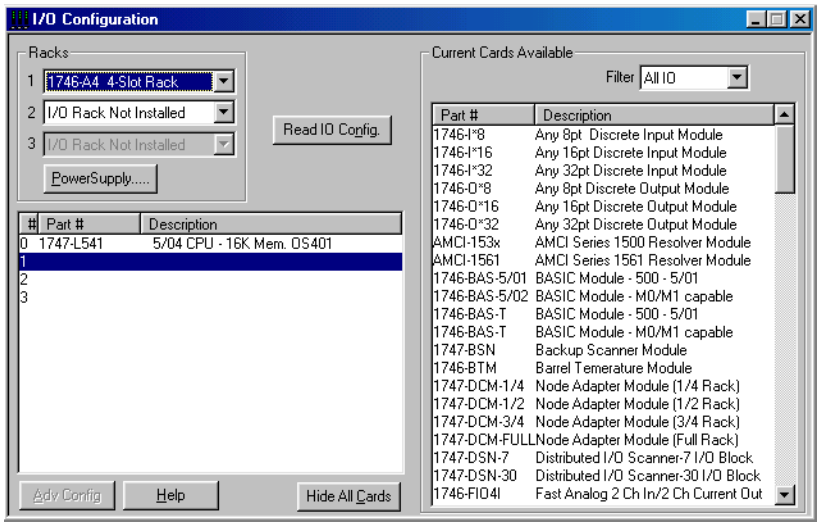
The window changes to the current project.

The rack components are organised by clicking **I/O Configuration**.



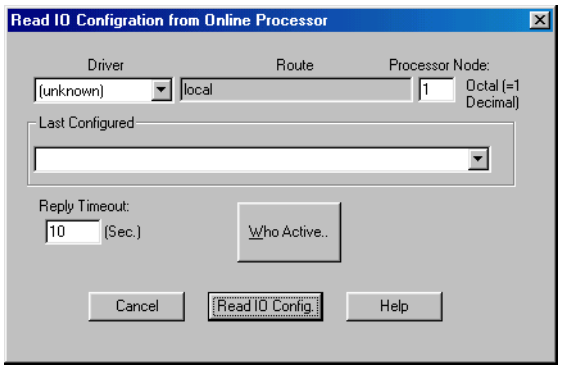
2175DeN093

3. Select **Read IO Config** to read the I/Os from the CPU



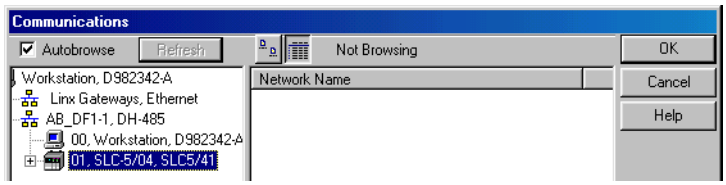
2175DeN094

4. Select **Who Active**. The active driver will be selected.



2175DeN095

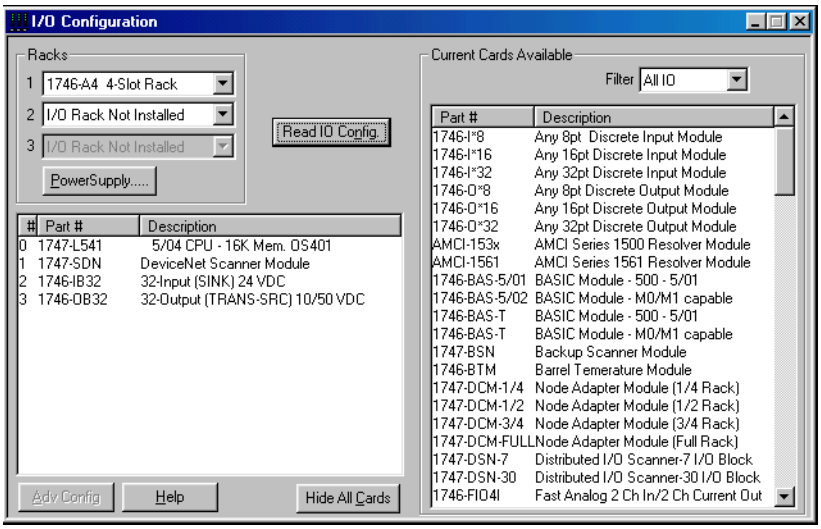
5. Select *SLC 5/04*. Confirm with **OK**.



2175DeN096

6. Select **Read IO config** to automatically start search and 'fill' the rack.

The individual slots are equipped with the components shown in Tab. 14-1



2175DeN097

The I/O configuration is completed. Close the window with **X** .

14.2 Control program programming

The SLC programming is explained by means of the example named “82VEC2175PZD_Para.RSS”.

Note!

This example is also available on the internet (“http://www.Lenze.de)

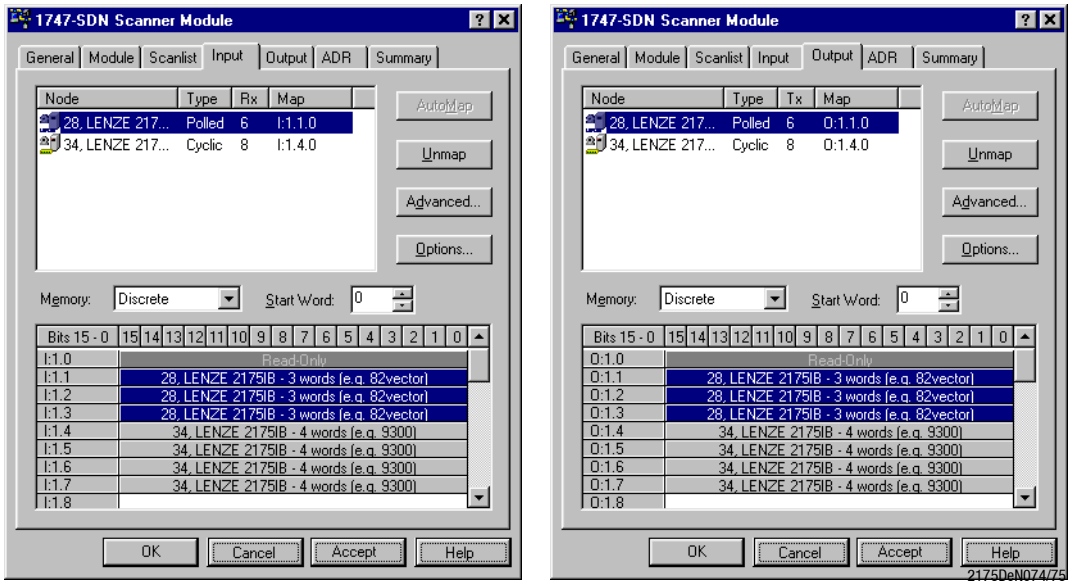
Path *Service* → *Downloads* → *DeviceNet*

14.2.1 Hardware addressing

DeviceNet is configured with the following node addresses

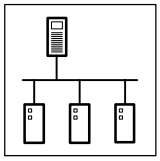
- Node address 28 (2175 fieldbus module with 8200 vector)
- Node address 34 (2175 fieldbus module with 9300 Servo)

The address assignment is to be configured in the scanner properties with “RSNetWorx”.



Default setting of input and output words for Lenze controllers:

Word	PCD	AIF-IN controller input	AIF-OUT controller output	Controller	
				8200 vector	9300 servo inverter
1	1	AIF-CTRL = C135	AIF-Stat = C150	•	•
2	2	AIF-W1 = Speed setpoint	AIF-W1 = Act. speed value	•	•
2	3	AIF-W2 = Open	AIF-W2 =	•	•
3	4	AIF-W3 = Open	AIF-W3 =	-	•



14.2.2 Explicit messages

"Explicit Messages" can transfer parameter data between controllers and the SLC-CPU via the DeviceNet parameter channel.

Data between the "SLC5/04-CPU", DeviceNet scanner "1747" and the DeviceNet device is transferred via "module files":

Module file	DeviceNet data direction	Data path
• M0	Output data	from the CPU to the DeviceNet node via a scanner
• M1	Input data	from the DeviceNet node to the CPU via a scanner

The CPU writes or reads module file data by means of copy commands from the scanner. The data are organized by the CPU memory of the processary. Further data processing then uses the data from the processor memory.

Addressing:

DeviceNet data direction	Module file	Scanner rack position	Data length in words (max. 224 words)
Output data	M0	1.	224
Input data	M1		

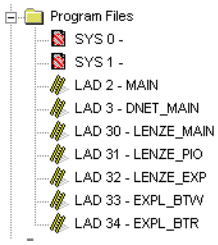
DeviceNet output data M0

Word	HIGH byte	LOW byte		Explanation
0	TXID	COMMAND	depending on scanner	Data as hexadecimal values • TXID Order number • COMMAND 01 (execute block) 04 (clear response buffer) • PORT 00, with SLC only 00 • SIZE Data length in byte • SERVICE 0E (GET SINGLE ATTRIBUTE) 10 (SET SINGLE ATTRIBUTE) • MAC ID DeviceNet node address • User data – CLASS = 0064 – INSTANCE = 0001 – ATTRIBUTE = 0001 – L code – L subcode – Value
1	PORT	SIZE		
2	SERVICE	MAC ID		
3	CLASS		Parameter transfer data	
4	INSTANCE			
5	ATTRIBUTE			
6	L-Code			
7	L subcode			
8	Value (Low word)			
9	Value (High word)			

DeviceNet input data M1

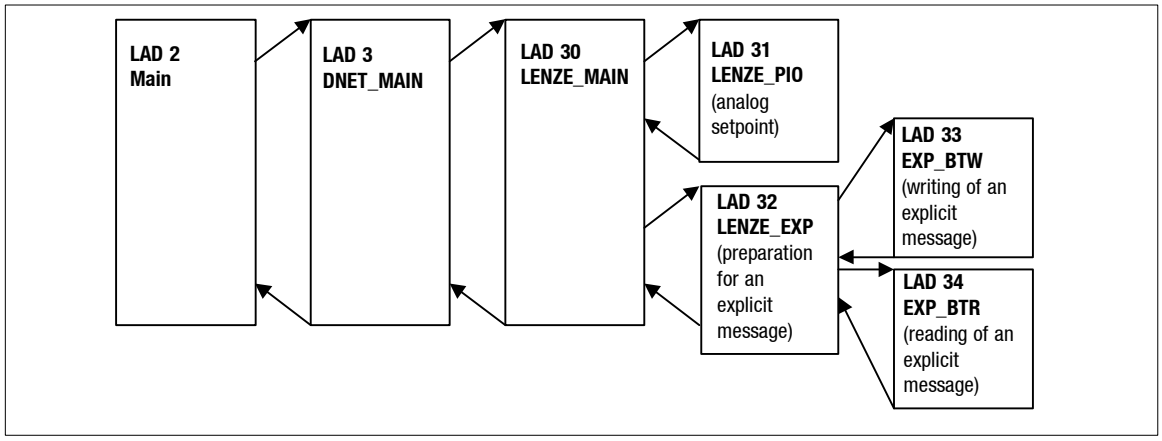
Word	HIGH byte	LOW byte	Data content	Explanation
0	TXID	STATUS	depending on scanner	Data as hexadecimal values • TXID Order number • STATUS 01 (transaction successful) (excerpt) 03 (slave not in scan list) • PORT 00, with SLC only 00 • SIZE Data length in byte • SERVICE • MAC ID DeviceNet node address • User data – CLASS = 0064 – INSTANCE = 0001 – ATTRIBUTE = 0001 – L code – L subcode – Value
1	PORT	SIZE		
2	SERVICE	MAC ID		
3	L code		Parameter transfer data	
4	L subcode			
5	Value (LOW word)			
6	Value (HIGH word)			
10				

14.3 Program components



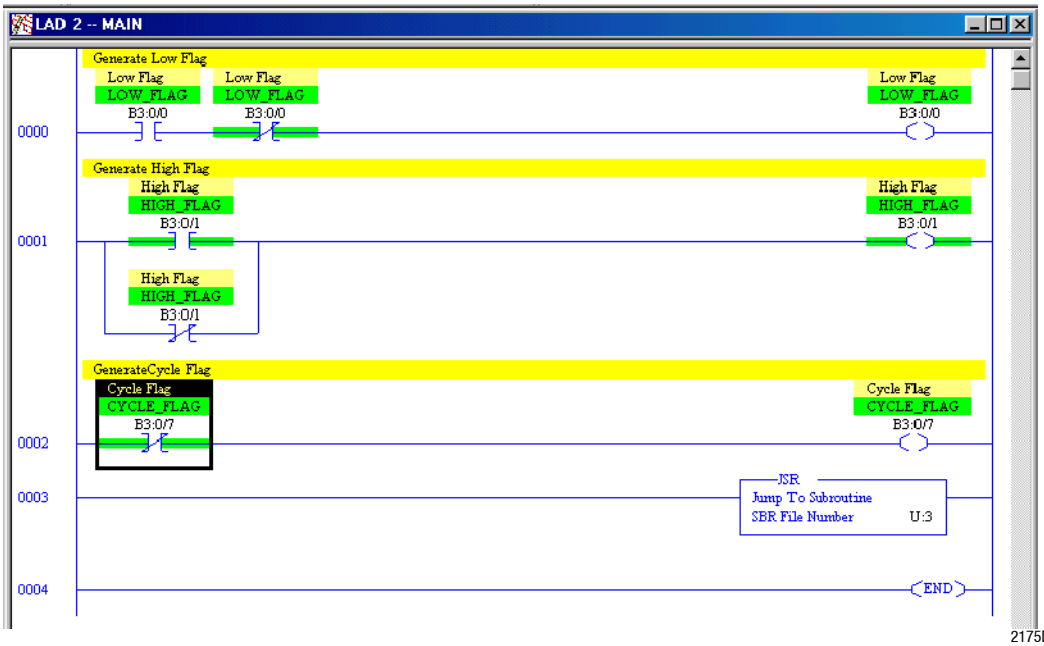
2175DeN098

Program component nesting:



14.3.1 LAD 2-Main

- Program basis
- The program is in first position of the processing table



2175DeN101

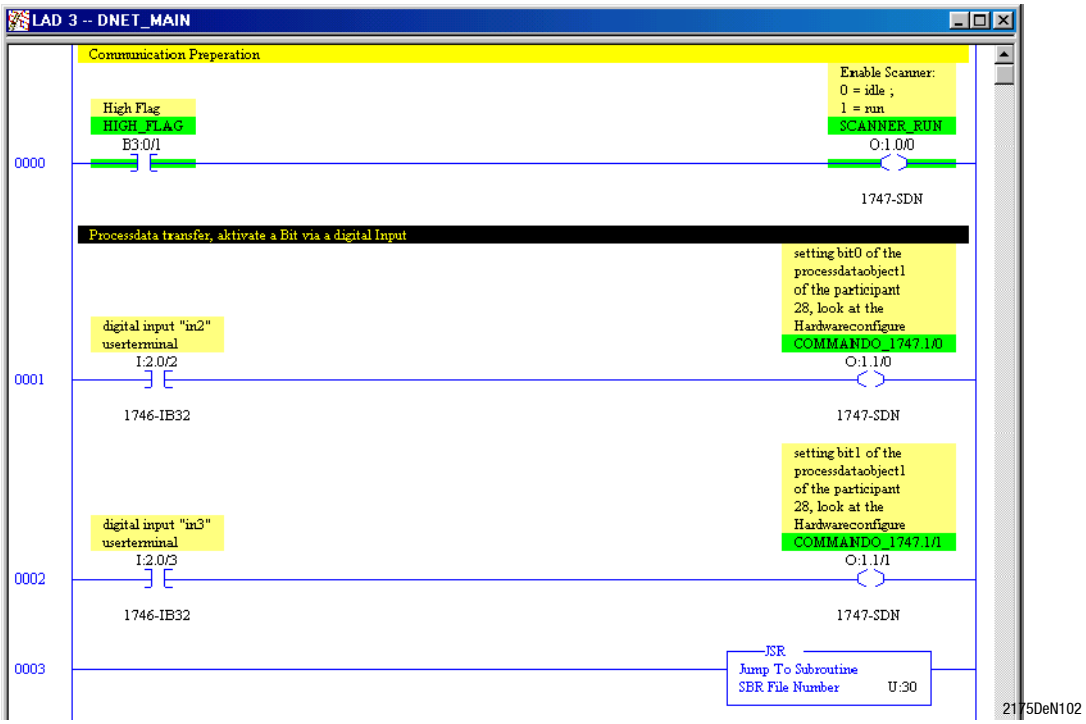
Programmed marker bits:

- Rung 0000 = Zero marker bit always defined as zero
- Rung 0001 = One marker bit always defined as one
- Rung 0002 = Cycle marker bit toggles after every cycle
- Rung 0003 Jump to subroutine 3, change without any conditions

14.3.2 LAD 3 - DNET_MAIN

The 1747 scanner is activated here.

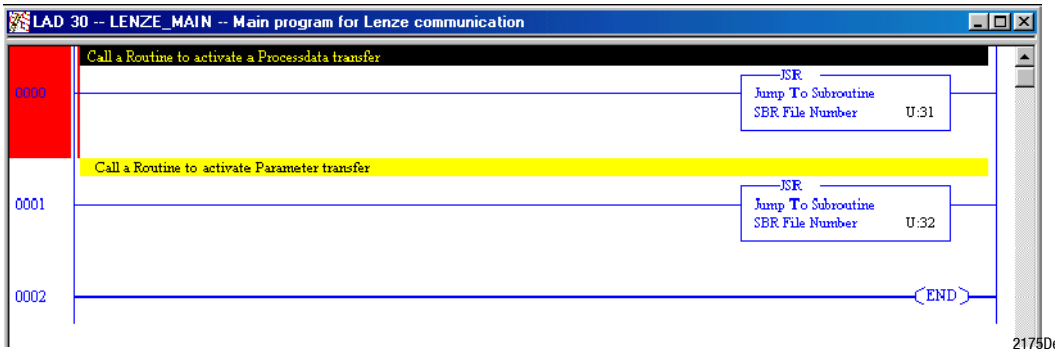
Two bits of a DeviceNet device are activated via digital inputs.



- Rung 0000 = Scanner enable via the ”One marker bit” (generated in LAD 2). The scanner is always in “Run” status
- Rung 0001: Digital input 2 sets bit 0 of the AIF-CTRL of the controller
- Rung 0002: Digital input 3 sets bit 1 of the AIF-CTRL of the controllers
- Rung 0003: Jump to subroutine 30, change without considering any conditions

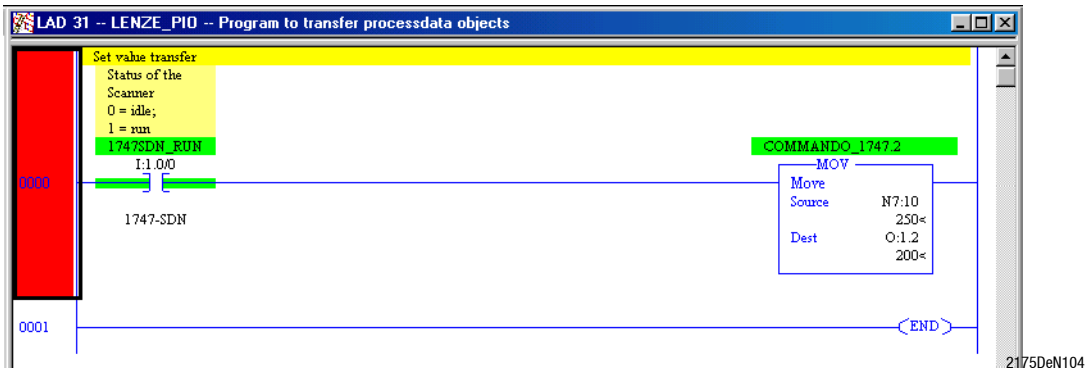
14.3.3 LAD30 - LENZE_MAIN

Jump to routines 31 and 32



14.3.4 LAD31 - LENZE_PIO

Sending of the controller setpoint



The example describes the writing of the speed setpoint to the 8200 vector frequency inverter.

The speed setpoint is normalised with $24000_{dec} \equiv 480Hz$.

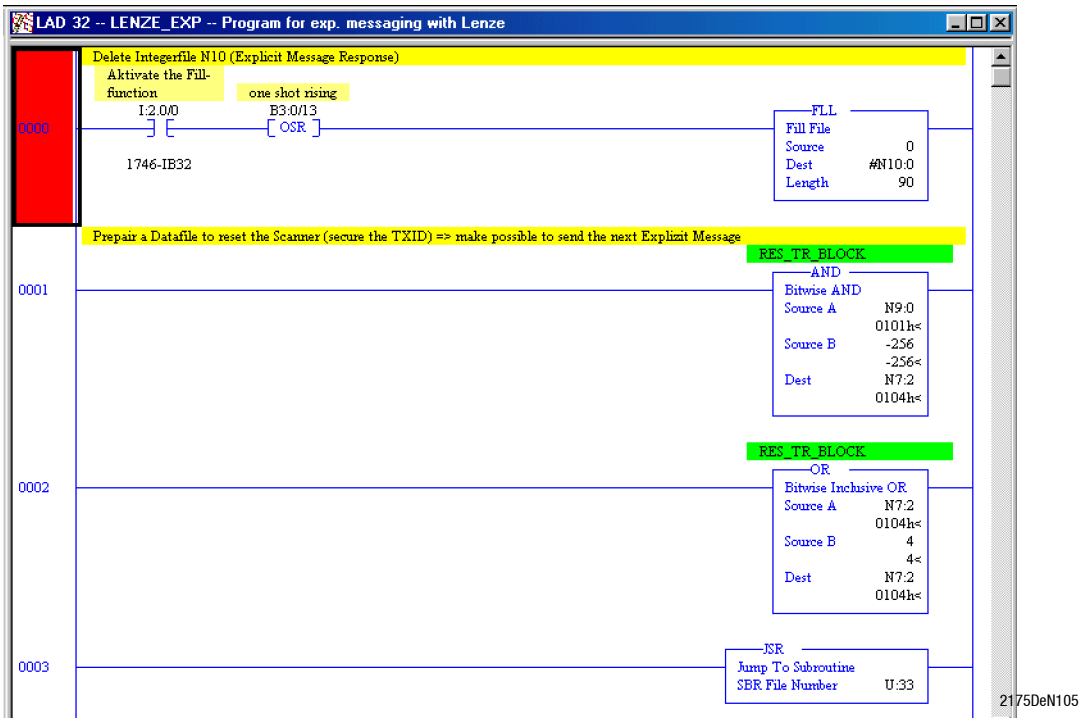
The value written to PCD2 (O:1.2) is = 500_{dec} and corresponds to a field frequency of 10 Hz.

Note!

Process data normalisation is described in detail in the corresponding Operating Instructions.

14.3.5 LAD32 - LENZE_EXP

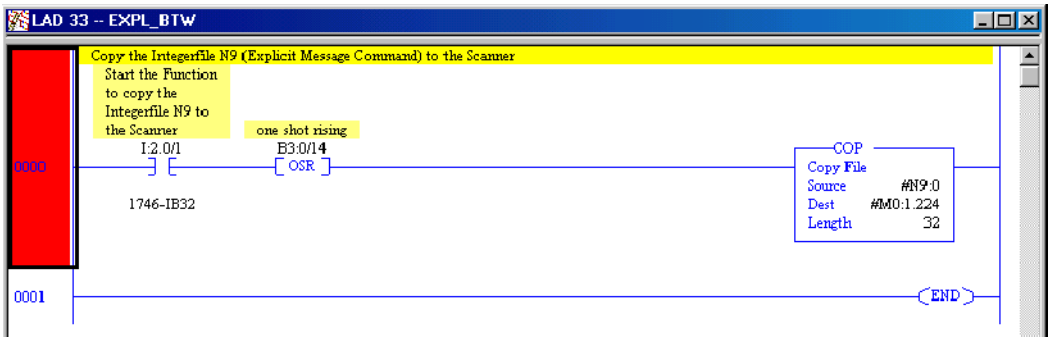
Preparations for explicit messages



- Rung 0000: Data area N10:0 is set zero. This data area is used to save input data of explicit messages. Here the order is activated via a digital input.
- Rung 0001: The order number (TXID) of an explicit message of the last order is filtered and saved in data area N7:2.
- Rung 0002: Data area N7:2 is consolidated ”Scanner reset”. Thus the data area N7:2 contains order number (TXID) and reset command (command = 04) for the scanner to be able to send the next explicit message.
- Rung 0003: Jump to subroutine 33, change without any conditions
- Rung 0004: Jump to subroutine 34, change without any conditions

14.3.6 LAD33 - EXPL_BTW

Explicit message output data



2175DeN106

Bit I:2.0/1 sets the copying command “COP”. The data are copied from the data word N9:0 to the scanner DeviceNet output area M0:1 and sent to the DeviceNet device.

The signal for starting the copying process is sent via a digital input. The function “OSR” converts the input signal into a pulse (pulse time: one cyle).

The data sent from the scanner as explicit message are prepared in data word N9:0 before they are copied as order to the scanner DeviceNet output area M0:1.

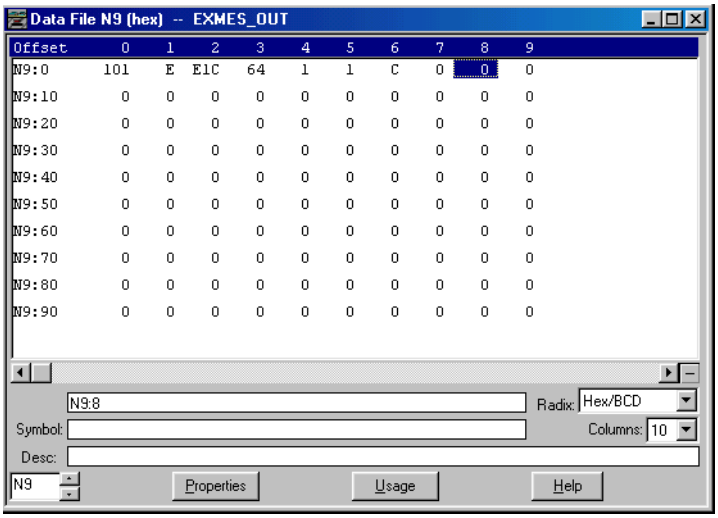
Explicit message data contents

Example for parameter data transfer of code L-C0012 = 000C$_{hex}$:

Word	High byte			Low byte		
0	TXID	01	Order number	Command	01 04	Execute block Clear response buffer
1	Port	00	with SLC only 0	Size	0E	User data size in byte
2	Service	0E 10	Get attribute single Set attribute single	MAC ID		DeviceNet node address
3	Class	00	(for Lenze codes)	Class	64	(for Lenze codes)
4	Instance	00	(for Lenze codes)	Instance	01	(for Lenze codes)
5	Attribute	00	(for Lenze codes)	Âttribute	01	(for Lenze codes)
6	L code	00		L code	0C	
7		00		Subcode	00	

Data area N9:0 is filled as follows:

N9:0	0101	TXID + command	Order number = 01 + command = 01 **Execute block**
N9:1	000E	Port + size	Port = 00 + size = E = 14 byte
N9:2	0E1C	Service + Mac ID	Service = 0E = Get single attribute + Mac ID = 1C = 28 node address
N9:3	0064	Class	Class = 0064 = 100 with Lenze controllers
N9:4	0001	Instance	Instance = 0001 with Lenze controllers
N9:5	0001	Attribute	Attribute = 0001 with Lenze controllers
N9:6	000C	L code	L code = 000C = L-C0012 (acceleration time)
N9:7	0000	Subcode	No subcode



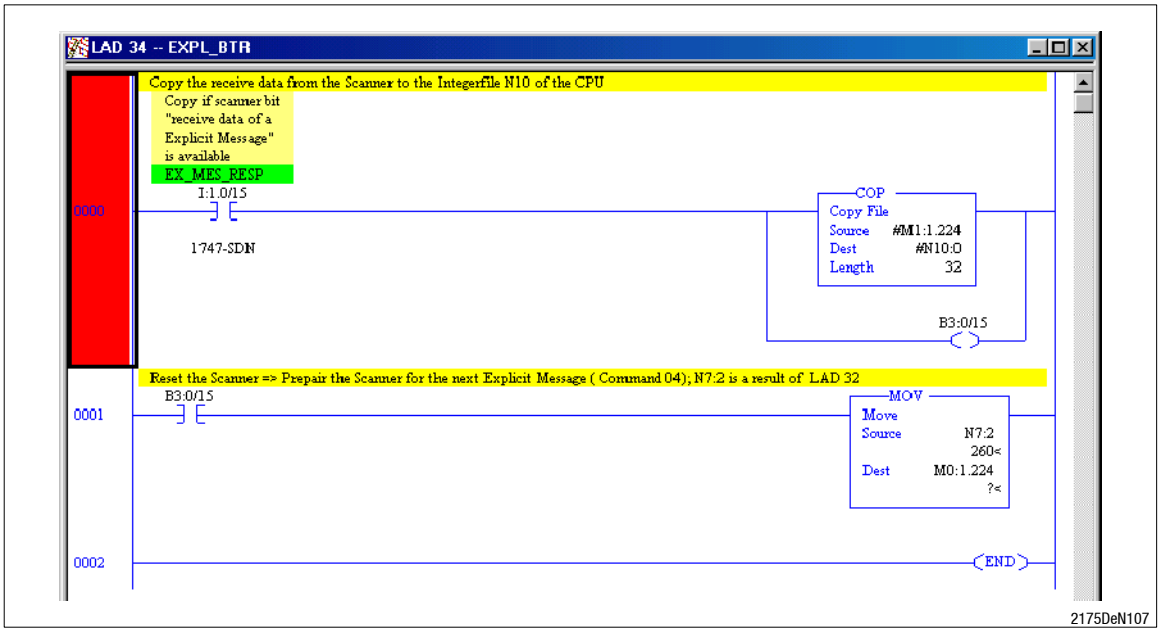
2175DeN108

Fig. 14-1 Data display in “Data File N9”

The order response is read from the subroutine LAD 34 (see (📖 14-28))

14.3.7 LAD34 - EXPL_BTR

The scanner content is read.



Scanner status bit 15 indicates that a response is available in scanner area M1:1. The scanner status bit activates the copy command and sets a marker bit.

The marker bit activates the scanner reset to ensure the next order.

The scanner input area is copied to the data area N10:0 with 32 words, i.e. the data received from the scanner are saved. The data can be processed from the data word N10:0.

The scanner must be reset for the next order.

This is achieved by writing the reset command for the corresponding order and TXID, e.g. 0104_{hex}. The order is reset with TXID 01.

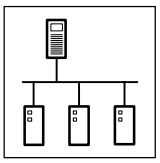
Word	High byte		Low byte	
0	TXID = **01**	Order No. (e.g. 01)	Command = **04**	01= Execute block 04= Clear response buffer

The read order response in N10:0 has the following structure:

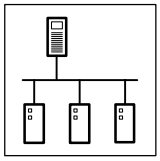
Word	High byte		Low byte	
0	TXID	01 (order No.)	Status	01 = Transaction successful
1	Port	00 (with SLC only 0)	Size	0E (user data size [byte])
2	Service		MAC ID	DeviceNet node address
3	L-code	00	L-code	0C
4		00	Subcode	00
5	Value	LOW word	Value	LOW word
6	Value	HIGH word	Value	HIGH word

Response to write request of code L-C0012 = 2.5sec:

N10:0	0101	TXID + status	01 (order No.) + 01(status): Transaction successful
N10:1	0008	Port + size	00 (port) + 08 (size, 8 bytes user data)
N10:2	8E1C	Service + Mac ID	8E (service) + 1C (Mac ID): 28 (node address)
N10:3	000C	L code	000C: L-C0012 (acceleration time)
N10:4	0000	Subcode	No subcode
N10:5	61A8	LOW word value	25000 ≡ 2.5 sec
N10:6	0000	HIGH word value	0000: End of user data

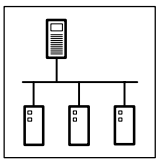


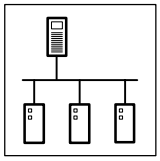
15 Table of keywords



DeviceNet

Table of keywords





DeviceNet

Table of keywords