

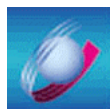
Lenze

Handbuch

IEC 61131-3

inside

***Global Drive
PLC Developer Studio***



Global Drive

Funktionsbibliothek

Lenze8200Drive.lib

Die Funktionsbibliothek **Lenze8200Drive.lib** kann für folgende Lenze PLC's verwendet werden:

	Typenbezeichnung	ab Hardwarestand	ab Softwarestand
9300 Servo PLC	EVS93XX-xT	6A	6.0
Drive PLC	EPL10200	1A	6.0
ECSxA	ECSxAxxx	1C	7.0

Wichtiger Hinweis:

Die Software wird dem Benutzer in der vorliegenden Form zur Verfügung gestellt. Alle Risiken hinsichtlich der Qualität und der durch ihren Einsatz ermittelten Ergebnisse verbleiben beim Benutzer. Entsprechende Sicherheitsvorkehrungen gegen eventuelle Fehlbedienungen sind vom Benutzer vorzusehen.

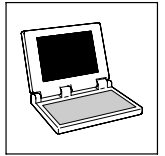
Wir übernehmen keine Verantwortung für direkt oder indirekt entstandene Schäden, z. B. Gewinnverluste, Auftragsverluste oder geschäftliche Beeinträchtigungen jeglicher Art.

© 2002 Lenze Drive Systems GmbH

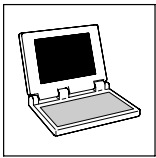
Ohne besondere schriftliche Genehmigung von Lenze Drive Systems GmbH darf kein Teil dieser Dokumentation vervielfältigt oder Dritten zugänglich gemacht werden.

Wir haben alle Angaben in dieser Dokumentation mit größter Sorgfalt zusammengestellt und auf Übereinstimmung mit der beschriebenen Hard- und Software geprüft. Trotzdem können wir Abweichungen nicht ganz ausschließen. Wir übernehmen keine juristische Verantwortung oder Haftung für Schäden, die dadurch eventuell entstehen. Notwendige Korrekturen werden wir in die nachfolgenden Auflagen einarbeiten.

Alle in dieser Dokumentation aufgeführten Markennamen sind Warenzeichen ihrer jeweiligen Besitzer.

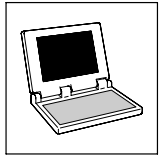


1 Vorwort und Allgemeines	1-1
1.1 Über dieses Handbuch	1-1
1.1.1 Konventionen in diesem Handbuch	1-1
1.1.2 Aufbau der Beschreibungen	1-2
1.1.3 Piktogramme in diesem Handbuch	1-2
1.1.4 Verwendete Begriffe	1-2
1.2 Versionskennung der Funktionsbibliothek	1-3
2 Einführung	2-1
2.1 Allgemeines	2-1
2.2 Parametrierung des Frequenzumrichters	2-2
2.2.1 Vergabe der Systembus-Knotenadressen für die Frequenzumrichter	2-3
2.2.2 Funktionsblockaufruf	2-3
2.3 Prozeßdatenaustausch mit dem Frequenzumrichter	2-4
3 Funktionen/Funktionsblöcke	3-1
3.1 L_8200Parameter - Codestellen initialisieren	3-1
3.2 L_8200Initialization - Frequenzumrichter initialisieren	3-6
3.3 L_8200DataControl - Prozeßdaten senden/empfangen	3-9
3.4 L_8200CtrlWord - Vorgabe boolescher Steuersignale	3-14
3.5 L_8200StatusWord - Ausgabe boolescher Statussignale	3-15



Funktionsbibliothek Lenze8200Drive.lib

Inhalt



1 Vorwort und Allgemeines

1.1 Über dieses Handbuch

Dieses Handbuch enthält Informationen zur Funktionsbibliothek **Lenze8200Drive.lib** für das **Drive PLC Developer Studio**.

- In der Funktionsbibliothek **Lenze8200Drive.lib** sind Funktionsblöcke enthalten, mit denen ein Datenaustausch zwischen der PLC und einem bzw. mehreren Frequenzumrichtern der Reihe 8200 über Systembus (CAN) realisiert werden kann.
- Bis zu 45 Parameter (Codestellen) können von der PLC aus im Frequenzumrichter parametrierbar werden, desweiteren ist ein zyklischer Austausch von Prozeßdaten mit dem Frequenzumrichter möglich.

1.1.1 Konventionen in diesem Handbuch

Dieses Handbuch verwendet folgende Konventionen zur Unterscheidung von verschiedenen Arten von Informationen:

Variablenbezeichner

sind im erklärenden Text in Kursivschrift dargestellt:

- "Mit *bReset...*"



Tip!

Informationen zu den Konventionen, die für die Variablenbezeichner von Lenze Systembausteinen, Funktionsblöcken sowie Funktionen verwendet werden, um eine einheitliche und durchgängige Benennung zu gewährleisten und dadurch die Lesbarkeit von SPS-Programmen zu fördern, finden Sie im Anhang der DDS-Online-Dokumentation "Einführung in die IEC61131-3 Programmierung".

Lenze Funktionen/Funktionsblöcke

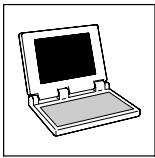
erkennen Sie am Namen. Sie beginnen immer mit einem "L_":

- "Mit der Funktion **L_TBConvBitsToByte...**"

Programmlistings

sind in der Schriftart "Courier" aufgeführt, wobei Schlüsselwörter fett ausgezeichnet sind:

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



Funktionsbibliothek Lenze8200Drive.lib

Vorwort und Allgemeines

1.1.2 Aufbau der Beschreibungen

Die Beschreibungen der einzelnen Funktionen/Funktionsblöcke sowie Systembausteine in diesem Handbuch sind einheitlich nach folgendem Schema aufgebaut:

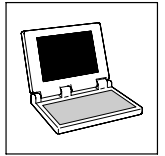
	Funktion	Funktionsblock (FB)/ Systembaustein (SB)
①	Überschrift mit Funktion und Bezeichner	
②	Deklaration der Funktion: • Datentyp des Rückgabewertes • Funktionsbezeichner • Liste der Übergabeparameter	-
③	Kurzbeschreibung mit den wichtigsten Eigenschaften	
④	Grafische Darstellung der Funktion mit den zugehörigen Variablen • Übergabeparameter • Rückgabewert	Grafische Darstellung des FB/SB mit den zugehörigen Variablen • Eingangsvariablen • Ausgangsvariablen
⑤	Tabelle mit Informationen zu den Übergabeparametern: • Bezeichner • Datentyp • Einstellmöglichkeiten • Info	Tabelle mit Informationen zu den Ein- und Ausgangsvariablen: • Bezeichner • Datentyp • Variablentyp • Einstellmöglichkeiten • Info
⑥	Tabelle mit Informationen zum Rückgabewert: • Datentyp des Rückgabewertes • Mögliche Rückgabewerte und ihre Bedeutung	-
⑦	Weitere Informationen (Anmerkungen, Tips, Anwendungsbeispiele, etc.)	

1.1.3 Piktogramme in diesem Handbuch

	verwendete Piktogramme	Signalwörter	
Warnung vor Sachschäden		Stop!	Warnt vor möglichen Sachschäden . Mögliche Folgen bei Mißachtung: Beschädigung des Antriebsreglers/Antriebssystems oder seiner Umgebung.
Sonstige Hinweise		Tip! Hinweis!	Kennzeichnet einen Tip bzw. Hinweis.

1.1.4 Verwendete Begriffe

Begriff	Im folgenden Text verwendet für
DDS	Drive PLC Developer Studio
FB	Funktionsblock
GDC	Global Drive Control (Parametrier-Programm von Lenze)
Parameter-Codestellen	Codestellen zum Einstellen der Funktionalität eines Funktionsblocks
PLC	9300 Servo PLC - Drive PLC - Achsmodul ECSxA "Application"
SB	Systembaustein



1.2 Versionskennung der Funktionsbibliothek

Die Version der Funktionsbibliothek können Sie über die globale Konstante `C_w[Funktionsbibliotheksname]Version` abfragen.

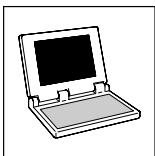
Versionskennung ab PLC Software-Stand 7.x:

Konstante	Bedeutung	Beispiel Wert
<code>C_w[Funktionsbibliotheksname]VersionER</code>	Externes Release	01
<code>C_w[Funktionsbibliotheksname]VersionEL</code>	Externes Level	05
<code>C_w[Funktionsbibliotheksname]VersionIR</code>	Internes Release	00
<code>C_w[Funktionsbibliotheksname]VersionBN</code>	Build-Nummer	00

Version: 01 05 00 00

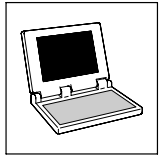
Der Wert dieser Konstanten ist hexadezimal-kodiert.

- Im Beispiel bedeutet der Wert "01050000" Version "1.05".



Funktionsbibliothek Lenze8200Drive.lib

Vorwort und Allgemeines



2 Einführung

2.1 Allgemeines

Durch Einbinden der Funktionsbibliothek **Lenze8200Drive.lib** in ein PLC-Programm soll die Ansteuerung eines Frequenzumrichters erleichtert werden.

Im Wesentlichen sind dazu zwei Funktionsblöcke erforderlich:

1. Ein Funktionsblock zur Parametrierung des Frequenzumrichters.
2. Ein Funktionsblock zum Austausch der Prozeßdaten zwischen PLC und Frequenzumrichter.

Die Parametrierung kann individuell über das PLC-Programm erfolgen, hierbei können bis zu 45 Parameter eingestellt werden.

Konzepte für eine Parametrierung

Für die Parametrierung werden die folgenden zwei Konzepte unterstützt:

- Parametrierung über direkte Anweisungen in einem Funktionsblock
- Parametrierung über Codestellen (Parametrierung "online" möglich)



Hinweis!

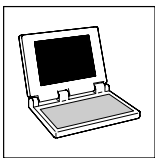
Für die Ansteuerung des Frequenzumrichters von der PLC aus müssen folgende Voraussetzungen im **Frequenzumrichter** erfüllt sein:

- Der Werksabgleich ist geladen.
(Beim Frequenzumrichter 8200vector z. B. über Codestelle C0002)
- Die Systembus-Knotenadresse ist korrekt eingestellt.
(Beim Frequenzumrichter 8200vector z. B. über Codestelle C0350)
- Die veränderten Kommunikationseinstellungen der Systembus-Schnittstelle wurden durch einen Reset-Node übernommen.
(Beim Frequenzumrichter 8200vector z. B. über Codestelle C0358)



Tip!

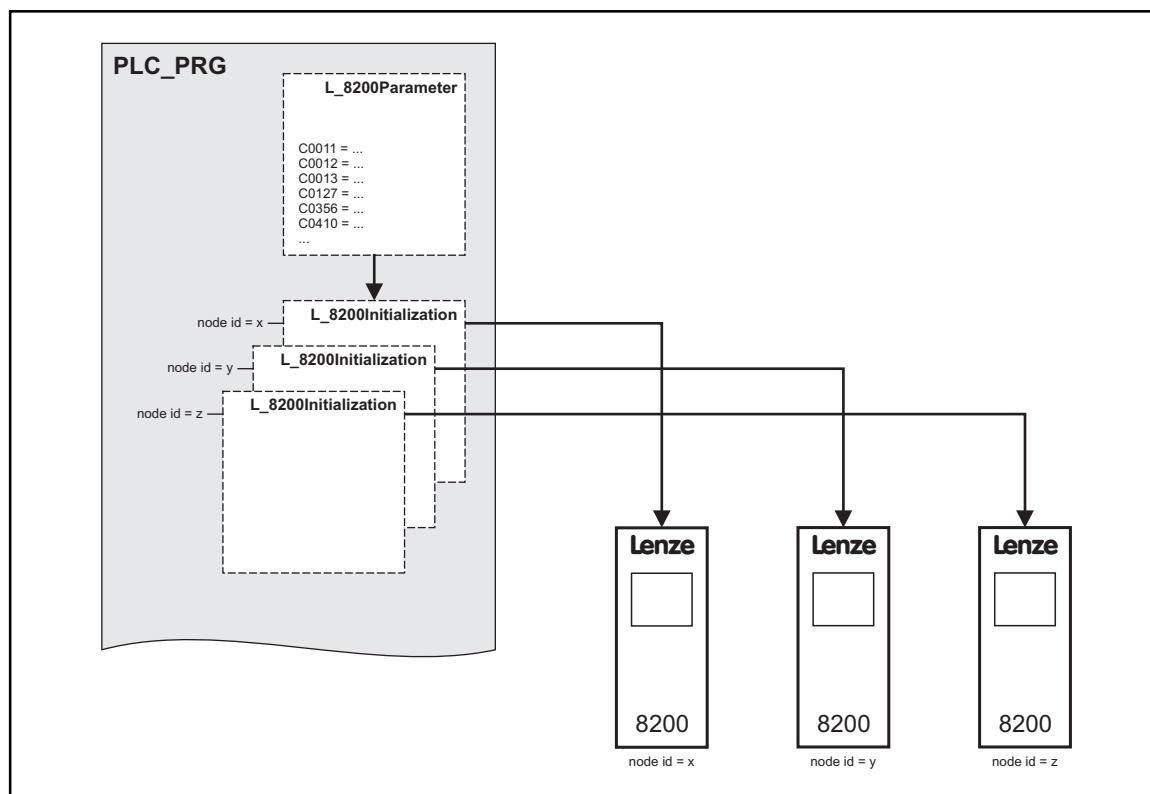
Ausführliche Informationen zum Systembus (CAN) sowie zur Funktionsbibliothek **LenzeCanDrv.lib** finden Sie im Handbuch "Systembus (CAN) bei Lenze PLC-Geräten".

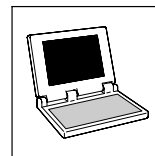


2.2 Parametrierung des Frequenzumrichters

Die Parameterdaten sollen bei jedem Programmstart **nur einmal** in der Initialisierungsphase zum Frequenzumrichter geschrieben werden, daher hat der Aufruf der FBs **L_8200Parameter** und **L_8200Initialization** in der Systemtask **PLC_PRG** zu erfolgen.

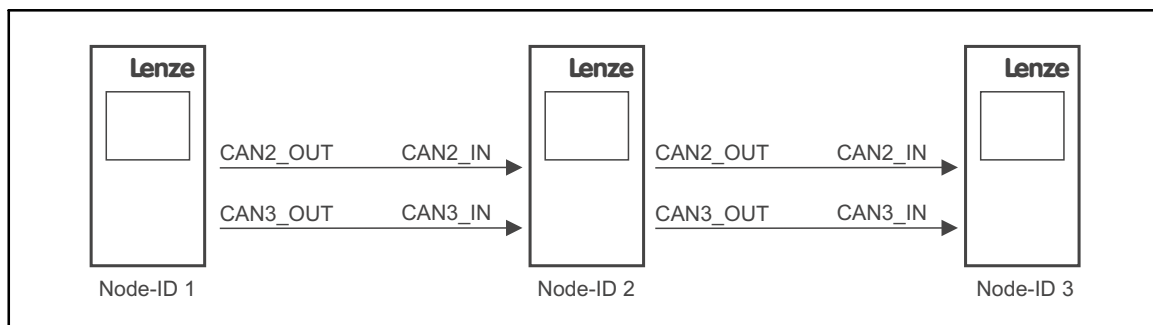
- Mit dem FB **L_8200Parameter** werden die Codestellen mit den entsprechenden Werten initialisiert.
- Mit dem FB **L_8200Initialization** erfolgt danach der Parametertransfer zum entsprechenden Frequenzumrichter.



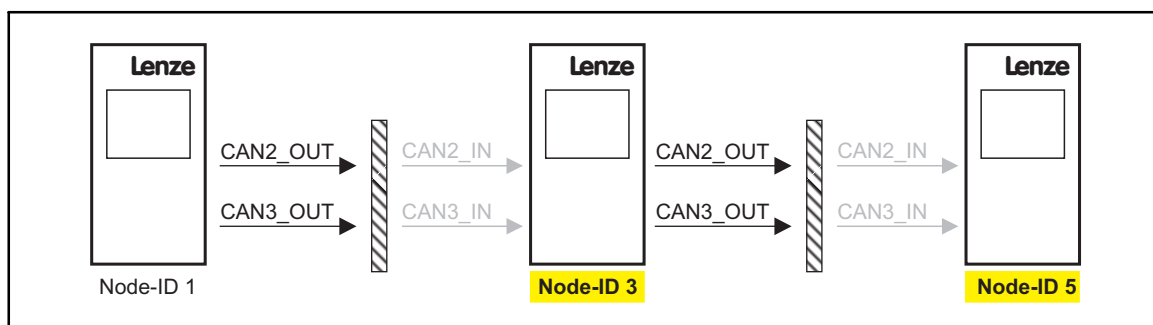


2.2.1 Vergabe der Systembus-Knotenadressen für die Frequenzumrichter

Werden Lenze-Geräte mit Knotenadressen in lückenfrei aufsteigender Reihenfolge versehen, dann sind die Identifier der ereignisgesteuerten Datenobjekte (CAN2_IO/CAN3_IO) werksseitig so eingestellt, daß eine Kommunikation von Gerät zu Gerät möglich ist:



Damit mehrere Frequenzumrichter mit unterschiedlichen Werten parametrieren werden können, ist es erforderlich, jeweils mindestens eine Knotenadresse in der Reihenfolge auszulassen, so daß hierdurch die automatische Weiterleitung der Daten nicht mehr wirksam ist:



- Auf diese Weise lassen sich max. 32 Frequenzumrichter parametrieren (mit Knotenadresse 1 - 3 - 5 - 7 - ... - 63).



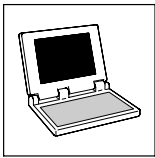
Hinweis!

Eine veränderte Knotenadresse im Frequenzumrichter ist erst nach einem Reset-Node wirksam. (Beim Frequenzumrichter 8200vector z. B. über Codestelle C0358)

2.2.2 Funktionsblockaufruf

Der FB **L_8200Initialization** muß zwingend zyklisch aufgerufen werden, da die Daten intern über den FB **L_ParWrite** zum Frequenzumrichter übertragen werden.

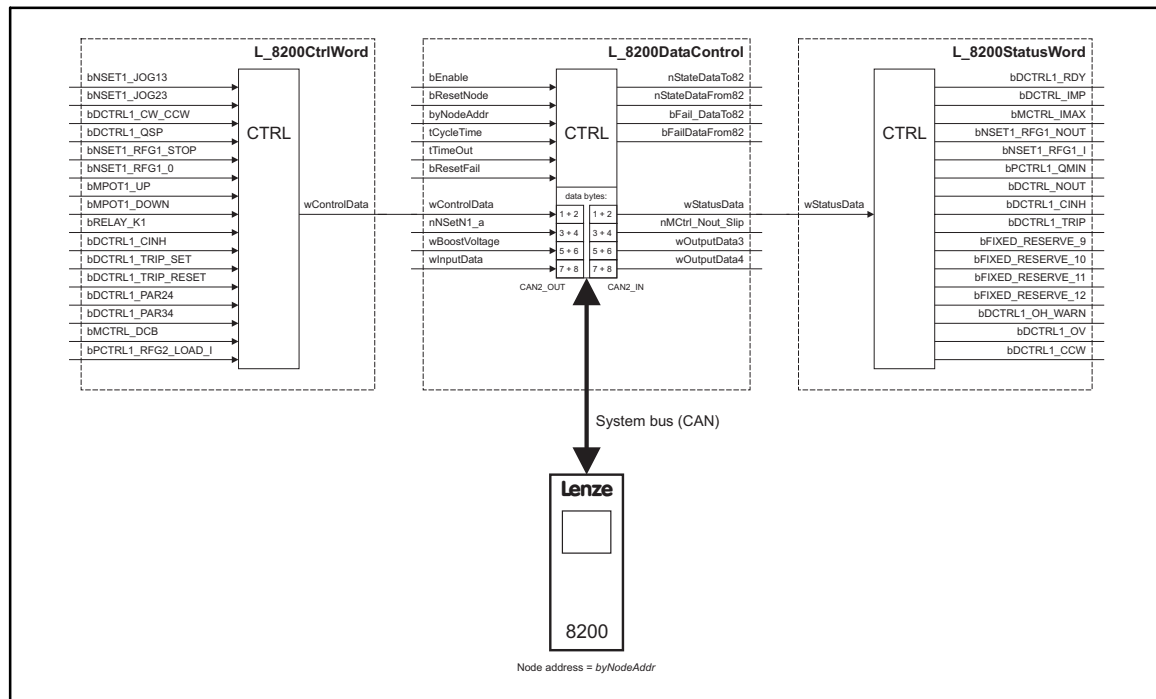
- Je nach Anzahl der zu übertragenden Daten kann die Übertragung mehrere Zyklen andauern.
- Wurde die Übertragung erfolgreich durchgeführt, so wird der Meldeausgang *bParDone* auf TRUE gesetzt.
- Nach erfolgreicher Übertragung (*bParDone* = TRUE) ist die Parametrierung abgeschlossen und ein weiterer Aufruf der FBs **L_8200Parameter** und **L_8200Initialization** ist nicht mehr erforderlich.



2.3 Prozeßdatenaustausch mit dem Frequenzumrichter

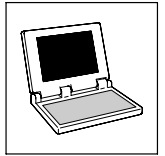
Im Gegensatz zu den Parameterdaten müssen die Prozeßdaten für eine Aktualisierung der Ein- und Ausgangsinformationen zwischen der PLC und dem Frequenzumrichter zyklisch ausgetauscht werden.

- Mit Hilfe des FBs **L_8200CtrlWord** können Sie das zu sendende Steuerwort in Form boolescher Steuersignale vorgeben. (▢ 3-14)
- Über den FB **L_8200DataControl** erfolgt der eigentliche Prozeßdatenaustausch mit dem entsprechenden Frequenzumrichter. (▢ 3-9)
- Mit Hilfe des FBs **L_8200StatusWord** können Sie das empfangende Statuswort zur Auswertung und Weiterverarbeitung in einzelne boolesche Statussignale aufteilen. (▢ 3-15)



Hinweis!

Die Taskzykluszeit ist abhängig von der Anzahl der verwendeten Objekte.



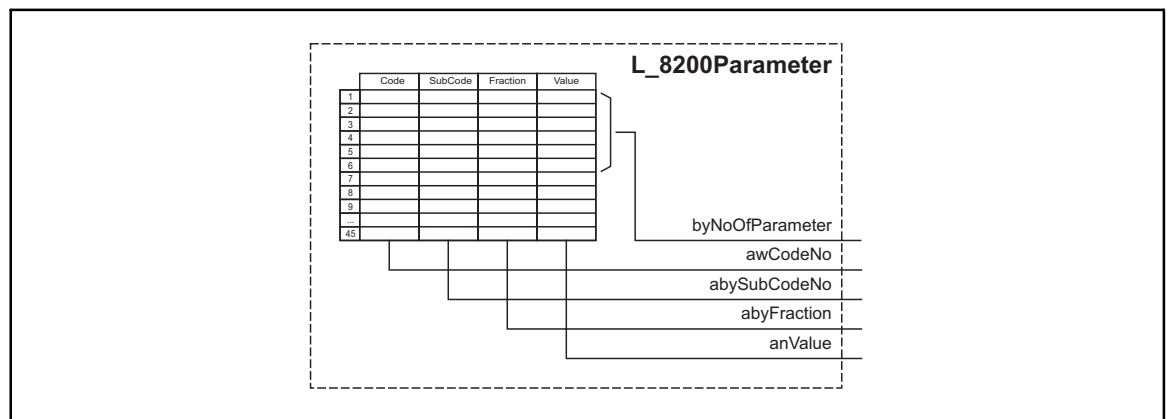
3 Funktionen/Funktionsblöcke

3.1 L_8200Parameter - Codestellen initialisieren

Funktionsblock

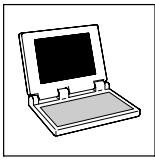
Mit diesem FB können bis zu 45 Parameter (Codestellen) im Frequenzumrichter parametrierbar werden. Für die Parametrierung werden die folgenden zwei Konzepte unterstützt:

- Parametrierung über direkte Anweisungen in einem Funktionsblock
- Parametrierung über Codestellen (Parametrierung "online" möglich)



FB-Aufruf in:	☒ Zyklischer Task (PLC_PRG)	☒ Zeitgesteuerter Task (INTERVAL)	☒ Ereignisgesteuerter Task (EVENT)	☐ Interrupt-Task
----------------------	---	---	--	---

Eingänge	Datentyp	Info/Einstellmöglichkeiten
byNoOfParameter	Byte	Anzahl der zu übertragenden Parameter (max. 45)
awCodeNo	Array of Word	Nummer der Codestelle, die als Parameter 1..45 übertragen werden soll.
abySubCodeNo	Array of Byte	Nummer der Subcodestelle, die als Parameter 1..45 übertragen werden soll.
abyFraction	Array of Byte	Anzahl der Kommastellen für Parameter 1..45 • Siehe auch Beschreibung zum FB L_ParWrite im Handbuch Funktionsbibliothek LenzeDrive.lib.
anValue	Array of Integer	Wert der Codestelle, die als Parameter 1..45 übertragen werden soll.



Funktionsbibliothek Lenze8200Drive.lib

Funktionen/Funktionsblöcke

L_8200Parameter - Codestellen initialisieren

Konzept I: Parametrierung über direkte Anweisungen in einem Funktionsblock

Hierfür ist ein neuer Funktionsblock nach Vorlage des FB **L_8200Parameter** zu erstellen, in dem die Parameterdaten durch direkte Zuweisungen vorgegeben werden.



Hinweis!

Beachten Sie, daß Änderungen an diesem FB (z. B. durch geänderte Zuweisungen) ein erneutes Kompilieren und einen erneuten Programmdownload in die PLC erfordern.

- Soll eine "Online"-Änderung der Parameter über den Parameter-Monitor oder das Keypad möglich sein, so ist hierfür das Konzept II ("Parametrierung über Codestellen") zu verwenden.



Tip!

Der FB **L_8200Parameter** enthält bereits **auskommentierte** Zuweisungen für alle zur Ansteuerung des Frequenzumrichters relevanten Parameter.

- Eine Auflistung dieser Parameter finden Sie in der Tabelle auf der folgenden Seite.
- Die Namensgebung der Ein-/Ausgänge der FBs **L_8200CtrlWord** und **L_8200StatusWord** ist speziell auf diese Parameterauswahl abgestimmt.
- Sie können diese Vorbelegung verwenden, indem Sie die Zuweisungen einfach in den neu angelegten FB kopieren und die Kommentarzeichen entfernen, so daß die Zuweisungen wirksam werden.
- Natürlich können Sie die Zuweisungen auch entsprechend Ihrer Anwendung anpassen.

Vorgehensweise

1. Legen Sie einen neuen Funktionsblock an.
2. Kopieren Sie über die Bibliotheksverwaltung den Quelltext des FB **L_8200Parameter** in den neu angelegten Funktionsblock.
3. Entfernen Sie die Kommentarzeichen der Zuweisungen, um die Vorbelegung zu übernehmen bzw. passen Sie die Zuweisungen an Ihre Anwendung an.

```
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;

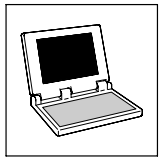
(* deceleration time Tif *)
awCodeNo[3]:=13;    abySubCodeNr[3]:=0;    anValue[3]:=0010;    abyFraction[3]:=2;

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

...
```

Abb. 3-1

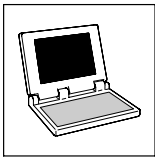
Beispiel: Quelltext (Auszug) eines neu erstellten FBs basierend auf dem FB **L_8200Parameter**



Nr.	Codestelle	Subcodestelle	Einstellung	Info/Bedeutung
1	C0011	0	50	Maximale Ausgangsfrequenz = 50 Hz (Werkseinstellung)
2	C0012	0	0.1	Hochlaufzeit Hauptsollwert; Tir = 0.1 s (Werkseinstellung)
3	C0013	0	0.1	Ablaufzeit Hauptsollwert; Tif = 0.1 s (Werkseinstellung)
4	C0127	0	1	Sollwertvorgabe normiert
5	C0356	2	50	CAN2-OUT Zykluszeit = 50 ms
6	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	6	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	MCTRL-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 ⇒ MCTRL-IMAX
27	C0418	4	4	CAN-OUT2-BIT3 ⇒ DCTRL-RFG1=NOUT
28	C0418	5	5	CAN-OUT2-BIT4 ⇒ NSET-RFG1-I=0
29	C0418	6	6	CAN-OUT2-BIT5 ⇒ PCTRL-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	Dynamische Parametersatzänderung
42				nicht belegt
43				nicht belegt
44				nicht belegt
45				nicht belegt

Tab. 3-1

Parametervorbelegung, die als Kommentar im FB **L_8200Parameter** enthalten ist.



Funktionsbibliothek Lenze8200Drive.lib

Funktionen/Funktionsblöcke

L_8200Parameter - Codestellen initialisieren

Konzept II: Parametrierung über Codestellen

Durch eine Verknüpfung der Ausgänge des FBs mit User-Codestellen in der PLC können die entsprechenden Parameter des Frequenzumrichters direkt im Parameter-Monitor vorgegeben werden.



Hinweis!

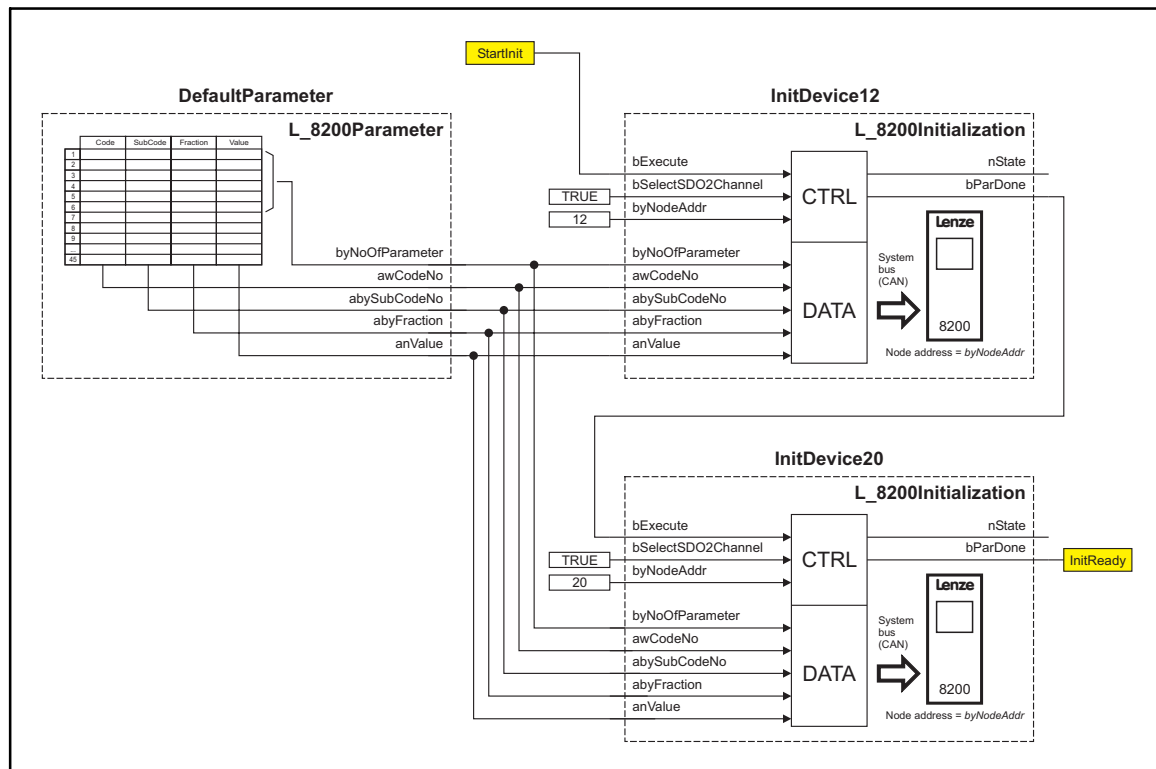
Statt einer Kopie können Sie bei diesem Konzept eine Instanz des FB **L_8200Parameter** verwenden, beachten Sie aber, daß vor dem Programmstart alle Parameterwerte über die entsprechenden Codestellen/Subcodestellen vorgegeben werden müssen.

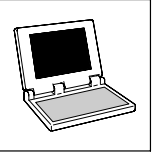
Vorgehensweise

1. Legen Sie eine Instanz des Funktionsblocks **L_8200Parameter** an.
2. Weisen Sie den Ausgängen der Instanz mit Hilfe des **Instanz-Parametermanager** User-Codestellen zu
3. Kompilieren Sie das Projekt, damit im **Parameter-Monitor** die entsprechenden Codestellen/Subcodestellen zwecks Parametrierung zur Verfügung stehen.

Beispiel I: Mehrere Frequenzumrichter mit identischen Parametern einstellen

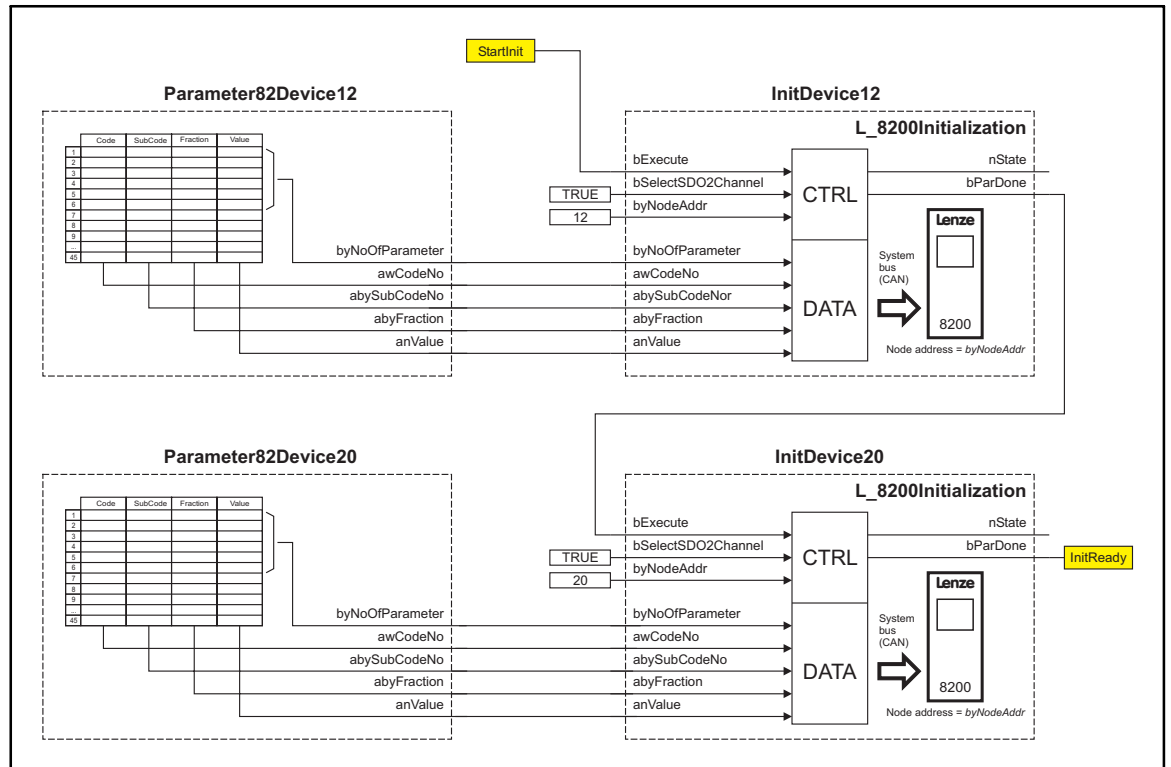
Sollen mehrere Frequenzumrichter mit identischen Parametern eingestellt werden, so wird nur ein FB mit den eingestellten Parametern benötigt.

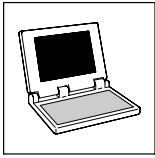




Beispiel I: Mehrere Frequenzumrichter mit unterschiedlichen Parametern einstellen

Sollen mehrere Frequenzumrichter mit unterschiedlichen Werten parametrisiert werden, so werden entsprechend viele FB's benötigt.





Funktionsbibliothek Lenze8200Drive.lib

Funktionen/Funktionsblöcke

L_8200Initialization - Frequenzumrichter initialisieren

3.2

L_8200Initialization - Frequenzumrichter initialisieren

Funktionsblock

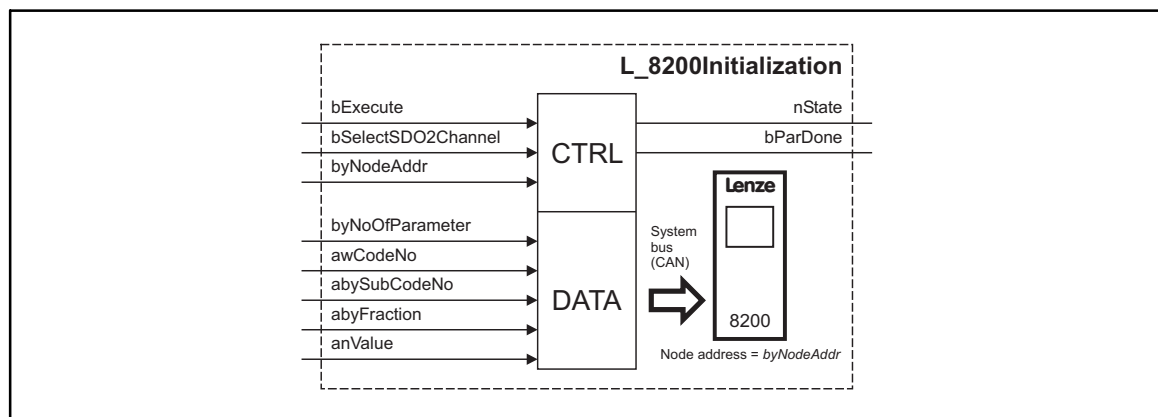
Mit diesem FB wird der Frequenzumrichter mit den Parametern initialisiert, die zuvor mit Hilfe einer Instanz bzw. Kopie des FB **L_8200Parameter** definiert wurden.



Hinweis!

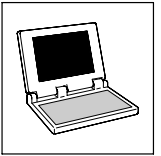
Der FB muß zyklisch aufgerufen werden, da die Daten intern über den FB **L_ParWrite** zum Frequenzumrichter übertragen werden.

- Je nach Anzahl der zu übertragenden Daten kann die Übertragung mehrere Zyklen andauern.
- Wurde die Übertragung erfolgreich durchgeführt, so wird der Meldeausgang **bParDone** auf TRUE gesetzt.
- Nach erfolgreicher Übertragung (**bParDone** = TRUE) ist die Parametrierung abgeschlossen und ein weiterer Aufruf der FBs **L_8200Parameter** und **L_8200Initialization** ist nicht mehr erforderlich.



FB-Aufruf in:	☐ Zyklischer Task (PLC_PRG)	☒ Zeitgesteuerter Task (INTERVAL)	☐ Ereignisgesteuerter Task (EVENT)	☐ Interrupt-Task
----------------------	--	---	---	---

Eingänge	Datentyp	Info/Einstellmöglichkeiten
bExecute	Bool	Start der Datenübertragung zum Frequenzumrichter FALSE → TRUE Datenübertragung starten
bSelectSDOChannel	Bool	Auswahl des Parameterkanals beim Frequenzumrichter FALSE SDO 1 TRUE SDO 2
byNodeAddr	Byte	Geräteadresse (1 ... 63) des Frequenzumrichters, der initialisiert werden soll. • Siehe hierzu auch Kap. 2.2.1. (☞ 2-3)
byNoOfParameter	Byte	Übergabe der Anzahl der zu übertragenden Parameter
awCodeNo	Array of Word	Übergabe der Codestellen
abySubCodeNo	Array of Byte	Übergabe der Subcodestellen
abyFraction	Array of Byte	Übergabe der Kommastellen • Siehe auch Beschreibung zum FB L_ParWrite im Handbuch Funktionsbibliothek LenzeDrive.lib .
anValue	Array of Integer	Übergabe der Codestellenwerte



Ausgänge	Datentyp	Wert/Bedeutung
nState	Integer	Status
		Statusmeldungen:
		0 Die Datenübertragung wurde ausgeführt, der Frequenzumrichter ist initialisiert.
		1 Die Datenübertragung ist aktiv.
		Fehlermeldungen:
		2001 Beim Bustransport ist ein Fehler aufgetreten.
		2002 Bei der Datenübertragung ist ein Time-Out aufgetreten.
		2004 Zugriff verweigert - falsches Zielsystem: Subindex existiert nicht.
		2005 Zugriff verweigert - falsches Zielsystem: Index existiert nicht.
		2013 Zugriff verweigert - falsches Zielsystem: Ungültiger Wertebereich.
		2111 Zugriff verweigert - falsches Zielsystem: Mehr als 4 Nachkommastellen angegeben oder kein Index mit Fixkomma-Format.
		2117 Kommunikationskanal existiert nicht. • Codestelle C2118 der PLC überprüfen (Sendekanal für CAN-Parameterkommunikation).
bParDone	Bool	Meldesignal "Datenübertragung wurde ausgeführt"
		TRUE Die Datenübertragung wurde erfolgreich ausgeführt.

Beispiel: Initialisierung des Frequenzumrichters mit den voreingestellten Parametern

In diesem Beispiel wird ein Frequenzumrichter mit der Systembus-Knotenadresse 12 mit den im FB **L_8200Parameter** voreingestellten Parametern initialisiert, anschließend wird ein zweiter Frequenzumrichter mit der Systembus-Knotenadresse 20 mit den gleichen Parametern initialisiert.



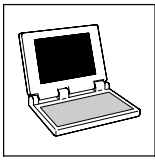
Hinweise!

Führen Sie vor einer Kommunikation mit der PLC in den entsprechenden Frequenzumrichtern jeweils folgende Schritte durch:

1. Laden Sie den Werksabgleich zu laden.
(Beim Frequenzumrichter 8200vector z. B. über Codestelle C0002)
2. Stellen Sie die Systembus-Knotenadresse korrekt ein.
(Beim Frequenzumrichter 8200vector z. B. über Codestelle C0350)
– Weitere Informationen zur Vergabe der Knotenadressen finden Sie im Kap. 2.2.1. (2-3)
3. Führen Sie einen Reset-Node aus, um die veränderten Kommunikationseinstellungen der Systembus-Schnittstelle zu übernehmen.
(Beim Frequenzumrichter 8200vector z. B. über Codestelle C0358)

Führen Sie anschließend die folgenden Schritte im Drive PLC Developer Studio durch:

1. Binden Sie mit Hilfe der Bibliotheksverwaltung die Funktionsbibliotheken **Lenze8200Drive.lib**, **LenzeDrive.lib** und **LenzeCanDrv.lib** in Ihr Projekt ein.
2. Erstellen Sie einen neuen FB mit Namen "DefaultParameter82" und kopieren Sie den Quelltext des FBs **L_8200Parameter** in den neuen FB **DefaultParameter82**.
3. Entfernen Sie im FB **DefaultParameter82** die Kommentare um die Zuweisungen, so daß die Zuweisungen für den Compiler gültig werden.



Funktionsbibliothek Lenze8200Drive.lib

Funktionen/Funktionsblöcke

L_8200Initialization - Frequenzumrichter initialisieren

- Erstellen Sie zwei Instanzen des FBs **L_8200Initialization**.
- Binden Sie die FBs entweder direkt in das Hauptprogramm **PLC_PRG** oder in ein Programm ein, das von **PLC_PRG** zyklisch aufgerufen wird, und verknüpfen Sie die FBs untereinander wie folgt:

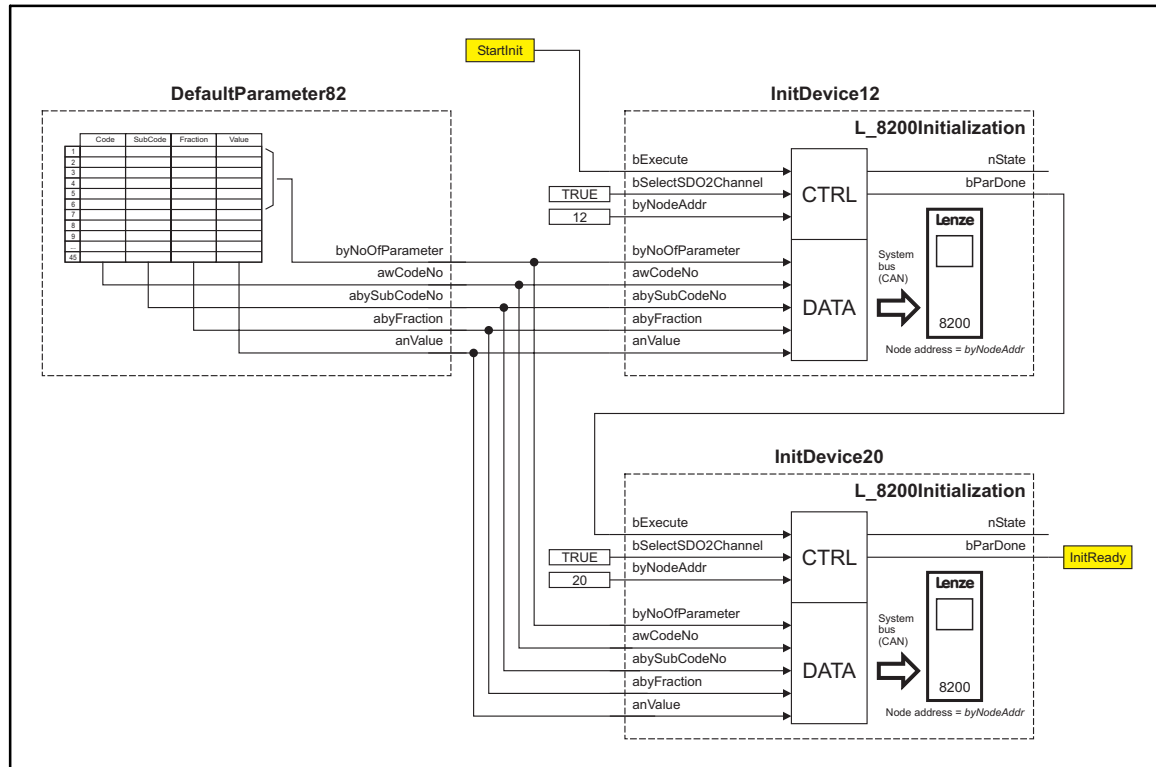
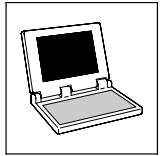


Abb. 3-2 Verknüpfung der FBs untereinander

Ablauf der Initialisierung

- Über den FB **DefaultParameter82**, der auf dem FB **L_8200Parameter** basiert, werden die Parameter an beide Instanzen des **L_8200Initialization** übergeben.
- Durch Setzen von **StartInit** auf TRUE wird die Initialisierung gestartet, die Daten werden an den Frequenzumrichter mit der Knotenadresse 12 über dessen Parameterkanal SDO2 übermittelt.
- Wurden alle Daten erfolgreich übertragen, so wird der Ausgang **bParDone** der Instanz "InitDevice12" auf TRUE gesetzt, dieses Signal wird als Startsignal für die Datenübertragung zum zweiten Frequenzumrichter mit der Knotenadresse 20 verwendet.
- Wurden auch die Daten zum zweiten Frequenzumrichter erfolgreich übertragen, so wird die Variable **InitReady** auf TRUE gesetzt, dieses Signal kann z. B. für den Start der Prozeßdatenübertragung über den FB **L_8200DataControl** verwendet werden.
- Nach erfolgreicher Datenübertragung (**InitReady** = TRUE) ist ein weiterer Aufruf dieser FBs nicht mehr erforderlich.



3.3 L_8200DataControl - Prozeßdaten senden/empfangen

Funktionsblock

Über diesen FB wird der Prozeßdatenaustausch mit dem Frequenzumrichter durchgeführt.

- Der FB arbeitet intern mit einem freien CAN-Sende- und Empfangsobjekt, so daß 8 Byte Nutzdaten gesendet bzw. empfangen werden können.

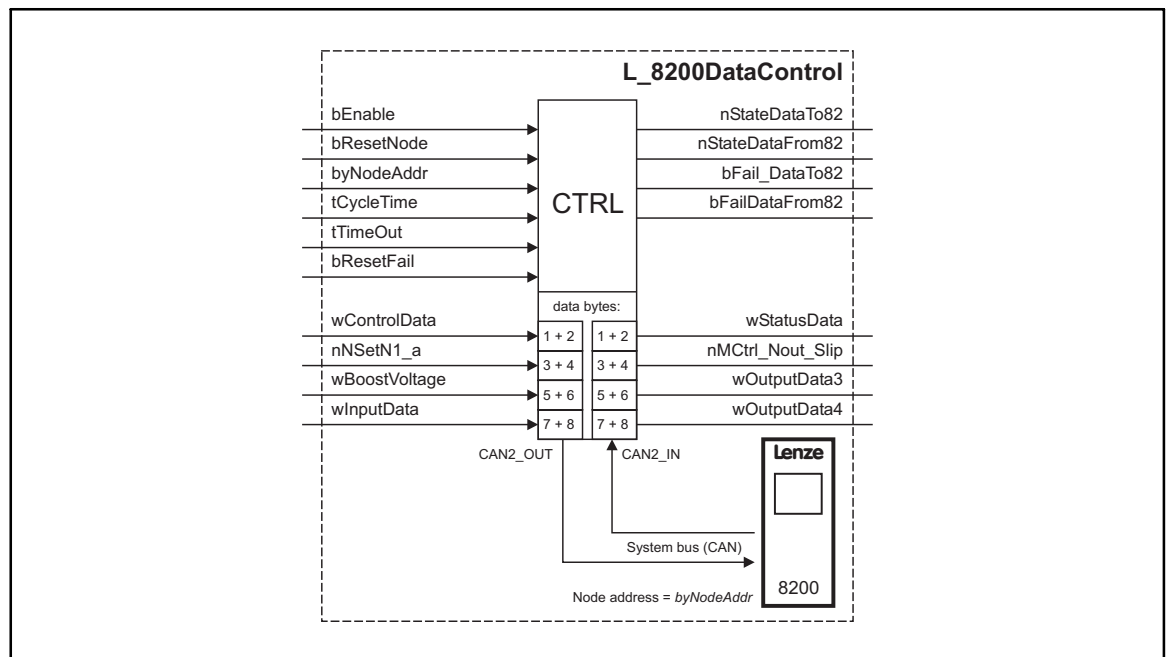


Hinweise!

Für die Bereitstellung von CAN-Sende- und Empfangsobjekten ist die Einbindung der Funktionsbibliothek **LenzeCanDrv.lib** in das Projekt erforderlich.

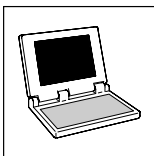
- Damit der Datenaustausch über die freien CAN-Objekte funktioniert, muß der Treiber für die freien CAN-Objekte zuvor in einem Programm (z. B. "PLC_ColdStart") initialisiert worden sein.

Der FB **L_8200DataControl** ist in einem taskgesteuerten Programm aufzurufen.



FB-Aufruf in:	☐ Zyklischer Task (PLC_PRG)	☒ Zeitgesteuerter Task (INTERVAL)	☐ Ereignisgesteuerter Task (EVENT)	☐ Interrupt-Task
----------------------	--	---	---	---

Eingänge	Datentyp	Info/Einstellmöglichkeiten
bEnable	Bool	Start der Prozeßdatenübertragung FALSE $\rightarrow$ TRUE Prozeßdatenübertragung starten.
bResetNode	Bool	Reset-Node im Frequenzumrichter durchführen. - Durch einen Reset-Node werden im Frequenzumrichter kommunikationsrelevante Parametereinstellungen übernommen und im Anschluß daran wechselt der Frequenzumrichter in den Zustand "Operational". - Der Reset-Node wird nur im Frequenzumrichter mit der Geräteadresse <i>byNodeAddr</i> durchgeführt. FALSE $\rightarrow$ TRUE Reset-Node durchführen.
byNodeAddr	Byte	Geräteadresse (1 ... 63) des Frequenzumrichters, mit dem der Prozeßdatenaustausch erfolgen soll.
tCycleTime	Time	Zykluszeit für das Senden der Prozeßdaten zum Frequenzumrichter.
tTimeOut	Time	Überwachungszeit (Time-Out) für den Empfang der Prozeßdaten vom Frequenzumrichter.



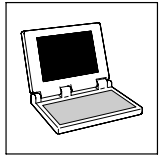
Funktionsbibliothek Lenze8200Drive.lib

Funktionen/Funktionsblöcke

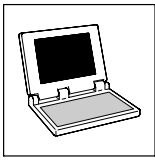
L_8200DataControl - Prozeßdaten senden/empfangen

Eingänge	Datentyp	Info/Einstellmöglichkeiten
bResetFail	Bool	Meladesignale für Fehlermeldungen zurücksetzen FALSE $\rightarrow$ TRUE Die Meladesignale <i>bFail_DataTo82</i> und <i>bFail_DataFrom82</i> werden zurückgesetzt.
wControlData	Word	Prozeßdatenwort (Steuerwort), das zum Frequenzumrichter gesendet werden soll.
nNSetN1_a	Integer	Sollwertvorgabe der Drehzahl für den Frequenzumrichter in [Hz] • Normiert auf $f_{\max}$, d. h. $16384 \equiv 100 \% f_{\max}$ (C0011) des Frequenzumrichters
wBoostVoltage	Word	Umin-Anhebung im Frequenzumrichter • Normiert auf U_n , d. h. $16384 \equiv 100 \% U_n$ des Frequenzumrichters
wInputData	Word	Sendebytes 7 + 8, frei belegbar

Ausgänge	Datentyp	Wert/Bedeutung
nStateDataTo82	Integer	Status des CAN-Sendeobjekts
		Fehlermeldungen:
		-300 Time-Out, Sendedaten vom Frequenzumrichter außerhalb der Überwachungszeit <i>tTimeOut</i> .
		-150 Systembus (CAN) befindet sich nicht im Zustand "Operational".
		-121 Falsche Treibernummer für CAN-Treiber. • Überprüfen Sie die Initialisierung des CAN-Treibers.
		-120 Der CAN-Treiber ist momentan nicht initialisiert. • Überprüfen Sie die Initialisierung des CAN-Treibers.
		-119 Der Sendeauftragsspeicher ist voll. Der Sendeauftrag konnte nicht mehr eingetragen werden. • Erhöhen Sie die Aufrufzykluszeit des FBs.
		-118 Es steht kein freier CAN-Kanal zur Verfügung.
		-12 Der Message-Identifizier (COB-ID) ist außerhalb des zulässigen Bereiches (interner Fehler).
		-11 Der Zeiger <i>p/OAdress</i> zeigt nicht auf SPS-RAM (interner Fehler).
		Statusmeldungen:
		0 Die Prozeßdaten wurden zum Frequenzumrichter gesendet.
		1 Der Sendeauftrag ist noch nicht abgeschlossen und steht noch im Auftragsspeicher.
		2 Die Datenübertragung ist inaktiv. • Die Parametrierung des Frequenzumrichters ist noch nicht abgeschlossen oder <i>bEnable</i> ist auf FALSE gesetzt.
		10 Die Telegrammlänge ist grösser als 8 Byte und wird auf 8 Byte begrenzt.



Ausgänge	Datentyp	Wert/Bedeutung
nStateDataFrom82	Integer	Status des CAN-Empfangsobjekts
		Fehlermeldungen:
		-300 Time-Out, Sendedaten vom Frequenzumrichter außerhalb der Überwachungszeit <i>tTimeOut</i> .
		-150 Systembus (CAN) befindet sich nicht im Zustand "Operational".
		-121 Falsche Treibernummer für CAN-Treiber. • Überprüfen Sie die Initialisierung des CAN-Treibers.
		-120 Der CAN-Treiber ist momentan nicht initialisiert. • Überprüfen Sie die Initialisierung des CAN-Treibers.
		-12 Der Message-Identifizier (COB-ID) ist außerhalb des zulässigen Bereiches (interner Fehler).
		-11 Der Zeiger <i>pIOAdress</i> zeigt nicht auf SPS-RAM (interner Fehler).
		Statusmeldungen:
		0 Die Prozeßdaten wurden vom Frequenzumrichter empfangen.
		2 Die Datenübertragung ist inaktiv. • Die Parametrierung des Frequenzumrichters ist noch nicht abgeschlossen oder <i>bEnable</i> ist auf FALSE gesetzt.
		10 Die Telegrammlänge ist grösser als 8 Byte und wird auf 8 Byte begrenzt.
		200 Die Daten im Empfangsspeicher wurden überschrieben. • Verringern Sie den Aufrufzyklus des FB oder erhöhen Sie den Sendezyklus, mit dem der Frequenzumrichter Daten sendet.
bFail_DataTo82	Bool	Meldesignal "Sendeauftrag wurde abgebrochen bzw. nicht ausgeführt".
		TRUE Sendeauftrag wurde abgebrochen bzw. nicht ausgeführt. • Am Ausgang <i>nStateDataTo82</i> wird die entsprechende Fehlernummer ausgegeben.
bFail_DataFrom82	Bool	Meldesignal "Empfangsauftrag wurde abgebrochen bzw. nicht ausgeführt".
		TRUE Empfangsauftrag wurde abgebrochen bzw. nicht ausgeführt. • Am Ausgang <i>nStateDataFrom82</i> wird die entsprechende Fehlernummer ausgegeben.
wStatusData	Word	Prozeßdatenwort (Statuswort), das vom Frequenzumrichter gesendet wurde.
nMCtrl_Nout_Slip	Integer	Ausgabe der Ist-Drehzahl (inklusive Schlupf) des Frequenzumrichters in [Hz].
wOutputData3	Word	Empfangsbytes 5 + 6, frei belegbar
wOutputData3	Word	Empfangsbytes 7 + 8, frei belegbar



Funktionsbibliothek Lenze8200Drive.lib

Funktionen/Funktionsblöcke

L_8200DataControl - Prozeßdaten senden/empfangen

Prozeßdatenaustausch

Wird der Eingang *bEnable* auf TRUE gesetzt, so beginnt der FB **L_8200DataControl** mit dem Prozeßdatenaustausch.

- Der Datenaustausch erfolgt über den Identifizierungsbereich des Prozeßdatenobjektes **CAN2_IO**, dieser setzt sich zusammen aus dem Basis-Identifizierungsbereich und der Knotenadresse des Frequenzumrichters, die am Eingang *byNodeAdr* vorzugeben ist:

$$\text{Identifizierung} = \text{Basis-Identifizierung} + \text{Knotenadresse des Frequenzumrichters}$$

	Basis-Identifizier	
	dez	hex
CAN2_IO (ereignis- bzw. zeitgesteuerte Prozeßdaten)		
	CAN2_IN	640280
	CAN2_OUT	641281

- Durch eine FALSE-TRUE-Flanke am Eingang *bResetNode* werden im Frequenzumrichter mit der Knotenadresse *byNodeAdr* kommunikationsrelevante Daten übernommen und im Anschluß daran erfolgt im Frequenzumrichter ein Zustandswechsel nach "Operational".
- Die Daten werden mit der Zykluszeit *tCycleTime* zum Frequenzumrichter gesendet.



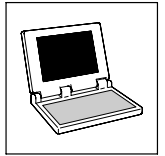
Tip!

Eine grundlegende Voraussetzung für die Prozeßdatenübertragung ist neben der korrekt einzustellenden Knotenadresse, daß sich der Frequenzumrichter im Zustand "Operational" befindet.

- Um dies sicherzustellen, sollte der Eingang *bResetNode* immer auf TRUE gesetzt sein.

Belegung der Sende- und Empfangsbytes

Sendebyte	Belegung	Empfangsbyte	Belegung
1 + 2	Steuerwort <i>wControlData</i>	1 + 2	Statuswort <i>wStatusData</i>
3 + 4	Sollwert <i>nNSetN1_a</i>	3 + 4	Istwert <i>nMCtrl_Nout_Slip</i>
5 + 6	$U_{\min}$ -Anhebung <i>wBoostVoltage</i>	5 + 6	frei belegbar über <i>wOutputData3</i>
7 + 8	frei belegbar über <i>wInputData</i>	7 + 8	frei belegbar über <i>wOutputData4</i>



Zeitüberwachung

Über den Eingang *tTimeOut* kann eine Zeitüberwachung für Empfangsdaten vom Frequenzumrichter realisiert werden.

- Bei einer Zeit $> t\#0ms$ ist die Überwachung aktiviert, andernfalls ist die Überwachung abgeschaltet.
- Die Überwachungszeit ist auf die Zyklus- und Taskzeit des Prozeßdatenobjektes **CAN2_IO** abzustimmen.

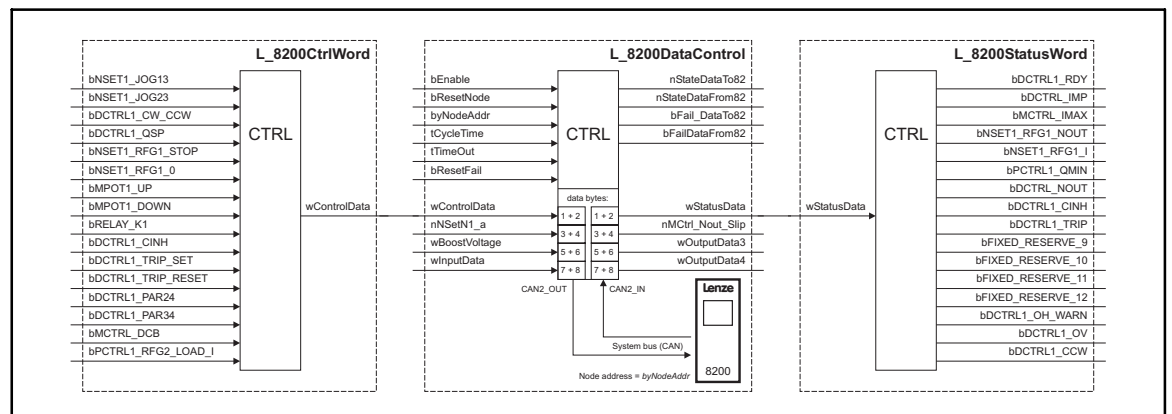
Statusmeldungen

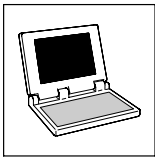
Die Statusausgänge *nStateDataTo82* und *nStateDataFrom82* zeigen den aktuellen Zustand der Sende- und Empfangsobjekte an.

- Im Fehlerfall wird über diese Statusausgänge eine Fehlernummer angezeigt und der entsprechende Meldeausgang *bFail_DataTo82* bzw. *bFail_DataFrom82* wird auf TRUE gesetzt.
- Durch Setzen von *bReset* auf TRUE können die Status- und Meldeausgänge zurückgesetzt werden.
- Die letzte Melde- bzw. Fehlernummer wird in einer internen Variable (*nLastMessageNr* bzw. *nLastFailureNr*) abgelegt.

Vorgabe boolescher Steuersignale und Ausgabe boolescher Statusmeldungen

- Mit Hilfe des FBs **L_8200CtrlWord** können Sie das zu sendende Steuerwort in Form boolescher Steuersignale vorgeben. (☞ 3-14)
- Mit Hilfe des FBs **L_8200StatusWord** können Sie das empfangende Statuswort zur Auswertung und Weiterverarbeitung in einzelne boolesche Statussignale aufteilen. (☞ 3-15)





Funktionsbibliothek Lenze8200Drive.lib

Funktionen/Funktionsblöcke

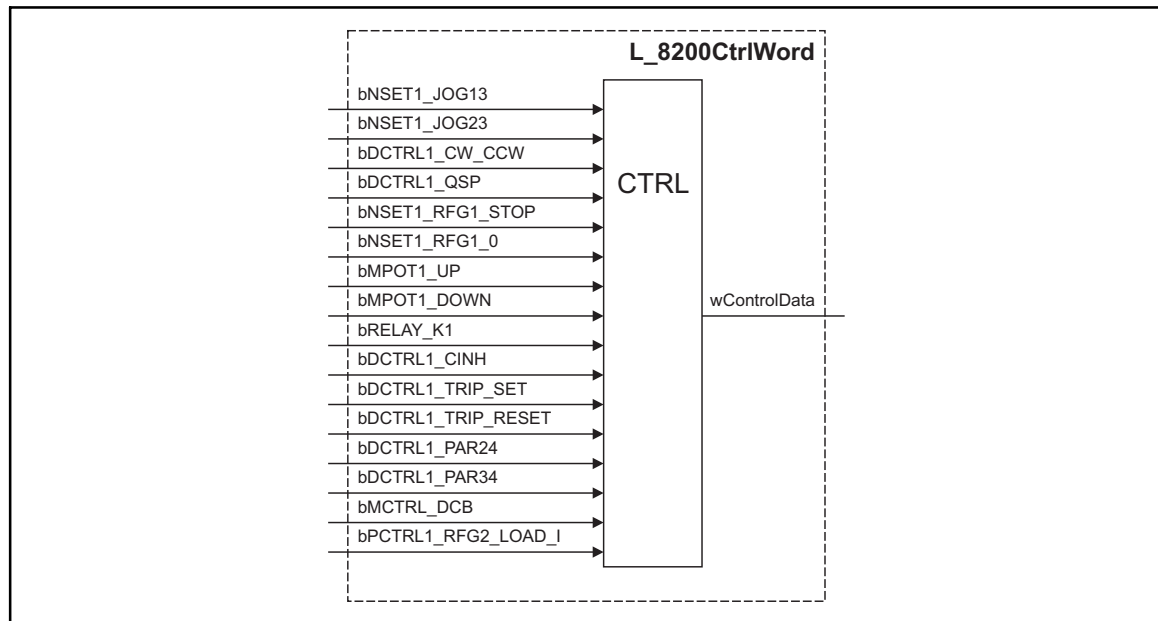
L_8200CtrlWord - Vorgabe boolescher Steuersignale

3.4

L_8200CtrlWord - Vorgabe boolescher Steuersignale

Funktionsblock

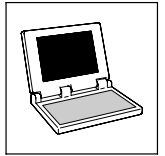
Mit diesem FB kann das zum Frequenzumrichter zu sendende Steuerwort, das der FB **L_8200DataControl** an seinem Eingang *wControlData* erwartet, in Form einzelner boolescher Steuersignale vorgegeben werden.



FB-Aufruf in:	☒ Zyklischer Task (PLC_PRG)	☒ Zeitgesteuerter Task (INTERVAL)	☒ Ereignisgesteuerter Task (EVENT)	☐ Interrupt-Task
----------------------	---	---	--	---

Eingänge	Datentyp	Info/Einstellmöglichkeiten
bNSET1_JOG13	Bool	Ausführliche Informationen zu der Bedeutung der einzelnen Steuersignale finden Sie in der Betriebsanleitung zum Frequenzumrichter Reihe 8200. Hinweis! Die Zusammensetzung des Steuerwortes ist von der Parametrierung des Frequenzumrichters abhängig! Die hier aufgeführten Steuersignale sind von der Namensgebung auf die auskommentierten Zuweisungen im FB L_8200Parameter abgestimmt. (📖 3-1)
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	

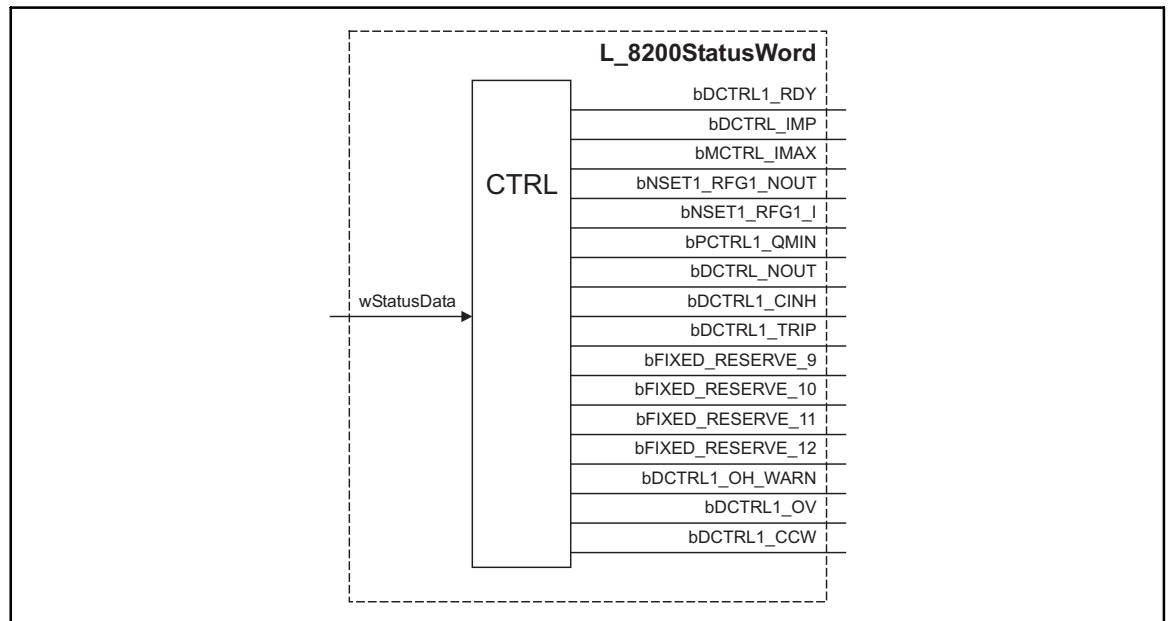
Ausgänge	Datentyp	Wert/Bedeutung
wControlData	Word	Steuerwort, das zum Frequenzumrichter gesendet werden soll. Verbinden Sie diesen Ausgang mit dem Eingang <i>wControlData</i> des FB L_8200DataControl. (📖 3-9)



3.5 L_8200StatusWord - Ausgabe boolescher Statussignale

Funktionsblock

Mit diesem FB kann das vom Frequenzumrichter empfangene Statuswort, das der FB **L_8200DataControl** an seinem Ausgang *wStatusData* zur Verfügung stellt, in einzelne boolesche Statussignale aufgeteilt werden.



FB-Aufruf in:	☒ Zyklischer Task (PLC_PRG)	☒ Zeitgesteuerter Task (INTERVAL)	☒ Ereignisgesteuerter Task (EVENT)	☐ Interrupt-Task
----------------------	---	---	--	---

Eingänge	Datentyp	Info/Einstellmöglichkeiten
wStatusData	Word	Statuswort, das vom Frequenzumrichter empfangen wurde. Verbinden Sie diesen Eingang mit dem Ausgang <i>wStatusData</i> des FB L_8200DataControl. (□ 3-9)

Ausgänge	Datentyp	Wert/Bedeutung
bDCTRL1_RDY	Bool	Ausführliche Informationen zu der Bedeutung der einzelnen Statussignale finden Sie in der Betriebsanleitung zum Frequenzumrichter Reihe 8200. Hinweis! Die Zusammensetzung des Statuswortes ist von der Parametrierung des Frequenzumrichters abhängig! Die hier aufgeführten Statussignale sind von der Namensgebung auf die auskommentierten Zuweisungen im FB L_8200Parameter abgestimmt. (□ 3-1)
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	