BAUMÜLLER

Optionskarte CAN-Master für Ωmega Drive-Line II CAN-M-01

Technische Beschreibung
und Betriebsanleitung

Stand: Juli 2001

D	5.00031.02

BAUMÜLLER

ΩMEGA OPTIONSKARTE CAN-MASTER FÜR DRIVE-LINE II
CAN-M-01

Technische Beschreibung und Betriebsanleitung

Stand: Juli 2001

5.00031.02

Diese Betriebsanleitung ist nur als Ergänzung der Technischen Beschreibung und Betriebsanleitung des zugehörigen Grundgeräts zu verstehen.

VOR INBETRIEBNAHME DIE BETRIEBSANLEITUNG UND SICHERHEITSHINWEISE LESEN UND BEACHTEN

Diese Betriebsanleitung enthält die erforderlichen Informationen für den bestimmungsgemäßen Gebrauch der darin beschriebenen Produkte. Sie wendet sich an technisch qualifiziertes Personal, welches speziell ausgebildet ist und gründlich mit allen Warnungen und Instandhaltungsmaßnahmen vertraut ist. Die Geräte sind nach dem Stand der Technik gefertigt und betriebssicher. Sie lassen sich gefahrlos installieren, in Betrieb setzen und funktionieren problemlos, wenn sichergestellt ist, daß die Sicherheitshinweise beachtet werden.

Die Inbetriebnahme ist solange untersagt, bis festgestellt wurde, daß die Maschine, in die diese Komponente eingebaut ist, den Bestimmungen der EG-Maschinenrichtlinie entspricht.

Mit der Übergabe der vorliegenden technischen Beschreibung und Betriebsanleitung werden frühere Beschreibungen des entsprechenden Produktes außer Kraft gesetzt. Die Firma Baumüller behält sich vor, im Rahmen der eigenen Weiterentwicklung der Produkte die technischen Daten und ihre Handhabung von Baumüller-Produkten zu ändern.

Hersteller- und Lieferadresse:
Baumüller Nürnberg GmbH
Ostendstr. 80
90482 Nürnberg
Tel. 09 11/54 32 - 0
Telefax 09 11/54 32 - 1 30

Ursprungsland: Deutschland

Herstelldatum: ersichtlich aus der Fabrikationsnummer des Geräts

INHALTSVERZEICHNIS

1 SICHERHEITSHINWEISE

Allgemeine Hinweise

Diese Betriebsanleitung enthält die erforderlichen Informationen für den bestimmungsgemäßen Gebrauch der darin beschriebenen Produkte. Sie wendet sich an technisch qualifiziertes Personal, welches speziell ausgebildet ist und gründlich mit allen Warnungen und Instandhaltungsmaßnahmen vertraut ist.

Die Einheiten sind nach dem Stand der Technik gefertigt und betriebssicher. Sie lassen sich gefahrlos installieren und in Betrieb setzen und funktionieren problemlos, wenn sichergestellt ist, daß die Hinweise der Betriebsanleitung beachtet werden.

Gefahrenhinweise

Die Hinweise dienen einerseits der persönlichen Sicherheit des Anwenders und andererseits der Sicherheit vor Beschädigung der beschriebenen Produkte oder angeschlossenen Geräte.

Die verwendeten Begriffe haben im Sinne der Betriebsanleitung und der Hinweise auf den Produkten selbst folgende Bedeutung:

GEFAHR

Bedeutet, daß **Tod**, **schwere Körperverletzung** oder **erheblicher Sachschaden** eintreten **werden**, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

WARNUNG

bedeutet, daß **Tod**, **schwere Körperverletzung** oder **erheblicher Sachschaden** eintreten **können**, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

HINWEIS

ist eine **wichtige Information** über das Produkt, die Handhabung des Produktes oder den jeweiligen Teil der Dokumentation, auf den besonders aufmerksam gemacht werden soll.

Qualifiziertes Personal

Qualifiziertes Personal im Sinne der sicherheitsbezogenen Hinweise in dieser Betriebsanleitung oder auf den Produkten selbst sind Personen, die mit Montage, Inbetriebsetzung und Betrieb des Produktes vertraut sind und über die ihrer Tätigkeit entsprechenden Qualifikation verfügen:

- Ausbildung oder Unterweisung bzw. Berechtigung Stromkreise und Geräte gemäß den Standards der Sicherheitstechnik in Betrieb zu nehmen, zu erden und zu kennzeichnen.
- Ausbildung oder Unterweisung gemäß den Standards der Sicherheitstechnik in Pflege und Gebrauch angemessener Sicherheitsausrüstung.

Bestimmungsgemäßer Gebrauch

WARNUNG

Die Einheit / das System darf nur für die in der Betriebsanleitung vorgesehenen Einsatzfälle und nur in Verbindung mit von der BAUMÜLLER NÜRNBERG GmbH empfohlenen bzw. zugelassenen Fremdgeräten und -komponenten verwendet werden.

Eigenmächtige Umbauten und Veränderungen an der Einheit sind aus Sicherheitsgründen nicht gestattet. Der Bediener ist verpflichtet, eintretende Veränderungen, die die Sicherheit der Einheit / des Systems beeinträchtigen könnten, sofort zu melden.

2 TECHNISCHE DATEN

2.1 Allgemeines

In einer Ωmega Drive-Line II ist ein CANsync-Master-Knoten integriert. Für die Nutzung des CAN ist die zusätzliche CAN-Master-Baugruppe CAN-M-01 vorzusehen.

Die Optionskarte CAN-M-01 ermöglicht

- die Kommunikation mit bis zu 32 CAN-Teilnehmern
  - davon maximal 16 I/O-Module bei 125 kBit/s
  - davon maximal 8 I/O-Module bei 250 kBit/s
  - davon maximal 16 Regler mit CAN-Interface
  - davon maximal 16 Ωmegas mit CAN-Interface
  - davon maximal 16 Absolutwertgeber

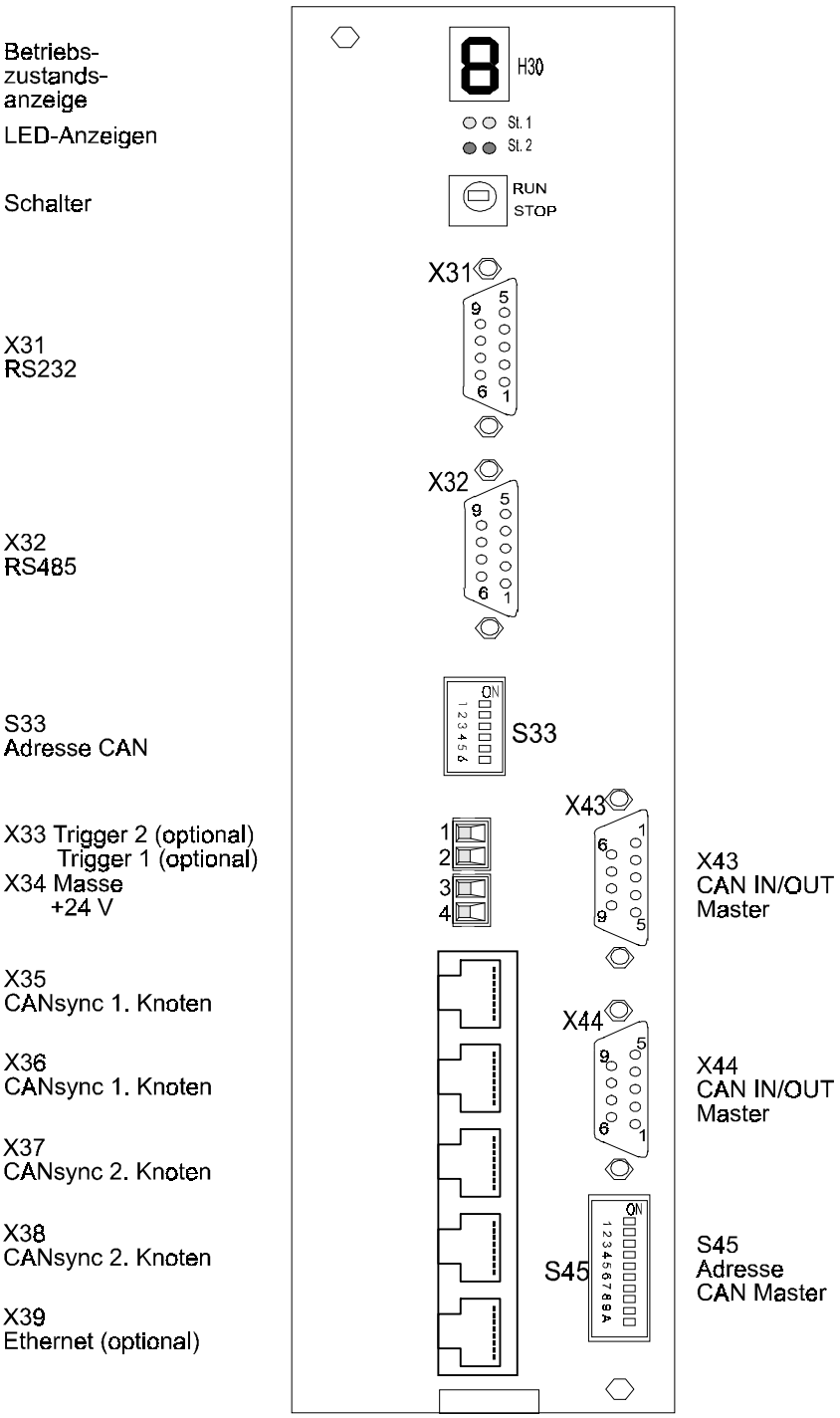
2.2 Technische Daten CAN-Master

CPU	8-Bit CPU 16 MHz
Betriebsspannung	+5 V
Stromaufnahme	max. 1A
Umgebungstemperatur	0 ... 55°C
Lagertemperatur	-15 ... 70°C
Luftfeuchtigkeit	max. 90%
Speicher	32 kByte RAM, 64 kByte EPROM
Ankopplung an Ωmega Drive-Line II	Dual Port Ram 2k x 16
CAN-Controller	SJA1000T
Physical Layer	ISO 11898
Baudrate	max. 1 Mbit/s
Potentialtrennung	Optokoppler, DC/DC-Wandler
Steckerverbindung zum CAN-Bus	SUB-D-Stecker und -Buchse 9-polig

3 INSTALLATION

3.1 Kontrollen vor dem Einschalten

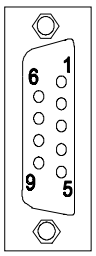
- Anschluß der Stecker



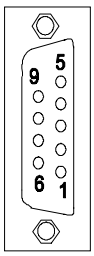
3.2 Steckerbelegung

- CAN-IN / -OUT

X 43 SUB-D-Stiftleiste
X 44 SUB-D-Buchse



X 43



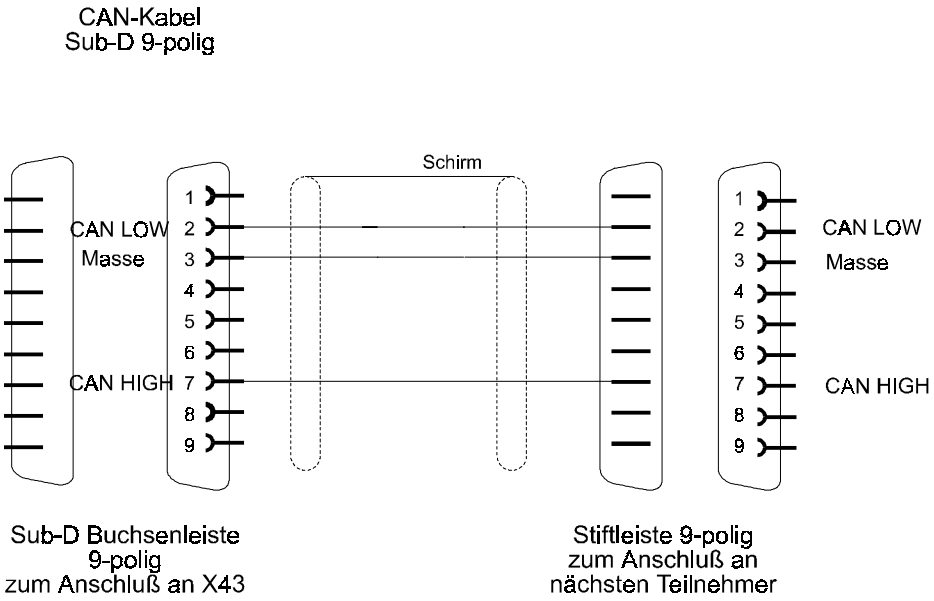
X 44

Pin Nr.	Belegung
1	reserviert
2	CAN LOW bus line (dominant low)
3	GND Ground
4	reserviert
5	reserviert
6	reserviert
7	CAN HIGH bus line (dominant high)
8	reserviert
9	reserviert

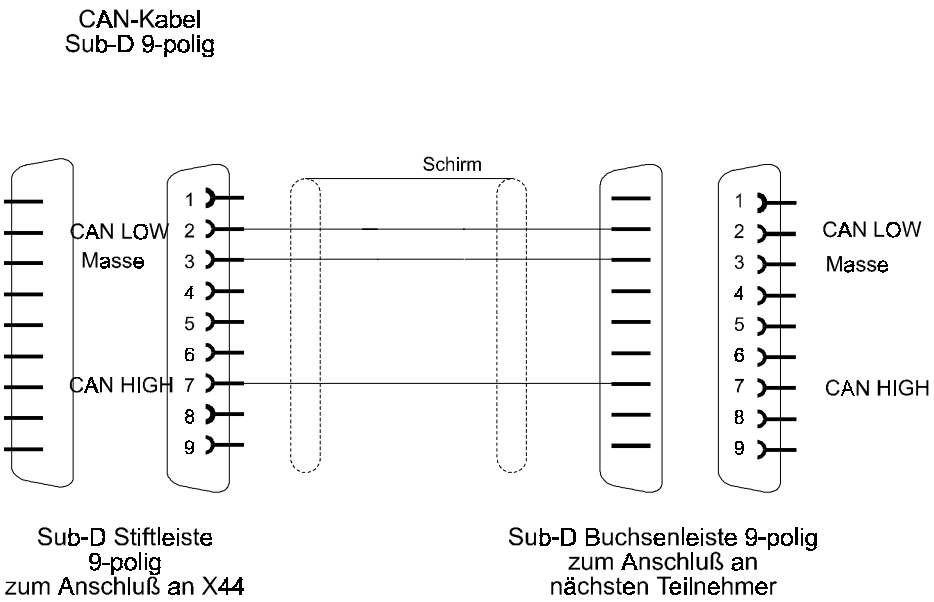
3.3 Anschlußkabel

Anschlußkabel für weitere CAN-Teilnehmer

- Anschluß 9-polig für X43



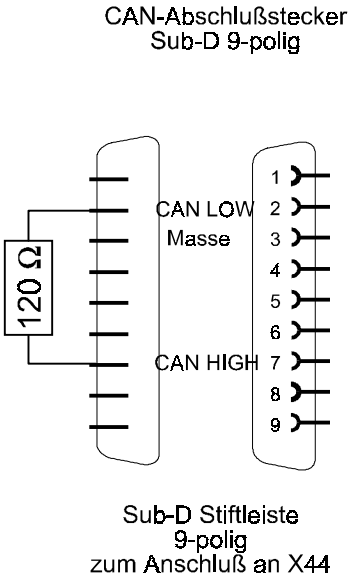
- Anschluß 9-polig für X44



3.4 Zubehör

Abschlußstecker 120 Ω

- Anschluß 9-polig



4 CAN-KNOTEN

Als Controller wird der Basic-CAN-Controller SJA1000T eingesetzt. Durch die integrierte Firmware kann ein Full-CAN-Controller mit einer erheblich erweiterten Objekthaltung nachgebildet werden. Die Busankopplung erfolgt nach ISO 11898 (CAN High-Speed).

Die Busankopplung ist über schnelle Optokoppler galvanisch getrennt und wird durch einen lokalen DC/DC-Wandler versorgt.

Merkmale

- bis zu 16 Antriebe (mit Baumüller CAN-Interface-Anschaltung)
- bis zu 16 Absolutwertgeber
- bis zu 16 dezentrale I/O-Module
- bis zu 15 weitere Ωmega-CAN-Anschaltungen

bedienbar über integrierte Firmware.

CAN Eigenschaften

- Serielles asynchrones Bussystem
- Echtzeitfähig (max. 1 Mbit/s bei 40 m Busausdehnung)
- Broadcast/Multicast und point to point Kommunikation
- Leistungsstarke Fehlererkennung und -behandlung
- Hohe Zuverlässigkeit (Hamming-Distanz = 6)
- Multimaster
- Prioritätsgesteuerte Buszuteilung
- Garantierte maximale Latenzzeit für hoch priorisierte Nachrichten
- Offenes System
- International standardisiert

4.1 Hinweise zu Programmierung

Zur Programmierung des CAN stehen innerhalb eines PROPROG wt II Projekts unter IEC 61131-3 die Funktionsbausteine aus der Bibliothek CAN_DLII_20bd00 (oder höher) zur Verfügung (siehe “Funktionsbausteine für CAN” auf Seite 53).

Basisadresse

Basisadresse Optionskarte CAN-Master CAN-M-01	%MB3.3000000

Initialisierung

Die Initialisierung der CAN-Anschaltung ist nur nach einem Systemstart (Reset oder Power up) nötig. Es sind folgende FBs im Kalt- und Warmstart aufzurufen:

FB CAN_DIG_INPUT_INIT
FB CAN_DIG_OUTPUT_INIT
FB CAN_DRIVE_INIT
FB CAN_INIT

Die Initialisierung erfolgt mit dem FB CAN_INIT. Die FBs CAN_DIG_INPUT_INIT, CAN_DIG_OUTPUT_INIT und CAN_DRIVE_INIT dienen der einfachen Beschaltung einiger Eingänge des FB CAN_INIT.

Prozeßdatenkommunikation

Für die Prozeßdatenkommunikation wird der FB CAN_PD_COMM in einer zyklischen Task verwendet.

Bedarfsdatenkommunikation

Für die Bedarfsdatenkommunikation stehen die FBs

FB CAN_OBJ_READ
FB CAN_OBJ_WRITE
FB CAN_PA_LIST_READ
FB CAN_PA_LIST_WRITE
FB CAN_PAR_READ
FB CAN_PAR_WRITE
FB CAN_PE_LIST_READ
FB CAN_PE_LIST_WRITE
FB CAN_SD_CONTROL

zur Verfügung.

Im Folgenden wird die Belegung des Kommunikations-RAM im Ωmega für die CAN-M-01-Anschaltung erläutert. Für den Zugriff auf die einzelnen Register im Kommunikations-RAM wurden Strukturen geschaffen, die den komfortablen Zugriff auf das Kommunikations-RAM ermöglichen.

Bei der Initialisierung der CAN-M-01-Anschaltung haben die Register im Kommunikations-RAM eine andere Bedeutung als beim zyklischen Betrieb.

Deshalb gibt es für die Initialisierung die Struktur (STRUCT)

CAN_INIT_BMSTRUCT

und für den zyklischen Betrieb die Struktur

CAN_CTRL_BMSTRUCT

Diese Strukturen enthalten

- 8-Bit-Elemente,
- 16-Bit-Elemente,
- 32-Bit-Elemente,
- Strukturen aus den o.g. Elementen
- Felder (ARRAY) aus den o.g. Elementen und Strukturen

Den in einer Struktur verwendeten Datentypen (8-, 16-, 32-Bit-Elemente, Strukturen und Feldern) wurden Kurzbezeichnungen vorangefügt. Dies dient der Übersichtlichkeit bei der Verwendung der Strukturen in der Programmierung.

Datentyp	**Kurzbezeichnung**	**Anzahl der Bits**
BYTE	b	8
WORD	w	16
DWORD (double word)	d	32
SINT (short integer)	si	8
INT (integer)	i	16
DINT (double integer)	di	32
USINT (unsigned short integer)	us	8
UINT (unsigned integer)	u	16
UDINT (unsigned double integer)	ud	32
STRUCT	_ (underline)	-
ARRAY	a	-

Weitere, nicht in den Strukturen verwendete Datentypen sind:

Datentyp	**Kurzbezeichnung**	**Anzahl der Bits**
BOOL (bit)	x	1
TIME	t	-

Erläuterung zur Deklaration der globalen Variablen

Die Strukturen werden bei der Programmierung in PROPROG wt II der Basisadresse des jeweiligen Kommunikations-RAM für eine CAN-Master-Anschaltung zugeordnet.

Für die Initialisierung wird eine globale Variable vom Typ CAN_INIT_BMSTRUCT angelegt, die auf die Basisadresse der CAN-M-01-Anschaltung fixiert wird.

z. B. `_CAN_INIT_OPT AT %MB3.3000000 : CAN_INIT_BMSTRUCT;`

für den CAN-Knoten auf der Optionskarte CAN-M-01 am Ωmega Drive-Line II,

Dabei ist z. B.

`_CAN_INIT_OPT`	der Variablenname mit der Datentypkurzbezeichnung „_“ für STRUCT
`CAN_INIT_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-M-01-Anschaltung am Ωmega Drive-Line II.

Für den zyklischen Betrieb wird eine globale Variable vom Typ CAN_CTRL_BMSTRUCT angelegt, die auf die Basisadresse der CAN-Master-Anschaltung fixiert wird.

z. B. `_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;`

für den CAN-Knoten auf der Optionskarte CAN-M-01 am Ωmega Drive-Line II,

Dabei ist z. B.

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung „_“ für STRUCT
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-M-01-Anschaltung am Ωmega Drive-Line II.

In den nachfolgenden Tabellen wird der Variablenname durch * ersetzt.

Auf das Register `*.w_CPU_CONTROL` greift man demzufolge über

`_CAN_INIT_OPT.w_CPU_CONTROL` zu,

auf `*.w_OPTION_STATUS` greift man über

`_CAN_INIT_OPT.w_OPTION_STATUS` zu.

Dabei ist z. B.

`_CAN_INIT_OPT`	der Variablenname mit der Datentypkurzbezeichnung „_“ für STRUCT
`w_CPU_CONTROL`	das Steuerregister mit der Datentypkurzbezeichnung „w“ für WORD

Die Register *.w_CPU_CONTROL und *.w_OPTION_STATUS können auch über die Struktur für den zyklischen Betrieb angesprochen werden. Der Zugriff ist dann über

_CAN_CTRL_OPT.w_CPU_CONTROL und

_CAN_CTRL_OPT.w_OPTION_STATUS möglich.

Dabei ist z. B.

_CAN_CTRL_OPT	der Variablenname mit der Datentypkurzbezeichnung „_“ für STRUCT
w_OPTION_STATUS	das Statusregister mit der Datentypkurzbezeichnung „w“ für WORD

Beispiel für den Zugriff auf ein Element eines Feldes, das in der Struktur verwendet wird:

lt. Tabelle: *.a_VALUE_DI_0_15[3]
Zugriff: _CAN_CTRL_OPT.a_VALUE_DI_0_15[3]

Dabei ist

_CAN_CTRL_OPT	der Variablenname mit der Datentypkurzbezeichnung „_“ für STRUCT
a_VALUE_DI_0_15[3]	das Register für das Eingangsabbild des Digitalen Eingangsmoduls CAN-DI-16-01 mit der CAN-Node-ID 3 (-> [3]) und mit der Datentypkurzbezeichnung „a“ für ARRAY. Der Datentyp der Elemente des Feldes (des Eingangsabbildes) wird der entsprechenden Tabelle und der Beschreibung entnommen.

Beispiel für den Zugriff auf ein Element einer (Sub-) Struktur, die selbst Element eines Feldes ist, das in der Struktur verwendet wird:

lt. Tabelle: *.a_FIFO_0_31[31].w_ID
Zugriff: _CAN_CTRL_OPT.a_FIFO_0_31[31].w_ID

Dabei ist

_CAN_CTRL_OPT	der Variablenname mit der Datentypkurzbezeichnung „_“ für STRUCT
a_FIFO_0_31[31]	das Feld mit den Empfangsgsdaten des FIFO-Buffers für das am Eintrag 31 eingetragene Telegramm mit der Datentypkurzbezeichnung „a“ für ARRAY
w_ID	das Register für den Identifier des am Eintrag 31 eingetragenen Telegramms mit der Datentypkurzbezeichnung „w“ für WORD

Allgemeine Register der CAN-Anschaltung

Register	Inhalt
*.w_CAN_STATUS	CAN-Status
*.w_OMEGA_NR	über Dip-Schalter eingestellte Ωmega-Nummer
*.i_SW1_NR	Karten-Softwarenummer
*.i_SW1_RELEASE	Softwarestand inkompatibel und kompatibel

CAN-Status

Bei jedem Durchlaufen des CAN-Prozessorzyklusses wird der CAN-Status auf *.w_CAN_STATUS ausgegeben.

Bedeutung:

Bit-Nr.	Bedeutung (Bit = TRUE)
0	reserviert
1	Overrun: eine CAN-Nachricht konnte nicht empfangen werden
2	CAN-Sendepuffer ist frei
3	CAN-Sendeauftrag wurde erfolgreich durchgeführt
4	es wird gerade eine CAN-Meldung empfangen
5	es wird gerade eine CAN-Meldung gesendet
6	Fehler vorhanden (Warnung)
7	CAN-Knoten ist deaktiviert (BUS-off)
8-15	reserviert

Ωmega-Nummer

Im Register *.w_OMEGA_NR wird die über die DIP-Schalter (S45) eingestellte Ωmega-Nummer angezeigt. Bei einer CAN-Anschaltung ist diese Nummer die Ωmega-Nummer.

Softwarenummer und Softwarestand

Im Register *.i_SW1_NR wird die Nummer der CAN-Software auf dem Ωmega Drive-Line II angezeigt.

Im Register *.i_SW1_RELEASE wird der inkompatible und der kompatible Stand der CAN-Software auf dem Ωmega Drive-Line II angezeigt.

Über das Register *.w_CPU_CONTROL erfolg die Steuerung der CAN-Anschaltung. Die Anzeige des aktuellen Zustands erfolgt im Register *.w_OPTION_STATUS.

Register	Inhalt
*.w_CPU_CONTROL	Steuerregister CAN-Anschaltung
*.w_OPTION_STATUS	Statusregister CAN-Anschaltung

(* entspricht zum Beispiel bei der Initialisierung _CAN_INIT_OPT, nach der Initialisierung zum Beispiel _CAN_CTRL_OPT)

Steuerregister CAN-Anschaltung	Bedeutung
16#0000	Neuanlauf
16#0001	Test Handshake
16#0002	Initialisierungsdaten übernehmen
16#0080	(Bit 7 = TRUE) Reset CAN-Controller
16#0100	Betrieb starten

Statusregister CAN-Anschaltung	Bedeutung
16#0001	Anlauf
16#0002	Warten auf Initialisierungsdaten übernehmen
16#0003	Warten auf Start
16#0100	Betrieb ist gestartet

4.2 Konfigurierung

4.2.1 Bus-Timing

Durch das Anschließen der Eingänge b_BIT_TIMING0 und b_BIT_TIMING1 am FB CAN_INIT kann das Bus-Timing des CAN-Controllers SJA1000T individuell initialisiert werden.

Bus Timing Register b_BIT_TIMING0							
7	6	5	4	3	2	1	0
SJW.1	SJW.0	BRP.5	BRP.4	BRP.3	BRP.2	BRP.1	BRP.0

Bus Timing Register b_BIT_TIMING1							
7	6	5	4	3	2	1	0
SAM	TSEG2.2	TSEG2.1	TSEG2.0	TSEG1.3	TSEG1.2	TSEG1.1	TSEG1.0

Im Bus-Timing Register 0 werden die Werte für Baud Rate Prescaler *(BRP)* und Synchronization Jump Width (*SJW*) festgelegt.

Daraus läßt sich die Periodendauer des Systemtakts bestimmen:

$t_{SCL} = 2t_{CLK}$ (32BRP.5 + 16BRP.4 + 8BRP.3 + 4BRP.2 + 2BRP.1 + BRP.0 + 1)

t_{CLK} = 62,5ns

Die Synchronisationssprungweite SJW (Synchronization Jump Width) läßt sich folgendermaßen bestimmen:

$t_{SJW} = t_{SCL}$ (2SJW.1 + SJW.0 + 1)

Im Bus-Timing Register 1 wird die Länge der Time Segmente 1 (*TSEG1*) und 2 (*TSEG2*), sowie die Anzahl der Abtastungen pro Bit (*SAM*) festgelegt.

$t_{SEG1} = t_{SCL}$ (8TSEG1.3 + 4TSEG1.2 + 2TSEG1.1 + TSEG1.0 + 1)

$t_{SEG2} = t_{SCL}$ (4TSEG2.2 + 2TSEG2.1 + TSEG2.0 + 1)

Anzahl der Abtastungen:

Bit	Wert	Bedeutung
SAM	1	3 Abtastungen
	0	1 Abtastung (Vorzugsinit.)

Bus-Timing (Definitionen)

– ISO

Sync_Seg	Prop_Seg	Phase_Seg1	Phase_Seg2
1	1-8	1-8	1-8
8-25 * t_{SCL}			

– SJA1000T

Sync_Seg	TSEG1	TSEG2
1	2-16	2-8
5-25 * t_{SCL}		

- Sync_SEG Synchronisations Segment
- TSEG1 entspricht Prop_Seg + Phase_Seg1;
  definiert Lage Abtastzeitpunkt
- TSEG2 entspricht Phase_Seg2;
  Synchronisationspuffer;
- N_{SJW} Synchronization Jump Width;
  Anzahl von Bitzeiteinheiten (1-4), um die die Bitzeit bei Nachsynchronisation verkürzt bzw. verlängert werden kann.

4.2.2 Akzeptanzfilter

Der Akzeptanzfilter des CAN-Controllers SJA1000T wird über die beiden Eingänge b_ACCEPT_MASK (AM) und b_ACCEPT_CODE (AC) am FB CAN_INIT eingestellt.

Acceptance Code Register ACR							
7	6	5	4	3	2	1	0
AC.7	AC.6	AC.5	AC.4	AC.3	AC.2	AC.1	AC.0

Acceptance Mask Register AMR							
7	6	5	4	3	2	1	0
AM.7	AM.6	AM.5	AM.4	AM.3	AM.2	AM.1	AM.0

Die Bits AC.7 - AC.0 im Acceptance Code Register und die 8 höchstwertigen Identifierbits Id10 - Id3 eines Objekts müssen an den Bitpositionen gleich sein, die im Acceptance Mask Register als relevant gekennzeichnet sind.

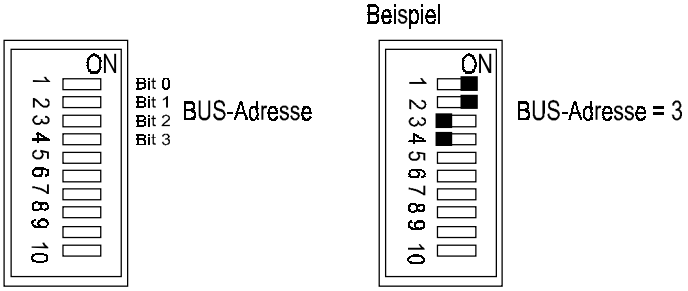
Acceptance Mask Bit	**Wert**	**Bedeutung**
AM.7 - AM.0	1	diese Bitposition ist "don't care" für die Akzeptanz eines Objekts
	0	diese Bitposition ist "relevant" für die Akzeptanz eines Objekts

Da durch diese Akzeptanzfilterung nur die Identifierbits Id10 - Id3 berücksichtigt werden, ist der Filter immer für mindestens 8 Objekte durchlässig.

4.2.3 Einstellung der Bus-Adresse

(ab Softwarestand 3.01)

Über die DIP-Schalter 1 - 4 (S45 auf der CAN-M-01) wird die Bus-Adresse binär codiert eingestellt mit der das Ωmega angesprochen werden kann. Für jedes Ωmega an einem CAN-Bus muß eine von den anderen verschiedene Adresse (Ωmega-Nummer, auch CAN-Node-ID) eingestellt werden.



4.3 Betrieb dezentraler I/O Module

4.3.1 Adressierung

Belegung Registerbereich Knoten1 (Auszug):

Adresse / Register	Bedeutung
*.w_DIG_IN_LIST	Dig_In_List
*.w_DIG_OUT_LIST	Dig_Out_List
*.a_CTRL_IO_0_15[0]	Steuerregister IO-Modul 0
*.a_CTRL_IO_0_15[1]	Steuerregister IO-Modul 1
*.a_CTRL_IO_0_15[2]	Steuerregister IO-Modul 2
•	•
•	•
•	•
*.a_CTRL_IO_0_15[14]	Steuerregister IO-Modul 14
*.a_CTRL_IO_0_15[15]	Steuerregister IO-Modul 15
*.a_VALUE_DI_0_15[0]	Eingangsabbild Digital-Input-Modul 0
*.a_VALUE_DI_0_15[1]	Eingangsabbild Digital-Input-Modul 1
*.a_VALUE_DI_0_15[2}	Eingangsabbild Digital-Input-Modul 2
•	•
•	•
•	•
*.a_VALUE_DI_0_15[14]	Eingangsabbild Digital-Input-Modul 14
*.a_VALUE_DI_0_15[15]	Eingangsabbild Digital-Input-Modul 15
*.a_VALUE_DO_0_15[0]	Ausgang Digital-Output-Modul 0
*.a_VALUE_DO_0_15[1]	Ausgang Digital-Output-Modul 1
*.a_VALUE_DO_0_15[2]	Ausgang Digital-Output-Modul 2
•	•
•	•
•	•
*.a_VALUE_DO_0_15[14]	Ausgang Digital-Output-Modul 14
*.a_VALUE_DO_0_15[15]	Ausgang Digital-Output-Modul 15

* entspricht zum Beispiel _CAN_CTRL_OPT.

4.3.2 Kommunikation

*.w_DIG_IN_LIST: **Dig_In_List**

Statusregister, das anzeigt, an welcher Knotennummer sich ein digitales Eingangsmodul befindet. Es kann zur Überprüfung der Initialisierung benutzt werden. Dabei entspricht Bit 0 Modul 0 und Bit 15 Modul 15. Wenn das entsprechende Bit „TRUE“ ist, handelt es sich bei dem Modul mit der entsprechenden Knotennummer um ein CAN-DI-16 Modul.

*.w_DIG_OUT_LIST: **Dig_Out_List**

Statusregister, das anzeigt, an welcher Knotennummer sich ein digitales Ausgangsmodul befindet. Es kann zur Überprüfung der Initialisierung benutzt werden. Dabei entspricht Bit 0 Modul 0 und Bit 15 Modul 15. Wenn das entsprechende Bit „TRUE“ ist, handelt es sich bei dem Modul mit der entsprechenden Knotennummer um ein CAN-DO-16 Modul.

*.a_CTRL_IO_0_15[0] bis *.a_CTRL_IO_0_15[15]: **Steuerregister für dezentrale I/O-Module 0 - 15 (siehe unten)**

Nach durchlaufener Initialisierung wird angezeigt, ob sich das Modul richtig gemeldet hat. Die Einträge sind vom Datentyp BYTE.

16#20:kein Modul (Modul fehlt, fehlende Spannungsversorgung, CAN-Bus nicht gesteckt)

16#00:Modul mit entsprechender Adresse vorhanden.

*.a_VALUE_DI_0_15[0] bis *.a_VALUE_DI_0_15[15]: **Datenbereich digitale Eingangsmodule**

Bereich, in dem die Eingangszustände der digitalen Eingangsmodule abgebildet werden. An Adressen, an denen sich kein bzw. ein Modul anderen Typs befindet, ist der Inhalt 0. Die Einträge sind vom Datentyp WORD.

*.a_VALUE_DO_0_15[0] bis *.a_VALUE_DO_0_15[15]: **Datenbereich digitale/Relais Ausgangsmodule**

Bereich, in den die Ausgangswörter der digitalen bzw. Relais Ausgangsmodule geschrieben werden. An Adressen, an denen sich kein bzw. ein Modul anderen Typs befindet, werden eventuell eingetragene Daten ignoriert. Die Einträge sind vom Datentyp WORD.

Digital Input (CAN-DI 16)

Der FB CAN_DIG_INPUT_INIT bietet die Möglichkeit zur Einstellung verschiedener Betriebsarten.

- Betriebsart 1:
  nicht implementiert

- Betriebsart 2:
  Anfordern von Eingangsdaten über das Steuerregister.
  Durch das Beschreiben des zum jeweiligen Eingangsmodul gehörenden Steuerregister mit 16#01 wird das Eingangswort durch ein Remote-Frame angefordert.
  Stehen die gewünschten Daten an der entsprechenden Adresse im Eingangsdatenbereich, wird dies durch 16#02 im Steuerregister angezeigt.
- Betriebsart 3:
  Zyklisches Anfordern von Eingangsdaten. Die Zykluszeit ist bei der Initialisierung festzulegen. Die gewünschten Daten stehen an der entsprechenden Adresse im Eingangsdatenbereich.
  Es ist darauf zu achten, daß auch in dieser Betriebsart der Empfang des Eingangswortes mit 16#02 im Steuerregister des jeweiligen Moduls angezeigt wird.

HINWEIS

Unabhängig von der eingestellten Betriebsart, kann durch die beiden Register Eingang w_DI_POS_EDGE bzw. w_DI_NEG_EDGE ein selbständiges Melden des Moduls bei einer Änderung des Eingangszustandes erreicht werden (siehe Technische Beschreibung CAN-DI-16).

Digital/Relais Output (CAN-DO 16/CAN-DO 8R)

Der FB CAN_DIG_OUTPUT_INIT bietet die Möglichkeit zur Einstellung verschiedener Betriebsarten.

- Betriebsart 1:
  nicht implementiert
- Betriebsart 2:
  Senden von Ausgangsdaten über das Steuerregister.
  Durch das Beschreiben des zum jeweiligen Ausgangsmoduls gehörenden Steuerregister mit 16#03 wird das Ausgangswort auf das Modul geschrieben. Die gewünschten Daten müssen an der entsprechenden Adresse im Ausgangsdatenbereich zur Verfügung stehen.
  Das erfolgreiche Senden des Ausgangswortes wird durch 16#04 im Steuerregister angezeigt.
- Betriebsart 3:
  Zyklisches Senden von Ausgangsdaten. Die Zykluszeit ist bei der Initialisierung festzulegen. Die gewünschten Daten müssen an der entsprechenden Adresse im Ausgangsdatenbereich zur Verfügung stehen.
  Es ist darauf zu achten, daß auch in dieser Betriebsart das Senden des Ausgangswortes mit 16#04 im Steuerregister des jeweiligen Moduls angezeigt wird.

4.4 Betrieb von maximal 16 Reglern

4.4.1 Adressierung

Für jeden der maximal 16 Regler mit CAN-Anschaltung (im folgenden Regler bzw. Drive genannt) ist ein Registerbereich von 32 Worten im Kommunikations-RAM reserviert.

Belegung Registerbereich:

Adresse / Register	Bedeutung
*.a_DRIVE_0_15[0].w_D_CONTROLWORD	Drive 0 Steuerwort
*.a_DRIVE_0_15[0].w_D_STATUSWORT	Drive 0 Statuswort
*.a_DRIVE_0_15[0].w_D_DUMMY_0	Drive 0 reserviert
*.a_DRIVE_0_15[0].w_D_WRVALUE_0	Drive 0 Sollwert 0
*.a_DRIVE_0_15[0].d_D_WRVALUE_1	Drive 0 Sollwert 1
*.a_DRIVE_0_15[0].d_D_WRVALUE_2	Drive 0 Sollwert 2
*.a_DRIVE_0_15[0].d_D_WRVALUE_3	Drive 0 Sollwert 3
*.a_DRIVE_0_15[0].w_D_DUMMY_1	Drive 0 reserviert
*.a_DRIVE_0_15[0].w_D_RDVALUE_0	Drive 0 Istwert 0
*.a_DRIVE_0_15[0].d_D_RDVALUE_1	Drive 0 Istwert 1
*.a_DRIVE_0_15[0].d_D_RDVALUE_2	Drive 0 Istwert 2
*.a_DRIVE_0_15[0].d_D_RDVALUE_3	Drive 0 Istwert 3
*.a_DRIVE_0_15[0].w_D_SD_0	Drive 0 Bedarfsdaten 0
*.a_DRIVE_0_15[0].w_D_SD_1	Drive 0 Bedarfsdaten 1
*.a_DRIVE_0_15[0].w_D_SD_2	Drive 0 Bedarfsdaten 2
*.a_DRIVE_0_15[0].w_D_SD_3	Drive 0 Bedarfsdaten 3
*.a_DRIVE_0_15[0].b_D_CTRLREG_STATUSWORD	Drive 0 Steuerregister Statuswort
*.a_DRIVE_0_15[0].w_D_CTRLREG_CONTROLWORD	Drive 0 Steuerregister Steuerwort
*.a_DRIVE_0_15[0].w_D_DUMMY_2	Drive 0 reserviert
*.a_DRIVE_0_15[0].b_D_CTRLREG_WRVALUE_0	Drive 0 Steuerregister Sollwert 0
*.a_DRIVE_0_15[0].b_D_CTRLREG_WRVALUE_1	Drive 0 Steuerregister Sollwert 1
*.a_DRIVE_0_15[0].b_D_CTRLREG_WRVALUE_2	Drive 0 Steuerregister Sollwert 2
*.a_DRIVE_0_15[0].b_D_CTRLREG_WRVALUE_3	Drive 0 Steuerregister Sollwert 3
*.a_DRIVE_0_15[0].b_D_CTRLREG_RDVALUE_0	Drive 0 Steuerregister Istwert 0
*.a_DRIVE_0_15[0].b_D_CTRLREG_RDVALUE_1	Drive 0 Steuerregister Istwert 1
*.a_DRIVE_0_15[0].b_D_CTRLREG_RDVALUE_2	Drive 0 Steuerregister Istwert 2
*.a_DRIVE_0_15[0].b_D_CTRLREG_RDVALUE_3	Drive 0 Steuerregister Istwert 3
*.a_DRIVE_0_15[0].b_D_CTRLREG_SD	Drive 0 Steuerregister Bedarfsdaten
*.a_DRIVE_0_15[0].b_D_CC_REG	Drive 0 Kommunikationskontrolle Bedarfs-daten
*.a_DRIVE_0_15[0].w_D_ERROR_NR	Drive 0 Fehlernummer Bedarfsdaten
*.a_DRIVE_0_15[0].w_D_PARAMETER_NR	Drive 0 Parameternummer Bedarfsdaten
*.a_DRIVE_0_15[0].b_D_FORMAT	Drive 0 Format Bedarfsdaten
*.a_DRIVE_0_15[0].b_D_ELEMENT	Drive 0 Element Bedarfsdaten
*.a_DRIVE_0_15[1].w_D_CONTROLWORD	Drive 1 Steuerwort
*.a_DRIVE_0_15[1].w_D_STATUSWORT	Drive 1 Statuswort
*.a_DRIVE_0_15[1].w_D_DUMMY_0	Drive 1 reserviert
*.a_DRIVE_0_15[1].w_D_WRVALUE_0	Drive 1 Sollwert 0

Adresse / Register	Bedeutung
*.a_DRIVE_0_15[1].d_D_WRVALUE_1	Drive 1 Sollwert 1
*.a_DRIVE_0_15[1].d_D_WRVALUE_2	Drive 1 Sollwert 2
*.a_DRIVE_0_15[1].d_D_WRVALUE_3	Drive 1 Sollwert 3
*.a_DRIVE_0_15[1].w_D-DUMMY_1	Drive 1 reserviert
*.a_DRIVE_0_15[1].w_D_RDVALUE_0	Drive 1 Istwert 0
*.a_DRIVE_0_15[1].d_D_RDVALUE_1	Drive 1 Istwert 1
*.a_DRIVE_0_15[1].d_D_RDVALUE_2	Drive 1 Istwert 2
*.a_DRIVE_0_15[1].d_D_RDVALUE_3	Drive 1 Istwert 3
*.a_DRIVE_0_15[1].w_D_SD_0	Drive 1 Bedarfsdaten 0
*.a_DRIVE_0_15[1].w_D_SD_1	Drive 1 Bedarfsdaten 1
*.a_DRIVE_0_15[1].w_D_SD_2	Drive 1 Bedarfsdaten 2
*.a_DRIVE_0_15[1].w_D_SD_3	Drive 1 Bedarfsdaten 3
*.a_DRIVE_0_15[1].b_D_CTRLREG_STATUSWORD	Drive 1 Steuerregister Statuswort
*.a_DRIVE_0_15[1].b_D_CTRLREG_CONTROLWORD	Drive 1 Steuerregister Steuerwort
*.a_DRIVE_0_15[1].w_D_DUMMY_2	Drive 1 reserviert
*.a_DRIVE_0_15[1].b_D_CTRLREG_WRVALUE_0	Drive 1 Steuerregister Sollwert 0
*.a_DRIVE_0_15[1].b_D_CTRLREG_WRVALUE_1	Drive 1 Steuerregister Sollwert 1
*.a_DRIVE_0_15[1].b_D_CTRLREG_WRVALUE_2	Drive 1 Steuerregister Sollwert 2
*.a_DRIVE_0_15[1].b_D_CTRLREG_WRVALUE_3	Drive 1 Steuerregister Sollwert 3
*.a_DRIVE_0_15[1].b_D_CTRLREG_RDVALUE_0	Drive 1 Steuerregister Istwert 0
*.a_DRIVE_0_15[1].b_D_CTRLREG_RDVALUE_1	Drive 1 Steuerregister Istwert 1
*.a_DRIVE_0_15[1].b_D_CTRLREG_RDVALUE_2	Drive 1 Steuerregister Istwert 2
*.a_DRIVE_0_15[1].b_D_CTRLREG_RDVALUE_3	Drive 1 Steuerregister Istwert 3
*.a_DRIVE_0_15[1].b_D_CTRLREG_SD	Drive 1 Steuerregister Bedarfsdaten
*.a_DRIVE_0_15[1].b_D_CC_REG	Drive 1 Kommunikationskontrolle Bedarfs-daten
*.a_DRIVE_0_15[1].w_D_ERROR_NR	Drive 1 Fehlernummer Bedarfsdaten
*.a_DRIVE_0_15[1].w_D_PARAMETER_NR	Drive 1 Parameternummer Bedarfsdaten
*.a_DRIVE_0_15[1].b_D_FORMAT	Drive 1 Format Bedarfsdaten
*.a_DRIVE_0_15[1].b_D_ELEMENT	Drive 1 Element Bedarfsdaten
•	•
•	•
•	•
*.a_DRIVE_0_15[15].w_D_CONTROLWORD	Drive 15 Steuerwort
*.a_DRIVE_0_15[15].w_D_STATUSWORT	Drive 15 Statuswort
*.a_DRIVE_0_15[15].w_D_DUMMY_0	Drive 15 reserviert
*.a_DRIVE_0_15[15].w_D_WRVALUE_0	Drive 15 Sollwert 0
*.a_DRIVE_0_15[15].d_D_WRVALUE_1	Drive 15 Sollwert 1
*.a_DRIVE_0_15[15].d_D_WRVALUE_2	Drive 15 Sollwert 2
*.a_DRIVE_0_15[15].d_D_WRVALUE_3	Drive 15 Sollwert 3
*.a_DRIVE_0_15[15].w_D-DUMMY_1	Drive 15 reserviert
*.a_DRIVE_0_15[15].w_D_RDVALUE_0	Drive 15 Istwert 0
*.a_DRIVE_0_15[15].d_D_RDVALUE_1	Drive 15 Istwert 1
*.a_DRIVE_0_15[15].d_D_RDVALUE_2	Drive 15 Istwert 2
*.a_DRIVE_0_15[15].d_D_RDVALUE_3	Drive 15 Istwert 3
*.a_DRIVE_0_15[15].w_D_SD_0	Drive 15 Bedarfsdaten 0
*.a_DRIVE_0_15[15].w_D_SD_1	Drive 15 Bedarfsdaten 1

Adresse / Register	Bedeutung
*.a_DRIVE_0_15[15].w_D_SD_2	Drive 15 Bedarfsdaten 2
*.a_DRIVE_0_15[15].w_D_SD_3	Drive 15 Bedarfsdaten 3
*.a_DRIVE_0_15[15].b_D_CTRLREG_STATUSWORD	Drive 15 Steuerregister Statuswort
*.a_DRIVE_0_15[15].b_D_CTRLREG_CONTROLWORD	Drive 15 Steuerregister Steuerwort
*.a_DRIVE_0_15[15].w_D_DUMMY_2	Drive 15 reserviert
*.a_DRIVE_0_15[15].b_D_CTRLREG_WRVALUE_0	Drive 15 Steuerregister Sollwert 0
*.a_DRIVE_0_15[15].b_D_CTRLREG_WRVALUE_1	Drive 15 Steuerregister Sollwert 1
*.a_DRIVE_0_15[15].b_D_CTRLREG_WRVALUE_2	Drive 15 Steuerregister Sollwert 2
*.a_DRIVE_0_15[15].b_D_CTRLREG_WRVALUE_3	Drive 15 Steuerregister Sollwert 3
*.a_DRIVE_0_15[15].b_D_CTRLREG_RDVALUE_0	Drive 15 Steuerregister Istwert 0
*.a_DRIVE_0_15[15].b_D_CTRLREG_RDVALUE_1	Drive 15 Steuerregister Istwert 1
*.a_DRIVE_0_15[15].b_D_CTRLREG_RDVALUE_2	Drive 15 Steuerregister Istwert 2
*.a_DRIVE_0_15[15].b_D_CTRLREG_RDVALUE_3	Drive 15 Steuerregister Istwert 3
*.a_DRIVE_0_15[15].b_D_CTRLREG_SD	Drive 15 Steuerregister Bedarfsdaten
*.a_DRIVE_0_15[15].b_D_CC_REG	Drive 15 Kommunikationskontrolle Bedarfsdaten
*.a_DRIVE_0_15[15].w_D_ERROR_NR	Drive 15 Fehlernummer Bedarfsdaten
*.a_DRIVE_0_15[15].w_D_PARAMETER_NR	Drive 15 Parameternummer Bedarfsdaten
*.a_DRIVE_0_15[15].b_D_FORMAT	Drive 15 Format Bedarfsdaten
*.a_DRIVE_0_15[15].b_D_ELEMENT	Drive 15 Element Bedarfsdaten

* entspricht zum Beispiel _CAN_CTRL_OPT.

4.4.2 Kommunikation

In der nachfolgenden Beschreibungen werden folgende Abkürzungen verwendet:

16#	hexadezimal
2#	binär
ID	Identifier
IDB1	Identifierbits 10-3
IDB2	Identifierbits 2-0
XXXX	angesprochene Achsennummer (0-15)
RTR	Remote Bit
DLC	Anzahl der Datenbytes
DB0-DB7	Datenbytes 0 bis 7

Die angegebenen Adressen gelten für den Regler mit CAN-Node-ID 0 (bei Reglern mit anderen CAN-Node-ID muß a_DRIVE_0_15[CAN-Node-ID] verwendet werden). (siehe oben)

Die Bedeutung der Parameter muß der jeweiligen Reglerbeschreibung entnommen werden.

Prozeßdaten

Für die Prozeßdatenkommunikation kann der FB CAN_PD_COMM verwendet werden. Bei den Prozeßdaten handelt es sich um Nachrichten mit hoher Priorität.

- - Steuerwort
- - Statuswort
- - Sollwert 0-3
- - Istwert 0-3

Steuerwort (→ FB CAN_PD_COMM)

Parameternummer im Regler: 120

Um ein Steuerwort auf dem Bus zur Verfügung zu stellen sind 2 Möglichkeiten vorgesehen:

- Schreiben von 16#03 ins *Steuerregister Steuerwort*
  (*.a_DRIVE_0_15[0].b_D_CTRLREG_CONTROLWORD)
  Der Inhalt von *.a_DRIVE_0_15[0].w_D_CONTROLWORD *(Steuerwort*)
  wird auf dem Bus zur Verfügung gestellt.
  Das wird durch 16#04 im *Steuerregister Steuerwort* quittiert.
- Änderung des Steuerwortes
  Wird das Steuerwort geändert, wird der neue Inhalt automatisch auf dem Bus zur Verfügung gestellt.
  Das erfolgreiche Senden wird mit 16#04 im entsprechenden Steuerregister quittiert.

Kommunikationsobjekt:

ID	RTR	DLC	DB0	DB1
2#0000XXXX001	0	2	Steuerwort	

Statuswort (→ FB CAN_PD_COMM)

Parameternummer im Regler: 121

Zur Anforderung des Statuswortes muß das *Steuerregister Statuswort*
(*.a_DRIVE_0_15[0].b_D_CTRLREG_STATUSWORD) mit 16#01 beschrieben werden.

Dies bewirkt folgende Anforderung:

Statuswort-Anforderung

ID	RTR	DLC
2#0001XXXX001	1	0

Als Antwort wird folgende Nachricht erwartet:

Dieses Objekt wird auch ohne Anforderung empfangen, d. h. das Statuswort muß nicht explizit angefordert werden, da es vom Antriebsknoten bei Änderung des Status selbstständig gesendet wird.

Kommunikationsobjekt:

ID	RTR	DLC	DB0	DB1
2#0001XXXX001	0	2	Statuswort	

Der Empfang wird mit 16#02 im *Steuerregister Statuswort* quittiert. Danach stehen die neuen Daten im Register *.a_DRIVE_0_15[0].w_D_STATUSWORT *(Statuswort)* zu Verfügung.

Sollwerte (→ FB CAN_PD_COMM)

Es stehen maximal vier Sollwerte zu Verfügung, die auf Wunsch zyklisch gesendet werden können.

Als Sollwert kann jeder beschreibbare Parameter, den der Regler zu Verfügung stellt, eingestellt werden. Die Einstellung der gewünschten Parameternummern erfolgt mit dem FB CAN_DRIVE_INIT. Weiterhin ist eine Einstellung der gewünschten Parameternummern mit Hilfe von “PA-Daten schreiben” (wird getrennt behandelt) möglich. Dabei ist darauf zu achten, daß die übertragene Sollwertlänge mit dem Format des Parameters im Regler übereinstimmt. Es werden folgende Datentypen unterstützt:

Sollwert 0:	- Wort	DLC=2;
Sollwert 1-3	- Wort	DLC=2;
	- Doppelwort	DLC=4;

Das jeweilige Format für Sollwert **x** wird im *Steuerregister Sollwert*
(*.a_DRIVE_0_15[0].b_D_CTRLREG_WRVALUE_**x**) festgelegt:

2#xxxx 0001:	Wort
2#xxxx 0010:	Doppelwort

Die zu sendenden Sollwerte sind in das entsprechende Register *Sollwert0-3*
(*.a_DRIVE_0_15[0].w_D_WRVALUE_0 bis *.a_DRIVE_0_15[0].w_D_WRVALUE_3) einzutragen.

Ist keine zyklische Übertragung gewählt, wird die Übertragung über das *Steuerregister Sollwert* gesteuert. Das Schreiben eines Sollwertes auf den Bus wird durch 2#0011xxxx im *Steuerregister Sollwert* gestartet. Das erfolgreiche Senden wird mit 2#0100xxxx bestätigt.

Kommunikationsobjekt:

IDB0	IDB1	RTR	DLC	DB0	DB1	DB2	DB3
2#0010XXXX	SW-Nr.	0	1-4	Sollwert-Daten			

SW_Nr: Sollwert-Nummer 0-3

Beispiel: Übertragen von Sollwert 1 an den Regler mit CAN-Node-ID 0 (keine Zykluszeit gesetzt)

- Einstellen der gewünschten Parameternummer FB CAN_DRIVE_INIT, FB CAN_INIT (Initialisierung
- Daten an vorgesehene Adresse schreiben:
  Daten in *.a_DRIVE_0_15[0].d_D_WRVALUE_1
- Absenden der Nachricht über *Steuerregister Sollwert 1*
  (*.a_DRIVE_0_15[0].b_D_CTRLREG_WRVALUE_1).

  - Wort: 16#31

  - Doppelwort: 16#32

  Ist der Sollwert gesendet, wird dies im *Steuerregister Sollwert 1* angezeigt

  - Wort 16#41

  - Doppelwort 16#42

Istwerte (→ FB CAN_PD_COMM)

Es stehen maximal vier Istwerte zu Verfügung, die auf Wunsch zyklisch angefordert werden können. Als Istwert kann jeder Parameter, den der Regler zu Verfügung stellt, eingestellt werden. Die Einstellung der gewünschten Parameternummern erfolgt mit dem FB CAN_DRIVE_INIT. Weiterhin ist eine Einstellung der gewünschten Parameternummern mit Hilfe von “PE-Daten schreiben” (wird getrennt behandelt) möglich. Es werden folgende Datentypen unterstützt:

Istwert 0:	- Wort	DLC=2;
Istwert 1-3	- Wort	DLC=2;
	- Doppelwort	DLC=4;

Zur Anforderung eines Istwertes vom Regler muß das *Steuerregister Istwert* **x**
(*.a_DRIVE_0_15[0].b_D_CTRLREG_RDVALUE_**x**) mit 16#01 beschrieben werden. Dies bewirkt folgende Anforderung:

IDB0	IDB1	RTR	DLC
2#0011XXXX	IW-Nr.	1	0

IW-Nr: Istwert-Nummer 0-3

Als Antwort wird folgende Nachricht erwartet.

Kommunikationsobjekt:

IDB0	IDB1	RTR	DLC	DB0	DB1	DB2	DB3
2#0011XXXX	IW-Nr.	0	1-4	Istwert-Daten			

Beispiel: Anfragen von Istwert 0 vom Regler mit CAN-Node-ID 0 (keine zyklische Anforderung)

- Einstellen der gewünschten Parameternummer FB CAN_DRIVE_INIT, FB CAN_INIT (Initialisierung)
- Eintragen von 16#01 ins *Steuerregister Istwert 0*
  (*.a_DRIVE_0_15[0].b_D_CTRLREG_RDVALUE_0)
- Der Empfang des gewünschten Istwertes wird durch 16#02 im *Steuerregister Istwert 0* angezeigt.
- Der empfangene Istwert steht in *.a_DRIVE_0_15[0].w_D_RDVALUE_0 zu Verfügung

Bedarfsdaten

Für die Bedarfsdaten-Kommunikation können die FBs CAN_PA_LIST_READ, CAN_PA_LIST_WRITE, CAN_PE_LIST_READ, CAN_PE_LIST_WRITE, CAN_PAR_READ, CAN_PAR_WRITE, CAN_OBJ_READ, CAN_OBJ_WRITE und CAN_SD_CONTROL verwendet werden. Die Bedarfsdatenübertragung dient zum direkten Zugriff auf alle Parameter des Reglers bzw. auf Objekte, die in der Objektliste eines Reglers enthalten sind. Im Gegensatz zur Prozeßdatenübertragung werden die Parameternummern (Objektnummern) bei jeder Nachricht mit übertragen. Alle Schreib- bzw. Lesezugriffe werden von der CAN-Anschaltung auf dem jeweiligen Regler bestätigt.

PA-Nachricht

Die PA-Nachrichten dienen zum Lesen bzw. Schreiben der Parameternummern der Sollwerte, die mittels Sollwertübertragung (Prozeßdaten) gesendet werden sollen. Zu diesem Zweck befindet sich auf dem Regler eine Liste in der eine Defaulteinstellung abgelegt ist. Wird diese überschrieben, kann die neue Einstellung mit Hilfe der Gesamtparameterabspeicherung (Parameter 190) übernommen werden.

Wortnummer	PA-Parameternummernliste
0	Parameternummer von Sollwert 0
1	Parameternummer von Sollwert 1
2	Parameternummer von Sollwert 2
3	Parameternummer von Sollwert 3

Bei der Festlegung der Sollwertnummern ist darauf zu achten, daß der Sollwert 0 eine maximale Länge von 2 Byte haben darf. Für nicht benötigte Sollwerte muß als Parameternummer 16#0000 eingetragen werden.

PA-Daten schreiben (→ FB CAN_PA_LIST_WRITE)

Mit der Funktion PA-Daten schreiben wird eine neue PA-Parameternummernliste im Regler eingestellt.

– Eintragen der 4 Parameternummern in *Bedarfsdaten 0-3*
(*.a_DRIVE_0_15[0].w_D_SD_0 bis *.a_DRIVE_0_15[0].w_D_SD_3)

– Eintragen 16#0B in das *Steuerregister Bedarfsdaten*
(*.a_DRIVE_0_15[0].b_D_CTRLREG_SD)

Kommunikationsobjekt:

ID	RTR	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
2#1000XXXX100	0	8	Sollwert-Nr.0		Sollwert-Nr.1		Sollwert-Nr.2		Sollwert-Nr.3	

Anmerkung: Die Freigabe der Sollwerte erfolgt mit Hilfe des Objekts 16#6002

Zur Bestätigung des erfolgreichen Schreibzugriffs wird folgende Response erwartet:

ID	RTR	DLC	DB0
2#1000XXXX101	0	1	16#00

Der Empfang der Response wird mit 16#0C *im Steuerregister Bedarfsdaten* angezeigt.

Kann der Schreibzugriff nicht durchgeführt werden wird folgende Fehlermeldung erwartet.

ID	RTR	DLC	DB0	DB1	DB2
2#1000XXXX101	0	3	16#80	16#00	16#41

Der Fehlerfall wird mit 16#2A im *Steuerregister Bedarfsdaten* angezeigt.

PA-Daten lesen (→ FB CAN_PA_LIST_READ)

Mit der Funktion PA-Daten lesen kann die aktuelle PA-Parameternummernliste im Regler ausgelesen werden.

– Eintragen von 16#09 in das *Steuerregister Bedarfsdaten*
(*.a_DRIVE_0_15[0].b_D_CTRLREG_SD)

Anforderung:

ID	RTR	DLC
2#1000XXXX100	1	0

Die angeforderten Daten werden in folgender Form erwartet:

Kommunikationsobjekt:

ID	RTR	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
2#1000XXXX101	0	8	Sollwert-Nr.0		Sollwert-Nr.1		Sollwert-Nr.2		Sollwert-Nr.3	

Der Empfang der Response wird mit 16#0A im *Steuerregister Bedarfsdaten* angezeigt.

Die empfangenen 4 Parameternummern stehen in den Registern *Bedarfsdaten 0-3*
(*.a_DRIVE_0_15[0].w_D_SD_0 bis *.a_DRIVE_0_15[0].w_D_SD_3)

Kann der Lesezugriff nicht durchgeführt werden, wird folgende Fehlermeldung erwartet.

ID	RTR	DLC	DB0	DB1	DB2
2#1000XXXX011	0	3	16#20	Fehler-Nr.	

Der Fehlerfall wird mit 16#2A im *Steuerregister Bedarfsdaten* angezeigt.

PE-Nachricht

Die PE-Nachrichten dienen zum Lesen bzw. Schreiben der Parameternummern der Istwerte, die mittels Istwertübertragung (Prozeßdaten) von der CAN-Anschaltung des Reglers gesendet werden sollen. Zu diesem Zweck befindet sich auf dem Regler eine Liste in der eine Defaulteinstellung abgelegt ist. Wird diese überschrieben, kann die neue Einstellung mit Hilfe der Gesamtparameterabspeicherung (Parameter 190) übernommen werden.

Wortnummer	PE-Parameternummernliste
0	Parameternummer von Istwert 0
1	Parameternummer von Istwert 1
2	Parameternummer von Istwert 2
3	Parameternummer von Istwert 3

Bei der Festlegung der Istwertnummern ist darauf zu achten, daß der Istwert 0 eine maximale Länge von 2 Byte haben darf. Für nicht benötigte Istwerte muß als Parameternummer 16#0000 eingetragen werden.

PE-Daten schreiben (→ FB CAN_PE_LIST_WRITE)

Mit der Funktion PE-Daten schreiben wird eine neue PE-Parameternummernliste im Regler eingestellt.

– Eintragen der 4 Parameternummern *in Bedarfsdaten 1-4*
(*.a_DRIVE_0_15[0].w_D_SD_0 bis *.a_DRIVE_0_15[0].w_D_SD_3)

– Eintragen 16#0F in das *Steuerregister Bedarfsdaten*
(*.a_DRIVE_0_15[0].b_D_CTRLREG_SD)

Kommunikationsobjekt:

ID	RTR	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
2#1000XXXX110	0	8	Istwert-Nr.0		Istwert-Nr.1		Istwert-Nr.2		Istwert-Nr.3	

Zur Bestätigung des erfolgreichen Schreibzugriffs wird folgende Response erwartet:

ID	RTR	DLC	DB0
2#1000XXXX111	0	1	16#00

Der Empfang der Response wird mit 16#10 im *Steuerregister Bedarfsdaten* angezeigt.

Kann der Lesezugriff nicht durchgeführt werden wird folgende Fehlermeldung erwartet.

ID	RTR	DLC	DB0	DB1	DB2
2#1000XXXX111	0	3	16#80	16#00	16#41

Der Fehlerfall wird mit 16#2E im *Steuerregister Bedarfsdaten* angezeigt.

PE-Daten lesen (→ FB CAN_PE_LIST_READ)

Mit der Funktion PE-Daten lesen kann die aktuelle PE-Parameternummernliste im Regler ausgelesen werden.

– Eintragen von 16#0D in das *Steuerregister Bedarfsdaten*
(*.a_DRIVE_0_15[0].b_D_CTRLREG_SD)

Anforderung:

ID	RTR	DLC
2#1000XXXX110	1	0

Die angeforderten Daten werden in folgender Form erwartet:

Kommunikationsobjekt:

ID	RTR	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
2#1000XXXX111	0	8	Istwert-Nr.0		Istwert-Nr.1		Istwert-Nr.2		Istwert-Nr.3	

Der Empfang der Response wird mit 16#0E im *Steuerregister Bedarfsdaten* angezeigt.

Die empfangenen 4 Parameternummern stehen in den Registern *Bedarfsdaten 0-3*
(*.a_DRIVE_0_15[0].w_D_SD_0 bis *.a_DRIVE_0_15[0].w_D_SD_3)

Kann der Lesezugriff nicht durchgeführt werden wird folgende Fehlermeldung erwartet.

ID	RTR	DLC	DB0	DB1	DB2
2#1000XXXX011	0	3	16#20	Fehler-Nr	

Der Fehlerfall wird mit 16#2E im *Steuerregister Bedarfsdaten* angezeigt.

Parameter schreiben (→ FB CAN_PAR_WRITE)

Mit der Funktion Parameter schreiben kann auf alle Parameter des Reglers zugegriffen werden. Es können Daten im Format Wort und Doppelwort übertragen werden. Dazu ist folgende Vorgehensweise notwendig:

- Eintragen der Regler-Parameternummer (*.a_DRIVE_0_15[0].w_D_PARAMETER_NR)
- Eintragen des zu übertragenden Datenformats (*.a_DRIVE_0_15[0].b_D_FORMAT)

  16#01: Wort

  16#02: Doppelwort
- Eintragen des gewünschten Elements des Parameters (*.a_DRIVE_0_15[0].b_D_ELEMENT)

  z. B. 16#07:Parameter-Daten
- Eintragen der zu übertragenden Daten

  Wort: *.a_DRIVE_0_15[0].w_D_SD_0

  Doppelwort (Lowword): *.a_DRIVE_0_15[0].w_D_SD_0

  Doppelwort (Highword): *.a_DRIVE_0_15[0].w_D_SD_1
- Eintragen von 16#07 in das *Steuerregister Bedarfsdaten*
  (*.a_DRIVE_0_15[0].b_D_CTRLREG_SD)

Kommunikationsobjekt:

ID	RTR	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
2#1000XXXX100	0	5-8	Format		PA-Nr.		Daten 1-4Bytes			

Zur Bestätigung des erfolgreichen Schreibzugriffs wird folgende Response erwartet:

ID	RTR	DLC	DB0
2#1000XXXX011	0	1	16#08

Der Empfang der Response wird mit 16#08 im *Steuerregister Bedarfsdaten* angezeigt.

Kann der Parameter nicht geschrieben werden wird folgende Fehlermeldung erwartet:

ID	RTR	DLC	DB0	DB1	DB2
2#1000XXXX011	0	3	16#28	Fehler-Nr.	

Der Fehlerfall wird mit 16#28 im *Steuerregister Bedarfsdaten* angezeigt. Die zugehörige Fehlernummer steht in *.a_DRIVE_0_15[0].w_D_ERROR_NR zu Verfügung.

Wert	Bedeutung
16#0000	Kein Fehler aufgetreten
16#FFFF	Fehler aufgetreten
16#FFFE	Wert kleiner Minimalwert
16#FFFD	Wert größer Maximalwert
16#FFFC	Element nicht veränderbar
16#FFFB	Element nicht vorhanden
16#FFFA	Daten nicht verfügbar (z. B. in Bearbeitung)
16#FFF9	Fehler beim Datenformat

Parameter lesen (→ FB CAN_PAR_READ)

Mit der Funktion Parameter lesen kann auf alle Parameter des Reglers zugegriffen werden. Es können Daten im Format Wort und Doppelwort gelesen werden. Dazu ist folgende Vorgehensweise notwendig:

- Eintragen der Regler-Parameternummer (*.a_DRIVE_0_15[0].w_D_PARAMETER_NR)
- Eintragen des zu übertragenden Datenformats (*.a_DRIVE_0_15[0].b_D_FORMAT)

  16#01: Wort

  16#02: Doppelwort
- Eintragen des gewünschten Elements des Parameters (*.a_DRIVE_0_15[0].b_D_ELEMENT)

  z. B. 16#07:Parameter-Daten
- Eintragen von 16#05 in das *Steuerregister Bedarfsdaten*
  (*.a_DRIVE_0_15[0].b_D_CTRLREG_SD)

Anforderung:

ID	RTR	DLC	DB0	DB1	DB2	DB3
2#1000XXXX010	0	4	For-mat	Ele-ment	PA-Nr.	

Der angeforderte Parameter wird in folgender Form erwartet:

Kommunikationsobjekt:

ID	RTR	DLC	DB0	DB1	DB2	DB3	DB4
2#1000XXXX011	0	2-5	For-mat	Daten 1-4 Byte			

Der Empfang der Response wird mit 16#06 im *Steuerregister Bedarfsdaten* angezeigt.

Die empfangenen Daten stehen in den Registern Bedarfsdaten 0 (und 1 wenn Doppelwort)

Wort: *.a_DRIVE_0_15[0].w_D_SD_0

Doppelwort (Lowword): *.a_DRIVE_0_15[0].w_D_SD_0

Doppelwort (Highword): *.a_DRIVE_0_15[0].w_D_SD_1

Kann der Parameter nicht gelesen werden wird folgende Fehlermeldung erwartet.

ID	RTR	DLC	DB0	DB1	DB2
2#1000XXXX011	0	3	16#20	Fehler-Nr.	

Der Fehlerfall wird mit 16#26 im *Steuerregister Bedarfsdaten* angezeigt. Die zugehörige Fehlernummer steht in *.a_DRIVE_0_15[0].w_D_ERROR_NR zu Verfügung.

Wert	Bedeutung
16#0000	Kein Fehler aufgetreten
16#FFFF	Fehler aufgetreten
16#FFFE	Wert kleiner Minimalwert
16#FFFD	Wert größer Maximalwert
16#FFFC	Element nicht veränderbar
16#FFFB	Element nicht vorhanden
16#FFFA	Daten nicht verfügbar (z. B. in Bearbeitung)
16#FFF9	Fehler beim Datenformat

Objekt schreiben (→ FB CAN_OBJ_WRITE)

Mit der Funktion Objekt schreiben kann auf alle Objekte der Objektliste zugegriffen werden. Es können Daten im Format Byte, Wort und Doppelwort übertragen werden. Dazu ist folgende Vorgehensweise notwendig:

- Eintragen der Objektnummer (*.a_DRIVE_0_15[0].w_D_PARAMETER_NR)
- Eintragen des zu übertragenden Datenformats (*.a_DRIVE_0_15[0].b_D_FORMAT)

  16#00: Byte

  16#01: Wort

  16#02: Doppelwort

– Eintragen des gewünschten Subindex des Objekts (*.a_DRIVE_0_15[0].b_D_ELEMENT)
z. B. 16#00: Beschreiben des gesamten Objekts

– Eintragen der zu übertragenden Daten

Wort: *.a_DRIVE_0_15[0].w_D_SD_0

Doppelwort (Lowword): *.a_DRIVE_0_15[0].w_D_SD_0

Doppelwort (Highword): *.a_DRIVE_0_15[0].w_D_SD_1

– Eintragen von 16#03 in das *Steuerregister Bedarfsdaten*
(*.a_DRIVE_0_15[0].b_D_CTRLREG_SD)

Kommunikationsobjekt:

ID	RTR	DLC	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7
2#1000XXXX000	0	5-8	For-mat	Subin-dex	Objekt-Nr.		Daten 1-4 Bytes			

Zur Bestätigung des erfolgreichen Schreibzugriffs wird folgende Response erwartet:

ID	RTR	DLC	DB0
2#1000XXXX001	0	1	16#00

Der Empfang der Response wird mit 16#04 im *Steuerregister Bedarfsdaten* angezeigt.

Kann das Objekt nicht geschrieben werden wird folgende Fehlermeldung erwartet.

ID	RTR	DLC	DB0
2#1000XXXX001	0	1	16#50

Der Fehlerfall wird mit 16#22 im *Steuerregister Bedarfsdaten* angezeigt.

Objekt lesen (→ FB CAN_OBJ_READ)

Mit der Funktion Objekt lesen kann auf alle Objekte der Objektliste zugegriffen werden. Es können Daten im Format Byte, Wort und Doppelwort gelesen werden. Dazu ist folgende Vorgehensweise notwendig:

– Eintragen der Objektnummer (*.a_DRIVE_0_15[0].w_D_PARAMETER_NR)

– Eintragen des zu übertragenden Datenformats (*.a_DRIVE_0_15[0].b_D_FORMAT)

16#01: Wort

16#02: Doppelwort

– Eintragen des gewünschten Subindex des Objekts (*.a_DRIVE_0_15[0].b_D_ELEMENT)
z. B. 16#00: lesen des gesamten Objekts

– Eintragen von 16#01 in das *Steuerregister Bedarfsdaten*
(*.a_DRIVE_0_15[0].b_D_CTRLREG_SD)

Anforderung:

ID	RTR	DLC	DB0	DB1	DB2	DB3
2#1000XXXX000	0	4	For-mat	Ele-ment	Objekt-Nr.	

Der angeforderte Parameter wird in folgender Form erwartet:

Kommunikationsobjekt:

ID	RTR	DLC	DB0	DB1	DB2	DB3	DB4
2#1000XXXX001	0	2-5	For-mat	Daten 1-4 Byte			

Der Empfang der Response wird mit 16#02 im *Steuerregister Bedarfsdaten* angezeigt.

Die empfangenen Daten stehen in den Registern Bedarfsdaten 0 (und 1 wenn Doppelwort)

Wort:		*.a_DRIVE_0_15[0].w_D_SD_0
Doppelwort	(Lowword):	*.a_DRIVE_0_15[0].w_D_SD_0
Doppelwort	(Highword):	*.a_DRIVE_0_15[0].w_D_SD_1

Kann das Objekt nicht gelesen werden wird folgende Fehlermeldung erwartet.

ID	RTR	DLC	DB0
2#1000XXXX001	0	1	16#50

Der Fehlerfall wird mit 16#22 im *Steuerregister Bedarfsdaten* angezeigt.

4.5 Betrieb von maximal 16 Encoder

4.5.1 Adressierung

Anmerkung: Vorgesehen sind Absolutwertgeber vom Typ HE-65-M-CAN

Auflösung pro Umdrehung:	4096 Schritte
Meßbereich:	4096 Umdrehungen
Encoderkapazität:	24 Bit
Betriebsspannung:	11-27V_{DC}
Ausgabecode:	Binär
Baudrate:	250/500 kbit/s
Schnittstelle:	galvanisch getrennt

Adresse / Register	**Bedeutung**
*.w_BRC_ENCODER	Steuerregister Broadcast
*.CTRL_ENCODER_0_15[0]	Steuerregister Encoder 0
*.CTRL_ENCODER_0_15[1]	Steuerregister Encoder 1
*.CTRL_ENCODER_0_15[2]	Steuerregister Encoder 2
•	•
•	•
•	•
*.CTRL_ENCODER_0_15[14]	Steuerregister Encoder 14
*.CTRL_ENCODER_0_15[15]	Steuerregister Encoder 15
*.a_ABS_POS_0_15[0]	absolute Position Encoder 0
*.a_ABS_POS_0_15[1]	absolute Position Encoder 1
*.a_ABS_POS_0_15[2]	absolute Position Encoder 2
•	•
•	•
•	•
*.a_ABS_POS_0_15[14]	absolute Position Encoder 14
*.a_ABS_POS_0_15[15]	absolute Position Encoder 15

* entspricht zum Beispiel _CAN_CTRL_OPT.

4.5.2 Kommunikation

Die absoluten Encoder-Positionen der angeschlossenen Absolutwertgeber wird durch Schreiben von 16#01 in das *Steuerregister Broadcast* (*.w_BRC_ENCODER) abgefragt (Format WORD). Dies bewirkt ein Datentelegramm mit Identifier “0” und Data Length Code “0”. Nach Absenden dieses Telegramms wird das *Steuerregister Broadcast* wieder auf 16#00 gesetzt.

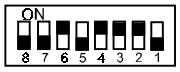
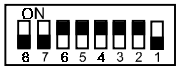
Der Empfang der einzelnen Encoder-Positionen wird durch 16#02 in den *Steuerregister Encoder 0-15 (* *.CTRL_ENCODER_0_15[0] bis *.CTRL_ENCODER_0_15[15] *)* bestätigt (Format BYTE).

Die Position steht in folgendem Format zu Verfügung (Beispiel Encoder1):
(Format DWORD)

	*.a_ABS_POS_0_15[1]			
Byte	High-Word High-Byte	High-Word Low-Byte	Low-Word High-Byte	Low-Word Low-Byte
Encoder-Pos.	2^{23}... 2^{16}	2^{15}... 2^{8}	2^{7}... 2^{0}	0

Die Position wird in den jeweiligen Registern *absolute Position Encoder 0-15* (*.a_ABS_POS_0_15[0] bis *.a_ABS_POS_0_15[15]) ausgegeben

Einstellung der Encodernummer über DIP-Schalter am Encoder:

Encodernummern	DIP-Schalter ID-Bit
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	

4.6 Senden eines beliebigen Telegramms

Diese Funktion ermöglicht das Senden beliebiger Telegramme mit maximal 8 Byte Nutzdaten. Zu diesem Zweck ist folgende Struktur vorgegeben:

Adresse / Register	**Bedeutung**							
*.b_STEUREG_TELE	Steuerregister Telegramm							
*.b_TELE_DUMMY	reserviert							
*.b_TELE_ADR_LOW	Id2	Id1	Id0	RTR	DLC3	DLC2	DLC1	DLC0
*.b_TELE_ADR_HIGH	Id10	Id9	Id8	Id7	Id6	Id5	Id4	Id3
*.b_TELE_DB_0	Datenbyte 0							
*.b_TELE_DB_1	Datenbyte 1							
*.b_TELE_DB_2	Datenbyte 2							
*.b_TELE_DB_3	Datenbyte 3							
*.b_TELE_DB_4	Datenbyte 4							
*.b_TELE_DB_5	Datenbyte 5							
*.b_TELE_DB_6	Datenbyte 6							
*.b_TELE_DB_7	Datenbyte 7							

* entspricht zum Beispiel _CAN_CTRL_OPT.

Nach Eintragen des gewünschten Datenrahmen wird dieser durch Schreiben von 16#01 in das *Steuerreg. Telegramm* auf dem Bus zu Verfügung gestellt. Das erfolgreiche Senden wird durch 16#02 im Steuerregister bestätigt.

(Bei Senden von Einzeltelegrammen muß am FB CAN_INIT am Eingang us_MAX_DRIVE_NR_COMM ein Wert >0 und <17 angegeben sein.)

4.7 Betrieb von maximal 16 Ωmegas mit CAN-Anschaltung

4.7.1 Adressierung

Für die Kommunikation mit maximal 16 Ωmegas ist für jedes Ωmega ein Registerbereich von 32 Bytes im Kommunikations-RAM der CAN-Anschaltung reserviert. Dieser Registerbereich hat folgende Bedeutung:

Adresse / Register	Bedeutung
*.a_OMEGA_0_15[0].b_O_STEUREG_SEND	Ωmega 0 Steuerregister Senden
*.a_OMEGA_0_15[0].b_O_STEUREG_EMPF	Ωmega 0 Steuerregister Empfangen
*.a_OMEGA_0_15[0].b_O_STATREG_SEND	Ωmega 0 Statusregister Senden
*.a_OMEGA_0_15[0].b_O_STATREG_SEND	Ωmega 0 Statusregister Empfangen
*.a_OMEGA_0_15[0].b_O_LENG_SEND	Ωmega 0 Längenregister der Sendenachricht
*.a_OMEGA_0_15[0].b_O_DUMMY0	Ωmega 0 reserviert
*.a_OMEGA_0_15[0].b_O_LENG_EMPF	Ωmega 0 Längenregister der Empfangsnachricht
*.a_OMEGA_0_15[0].b_O_ABS_BRC	Ωmega 0 Absenderregister bei Broadcastmeldung
*.a_OMEGA_0_15[0].b_O_SEND_DB_0	Ωmega 0 Sendenachricht Datenbyte 0
*.a_OMEGA_0_15[0].b_O_SEND_DB_1	Ωmega 0 Sendenachricht Datenbyte 1
*.a_OMEGA_0_15[0].b_O_SEND_DB_2	Ωmega 0 Sendenachricht Datenbyte 2
*.a_OMEGA_0_15[0].b_O_SEND_DB_3	Ωmega 0 Sendenachricht Datenbyte 3
*.a_OMEGA_0_15[0].b_O_SEND_DB_4	Ωmega 0 Sendenachricht Datenbyte 4
*.a_OMEGA_0_15[0].b_O_SEND_DB_5	Ωmega 0 Sendenachricht Datenbyte 5
*.a_OMEGA_0_15[0].b_O_SEND_DB_6	Ωmega 0 Sendenachricht Datenbyte 6
*.a_OMEGA_0_15[0].b_O_SEND_DB_7	Ωmega 0 Sendenachricht Datenbyte 7
*.a_OMEGA_0_15[0].b_O_EMPF_DB_0	Ωmega 0 Empfangsnachricht Datenbyte 0
*.a_OMEGA_0_15[0].b_O_EMPF_DB_1	Ωmega 0 Empfangsnachricht Datenbyte 1
*.a_OMEGA_0_15[0].b_O_EMPF_DB_2	Ωmega 0 Empfangsnachricht Datenbyte 2
*.a_OMEGA_0_15[0].b_O_EMPF_DB_3	Ωmega 0 Empfangsnachricht Datenbyte 3
*.a_OMEGA_0_15[0].b_O_EMPF_DB_4	Ωmega 0 Empfangsnachricht Datenbyte 4
*.a_OMEGA_0_15[0].b_O_EMPF_DB_5	Ωmega 0 Empfangsnachricht Datenbyte 5
*.a_OMEGA_0_15[0].b_O_EMPF_DB_6	Ωmega 0 Empfangsnachricht Datenbyte 6
*.a_OMEGA_0_15[0].b_O_EMPF_DB_7	Ωmega 0 Empfangsnachricht Datenbyte 7
*.a_OMEGA_0_15[0].d_O_DUMMY1	Ωmega 0 reserviert (Doppelwort)
*.a_OMEGA_0_15[0].d_O_DUMMY2	Ωmega 0 reserviert (Doppelwort)
*.a_OMEGA_0_15[1].b_O_STEUREG_SEND	Ωmega 1 Steuerregister Senden
*.a_OMEGA_0_15[1].b_O_STEUREG_EMPF	Ωmega 1 Steuerregister Empfangen
*.a_OMEGA_0_15[1].b_O_STATREG_SEND	Ωmega 1 Statusregister Senden
*.a_OMEGA_0_15[1].b_O_STATREG_SEND	Ωmega 1 Statusregister Empfangen
*.a_OMEGA_0_15[1].b_O_LENG_SEND	Ωmega 1 Längenregister der Sendenachricht
*.a_OMEGA_0_15[1].b_O_DUMMY0	Ωmega 1 reserviert
*.a_OMEGA_0_15[1].b_O_LENG_EMPF	Ωmega 1 Längenregister der Empfangsnachricht
*.a_OMEGA_0_15[1].b_O_ABS_BRC	Ωmega 1 Absenderregister bei Broadcastmeldung
*.a_OMEGA_0_15[1].b_O_SEND_DB_0	Ωmega 1 Sendenachricht Datenbyte 0
*.a_OMEGA_0_15[1].b_O_SEND_DB_1	Ωmega 1 Sendenachricht Datenbyte 1
*.a_OMEGA_0_15[1].b_O_SEND_DB_2	Ωmega 1 Sendenachricht Datenbyte 2
*.a_OMEGA_0_15[1].b_O_SEND_DB_3	Ωmega 1 Sendenachricht Datenbyte 3
*.a_OMEGA_0_15[1].b_O_SEND_DB_4	Ωmega 1 Sendenachricht Datenbyte 4
*.a_OMEGA_0_15[1].b_O_SEND_DB_5	Ωmega 1 Sendenachricht Datenbyte 5
*.a_OMEGA_0_15[1].b_O_SEND_DB_6	Ωmega 1 Sendenachricht Datenbyte 6
*.a_OMEGA_0_15[1].b_O_SEND_DB_7	Ωmega 1 Sendenachricht Datenbyte 7

Adresse / Register	Bedeutung
*.a_OMEGA_0_15[1].b_O_EMPF_DB_0	Ωmega 1 Empfangsnachricht Datenbyte 0
*.a_OMEGA_0_15[1].b_O_EMPF_DB_1	Ωmega 1 Empfangsnachricht Datenbyte 1
*.a_OMEGA_0_15[1].b_O_EMPF_DB_2	Ωmega 1 Empfangsnachricht Datenbyte 2
*.a_OMEGA_0_15[1].b_O_EMPF_DB_3	Ωmega 1 Empfangsnachricht Datenbyte 3
*.a_OMEGA_0_15[1].b_O_EMPF_DB_4	Ωmega 1 Empfangsnachricht Datenbyte 4
*.a_OMEGA_0_15[1].b_O_EMPF_DB_5	Ωmega 1 Empfangsnachricht Datenbyte 5
*.a_OMEGA_0_15[1].b_O_EMPF_DB_6	Ωmega 1 Empfangsnachricht Datenbyte 6
*.a_OMEGA_0_15[1].b_O_EMPF_DB_7	Ωmega 1 Empfangsnachricht Datenbyte 7
*.a_OMEGA_0_15[1].d_O_DUMMY1	Ωmega 1 reserviert (Doppelwort)
*.a_OMEGA_0_15[1].d_O_DUMMY2	Ωmega 1 reserviert (Doppelwort)
•	•
•	•
•	•
*.a_OMEGA_0_15[15].b_O_STEUREG_SEND	Ωmega 15 Steuerregister Senden
*.a_OMEGA_0_15[15].b_O_STEUREG_EMPF	Ωmega 15 Steuerregister Empfangen
*.a_OMEGA_0_15[15].b_O_STATREG_SEND	Ωmega 15 Statusregister Senden
*.a_OMEGA_0_15[15].b_O_STATREG_SEND	Ωmega 15 Statusregister Empfangen
*.a_OMEGA_0_15[15].b_O_LENG_SEND	Ωmega 15 Längenregister der Sendenachricht
*.a_OMEGA_0_15[15].b_O_DUMMY0	Ωmega 15 reserviert
*.a_OMEGA_0_15[15].b_O_LENG_EMPF	Ωmega 15 Längenregister der Empfangsnachricht
*.a_OMEGA_0_15[15].b_O_ABS_BRC	Ωmega 15 Absenderregister bei Broadcastmeldung
*.a_OMEGA_0_15[15].b_O_SEND_DB_0	Ωmega 15 Sendenachricht Datenbyte 0
*.a_OMEGA_0_15[15].b_O_SEND_DB_1	Ωmega 15 Sendenachricht Datenbyte 1
*.a_OMEGA_0_15[15].b_O_SEND_DB_2	Ωmega 15 Sendenachricht Datenbyte 2
*.a_OMEGA_0_15[15].b_O_SEND_DB_3	Ωmega 15 Sendenachricht Datenbyte 3
*.a_OMEGA_0_15[15].b_O_SEND_DB_4	Ωmega 15 Sendenachricht Datenbyte 4
*.a_OMEGA_0_15[15].b_O_SEND_DB_5	Ωmega 15 Sendenachricht Datenbyte 5
*.a_OMEGA_0_15[15].b_O_SEND_DB_6	Ωmega 15 Sendenachricht Datenbyte 6
*.a_OMEGA_0_15[15].b_O_SEND_DB_7	Ωmega 15 Sendenachricht Datenbyte 7
*.a_OMEGA_0_15[15].b_O_EMPF_DB_0	Ωmega 15 Empfangsnachricht Datenbyte 0
*.a_OMEGA_0_15[15].b_O_EMPF_DB_1	Ωmega 15 Empfangsnachricht Datenbyte 1
*.a_OMEGA_0_15[15].b_O_EMPF_DB_2	Ωmega 15 Empfangsnachricht Datenbyte 2
*.a_OMEGA_0_15[15].b_O_EMPF_DB_3	Ωmega 15 Empfangsnachricht Datenbyte 3
*.a_OMEGA_0_15[15].b_O_EMPF_DB_4	Ωmega 15 Empfangsnachricht Datenbyte 4
*.a_OMEGA_0_15[15].b_O_EMPF_DB_5	Ωmega 15 Empfangsnachricht Datenbyte 5
*.a_OMEGA_0_15[15].b_O_EMPF_DB_6	Ωmega 15 Empfangsnachricht Datenbyte 6
*.a_OMEGA_0_15[15].b_O_EMPF_DB_7	Ωmega 15 Empfangsnachricht Datenbyte 7
*.a_OMEGA_0_15[15].d_O_DUMMY1	Ωmega 15 reserviert (Doppelwort)
*.a_OMEGA_0_15[15].d_O_DUMMY2	Ωmega 15 reserviert (Doppelwort)

* entspricht zum Beispiel _CAN_CTRL_OPT.

4.7.2 Kommunikation

Senden von Einzelmeldungen von Ωmega an Ωmega

Es können zwischen Ωmegas Nachrichten von 0 bis 8 Bytes Länge versendet werden. Dazu trägt man die zusendenden Daten in den Sendedatenbereich des Ωmegas an den die Nachricht gesendet werden soll ein. Die Anzahl der Datenbytes der Nachricht trägt man in das Sendelängenregister ein. Die Nachricht wird durch das Eintragen von 16#05 in das Sendesteuerregister abgeschickt. Sobald die Nachricht auf dem CAN gesendet wurde trägt der CAN-Prozessor als Quittung ins Sendesteuerregister und ins Sendestatusregister 16#04 ein. Damit der parallele Zugriff von Ωmega und CAN-Prozessor auf den Datenbereich konfliktfrei abläuft, muß vom Ωmega zum Senden ins Sendestatusregister (zuerst!) und ins Sendesteuerregister 16#05 eingetragen werden. Entsprechend muß bei der Abfrage, ob der Sendevorgang durchgeführt wurde und die nächste Nachricht gesendet werden kann, das Sendesteuerregister (zuerst abfragen!) und das Sendestatusregister den Wert 16#04 enthalten.

Beispiel: Vom Ωmega 3 soll an das Ωmega 5 der Wert 16#7E8C gesendet werden.

Dazu wird im Ωmega 3 im Sendedatenbereich vom Ωmega 5 auf

_CAN_CTRL_OPT.a_OMEGA_0_15[5].b_O_SEND_DB_0

der Wert 16#8C und auf

_CAN_CTRL_OPT.a_OMEGA_0_15[5].b_O_SEND_DB_1

der Wert 16#7E eingetragen.

Ins Sendelängenregister

_CAN_CTRL_OPT.a_OMEGA_0_15[5].b_O_LENG_SEND

wird die Länge 16#02 eingetragen.

Durch Eintragen von 16#05 ins Sendestatusregister

_CAN_CTRL_OPT.a_OMEGA_0_15[5].b_O_STATREG_SEND

und ins Sendesteuerregister

_CAN_CTRL_OPT.a_OMEGA_0_15[5].b_O_STEUREG_SEND

wird die Nachricht (s.u.) auf dem CAN gesendet.

Das erfolgte Absenden wird im Sendesteuerregister

_CAN_CTRL_OPT.a_OMEGA_0_15[5].b_O_STEUREG_SEND

und Sendestatusregister

_CAN_CTRL_OPT.a_OMEGA_0_15[5].b_O_STATREG_SEND

mit 16#04 angezeigt.

ID 1	ID 2	RTR	DLC	DB 0	DB 1
2#01000101	011	0	2	16#8C	16#7E

Allgemein sieht die CAN-Nachricht bei Absender Ωmega 0 - 7 so aus:

ID 1	ID 2	RTR	DLC	DB 0 - DB 7
2#0100Empfänger-Nr	Absender-Nr	0	Länge in Bytes	Datenbytes 0 bis 7

Allgemein sieht die CAN-Nachricht bei Absender Ωmega 8 - 15 so aus:

ID 1	ID 2	RTR	DLC	DB 0 - DB 7
2#0110Empfänger-Nr	Absender-Nr - 8	0	Länge in Bytes	Datenbytes 0 bis 7

Senden von Broadcastmeldungen von Ωmega an Ωmega

Es können von jedem Ωmega Nachrichten von 0 bis 8 Bytes Länge versendet werden, die von allen anderen Ωmegas am CAN-Bus empfangen werden. Dazu trägt man die zusendenden Daten in den Sendedatenbereich des Ωmegas ein, der die Nachricht sendet. Die Anzahl der Datenbytes der Nachricht trägt man in das Sendelängenregister ein. Die Nachricht wird durch das Eintragen von 16#09 in das Sendesteuerregister abgeschickt. Sobald die Nachricht auf dem CAN gesendet wurde, trägt der CAN-Prozessor als Quittung ins Sendesteuerregister und ins Sendestatusregister 16#08 ein. Damit der parallele Zugriff von Ωmega und CAN-Prozessor auf den Datenbereich konfliktfrei abläuft, muß vom Ωmega zum Senden ins Sendestatusregister (zuerst !) und ins Sendesteuerregister 16#09 eingetragen werden. Entsprechend muß bei der Abfrage, ob der Sendevorgang durchgeführt wurde und die nächste Nachricht gesendet werden kann, das Sendesteuerregister (zuerst abfragen!) und das Sendestatusregister den Wert 16#08 enthalten.

Beispiel: Vom Ωmega 3 soll als Broadcastmeldung der Wert 16#A4 gesendet werden.

Dazu wird im Ωmega 3 im Sendedatenbereich vom Ωmega 3 auf

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_SEND_DB_0

der Wert 16#A4 eingetragen.

Ins Sendelängenregister

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_LENG_SEND

wird die Länge 16#01 eingetragen.

Durch Eintragen von 16#09 ins Sendestatusregister

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_STATREG_SEND

und ins Sendesteuerregister

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_STEUREG_SEND

wird die Nachricht (s. u.) auf dem CAN gesendet.

Das erfolgte Absenden wird im Sendesteuerregister

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_STEUREG_SEND

und Sendestatusregister

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_STATREG_SEND

mit 16#08 angezeigt.

ID 1	ID 2	RTR	DLC	DB 0
2#00010011	100	0	1	16#A4

Allgemein sieht die Broadcast-Nachricht so aus:

ID 1	ID 2	RTR	DLC	DB 0 - DB 7
2#0001Absender-Nr	100	0	Länge in Bytes	Datenbytes 0 bis 7

Empfangen von Einzelmeldungen von Ωmega an Ωmega

Um eine Einzelmeldung von einem anderen Ωmega zu empfangen, muß man den Empfang freigeben indem man im Datenbereich des Ωmega, von dem man die Nachricht empfangen will ins Empfangssteuerregister ein 16#03 einträgt. Sobald eine Nachricht von diesem Ωmega auf dem CAN empfangen wurde, trägt der CAN-Prozessor die empfangenen Datenbytes im Empfangsdatenbereich, die Länge im Empfangslängenregister und als Quittung ins Empfangssteuerregister und ins Empfangsstatusregister 16#02 ein. Damit der parallele Zugriff von Ωmega und CAN-Prozessor auf den Datenbereich konfliktfrei abläuft, muß vom Ωmega zum Starten des Empfangsvorgangs ins Empfangsstatusregister (zuerst!) und ins Empfangssteuerregister 16#03 eingetragen werden. Entsprechend muß bei der Abfrage, ob eine Nachricht empfangen wurde, das Empfangssteuerregister (zuerst abfragen!) und das Empfangsstatusregister den Wert 16#02 enthalten.

Falls am CAN eine Nachricht empfangen wurde, und das Empfangssteuerregister nicht auf 16#03 steht, dann wird diese Nachricht ignoriert und nicht in den Empfangsdatenspeicher geschrieben.

Beispiel: Ωmega 9 soll eine Meldung von Ωmega 0 empfangen. Diese Meldung enthält das Datenbyte 16#75.

Dazu wird im Ωmega 9 im Datenbereich vom Ωmega 0 ins Empfangsstatusregister

_CAN_CTRL_OPT.a_OMEGA_0_15[0].b_O_STATREG_EMPF

und ins Empfangssteuerregister

_CAN_CTRL_OPT.a_OMEGA_0_15[0].b_O_STEUREG_EMPF

der Wert 16#03 eingetragen.

Nachdem die Nachricht vom CAN empfangen wurde, steht im Empfangslängenregister

_CAN_CTRL_OPT.a_OMEGA_0_15[0].b_O_LENG_EMPF

die Länge 16#01 und im Empfangsdatenbereich im

_CAN_CTRL_OPT.a_OMEGA_0_15[0].b_O_EMPF_DB_0

das Datenbyte 16#75.

Der erfolgte Empfang wird durch 16#02 im Empfangssteuerregister

_CAN_CTRL_OPT.a_OMEGA_0_15[0].b_O_STEUREG_EMPF

und Empfangsstatusregister

_CAN_CTRL_OPT.a_OMEGA_0_15[0].b_O_STATREG_EMPF

angezeigt.

Empfangen von Broadcastmeldungen von Ωmega an Ωmega

Um eine Broadcastmeldung von einem anderen Ωmega zu empfangen, muß man den Empfang freigeben indem man im Datenbereich des Ωmega, das die Nachricht empfangen will, ins Empfangssteuerregister ein 16#07 einträgt. Sobald eine Broadcast-Nachricht auf dem CAN empfangen wurde trägt der CAN-Prozessor die empfangenen Datenbytes im Empfangsdatenbereich, die Länge im Empfangslängenregister und als Quittung ins Empfangssteuerregister und ins Empfangsstatusregister 16#06 ein. Die Absendernummer der Broadcastnachricht kann im Absenderregister abgelesen werden. Damit der parallele Zugriff von Ωmega und CAN-Prozessor auf den Datenbereich konfliktfrei abläuft, muß vom Ωmega zum Starten des Empfangsvorgangs ins Empfangsstatusregister (zuerst!) und ins Empfangssteuerregister 16#07 eingetragen werden. Entsprechend muß bei der Abfrage, ob eine Nachricht empfangen wurde, das Empfangssteuerregister (zuerst abfragen!) und das Empfangsstatusregister den Wert 16#06 enthalten.

Falls am CAN eine Broadcastnachricht empfangen wurde, und das Empfangssteuerregister nicht auf 16#07 steht, dann wird diese Nachricht ignoriert und nicht in den Empfangsdatenspeicher geschrieben.

Beispiel: Ωmega 3 soll eine Broadcastmeldung empfangen. Die nächste Broadcastmeldung kommt von Ωmega 7 mit dem Wert 16#79E47F.

Dazu wird im Ωmega 3 im Datenbereich vom Ωmega 3 ins Empfangsstatusregister

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_STATREG_EMPF

und ins Empfangssteuerregister

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_STEUREG_EMPF

der Wert 16#07 eingetragen.

Nachdem die Nachricht vom CAN empfangen wurde, steht im Empfangslängenregister

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_LENG_EMPF

die Länge 16#03, im Absenderregister

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_ABS_BRC

die Absendernummer 16#07 und im Empfangsdatenbereich in

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_EMPF_DB_0 : 16#7F

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_EMPF_DB_1 : 16#E4

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_EMPF_DB_2 : 16#79.

Der erfolgte Empfang wird durch 16#06 im Empfangssteuerregister

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_STEUREG_EMPF

und Empfangsstatusregister

_CAN_CTRL_OPT.a_OMEGA_0_15[3].b_O_STATREG_EMPF

angezeigt.

4.8 Betrieb des FIFO-Buffers

4.8.1 Adressierung

Für die Nutzung der FIFO-Funktionalität ist ein Registerbereich von 512 Bytes im Kommunikations-RAM der CAN-Anschaltung reserviert. Dieser Registerbereich hat folgende Bedeutung:

Adresse / Register	Bedeutung
*.a_FIFO_0_31[0].b_BYTE0	Lesen-Index
*.a_FIFO_0_31[0].b_BYTE1	Schreiben-Index
*.a_FIFO_0_31[0].w_ID	Identifier und Längeninformation Nachricht 0
*.a_FIFO_0_31[0].b_DATABYTE0	Datenbyte 0 Nachricht 0
*.a_FIFO_0_31[0].b_DATABYTE1	Datenbyte 1 Nachricht 0
*.a_FIFO_0_31[0].b_DATABYTE2	Datenbyte 2 Nachricht 0
*.a_FIFO_0_31[0].b_DATABYTE3	Datenbyte 3 Nachricht 0
*.a_FIFO_0_31[0].b_DATABYTE4	Datenbyte 4 Nachricht 0
*.a_FIFO_0_31[0].b_DATABYTE5	Datenbyte 5 Nachricht 0
*.a_FIFO_0_31[0].b_DATABYTE6	Datenbyte 6 Nachricht 0
*.a_FIFO_0_31[0].b_DATABYTE7	Datenbyte 7 Nachricht 0
*.a_FIFO_0_31[0].d_DWORD	reserviert Nachricht 0
*.a_FIFO_0_31[1].b_BYTE0	reserviert Nachricht 1
*.a_FIFO_0_31[1].b_BYTE1	reserviert Nachricht 1
*.a_FIFO_0_31[1].w_ID	Identifier und Längeninformation Nachricht 1
*.a_FIFO_0_31[1].b_DATABYTE0	Datenbyte 0 Nachricht 1
*.a_FIFO_0_31[1].b_DATABYTE1	Datenbyte 1 Nachricht 1
*.a_FIFO_0_31[1].b_DATABYTE2	Datenbyte 2 Nachricht 1
*.a_FIFO_0_31[1].b_DATABYTE3	Datenbyte 3 Nachricht 1
*.a_FIFO_0_31[1].b_DATABYTE4	Datenbyte 4 Nachricht 1
*.a_FIFO_0_31[1].b_DATABYTE5	Datenbyte 5 Nachricht 1
*.a_FIFO_0_31[1].b_DATABYTE6	Datenbyte 6 Nachricht 1
*.a_FIFO_0_31[1].b_DATABYTE7	Datenbyte 7 Nachricht 1
*.a_FIFO_0_31[1].d_DWORD	reserviert Nachricht 1
•	•
•	•
•	•
*.a_FIFO_0_31[31].b_BYTE0	reserviert Nachricht 31
*.a_FIFO_0_31[31].b_BYTE1	reserviert Nachricht 31
*.a_FIFO_0_31[31].w_ID	Identifier und Längeninformation Nachricht 31
*.a_FIFO_0_31[31].b_DATABYTE0	Datenbyte 0 Nachricht 31
*.a_FIFO_0_31[31].b_DATABYTE1	Datenbyte 1 Nachricht 31
*.a_FIFO_0_31[31].b_DATABYTE2	Datenbyte 2 Nachricht 31
*.a_FIFO_0_31[31].b_DATABYTE3	Datenbyte 3 Nachricht 31
*.a_FIFO_0_31[31].b_DATABYTE4	Datenbyte 4 Nachricht 31
*.a_FIFO_0_31[31].b_DATABYTE5	Datenbyte 5 Nachricht 31
*.a_FIFO_0_31[31].b_DATABYTE6	Datenbyte 6 Nachricht 31
*.a_FIFO_0_31[31].b_DATABYTE7	Datenbyte 7 Nachricht 31
*.a_FIFO_0_31[31].d_DWORD	reserviert Nachricht 31

* entspricht zum Beispiel _CAN_CTRL_OPT.

Aufbau von Identifier und Längeninformation:

*.a_FIFO_0_31[0].w_ID															
IDENT_1			RB	LENGTH				IDENT_0_H				IDENT_0_L			
Id2	Id1	Id0	RB	L3	L2	L1	L0	Id10	Id9	Id8	Id7	Id6	Id5	Id4	Id3

Im Eintrag LENGTH steht die Anzahl der Datenbytes der CAN-Nachricht.

4.8.2 Betrieb

CAN-Nachrichten, die in den FIFO-Buffer eingetragen werden, haben Identifier, die der CAN-Anschaltung nicht im einzelnen bekannt sind.

Diese Nachrichten werden der Reihenfolge nach in den FIFO-Buffer eingetragen. Die Nachricht 0 wird ab *.a_FIFO_0_31[0].w_ID, die Nachricht 1 ab *.a_FIFO_0_31[1].w_ID eingetragen, usw...

Nach jeder eingetragenen Nachricht wird der Schreiben-Index *.a_FIFO_0_31[0].b_BYTE1 um 1 erhöht.

Wird eine Nachricht aus dem FIFO-Buffer gelesen, muß von der **Applikation** der Lesen-Index *.a_FIFO_0_31[0].b_BYTE0 um 1 erhöht werden.

Ist der Schreiben-Index um 1 kleiner als der Lesen-Index, wird keine Nachricht mehr eingetragen. (Es ist zu beachten, daß bei der Betrachtung von 5 Bits 31 + 1 = 0 ist.)

Wird jetzt eine Nachricht gelesen, muß der Lesen-Index von der **Applikation** um 1 erhöht werden und die nächste empfangene Nachricht wird wieder in den FIFO-Buffer eingetragen (sowie der Schreiben-Index um 1 erhöht).

Beispiel:

Der Lesen-Index steht auf 31, der Schreiben-Index steht auf 29.

Jetzt wird die nächste, einzutragende Nachricht empfangen.

Diese Nachricht wird ab

_CAN_CTRL_OPT.a_FIFO_0_31[29].w_ID

eingetragen und der Schreiben-Index

_CAN_CTRL_OPT.a_FIFO_0_31[0].b_BYTE1

von 29 (16#1D) auf 30 (16#1E) erhöht.

Nun wird eine weitere einzutragende Nachricht empfangen.

Der Schreiben-Index

_CAN_CTRL_OPT.a_FIFO_0_31[0].b_BYTE1

ist jetzt um 1 kleiner als der Lesen-Index

_CAN_CTRL_OPT.a_FIFO_0_31[0].b_BYTE0

und die Nachricht wird nicht eingetragen.

Jetzt wird eine Nachricht von der Applikation gelesen. Dabei muß **von** der **Applikation** der **Lesen-Index um 1 erhöht** werden, in diesem Fall muß der Lesen-Index

_CAN_CTRL_OPT.a_FIFO_0_31[0].b_BYTE0

wieder 0 (16#00) werden

Beispiel: (31+1) AND 0x1F ergibt binär gesehen

	2#0001 1111	31	16#1F
+	2#0000 0001	01	16#01
	2#0010 0000	32	16#20
AND	2#0001 1111	31	16#1F
=	2#0000 0000	00	16#00

Nun wird eine weitere, einzutragende Nachricht empfangen. Der Schreiben-Index

_CAN_CTRL_OPT.a_FIFO_0_31[0].b_BYTE1

steht noch immer auf 30 (16#1E).

Diese Nachricht wird ab

_CAN_CTRL_OPT.a_FIFO_0_31[30].w_ID

eingetragen und der Schreiben-Index

_CAN_CTRL_OPT.a_FIFO_0_31[0].b_BYTE1

wird auf 31 (16#1F) erhöht.

5 FUNKTIONSBAUSTEINE FÜR CAN

5.1 Übersicht

Zusätzlich zu den Standardfunktionen können Sie herstellerdefinierte Funktionen verwenden, wenn Sie herstellerdefinierte Bibliotheken in einem Projekt angemeldet haben.

HINWEIS

Das Anmelden von Bibliotheken ist in der allgemeinen Hilfe beschrieben.

Folgende Funktionsbausteine für CAN sind verfügbar:

Funktion	**Kurzbeschreibung**
CAN_BKR_INIT	CAN BKR Initialisierung
CAN_DIG_INPUT_INIT	Initialisierung der Daten von einem CAN-DI-16-01/2-Modul
CAN_DIG_OUTPUT_INIT	Initialisierung der Daten von einem CAN-DO-16-01/2-Modul
CAN_DRIVE_INIT	Initialisierung der Daten von einem Regler
CAN_INIT	Initialisierung der dezentralen I/O-Module und Regler
CAN_INIT_FB1	wird von FB CAN_INIT benutzt
CAN_INIT_FB2	wird von FB CAN_INIT benutzt
CAN_OBJ_READ	Lesen eines Objektes des Reglers
CAN_OBJ_WRITE	Schreiben eines Objektes des Reglers
CAN_PA_LIST_READ	Lesen der aktuellen Sollwert-Parameter-nummern-Liste für die zyklische Kommuni-kation im Regler
CAN_PA_LIST_WRITE	Schreiben einer neuen Sollwert-Parameter-nummern-Liste für die zyklische Kommuni-kation in den Regler
CAN_PAR_READ	Lesen eines Parameters des Reglers
CAN_PAR_WRITE	Schreiben eines Parameters des Reglers
CAN_PD_COMM	Prozeßdatenkommunikation über CAN zwi-schen Ωmega und Regler
CAN_PE_LIST_READ	Lesen der aktuellen Istwert-Parameternum-mern-Liste für die zyklische Kommunikation im Regler
CAN_PE_LIST_WRITE	Schreiben einer neuen Istwert-Parameter-nummern-Liste für die zyklische Kommuni-kation in den Regler
CAN_SD_CONTROL	Überwacht Bedarfsdatenkommunikation eines Antriebs

5.2 CAN_BKR_INIT

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um die Daten eines BKR bzw. DSM (Scheibenläufermotor) für die CAN-Initialisierung aufzubereiten.

HINWEIS

Dieser FB wird zusammen mit dem FB CAN_INIT eingesetzt.

Parameter Eingang	Datentyp	Beschreibung
us_BKR_NR	USINT 0 bis 15	Gerätenummer der CAN-Anschaltung des BKR
t_WR_TIME0	TIME 0 bis 255 ms	Sollwert-Zykluszeit
t_RD_TIME0	TIME 0 bis 255 ms	Istwert-Zykluszeit

Parameter Ausgang	Datentyp	Beschreibung
_CAN_DRIVE_DATA	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den BKR mit der Gerätenummer us_BKR_NR
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit

Der FB CAN_BKR_INIT bereitet die zur CAN-Initialisierung notwendigen Daten für einen BKR auf und stellt die Daten am Ausgang _CAN_DRIVE_DATA (für den FB CAN_INIT) zur Verfügung. Die Initialisierung der CAN-Anschaltung mit den Daten für den BKR erfolgt durch den FB CAN_INIT.

Eingang us_BKR_NR:

Am Eingang us_BKR_NR wird die Gerätenummer des BKR (Adresse am CAN-Bus; eingestellt über Parameter 41 im BKR) angegeben.

Eingang t_WR_TIME0:

Am Eingang t_WR_TIME0 wird die Zykluszeit für den Sollwert 0 in ms angegeben (z. B. 20 ms → t_WR_TIME0 = 20ms).

Eingang t_RD_TIME0:

Der Eingang t_RD_TIME0 wird zur Zeit nicht verwendet.

Die Istwertzykluszeit (für den Istwert 0) wird über Parameter 43 und 44 im BKR eingestellt.

Ausgang _CAN_DRIVE_DATA:

Am Ausgang _CAN_DRIVE_DATA werden die (aufbereiteten) Daten für die CAN-Initialisierung ausgegeben.

Am Ausgang _CAN_DRIVE_DATA wird eine Variable vom Datentyp CAN_INIT_DRIVE_BMSTRUCT angeschlossen.

HINWEIS

Diese Variable wird (je nach us_BKR_NR) an einem der Eingänge _CAN_DRIVE0 bis _CAN_DRIVE15 des FB CAN_INIT angeschlossen.

Beispiel:

Die Daten des BKR mit der Gerätenummer 12 (us_BKR_NR = 12) werden dem FB CAN_INIT übergeben.

→ FB CAN_BKR_INIT, Ausgang _CAN_DRIVE_DATA wird mit FB CAN_INIT, Eingang _CAN_DRIVE12 verbunden.

Ausgänge x_ERR, b_ERR:

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	**Fehler**
0	Ungültige Gerätenummer (us_BKR_NR nicht 0 bis 15)
1	Sollwertzykluszeit (t_WR_TIME0) größer als 255ms
2 - 7	Reserviert

5.3 CAN_DIG_INPUT_INIT

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um die Daten eines CAN-DI-16-01 bzw. CAN-DI-16-02 Moduls für die CAN-Initialisierung aufzubereiten.

HINWEIS

Dieser FB wird zusammen mit dem FB CAN_INIT eingesetzt.

Parameter Eingang	Datentyp	Beschreibung
us_DI_NR	USINT 0 bis 15	Gerätenummer des CAN-DI-16-xx Moduls
us_DI_MODE	USINT 2, 3	Betriebsart des CAN-DI-16-xx Moduls
t_DI_TIME	TIME 0 bis 255 ms [a)]	½ Zykluszeit des CAN-DI-16-xx Moduls (bei us_DI_MODE = 3)
w_DI_NEG_EDGE	WORD	Senden bei negativen Flanken
w_DI_POS_EDGE	WORD	Senden bei positiven Flanken

[a)] siehe Beschreibung Eingang t_DI_TIME

Parameter Ausgang	Datentyp	Beschreibung
_CAN_DI_DATA	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-DI-16-xx Modul mit der Gerätenummer us_DI_NR
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit

Der FB CAN_DIG_INPUT_INIT bereitet die zur CAN-Initialisierung notwendigen Daten für ein CAN-DI-16-xx Modul auf und stellt die Daten am Ausgang _CAN_DI_DATA (für den FB CAN_INIT) zur Verfügung.

Eingang us_DI_NR:

Am Eingang us_DI_NR wird die Gerätenummer des CAN-DI-16-xx Moduls (Adresse am CAN-Bus; eingestellt über DIP-Schalter auf dem CAN-DI-16-xx Modul) angegeben.

Eingang us_DI_MODE:

Am Eingang us_DI_MODE wird die Betriebsart des CAN-DI-16-xx Moduls eingestellt.

us_DI_MODE	Bedeutung
0, 1	Reserviert
2	Polling-Betrieb (azyklisches Anfordern des Eingangswortes durch den Anwender)
3	Zyklisches Anfordern des Eingangswortes durch die CAN-Anschaltung
4 -255	Reserviert

Eingang t_DI_TIME:

Die Zykluszeit für das zyklisches Anfordern des Eingangswortes durch die CAN-Anschaltung (us_DI_MODE = 3) wird am Eingang t_DI_TIME festgelegt. Es ist ein Wert zwischen 1 und 255 einzustellen.

Die Zykluszeit wird wie folgt berechnet:

Zykluszeit = t_DI_TIME * 2 ms

Beispiel:

t_DI_TIME = 10 ms $\rightarrow$ Zykluszeit = 20 ms

HINWEIS

Eine Überprüfung der Werte am Eingang t_DI_TIME findet nicht statt.

Beim Polling-Betrieb (azyklisches Anfordern des Eingangswortes durch den Anwender) muß t_DI_TIME = 0 sein.

Eingang w_DI_NEG_EDGE:

Am Eingang w_DI_NEG_EDGE kann angegeben werden, bei welchen Eingängen des CAN-DI-16-xx Moduls eine negative Flanke das Senden des Eingangswortes auslöst. Voreinstellung ist 16#0000, d. h. es wird bei negativen Flanken das Eingangswort nicht gesendet. [1)]

Eingang w_DI_POS_EDGE:

Am Eingang w_DI_POS_EDGE kann angegeben werden, bei welchen Eingängen des CAN-DI-16-xx Moduls eine positive Flanke das Senden des Eingangswortes auslöst. Voreinstellung ist 16#0000, d. h. es wird bei positiven Flanken das Eingangswort nicht gesendet. [1)]

Ausgang _CAN_DI_DATA:

Am Ausgang _CAN_DI_DATA werden die (aufbereiteten) Daten für die CAN-Initialisierung ausgegeben.

Am Ausgang _CAN_DI_DATA wird eine Variable vom Datentyp CAN_INIT_IO_BMSTRUCT angeschlossen.

[1)] Bei us_DI_MODE = 3 wird durch diese Funktion zusätzliche CAN-Buslast erzeugt.

HINWEIS

Diese Variable wird (je nach us_DI_NR) an einem der Eingänge _CAN_DIG_INOUT0 bis _CAN_DIG_INOUT15 des FB CAN_INIT angeschlossen.

Beispiel:

Die Daten des CAN-DI-16-xx Moduls 12 (us_DI_NR = 12) werden dem FB CAN_INIT übergeben.

→ FB CAN_DIG_INPUT_INIT, Ausgang _CAN_DI_DATA wird mit FB CAN_INIT, Eingang _CAN_DIG_INOUT12 verbunden.

Ausgänge x_ERR, b_ERR:

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Ungültige Gerätenummer (us_DI_NR nicht 0 bis 15)
1	Ungültige Betriebsart (nicht 2, 3)
2	Keine Zykluszeit bei us_DI_MODE = 3
3	Es wurde eine Zykluszeit bei us_DI_MODE ≠ 3 angegeben
4 - 7	Reserviert

5.4 CAN_DIG_OUTPUT_INIT

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um die Daten eines CAN-DI-16-01 bzw. CAN-DI-16-02 Moduls für die CAN-Initialisierung aufzubereiten.

HINWEIS

Dieser FB wird zusammen mit dem FB CAN_INIT eingesetzt.

Parameter Eingang	Datentyp	Beschreibung
us_DO_NR	USINT 0 bis 15	Gerätenummer des CAN-DO-16-xx Moduls
us_DO_MODE	USINT 2, 3	Betriebsart des CAN-DO-16-xx Moduls
t_DO_TIME	TIME0 bis 255 ms [a)]	½ Zykluszeit des CAN-DO-16-xx Moduls (bei us_DO_MODE = 3)
w_DO_OUT_EN	WORD	Freigabe der Ausgänge des CAN-DO-16-xx Moduls

a) siehe Beschreibung Eingang t_DO_TIME

Parameter Ausgang	Datentyp	Beschreibung
_CAN_DO_DATA	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-DO-16-xx Modul mit der Gerätenummer us_DO_NR
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit

Der FB CAN_DIG_OUTPUT_INIT bereitet die zur CAN-Initialisierung notwendigen Daten für ein CAN-DO-16-xx Modul auf und stellt die Daten am Ausgang _CAN_DO_DATA (für den FB CAN_INIT) zur Verfügung.

Eingang us_DO_NR:

Am Eingang us_DO_NR wird die Gerätenummer des CAN-DO-16-xx Moduls (Adresse am CAN-Bus; eingestellt über DIP-Schalter auf dem CAN-DO-16-xx Modul) angegeben.

Eingang us_DO_MODE:

Am Eingang us_DO_MODE wird die Betriebsart des CAN-DO-16-xx Moduls eingestellt.

us_DO_MODE	Bedeutung
0, 1	Reserviert
2	Polling-Betrieb (azyklisches Senden des Ausgangswortes durch den Anwender)
3	Zyklisches Senden des Ausgangswortes durch die CAN-Anschaltung
4 - 255	Reserviert

Eingang t_DO_TIME:

Die Zykluszeit für zyklisches Senden des Ausgangswortes durch die CAN-Anschaltung (us_DO_MODE = 3) wird am Eingang t_DO_TIME festgelegt. Es ist ein Wert zwischen 1 und 255 einzustellen.

Die Zykluszeit wird wie folgt berechnet:

Zykluszeit = t_DO_TIME * 2 ms

Beispiel:

t_DO_TIME = 10 ms $\rightarrow$ Zykluszeit = 20 ms

HINWEIS

Eine Überprüfung der Werte am Eingang t_DI_TIME findet nicht statt.

Beim Polling-Betrieb (azyklisches Senden des Ausgangswortes durch den Anwender) muß t_DI_TIME = 0 sein.

Eingang w_DO_OUT_EN:

Am Eingang w_DO_OUT_EN kann angegeben werden, welche Ausgänge des CAN-DO-16-xx Moduls zum Schalten freigegeben sind. Voreinstellung ist 16#FFFF, d. h. alle Ausgänge sind zum Schalten freigegeben.

Ausgang _CAN_DO_DATA:

Am Ausgang _CAN_DO_DATA werden die (aufbereiteten) Daten für die CAN-Initialisierung ausgegeben.

Am Ausgang _CAN_DO_DATA wird eine Variable vom Datentyp CAN_INIT_IO_BMSTRUCT angeschlossen.

HINWEIS

Diese Variable wird (je nach us_DO_NR) an einem der Eingänge _CAN_DIG_INOUT0 bis _CAN_DIG_INOUT15 des FB CAN_INIT angeschlossen.

Beispiel:

Die Daten des CAN-DO-16-xx Moduls 7 (us_DO_NR = 7) werden dem FB CAN_INIT übergeben.

→ FB CAN_DIG_OUTPUT_INIT, Ausgang _CAN_DO_DATA wird mit FB CAN_INIT, Eingang _CAN_DIG_INOUT7 verbunden.

Ausgänge x_ERR, b_ERR:

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Gerätenummer (us_DO_NR nicht 0 bis 15)
1	Ungültige Betriebsart (nicht 2, 3)
2	Keine Zykluszeit bei us_DO_MODE = 3
3	Es wurde eine Zykluszeit bei us_DO_MODE ≠ 3 angegeben
4 - 7	Reserviert

5.5 CAN_DRIVE_INIT

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um die Daten eines Reglers (Drive-Controller) für die CAN-Initialisierung aufzubereiten.

HINWEIS

Dieser FB wird zusammen mit dem FB CAN_INIT eingesetzt.

Parameter Eingang	Datentyp	Beschreibung
us_DRIVE_NR	USINT 0 bis 15	Gerätenummer der CAN-Anschaltung des Reglers
u_WR_PAR_NR0	UINT	Sollwert-Parameternummer 0
us_WR_FORMAT0	USINT 0, 1	Sollwert-Format 0
t_WR_TIME0	TIME 0 bis 255 ms	Sollwert-Zykluszeit 0
u_WR_PAR_NR1	UINT	Sollwert-Parameternummer 1
us_WR_FORMAT1	USINT 1, 2	Sollwert-Format 1
t_WR_TIME1	TIME 0 bis 255 ms	Sollwert-Zykluszeit 1
u_WR_PAR_NR2	UINT	Sollwert-Parameternummer 2
us_WR_FORMAT2	USINT 1, 2	Sollwert-Format 2
t_WR_TIME2	TIME 0 bis 255 ms	Sollwert-Zykluszeit 2
u_WR_PAR_NR3	UINT	Sollwert-Parameternummer 3
us_WR_FORMAT3	USINT 1, 2	Sollwert-Format 3
t_WR_TIME3	TIME 0 bis 255 ms	Sollwert-Zykluszeit 3
u_RD_PAR_NR0	UINT	Istwert-Parameternummer 0
us_RD_FORMAT0	USINT 0, 1	Istwert-Format 0
t_RD_TIME0	TIME 0 bis 255 ms	Istwert-Zykluszeit 0
u_RD_PAR_NR1	UINT	Istwert-Parameternummer 1
us_RD_FORMAT1	USINT 1, 2	Istwert-Format 1
t_RD_TIME1	TIME 0 bis 255 ms	Istwert-Zykluszeit 1
u_RD_PAR_NR2	UINT	Istwert-Parameternummer 2
us_RD_FORMAT2	USINT 1, 2	Istwert-Format 2

Parameter Eingang	Datentyp	Beschreibung
t_RD_TIME2	TIME 0 bis 255 ms	Istwert-Zykluszeit 2
u_RD_PAR_NR3	UINT	Istwert-Parameternummer 3
us_RD_FORMAT3	USINT 1, 2	Istwert-Format 3
t_RD_TIME3	TIME 0 bis 255 ms	Istwert-Zykluszeit 3

Parameter Ausgang	Datentyp	Beschreibung
_CAN_DRIVE_DATA	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit der Gerätenummer us_DRIVE_NR
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit

Der FB CAN_DRIVE_INIT bereitet die zur CAN-Initialisierung notwendigen Daten für einen Regler auf und stellt die Daten am Ausgang _CAN_DRIVE_DATA (für den FB CAN_INIT) zur Verfügung.

Eingang us_DRIVE_NR:

Am Eingang us_DRIVE_NR wird die Gerätenummer des Reglers (Adresse am CAN-Bus; eingestellt über DIP-Schalter auf der CAN-Anschaltung des Reglers) angegeben.

Eingänge u_WR_PAR_NR0, us_WR_FORMAT0, t_WR_TIME0:

Am Eingang u_WR_PAR_NR0 wird die Parameternummer des Sollwerts 0 angegeben.

Am Eingang us_WR_FORMAT0 wird das Format des Sollwerts 0 angegeben.

HINWEIS

Der Sollwert 0 darf das Format Byte oder Wort haben.
Die Sollwerte 1, 2 und 3 dürfen das Format Wort oder Doppelwort haben.

Formatangabe:	us_WR_FORMAT0 = 0:	Format Byte	(Sollwert 0)
	us_WR_FORMAT0 = 1:	Format Wort	(Sollwert 0, 1, 2 und 3)
	us_WR_FORMAT0 = 2:	Format Doppelwort	(Sollwert 1, 2 und 3)

Am Eingang t_WR_TIME0 wird die Zykluszeit für den Sollwert 0 in ms angegeben (z. B. 20 ms → t_WR_TIME0 = 20ms).

Eingänge u_WR_PAR_NR1, us_WR_FORMAT1, t_WR_TIME1:

Am Eingang u_WR_PAR_NR1 wird die Parameternummer des Sollwerts 1 angegeben.

Am Eingang us_WR_FORMAT1 wird das Format des Sollwerts 1 angegeben.

Am Eingang t_WR_TIME1 wird die Zykluszeit des Sollwerts 1 in ms angegeben.

Eingänge u_WR_PAR_NR2, us_WR_FORMAT2, t_WR_TIME2:

Am Eingang u_WR_PAR_NR2 wird die Parameternummer des Sollwerts 2 angegeben.

Am Eingang us_WR_FORMAT2 wird das Format des Sollwerts 2 angegeben.

Am Eingang t_WR_TIME2 wird die Zykluszeit des Sollwerts 2 in ms angegeben.

Eingänge u_WR_PAR_NR3, us_WR_FORMAT3, t_WR_TIME3:

Am Eingang u_WR_PAR_NR3 wird die Parameternummer des Sollwerts 3 angegeben.

Am Eingang us_WR_FORMAT3 wird das Format des Sollwerts 3 angegeben.

Am Eingang t_WR_TIME3 wird die Zykluszeit des Sollwerts 3 in ms angegeben.

Eingänge u_RD_PAR_NR0, us_RD_FORMAT0, t_RD_TIME0:

Am Eingang u_RD_PAR_NR0 wird die Parameternummer des Istwerts 0 angegeben.

Am Eingang us_RD_FORMAT0 wird das Format des Istwerts 0 angegeben.

HINWEIS

Der Istwert 0 darf das Format Byte oder Wort haben.
Die Istwerte 1, 2 und 3 dürfen das Format Wort oder Doppelwort haben.

Formatangabe:	us_RD_FORMAT0 = 0:	Format Byte	(Istwert 0)
	us_RD_FORMAT0 = 1:	Format Wort	(Istwert 0, 1, 2 und 3)
	us_RD_FORMAT0 = 2:	Format Doppelwort	(Istwert 1, 2 und 3)

Am Eingang t_RD_TIME0 wird die Zykluszeit für den Istwert 0 in ms angegeben (z. B. 20 ms → t_RD_TIME0 = 20ms).

Eingänge u_RD_PAR_NR1, us_RD_FORMAT1, t_RD_TIME1:

Am Eingang u_RD_PAR_NR1 wird die Parameternummer des Istwerts 1 angegeben.

Am Eingang us_RD_FORMAT1 wird das Format des Istwerts 1 angegeben.

Am Eingang t_RD_TIME1 wird die Zykluszeit des Istwerts 1 in ms angegeben.

Eingänge u_RD_PAR_NR2, us_RD_FORMAT2, t_RD_TIME2:

Am Eingang u_RD_PAR_NR2 wird die Parameternummer des Istwerts 2 angegeben.

Am Eingang us_RD_FORMAT2 wird das Format des Istwerts 2 angegeben.

Am Eingang t_RD_TIME2 wird die Zykluszeit des Istwerts 2 in ms angegeben.

Eingänge u_RD_PAR_NR3, us_RD_FORMAT3, t_RD_TIME3:

Am Eingang u_RD_PAR_NR3 wird die Parameternummer des Istwerts 3 angegeben.

Am Eingang us_RD_FORMAT3 wird das Format des Istwerts 3 angegeben.

Am Eingang t_RD_TIME3 wird die Zykluszeit des Istwerts 3 in ms angegeben.

Ausgang _CAN_DRIVE_DATA:

Am Ausgang _CAN_DRIVE_DATA werden die (aufbereiteten) Daten für die CAN-Initialisierung ausgegeben.

Am Ausgang _CAN_DRIVE_DATA wird eine Variable vom Datentyp CAN_INIT_DRIVE_BMSTRUCT angeschlossen.

HINWEIS

Diese Variable wird (je nach us_DRIVE_NR) an einem der Eingänge _CAN_DRIVE0 bis _CAN_DRIVE15 des FB CAN_INIT angeschlossen.

Beispiel:

Die Daten des Reglers mit CAN-Anschaltung und Gerätenummer 7 (us_DRIVE_NR = 7) werden dem FB CAN_INIT übergeben.

→ FB CAN_DRIVE_INIT, Ausgang _CAN_DRIVE_DATA wird mit FB CAN_INIT, Eingang _CAN_DRIVE7 verbunden.

Ausgänge x_ERR, b_ERR:

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Ungültige Gerätenummer (us_DRIVE_NR größer 15)
1	Ungültiges Format Sollwert 0 (us_WR_FORMAT0 nicht 0 oder 1)
2	Ungültiges Format Sollwert 1, 2 oder 3 (us_WR_FORMAT1, us_WR_FORMAT2 und/oder us_WR_FORMAT3 nicht 1 oder 2)
3	Ungültiges Format Istwert 0 (us_RD_FORMAT0 nicht 0 oder 1)
4	Ungültiges Format Istwert 1, 2 oder 3 (us_RD_FORMAT1, us_RD_FORMAT2 und/oder us_RD_FORMAT3 nicht 1 oder 2)
5	Mindestens eine Soll- oder Istwert-Zykluszeit ist größer 255ms (t_WR_TIMEx, t_RD_TIMEx > 255 ms; x = 0, 1, 2, 3)
6 - 7	Reserviert

5.6 CAN_INIT

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um eine CAN-Anschaltung am Ωmega Drive-Line II zu initialisieren.

Dabei kann die Kommunikation zu CAN-DI-16-xx Modulen und CAN-DO-16-xx Modulen (max. 16, im folgenden CAN-IO-Module) sowie Reglern mit CAN-Anschaltung (max. 16, im folgenden Regler) am CAN-Bus eingerichtet werden.
(CAN-DI-16-xx: Digital Input Modul, CAN-DO-16-xx: Digital Output Modul)

HINWEIS

Der FB CAN_INIT verwendet die Bibliothek BM_TYPES_20bd00 oder höher.

Parameter Eingang	Datentyp	Beschreibung
_BASE_INIT	CAN_INIT_BMSTRUCT	Initialisierungsdaten für die CAN-Anschaltung
_BASE_CTRL	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
us_NR_OF_IO_TO_INIT	USINT 0, 1 bis 16	Anzahl der CAN-IO-Module am CAN-Bus
us_MAX_IO_NR_COMM	USINT 0, 1 bis 16	(max. Gerätenummer + 1) der CAN-IO-Module am CAN-Bus
us_MAX_DRIVE_NR_COMM	USINT 0, 1 bis 16	(max. Gerätenummer + 1) der Regler mit CAN-Anschaltung am CAN-Bus
us_MAX_OMEGA_NR_COMM	USINT0, 1 bis 16	(max. Gerätenummer + 1) der Ωmegas mit CAN-Anschaltung am CAN-Bus
us_NR_OF_OMEGAS	USINT 0, 1 bis 16	Anzahl der Ωmegas mit CAN-Anschaltung am CAN-Bus
x_RECAL_SL	BOOL	Rekalibrierslave
b_ACCEPT_MASK	BYTE	Acceptance Mask
b_ACCEPT_CODE	BYTE	Acceptance Code
b_BIT_TIMING0	BYTE	Bus-Timing
b_BIT_TIMING1	BYTE	Bus-Timing
us_BAUDRATE	USINT 0, 1 bis 6	Baudrate
_CAN_DIG_INOUT0	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 0
_CAN_DIG_INOUT1	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 1
_CAN_DIG_INOUT2	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 2

Parameter Eingang	Datentyp	Beschreibung
_CAN_DIG_INOUT3	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 3
_CAN_DIG_INOUT4	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 4
_CAN_DIG_INOUT5	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 5
_CAN_DIG_INOUT6	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 6
_CAN_DIG_INOUT7	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 7
_CAN_DIG_INOUT8	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 8
_CAN_DIG_INOUT9	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 9
_CAN_DIG_INOUT10	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 10
_CAN_DIG_INOUT11	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 11
_CAN_DIG_INOUT12	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 12
_CAN_DIG_INOUT13	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 13
_CAN_DIG_INOUT14	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 14
_CAN_DIG_INOUT15	CAN_INIT_IO_BMSTRUCT	Initialisierungsdaten für das CAN-IO-Modul mit Gerätenummer 15
_CAN_DRIVE0	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 0
_CAN_DRIVE1	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 1
_CAN_DRIVE2	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 2
_CAN_DRIVE3	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 3
_CAN_DRIVE4	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 4
_CAN_DRIVE5	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 5
_CAN_DRIVE6	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 6
_CAN_DRIVE7	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 7
_CAN_DRIVE8	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 8
_CAN_DRIVE9	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 9
_CAN_DRIVE10	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 10
_CAN_DRIVE11	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 11
_CAN_DRIVE12	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 12

Parameter Eingang	Datentyp	Beschreibung
_CAN_DRIVE13	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 13
_CAN_DRIVE14	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 14
_CAN_DRIVE15	CAN_INIT_DRIVE_BMSTRUCT	Initialisierungsdaten für den Regler mit Gerätenummer 15
t_TIME	TIME	Überwachungszeit

Parameter Ausgang	Datentyp	Beschreibung
_BASE_INIT	CAN_INIT_BMSTRUCT	Initialisierungsdaten für die CAN-Anschaltung
_BASE_CTRL	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
_CAN_DATA	CAN_INIT_DATA_BMSTRUCT	Initialisierungsdaten des CAN
w_ERR_IO	WORD	Fehlerwort CAN-IO-Module
w_ERR1_DRIVES	WORD	Fehlerwort 1 Regler
w_ERR2_DRIVES	WORD	Fehlerwort 2 Regler
w_ERR	WORD	Fehlerwort allgemein
x_ERR	BOOL	Fehlerbit
w_OK_IO	WORD	OK-Wort CAN-IO-Module
w_OK_DRIVES	WORD	OK-Wort Regler
x_OK	BOOL	OK-Bit

Der FB CAN_INIT führt die Initialisierung einer CAN-Anschaltung am Ωmega Drive-Line II und der Kommunikation zu CAN-IO-Modulen, Ωmega mit CAN-Anschaltung und Reglern mit CAN-Anschaltung durch. Dazu werden folgende Schritte abgearbeitet:

- Initialisierung des CAN-Anschaltung am Ωmega Drive-Line II
- Initialisierung der CAN-IO-Module CAN-DI-16-xx. Die Initialisierungsdaten werden an den FBs CAN_DIG_INPUT_INIT angegeben.
- Initialisierung der CAN-IO-Module CAN-DO-16-xx. Die Initialisierungsdaten werden an den FBs CAN_DIG_OUTPUT_INIT angegeben.
- Initialisierung der Regler mit CAN-Anschaltung. Die Initialisierungsdaten werden an den FBs CAN_DRIVE_INIT angegeben
- Überprüfen der Kommunikation zu den Reglern mit CAN-Anschaltung.
- Überprüfen der Kommunikation zu den CAN-IO-Modulen.

Initialisierung CAN-Anschaltung am Ωmega Drive-Line II:

Ein-/Ausgang _BASE_INIT:

An _BASE_INIT muß eine globale Variable vom Datentyp CAN_INIT_BMSTRUCT angeschlossen werden.

Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_INIT_OPT AT %MB3.3000000 : CAN_INIT_BMSTRUCT;
```

dabei ist:

`_CAN_INIT_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_INIT_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Ein-/Ausgang _BASE_CTRL:

An _BASE_CTRL muß eine globale Variable vom Datentyp CAN_CTRL_BMSTRUCT angeschlossen werden.

Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;
```

dabei ist:

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Eingang us_NR_OF_IO_TO_INIT:

Am Eingang us_NR_OF_IO_TO_INIT wird die Anzahl der zu initialisierenden CAN-IO-Module angegeben. Wird der Eingang nicht belegt, ergibt sich die Voreinstellung us_NR_OF_IO_TO_INIT = 0.

Liegt die Anzahl der zu initialisierenden CAN-IO-Module nicht im Bereich 0 (keine), 1 bis 16, wird im Fehlerwort w_ERR das Bit 1 gesetzt.

Eingang us_MAX_IO_NR_COMM:

Dieser Eingang gibt der CAN-Anschaltung an, bis zu welchem CAN-IO-Modul die CAN-Anschaltung Daten bearbeitet.

Am Eingang us_MAX_IO_NR_COMM wird die "größte Gerätenummer eines CAN-IO-Moduls am CAN-Bus + 1" angegeben.

Wird der Eingang nicht belegt, ergibt sich die Voreinstellung us_MAX_IO_NR_COMM = 0, d. h. es werden von keinem CAN-IO-Modul Daten bearbeitet.

Liegt die "größte Gerätenummer eines CAN-IO-Moduls am CAN-Bus + 1" nicht im Bereich 0 (keine), 1 bis 16, wird im Fehlerwort w_ERR das Bit 2 gesetzt.

Eingang us_MAX_DRIVE_NR_COMM:

Dieser Eingang gibt der CAN-Anschaltung an, bis zu welchem Regler die CAN-Anschaltung Daten bearbeitet.

Am Eingang us_MAX_DRIVE_NR_COMM wird die "größte Gerätenummer eines Reglers am CAN-Bus + 1" angegeben.

Wird der Eingang nicht belegt, ergibt sich die Voreinstellung us_MAX_DRIVE_NR_COMM = 0, d. h. es werden von keinem Regler Daten bearbeitet.

Liegt die "größte Gerätenummer eines Reglers am CAN-Bus + 1" nicht im Bereich 0 (keine), 1 bis 16, wird im Fehlerwort w_ERR das Bit 3 gesetzt.

Eingang us_MAX_OMEGA_NR_COMM:

Dieser Eingang gibt der CAN-Anschaltung an, bis zu welchem Ωmega die CAN-Anschaltung Daten bearbeitet.

Am Eingang us_MAX_OMEGA_NR_COMM wird die "größte Gerätenummer eines Ωmegas am CAN-Bus + 1" angegeben.

Wird der Eingang nicht belegt, ergibt sich die Voreinstellung us_MAX_OMEGA_NR_COMM = 0, d. h. es werden von keinem Ωmega Daten bearbeitet.

Liegt die "größte Gerätenummer eines Ωmegas am CAN-Bus + 1" nicht im Bereich 0 (keine), 1 bis 16, wird im Fehlerwort w_ERR das Bit 4 gesetzt.

Eingang us_NR_OF_OMEGAS:

Am Eingang us_NR_OF_OMEGAS wird angegeben wie viele Ωmegas (inkl. Master) am CAN-Bus angeschlossen sind. Diese Angabe dient der Überprüfung der Anzahl der Teilnehmer am CAN-Bus. Wird der Eingang nicht belegt, ergibt sich die Voreinstellung us_NR_OF_OMEGAS = 0. Ist die Anzahl der Ωmegas größer 16, wird im Fehlerwort w_ERR das Bit 5 gesetzt.

Eingang x_RECAL_SL:

Bei mehreren Ωmegas am CAN-Bus muß festgelegt werden welche Ωmegas Rekalibrierslaves sind und welcher Ωmega Rekalibriermaster ist, d. h. welcher Ωmega das zyklische Senden des Rekalibrier-Telegramms übernimmt.

Am Eingang x_RECAL_SL wird beim Rekalibriermaster FALSE angegeben und bei allen Rekalibrierslaves wird x_RECAL_SL = TRUE angegeben.

Voreinstellung ist x_RECAL_SL = FALSE, d. h. die zu initialisierende CAN-Anschaltung am Ωmega wird Rekalibriermaster.

Eingänge b_ACCEPT_MASK, b_ACCEPT_CODE:

An den Eingängen b_ACCEPT_MASK und b_ACCEPT_CODE kann der Akzeptanzfilter der CAN-Anschaltung eingestellt werden. Näheres hierzu entnehmen Sie bitte der Technischen Beschreibung und Betriebsanleitung CAN-M-01. Werden die Eingänge nicht belegt, ergeben sich die Voreinstellungen b_ACCEPT_MASK = 16#FF und b_ACCEPT_CODE = 16#FF, d. h. alle Objekte werden berücksichtigt.

Eingänge b_BIT_TIMING0, b_BIT_TIMING1, us_BAUDRATE:

Am Eingang us_BAUDRATE wird die Baudrate für den CAN-Bus eingestellt. Es darf maximal die Baudrate eingestellt werden, die alle Teilnehmer am CAN-Bus "verstehen".

Für 6 verschiedene Baudrates ist das Bus-Timing berechnet und wird vom FB CAN_INIT der CAN-Anschaltung übergeben.

Voreinstellung ist us_BAUDRATE = 3, d. h. die Baudrate 125 kBit/s wird eingestellt, wenn us_BAUDRATE nicht belegt ist.

Baudrate	us_BAUDRATE	b_BIT_TIMING0	b_BIT_TIMING1
	0	individuell	individuell
20 kBit/s	1	16#53	16#2F
50 kBit/s	2	16#47	16#2F
125 kBit/s	3	16#03	16#1C
250 kBit/s	4	16#01	16#1C
500 kBit/s	5	16#00	16#1C
1000 kBit/s	6	16#00	16#14

Wird bei der Initialisierung angegeben, daß sich CAN-IO-Module am CAN-Bus befinden und ist die Baudrate auf mehr als 250 kBit/s eingestellt, wird im Fehlerwort w_ERR das Bit 10 gesetzt.

D. h. ist der Eingang us_NR_OF_IO_TO_INIT $\neq 0$ **oder** us_MAX_IO_NR_COMM $\neq 0$ **und** us_BAUDRATE > 4, wird im Fehlerwort w_ERR das Bit 10 gesetzt.

Ist der Wert us_BAUDRATE > 6, wird im Fehlerwort w_ERR das Bit 6 gesetzt.

An den Eingängen b_BIT_TIMING0 und b_BIT_TIMING1 kann das Bus-Timing der CAN-Anschaltung individuell eingestellt werden. Näheres hierzu entnehmen Sie bitte der Technischen Beschreibung und Betriebsanleitung CAN-M-01.

Die Einstellungen dieser Eingänge wird übernommen, wenn der Eingang us_BAUDRATE = 0.

Die Einstellungen dieser Eingänge wird ignoriert, wenn der Eingang us_BAUDRATE mit einem Wert von 1 bis 6 belegt ist.

HINWEIS

Die Werte an den Eingängen b_BIT_TIMING0 und b_BIT_TIMING1 werden **nicht** übernommen, wenn der Eingang us_BAUDRATE $\neq 0$ ist.

Die Werte an den Eingängen b_BIT_TIMING0 und b_BIT_TIMING1 werden **nur** übernommen, wenn der Eingang us_BAUDRATE = 0 ist.

Am CAN-Bus können maximal 32 Teilnehmer angeschlossen werden. Ein Ωmega muß der Rekalibriermaster sein. Es können 31 Teilnehmer an dem CAN-Bus (zusätzlich zum Rekalibriermaster) angeschlossen werden. Eine Überprüfung der Anzahl der Teilnehmer am CAN-Bus findet während der Initialisierung statt.

Liegt die Anzahl der CAN-IO-Module, der Regler und der Ωmegas (inkl. Rekalibriermaster) außerhalb des Bereiches 0 (keine), 1 bis 32, wird im Fehlerwort w_ERR das Bit 7 gesetzt.

Ist Aufgrund der Eingangsbelegung ein Fehler in das Fehlerwort w_ERR eingetragen, wird die Überprüfung der Kommunikation zu den CAN-IO-Modulen und den Reglern nicht durchgeführt, der Ausgang x_OK bleibt FALSE, im Fehlerwort w_ERR wird das Bit 11 gesetzt und das Fehlerbit x_ERR wird gesetzt. Die Initialisierung der CAN-Anschaltung und der Kommunikation zu den CAN-IO-Modulen und den Reglern wurde nicht korrekt ausgeführt. Anhand der Angaben im Fehlerwort w_ERR (siehe unten) ist die Ursache zu suchen und zu beseitigen.

Initialisierung der CAN-IO-Module CAN-DI-16-xx und CAN-DO-16-xx:

Eingänge _CAN_DIG_INOUT0 bis _CAN_DIG_INOUT15:

An den Eingängen _CAN_DIG_INOUT0 bis _CAN_DIG_INOUT15 werden die Initialisierungsdaten der maximal 16 CAN-IO-Module angeschlossen. Die Initialisierungsdaten haben den Datentyp CAN_INIT_IO_BMSTRUCT und stehen am jeweiligen FB CAN_DIG_INPUT_INIT am Ausgang _CAN_DI_DATA oder am jeweiligen FB CAN_DIG_OUTPUT_INIT am Ausgang _CAN_DO_DATA zur Verfügung.

Initialisierung der Regler mit CAN-Anschaltung:

Eingänge _CAN_DRIVE0 bis _CAN_DRIVE15:

An den Eingängen _CAN_DRIVE0 bis _CAN_DRIVE15 werden die Initialisierungsdaten der maximal 16 Regler angeschlossen. Die Initialisierungsdaten haben den Datentyp CAN_INIT_DRIVE_BMSTRUCT und stehen am jeweiligen FB CAN_DRIVE_INIT (oder CAN_BKR_INIT) am Ausgang _CAN_DRIVE_DATA zur Verfügung.

Eingang t_TIME:

Am Eingang t_TIME wird die Überwachungszeit angegeben. Voreinstellung ist t_TIME = 10s. Ist die Initialisierung der CAN-Anschaltung und der Kommunikation zu den CAN-IO-Modulen, Ωmega mit CAN-Anschaltung und den Reglern mit CAN-Anschaltung während dieser Zeit nicht erfolgreich abgeschlossen, wird im Fehlerwort w_ERR das Bit 8 gesetzt.

Am Ausgang _CAN_DATA wird eine globale Variable vom Datentyp CAN_INIT_DATA_BMSTRUCT angeschlossen.

Beispiel:

```
_CAN_INIT_DATA_OPT : CAN_INIT_DATA_BMSTRUCT;
```

dabei ist:

`_CAN_INIT_DATA_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_INIT_DATA_BMSTRUCT`	der Datentyp

Diese Variable enthält die Initialisierungsdaten der Regler mit CAN-Anschaltung, der Ωmega mit CAN-Anschaltung und der CAN-IO-Module dieser CAN-Anschaltung und wird bei einigen FBs für die Kommunikation benötigt.

Überprüfen der Kommunikation zu den Reglern mit CAN-Anschaltung:

Ausgänge w_OK_DRIVES, w_ERR1_DRIVES, w_ERR2_DRIVES:

Die Kommunikation zu den Reglern wird durch das Anfordern des Statuswortes geprüft. Wurde das Statuswort empfangen, werden ggf. die Parameternummern der Soll- und Istwerte sowie die Istwert-Zykluszeiten an die Regler übertragen.

Waren alle Übertragungen erfolgreich, wird im OK-Wort w_OK_DRIVES das Bit entsprechend der Gerätenummer des Reglers am CAN-Bus gesetzt. Für den Regler mit der Gerätenummer 0 wird das Bit 0 in w_OK_DRIVES gesetzt, für den Regler mit der Gerätenummer 1 wird das Bit 1 in w_OK_DRIVES gesetzt usw.

Trat ein Fehler auf, wird im Fehlerwort w_ERR1_DRIVES oder w_ERR2_DRIVES das Bit entsprechend der Gerätenummer des Reglers am CAN-Bus gesetzt.

Für den Regler mit der Gerätenummer 0 wird das Bit 0 in w_ERR1_DRIVES oder w_ERR2_DRIVES gesetzt, für den Regler mit der Gerätenummer 1 wird das Bit 1 in w_ERR1_DRIVES oder w_ERR2_DRIVES gesetzt usw.

w_ERR1_DRIVES: Timeout bei der Statuswortanforderung, beim Übertragen der Parameterlisten, der Istwert-Zykluszeiten oder bei der Sollwertfreigabe

w_ERR2_DRIVES: Kommunikationsfehler beim Übertragen der Parameterlisten, der Istwert-Zykluszeiten oder bei der Sollwertfreigabe

Überprüfen der Kommunikation zu den CAN-IO-Modulen:

Ausgänge w_OK_IO, w_ERR_IO:

Bei der Überprüfung der Kommunikation der CAN-IO-Module wird von den initialisierten CAN-DI-16-xx Modulen das Eingangswort angefordert. Wurde das Eingangswort empfangen, wird im OK-Wort w_OK_IO das Bit entsprechend der Gerätenummer des CAN-DI-16-xx Moduls am CAN-Bus gesetzt.

Für das CAN-DI-16-xx Modul mit der Gerätenummer 0 wird das Bit 0 in w_OK_IO gesetzt, für das CAN-DI-16-xx Modul mit der Gerätenummer 1 wird das Bit 1 in w_OK_IO gesetzt usw..

Trat ein Fehler auf, wird im Fehlerwort w_ERR_IO das Bit entsprechend der Gerätenummer des CAN-DI-16-xx Moduls am CAN-Bus gesetzt.

Für das CAN-DI-16-xx Modul mit der Gerätenummer 0 wird das Bit 0 in w_ERR_IO gesetzt, für das CAN-DI-16-xx Modul mit der Gerätenummer 1 wird das Bit 1 in w_ERR_IO gesetzt usw..

An die initialisierten CAN-DO-16-xx Module wird ein Ausgangswort gesendet. Die Bereitstellung des Ausgangswortes auf dem CAN-Bus wird quittiert. Dann wird im OK-Wort w_OK_IO das Bit entsprechend der Gerätenummer des CAN-DO-16-xx Moduls am CAN-Bus gesetzt.

Für das CAN-DO-16-xx Modul mit der Gerätenummer 0 wird das Bit 0 in w_OK_IO gesetzt, für das CAN-DO-16-xx Modul mit der Gerätenummer 1 wird das Bit 1 in w_OK_IO gesetzt usw..

Trat ein Fehler auf, wird im Fehlerwort w_ERR_IO das Bit entsprechend der Gerätenummer des CAN-DO-16-xx Moduls am CAN-Bus gesetzt.

Für das CAN-DO-16-xx Modul mit der Gerätenummer 0 wird das Bit 0 in w_ERR_IO gesetzt, für das CAN-DO-16-xx Modul mit der Gerätenummer 1 wird das Bit 1 in w_ERR_IO gesetzt usw..

Ausgang x_OK:

Der erfolgreiche Abschluß der Initialisierung der CAN-Anschaltung am Ωmega Drive-Line II und der Kommunikation zu den CAN-IO-Modulen und Reglern wird durch x_OK = TRUE signalisiert.

Ausgänge x_ERR, w_ERR:

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und die entsprechenden Fehlerworte ausgegeben. Der Ausgang x_OK bleibt dann FALSE.

Fehlerwort w_ERR:

Bit-Nr.	Fehler
0	Timeout beim Handshake mit der CAN-Anschaltung
1	us_NR_OF_IO_TO_INIT größer 16
2	us_MAX_IO_NR_COMM größer 16
3	us_MAX_DRIVE_NR_COMM größer 16
4	us_MAX_OMEGA_NR_COMM größer 16
5	us_NR_OF_OMEGAS größer 16

Bit-Nr.	Fehler
6	us_BAUDRATE größer 6
7	Anzahl der Teilnehmer am CAN-Bus größer 32
8	Timeout (Initialisierung der CAN-Anschaltung und der Kommunikation zu CAN-IO-Modulen und Reglern)
9	Fehler bei Initialisierungsdaten mindestens eines CAN-IO-Moduls
10	us_BAUDRATE > 4, wenn us_NR_OF_IO_TO_INIT $\neq$ 0 oder us_MAX_IO_NR_COMM $\neq$ 0
11	Initialisierung der CAN-Anschaltung nicht abgeschlossen
12 - 15	Reserviert

Fehlerwort w_ERR_IO:

Bit-Nr.	Bedeutung
0	CAN-IO-Modul mit Gerätenummer 0: Fehler
1	CAN-IO-Modul mit Gerätenummer 1: Fehler
2	CAN-IO-Modul mit Gerätenummer 2: Fehler
3	CAN-IO-Modul mit Gerätenummer 3: Fehler
4	CAN-IO-Modul mit Gerätenummer 4: Fehler
5	CAN-IO-Modul mit Gerätenummer 5: Fehler
6	CAN-IO-Modul mit Gerätenummer 6: Fehler
7	CAN-IO-Modul mit Gerätenummer 7: Fehler
8	CAN-IO-Modul mit Gerätenummer 8: Fehler
9	CAN-IO-Modul mit Gerätenummer 9: Fehler
10	CAN-IO-Modul mit Gerätenummer 10: Fehler
11	CAN-IO-Modul mit Gerätenummer 11: Fehler
12	CAN-IO-Modul mit Gerätenummer 12: Fehler
13	CAN-IO-Modul mit Gerätenummer 13: Fehler
14	CAN-IO-Modul mit Gerätenummer 14: Fehler
15	CAN-IO-Modul mit Gerätenummer 15: Fehler

Fehlerworte w_ERR1_DRIVES und w_ERR2_DRIVES:

Bit-Nr.	Bedeutung
0	Regler mit Gerätenummer 0: Fehler
1	Regler mit Gerätenummer 1: Fehler
2	Regler mit Gerätenummer 2: Fehler
3	Regler mit Gerätenummer 3: Fehler
4	Regler mit Gerätenummer 4: Fehler
5	Regler mit Gerätenummer 5: Fehler
6	Regler mit Gerätenummer 6: Fehler
7	Regler mit Gerätenummer 7: Fehler
8	Regler mit Gerätenummer 8: Fehler
9	Regler mit Gerätenummer 9: Fehler
10	Regler mit Gerätenummer 10: Fehler
11	Regler mit Gerätenummer 11: Fehler
12	Regler mit Gerätenummer 12: Fehler
13	Regler mit Gerätenummer 13: Fehler
14	Regler mit Gerätenummer 14: Fehler

Bit-Nr.	Bedeutung
15	Regler mit Gerätenummer 15: Fehler

OK-Wort w_OK_IO:

Bit-Nr.	Bedeutung
0	CAN-IO-Modul mit Gerätenummer 0: OK
1	CAN-IO-Modul mit Gerätenummer 1: OK
2	CAN-IO-Modul mit Gerätenummer 2: OK
3	CAN-IO-Modul mit Gerätenummer 3: OK
4	CAN-IO-Modul mit Gerätenummer 4: OK
5	CAN-IO-Modul mit Gerätenummer 5: OK
6	CAN-IO-Modul mit Gerätenummer 6: OK
7	CAN-IO-Modul mit Gerätenummer 7: OK
8	CAN-IO-Modul mit Gerätenummer 8: OK
9	CAN-IO-Modul mit Gerätenummer 9: OK
10	CAN-IO-Modul mit Gerätenummer 10: OK
11	CAN-IO-Modul mit Gerätenummer 11: OK
12	CAN-IO-Modul mit Gerätenummer 12: OK
13	CAN-IO-Modul mit Gerätenummer 13: OK
14	CAN-IO-Modul mit Gerätenummer 14: OK
15	CAN-IO-Modul mit Gerätenummer 15: OK

OK-Wort w_OK_DRIVES:

Bit-Nr.	Bedeutung
0	Regler mit Gerätenummer 0: OK
1	Regler mit Gerätenummer 1: OK
2	Regler mit Gerätenummer 2: OK
3	Regler mit Gerätenummer 3: OK
4	Regler mit Gerätenummer 4: OK
5	Regler mit Gerätenummer 5: OK
6	Regler mit Gerätenummer 6: OK
7	Regler mit Gerätenummer 7: OK
8	Regler mit Gerätenummer 8: OK
9	Regler mit Gerätenummer 9: OK
10	Regler mit Gerätenummer 10: OK
11	Regler mit Gerätenummer 11: OK
12	Regler mit Gerätenummer 12: OK
13	Regler mit Gerätenummer 13: OK
14	Regler mit Gerätenummer 14: OK
15	Regler mit Gerätenummer 15: OK

5.7 CAN_INIT_FB1

Beschreibung

Dieser Funktionsbaustein für CAN wird vom FB CAN_INIT benutzt.

5.8 CAN_INIT_FB2

Beschreibung

Dieser Funktionsbaustein für CAN wird vom FB CAN_INIT benutzt.

5.9 CAN_OBJ_READ

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um ein Objekt eines Reglers mit CAN-Anschaltung zu lesen.

HINWEIS

Der FB CAN_OBJ_READ verwendet die Bibliothek BM_TYPES_20bd00 oder höher.

Dieser FB wird zusammen mit dem FB CAN_SD_CONTROL eingesetzt.

Parameter Eingang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
us_DRIVE_NR	USINT 0 bis 15	Gerätenummer des Reglers am CAN-Bus
w_OBJ_NR	WORD	Objekt-Nummer
us_OBJ_FORMAT	USINT 1, 2	Objekt-Format
us_OBJ_SUBINDEX	USINT	Objekt-Subindex
x_EN	BOOL	Freigabe
x_RESET	BOOL	Reset

Parameter Ausgang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
ud_OBJ_VALUE	UDINT	gelesener Objekt-Wert
x_BUSY	BOOL	Kommunikation ist aktiv
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit
x_OK	BOOL	OK-Bit

Der FB CAN_OBJ_READ übergibt mit der Objekt-Nummer (w_OBJ_NR), dem Objekt-Format (us_OBJ_FORMAT) und dem Objekt-Subindex (us_OBJ_SUBINDEX) einen Objekt-Lesen-Auftrag an den Regler mit der Gerätenummer us_DRIVE_NR. Der Regler bearbeitet den Objekt-Lesen-Auftrag und gibt das Ergebnis der Kommunikation zurück. Der Inhalt des Objekts wird am Ausgang ud_OBJ_VALUE ausgegeben.

Ein-/Ausgang _BASE:

An _BASE muß eine globale Variable vom Datentyp CAN_CTRL_BMSTRUCT angeschlossen werden.

Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden und ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Eingang _BASE_CTRL angeschlossen worden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;
```

dabei ist:

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Eingang us_DRIVE_NR:

Am Eingang us_DRIVE_NR wird die Gerätenummer des Reglers am CAN-Bus angegeben, an den der Objekt-Lesen-Auftrag übergeben wird.

Eingang u_OBJ_NR:

Die Objekt-Nummer für den Objekt-Lesen-Auftrag wird am Eingang u_OBJ_NR angegeben.

Eingang us_OBJ_FORMAT:

Am Eingang us_OBJ_FORMAT wird das Format des zu lesenden Objekt-Werts angegeben. us_OBJ_FORMAT = 1 bedeutet Format Wort, us_OBJ_FORMAT = 2 bedeutet Format Doppelwort. Voreinstellung ist us_OBJ_FORMAT = 1, d. h. Format Wort.

Eingang us_OBJ_SUBINDEX:

Der Objekt-Subindex wird am Eingang us_OBJ_SUBINDEX eingestellt. Voreinstellung ist us_OBJ_SUBINDEX = 0, d. h. lesen des gesamten Objekts.

Eingang x_EN:

Die Kommunikation wird mit x_EN = TRUE gestartet. Wird x_EN auf FALSE gesetzt, bevor x_BUSY = FALSE ist, wird von einem bewußten Abbruch der Kommunikation ausgegangen. Der FB CAN_OBJ_READ muß dann mit x_RESET = TRUE zurückgesetzt werden.

Eingang x_RESET:

Mit x_RESET = TRUE kann der FB zurückgesetzt werden.

Ausgang ud_OBJ_VALUE:

Der gelesene Objekt-Wert wird am Ausgang ud_OBJ_VALUE ausgegeben.

Ausgang x_BUSY:

Der Ausgang x_BUSY zeigt mit TRUE an, daß die Kommunikation aktiv ist.

Ausgang x_OK:

Der Ausgang x_OK wird auf TRUE gesetzt, wenn die Kommunikation erfolgreich abgeschlossen ist.

Ausgänge x_ERR, b_ERR

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Ungültige Gerätenummer (us_DRIVE_NR größer 15)
1	Fehler in der Kommunikation (Objekt lesen)
2	Ungültiges Format (us_OBJ_FORMAT größer 2)
3 - 7	Reserviert

5.10 CAN_OBJ_WRITE

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um auf ein Objekt des Reglers zu schreiben.

HINWEIS

Der FB CAN_OBJ_WRITE verwendet die Bibliothek BM_TYPES_20bd00 oder höher.

Dieser FB wird zusammen mit dem FB CAN_SD_CONTROL eingesetzt.

Parameter Eingang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
us_DRIVE_NR	USINT 0 bis 15	Gerätenummer des Reglers am CAN-Bus
w_OBJ_NR	WORD	Objekt-Nummer
us_OBJ_FORMAT	USINT 1, 2	Objekt-Format
us_OBJ_SUBINDEX	USINT	Objekt-Subindex
ud_OBJ_VALUE	UDINT	zu sendender Objekt-Wert
x_EN	BOOL	Freigabe
x_RESET	BOOL	Reset

Parameter Ausgang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
x_BUSY	BOOL	Kommunikation ist aktiv
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit
x_OK	BOOL	OK-Bit

Der FB CAN_OBJ_WRITE übergibt mit der Objekt-Nummer (w_OBJ_NR), dem Objekt-Format (us_OBJ_FORMAT), dem Objekt-Subindex (us_OBJ_SUBINDEX) und dem Objekt-Wert (ud_OBJ_VALUE) einen Objekt-Schreiben-Auftrag an den Regler mit der Gerätenummer us_DRIVE_NR. Der Regler bearbeitet den Objekt-Schreiben-Auftrag und gibt das Ergebnis der Kommunikation zurück.

Ein-/Ausgang _BASE:

An _BASE muß eine globale Variable vom Datentyp CAN_CTRL_BMSTRUCT angeschlossen werden.

Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden und ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Eingang _BASE_CTRL angeschlossen worden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;
```

dabei ist:

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Eingang us_DRIVE_NR:

Am Eingang us_DRIVE_NR wird die Gerätenummer des Reglers am CAN-Bus angegeben, an den der Objekt-Schreiben-Auftrag übergeben wird.

Eingang w_OBJ_NR:

Die Objekt-Nummer für den Objekt-Schreiben-Auftrag wird am Eingang w_OBJ_NR angegeben.

Eingang us_OBJ_FORMAT:

Am Eingang us_OBJ_FORMAT wird das Format des zu sendenden Objekt-Werts angegeben. us_OBJ_FORMAT = 1 bedeutet Format Wort, us_OBJ_FORMAT = 2 bedeutet Format Doppelwort. Voreinstellung ist us_OBJ_FORMAT = 1, d. h. Format Wort.

Eingang us_OBJ_SUBINDEX:

Der Objekt-Subindex wird am Eingang us_OBJ_SUBINDEX eingestellt. Voreinstellung ist us_OBJ_SUBINDEX = 0, d. h. schreiben des gesamten Objekts.

Eingang ud_OBJ_VALUE:

Der zu schreibende Objekt-Wert wird am Eingang ud_OBJ_VALUE angeschlossen.

Eingang x_EN:

Die Kommunikation wird mit x_EN = TRUE gestartet. Wird x_EN auf FALSE gesetzt, bevor x_BUSY = FALSE ist, wird von einem bewußten Abbruch der Kommunikation ausgegangen. Der FB CAN_OBJ_WRITE muß dann mit x_RESET = TRUE zurückgesetzt werden.

Eingang x_RESET:

Mit x_RESET = TRUE kann der FB zurückgesetzt werden.

Ausgang x_BUSY:

Der Ausgang x_BUSY zeigt mit TRUE an, daß die Kommunikation aktiv ist.

Ausgang x_OK:

Der Ausgang x_OK wird auf TRUE gesetzt, wenn die Kommunikation erfolgreich abgeschlossen ist.

Ausgänge x_ERR, b_ERR:

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Ungültige Gerätenummer (us_DRIVE_NR größer 15)
1	Fehler in der Kommunikation (Objekt schreiben)
2	Ungültiges Format (> 2)
3 - 7	Reserviert

5.11 CAN_PA_LIST_READ

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um die aktuelle Sollwert-Parameternummern-Liste (PA-Daten-Liste) für die zyklische Kommunikation im Regler zu lesen.

Die Sollwert-Parameternummern werden in der Regel am FB CAN_DRIVE_INIT angegeben und mittels FB CAN_INIT an den Regler gesendet.

HINWEIS

Der FB CAN_PA_LIST_READ verwendet die Bibliothek BM_TYPES_20bd00 oder höher.

Dieser FB wird zusammen mit dem FB CAN_SD_CONTROL eingesetzt.

Parameter Eingang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
us_DRIVE_NR	USINT 0 bis 15	Gerätenummer des Reglers am CAN-Bus
x_EN	BOOL	Freigabe
x_RESET	BOOL	Reset

Parameter Ausgang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
u_WR_PAR_NR0	UINT	Sollwert-Parameternummer 0
u_WR_PAR_NR1	UINT	Sollwert-Parameternummer 1
u_WR_PAR_NR2	UINT	Sollwert-Parameternummer 2
u_WR_PAR_NR3	UINT	Sollwert-Parameternummer 3
x_BUSY	BOOL	Kommunikation ist aktiv
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit
x_OK	BOOL	OK-Bit

Der FB CAN_PA_LIST_READ übergibt einen PA-Daten-Lesen-Auftrag an den Regler mit der Gerätenummer "us_DRIVE_NR". Der Regler bearbeitet den PA-Daten-Lesen-Auftrag und gibt das Ergebnis der Kommunikation zurück. Der Inhalt der PA-Daten-Liste wird an den Ausgängen u_WR_PAR_NR0 bis u_WR_PAR_NR3 ausgegeben.

Ein-/Ausgang _BASE:

An _BASE muß eine globale Variable vom Datentyp CAN_CTRL_BMSTRUCT angeschlossen werden.

Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden und ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Eingang _BASE_CTRL angeschlossen worden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;
```

dabei ist:

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Eingang us_DRIVE_NR:

Am Eingang us_DRIVE_NR wird die Gerätenummer des Reglers am CAN-Bus angegeben, an den der PA-Daten-Lesen-Auftrag übergeben wird.

Eingang x_EN:

Die Kommunikation wird mit x_EN = TRUE gestartet. Wird x_EN auf FALSE gesetzt, bevor x_BUSY = FALSE ist, wird von einem bewußten Abbruch der Kommunikation ausgegangen. Der FB CAN_PA_LIST_READ muß dann mit x_RESET = TRUE zurückgesetzt werden.

Eingang x_RESET:

Mit dem Eingang x_RESET = TRUE kann der FB zurückgesetzt werden.

Ausgänge u_WR_PAR_NR0 bis u_WR_PAR_NR3:

Der Inhalt der PA-Daten-Liste des Reglers, d.h. die Sollwert-Parameternummern 0 bis 3, werden an den Ausgängen u_WR_PAR_NR0 bis u_WR_PAR_NR3 ausgegeben.

Ausgang x_BUSY:

Der Ausgang x_BUSY zeigt mit TRUE an, daß die Kommunikation aktiv ist.

Ausgang x_OK:

Der Ausgang x_OK wird auf TRUE gesetzt, wenn die Kommunikation erfolgreich abgeschlossen ist.

Ausgänge x_ERR, b_ERR:

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Ungültige Gerätenummer (us_DRIVE_NR größer 15)
1	Fehler in der Kommunikation (PA-Daten lesen)
2 - 7	Reserviert

5.12 CAN_PA_LIST_WRITE

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um eine neue Sollwert-Parameternummern-Liste (PA-Daten-Liste) für die zyklische Kommunikation im Regler einzustellen.

Die Sollwert-Parameternummern, Sollwert-Formate und die Sollwert-Zykluszeiten werden in der Regel am FB CAN_DRIVE_INIT angegeben und mittels FB CAN_INIT an den Regler gesendet.

Der FB CAN_PA_LIST_WRITE ermöglicht eine nachträgliche Änderung der Sollwert-Parameternummern und -Formate im Regler unter Beibehaltung der Sollwert-Zykluszeiten im Ωmega Drive-Line II.

HINWEIS

Zur Freigabe der neuen Sollwert-Parameternummern-Liste muß das Objekt 16#6002 gesendet werden. Siehe dazu bitte auch die Beschreibung "Optionskarte CAN-Interface".

Der FB CAN_PA_LIST_WRITE verwendet die Bibliothek BM_TYPES_20bd00 oder höher.

Dieser FB wird zusammen mit dem FB CAN_SD_CONTROL eingesetzt.

Parameter Eingang	**Datentyp**	**Beschreibung**
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
_CAN_DATA	CAN_INIT_DATA_BMSTRUCT	Initialisierungsdaten des CAN
us_DRIVE_NR	USINT 0 bis 15	Gerätenummer des Reglers am CAN-Bus
u_WR_PAR_NR0	UINT	Sollwert-Parameternummer 0
us_WR_FORMAT0	USINT 0, 1	Sollwert-Format 0
u_WR_PAR_NR1	UINT	Sollwert-Parameternummer 1
us_WR_FORMAT1	USINT 1, 2	Sollwert-Format 1
u_WR_PAR_NR2	UINT	Sollwert-Parameternummer 2
us_WR_FORMAT2	USINT 1, 2	Sollwert-Format 2
u_WR_PAR_NR3	UINT	Sollwert-Parameternummer 3
us_WR_FORMAT3	USINT 1, 2	Sollwert-Format 3
x_EN	BOOL	Freigabe
x_RESET	BOOL	Reset
x_BUSY	BOOL	Kommunikation ist aktiv

Parameter Ausgang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
_CAN_DATA	CAN_INIT_DATA_BMSTRUCT	Initialisierungsdaten des CAN
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit
x_OK	BOOL	OK-Bit

Der FB CAN_PA_LIST_WRITE übergibt mit den Werten der Eingänge u_WR_PAR_NR0 bis u_WR_PAR_NR3 und us_WR_FORMAT0 bis us_WR_FORMAT3 einen PA-Daten-Schreiben-Auftrag an den Regler mit der Gerätenummer us_DRIVE_NR. Der Regler bearbeitet den PA-Daten-Schreiben-Auftrag und gibt das Ergebnis der Kommunikation zurück.

Die geänderten Sollwert-Parameternummern und Sollwert-Formate werden in die Variable, die die Initialisierungsdaten der Regler und der CAN-IO-Module enthält, eingetragen (siehe Ein-/Ausgang _CAN_DATA). Diese Variable wird bei FB CAN_PD_COMM für die Prozeßdatenkommunikation und bei FB CAN_SD_CONTROL für die Bedarfsdatenkommunikations-Überwachung benötigt.

Zur Freigabe der neuen Sollwert-Parameternummern-Liste muß anschließend mit dem FB CAN_OBJ_WRITE das Objekt 16#6002 gesendet werden. Siehe Beschreibung "Optionskarte CAN-Interface".

Ein-/Ausgang _BASE:

An _BASE muß eine globale Variable vom Datentyp CAN_CTRL_BMSTRUCT angeschlossen werden. Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden und ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Eingang _BASE_CTRL angeschlossen worden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;
```

dabei ist:

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Ein-/Ausgang _CAN_DATA:

An _CAN_DATA wird eine globale Variable vom Datentyp CAN_INIT_DATA_BMSTRUCT angeschlossen.

Diese Variable ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Ausgang _CAN_DATA angeschlossen worden.

Beispiel:

```
_CAN_INIT_DATA_OPT: CAN_INIT_DATA_BMSTRUCT;
```

dabei ist:

`_CAN_INIT_DATA_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_INIT_DATA_BMSTRUCT`	der Datentyp

Diese Variable enthält die Initialisierungsdaten der Regler und der CAN-IO-Module dieser CAN-Anschaltung und wird bei einigen FBs für die Kommunikation benötigt.

Eingang us_DRIVE_NR:

Am Eingang us_DRIVE_NR wird die Gerätenummer des Reglers am CAN-Bus angegeben, an den der PA-Daten-Schreiben-Auftrag übergeben wird.

Eingänge u_WR_PAR_NR0 bis u_WR_PAR_NR3:

Die Sollwert-Parameternummern werden an den Eingängen u_WR_PAR_NR0 bis u_WR_PAR_NR3 angegeben. Bei nicht belegten Eingängen wird als Sollwert-Parameternummer die Parameternummer 0 an den Regler gesendet.

Eingänge us_WR_FORMAT0 bis us_WR_FORMAT3:

Die Formate der Sollwerte werden an den Eingängen us_WR_FORMAT0 bis us_WR_FORMAT3 angegeben.

HINWEIS

Der Sollwert 0 darf das Format Byte oder Wort haben. Die Sollwerte 1, 2 und 3 dürfen das Format Wort oder Doppelwort haben.

Formatangabe:	us_WR_FORMATx = 0:	Format Byte	(Sollwert 0)
	us_WR_FORMATx = 1:	Format Wort	(Sollwert 0, 1, 2 und 3)
	us_WR_FORMATx = 2:	Format Doppelwort	(Sollwert 1, 2 und 3)
	x = 0, 1, 2, 3		

Die bei der Initialisierung am FB CAN_DRIVE_INIT angegebenen Sollwert-Zykluszeiten werden beibehalten.

Eingang x_EN:

Die Kommunikation wird mit x_EN = TRUE gestartet. Wird x_EN auf FALSE gesetzt, bevor x_BUSY = FALSE ist, wird von einem bewußten Abbruch der Kommunikation ausgegangen. Der FB CAN_PA_LIST_WRITE muß dann mit x_RESET = TRUE zurückgesetzt werden.

Eingang x_RESET:

Mit x_RESET = TRUE kann der FB zurückgesetzt werden.

Ausgang x_BUSY:

Der Ausgang x_BUSY zeigt mit TRUE an, daß die Kommunikation aktiv ist.

Ausgang x_OK:

Der Ausgang x_OK wird auf TRUE gesetzt, wenn die Kommunikation erfolgreich abgeschlossen ist.

Ausgänge x_ERR, b_ERR:

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Ungültige Gerätenummer (us_DRIVE_NR größer 15)
1	Fehler in der Kommunikation (PA-Daten schreiben)
2 - 7	Reserviert

5.13 CAN_PAR_READ

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um einen Parameter eines Reglers mit CAN-Anschaltung zu lesen.

HINWEIS

Der FB CAN_PAR_READ verwendet die Bibliothek BM_TYPES_20bd00 oder höher.

Dieser FB wird zusammen mit dem FB CAN_SD_CONTROL eingesetzt.

Parameter Eingang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
us_DRIVE_NR	USINT 0 bis 15	Gerätenummer des Reglers am CAN-Bus
u_PAR_NR	UINT	Parameter-Nummer
us_PAR_FORMAT	USINT 1, 2	Parameter-Format
us_PAR_ELEMENT	USINT 7	Parameter-Element
x_EN	BOOL	Freigabe
x_RESET	BOOL	Reset

Parameter Eingang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
ud_PAR_VALUE	UDINT	gelesener Parameter-Wert
x_BUSY	BOOL	Kommunikation ist aktiv
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit
x_OK	BOOL	OK-Bit

Der FB CAN_PAR_READ übergibt mit den Werten der Eingänge u_PAR_NR, us_PAR_FORMAT und us_PAR_ELEMENT einen Parameter-Lesen-Auftrag an den Regler mit der Gerätenummer "us_DRIVE_NR". Der Regler bearbeitet den Parameter-Lesen-Auftrag und gibt das Ergebnis der Kommunikation zurück. Der Inhalt des Parameter-Elements wird am Ausgang ud_PAR_VALUE ausgegeben.

Ein-/Ausgang _BASE:

An _BASE muß eine globale Variable vom Datentyp CAN_CTRL_BMSTRUCT angeschlossen werden.

Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden und ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Eingang _BASE_CTRL angeschlossen worden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;
```

dabei ist:

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Eingang us_DRIVE_NR:

Am Eingang us_DRIVE_NR wird die Gerätenummer des Reglers am CAN-Bus angegeben.

Eingang u_PAR_NR:

Die Parameter-Nummer für den Parameter-Lesen-Auftrag wird am Eingang u_PAR_NR angegeben.

Eingang us_PAR_FORMAT:

Am Eingang us_PAR_FORMAT wird das Format des zu lesenden Parameter-Werts angegeben. us_PAR_FORMAT = 1 bedeutet Format Wort, us_PAR_FORMAT = 2 bedeutet Format Doppelwort. Voreinstellung ist us_PAR_FORMAT = 1, d. h. Format Wort.

Eingang us_PAR_ELEMENT:

Das Parameter-Element wird am Eingang us_PAR_ELEMENT eingestellt. Voreinstellung ist us_PAR_ELEMENT = 7, d. h. lesen des Parameter-Werts.

Eingang x_EN:

Die Kommunikation wird mit x_EN = TRUE gestartet. Wird x_EN auf FALSE gesetzt, bevor x_BUSY = FALSE ist, wird von einem bewußten Abbruch der Kommunikation ausgegangen. Der FB CAN_OBJ_READ muß dann mit x_RESET = TRUE zurückgesetzt werden.

Eingang x_RESET:

Mit x_RESET = TRUE kann der FB zurückgesetzt werden.

Ausgang ud_PAR_VALUE:

Das gelesene Parameter-Element wird am Ausgang ud_PAR_VALUE zu Verfügung gestellt.

Ausgang x_BUSY:

Der Ausgang x_BUSY zeigt mit TRUE an, daß die Kommunikation aktiv ist.

Ausgang x_OK:

Der Ausgang x_OK wird auf TRUE gesetzt, wenn die Kommunikation erfolgreich abgeschlossen ist.

Ausgänge x_ERR, b_ERR

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Ungültige Gerätenummer (us_DRIVE_NR größer 15)
1	Fehler in der Kommunikation (Parameter lesen)
2	Ungültiges Format (> 2)
3 - 7	Reserviert

5.14 CAN_PAR_WRITE

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um einen Parameter des Reglers zu schreiben.

HINWEIS

Der FB CAN_PAR_WRITE verwendet die Bibliothek BM_TYPES_20bd00 oder höher.

Dieser FB wird zusammen mit dem FB CAN_SD_CONTROL eingesetzt.

Parameter Eingang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
us_DRIVE_NR	USINT 0 bis 15	Gerätenummer des Reglers am CAN-Bus
u_PAR_NR	UINT	Parameter-Nummer
us_PAR_FORMAT	USINT 1, 2	Parameter-Format
us_PAR_ELEMENT	USINT 7	Parameter-Element
ud_PAR_VALUE	UDINT	zu sendender Parameter-Wert
x_EN	BOOL	Freigabe
x_RESET	BOOL	Reset
x_BUSY	BOOL	Kommunikation ist aktiv

Parameter Ausgang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit
x_OK	BOOL	OK-Bit

Der FB CAN_PAR_WRITE übergibt mit den Werten der Eingänge u_PAR_NR, us_PAR_FORMAT, us_PAR_ELEMENT und ud_PAR_VALUE einen Parameter-Schreiben-Auftrag an den Regler mit der Gerätenummer "us_DRIVE_NR". Der Regler bearbeitet den Parameter-Schreiben-Auftrag und gibt das Ergebnis der Kommunikation zurück.

Ein-/Ausgang _BASE:

An _BASE muß eine globale Variable vom Datentyp CAN_CTRL_BMSTRUCT angeschlossen werden.

Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden und ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Eingang _BASE_CTRL angeschlossen worden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;
```

dabei ist:

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Eingang us_DRIVE_NR:

Am Eingang us_DRIVE_NR wird die Gerätenummer des Reglers am CAN-Bus angegeben.

Eingang u_PAR_NR:

Die Parameter-Nummer für den Parameter-Schreiben-Auftrag wird am Eingang u_PAR_NR angegeben.

Eingang us_PAR_FORMAT:

Am Eingang us_PAR_FORMAT wird das Format des zu sendenden Parameter-Werts angegeben. us_PAR_FORMAT = 1 bedeutet Format Wort, us_PAR_FORMAT = 2 bedeutet Format Doppelwort. Voreinstellung ist us_PAR_FORMAT = 1, d. h. Format Wort.

Eingang us_PAR_ELEMENT:

Das Parameter-Element (z.B. us_PAR_ELEMENT = 7 - Parameter-Daten) wird am Eingang us_PAR_ELEMENT eingestellt. Voreinstellung ist us_PAR_ELEMENT = 7, d. h. Parameter-Daten.

Eingang ud_PAR_VALUE:

Der zu übertragende/schreibende Wert wird am Eingang ud_PAR_VALUE angeschlossen.

Eingang x_EN:

Die Kommunikation wird mit x_EN = TRUE gestartet. Wird x_EN auf FALSE gesetzt, bevor x_BUSY = FALSE ist, wird von einem bewußten Abbruch der Kommunikation ausgegangen. Der FB CAN_PAR_WRITE muß dann mit x_RESET = TRUE zurückgesetzt werden.

Eingang x_RESET:

Mit x_RESET = TRUE kann der FB zurückgesetzt werden.

Ausgang x_BUSY:

Der Ausgang x_BUSY zeigt mit TRUE an, daß der FB aktiv ist.

Ausgang x_OK:

Der Ausgang x_OK wird auf TRUE gesetzt, wenn die Kommunikation erfolgreich abgeschlossen ist.

Ausgänge x_ERR, b_ERR

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Ungültige Gerätenummer (us_DRIVE_NR größer 15)
1	Fehler in der Kommunikation (Parameter schreiben)
2	ungültiges Format (> 2)
3 - 7	Reserviert

HINWEIS

Bit 1 wird gesetzt, wenn ein Fehler in der Kommunikation auftrat. Dies kann z. B. durch eine Wertebereichsverletzung erfolgen. Einige Wortparameter dürfen z. B. nicht mit Werten > 16#3FFF beschrieben werden, andere Parameter dürfen nicht 0 sein. Siehe hierzu bitte auch die Regler- bzw. Parameterbeschreibungen.

5.15 CAN_PD_COMM

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um die Prozeßdatenkommunikation über CAN zwischen Ωmega und Regler durchzuführen. Steuerwort und Sollwerte werden an den Regler gesendet, Statuswort und Istwerte werden vom Regler empfangen und ausgegeben.

Die Soll- und Istwert-Parameternummern werden in der Initialisierung am FB CAN_DRIVE_INIT eingestellt und mittels FB CAN_INIT und an den Regler gesendet.

HINWEIS

Der FB CAN_PD_COMM verwendet die Bibliothek BM_TYPES_20bd00 oder höher.

Parameter Eingang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
_CAN_DATA	CAN_INIT_DATA_BMSTRUCT	Initialisierungsdaten des CAN
us_DRIVE_NR	USINT 0 bis 15	Gerätenummer des Reglers am CAN-Bus
w_CONTROLWORD	WORD	Steuerwort
x_ACYCLIC	BOOL	zyklisch / azyklisch senden und empfangen
ud_WR_VALUE0	UDINT	Sollwert 0
ud_WR_VALUE1	UDINT	Sollwert 1
ud_WR_VALUE2	UDINT	Sollwert 2
ud_WR_VALUE3	UDINT	Sollwert 3
t_TIME	TIME > 0, bis 30 s	Überwachungszeit
x_EN	BOOL	Freigabe
x_RESET	BOOL	Reset

Parameter Ausgang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
_CAN_DATA	CAN_INIT_DATA_BMSTRUCT	Initialisierungsdaten des CAN
w_STATUSWORD	WORD	Statuswort
ud_RD_VALUE0	UDINT	Istwert 0
ud_RD_VALUE1	UDINT	Istwert 1
ud_RD_VALUE2	UDINT	Istwert 2
ud_RD_VALUE3	UDINT	Istwert 3
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit
x_OK	BOOL	OK-Bit

Der FB CAN_PD_COMM sendet das Steuerwort und die Sollwerte an den Regler und empfängt das Statuswort und die Istwerte vom Regler.

Die Parameternummern, die Formate und ggf. die Zykluszeiten der Sollwerte und Istwerte werden bei der Initialisierung der CAN-Anschaltung und der Kommunikation mit den Reglern angegeben (Näheres hierzu entnehmen Sie bitte den Beschreibungen der FBs CAN_DRIVE_INIT und CAN_INIT.)

Ein-/Ausgang _BASE:

An _BASE muß eine globale Variable vom Datentyp CAN_CTRL_BMSTRUCT angeschlossen werden.

Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden und ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Eingang _BASE_CTRL angeschlossen worden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;
```

dabei ist:

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Ein-/Ausgang _CAN_DATA:

An _CAN_DATA wird eine globale Variable vom Datentyp CAN_INIT_DATA_BMSTRUCT angeschlossen.

Diese Variable ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Ausgang _CAN_DATA angeschlossen worden.

Beispiel:

```
_CAN_INIT_DATA_OPT : CAN_INIT_DATA_BMSTRUCT;
```

dabei ist:

`_CAN_INIT_DATA_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_INIT_DATA_BMSTRUCT`	der Datentyp

Diese Variable enthält die Initialisierungsdaten der Regler und der CAN-IO-Module dieser CAN-Anschaltung und wird bei einigen FBs für die Kommunikation benötigt.

Eingang us_DRIVE_NR:

Am Eingang us_DRIVE_NR wird die Gerätenummer des Reglers am CAN-Bus angegeben.

Eingang w_CONTROLWORD:

Das an den Regler zu sendende Steuerwort wird am Eingang w_CONTROLWORD angeschlossen.

Eingang x_ACYCLIC:

Mit dem Eingang x_ACYCLIC wird festgelegt, ob die Soll- und Istwerte zyklisch oder azyklisch übertragen werden. Ist x_ACYCLIC = FALSE, werden die Soll- und Istwerte zyklisch mit ihren (im FB CAN_DRIVE_INIT festgelegten) Zykluszeiten übertragen. Ist x_ACYCLIC = TRUE, werden die Soll- und Istwerte azyklisch übertragen. Hierzu werden in die jeweiligen Steuerregister der Soll- und Istwerte eine Sendeaufforderung (für die Sollwerte) oder eine Empfangsanforderung (für die Istwerte) eingetragen. Voreinstellung ist x_ACYCLIC = FALSE (Soll- und Istwerte werden zyklisch mit ihren (im FB CAN_DRIVE_INIT angegebenen) Zykluszeiten übertragen).

Eingänge ud_WR_VALUE0 bis ud_WR_VALUE3:

Die Sollwerte werden an den Eingängen ud_WR_VALUE0 bis ud_WR_VALUE3 angeschlossen. Die Reihenfolge der Sollwerte entspricht der am FB CAN_DRIVE_INIT angegebenen Reihenfolge der Sollwert-Parameternummern.

HINWEIS

Sind im FB CAN_DRIVE_INIT Zykluszeiten für die Soll- und Istwerte angegeben (oder sind Istwert-Zykluszeiten im Regler gespeichert) und wird im FB CAN_PD_COMM die azyklische Übertragung gewählt, erhöht sich die Buslast, da zusätzliche Nachrichten auf dem CAN-Bus gesendet und empfangen werden.

Eingang x_EN:

Mit dem Eingang x_EN erfolgt die Freigabe des FB CAN_PD_COMM für die Prozeßdatenkommunikation. Dazu wird x_EN auf TRUE gesetzt. Wird x_EN auf FALSE gesetzt, bevor x_OK = TRUE ist, wird von einem bewußten Abbruch der Kommunikation ausgegangen. Der FB sollte dann mit x_RESET = TRUE zurückgesetzt werden.

Eingang t_TIME:

Die Überwachungszeit ist am Eingang t_TIME eingestellt. Wird der Eingang t_TIME nicht belegt, ergibt sich eine Voreinstellung von 3 s.

Eingang x_RESET:

Mit x_RESET = TRUE kann der FB zurückgesetzt werden.

Ausgang w_STATUSWORD:

Am Ausgang w_STATUSWORD wird das vom Regler empfangene Statuswort ausgegeben.

Ausgänge ud_RD_VALUE0 bis ud_RD_VALUE3:

An den Ausgängen ud_RD_VALUE0 bis ud_RD_VALUE3 stehen die empfangenen Istwerte zur Verfügung. Die Reihenfolge der Sollwerte entspricht der am FB CAN_DRIVE_INIT angegebenen Reihenfolge der Istwert-Parameternummern.

Ausgang x_OK:

Der Ausgang x_OK gibt mit TRUE an, daß das Statuswort empfangen wurde.

Ausgänge x_ERR, b_ERR:

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Ungültige Gerätenummer (us_DRIVE_NR größer 15)
1	Mit dieser us_DRIVE_NR ist kein Regler initialisiert
2	Timeout
3 - 7	Reserviert

5.16 CAN_PE_LIST_READ

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um die aktuelle Istwert-Parameternummern-Liste (PE-Daten-Liste) für die zyklische Kommunikation im Regler zu lesen.

Die Istwert-Parameternummern werden in der Regel am FB CAN_DRIVE_INIT eingestellt und mittels FB CAN_INIT an den Regler gesendet.

HINWEIS

Der FB CAN_PE_LIST_READ verwendet die Bibliothek BM_TYPES_20bd00 oder höher.

Dieser FB wird zusammen mit dem FB CAN_SD_CONTROL eingesetzt.

Parameter Eingang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
us_DRIVE_NR	USINT 0 bis 15	Gerätenummer des Reglers am CAN-Bus
x_EN	BOOL	Freigabe
x_RESET	BOOL	Reset

Parameter Ausgang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
u_RD_PAR_NR0	UINT	Istwert-Parameternummer 0
u_RD_PAR_NR1	UINT	Istwert-Parameternummer 1
u_RD_PAR_NR2	UINT	Istwert-Parameternummer 2
u_RD_PAR_NR3	UINT	Istwert-Parameternummer 3
x_BUSY	BOOL	Kommunikation ist aktiv
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit
x_OK	BOOL	OK-Bit

Der FB CAN_PE_LIST_READ übergibt einen PE-Daten-Lesen-Auftrag an den Regler mit der Gerätenummer us_DRIVE_NR. Der Regler bearbeitet den PE-Daten-Lesen-Auftrag und gibt das Ergebnis der Kommunikation zurück. Der Inhalt des PE-Daten-Liste wird an den Ausgängen u_RD_PAR_NR0 bis u_RD_PAR_NR3 ausgegeben.

Ein-/Ausgang _BASE:

An _BASE muß eine globale Variable vom Datentyp CAN_CTRL_BMSTRUCT angeschlossen werden.

Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden und ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Eingang _BASE_CTRL angeschlossen worden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;
```

dabei ist:

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Eingang us_DRIVE_NR:

Am Eingang us_DRIVE_NR wird die Gerätenummer des Reglers am CAN-Bus angegeben.

Eingang x_EN:

Die Kommunikation wird mit x_EN = TRUE gestartet. Wird x_EN auf FALSE gesetzt, bevor x_BUSY = FALSE ist, wird von einem bewußten Abbruch der Kommunikation ausgegangen. Der FB CAN_PE_LIST_READ muß dann mit x_RESET = TRUE zurückgesetzt werden.

Eingang x_RESET:

Mit x_RESET = TRUE kann der FB zurückgesetzt werden.

Ausgänge u_RD_PAR_NR0 bis u_RD_PAR_NR3:

Der Inhalt der PE-Daten-Liste des Reglers, d.h. die Istwert-Parameternummern 0 bis 3, werden an den Ausgängen u_RD_PAR_NR0 bis u_RD_PAR_NR3 ausgegeben.

Ausgang x_BUSY:

Der Ausgang x_BUSY zeigt mit TRUE an, daß die Kommunikation aktiv ist.

Ausgang x_OK:

Der Ausgang x_OK wird auf TRUE gesetzt, wenn die Kommunikation erfolgreich abgeschlossen ist.

Ausgänge x_ERR, b_ERR

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Ungültige Gerätenummer (us_DRIVE_NR größer 15)
1	Fehler in der Kommunikation (PE-Daten lesen)
2 - 7	Reserviert

5.17 CAN_PE_LIST_WRITE

Beschreibung

Diesen Funktionsbaustein für CAN können Sie verwenden, um eine neue Istwert-Parameternummern-Liste (PE-Daten-Liste) für die zyklische Kommunikation im Regler einzustellen.

Die Istwert-Parameternummern, Istwert-Formate und die Istwert-Zykluszeiten werden in der Regel am FB CAN_DRIVE_INIT eingestellt und mittels FB CAN_INIT an den Regler gesendet.

Der FB CAN_PE_LIST_WRITE ermöglicht eine nachträgliche Änderung der Istwert-Parameternummern und -Formate unter Beibehaltung der Istwert-Zykluszeiten im Regler.

HINWEIS

Der FB PE_LIST_WRITE verwendet die Bibliothek BM_TYPES_20bd00 oder höher.

Dieser FB wird zusammen mit dem FB CAN_SD_CONTROL eingesetzt.

Parameter Eingang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
_CAN_DATA	CAN_INIT_DATA_BMSTRUCT	Initialisierungsdaten des CAN
us_DRIVE_NR	USINT 0 bis 15	Gerätenummer des Reglers am CAN-Bus
u_RD_PAR_NR0	UINT	Istwert-Parameternummer 0
us_RD_FORMAT0	USINT 0, 1	Istwert-Format 0
u_RD_PAR_NR1	UINT	Istwert-Parameternummer 1
us_RD_FORMAT1	USINT 1, 2	Istwert-Format 1
u_RD_PAR_NR2	UINT	Istwert-Parameternummer 2
us_RD_FORMAT2	USINT 1, 2	Istwert-Format 2
u_RD_PAR_NR3	UINT	Istwert-Parameternummer 3
us_RD_FORMAT3	USINT 1, 2	Istwert-Format 3
x_EN	BOOL	Freigabe
x_RESET	BOOL	Reset

Parameter Ausgang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
_CAN_DATA	CAN_INIT_DATA_BMSTRUCT	Initialisierungsdaten des CAN
x_BUSY	BOOL	Kommunikation ist aktiv
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit
x_OK	BOOL	OK-Bit

Der FB CAN_PE_LIST_WRITE übergibt mit den Werten der Eingänge u_RD_PAR_NR0 / us_RD_FORMAT0 bis Eingänge u_RD_PAR_NR3 / us_RD_FORMAT3 einen PE-Daten-Schreiben-Auftrag an den Regler mit der Gerätenummer "us_DRIVE_NR". Der Regler bearbeitet den PE-Daten-Schreiben-Auftrag und gibt das Ergebnis der Kommunikation zurück.

Die geänderten Istwert-Parameternummern und Istwert-Formate werden in die Variable, die die Initialisierungsdaten der Regler und der CAN-IO-Module enthält, eingetragen (siehe Ein-/Ausgang _CAN_DATA). Diese Variable wird bei einigen FBs für die Kommunikation benötigt.

Ein-/Ausgang _BASE:

An _BASE muß eine globale Variable vom Datentyp CAN_CTRL_BMSTRUCT angeschlossen werden.

Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden und ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Eingang _BASE_CTRL angeschlossen worden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;
```

dabei ist:

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Ein-/Ausgang _CAN_DATA:

An _CAN_DATA wird eine globale Variable vom Datentyp CAN_INIT_DATA_BMSTRUCT angeschlossen.

Diese Variable ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Ausgang _CAN_DATA angeschlossen worden.

Beispiel:

```
_CAN_INIT_DATA_OPT : CAN_INIT_DATA_BMSTRUCT;
```

dabei ist:

`_CAN_INIT_DATA_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_INIT_DATA_BMSTRUCT`	der Datentyp

Diese Variable enthält die Initialisierungsdaten der Regler und der CAN-IO-Module dieser CAN-Anschaltung und wird bei einigen FBs für die Kommunikation benötigt.

Eingang us_DRIVE_NR:

Am Eingang us_DRIVE_NR wird die Gerätenummer des Reglers am CAN-Bus angegeben.

Eingänge u_RD_PAR_NR0 bis u_RD_PAR_NR3:

Die Istwert-Parameternummern werden an den Eingängen u_RD_PAR_NR0 bis u_RD_PAR_NR3 angegeben. Bei nicht belegten Eingängen wird als Istwert-Parameternummer die Parameternummer 0 an den Regler gesendet.

Eingänge us_RD_FORMAT0 bis us_RD_FORMAT3:

Die Formate der Istwerte werden an den Eingängen us_RD_FORMAT0 bis us_RD_FORMAT3 angegeben.

HINWEIS

Der Istwert 0 darf das Format Byte oder Wort haben. Die Istwerte 1, 2 und 3 dürfen das Format Wort oder Doppelwort haben.

Formatangabe:	us_RD_FORMATx = 0:	Format Byte	(Istwert 0)
	us_RD_FORMATx = 1:	Format Wort	(Istwert 0, 1, 2 und 3)
	us_RD_FORMATx = 2:	Format Doppelwort	(Istwert 1, 2 und 3)
	x = 0, 1, 2, 3		

Die bei der Initialisierung am FB CAN_DRIVE_INIT angegebenen Istwert-Zykluszeiten werden beibehalten.

Eingang x_EN:

Die Kommunikation wird mit x_EN = TRUE gestartet. Wird x_EN auf FALSE gesetzt, bevor x_BUSY = FALSE ist, wird von einem bewußten Abbruch der Kommunikation ausgegangen. Der FB CAN_PE_LIST_WRITE muß dann mit x_RESET = TRUE zurückgesetzt werden.

Eingang x_RESET:

Mit dem Eingang x_RESET = TRUE kann der FB zurückgesetzt werden.

Ausgang x_BUSY:

Der Ausgang x_BUSY zeigt mit TRUE an, daß die Kommunikation aktiv ist.

Ausgang x_OK:

Der Ausgang x_OK wird auf TRUE gesetzt, wenn die Kommunikation erfolgreich abgeschlossen ist.

Ausgänge x_ERR, b_ERR:

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Ungültige Gerätenummer (us_DRIVE_NR größer 15)
1	Fehler in der Kommunikation (PE-Daten schreiben)
2 - 7	Reserviert

5.18 CAN_SD_CONTROL

Beschreibung

Diesen Funktionsbaustein für CAN verwenden Sie, um die Bedarfsdatenkommunikation eines Reglers zu überwachen.

HINWEIS

Der FB CAN_SD_CONTROL verwendet die Bibliothek BM_TYPES_20bd00 oder höher.

Der FB CAN_SD_CONTROL muß eingesetzt werden, wenn die FBs der Bedarfsdatenkommunikation
CAN_PAR_READ, CAN_PAR_WRITE,
CAN_OBJ_READ, CAN_OBJ_WRITE,
CAN_PA_LIST_READ, CAN_PA_LIST_WRITE,
CAN_PE_LIST_READ und/oder CAN_PE_LIST_WRITE
genutzt werden.

Parameter Eingang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
_CAN_DATA	CAN_INIT_DATA_BMSTRUCT	Initialisierungsdaten des CAN
us_DRIVE_NR	USINT 0 bis 15	Gerätenummer des Reglers am CAN-Bus
us_MAX_NR_OF_COMM	USINT	maximale Anzahl der Kommunikations-versuche
t_TIME	TIME > 0 s	Überwachungszeit
x_RESET	BOOL	Reset

Parameter Eingang	Datentyp	Beschreibung
_BASE	CAN_CTRL_BMSTRUCT	Betriebsdaten für die CAN-Anschaltung
_CAN_DATA	CAN_INIT_DATA_BMSTRUCT	Initialisierungsdaten des CAN
b_ERR	BYTE	Fehlerbyte
x_ERR	BOOL	Fehlerbit

Ein-/Ausgang _BASE:

An _BASE muß eine globale Variable vom Datentyp CAN_CTRL_BMSTRUCT angeschlossen werden.

Diese Variable muß über die Deklaration der globalen Variablen auf die Basisadresse der CAN-Anschaltung gelegt werden und ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Eingang _BASE_CTRL angeschlossen worden.

Beispiel:

Optionskarte CAN-M-01 für Ωmega Drive-Line II

```
_CAN_CTRL_OPT AT %MB3.3000000 : CAN_CTRL_BMSTRUCT;
```

dabei ist:

`_CAN_CTRL_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_CTRL_BMSTRUCT`	der Datentyp
`%MB3.3000000`	die Basisadresse der CAN-Anschaltung auf der Optionskarte CAN-M-01

Ein-/Ausgang _CAN_DATA:

An _CAN_DATA wird eine globale Variable vom Datentyp CAN_INIT_DATA_BMSTRUCT angeschlossen.

Diese Variable ist im allgemeinen bereits bei der Initialisierung der CAN-Anschaltung am FB CAN_INIT, Ausgang _CAN_DATA angeschlossen worden.

Beispiel:

```
_CAN_INIT_DATA_OPT: CAN_INIT_DATA_BMSTRUCT;
```

dabei ist:

`_CAN_INIT_DATA_OPT`	der Variablenname mit der Datentypkurzbezeichnung "_" für Struct
`CAN_INIT_DATA_BMSTRUCT`	der Datentyp

Diese Variable enthält die Initialisierungsdaten der Regler und der CAN-IO-Module dieser CAN-Anschaltung und wird bei einigen FBs für die Kommunikation benötigt.

Eingang us_DRIVE_NR:

Am Eingang us_DRIVE_NR wird die Gerätenummer des Reglers am CAN-Bus angegeben.

Eingang us_MAX_NR_OF_COMM:

Die maximale Anzahl der Kommunikationsversuche wird am Eingang us_MAX_NR_OF_COMM eingestellt. Ist der Eingang us_MAX_NR_OF_COMM nicht belegt, ergibt sich eine Voreinstellung von us_MAX_NR_OF_COMM = 1.

Die Überwachungszeit wird am Eingang t_TIME in s eingestellt. Wird der Eingang t_TIME nicht belegt, ergibt sich eine Voreinstellung von 3 s.

Eingang x_RESET:

Mit dem Eingang x_RESET = TRUE kann der FB zurückgesetzt werden.

Ausgänge x_ERR, b_ERR:

Falls ein Fehler auftritt, wird das Fehlerbit x_ERR auf TRUE gesetzt und das Fehlerbyte b_ERR ausgegeben.

Fehlerbyte b_ERR:

Bit-Nr.	Fehler
0	Allgemein Timeout aufgetreten
1	Allgemein Fehler
2	Regler nicht initialisiert
3	Es finden keine Kommunikationsversuche statt
4 - 7	Reserviert

HINWEIS

Wenn keine Bedarfsdatenkommunikation stattfindet, wird ein Timer gestartet und das Fehlerbyte auf 16#00 gesetzt. Läuft der Timer ab, wird das Bit 3 des Fehlerbytes gesetzt. Beim nächsten Aufruf des FB CAN_SD_CONTROL wird der Timer wieder gestartet und das Fehlerbyte wird wieder 16#00. D. h. das Bit 3 des Fehlerbytes ist dann jeweils nur bis zum nächsten Aufruf des FB gesetzt.

6 INDEX