

Lenze

Manual

IEC 61131-3

inside

***Global Drive
PLC Developer Studio***



Global Drive

Function library

Lenze8200Drive.lib

The function library **Lenze8200Drive.lib** can be used for the following Lenze PLCs:

	Type	from hardware version	from software version
9300 Servo PLC	EVS93XX-xT	6A	6.0
Drive PLC	EPL10200	1A	6.0
ECSxA	ECSxAxxx	1C	7.0

Important note:

The software is supplied to the user as described in this document. Any risks resulting from its quality or use remain the responsibility of the user. The user must provide all safety measures protecting against possible maloperation.

We do not take any liability for direct or indirect damage, e.g. profit loss, order loss or any loss regarding business.

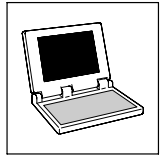
© 2002 Lenze Drive Systems GmbH

No part of this documentation may be copied or made available to third parties without the explicit written approval of Lenze Drive Systems GmbH.

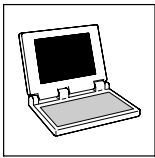
All information given in this documentation has been carefully selected and tested for compliance with the hardware and software described. Nevertheless, discrepancies cannot be ruled out. We do not accept any responsibility or liability for any damage that may occur. Required corrections will be included in updates of this documentation.

All product names mentioned in this documentation are trademarks of the corresponding owners.

Version 1.0 03/2005

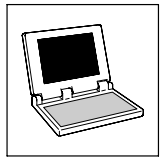


1 Preface and general information	1-1
1.1 About this Manual	1-1
1.1.1 Conventions used in this Manual	1-1
1.1.2 Layout of the descriptions	1-2
1.1.3 Pictographs used in this Manual	1-2
1.1.4 Terminology used	1-2
1.2 Version identifiers of the function library	1-3
2 Introduction	2-1
2.1 General	2-1
2.2 Frequency inverter parameterisation	2-2
2.2.1 Assigning the system bus node addresses to the frequency inverters	2-3
2.2.2 Function block activation	2-3
2.3 Process data exchange with the frequency inverter	2-4
3 Functions/function blocks	3-1
3.1 L_8200Parameter - Initialisation of codes	3-2
3.2 L_8200Initialization - Frequency inverter initialisation	3-7
3.3 L_8200DataControl - Send/receive process data	3-10
3.4 L_8200CtrlWord - Selection of boolean control signals	3-15
3.5 L_8200StatusWord - Output of boolean status signals	3-16



Function library Lenze8200Drive.lib

Contents



1 Preface and general information

1.1 About this Manual

This Manual contains information about function library **Lenze8200Drive.lib** for the **Drive PLC Developer Studio**.

- The function blocks of function library **Lenze8200Drive.lib** can be used to exchange data between the PLC and one or more frequency inverters of the 8200 series via the system bus (CAN).
- It is possible to assign up to 45 frequency inverter parameters (codes) from the PLC and cyclically exchange process data with the frequency inverter.

1.1.1 Conventions used in this Manual

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

Variable identifiers

are written in italics in the explanation:

- "Use *bReset*..."



Tip!

Information about the conventions used for the variables of the Lenze system blocks, function blocks and functions can be found in the appendix of the DDS online documentation "Introduction into IEC61131-3 programming". The conventions ensure universal and uniform labelling and make reading the PLC program easier.

Lenze functions/function blocks

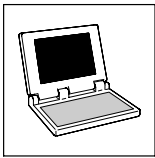
can be recognized by their names. They always begin with an "L_":

- "With the function **L_TBConvBitsToByte**..."

Program listings

are written in "Courier", keywords are printed in bold:

- " **IF** (ReturnValue < 0) **THEN**..."



Function library Lenze8200Drive.lib

Preface and general information

1.1.2 Layout of the descriptions

All function/function block and system block descriptions contained in this Manual have the same structure:

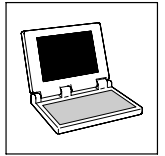
	Function	Function block (FB)/ System block (SB)
①	Headline stating the function and the function identifier	
②	Declaration of the function: • Data type of the return value • Function identifier • List of transfer parameters	-
③	Short description of the most important properties	
④	Function chart including all associated variables • Transfer parameters • Return value	FB/SB chart including all associated variables • Input variables • Output variables
⑤	Table giving information about the transfer parameters: • Identifier • Data type • Possible settings • Info	Table giving information about the input and output variables: • Identifier • Data type • Type of variable • Possible settings • Info
⑥	Table giving information about the return value: • Data type of the return value • Possible return values and their meaning:	-
⑦	Additional information (Notes, tips, application examples, etc.)	

1.1.3 Pictographs used in this Manual

	Use of pictographs	Signal words	
Warning of material damage		Stop!	Warns of potential damage to material . Possible consequences if disregarded: Damage to the controller/drive system or its environment.
Other notes		Tip! Note!	Indicates a tip or note.

1.1.4 Terminology used

Term	In the following text used for
DDS	Drive PLC Developer Studio
FB	Function block
GDC	Global Drive Control (parameterization program from Lenze)
Parameter codes	Codes for setting the functionality of a function block
PLC	9300 Servo PLC - Drive PLC - ECSxA "Application" axis module
SB	System block



1.2 Version identifiers of the function library

The version of the function library can be found under the global constant `C_w[Function library name]Version`.

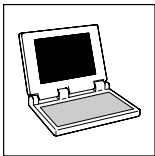
Version identifiers as of PLC software version 7.x:

Constant	Meaning	Example value
<code>C_w[FunctionLibraryName]VersionER</code>	External Release	01
<code>C_w[FunctionLibraryName]VersionEL</code>	External Level	05
<code>C_w[FunctionLibraryName]VersionIR</code>	Internal Release	00
<code>C_w[FunctionLibraryName]VersionBN</code>	Build No.	00

Version: 01 05 00 00

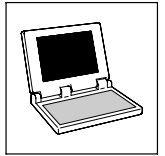
The value of this constant is a hexadecimal code.

- In the example, "01050000" stands for version "1.05".



Function library Lenze8200Drive.lib

Preface and general information



2 Introduction

2.1 General

Integrate function library **Lenze8200Drive.lib** in your PLC program to make frequency inverter control easier.

Two function blocks are required for this:

1. A function block for the parameterisation of the frequency inverter.
2. Another function block for process data exchange between PLC and frequency inverter.

The parameters can be set via the PLC program. It is possible to select up to 45 parameters.

Parameterisation concepts

The following parameterisation concepts are supported:

- Parameterisation through direct instructions in the FBs
- Parameterisation by means of codes ("online" parameterisation possible)



Note!

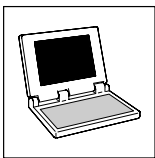
The **frequency inverter** must meet the following conditions for frequency inverter control from the PLC:

- The factory settings have been loaded.
(For the 8200vector frequency inverter, e. g. under code C0002)
- The system bus node address is correct.
(For the 8200vector frequency inverter, e. g. under code C0350)
- The new communication settings of the system bus interface have been accepted by a Reset-Node.
(For the 8200vector frequency inverter, e. g. under code C0358)



Tip!

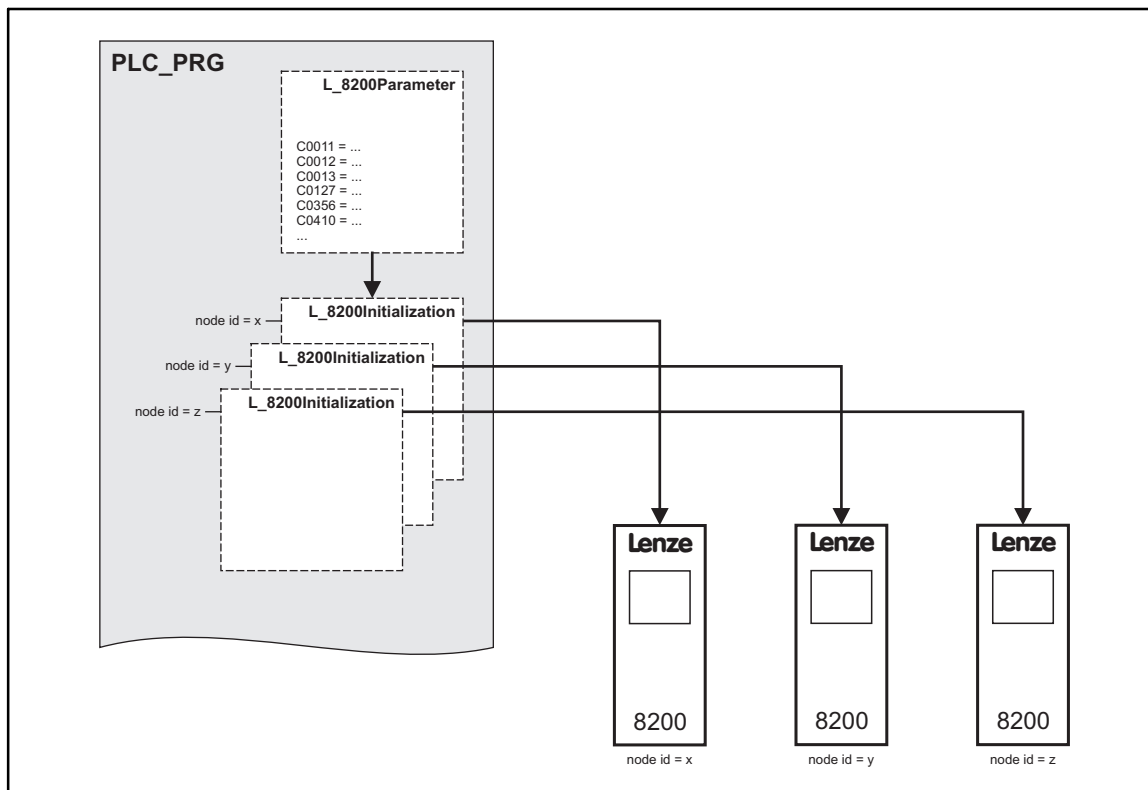
Detailed information about the system bus (CAN) and function library **LenzeCanDrv.lib** can be found in the Manual "System bus (CAN) for Lenze PLC devices".

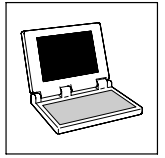


2.2 Frequency inverter parameterisation

When the program is started, the parameter data shall be written **only once** during initialisation to the frequency inverter. The FBs **L_8200Parameter** and **L_8200Initialization** therefore have to be called in the system task **PLC_PRG**.

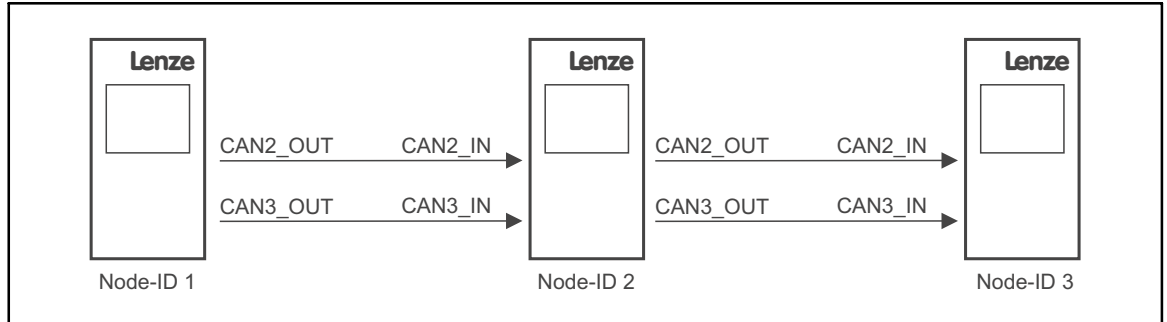
- The FB **L_8200Parameter** is used to initialise the codes with the corresponding values.
- After this, the FB **L_8200Initialization** is used to transfer the parameters to the individual frequency inverters.



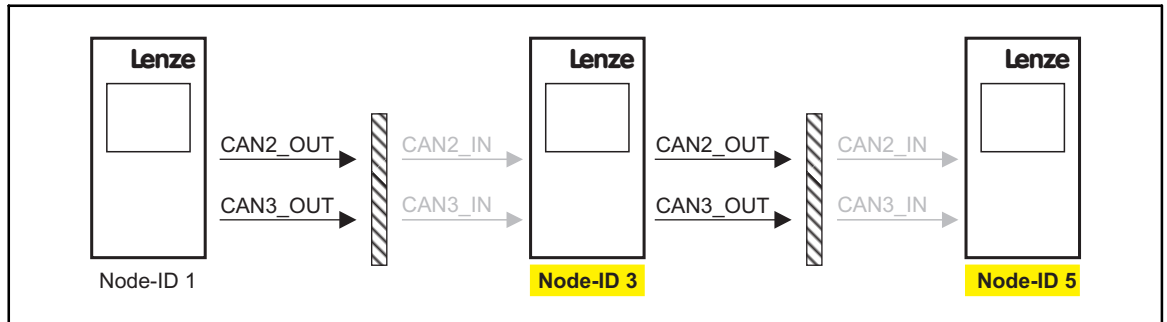


2.2.1 Assigning the system bus node addresses to the frequency inverters

If the node addresses are assigned in continuous, ascending order to the Lenze controllers, the default setting of the event-controlled data object identifiers (CAN2_IO/CAN3_IO) allows communication between the controllers:



If you want to parameterise a group of frequency inverters with different values you have to skip at least one node address between the inverters to interrupt the automatic data transfer:



- In this way, you can parameterise a maximum of 32 frequency inverters (with node address 1 - 3 - 5 - 7 - ... - 63).



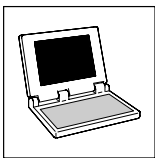
Note!

New node addresses will only become effective after a Reset-node. (For the 8200vector frequency inverter, e. g. under code C0358)

2.2.2 Function block activation

The FB **L_8200Initialization** must be called cyclically because the data are internally transmitted to the frequency inverter via the FB **L_ParWrite**.

- Depending on the amount of data to be transmitted, the data transfer may take several cycles.
- If the data transfer has been successful, the message output *bParDone* will be set to TRUE.
- When the data transfer has been completed (*bParDone* = TRUE), the parameterisation is complete and the FBs **L_8200Parameter** and **L_8200Initialization** need not be called again.



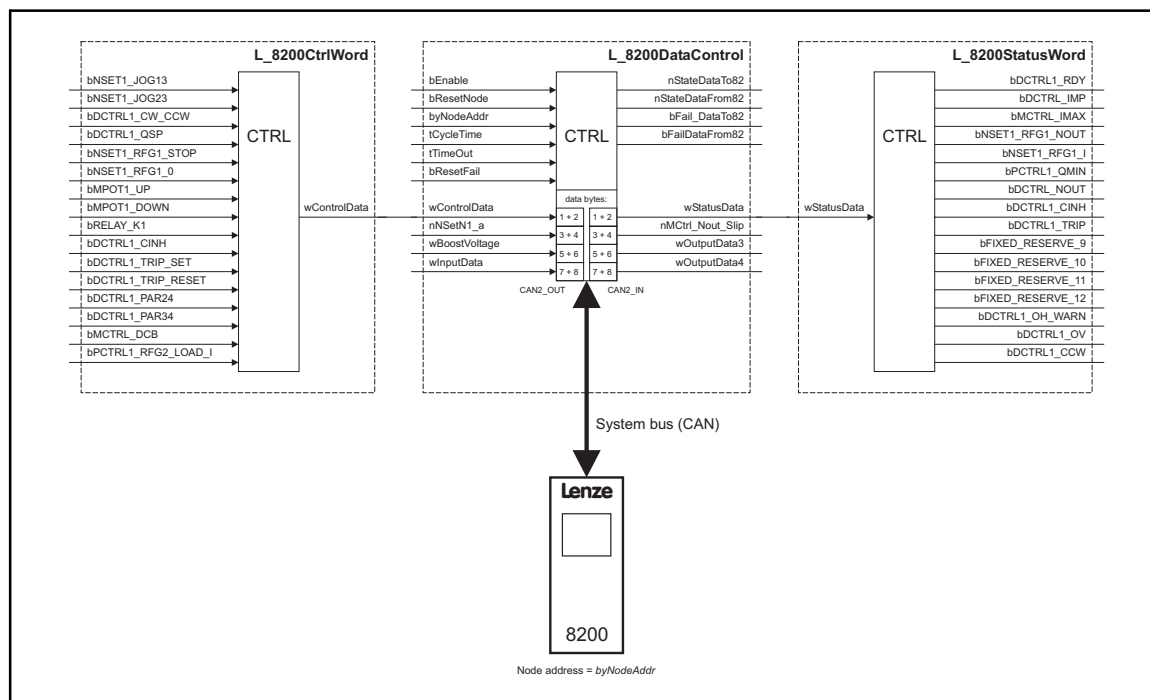
Function library *Lenze8200Drive.lib*

Introduction

2.3 Process data exchange with the frequency inverter

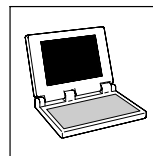
Unlike the parameter data, the process data must be cyclically exchanged between PLC and frequency inverter to update input and output information.

- The FB **L_8200CtrlWord** is used to select the control word to be transmitted as boolean control signals. (▢ 3-15)
- The FB **L_8200DataControl** is used for the actual process data exchange with the corresponding frequency inverter. (▢ 3-10)
- The FB **L_8200StatusWord** is used to divide the receive status word for evaluation and further processing into individual boolean status signals. (▢ 3-16)

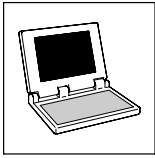


Note!

The task cycle time depends on the number of objects used.



3 Functions/function blocks



Function library Lenze8200Drive.lib

Functions/function blocks

L_8200Parameter - Initialisation of codes

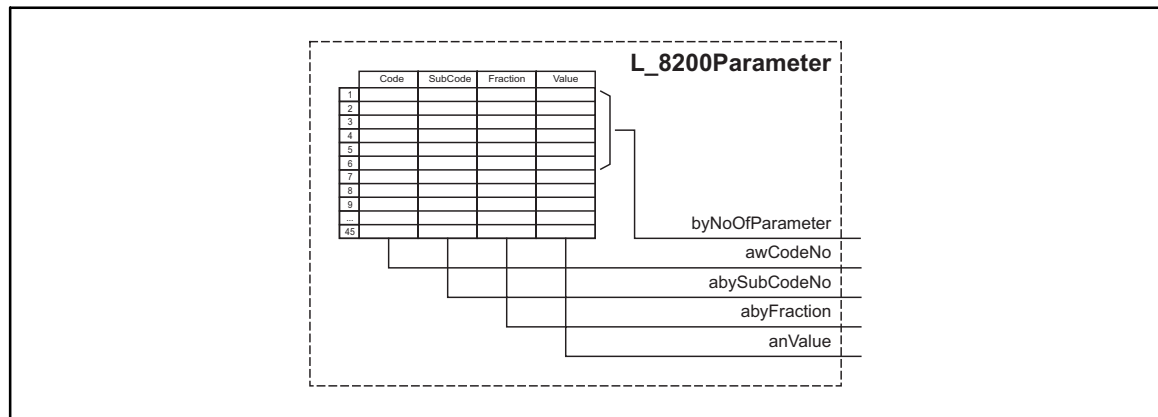
3.1 L_8200Parameter - Initialisation of codes

Function block

With this FB you can parameterise up to 45 parameters (codes) in the frequency inverter.

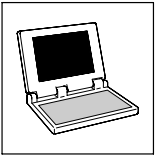
The following parameterisation concepts are supported:

- Parameterisation through direct instructions in the FBs
- Parameterisation by means of codes ("online" parameterisation possible)



FB call in:	☒ Cyclic task (PLC_PRG)	☒ Time-controlled task (INTERVAL)	☒ Event-controlled task (EVENT)	☐ Interrupt task
--------------------	---	---	---	---

Inputs	Data type	Information/possible settings
byNoOfParameter	Byte	Number of parameters to be transmitted (max. 45)
awCodeNo	Array of Word	Number of the code to be transmitted as parameter 1..45.
abySubCodeNo	Byte array	Number of the subcode to be transmitted as parameter 1..45.
abyFraction	Byte array	Number of decimal places for parameters 1..45 • See also FB description L_ParWrite in the Manual Function library LenzeDrive.lib.
anValue	Array of integers	Value of the code to be transmitted as parameter 1..45.



Concept I: Parameterisation through direct instructions in the FBs

For this, use the FB **L_8200Parameter** as template and create a new function block in which the parameter data are selected by direct assignments.



Note!

Please observe that FB changes (e. g. new assignments) require a new compilation and program download to the PLC.

- If you want to change the parameters "online" via the Parameter Monitor or keypad, use concept II ("Parameterisation by means of codes").



Tip!

The FB **L_8200Parameter** already includes **commented** assignments for all parameters relevant for frequency inverter control.

- The parameters are listed in the table on the next page.
- The names of the inputs and outputs of the FBs **L_8200CtrlWord** and **L_8200StatusWord** have been adapted to the selected parameters.
- If you want to use the pre-assignments, simply copy the assignments to the new FB and remove the comment characters so that the assignments will become effective.
- Of course, it is also possible to adapt the assignments to your application.

Proceed as follows:

1. Create a new function block.
2. Use the Library Manager and copy the source code of the FB **L_8200Parameter** to the new function block.
3. Remove the comment characters around the assignments to accept the pre-assignments or adapt the assignments to your application.

```
FUNCTION_BLOCK Parameter82Device1
VAR_OUTPUT
    byNoOfParameter      : BYTE;
    awCodeNo             : ARRAY [1..45] OF WORD;
    abySubCodeNo         : ARRAY [1..45] OF BYTE;
    abyFraction          : ARRAY [1..45] OF BYTE;
    anValue              : ARRAY [1..45] OF INT;
END_VAR

(* max. frequency *)
awCodeNo[1]:=11; abySubCodeNo[1]:=0; anValue[1]:=5000; abyFraction[1]:=2;

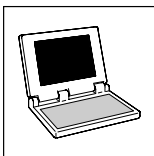
(* acceleration time Tir *)
awCodeNr[2]:=12; abySubCodeNo[2]:=0; anValue[2]:=0010; abyFraction[2]:=2;

(* deceleration time Tif *)
awCodeNo[3]:=13; abySubCodeNr[3]:=0; anValue[3]:=0010; abyFraction[3]:=2;

(* standardization for C0011 => 16384 *)
awCodeNo[4]:=127; abySubCodeNo[4]:=0; anValue[4]:=1; abyFraction[4]:=0;
...
```

Fig. 3-1

Example: Source code (extract) of a new FB based on the FB **L_8200Parameter**



Function library Lenze8200Drive.lib

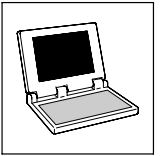
Functions/function blocks

L_8200Parameter - Initialisation of codes

No.	Code	Subcode	Setting	Info/meaning
1	C0011	0	50	Maximum output frequency = 50 Hz (default setting)
2	C0012	0	0.1	Main setpoint acceleration time; Tir = 0.1 s (default setting)
3	C0013	0	0.1	Main setpoint deceleration time; Tif = 0.1 s (default setting)
4	C0127	0	1	Normalised setpoint selection
5	C0356	2	50	CAN2-OUT cycle time = 50 ms
j6	C0410	1	70	NSET1-JOG1/3 ⇔ CAN-IN2-W1-BIT0
7	C0410	2	71	NSET1-JOG2/3 ⇔ CAN-IN2-W1-BIT1
8	C0410	3	72	DCTRL1-CW/CCW ⇔ CAN-IN2-W1-BIT2
9	C0410	4	73	DCTRL1-QSP ⇔ CAN-IN2-W1-BIT3
10	C0410	5	74	NSET1-RFG1-STOP ⇔ CAN-IN2-W1-BIT4
11	C0410	j6	75	NSET1-RFG1-0 ⇔ CAN-IN2-W1-BIT5
12	C0410	7	76	MPOT1-UP ⇔ CAN-IN2-W1-BIT6
13	C0410	8	77	MPOT1-DOWN ⇔ CAN-IN2-W1-BIT7
14	C0415	1	108	RELAY-K1 ⇔ CAN-IN2-W1-BIT8
15	C0410	10	79	DCTRL1-CINH ⇔ CAN-IN2-W1-BIT9
16	C0410	11	80	DCTRL1-TRIP-SET ⇔ CAN-IN2-W1-BIT10
17	C0410	12	81	DCTRL1-TRIP-RESET ⇔ CAN-IN2-W1-BIT11
18	C0410	13	82	DCTRL1-PAR 2/4 ⇔ CAN-IN2-W1-BIT12
19	C0410	14	83	DCTRL1-PAR 3/4 ⇔ CAN-IN2-W1-BIT13
20	C0410	15	84	MCTRL1-DCB ⇔ CAN-IN2-W1-BIT14
21	C0410	16	85	PCTRL1-RFG2-LOAD-I (A) ⇔ CAN-IN2-W1-BIT15
22	C0412	1	31	NSET1-N1 ⇔ CAN-IN2-W2
23	C0412	8	32	MCTRL1-VOLT-ADD ⇔ CAN-IN2-W3
24	C0418	1	16	CAN-OUT2-BIT0 ⇔ DCTRL1-RDY
25	C0418	2	2	CAN-OUT2-BIT1 ⇔ DCTRL1-IMP
26	C0418	3	3	CAN-OUT2-BIT2 ⇔ MCTRL1-IMAX
27	C0418	4	4	CAN-OUT2-BIT3 ⇔ DCTRL1-RFG1=NOUT
28	C0418	5	5	CAN-OUT2-BIT4 ⇔ NSET1-RFG1-I=0
29	C0418	j6	j6	CAN-OUT2-BIT5 ⇔ PCTRL1-QMIN
30	C0418	7	7	CAN-OUT2-BIT6 ⇔ DCTRL1-NOUT=0
31	C0418	8	8	CAN-OUT2-BIT7 ⇔ DCTRL1-CINH
32	C0418	9	25	CAN-OUT2-BIT8 ⇔ DCTRL1-TRIP
33	C0418	10	9	CAN-OUT2-BIT9 ⇔ FIXED-RESERVE-9
34	C0418	11	10	CAN-OUT2-BIT10 ⇔ FIXED-RESERVE-10
35	C0418	12	11	CAN-OUT2-BIT11 ⇔ FIXED-RESERVE-11
36	C0418	13	12	CAN-OUT2-BIT12 ⇔ FIXED-RESERVE-12
37	C0418	14	13	CAN-OUT2-BIT13 ⇔ DCTRL1-OH-WARN
38	C0418	15	14	CAN-OUT2-BIT14 ⇔ DCTRL1-OV
39	C0418	16	15	CAN-OUT2-BIT15 ⇔ DCTRL1-CCW
40	C0421	8	0	CAN-OUT2-W2 ⇔ MCTRL1-NOUT+SLIP
41	C0003	0	1	Dynamic parameter set changes
42				not assigned
43				not assigned
44				not assigned
45				not assigned

Tab. 3-1

Parameter pre-assignment contained as a comment in the FB L_8200Parameter.



Concept II: Parameterisation by means of codes

If the FB outputs are assigned to user codes in the PLC, it is possible to select the corresponding frequency inverter parameters directly via the Parameter Monitor.



Note!

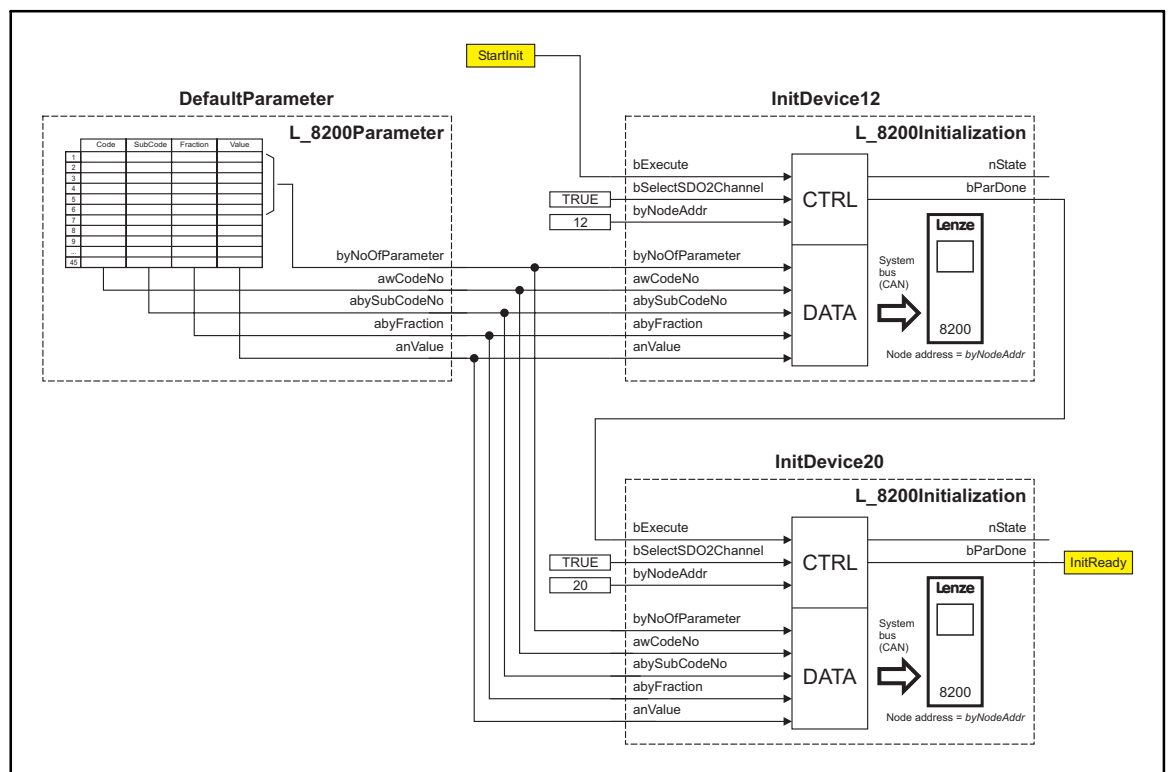
With this concept, it is possible to use an instance of the FBL **L_8200Parameter** instead of a copy. Please observe that all parameter values must be selected under the corresponding codes/subcodes before starting the program.

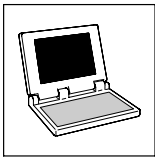
Proceed as follows:

1. Create an instance of the FB **L_8200Parameter**.
2. Use the **Instance Parameter Manager** to assign user codes to the outputs of the instance.
3. Compile the project to ensure that the codes/subcodes will be available for parameterisation in the **Parameter Monitor**.

Example I: Assignment of identical parameters to a number of frequency inverters

If you want to assign the same parameters to a number of frequency inverters, you only need one FB with the selected parameters.





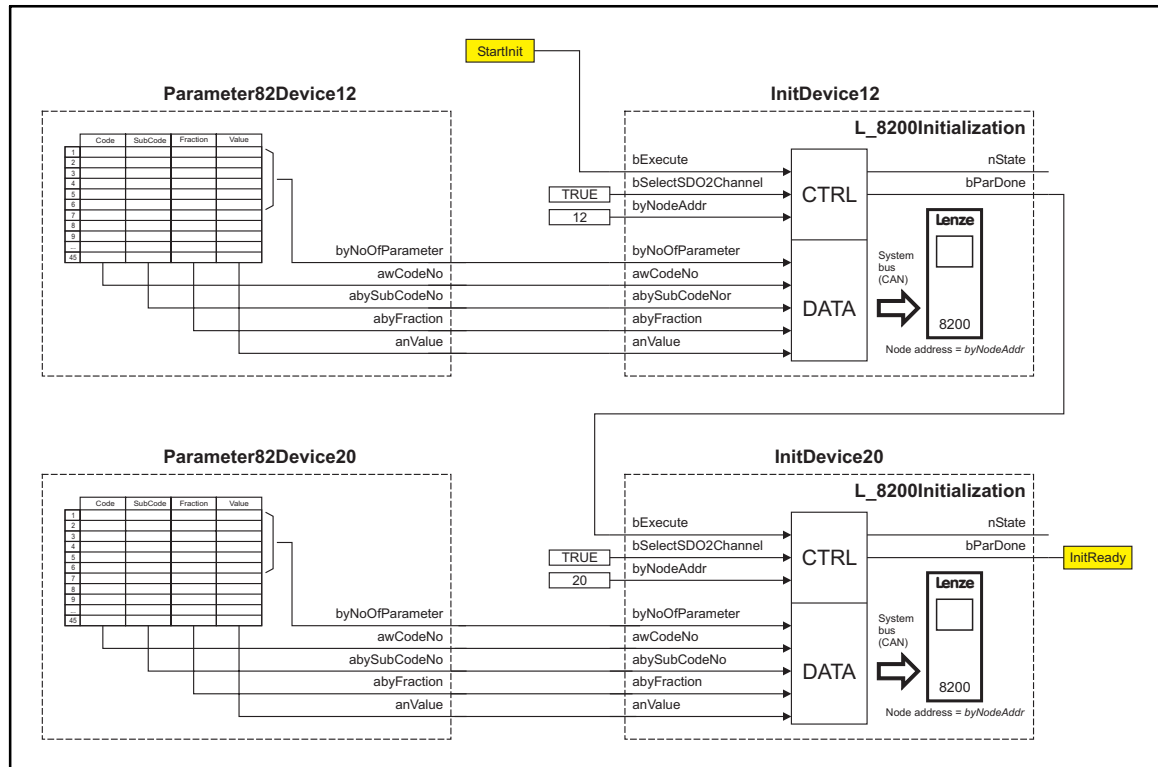
Function library Lenze8200Drive.lib

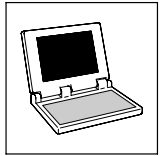
Functions/function blocks

L_8200Parameter - Initialisation of codes

Example II: Assignment of different parameters to a number of frequency inverters

If you want to assign different parameters to a number of frequency inverters, you need a corresponding number of FBs.





3.2 L_8200Initialization - Frequency inverter initialisation

Function block

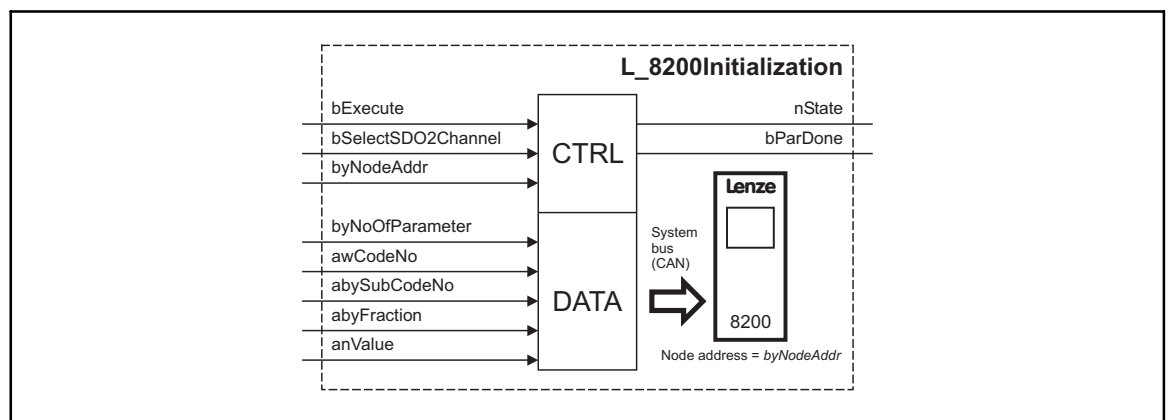
This FB is used to initialise the frequency inverter with the parameters which were previously defined by an instance or copy of the FB **L_8200Parameter**.



Note!

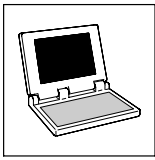
The FB must be called cyclically because the data are internally transmitted to the frequency inverter via the FBL **ParWrite**.

- Depending on the amount of data to be transmitted, the data transfer may take several cycles.
- If the data transfer has been successful, the message output **bParDone** will be set to TRUE.
- When the data transfer has been completed (**bParDone** = TRUE), the parameterisation is complete and the FBs **L_8200Parameter** and **L_8200Initialization** need not be called again.



FB call in:	☐ Cyclic task (PLC_PRG)	☒ Time-controlled task (INTERVAL)	☐ Event-controlled task (EVENT)	☐ Interrupt task
-------------	--	---	--	---

Inputs	Data type	Information/possible settings
bExecute	Bool	Start data transfer to the frequency inverter FALSE → TRUE Start data transfer
bSelectSDOChannel	Bool	Select the parameter channel for the frequency inverter FALSE SDO 1 TRUE SDO 2
byNodeAddr	Byte	Device address (1 ... 63) of the frequency inverter to be initialised. • See also chapter 2.2.1. (2-3)
byNoOfParameter	Byte	Transfer of the number of parameters to be transmitted
awCodeNo	Array of Word	Code transfer
abySubCodeNo	Byte array	Subcode transfer
abyFraction	Byte array	Transfer of decimal places • See also FB description L_ParWrite in the Manual Function library LenzeDrive.lib .
anValue	Array of integers	Transfer of code values



Function library **Lenze8200Drive.lib**

Functions/function blocks

L_8200Initialization - Frequency inverter initialisation

Outputs	Data type	Value/meaning
nState	Integer	Status
		Status messages:
		0 Data were transferred, frequency inverter has been initialised.
		1 Data transfer is active.
		Error messages:
		2001 An error occurred during bus transfer.
		2002 Time-out occurred during data transfer.
		2004 Access denied - wrong target system: Subindex does not exist.
		2005 Access denied - wrong target system: Index does not exist.
		2013 Access denied - wrong target system: Invalid range of values.
		2111 Access denied - wrong target system: In excess of 4 decimal places specified or no fixed-comma-format index.
		2117 Communication channel does not exist. • Check PLC code C2118 (transmission channel for CAN parameter communication).
bParDone	Bool	Message "Data transfer has been completed"
		TRUE Data transfer has been successful.

Example: Frequency inverter initialisation with preselected parameters

In the following example, a frequency inverter with the system bus node address 12 is initialised with the parameters preselected in the FB **L_8200Parameter**. Afterwards, another frequency inverter with the system bus node address 20 is initialised with the same parameters.



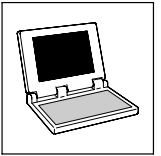
Note!

Before communication with the PLC, carry out the following steps:

1. Load the factory settings.
(For the 8200vector frequency inverter, e. g. under code C0002)
2. Select the correct system bus node address.
(For the 8200vector frequency inverter, e. g. under code C0350)
– More information about the node address assignment can be found in chapter 2.2.1. (□ 2-3)
3. Carry out a reset node to accept the new communication settings of the system bus interface.
(For the 8200vector frequency inverter, e. g. under code C0358)

After this, carry out the following steps in the Drive PLC Developer Studio:

1. Use the Library Manager to integrate function libraries **Lenze8200Drive.lib**, **LenzeDrive.lib** and **LenzeCanDrv.lib** in your project.
2. Create a new FB called "DefaultParameter82" and copy the source code of the FB **L_8200Parameter** to the new FB **DefaultParameter82**.
3. Remove the comment characters around the assignments in the FB **DefaultParameter82** so that the assignments will become valid for the compiler.



4. Create two instances of the FB **L_8200Initialization**.
5. Integrate the FBs either directly in the main program **PLC_PRG** or in a program which is cyclically activated by **PLC_PRG** and combine the FBs as follows:

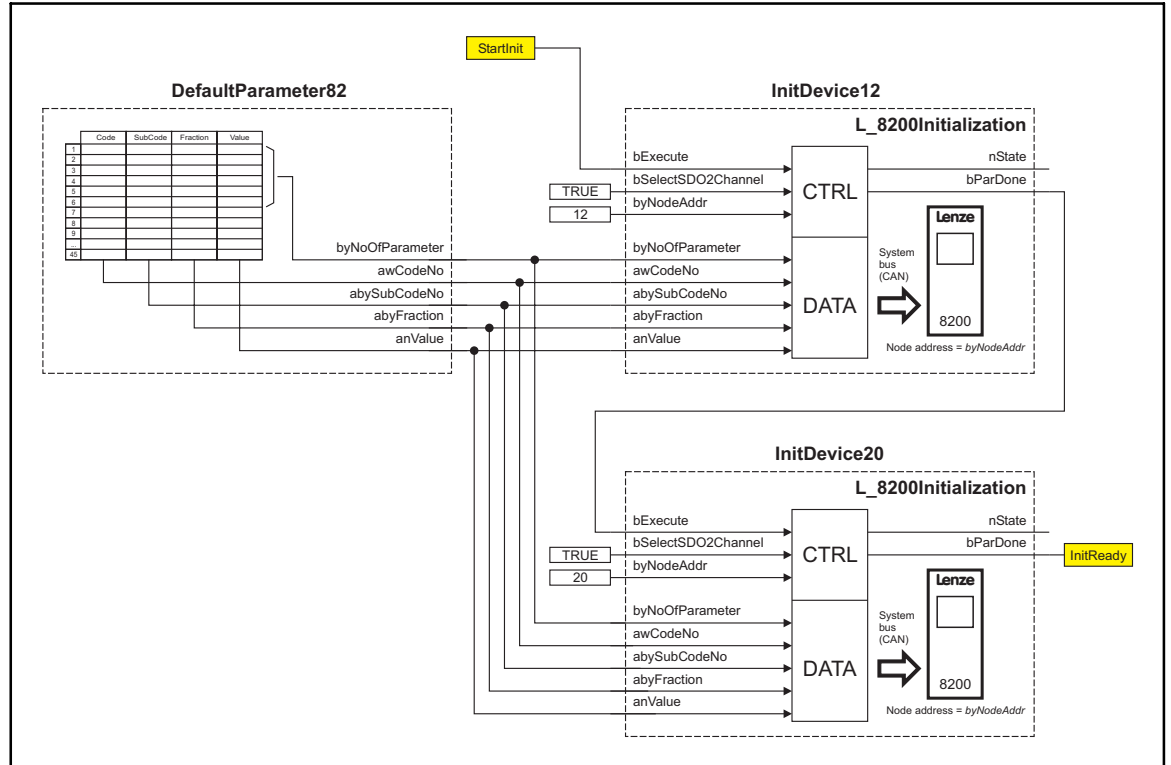
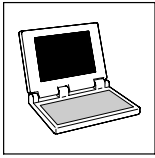


Fig. 3-2

Links between the FBs

Initialisation process

- Use the FB **DefaultParameter82** which is based on the FB **L_8200Parameter** and transmit the parameters to the instances of the FB **L_8200Initialization**.
- Set **StartInit** to TRUE to start the initialisation and transfer the data via parameter channel SDO2 to the frequency inverter with the node address 12.
- If all data were transmitted, the output **bParDone** of the instance "InitDevice12" will be set to TRUE. This is the start signal for the data transfer to the second frequency inverter with the node address 20.
- If the data were successfully transmitted to the second frequency inverter, the variable **InitReady** will be set to TRUE. This signal can, for instance, be used to start the process data transfer via the FB **L_8200DataControl**.
- After data transfer has been successful (**InitReady** = TRUE), the FB need not be activated again.



Function library *Lenze8200Drive.lib*

Functions/function blocks

L_8200DataControl - Send/receive process data

3.3

L_8200DataControl - Send/receive process data

Function block

This FB is used to exchange process data with the frequency inverter.

- The FB internally uses a free CAN send and receive object. It is therefore possible to transmit and receive 8 bytes of user data.

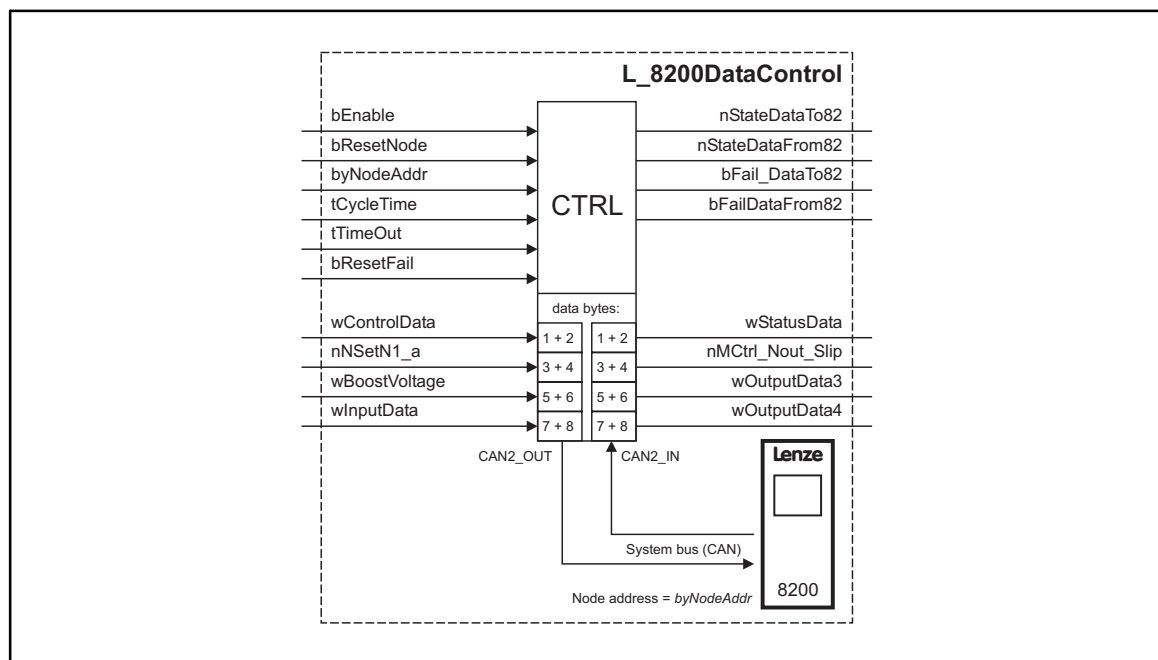


Note!

For the provision of free CAN send and receive objects, function library **LenzeCanDrv.lib** must be integrated in the project.

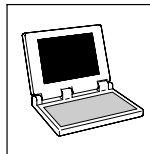
- For data exchange via the free CAN objects the driver for the free CAN objects must be initialised in a program (e. g. "PLC_ColdStart").

The FB **L_8200DataControl** must be called in a task-controlled program.



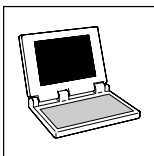
FB call in:	☐ Cyclic task (PLC_PRG)	☒ Time-controlled task (INTERVAL)	☐ Event-controlled task (EVENT)	☐ Interrupt task
--------------------	--	---	--	---

Inputs	Data type	Information/possible settings
bEnable	Bool	Start process data transfer FALSE → TRUE Start process data transfer.
bResetNode	Bool	Carry out a Reset-Node in the frequency inverter. • After a Reset-Node, the frequency inverter accepts the parameter settings which are relevant for communication and changes to "Operational". • The Reset-Node is only carried out in the frequency inverter with device address <i>byNodeAddr</i> . FALSE → TRUE Carry out Reset-Node.
byNodeAddr	Byte	Device address (1 ... 63) of the frequency inverter with which process data are to be exchanged.
tCycleTime	Time	Cycle time for sending the process data to the frequency inverter.
tTimeOut	Time	Monitoring time (time-out) for receiving the process data from the frequency inverter.
bResetFail	Bool	Reset message signals for error messages FALSE → TRUE The message signals <i>bFail_DataTo82</i> and <i>bFail_DataFrom82</i> are reset.



Inputs	Data type	Information/possible settings
wControlData	Word	Process data word (control word) to be sent to the frequency inverter.
nNSetN1_a	Integer	Speed setpoint selection for the frequency inverter in [Hz] Normalised to f_{max}, i. e. $16384 \equiv 100\% f_{max}$ (C0011) of the frequency inverter
wBoostVoltage	Word	U _{min} increase in the frequency inverter Normalised to U_r, i. e. $16384 \equiv 100\% U_r$ of the frequency inverter
wInputData	Word	Send bytes 7 + 8, freely assignable

Outputs	Data type	Value/meaning
nStateDataTo82	Integer	Status of the CAN send object
		Error messages:
		-300 Time-out, data transmitted by the frequency inverter are not within the monitoring time <i>tTimeOut</i> .
		-150 System bus (CAN) is not "Operational".
		-121 Wrong driver number for CAN driver. Check the initialisation of the CAN driver.
		-120 CAN driver currently not initialised. Check the initialisation of the CAN driver.
		-119 The transmit command memory is full. The transmit command could not be entered. Increase the activation cycle time of the FB.
		-118 There is no free CAN channel available.
		-12 The message identifier (COB-ID) is not within the permissible area (internal error).
		-11 The pointer <i>plOAdress</i> does not point to PLC RAM (internal error).
		Status messages:
		0 The process data were sent to the frequency inverter.
		1 The transmit command has not yet been completed and is still in the command memory.
		2 Data transfer is inactive. Frequency inverter parameterisation not yet complete or <i>bEnable</i> is set FALSE.
		10 Telegram longer than 8 bytes and limited to 8 bytes.

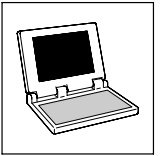


Function library Lenze8200Drive.lib

Functions/function blocks

L_8200DataControl - Send/receive process data

Outputs	Data type	Value/meaning
nStateDataFrom82	Integer	Status of the CAN receive object
		Error messages:
		-300 Time-out, data transmitted by the frequency inverter are not within the monitoring time <i>tTimeOut</i> .
		-150 System bus (CAN) is not "Operational".
		-121 Wrong driver number for CAN driver. • Check the initialisation of the CAN driver.
		-120 CAN driver currently not initialised. • Check the initialisation of the CAN driver.
		-12 The message identifier (COB-ID) is not within the permissible area (internal error).
		-11 The pointer <i>pIOAddress</i> does not point to PLC RAM (internal error).
		Status messages:
		0 The process data were received by the frequency inverter.
		2 Data transfer is inactive. • Frequency inverter parameterisation not yet complete or <i>bEnable</i> is set FALSE.
		10 Telegram longer than 8 bytes and limited to 8 bytes.
		200 Data in receive memory were overwritten. • Reduce the activation cycle of the FB or increase the data transmission cycle of the frequency inverter.
bFail_DataTo82	Bool	Message "Transmit command cancelled or not executed".
		TRUE Transmit command was cancelled or not executed. • The corresponding error number is output at <i>nStateDataTo82</i> .
bFail_DataFrom82	Bool	Message "Receive command cancelled or not executed".
		TRUE Receive command was cancelled or not executed. • The corresponding error number is output at <i>nStateDataFrom82</i> .
wStatusData	Word	Process data word (status word) sent by the frequency inverter.
nMCtrl_Nout_Slip	Integer	Actual frequency inverter speed (including slip) in [Hz].
wOutputData3	Word	Receive bytes 5 + 6, freely assignable
wOutputData3	Word	Receive bytes 7 + 8, freely assignable



Process data exchange

If the input *bEnable* is set to TRUE, the FB **L_8200DataControl** starts to exchange process data

- The data are exchanged via the identifiers of the process data object **CAN2_IO** which consist of the basic identifier and the node address of the frequency inverter selected at the input *byNodeAdr*:

$$\text{Identifier} = \text{Basis Identifier} + \text{node address of frequency inverter}$$

	Basic identifier	
	dec	hex
CAN2_IO (event or time-controlled process data)		
	CAN2_IN	640280
	CAN2_OUT	641281

- If a FALSE-TRUE transition occurs at the input *bResetNode*, the frequency inverter with the node address *byNodeAdr* accepts the data which are relevant for communication and changes to "Operational".
- The cycle time *tCycleTime* is used to send the data to the frequency inverter.



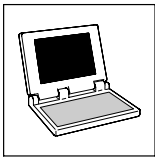
Tip!

Process data transfer is only possible if the node address is correct and the frequency inverter is in the state "Operational".

- Therefore, the input *bResetNode* should always be set to TRUE.

Assignment of send and receive bytes

Send byte	Assignment	Receive byte	Assignment
1 + 2	Control word <i>wControlData</i>	1 + 2	Status word <i>wStatusData</i>
3 + 4	Setpoint <i>nNSetN1_a</i>	3 + 4	Actual value <i>nMCtrl_Nout_Slip</i>
5 + 6	<i>a_{min}</i> boost <i>wBoostVoltage</i>	5 + 6	freely assignable via <i>wOutputData3</i>
7 + 8	freely assignable via <i>wInputData</i>	7 + 8	freely assignable via <i>wOutputData4</i>



Function library Lenze8200Drive.lib

Functions/function blocks

L_8200DataControl - Send/receive process data

Time monitoring

The input *tTimeOut* is used to monitor the transfer time of the data received from the frequency inverter.

- With a time > *t#0ms* the monitoring is activated, otherwise it is deactivated.
- The monitoring time must be adapted to the cycle and task time of the process data object **CAN2_IO**.

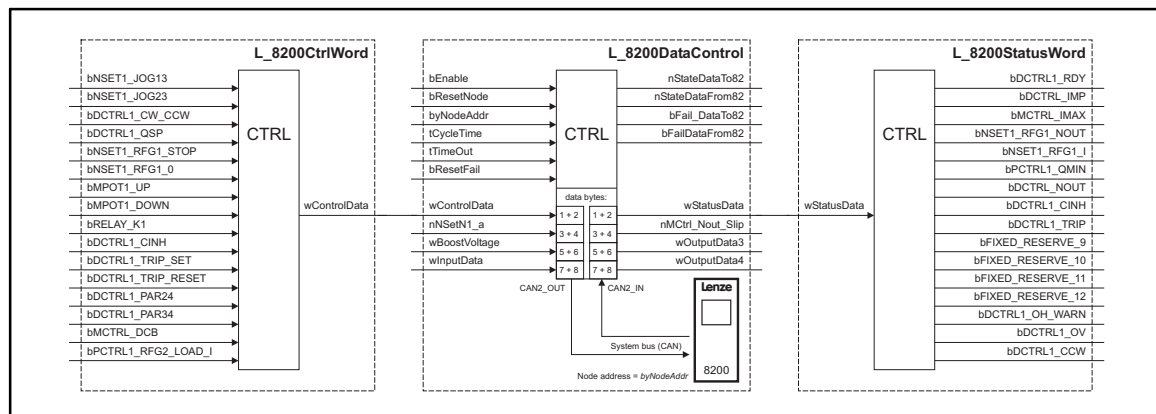
Status messages

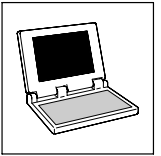
The status outputs *nStateDataTo82* and *nStateDataFrom82* indicate the current state of the send and receive objects.

- If an error occurs, an error number will be indicated and the corresponding message output *bFail_DataTo82* or *bFail_DataFrom82* will be set to TRUE.
- The status and message outputs can be reset by setting *bReset* to TRUE.
- The last message/error number is saved in an internal variable (*nLastMessageNo* or *nLastFailureNo*).

Selection of boolean control signals and output of boolean status messages

- The FB **L_8200CtrlWord** is used to select the control word to be transmitted as boolean control signals. (□ 3-15)
- The FB **L_8200StatusWord** is used to divide the receive status word for evaluation and further processing into individual boolean status signals. (□ 3-16)

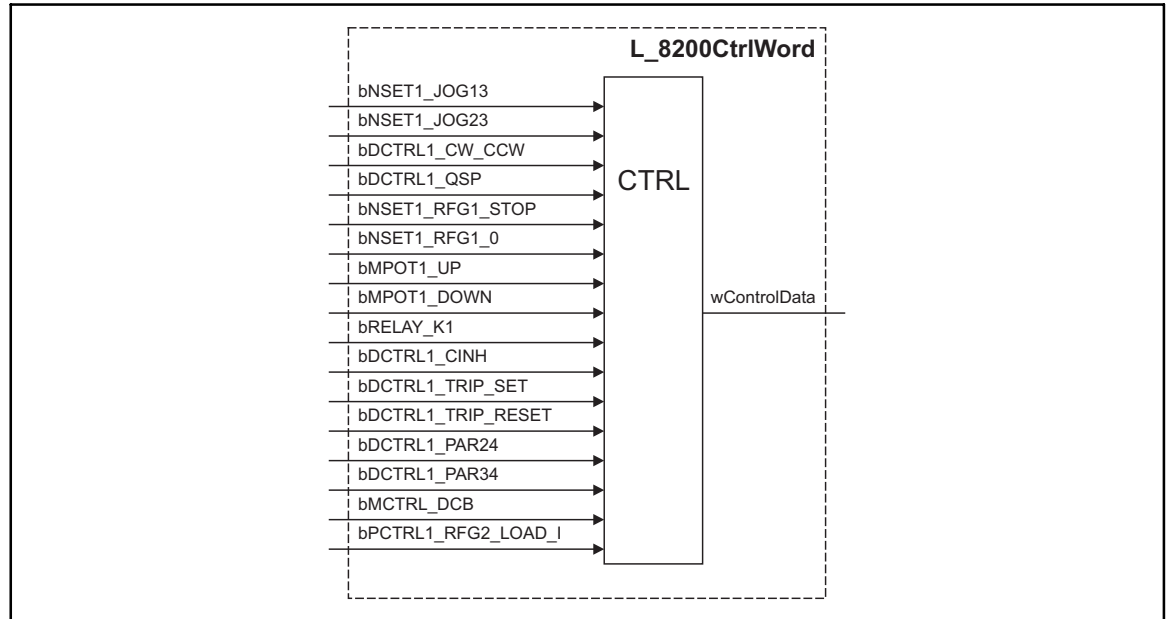




3.4 L_8200CtrlWord - Selection of boolean control signals

Function block

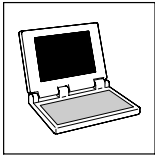
This FB can be used to select the control word to be sent to the frequency inverter which is expected by the FB **L_8200DataControl** at the FB input *wControlData* as boolean control signals.



FB call in:	☒ Cyclic task (PLC_PRG)	☒ Time-controlled task (INTERVAL)	☒ Event-controlled task (EVENT)	☐ Interrupt task
--------------------	---	---	---	---

Inputs	Data type	Information/possible settings
bNSET1_JOG13	Bool	Detailed information about the meaning of the individual control signals can be found in the Operating Instructions for the 8200 frequency inverter. Note! The composition of the control word depends on the parameterisation of the frequency inverter! The names of the control signals contained in this Manual have been adapted to the commented assignments in the FB L_8200Parameter. (3-2)
bNSET1_JOG23	Bool	
bDCTRL1_CW_CCW	Bool	
bDCTRL1_QSP	Bool	
bNSET1_RFG1_STOP	Bool	
bNSET1_RFG1_0	Bool	
bMPOT1_UP	Bool	
bMPOT1_DOWN	Bool	
bRELAY_K1	Bool	
bDCTRL1_CINH	Bool	
bDCTRL1_TRIP_SET	Bool	
bDCTRL1_TRIP_RESET	Bool	
bDCTRL1_PAR24	Bool	
bDCTRL1_PAR34	Bool	
bMCTRL_DCB	Bool	
bPCTRL1_RFG2_LOAD_I	Bool	

Outputs	Data type	Value/meaning
wControlData	Word	Control word to be sent to the frequency inverter. Connect this output with the input <i>wControlData</i> of the FB L_8200DataControl. (3-10)



Function library Lenze8200Drive.lib

Functions/function blocks

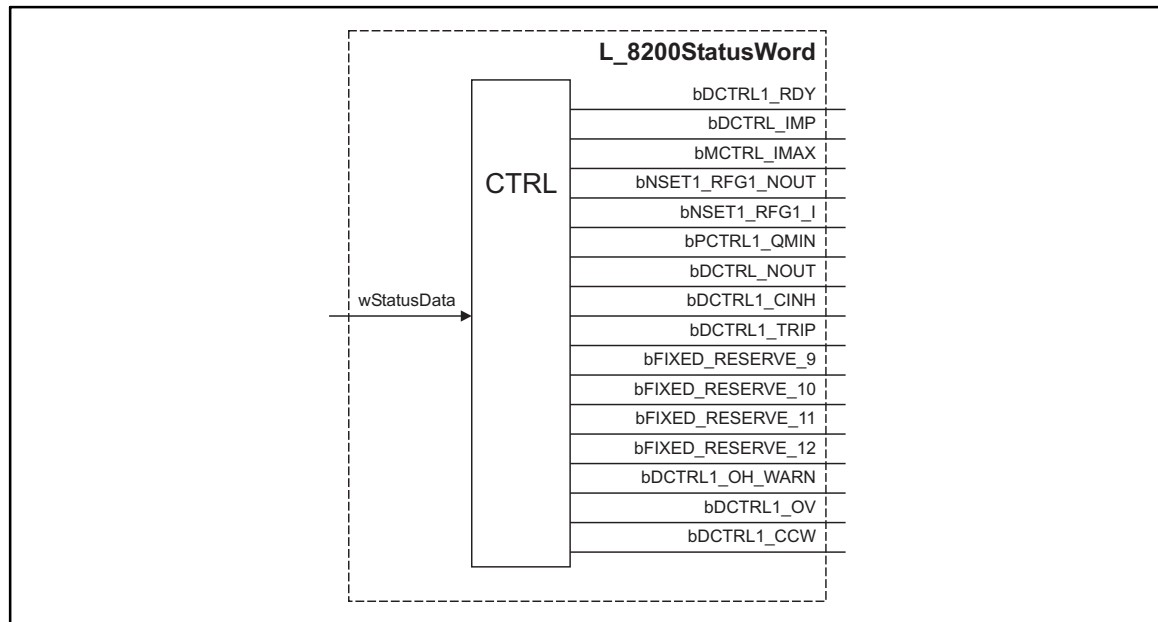
L_8200StatusWord - Output of boolean status signals

3.5

L_8200StatusWord - Output of boolean status signals

Function block

This FB is used to divide the status word received by the frequency inverter which is provided by the FB **L_8200DataControl** at the FB output *wStatusData* into boolean status signals.



FB call in:	☒ Cyclic task (PLC_PRG)	☒ Time-controlled task (INTERVAL)	☒ Event-controlled task (EVENT)	☐ Interrupt task
--------------------	---	---	---	---

Inputs	Data type	Information/possible settings
wStatusData	Word	Status word received by the frequency inverter. Connect this input with the output <i>wStatusData</i> of the FB L_8200DataControl. (3-10)

Outputs	Data type	Value/meaning
bDCTRL1_RDY	Bool	Detailed information about the meaning of the individual status signals can be found in the Operating Instructions for the 8200 frequency inverter. Note! The composition of the status word depends on the parameterisation of the frequency inverter! The names of the status signals contained in this Manual have been adapted to the commented assignments in the FB L_8200Parameter. (3-2)
bDCTRL1_IMP	Bool	
bMCTRL1_IMAX	Bool	
bNSET1_RFG1_NOUT	Bool	
bNSET1_RFG1_I	Bool	
bPCTRL1_QMIN	Bool	
bDCTRL1_NOUT	Bool	
bDCTRL1_CINH	Bool	
bDCTRL1_TRIP	Bool	
bFIXED_RESERVE_9	Bool	
bFIXED_RESERVE_10	Bool	
bFIXED_RESERVE_11	Bool	
bFIXED_RESERVE_12	Bool	
bDCTRL1_OH_WARN	Bool	
bDCTRL1_OV	Bool	
bDCTRL1_CCW	Bool	