CAN 300 PRO – CANopen® Slave

CAN Kommunikations-Baugruppen für S7-300 als CANopen® Slave

Anleitung

Ausgabe 3 / 22.12.2011

Systeme Helmholz **GmbH** | Hannberger Weg 2 | D-91091 Großenseebach
Phone +49 9135 7380-0 | Fax +49 9135 7380-110 | info@helmholz.de | www.helmholz.com

Systeme Helmholz GmbH

Hannberger Weg 2, 91091 Großenseebach

Hinweis:

Der Inhalt dieses Handbuches ist von uns auf die Übereinstimmung mit der beschriebenen Hard- und Software überprüft worden. Da dennoch Abweichungen nicht ausgeschlossen sind, können wir für die vollständige Übereinstimmung keine Gewährleistung übernehmen. Die Angaben in diesem Handbuch werden jedoch regelmäßig aktualisiert. Bitte beachten sie beim Einsatz der erworbenen Produkte jeweils die aktuellste Version des Handbuchs, die im Internet unter www.helmholz.de einsehbar ist und auch heruntergeladen werden kann.

Unsere Kunden sind uns wichtig. Wir freuen uns über Verbesserungsvorschläge und Anregungen.

Änderungen in diesem Dokument:

Stand	Datum	Änderung
1	16.06.2009	1. Version
2	16.08.2010	Hantierungsbausteine überarbeitet; BUSY hinzugefügt
3	22.12.2011	Fehlernummern ergänzt und kleinere Korrekturen

Inhaltsverzeichnis

1 Übersicht

1.1 Allgemein

Die CAN 300 PRO Baugruppe ist für den Einsatz in Siemens Automatisierungsgeräten der S7-300 Baureihe. Mit der CAN 300 PRO Baugruppe kann das Automatisierungsgerät an den CAN-Bus angeschlossen werden.

Neben den Anwendungen als CANopen® Master oder für Layer 2 Kommunikation kann die CAN 300 PRO die S7-300 SPS auch als CANopen® Slave in ein CANopen® Netz einbinden.

Durch Aufspielen einer CANopen® Slave Firmware, einer Slave-Definitionsdatei und mit den zugehörigen Hantierungsbausteinen kann die CAN 300 PRO Baugruppe als CANopen® Slave arbeiten.

Dieses Handbuch soll die Verwendung der Baugruppe als CANopen® Slave erläutern. Es ist als Ergänzung zum Standard Handbuch der CAN 300 PRO Baugruppe zu verwenden.

1.2 Anschlüsse

Die CAN 300 PRO Baugruppe hat hinter der Frontklappe einen 9polige SubD-Stecker für den CAN-Bus und einen USB-Anschluss für die Projektierung und Diagnose.

Steckerbelegung:

Pin	SubD-Stecker CAN
1	-
2	CAN Low
3	CAN GND
4	-
5	-
6	-
7	CAN High
8	-
9	-

Eine 24V Spannungsversorung ist nicht auf dem CAN-Bus Stecker aufgelegt.

1.3 LED-Anzeigen

Die LEDs an der Vorderseite der Baugruppe informieren über den Betriebszustand.

LED „SF“ (Orange):

Systemfehler: zeigt ein fehlerhaftes Projekt an.

LED „BF“ (Rot):

Diese LED zeigt einen CAN-Fehler an. Ein CAN-Fehler liegt vor, wenn die Fehlerzähler nicht Null sind und der CAN-Status nicht „OK“ ist oder ein CAN-FIFO-Overflow vorliegt.

Weitere Informationen erhalten Sie im Debug Modus der CANParam Software (s.a. Kap. 4.3).

LED „RX“ (Grün):

CAN-Bus Empfang aktiv: Zeigt den korrekten Empfang eines CAN-Telegrammes an.

LED „TX“ (Orange):

CAN-Bus Senden aktiv: Zeigt das korrekte Senden eines CAN-Telegrammes an.

LED „CPU“ (Orange):

Datenübertragung zur SPS aktiv: Zeigt die Übertragung eines Telegramms oder Kommandos am Rückwandbus (zwischen S7-CPU und Baugruppe) an.

LED „ON“ (Grün):

Dauerlicht zeigt an, dass die Baugruppe als CANopen® Slave im Modus „Operational“ ist. Langsames Blinken zeigt an, dass die Baugruppe im Modus „Preoperational“ oder „Stopped“ ist.

1.4 DIP-Switch

Der 10fach DIP-Schalter an der Gehäusevorderseite ist zur Einstellung der CAN-Baudrate und zur Festlegung der Nodeadresse vorgesehen.

Adresse	2^6	+ 64
	2^5	+ 32
	2^4	+ 16
	2^3	+ 8
	2^2	+ 4
	2^1	+ 2
	2^0	+ 1
Baud	2^2	+ 4
	2^1	+ 2
	2^0	+ 1

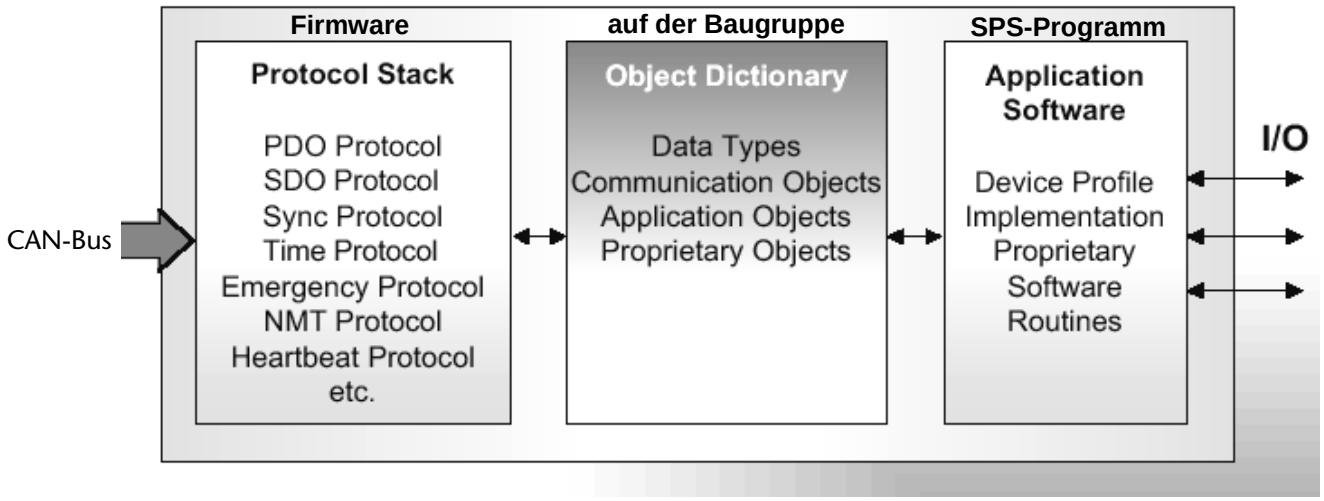
Baudraten:

0	1	2	3	4	5	6	7
10K	50K	100K	125K	250K	500K	800K	1M

2 CANopen® Slave Funktion

2.1 Objekte

Die Schnittstelle für alle Informationen eines CANopen® Slaves ist das „Object Dicitonary“. Die Objekte des Object Dictionary enthalten alle Informationen über den Zustand des Slaves und alle Prozesswerte, die mit dem CAN-Bus Master oder anderen Slaves ausgetauscht werden sollen (IO-Werte, Ist-Werte, Sollwerte, Fehlerzustände, etc.).



Für die Grundeinstellung des CANopen® Slaves sind die Objekte (1000...1FFF) implementiert. Diese Objekte identifizieren den CANopen® Slave gegenüber dem CANopen® Master und enthalten alle Zustandsinformationen.

Die Objekte 6000h…9FFFh sind für Prozessinformationen bei Verwendung von Standard CANopen® Profilen reserviert.

Die Objekte 2000h…3FFFh können herstellerspezifisch verwendet werden.

Die Objekttabelle liegt auf der Baugruppe und wird durch die Hantierungsbausteine mit der SPS ausgetauscht. Die Objekte werden durch die Slave-Definitionsdatei spezifiziert die in die Baugruppe eingespielt wird.

2.2 Node Adressen

Jeder CANopen® Slave muss eine Nodeadresse besitzen. Die Nodeadresse kann zwischen 1 und 127 liegen und kann in der Slave-Definitionsdatei oder über den DIP-Schalter eingestellt werden.

2.3 Netzmanagement

Die CANopen® Slave Hantierung unterstützt BootUp-Meldungen, Heartbeat und Nodeguarding.

Für das Nodeguarding werden die Objekte 100Ch („GuardTime“) und 100Dh („LifetimeFactor“) zur Überwachung verwendet. Läuft die hier eingestellte Zeit ab, so geht der CANopen® Slave selbstständig in den Zustand „Preoperational“.

Beim Heartbeat wird das Senden des Slave Heartbeats unterstützt (Zeit in Objekt 1017h „Producer Heartbeat Time“), sowie maximal 2 Conusmer Heartbeats (Objekt 1016h Subindex 1 und 2)zur Überwachung von empfangenen Heartbeats.

Eine nähere Erläuterung der CANopen® Telegramme entnehmen Sie bitte dem Kapitel 6.

2.4 Emergencies

Das Senden von Emergency Nachrichten und die Verwaltung des aktuellen Fehlerzustandes (Objekte 1001h und 1003h) werden unterstützt.

2.5 Servicedaten (SDO)

Das Lesen und Schreiben von SDOs (1-4 Byte) wird unterstützt. Die SDOs können mit Lese oder Schreib-/Leserechten versehen werden.

Lesen und Schreiben von SDOs mit mehr als 4 Byte (Segmented Transfer) wird ebenfalls unterstützt.

2.6 Prozessdaten (PDO)

Die CANopen® Slave Hantierung unterstützt bis zu 4 Sende- und Empfangs-PDOs (RPDO1-4, TPDO1-4). Die COB-IDs der PDOs sind fest in der Hantierung hinterlegt, eine Änderung ist nicht möglich.

TPDO1: 180h+Node-ID	RPDO1: 200h+Node-ID
TPDO2: 280h+Node-ID	RPDO2: 300h+Node-ID
TPDO3: 380h+Node-ID	RPDO3: 400h+Node-ID
TPDO3: 480h+Node-ID	RPDO3: 500h+Node-ID

Das Mapping der PDOs ist über die üblichen Objekte 1600h ff. und 1A00h ff. möglich.

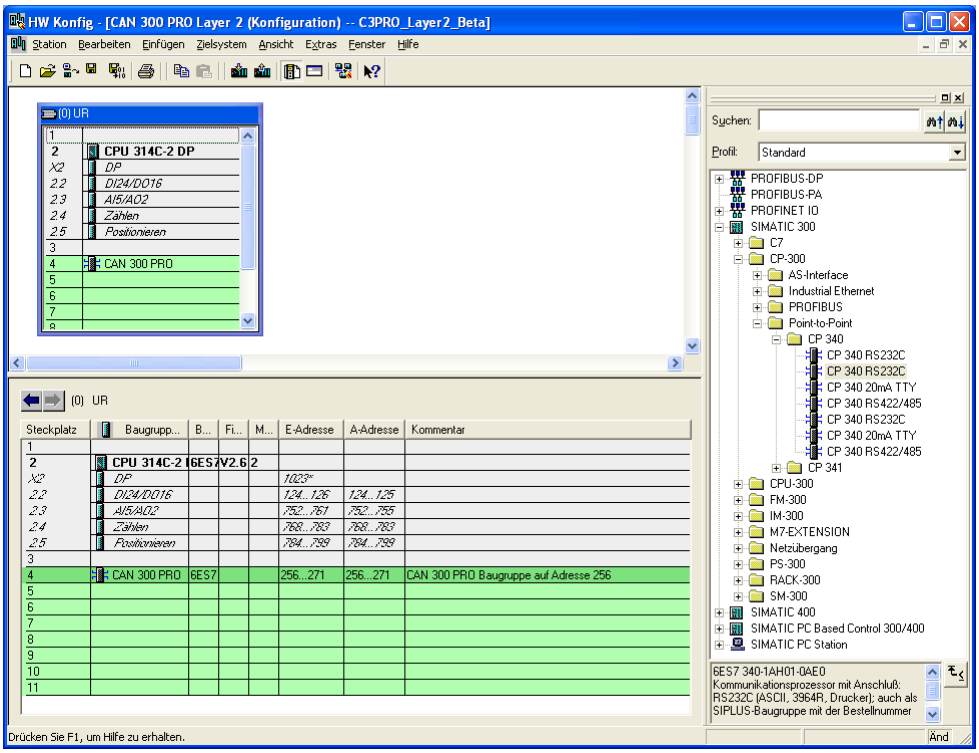
Als Transmission Type für TPDOs (Objekte 1800h ff.), stehen folgende Optionen zur Verfügung:

Senden nach x SYNC-Telegrammen:	1-240
Senden nur nach Anforderung (RTR):	252, 253
Senden nach Änderung:	254, 255

RPDO-Werte werden sofort nach Empfang übernommen.

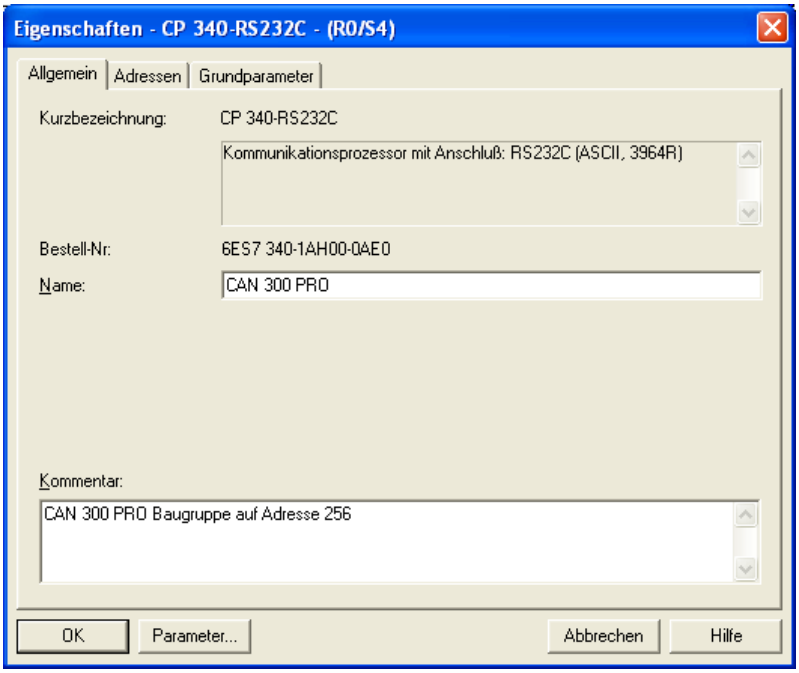
3 Projektierung in der SPS

Die CAN 300 PRO Baugruppe wird in der Programmiersoftware der SPS als CP 340 Kommunikationsbaugruppe projektiert.



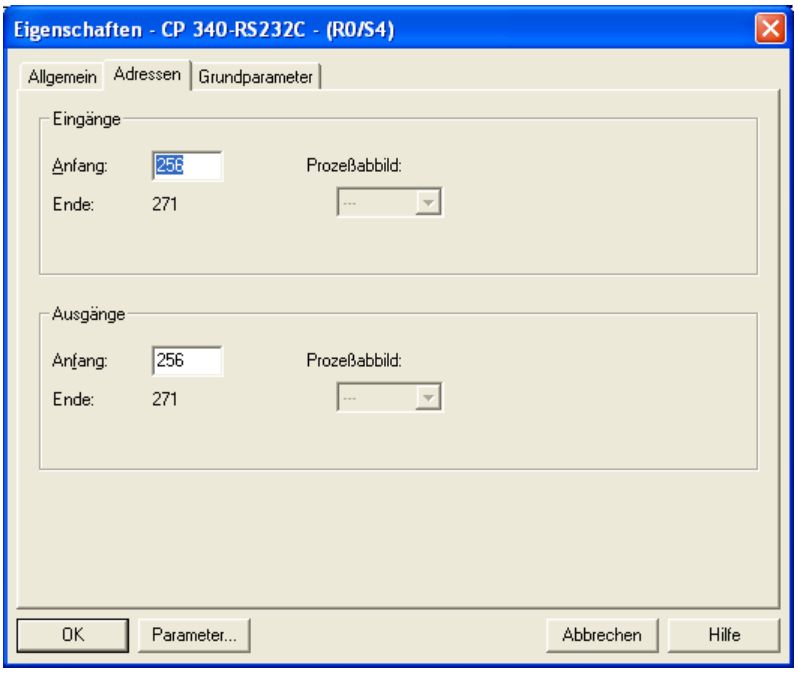
Beim Einsatz der CAN 300 PRO Baugruppe in einem ET200M System ist mit starken Performanceeinbußen zu rechnen.

Die Baugruppe kann überall dort eingesetzt werden, wo auch eine CP-Baugruppe erlaubt ist, d.h. auch im Erweiterungsrahmen nach einer Anschaltung.



Bei der Parametrierung der Baugruppe ist nur der Bereich der E/A-Adressen relevant. Alle anderen Einstellungen haben keine Auswirkung auf die Baugruppe.

*Die E/A-Adressen sollten **nicht** im zyklischen Prozessabbild liegen!*



Es wird in den Hantierungsbausteinen nur das Eingangsabbild verwendet, das Ausgangsabbild hat keine funktionale Bedeutung.

Zugriffe auf das Eingangsabbild können nur mit den Peripheriedirektzugriffsbefehlen durchgeführt werden: L PEB, L PEW, L PED.

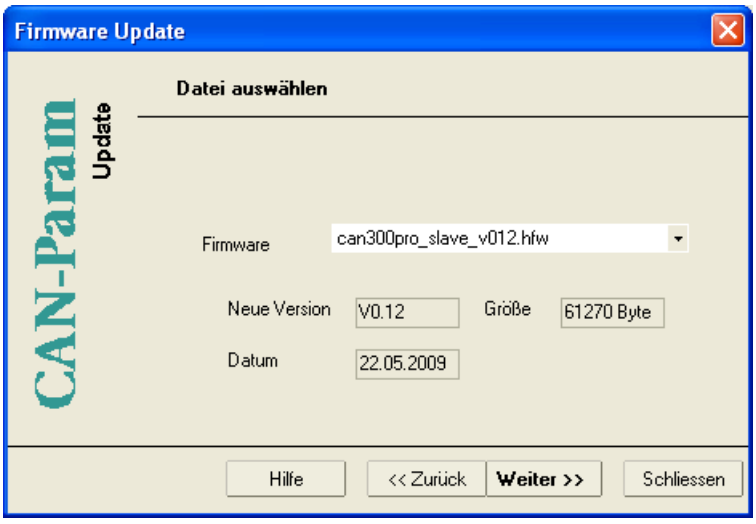
Bei der CPU 318 müssen die Peripherieadressen außerhalb des zyklischen Prozessabbildes liegen.

4 Projektierung der CAN 300 PRO Baugruppe

4.1 CANopen® Slave Betriebssystem übertragen

Die Projektierung der CAN Baugruppen wird auf dem PC mit der „CANParam V4.1x“-Software und höher durchgeführt. Im Auslieferungszustand befindet sich auf der CAN 300 PRO Baugruppe das Betriebssystem für einen CANopen® Master. Die Baugruppe muss zur ersten Inbetriebnahme mit der aktuellen CANopen® Slave Firmware versehen werden.

Dazu kann die in der CANParam Software vorhandene Funktion „Firmware Update“ im Menü „Online“ verwendet werden. Nach dem Start des Updates muss die aktuelle Slave-Firmware gewählt werden.



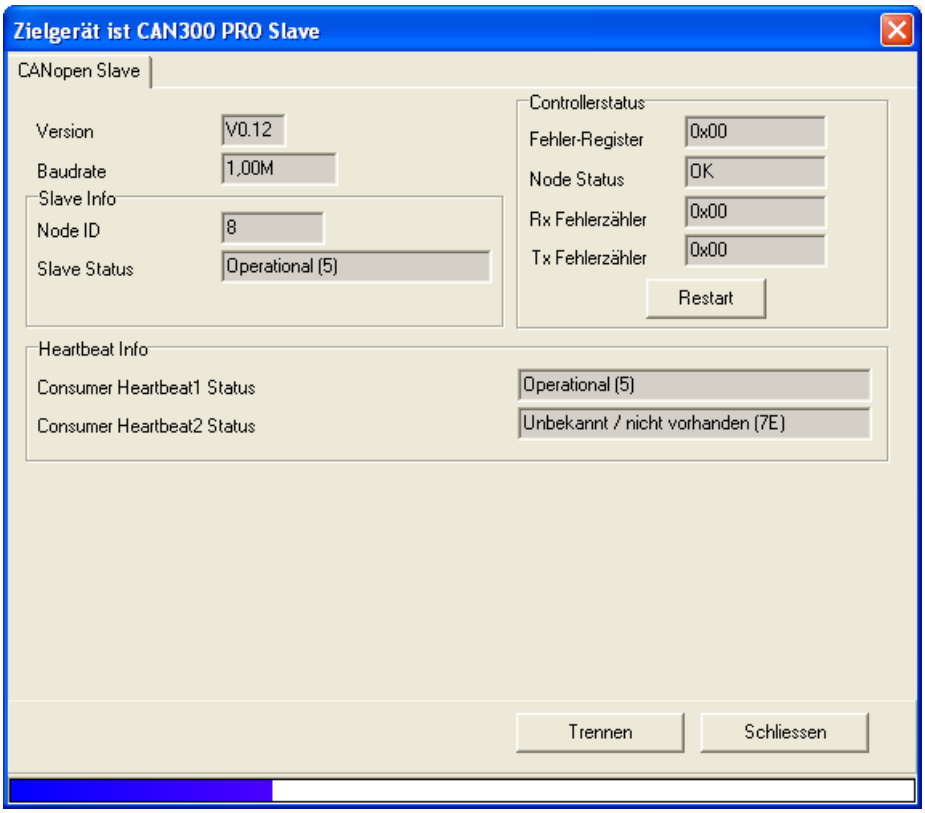
4.2 Slave Definitionsdatei übertragen

Im Menü „Online“ kann mit der Funktion „Sende Projekt aus Datei“ die gewünschte Slave Definitionsdatei zur Baugruppe gesendet werden.

Die Slave-Definitionsdatei ist eine Textdatei mit der Endung „PAR“. Diese Datei beschreibt den Aufbau und Grundzustand des Slaves mit allen SDOs und PDOs.

Eine Slave Definitionsdatei für das CANopen® Profil DS 401 wird mitgeliefert. Für andere CANopen® Anwendungen können beliebige Definitionsdateien erstellt werden, wenden Sie sich hierzu an den Systeme Helmholz Support.

4.3 Diagnose



Folgende Informationen liefert der CANopen® Debug-Dialog:

Version Versionsnummer des Betriebssystems

Baudrate aktive CAN-Baudrate

Controllerstatus Inhalt des CAN-Status-Registers:

Fehlerregister Inhalt des CAN-Fehler-Registers EFLG (Kap. 5.1.2)

Node-Status Inhalt des CAN-Status-Registers (s.o.):
"OK", "Warning", "Passiv", "Bus Off"

Rx Fehlerzähler Fehlerzähler CAN-Empfang

Tx Fehlerzähler Fehlerzähler CAN-Senden

Slave Informationen:

Node ID Aktuell gültige Node ID des Slaves

Slave Status 00 = Bootup
04 = Stop
05 = Operational
7F = Preoperational

Heartbeat Info:

An dieser Stelle wird der Zustand der überwachten Slaves angezeigt.

! *Node Status sollte immer auf „OK" stehen, damit eine störungsfreie CAN-Datenübertragung möglich ist.*

5 Programmierung in der SPS

Die Programmierung des CANopen® Slaves erfolgt in der SPS über Hantierungsbausteine und unter Verwendung von Informationen aus dem Prozessabbild der Baugruppe.

5.1 Prozessabbild in der SPS

Die CAN 300 PRO Baugruppe belegt 16 Bytes im Eingangs- und Ausgangs-Prozessabbild. Der Inhalt des Ausgangs-Prozessabbildes wird nicht verwendet.

Der Inhalt des Eingangs-Prozessabbildes kann vom Anwender in der Applikation zu Informationszwecken verwendet werden:

Byte	Bedeutung
0	Baugruppenstatus allgemein, CAN Sammelfehler Anzeige
1	CAN-Controllers Status (Register des CAN-Controllers)
2	FIFO Status Bits (Send & Receive)
3	CAN-Controller: TX-Fehlerzähler
4	CAN-Controller: RX-Fehlerzähler
5	CANopen® Slave Status (0, 4, 5, 0x7f)
6	Zustand Heartbeat Consumer 1
7	Zustand Heartbeat Consumer 2
8	aktive Node-ID des Slaves
9…15	*intern verwendet*

Zugriffe auf das Eingangsabbild können nur mit den Peripherie-direktzugriffsbefehlen durchgeführt werden: L PEB, L PEW, L PED.

5.1.1 Byte 0: Baugruppenstatus

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CAN-Controller Sammelfehler	0	Baugruppe ist CAN 300 PRO **Slave**	0	0	0	0	Baugruppe parametriert und läuft

Bit 0: Die CAN 300 PRO Baugruppe hat die Projektierung verarbeitet und ist betriebsbereit.

Bit 5: Dieses Bit ist immer 1, um die CAN 300 PRO Slave erkennen zu können.

Bit 7: Sammelfehlerbit für Fehler am CAN-Controller, genauere Auskunft über die Fehlerursache ist im Byte 1 zu erkennen.

5.1.2 Byte 1: Fehler-Status (EFLG) des CAN-Controllers

RX1OVR	RX0OVR	TXBO	TXEP	RXEP	TXWAR	RXWAR	EWARN
bit 7							bit 0

bit 7 **RX1OVR**: Receive Buffer 1 Overflow Flag
- Set when a valid message is received for RXB1 and CANINTF.RX1IF = 1
- Must be reset by MCU

bit 6 **RX0OVR**: Receive Buffer 0 Overflow Flag
- Set when a valid message is received for RXB0 and CANINTF.RX0IF = 1
- Must be reset by MCU

bit 5 **TXBO**: Bus-Off Error Flag
- Bit set when TEC reaches 255
- Reset after a successful bus recovery sequence

bit 4 **TXEP**: Transmit Error-Passive Flag
- Set when TEC is equal to or greater than 128
- Reset when TEC is less than 128

bit 3 **RXEP**: Receive Error-Passive Flag
- Set when REC is equal to or greater than 128
- Reset when REC is less than 128

bit 2 **TXWAR**: Transmit Error Warning Flag
- Set when TEC is equal to or greater than 96
- Reset when TEC is less than 96

bit 1 **RXWAR**: Receive Error Warning Flag
- Set when REC is equal to or greater than 96
- Reset when REC is less than 96

bit 0 **EWARN**: Error Warning Flag
- Set when TEC or REC is equal to or greater than 96 (TXWAR or RXWAR = 1)
- Reset when both REC and TEC are less than 96

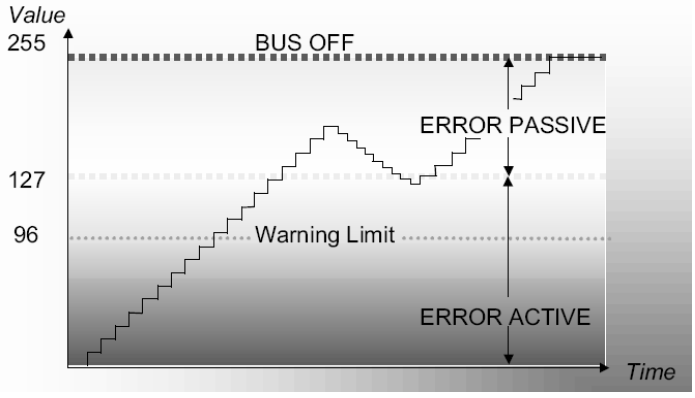
5.1.3 Byte 2: FIFO-Status Bits

Bit 7	Bit 6	Bit 5	Bit 4
Send-FIFO (high) halb voll	Send-FIFO (high oder low) Overflow	Send-FIFO (low) halb voll	Send-FIFOs (high & low) ganz leer

Bit 3	Bit 2	Bit 1	Bit 0
Receive-FIFO (high) halb voll	Receive -FIFO (high oder low) Overflow	Receive-FIFO (low) halb voll	Receive-FIFOs (high & low) ganz leer

5.1.4 Byte 3/4: CAN-Controller Tx/Rx Fehlerzähler

Der Fehlerzähler wird bei jedem fehlerhaft versendeten oder empfangenen CAN-Telegramm hochgezählt. Wenn ein CAN-Telegramm korrekt übertragen wurde, wird der Fehlerzähler wieder heruntergezählt. Wenn der Zähler größer als 96 ist, geht der CAN-Controller in den Modus „Warning“ (siehe 5.1.2). Sollte der Fehlerzähler 127 übersteigen, geht der CAN-Controller in „Error-Passiv“.



5.1.5 Byte 5: CANopen® Slave Status

In diesem Byte ist der aktuelle Zustand der CANopen® Slave Statemachine ersichtlich.

0 = Bootup
4 = Stop
5 = Operational
7F = Preoperational

5.1.6 Byte 6+7: Zustand Heartbeat Consumer 1+2

In diesem Byte ist der aktuelle Zustand der durch den Consumer Heartbeat überwachten Geräte ersichtlich:

0 = Bootup
4 = Stop
5 = Operational
7E = unbekannt/nicht vorhanden
7F = Preoperational

5.1.7 Byte 8: aktive Node-ID

Dieses Byte zeigt die aktive Node-ID des Slaves an.

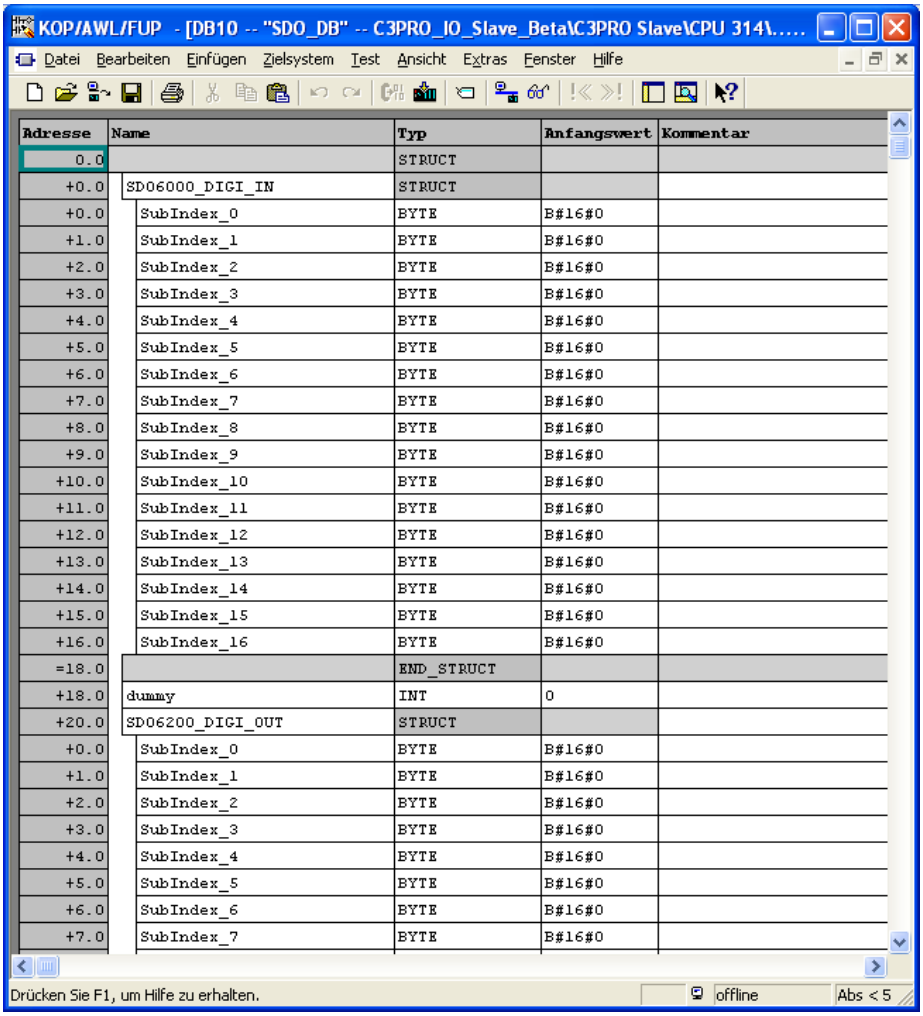
5.2 Hantierungsbausteine

Die Programmierung des CANopen® Slaves erfolgt in der SPS über folgende Hantierungsbausteine:

FB 90	**Get SDO Block**	SDO-Daten in die SPS holen
FB 91	**Send SDO Block**	SDO-Daten an die Baugruppe senden
FB 92	**SDO Write**	Ein SDO schreiben
FB 93	**SDO Read**	Ein SDO auslesen
FB 94	**Send Emergency**	Emergency Nachricht absetzen

5.2.1 SDO Datenbaustein

Die aktuellen Daten aller definierten SDOs werden in der CAN 300 PRO Baugruppe vorgehalten. Um mit diesen Daten in der SPS arbeiten zu können, werden diese am Anfang des Zyklus mit dem Funktionsbaustein FB 90 von der Baugruppe in einen Datenbaustein, den SDO-DB, kopiert.



Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	SD06000_DIGI_IN	STRUCT		
+0.0	SubIndex_0	BYTE	B#16#0	
+1.0	SubIndex_1	BYTE	B#16#0	
+2.0	SubIndex_2	BYTE	B#16#0	
+3.0	SubIndex_3	BYTE	B#16#0	
+4.0	SubIndex_4	BYTE	B#16#0	
+5.0	SubIndex_5	BYTE	B#16#0	
+6.0	SubIndex_6	BYTE	B#16#0	
+7.0	SubIndex_7	BYTE	B#16#0	
+8.0	SubIndex_8	BYTE	B#16#0	
+9.0	SubIndex_9	BYTE	B#16#0	
+10.0	SubIndex_10	BYTE	B#16#0	
+11.0	SubIndex_11	BYTE	B#16#0	
+12.0	SubIndex_12	BYTE	B#16#0	
+13.0	SubIndex_13	BYTE	B#16#0	
+14.0	SubIndex_14	BYTE	B#16#0	
+15.0	SubIndex_15	BYTE	B#16#0	
+16.0	SubIndex_16	BYTE	B#16#0	
=18.0		END_STRUCT		
+18.0	dummy	INT	0	
+20.0	SD06200_DIGI_OUT	STRUCT		
+0.0	SubIndex_0	BYTE	B#16#0	
+1.0	SubIndex_1	BYTE	B#16#0	
+2.0	SubIndex_2	BYTE	B#16#0	
+3.0	SubIndex_3	BYTE	B#16#0	
+4.0	SubIndex_4	BYTE	B#16#0	
+5.0	SubIndex_5	BYTE	B#16#0	
+6.0	SubIndex_6	BYTE	B#16#0	
+7.0	SubIndex_7	BYTE	B#16#0	

Es ist in der Slave-Definitionsdatei eine feste Zuordnung zwischen dem SDO und der Speicherstelle im SDO-Datenbaustein definiert.

Die Daten der SDOs können nun im SPS Zyklus verarbeitet und auch geändert werden. Am Ende des SPS Zyklus wird der komplette Inhalt des SDO-DBs mit dem FB 91 wieder zur Baugruppe übertragen.

Schreibzugriffe auf SDOs über den CAN-Bus haben Priorität!

Sollte in der Zwischenzeit vom CAN-Bus ein Schreibzugriff auf ein SDO stattgefunden haben, so hat dieses Priorität gegenüber den Veränderungen in der SPS.

Um den Datenbaustein nicht zu groß werden zu lassen und die Übertragungszeit zwischen SPS und der Baugruppe zu begrenzen, müssen nicht immer alle definierten SDOs kopiert werden.

In der Slave-Definitionsdatei werden nur diejenigen SDOs in den SDO-Datenbaustein „gemappt“, die auch eine Relevanz für den zyklischen Betrieb haben. Alle anderen SDOs, die selten oder nur einmalig gelesen oder geschrieben werden sollen, können über die Funktionsbausteine FB 92 und FB 93 gelesen und geschrieben werden.

5.2.2 FB 90 Get SDO Block

Der Funktionsbaustein Get SDO Block (FB90) holt den aktuellen Zustand der SDO-Tabelle in die SPS und legt diese Daten in einen beliebigen Datenbaustein ab.

Der FB sollte am Anfang des SPS Zyklus aufgerufen werden.

Parameter	Richtung	Typ	Beispiel
Base	IN	INT	256
SDO_Block	IN	ANY	P#DB10.DBX0.0 BYTE 100
STAT	OUT	WORD	MW 10
Err	OUT	BOOL	M 94.7
RetVal	OUT	INT	MW12
Busy	IN_OUT	BOOL	
Lock	IN_OUT	BOOL	

Base	Adresse der CAN 300 PRO Baugruppe
SDO_Block	ANY-Pointer auf den SDO-Datenbaustein
STAT	Status der Baugruppe, siehe Kapitel 5.2.7
Err	Fehler-Bit ist 0 bei erfolgreicher Ausführung
RetVal	Fehlernummer, siehe Kapitel 5.4
Busy/Lock	reserviert für ET200M Anwendung

5.2.3 FB 91 Send SDO Block

Der Funktionsbaustein Send SDO Block (FB91) sendet den aktuellen Zustand der SDO-Tabelle aus dem SDO-Datenbaustein in die Baugruppe.

Der FB sollte am Ende des SPS Zyklus aufgerufen werden.

Parameter	Richtung	Typ	Beispiel
Base	IN	INT	256
SDO_Block	IN	ANY	P#DB10.DBX0.0 BYTE 100
STAT	OUT	WORD	MW 10
Err	OUT	BOOL	M 95.7
RetVal	OUT	INT	MW14
Busy	IN_OUT	BOOL	
Lock	IN_OUT	BOOL	

Base	Adresse der CAN 300 PRO Baugruppe
SDO_Block	ANY-Pointer auf den SDO-Datenbaustein
STAT	Status der Baugruppe, siehe Kapitel 5.2.7
Err	Fehler-Bit ist 0 bei erfolgreicher Ausführung
RetVal	Fehlernummer, siehe Kapitel 5.4
Busy/Lock	reserviert für ET200M Anwendung

5.2.4 FB 92 SDO Write

Der Funktionsbaustein SDO Write (FB 92) schreibt einen SDO-Wert zur Baugruppe.

Parameter	Richtung	Typ	Beispiel
Base	IN	INT	256
Index	IN	WORD	W#16#2001
Subindex	IN	BYTE	B#16#0
SDO_Data	IN	DWORD	MD 30
SDO_Len	IN	BYTE	MB 34
RetVal	OUT	INT	MW 35
Activate	IN_OUT	Bool	M 39.0
Busy	IN_OUT	Bool	M 39.1
Err	IN_OUT	Bool	M 39.2
Done	IN_OUT	Bool	M 39.3

Base	Adresse der CAN 300 PRO Baugruppe
Index	SDO-Index
Subindex	SDO-Subindex
SDO_Data	Daten für das SDO (rechtsbündig)
SDO_Len	Größe des SDOs (1, 2, 4 Bytes)
RetVal	Fehlernummer, siehe Kapitel 5.4
Activate	Aktivieren des Auftrags
Busy	Auftrag läuft
Err	Auftrag mit Fehler beendet
Done	Auftrag ohne Fehler beendet

Der Wert des SDOs muss unabhängig der Daten-Länge rechtsbündig im Doppelwort liegen.

Aufrufbeispiel:

```
CALL  FB    92 , DB92
 Base    :=256
 Index   :=W#16#2001
 Subindex:=B#16#0
 SDO_Data:=MD30
 SDO_Len :=B#16#2
 RetVal  :=MW35
 Activate:=M39.0
 Busy    :=M39.1
 Err     :=M39.2
 Done    :=M39.3
…
```

5.2.5 FB 93 SDO Read

Der Funktionsbaustein SDO Read (FB 93) holt einen SDO aus der Baugruppe.

Parameter	Richtung	Typ	Beispiel
Base	IN	INT	256
Index	IN	WORD	W#16#2000
Subindex	IN	BYTE	B#16#0
RetVal	OUT	INT	MW 25
SDO_Data	IN_OUT	DWORD	MD 20
SDO_Len	IN_OUT	BYTE	MB 24
Activate	IN_OUT	Bool	M 39.0
Busy	IN_OUT	Bool	M 39.1
Err	IN_OUT	Bool	M 39.2
Done	IN_OUT	Bool	M 39.3

Base	Adresse der CAN 300 PRO Baugruppe
Index	SDO-Index
Subindex	SDO-Subindex
RetVal	Fehlernummer, siehe Kapitel 5.4
SDO_Data	Daten für das SDO (rechtsbündig)
SDO_Len	Größe des SDOs (1, 2, 4 Bytes)
Activate	Aktivieren des Auftrags
Busy	Auftrag läuft
Err	Auftrag mit Fehler beendet
Done	Auftrag ohne Fehler beendet

Der Wert liegt immer „rechtsbündig“ im Doppelwort und kann sofort weiterverarbeitet werden, bei 1 oder 2 Byte Werten wird das Doppelwort mit führenden Nullen gefüllt.

Aufrufbeispiel:

```
      CALL  FB    93 , DB93
       Base    :=256
       Index   :=W#16#2000
       Subindex:=B#16#0
       RetVal  :=MW25
       SDO_Data:=MD20
       SDO_Len :=MB24
       Activate:=M29.0
       Busy    :=M29.1
       Err     :=M29.2
       Done    :=M29.3

      UN    M     29.3
      SPB   next

      // empfangenen Wert verarbeiten
      R     M     29.3
      L     MD    20
      …

next: …
```

5.2.6 FB 94 Send Emergency

Der Funktionsbaustein Emergency (FB 94) sendet ein Emergency-Telegramm und trägt die Meldung in die entsprechenden SDOs ein.

Parameter	Richtung	Typ	Beispiel
Base	IN	INT	256
Emcy_Code	IN	WORD	W#16#1000
Emcy_State	IN	BYTE	B#16#03
Emcy_ManuErr	INT	DWORD	DW#16#00000000
STAT	OUT	WORD	MW 68
Err	OUT	BOOL	M 63.7
RetVal	OUT	INT	MW 70

Base	Adresse der CAN 300 PRO Baugruppe
Emcy_Code	Emergency-Code
Emcy_State	Emergency Status für Objekt 1001h
Emcy_ManuErr	Emergency Manufacturer Error
STAT	Status der Baugruppe, siehe Kapitel 5.2.7
Err	Fehler-Bit ist 0 bei erfolgreicher Ausführung
RetVal	Fehlernummer, siehe Kapitel 5.4

5.2.7 Parameter STAT

Der Parameter STAT hat bei allen Hantierungsbausteinen die gleiche Bedeutung und zeigt den Zustand der Baugruppe an:

Bit 15	Bit 14	Bit 13	Bit 12
CAN-Controller Sammelfehler	0	Baugruppe ist CAN 300 PRO Slave (immer 1)	0
Bit 11	**Bit 10**	**Bit 9**	**Bit 8**
0	0	0	Baugruppe läuft, Einlesen der Parameter abgeschlossen

Bit 7	Bit 6	Bit 5	Bit 4
Send-FIFO (high) halb voll	Send-FIFO (high oder low) Overflow	Send-FIFO (low) halb voll	Send-FIFOs (high & low) ganz leer
Bit 3	**Bit 2**	**Bit 1**	**Bit 0**
Receive-FIFO (high) halb voll	Receive -FIFO (high oder low) Overflow	Receive -FIFO (low) halb voll	Receive-FIFOs (high & low) ganz leer

Der Parameter STAT entspricht inhaltlich den Peripherieeingangs-bytes 0 und 2.

5.3 Abortcodes

Code	Bedeutung
0503 0000h	"Toggle Bit "wurde nicht alterniert
0504 0000h	SDO Protokoll "time out"
0504 0001h	Client/Server Kommando Bezeichner nicht gültig oder unbekannt
0504 0005h	Außerhalb des Speichers
0601 0000h	Zugriff auf dieses Objekt wird nicht unterstützt
0601 0001h	Versuchter Lesezugriff auf ein Objekt, das nur geschrieben werden kann
0601 0002h	Versuchter Schreibzugriff auf ein Objekt, das nur gelesen werden kann
0602 0000h	Objekt existiert im Objektverzeichnis nicht
0604 0041h	Objekt kann nicht in ein PDO "gemappt"werden
0604 0042h	Größe und Anzahl der "gemappten "Objekte übersteigt die mögliche PDO Länge
0604 0043h	Allgemeine Parameter –Inkompatibilität
0604 0047h	Allgemeine Inkompatibilität im Gerät
0606 0000h	Zugriffsverletzung aufgrund eines Hardwarefehlers
0607 0010h	Datentyp passt nicht, Länge des Service Parameters passt nicht
0607 0012h	Datentyp passt nicht, Länge des Service Parameters zu groß
0607 0013h	Datentyp passt nicht, Länge des Service Parameters zu klein
0609 0011h	Subindex existiert nicht
0609 0030h	Wertebereich des Parameters verlassen (nur für Schreibzugriffe)
0609 0031h	Wert des Parameters zu groß
0609 0032h	Wert des Parameters zu klein
0609 0036h	Maximaler Wert ist kleiner als minimaler Wert

5.4 Fehlercodes der FBs

Der Rückgabeparameter RetVal der Funktionsbausteine kann sowohl funktionsspezifische Fehler enthalten oder Fehlernummern der Siemens Systemfunktionsbausteine SFC 52, SFC 53, SFC 14 und SFC 20.

Fehlercodes der CAN-Hantierung:

80E1h: SDO-FBs: Parameter SDO-Len darf nicht 0 sein

80E2h: SDO-FBs: Parameter SDO-Len darf nicht größer 4 sein

80F1h: Baugruppe nicht Betriebsbereit

80F2h: Datensatz belegt

80F7h: CANopen® Slave noch im Bootup

8xF8h: SDO-Datenbaustein Pointer: Speicher zu klein

80FAh: Abortcode auf SDO-Auftrag empfangen

6 CANopen® Protokoll

6.1 Allgemein

CIA® = CAN in Automation e.V.

Das CANopen® Protokoll ist ein Schicht 7-Protokoll (Application Layer), das auf den CAN-Bus (ISO 11898) aufsetzt. Die Schichten 1 & 2 (Physical Layer/Data Link Layer) vom CAN-Bus bleiben unberührt.

Die CANopen® Kommunikationsprofile für die verschiedenen Anwendungen werden von der CIA verwaltet.

Die von der Anwendungsschicht bereitgestellten Dienstelemente ermöglichen die Realisierung einer über das Netzwerk verteilten Applikation. Diese Dienstelemente sind in der „CAN Application Layer (CAL) for Industrial Applications“ beschrieben.

Der 11 Bit Identifier und die 8 Datenbytes eines CAN-Layer 2 Telegrammes bekommen eine feste Bedeutung.

Jedes Gerät in einem CANopen® Netz hat eine feste Node-ID (Modulnummer, 1-127).

6.2 Objekte

Der Datenaustausch mit einem CANopen® Slave erfolgt entweder über fest definierte Servicedaten-Objekte (SDO) oder über frei konfigurierbare Prozessdaten-Objekte (PDO).

Jeder CANopen® Slave besitzt ein festes Verzeichniss von SDOs, die über eine Objektnummer (16 Bit) und einen Index (8 Bit) angesprochen werden.

Beispiel: Objekt 1000h/ Index 0 = Device Type, 32Bit Unsigned

SDOs mit 8/16/32 Bit Breite können mit einem CANopen® Telegramm gelesen und geschrieben werden. SDOs, die länger sind, werden über mehrere Telegramme übertragen. Für sehr große Datenmengen ist die SDO-Blockübertragung vorgesehen.

SDOs können bearbeitet werden, sobald ein CANopen® Slave betriebsbereit ist. Für die SDOs stehen nur die COB-ID Funktionen „SDO Anforderung“ oder „SDO Antwort“ bereit. Die Objektnummer, der Zugriffsmodus und Typ werden in den ersten 4 Bytes des CAN-Telegrammes hinterlegt.

Die letzen 4 Bytes des CAN-Telegrammes enthalten dann den Wert für das SDO.

PDOs enthalten die „Arbeitswerte“ eines CANopen® Slaves für den zyklischen Prozessbetrieb. Jeder CANopen® Slave kann mehrere PDOs verwalten (im Normalfall bis zu 4 zum Senden und 4 zum Empfangen).

Jedes der vorhandenen PDOs hat eine eigene COB-ID. In den 8 Datenbytes des Telegrammes können beliebige Informationen des CANopen® Slaves zum Lesen und Beschreiben „gemapped“ werden. Dieses können sowohl bereits vorhandene SDOs sein, als auch Aktualwerte des Slaves (z.B. Analogwert eines Einganges).

Die PDOs werden von den meisten CANopen® Slaves automatisch beim Anlauf gemapped. Die Zuordnung kann dann über bestimmte SDOs geändert werden.

6.3 Funktionen

Die CANopen® Funktionen teilen sich in drei Grundarten auf:

- SDO Lesen und Schreiben
- PDO Lesen und Schreiben
- Netzmanagement

Der Funktionscode wird in den oberen 4 Bit des Identifiers hinterlegt. Zusammen mit der Node-ID ergeben sie den COB-Identifier.

COB-Identifier (COB-ID):

10	9	8	7	6	5	4	3	2	1	0
Function				Node-ID						

Es ist mit speziellen Servicedata-Objekten (SDOs) möglich, einige COB-IDs auf andere Werte zu ändern. Dieses wird vom CANopen® Slave NICHT unterstützt!

Broadcast-Funktionen:

Funktion	Function code (binary)	Resulting COB-ID
NMT	0000	0h
SYNC	0001	80h
TIME STAMP	0010	100h

Node Funktionen:

Funktion	Function code (binary)	Resulting COB-ID
EMERGENCY	0001	81h –FFh
PDO1 (tx)	0011	181h – 1FFh
PDO1 (rx)	0100	201h – 27Fh
PDO2 (tx)	0101	281h – 2FFh
PDO2 (rx)	0110	301h – 37Fh
PDO3 (tx)	0111	381h – 3FFh
PDO3 (rx)	1000	401h – 47Fh
PDO4 (tx)	1001	481h – 4FFh
PDO4 (rx)	1010	501h – 57Fh
SDO (tx)	1011	581h – 5FFh
SDO (rx)	1100	601h – 67Fh
NMT Error Control	1110	701h – 77Fh

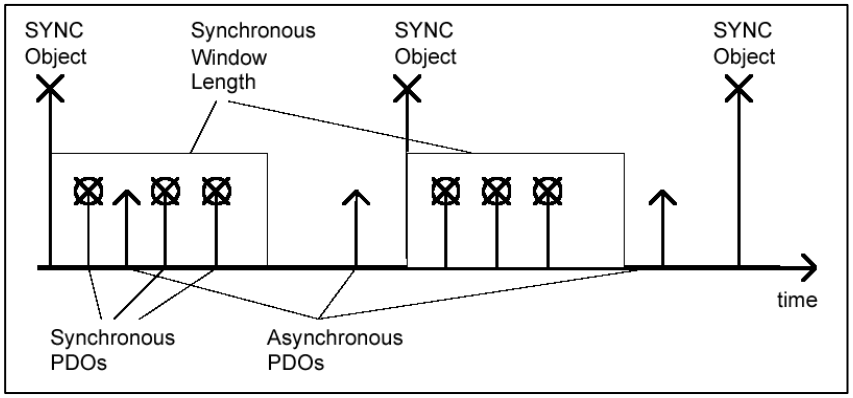
„Tx“ = wird vom Slave gesendet
„Rx“ = wird vom Slave empfangen

6.4 Netzmanagement

SYNC:

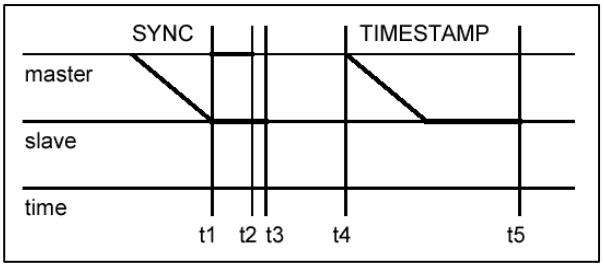
Das SYNC-Telegramm ist ein periodisches “Broadcast”–Telegramm und gibt den Basis-Bustakt vor. Um einen zeitlich äquidistanten Abstand zu ermöglichen, besitzt das SYNC-Telegramm eine hohe Priorität.

COB-ID: 80h



Time Stamp:

Das Time Stamp-Telegramm ist ein periodisches “Broadcast”–Telegramm und gibt die Systemzeit vor. Das Time Stamp-Telegramm wird üblicherweise direkt nach einem SYNC-Telegramm übertragen und gibt dann die Systemzeit des SYNC-Telegrammes an.



Um eine zeitlich genaue Übertragung zu ermöglichen, besitzt das Time Stamp-Telegramm eine hohe Priorität.

COB-ID: 100h

Nodeguarding:

Beim Nodeguarding überwacht der Master die CANopen® Slave Baugruppen durch zyklisch gesendete Telegramme an jeden Slave. Auf das Nodeguarding-Telegramm muss jeder CANopen® Slave mit einem Status-Telegramm antworten.

Mittels Nodeguarding kann die Steuerung den Ausfall eines CANopen® Slaves erkennen.

COB-ID: 700h + Node-ID +RTR

Antwort: COB-ID: 700h + Node-ID + 1 Byte Daten: Zustand

Lifeguarding:

Beim Lifeguarding überwacht jeder CANopen® Slave, ob der Master das einmal gestartete Nodeguarding kontinuierlich innerhalb bestimmter Zeitlimits durchführt.

Wenn das Nodeguarding Telegramm des Masters ausbleibt, kann die dezentrale Peripheriebaugruppe dieses mittels Lifeguarding feststellen und z.B. alle Ausgänge in den sicheren Zustand versetzen.

Heartbeat:

Die Heartbeat-Überwachung entspricht dem Nodeguarding, wobei aber keine Anforderungstelegramme vom CANopen® Master generiert werden. Das Heartbeat Telegramm wird vom Node selbsttätig gesendet und kann im Master ausgewertet werden.

Emergency-Message:

Sollte bei einem CANopen® Slave ein Störfall auftreten, so sendet er eine Emergency-Message auf den Bus.

COB-ID: 80h + Node-ID

Auf ein Emergency-Telegramm können alle Teilnehmer z.B. einen Notstop ausführen.

BootUp-Message:

CANopen® Slaves erzeugen nach dem Einschalten eine BootUp-Meldung, die der Master erkennen kann, um diesen neuen Teilnehmer zu initialisieren.

COB-ID: 700h + Node-ID + 1 Byte Daten: 00h

7 Anhang

7.1 Objektverzeichnis

Im Folgenden finden Sie eine Übersicht über die implementierten Objekte des CANopen® Slaves.

Die Systemobjekte sind immer vorhanden. Alle anderen Objekte werden durch die Slave-Definitionsdatei beim Einspielen in die Baugruppe angelegt.

7.1.1 Systemobjekte (1000h - 1FFFh)

Objekt	Beschreibung	Wertebereich	Info
1000	Device Type	UNSIGNED32	Profil 401
1001	Error Register	UNSIGNED8	Aktueller Fehlerstatus
1003	Predefined Error Field	ARRAY	
/0		UNSIGNED8	Anzahl Einträge
/1		UNSIGNED32	Aktueller Fehler
1004	PDOs supported	UNSIGNED32	4 Sende- und 4 Empfangs-PDOs
1005	COB-ID SYNC message	UNSIGNED32	ID für SYNC messages
1008	Manufacturer Device Name	STRING	
1009	Hardware Version	STRING	
100A	Software Version	STRING	
100B	aktive Node-Adresse	STRING	
100C	Guard Time	UNSIGNED16	Nodeguarding Überwachungszeit
100D	Lifetimefaktor	UNSIGNED8	Lifetime Faktor für Nodeguarding
1012	COB-ID Timestamp	UNSIGNED32	COB-ID für Timestamp Telegramme
1014	COB-ID Emergency	UNSIGNED32	COB-ID für Emergency
1016	Consumer Heartbeat	UNSIGNED8	
/0	Consumer Heartbeat 1	UNSIGNED32	
/1	Consumer Heartbeat 2	UNSIGNED32	
1017	Heartbeat Time	UNSIGNED16	Zeit des Sende-Heartbeats (500ms)
1018	Identity	RECORD	
/0		UNSIGNED8	Anzahl Einträge
/1		UNSIGNED32	Vendor-ID
/2		UNSIGNED32	Product-ID
/3		UNSIGNED32	Revision
/4		UNSIGNED32	Seriennummer
1029	Error behaviour	ARRAY	
/0		UNSIGNED8	Anzahl Einträge
/1		UNSIGNED8	
/2		UNSIGNED8	
1400	RPDO1 Comm Param	ARRAY	
/0		UNSIGNED8	Anzahl Einträge
/1		UNSIGNED32	COB-ID
/2		UNSIGNED8	Transmission Type (FFh=Event-Triggered)
1401	RPDO2 Comm Param	ARRAY	
…	…		
1402	RPDO3 Comm Param	ARRAY	
…	…		
1403	RPDO4 Comm Param	ARRAY	
…	…		
1600	RPDO1 Mapping	ARRAY	
/0		UNSIGNED8	Anzahl Einträge
/1		UNSIGNED32	1. Mapping
/2		UNSIGNED32	2. Mapping
/3		UNSIGNED32	3. Mapping
/4		UNSIGNED32	4. Mapping
/5		UNSIGNED32	5. Mapping

/6		UNSIGNED32	6. Mapping
/7		UNSIGNED32	7. Mapping
/8		UNSIGNED32	8. Mapping
1601	RPDO2 Mapping	ARRAY	
	…		
1602	RPDO3 Mapping	ARRAY	
	…		
1603	RPDO4 Mapping	ARRAY	
	…		
1800	TPDO1 Comm Param	ARRAY	
/0		UNSIGNED8	Anzahl Einträge
/1		UNSIGNED32	COB-ID
/2		UNSIGNED8	Transmission Type (FFh=Event-Triggered)
1801	TPDO2 Comm Param	ARRAY	
…	…		
1802	TPDO3 Comm Param	ARRAY	
…	…		
1803	TPDO4 Comm Param	ARRAY	
…	…		
1A00	TPDO1 Mapping	ARRAY	
/0		UNSIGNED8	Anzahl Einträge
/1		UNSIGNED32	1. Mapping
/2		UNSIGNED32	2. Mapping
/3		UNSIGNED32	3. Mapping
/4		UNSIGNED32	4. Mapping
/5		UNSIGNED32	5. Mapping
/6		UNSIGNED32	6. Mapping
/7		UNSIGNED32	7. Mapping
/8		UNSIGNED32	8. Mapping
1A01	TPDO2 Mapping	ARRAY	
	…		
1A02	TPDO3 Mapping	ARRAY	
	…		
1A03	TPDO4 Mapping	ARRAY	
	…		

7.1.2 Applikationsobjekte (6000h – 6FFFh)

Die Applikationsobjekte werden kundenspezifisch angepasst.

Standardobjekte für einen IO-Slave nach Profile DS401:

SDO 6000h, Subindex 1-16, Unsigned8: 16 Byte digitale Eingänge

SDO 6200h, Subindex 1-16, Unsigned8: 16 Byte digitale Ausgänge

SDO 6401h, Subindex 1-8, Unsigned16: 8 Words analoge Eingänge

SDO 6411h, Subindex 1-8, Unsigned16: 8 Words analoge Ausgänge

7.1.3 Herstellerspezifische Objekte (2000h – 3FFFh)

Diese Objekte werden bei Bedarf kundenspezifisch erstellt.

7.2 Weiterführende Dokumentation

Internet: www.can-cia.org

CAN Specification 2.0, Part A & Part B

Notizen