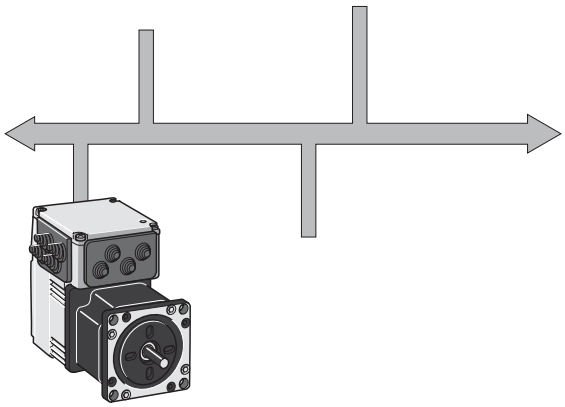
IL•1F CANopen DS301

Feldbusschnittstelle
Feldbushandbuch

V2.01, 11.2008



0198441113585, V2.01, 11.2008

www.schneider-electric.com

Schneider Electric

Wichtige Hinweise

Dieses Handbuch ist Teil des Produkts.

Lesen und befolgen Sie dieses Handbuch.

Bewahren Sie dieses Handbuch auf.

Geben Sie dieses Handbuch und alle zum Produkt gehörenden Unterlagen an alle Benutzer des Produktes weiter.

Lesen und beachten Sie besonders alle Sicherheitshinweise und das Kapitel "Bevor Sie beginnen - Sicherheitsinformationen".

Nicht alle Produkte sind in allen Ländern erhältlich.
Die Verfügbarkeit der Produkte entnehmen Sie bitte dem aktuellen Katalog.

Wir behalten uns das Recht vor ohne Ankündigung technische Änderungen vorzunehmen.

Alle Angaben sind technische Daten und keine zugesicherten Eigenschaften.

Die meisten Produktbezeichnungen sind auch ohne besondere Kennzeichnung als Warenzeichen der jeweiligen Inhaber zu betrachten.

0198441113585, V2.01, 11.2008

Inhaltsverzeichnis

Wichtige Hinweise 2

Inhaltsverzeichnis 3

Schreibkonventionen und Hinweiszeichen 7

1 Einführung 9

1.1 Dieses Handbuch 9

1.2 CAN-Bus 9

1.3 Feldbus-Geräte im CAN-Bus 10

1.4 Betriebsarten und Funktionen im Feldbus-Betrieb 10

1.5 Dokumentation und Literaturhinweise 11

2 Bevor Sie beginnen - Sicherheitsinformationen 13

3 Grundlagen 15

3.1 CANopen-Technik 15
3.1.1 CANopen-Beschreibungssprache 15
3.1.2 Kommunikationsschichten 15
3.1.3 Objekte 16
3.1.4 CANopen-Profile 18

3.2 Kommunikationsprofil 19
3.2.1 Objektverzeichnis 19
3.2.2 Kommunikationsobjekte 21
3.2.3 Kommunikationsbeziehungen 25

3.3 Servicedaten-Kommunikation 28
3.3.1 Übersicht 28
3.3.2 SDO-Datenaustausch 28
3.3.3 SDO-Nachricht 29
3.3.4 Daten schreiben und lesen 30

3.4 Prozessdaten-Kommunikation 33
3.4.1 Übersicht 33
3.4.2 PDO-Datenaustausch 34

3.5 Synchronisation 46

3.6 Netzwerk-Management-Dienste 48
3.6.1 NMT-Dienste zur Gerätekontrolle 49
3.6.2 NMT-Dienste zur Verbindungsüberwachung 50

4 Installation 53

0198441113585, V2.01, 11.2008

5 Inbetriebnahme 55

5.1 Geräteinbetriebnahme 55

5.2 Adresse und Baudrate 56

5.3 Inbetriebnahme der Feldbusverbindung 56
5.3.1 Feldbusbetrieb starten 56
5.3.2 Fehlersuche 57

5.4 SyCon CANopen Konfigurationssoftware 57
5.4.1 Neues Netzwerk erzeugen 58
5.4.2 Auswahl des CANopen Master 58
5.4.3 Einstellen der Busparameter 59
5.4.4 Auswahl und Einfügen der Knoten 60

6 Betrieb 61

6.1 Übersicht 61

6.2 Verwendung von SDO-Befehle 63
6.2.1 Parameter schreiben 63
6.2.2 Parameter lesen 64
6.2.3 Synchrone Fehler 64

6.3 Betriebszustände wechseln mit PDO4 65
6.3.1 Endstufe aktivieren und deaktivieren 65
6.3.2 "Quick Stop" auslösen 67
6.3.3 Fehler zurücksetzen 67

6.4 Beispiele zu den Betriebsarten mit PDO4 68
6.4.1 Betriebsart Punkt-zu-Punkt – Absolut-Positionierung 69
6.4.2 Betriebsart Punkt-zu-Punkt – Relativ-Positionierung 71
6.4.3 Betriebsart Geschwindigkeitsprofil 72
6.4.4 Maßsetzen 73
6.4.5 Betriebsart Referenzierung 74

6.5 Fehlersignalisierung über PDO4 75
6.5.1 Synchrone Fehler 75
6.5.2 Asynchrone Fehler 76

7 Diagnose und Fehlerbehebung 77

7.1 Fehlerdiagnose Feldbus-Kommunikation 77

7.2 Fehlerdiagnose über Feldbus 78
7.2.1 Meldungsobjekte 78
7.2.2 Meldungen über den Gerätezustand 78

7.3 CANopen-Fehlermeldungen 79
7.3.1 Fehlerregister 79
7.3.2 Fehlercode-Tabelle 79
7.3.3 SDO-Fehlernachricht ABORT 80

0198441113585, V2.01, 11.2008

8 Objektverzeichnis 81

8.1 Übersicht 81
8.1.1 Spezifikationen zu den Objekten 81
8.1.2 Objekte , Übersicht 82

8.2 Objekte des Slaves 83

9 Glossar 95

9.1 Einheiten und Umrechnungstabellen 95
9.1.1 Länge 95
9.1.2 Masse 95
9.1.3 Kraft 95
9.1.4 Leistung 95
9.1.5 Rotation 96
9.1.6 Drehmoment 96
9.1.7 Trägheitsmoment 96
9.1.8 Temperatur 96
9.1.9 Leiterquerschnitt 96

9.2 Begriffe und Abkürzungen 97

10 Stichwortverzeichnis 99

0198441113585, V2.01, 11.2008

0198441113585, V2.01, 11.2008

Schreibkonventionen und Hinweiszeichen

Arbeitsschritte

Wenn Arbeitsschritte nacheinander durchgeführt werden müssen, finden Sie folgende Darstellung:

- ■ Besondere Voraussetzungen für die nachfolgenden Arbeitsschritte
- ▶ Arbeitsschritt 1
- ◁ Besondere Reaktion auf diesen Arbeitsschritt
- ▶ Arbeitsschritt 2

Wenn zu einem Arbeitsschritt eine Reaktion angegeben ist, können Sie daran die korrekte Ausführung des Arbeitsschritts kontrollieren.

Wenn nicht anders angegeben, sind die einzelnen Handlungsschritte in der angegebenen Reihenfolge auszuführen.

Aufzählungen

Aufzählungen sind alphanumerisch oder nach der Priorität sortiert. Aufzählungen sind wie folgt aufgebaut:

- Aufzählungspunkt 1
- Aufzählungspunkt 2
  - Unterpunkt zu 2
  - Unterpunkt zu 2
- Aufzählungspunkt 3

Arbeitserleichterung

Information zur Arbeitserleichterung finden Sie bei diesem Symbol:

Hier erhalten Sie zusätzliche Informationen zur Erleichterung der Arbeit.

SI-Einheiten

SI-Einheiten sind die Originalwerte. Umgerechnete Einheiten stehen in Klammern hinter dem Originalwert und können gerundet sein.

Beispiel:
Minimaler Leiterquerschnitt: 1,5 mm^2 (AWG 14)

0198441113585, V2.01, 11.2008

0198441113585, V2.01, 11.2008

1 Einführung

1.1 Dieses Handbuch

Dieses Handbuch beschreibt die Feldbusbearbeitung für Produkte im Feldbus-Netzwerk, die über CANopen DS301 angesprochen werden.

1.2 CAN-Bus

Der CAN-Bus (CAN: **C**ontroller **A**rea **N**etwork) ist ursprünglich für die schnelle, kostengünstige Datenübertragung in der Kraftfahrzeug-Technik entwickelt worden. Inzwischen wird der CAN-Bus auch in der industriellen Automatisierung verwendet und für die Kommunikation auf Feldbusebene weiterentwickelt.

Merkmale des CAN-Bus

Der CAN-Bus ist ein standardisierter offener Bus, über den Geräte, Sensoren und Aktoren unterschiedlicher Hersteller miteinander kommunizieren. Merkmale des CAN-Bus sind

- Multi-Master-Fähigkeit

  Jeder Teilnehmer im Feldbus kann selbständig Daten senden und empfangen, ohne dabei auf eine "ordnende" Master-Funktionalität angewiesen zu sein.

- Nachrichtenorientierte Kommunikation

  Teilnehmer können ohne Neukonfiguration des Gesamtsystems in ein laufendes Netzwerk eingebunden werden. Die Adressbekanntgabe eines neuen Teilnehmers im Netzwerk ist nicht erforderlich.

- Priorisierung von Nachrichten

  Für zeitkritische Anwendungen werden Nachrichten mit hoher Priorität vorrangig übertragen.

- Restfehlerwahrscheinlichkeit

  Verschiedene Sicherungsverfahren im Netzwerk reduzieren die Wahrscheinlichkeit einer unentdeckten, fehlerhaften Datenübertragung auf unter 10^{-11}.

Übertragungstechnik

Im CAN-Bus sind mehrere Netzwerkteilnehmer über ein Buskabel miteinander verbunden. Jeder Netzwerkteilnehmer kann Nachrichten senden und empfangen. Die Daten zwischen den Netzwerkteilnehmern werden seriell übertragen.

Netzwerkteilnehmer

Beispiele für CAN-Bus-Geräte sind

- Automatisierungsgeräte, z.B. SPS
- PCs
- Ein- /Ausgangsmodule
- Antriebsverstärker
- Analysegeräte
- Sensoren und Aktoren

0198441113585, V2.01, 11.2008

1.3 Feldbus-Geräte im CAN-Bus

Verschiedene Feldbus-Geräte können im gleichen Feldbussegment betrieben werden. Über den CANopen-Bus besteht eine einheitliche Basis zum Austausch von Befehlen und Daten zwischen Slaveen und anderen Netzwerkteilnehmern.

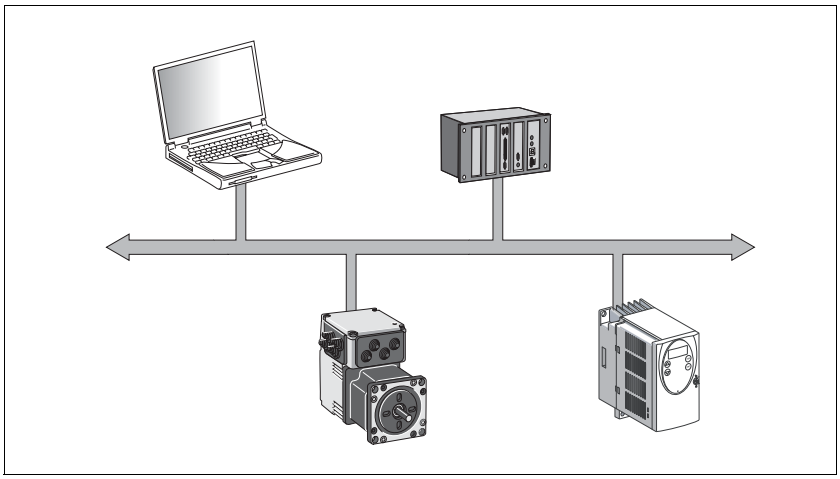


Bild 1.1 Feldbus-Geräte im Netzwerk

1.4 Betriebsarten und Funktionen im Feldbus-Betrieb

Dieses Handbuch beschreibt lediglich das Protokoll zum Slave. Die Beschreibung der Betriebsarten, Funktionen und aller Parameter finden Sie im Gerätehandbuch zum Slave in den Kapiteln "Betrieb" und "Parameter":

Betriebsarten

- Geschwindigkeitsprofil
- Punkt-zu-Punkt
- Referenzierung
- Manuellfahrt

Funktionen

- Definition der Drehrichtung
- Fahrprofilerzeugung
- Quick Stop
- Schnelle Positionserfassung

Einstellmöglichkeiten

Über den Feldbus lassen sich folgende Einstellungen vornehmen:

- Parameter lesen und schreiben
- Ein- und Ausgänge der 24-V-Signalschnittstelle überwachen
- Diagnose und Fehlerüberwachungsfunktionen aktivieren

Feldbusbetrieb

0198441113585, V2.01, 11.2008

1.5 Dokumentation und Literaturhinweise

Handbücher

Zu dem Produkt gehört außer diesem Feldbushandbuch noch folgendes Handbuch:

- **Produkthandbuch**, beschreibt die technischen Daten, die Installation, die Inbetriebnahme sowie sämtliche Betriebsarten und Funktionen.

Interessengemeinschaft CAN

CiA - CAN in Automation
Am Weichselgarten 26
D-91058 Erlangen
http://www.can-cia.org/

CANopen Standards

- CiA Standard 301 (DS301)
  CANopen application layer and communication profile
  V4.02, February 2002
- CiA Standard 402 (DSP402)
  Device profile for drives and motion control
  V2.0, July 2002
- ISO/DIS 11898: Controller Area Network (CAN) for high speed communication;1993
- EN 50325-4: Industrial communications subsystem based on ISO 11898 for controller device interfaces (CANopen); 2002

Literaturhinweise

Controller Area Network
Konrad Etschberger, Carl Hanser Verlag
ISBN 3-446-19431-2

0198441113585, V2.01, 11.2008

0198441113585, V2.01, 11.2008

2 Bevor Sie beginnen - Sicherheitsinformationen

Die in diesem Handbuch beschriebenen Informationen sind eine Ergänzung zum Produkthandbuch. Lesen Sie bevor Sie beginnen sorgfältig das Produkthandbuch.

0198441113585, V2.01, 11.2008

0198441113585, V2.01, 11.2008

3 Grundlagen

3.1 CANopen-Technik

3.1.1 CANopen-Beschreibungssprache

CANopen ist eine geräte- und herstellerunabhängige Beschreibungssprache zur Kommunikation im CAN-Bus. Über CANopen besteht eine einheitliche Basis zum Austausch von Befehlen und Daten zwischen CAN-Bus-Teilnehmern.

3.1.2 Kommunikationsschichten

CANopen nutzt die CAN-Bus-Technik zur Datenkommunikation.

CANopen setzt in Anlehnung an das ISO-OSI-Schichtenmodell auf die Netzwerk-Basisdienste der Datenkommunikation auf. 3 Schichten ermöglichen die Datenkommunikation im CAN-Bus.

- Physical Layer
- Data Link Layer
- Application Layer

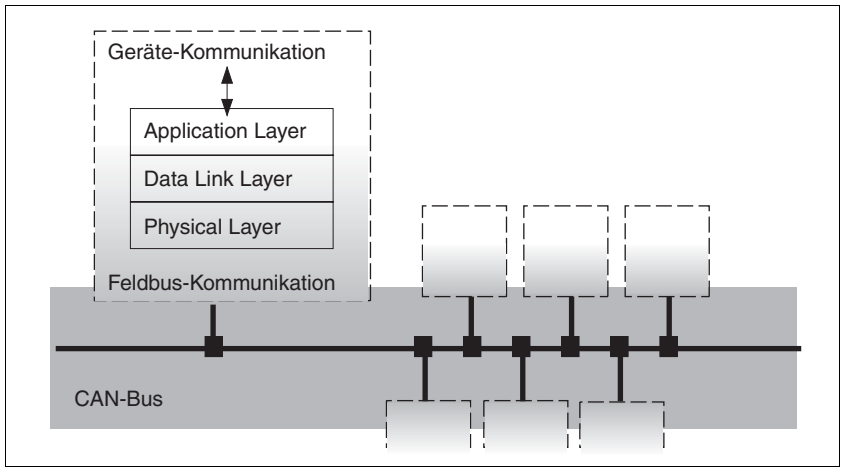


Bild 3.1 CANopen-Schichtenmodell

Physical Layer Die physikalische Schicht definiert die elektrischen Eigenschaften des CAN-Busses wie Steckverbinder, Kabellänge und -eigenschaften sowie Bit-Kodierung und Bit-Timing.

Data Link Layer Die Datensicherungsschicht sorgt für die Verbindung zwischen den Netzwerkteilnehmern. Sie teilt einzelnen Datenpaketen ihre Prioritäten zu und führt Fehlerüberwachung und -korrekturen durch.

Application Layer Die Anwendungsschicht nutzt Kommunikationsobjekte (COB) zum Datenaustausch zwischen den einzelnen Teilnehmern. Kommunikationsobjekte sind elementarer Bestandteil zur Erstellung einer CANopen-Anwendung.

0198441113585, V2.01, 11.2008

3.1.3 Objekte

Alle Abläufe unter CANopen werden über Objekte ausgeführt. Objekte übernehmen verschiedene Aufgaben, sie sorgen als Kommunikationsobjekte für den Datentransport zum Feldbus, steuern den Verbindungsaufbau oder überwachen die Netzwerkteilnehmer. Stehen Objekte in direkter Verbindung zum Gerät (gerätespezifische Objekte) können über diese die Gerätefunktionen genutzt und verändert werden.

Objektverzeichnis

Zentrale Verbindung für alle Objekte ist das Objektverzeichnis jedes Netzwerkteilnehmers. Hier finden andere Teilnehmer alle Objekte, über die sie mit dem Gerät in Verbindung treten.

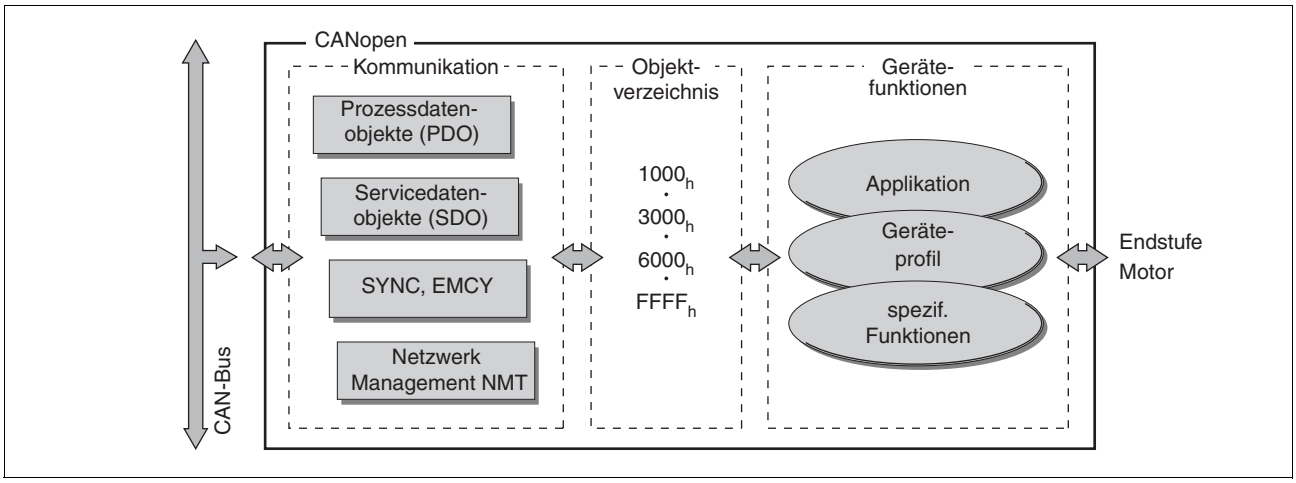


Bild 3.2 Gerätemodell mit Objektverzeichnis

Eingetragen sind Objekte zur Beschreibung der Datentypen und zur Ausführung der Kommunikationsaufgaben und Gerätefunktionen unter CANopen.

Objekt-Index

Jedes Objekt wird über einen 16 Bit-Index, der als vierstellige Hexadezimalzahl dargestellt wird, adressiert. Die Objekte sind in Gruppen im Objektverzeichnis angeordnet.

Index (hex)	Objektgruppen	Vom Antrieb unterstützt
0000_h	reserviert	Nein
0001_h-$009F_h$	Statische und komplexe Datentypen	Nein
$00A0_h$-$0FFF_h$	reserviert	Nein
1000_h-$1FFF_h$	Kommunikationsprofil, standardisiert im DS301	Ja
2000_h-$5FFF_h$	Herstellerspezifische Geräteprofile	Ja
6000_h-$9FFF_h$	Standardisierte Geräteprofile, z B. im DS402	Nein
$A000_h$-$FFFF_h$	reserviert	Nein

Tabelle 3.1 Objekt-Index

Eine Liste der CANopen Objekte finden Sie ab Seite 81.

0198441113585, V2.01, 11.2008

Objektgruppe Datentypen Mit den Datentypen erhalten die Nachrichten, die als Bitstrom über das Netzwerk laufen, für Sender und Empfänger die gleiche Bedeutung. Datentypen werden über die Objekte der Datentypen vereinbart.

Objektgruppen der Profile CANopen-Objekte übernehmen verschiedene Aufgaben im Feldbus-Betrieb. Profile stellen die Objekte entsprechend ihrer vorgegebenen Aufgaben zusammen.

0198441113585, V2.01, 11.2008

3.1.4 CANopen-Profile

Standardisierte Profile

Standardisierte Profile beschreiben Objekte, die auf verschiedenen Geräten ohne zusätzliche Anpassung eingesetzt werden. Die Interessengemeinschaft CAN in Automation e. V. (CiA) hat verschiedene Profile standardisiert. Dazu gehören:

- das Kommunikationsprofil DS301
- das Geräteprofil DSP402

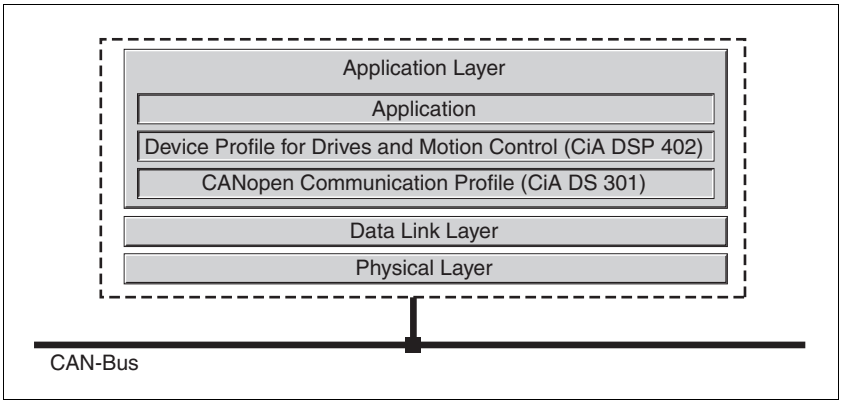


Bild 3.3 CANopen-Referenzmodell

Kommunikationsprofil DS301

Das Kommunikationsprofil DS301 bildet die Schnittstelle zwischen Geräteprofilen und CAN-Bus. Es wurde 1995 unter dem Namen DS301 spezifiziert und definiert einheitliche Standards für den gemeinsamen Datenaustausch zwischen unterschiedlichen Gerätetypen unter CANopen.

Die Objekte des Kommunikationsprofils übernehmen im Gerät die Aufgaben des Daten- und Parameteraustauschs mit anderen Netzwerkteilnehmern und initialisieren, steuern und überwachen das Gerät im Netzwerk.

Objekte des Kommunikationsprofils sind:

- Prozessdaten-Objekte (**P**rocess **D**ata **O**bjects = PDO)
- Servicedaten-Objekte (**S**ervice **D**ata **O**bjects = SDO)
- Objekte mit speziellen Funktionen zur Synchronisation `SYNC` und zur Fehlermeldung und -reaktion `EMCY`
- Objekte des Netzwerk-Managements NMT zur Initialisierung, Fehlerüberwachung und zur Statusüberwachung des Geräts.

Geräteprofil DSP402

Das Geräteprofil DSP402 beschreibt standardisierte Objekte für die Positionierung, Überwachung und Einstellung von Antrieben. Aufgaben der Objekte sind:

- Gerätekontrolle und Statusüberwachung (Device Control)
- standardisierte Parametrierung
- Wechsel, Kontrolle und Ausführung von Betriebsarten

WICHTIG: Der integrierte Antrieb unterstützt nicht das Geräteprofil CiA 402.

0198441113585, V2.01, 11.2008

Herstellerspezifische Profile

Mit Objekten standardisierter Geräteprofile werden die Basisfunktionen eines Gerätes genutzt. Der gesamte Funktionsumfang steht erst mit herstellerspezifischen Geräteprofilen zur Verfügung. In ihnen sind die Objekte definiert, mit denen unter CANopen die speziellen Funktionen eines Gerätes genutzt werden.

3.2 Kommunikationsprofil

CANopen führt die Kommunikation zwischen den Netzwerkteilnehmern über Objektverzeichnisse und Objekte aus. Mit Prozessdaten-Objekten (PDO) und Servicedaten-Objekten (SDO) kann ein Netzwerkteilnehmer die Objektdaten aus dem Objektverzeichnis eines anderen Teilnehmers anfordern und, wenn zulässig, geänderte Werte wieder zurückschreiben.

Durch den Zugriff auf die Objekte der Netzwerkteilnehmer lassen sich

- Parameterwerte austauschen
- Fahrfunktionen einzelner CAN-Bus-Teilnehmer starten
- Statusinformationen abfragen

3.2.1 Objektverzeichnis

Jeder CANopen-Teilnehmer verwaltet ein Objektverzeichnis, in dem alle Objekte für die Kommunikation aufgeführt sind.

Index, Subindex

Die Objekte werden im Objektverzeichnis über einen 16 Bit langen Index adressiert. Einer oder mehrere 8 Bit lange Subindexeinträge zu jedem Objekt geben einzelne Datenfelder im Objekt an. Index und Subindex werden in hexadezimaler Schreibweise angegeben, erkennbar an dem angehängten "$_h$".

Das folgende Beispiel zeigt Index- und Subindex-Einträge für das Objekt `receive PDO4 mapping,` 1603_h, zur Kennzeichnung für das Mapping in `R_PDO4`.

Index	Subindex	Objekt	Bedeutung
1603_h	00_h	`Number of elements`	Anzahl Subindices
1603_h	01_h	`1st mapped object R_PDO4`	Erstes Objekt für das Mapping in R_PDO4
1603_h	02_h	`2nd mapped object R_PDO4`	Zweites Objekt für das Mapping in R_PDO4
1603_h	03_h	`3rd mapped object R_PDO4`	Drittes Objekt für das Mapping in R_PDO4

Tabelle 3.2 Beispiele für Index- und Subindexeinträge

0198441113585, V2.01, 11.2008

Verzeichnisaufbau

Die Objekte sind im Objektverzeichnis nach Index-Werten sortiert. Tabelle 3.3 zeigt die Index-Bereiche des Objektverzeichnisses nach CANopen-Vereinbarung.

Indexbereich (hex)	Objektgruppen	Vom Antrieb unterstützt
0000_h	Reserviert	Nein
0001_h-$001F_h$	Statische Datentypen	Nein
0020_h-$003F_h$	Komplexe Datentypen	Nein
0040_h-$005F_h$	Herstellerspezifische Datentypen	Nein
0060_h-$007F_h$	Statische Datentypen zu den Geräteprofilen	Nein
0080_h-$009F_h$	Komplexe Datentypen zu den Geräteprofilen	Nein
$00A0_h$-$0FFF_h$	reserviert	Nein
1000_h-$1FFF_h$	Kommunikationsprofil	Ja
2000_h-$5FFF_h$	Herstellerspezifische Profile	Ja
6000_h-$9FFF_h$	Standardisierte Geräteprofile	Nein
$A000_h$-$FFFF_h$	reserviert	Nein

Tabelle 3.3 Index-Bereiche des Objektverzeichnisses

Objektbeschreibungen im Handbuch

Für die CANopen-Programmierung mit einem integrierten Antrieb werden die Objekte der folgenden Objektgruppen differenziert beschrieben:

- $1xxx_h$-Objekte: Kommunikationsobjekte in diesem Kapitel
- $3xxx_h$-Objekte: Herstellerspezifische Objekte, soweit sie zur Steuerung des integrierten Antriebs erforderlich sind.

Alle Betriebsarten und Funktionen des Antriebs werden durch herstellerspezifische Objekte gesteuert. Diese Funktionen und Objekte sind in der jeweiligen Gerätedokumentation beschrieben.

Die herstellerspezifischen Objekte werden ab Indexbereich 3000_h abgelegt. Um von den in der Gerätedokumentation angegebenen Indices auf den CAN-Index zu kommen, muss lediglich 3000_h addiert werden.

Beispiel:

Das Steuerwort für Zustandswechsel hat den Index 28 und den Subindex 1. Daraus ergibt sich im CAN-Protokoll Index $301C_h$ ($3000_h + 1C_h$[= 28_d]) und Subindex 1.

0198441113585, V2.01, 11.2008

3.2.2 Kommunikationsobjekte

Übersicht Die Kommunikationsobjekte sind mit dem CANopen-Kommunikationsprofil DS301 standardisiert. Ihren Aufgaben entsprechend können die Objekte in 4 Gruppen unterteilt werden:

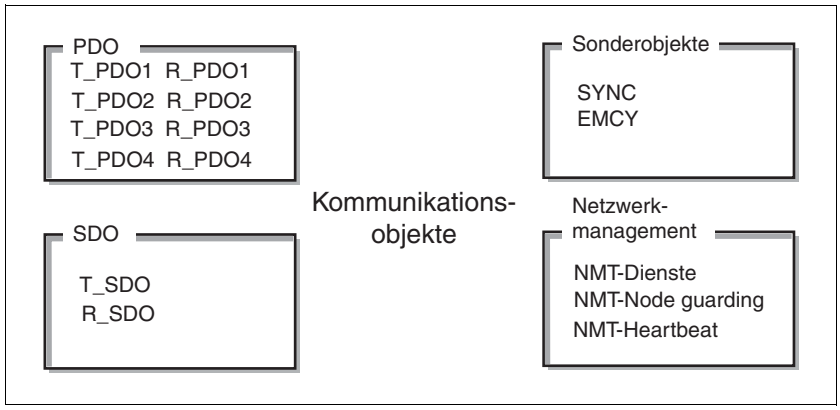


Bild 3.4 Kommunikationsobjekte, aus der Sicht des Teilnehmers gilt: T_..: "Transmit" - senden, R_..: "Receive" - empfangen

- Prozessdaten-Objekte PDO (process data object) zur Echtzeit-Übertragung von Prozessdaten
- Servicedaten-Objekte SDO (service data object) für den Schreib- und Lesezugriff auf das Objektverzeichnis
- Objekte zur Steuerung von CAN-Nachrichten:
  - SYNC-Objekt (synchronization object) zur Synchronisation von Netzwerkteilnehmern
  - EMCY-Objekt (emergency object), zur Fehleranzeige eines Gerätes oder seiner Peripherie.
- Dienste des Netzwerk-Managements:
  - NMT Dienste zur Initialisierung und Netzwerksteuerung (NMT: network management)
  - NMT Node Guarding zur Überwachung der Netzwerkteilnehmer
  - NMT Heartbeat zur Überwachung der Netzwerkteilnehmer

0198441113585, V2.01, 11.2008

CAN-Nachricht

Auf dem CAN-Bus werden Daten in CAN-Nachrichten ausgetauscht. Eine CAN-Nachricht überträgt das Kommunikationsobjekt und eine Vielzahl Verwaltungs- und Steuerinformationen, die für eine verlust- und fehlerfreie Übertragung der Daten sorgen.

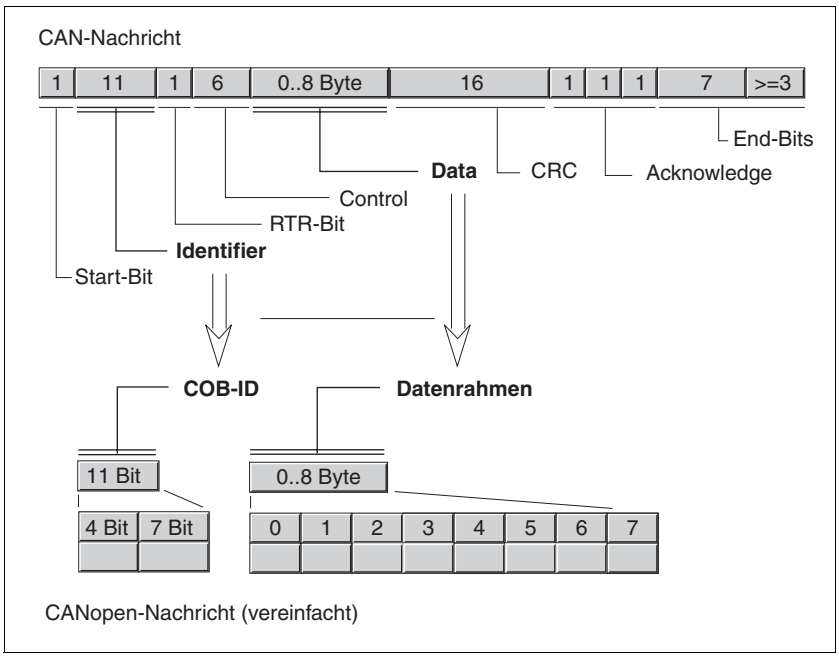


Bild 3.5 CAN-Nachricht und vereinfacht dargestellte CANopen-Nachricht

CANopen-Nachricht

Für die Arbeit mit CANopen-Objekten und für den Datenaustausch kann die CAN-Nachricht vereinfacht dargestellt werden, da die meisten Bits zur Fehlerkorrektur benutzt werden. Diese Bits werden automatisch von der Datensicherungsschicht, dem Data link layer des OSI-Schichtenmodells, aus der empfangenen Nachricht entfernt und vor dem Senden einer Nachricht eingefügt.

Die beiden Bitfelder "Identifier" und "Data" bilden die vereinfachte CANopen-Nachricht. Der "Identifier" entspricht der "COB-ID" und das "Data"-Feld dem maximal 8 Byte großen Datenrahmen einer CANopen-Nachricht.

0198441113585, V2.01, 11.2008

COB-ID

Die COB-ID (**C**ommunication **OB**ject**-Id**entifier) übernimmt 2 Aufgaben zur Steuerung von Kommunikationsobjekten:

- Busarbitrierung: Festlegung der Übertragungsprioritäten
- Identifikation von Kommunikationsobjekten

Für die CAN-Kommunikation ist ein 11-Bit COB-IDentifier nach der Spezifikation CAN 3.0A festgelegt, der sich aus 2 Teilen zusammensetzt:

- Funktionscode (function-code), 4 Bit groß
- Knotenadresse (Node-ID), 7 Bit groß.

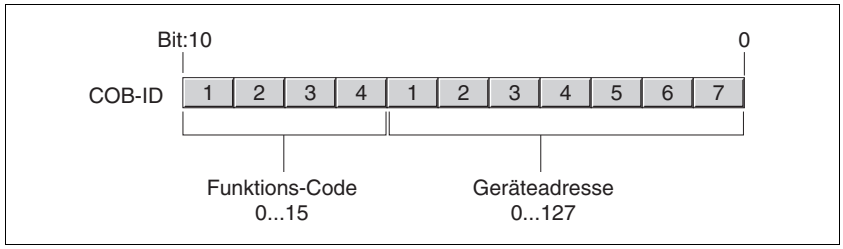


Bild 3.6 COB-ID mit Funktionscode und Knotenadresse

COB-IDs der Kommunikationsobjekte

Die folgende Tabelle zeigt die COB-IDs aller Kommunikationsobjekte entsprechend der Werkseinstellung. Die Spalte "Index der Objektparameter" gibt den Index spezieller Objekte an, mit denen die Einstellungen der Kommunikationsobjekte per SDO gelesen oder geändert werden können.

Kommunikationsobjekt	Funktions-code	Knotenadresse Node-ID [1...127]	COB-IDdezimal (hexadezimal)	Index der Objekt-Parameter
NMT Start/Stop Service	0 0 0 0	0 0 0 0 0 0 0	0 (0_h)	-
SYNC-Objekt	0 0 0 1	0 0 0 0 0 0 0	128 (80_h)	1005_h....1007_h
EMCY-Objekt	0 0 0 1	x x x x x x x	128 (80_h) + Node-ID	1014_h, 1015_h
T_PDO1 1)	0 0 1 1	x x x x x x x	384 (180_h) + Node-ID	1800_h
R_PDO1 1)	0 1 0 0	x x x x x x x	512 (200_h) + Node-ID	1400_h
T_PDO2 1)	0 1 0 1	x x x x x x x	640 (280_h) + Node-ID	1801_h
R_PDO2 1)	0 1 1 0	x x x x x x x	768 (300_h) + Node-ID	1401_h
T_PDO3 1)	0 1 1 1	x x x x x x x	896 (380_h) + Node-ID	1802_h
R_PDO3 1)	1 0 0 0	x x x x x x x	1024 (400_h) + Node-ID	1402_h
T_PDO4	1 0 0 1	x x x x x x x	1152 (480_h) + Node-ID	1803_h
R_PDO4	1 0 1 0	x x x x x x x	1280 (500_h) + Node-ID	1403_h
T_SDO	1 0 1 1	x x x x x x x	1408 (580_h) + Node-ID	-
R_SDO	1 1 0 0	x x x x x x x	1536 (600_h) + Node-ID	-
NMT error control	1 1 1 0	x x x x x x x	1792 (700_h) + Node-ID	
LMT Services 1)	1 1 1 1	1 1 0 0 1 0 x	2020 ($7E4_h$), 2021 ($7E5_h$)	
NMT Identify Service 1)	1 1 1 1	1 1 0 0 1 1 0	2022 ($7E6_h$)	
DBT Services 1)	1 1 1 1	1 1 0 0 x x x	2023 ($7E7_h$), 2024 ($7F8_h$)	
NMT Services 1)	1 1 1 1	1 1 0 1 0 0 x	2025 ($7E9_h$), 2026 ($7EA_h$)	

1) vom Gerät nicht unterstützt

Tabelle 3.4 COB-IDs aller Kommunikationsobjekte

0198441113585, V2.01, 11.2008

COB-IDs von PDOs können bei Bedarf geändert werden. Das Vergabeschema für COB-IDs gibt nur eine Grundeinstellung vor.

Funktionscode Der Funktionscode klassifiziert die Kommunikationsobjekte. Da die Bits des Funktionscodes in der COB-ID höher signifikant sind, steuert der Funktionscode gleichzeitig die Übertragungsprioritäten: Objekte mit kleinem Funktionscode werden mit hoher Priorität übertragen. So wird z.B. bei gleichzeitigem Buszugriff ein Objekt mit dem Funktionscode "1" vor einem Objekt mit dem Funktionscode "3" übertragen.

Knotenadresse Jeder Netzwerkteilnehmer wird vor dem Netzwerkbetrieb konfiguriert. Dabei erhält er eine eindeutige, 7 Bit lange Knotenadresse (Node-ID) zwischen 1 (01_h) und 127 ($7F_h$). Die Geräteadresse "0" ist "broadcast"-Sendungen vorbehalten, mit der Nachrichten gleichzeitig an alle Teilnehmer gesendet werden.

Beispiel Wahl einer COB-ID

Für ein Gerät mit der Knotenadresse 5 ist die COB-ID des Kommunikationsobjekts T_PDO1:

384+Node-ID = 384 (180_h) + 5 = 389 (185_h).

Datenrahmen Der Datenrahmen der CANopen-Nachricht kann bis zu 8 Byte Daten aufnehmen. Neben dem Datenrahmen für SDOs und PDOs sind im CANopen-Profil spezielle Rahmentypen festgelegt:

- Fehlerdatenrahmen
- Remote-Datenrahmen zur Anforderung einer Nachricht

Die Datenrahmen werden mit den jeweiligen Kommunikationsobjekten beschrieben.

0198441113585, V2.01, 11.2008

3.2.3 Kommunikationsbeziehungen

CANopen nutzt 3 Beziehungen für die Kommunikation zwischen Netzwerkteilnehmern:

- Master-Slave-Beziehung
- Client-Server-Beziehung
- Producer-Consumer-Beziehung

Master-Slave-Beziehung

Ein Master im Netzwerk steuert den Nachrichtenverkehr. Ein Slave antwortet nur auf Anforderung des Masters.

Die Master-Slave-Beziehung wird mit Netzwerkmanagement-Objekten eingesetzt, um einen kontrollierten Netzwerkstart zu ermöglichen und um die Verbindung von Teilnehmern zu überwachen.

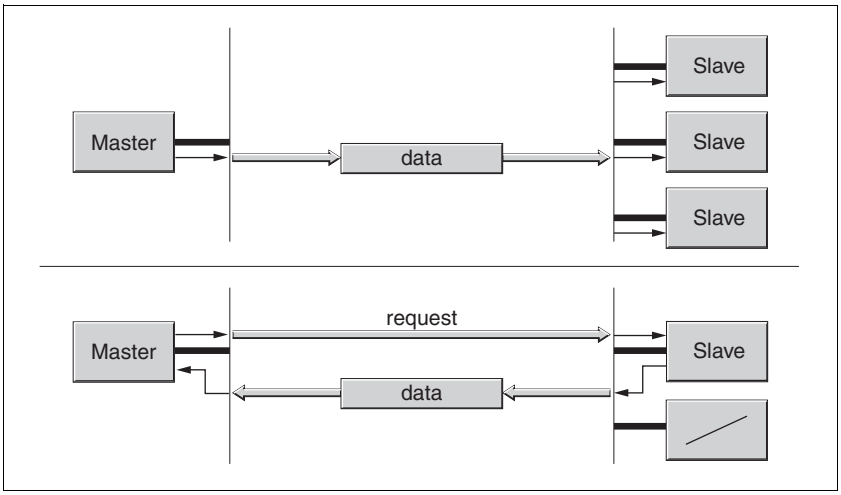


Bild 3.7 Master-Slave-Beziehungen

Der Nachrichtenaustausch kann unbestätigt und bestätigt ausgeführt werden. Sendet der Master eine unbestätigte CAN-Nachricht, kann sie von einem, von mehreren oder von keinem Slave empfangen werden.

Damit die Nachricht bestätigt wird, fordert der Master eine Nachricht von einem bestimmten Slave an, der dann mit den gewünschten Daten antwortet.

0198441113585, V2.01, 11.2008

Client-Server-Beziehung

Eine Client-Server-Beziehung wird zwischen 2 Teilnehmern aufgebaut. Der "Server" (engl. to serve: dienen) ist der Teilnehmer, dessen Objektliste während des Datenaustauschs benutzt wird. Der "Client" (Client: Kunde) adressiert und startet den Nachrichtenaustausch und erwartet eine Bestätigung vom Server.

Eine Client-Server-Beziehung wird mit SDOs eingesetzt, um Konfigurationsdaten und lange Nachrichten zu übertragen.

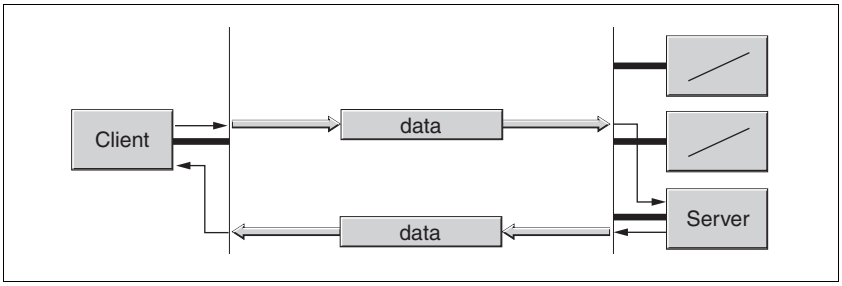


Bild 3.8 Client-Server-Beziehung

Der Client adressiert und überträgt eine CAN-Nachricht an einen Server. Der Server wertet die Nachricht aus und schickt als Bestätigung die Antwortdaten.

0198441113585, V2.01, 11.2008

Producer-Consumer-Beziehung

Die Producer-Consumer-Beziehung wird für den Nachrichtenaustausch von Prozessdaten eingesetzt, da die Beziehung einen schnellen Datenaustausch ohne Verwaltungsdaten ermöglicht.

Ein "Producer" (engl.: Hersteller) sendet Daten, ein "Consumer" (engl.: Verbraucher) empfängt sie.

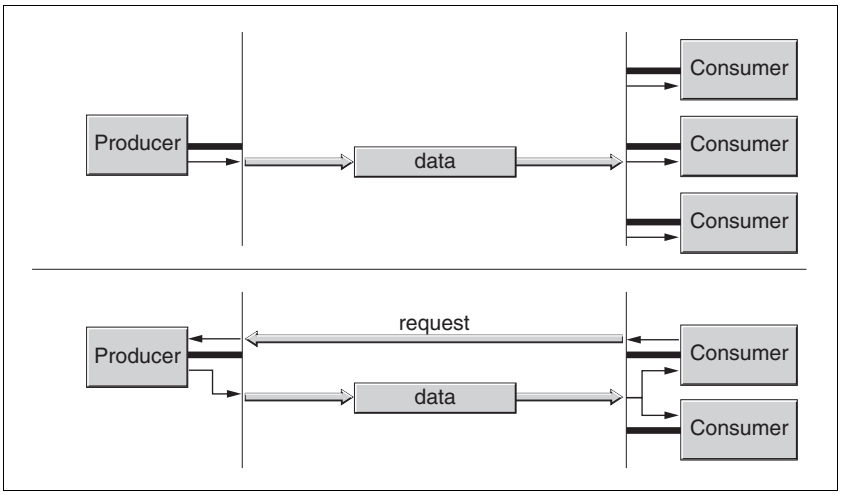


Bild 3.9 Producer-Consumer-Beziehungen

Der Producer sendet eine Nachricht, die von einem oder von mehreren Netzwerkteilnehmern empfangen werden kann. Eine Empfangsbestätigung erhält der Producer nicht. Ausgelöst werden kann die Nachrichtensendung

- über ein internes Ereignis, z.B. "Zielposition erreicht"
- über das Synchronisationsobjekt SYNC
- durch die Anforderung eines Consumers

Einzelheiten zur Funktion der Producer-Consumer-Beziehung und zur Anforderung von Nachrichten finden Sie im Kapitel 3.4 "Prozessdaten-Kommunikation".

0198441113585, V2.01, 11.2008

3.3 Servicedaten-Kommunikation

3.3.1 Übersicht

Mit Servicedaten-Objekten (SDO: **S**ervice **D**ata **O**bject) kann über Index und Subindex auf die Einträge eines Objektverzeichnisses zugegriffen werden. Die Werte der Objekte können gelesen und, wenn zulässig, auch geändert werden.

Jeder Netzwerkteilnehmer hat mindestens ein Server-SDO, um auf Lese- oder Schreibanforderungen eines anderen Teilnehmers reagieren zu können. Ein Client-SDO ist nur notwendig, um SDO-Nachrichten aus dem Objektverzeichnis eines anderen Teilnehmers anzufordern oder dort zu ändern.

Mit dem T_SDO eines SDO Client wird die Anforderung zum Datenaustausch gesendet, mit dem R_SDO empfangen. Der Datenrahmen eines SDO beträgt 8 Byte.

SDOs haben eine höhere COB-ID als PDOs und werden deshalb mit niedrigerer Priorität auf dem CAN-Bus übertragen.

3.3.2 SDO-Datenaustausch

Ein Servicedatenobjekt SDO überträgt Parameterdaten zwischen 2 Teilnehmern. Der Datenaustausch folgt der Client-Server-Beziehung. Server ist der Teilnehmer, auf dessen Objektverzeichnis sich eine SDO-Nachricht bezieht.

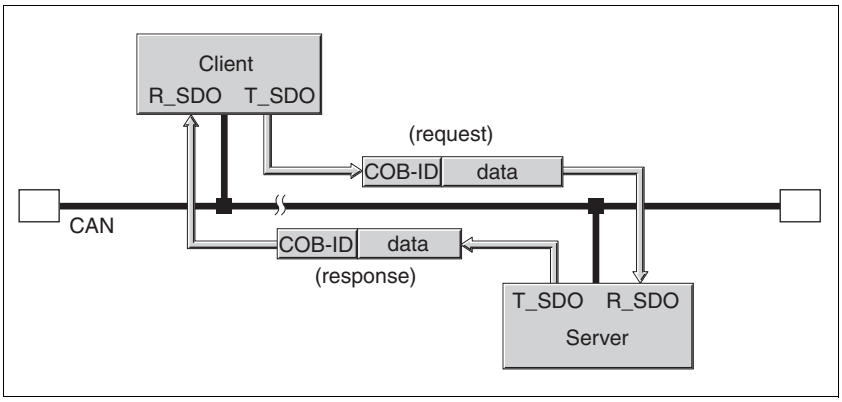


Bild 3.10 SDO-Nachrichtenaustausch mit Anfrage (request) und Antwort (response)

Nachrichtentypen

Die Client-Server-Kommunikation wird vom Client ausgelöst, um Parameterwerte an den Server zu übermitteln oder vom Server zu holen. In beiden Fällen startet der Client die Kommunikation mit einer Anfrage (request) und erhält vom Server eine Bestätigung (response).

0198441113585, V2.01, 11.2008

3.3.3 SDO-Nachricht

Eine SDO-Nachricht besteht vereinfacht aus der COB-ID und dem SDO-Datenrahmen, in dem bis zu 4 Byte Daten übertragen werden können. Längere Datenfolgen werden über ein spezielles Protokoll auf mehrere SDO-Nachrichten verteilt.

Das Gerät überträgt SDOs mit bis zu 4 Byte Datenlänge (Data). Größere Datenmengen wie z.B. 8 Byte-Werte des Datentyps "Visible String 8" können auf mehrere SDOs verteilt und nacheinander in 7 Byte-Blöcken übermittelt werden.

Beispiel Folgendes Bild zeigt ein Beispiel einer SDO-Nachricht.

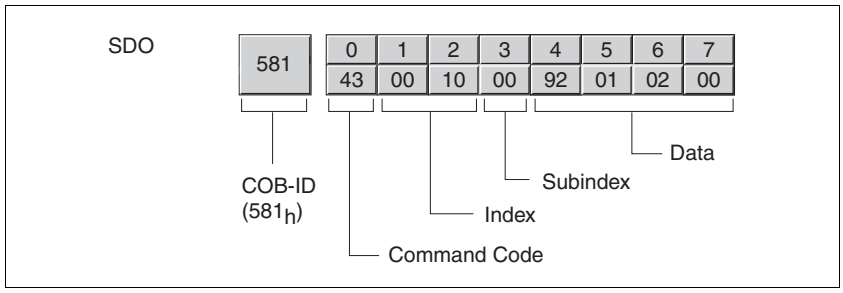


Bild 3.11 SDO-Nachricht, Beispiel

COB-ID und Datenrahmen R_SDO und T_SDO haben unterschiedliche COB-IDs.
Der Datenrahmen einer SDO-Nachricht besteht aus:

- Befehls-Code (ccd: command-code), in dem der SDO-Nachrichtentyp und die Datenlänge des übermittelten Werts verschlüsselt sind
- Index und Subindex, die auf das Objekt zeigen, dessen Daten mit der SDO-Nachricht transportiert werden
- Daten (Data), die bis zu 4 Byte umfassen

Auswertung von Zahlenwerten Index und Daten werden linksbündig im Intel-Format übertragen. Enthält das SDO Zahlenwerte über 1 Byte Länge, müssen die Daten vor und nach einer Übertragung byteweise umgestellt werden.

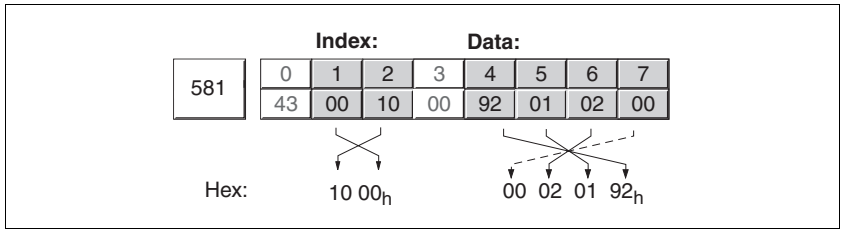


Bild 3.12 Umstellung von Zahlenwerten größer 1 Byte

0198441113585, V2.01, 11.2008

3.3.4 Daten schreiben und lesen

Daten schreiben

Der Client startet eine Schreib-Anforderung (write request) mit Übermittlung von Index, Subindex, Datenlänge und Wert.

Der Server sendet eine Bestätigung, ob die Daten korrekt verarbeitet wurden. Die Bestätigung enthält den gleichen Index und Subindex, aber keine Daten.

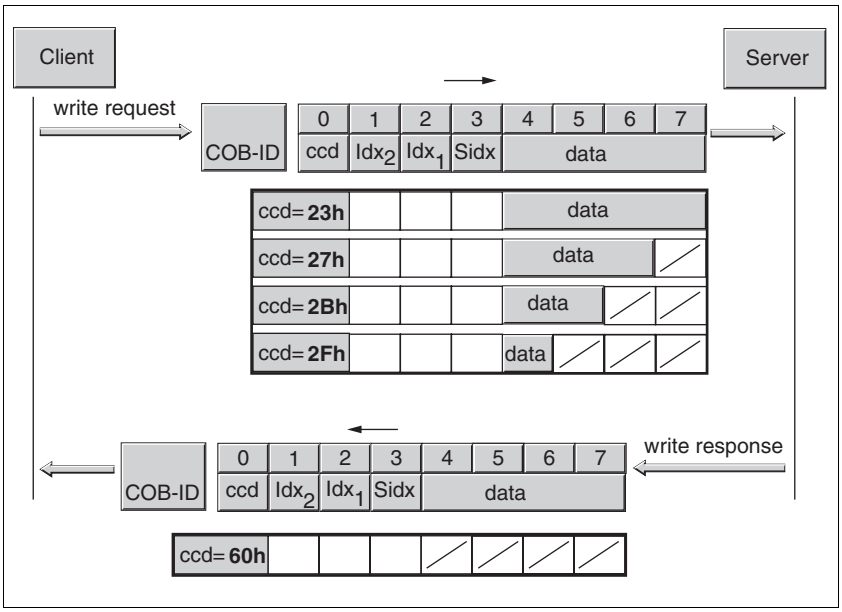


Bild 3.13 Parameterwert schreiben

Nicht genutzte Byte im Datenfeld sind in der Grafik mit einem Schrägstrich gekennzeichnet. Ihr Inhalt ist nicht definiert.

ccd-Codierung

Folgende Tabelle zeigt den Befehls-Code, um Parameterwerte zu schreiben. Er ist abhängig vom Nachrichtentyp und von der übertragenen Datenlänge.

Nachrichtentyp	**Genutzte Datenlänge**				
	4 Byte	**3 Byte**	**2 Byte**	**1 Byte**	
write request	23_h	27_h	$2B_h$	$2F_h$	Parameter senden
write response	60_h	60_h	60_h	60_h	Bestätigung
error response	80_h	80_h	80_h	80_h	Fehler

Tabelle 3.5 Befehls-Code zum Schreiben von Parameterwerten

0198441113585, V2.01, 11.2008

Daten lesen

Der Client startet eine Lese-Anforderung (read request) mit der Übermittlung von Index und Subindex, die auf das Objekt oder auf den Objektwert zeigen, dessen Wert er auslesen möchte.

Der Server bestätigt die Anfrage mit den gewünschten Daten. Die SDO-Antwort enthält den gleichen Index und Subindex. Die Länge der Antwortdaten ist im Befehls-Code "ccd" angegeben.

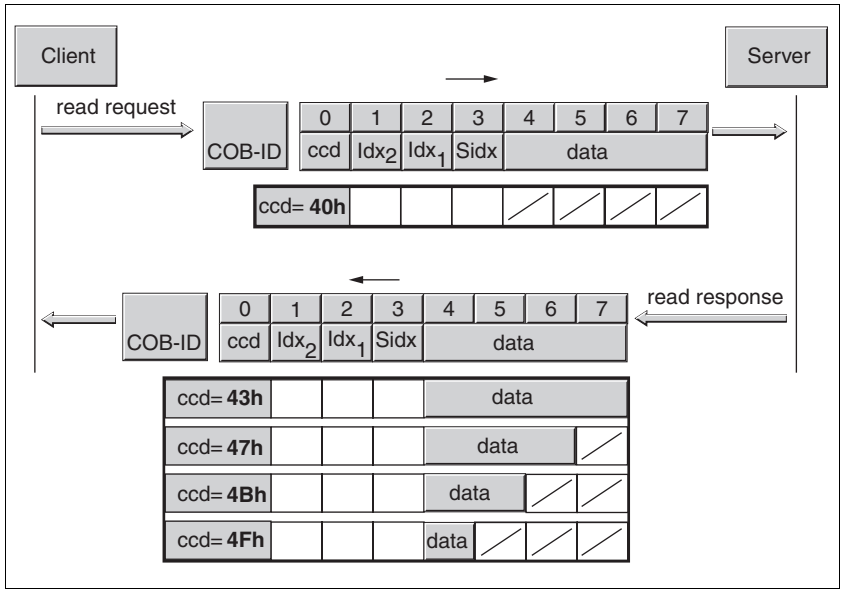


Bild 3.14 Parameterwert lesen

Nicht genutzte Bytes im Datenfeld sind in der Grafik mit einem Schrägstrich gekennzeichnet. Ihr Inhalt ist nicht definiert.

ccd-Kodierung

Folgende Tabelle zeigt den Befehls-Code, um einen Lesewert zu übertragen. Er ist abhängig vom Nachrichtentyp und der übertragenen Datenlänge.

Nachrichtentyp	**Genutzte Datenlänge**				
	4 Byte	**3 Byte**	**2 Byte**	**1 Byte**	
read request	40_h	40_h	40_h	40_h	Lesewert anfordern
read response	43_h	47_h	$4B_h$	$4F_h$	Lesewert rücksenden
error response	80_h	80_h	80_h	80_h	Fehler

Tabelle 3.6 Befehls-Code zur Übertragung eines Lesewerts

0198441113585, V2.01, 11.2008

Fehlerantwort Konnte eine Nachricht nicht fehlerfrei ausgewertet werden, sendet der Server eine Fehlermeldung. Details zur Auswertung der Fehlermeldung finden Sie im Kapitel 7 "Diagnose und Fehlerbehebung".

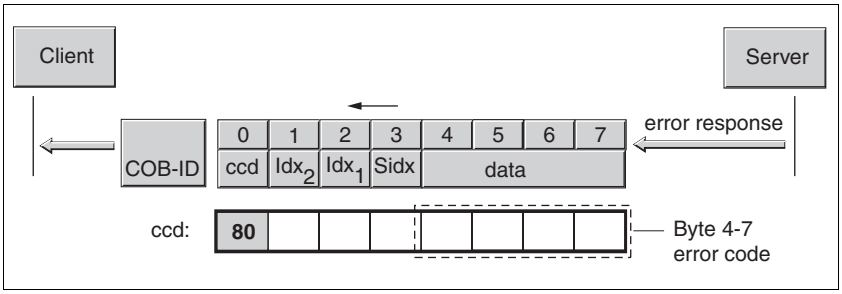


Bild 3.15 Antwort mit Fehlermeldung (error response)

0198441113585, V2.01, 11.2008

3.4 Prozessdaten-Kommunikation

3.4.1 Übersicht

Dieses Kapitel beschreibt den Informationsfluss aus Sicht des integrierten Antriebs in Übereinstimmung mit der CiA-Norm DS301. Die Bezeichnung "receive" bedeutet also einen Datenfluss vom Master zum integrierten Antrieb, wobei mit "transmit" ein Datenfluss vom integrierten Antrieb zum Master gemeint ist.

Prozessdaten-Objekte (PDO: **P**rocess **D**ata **O**bject) werden für den Echtzeit-Datenaustausch von Prozessdaten wie Ist- und Sollposition oder den Betriebszustand des Gerätes genutzt. Die Übertragung kann sehr schnell ausgeführt werden, weil sie ohne zusätzliche Verwaltungsdaten übermittelt wird und vom Empfänger nicht bestätigt werden muss.

Auch die flexible Datenlänge einer PDO-Nachricht erhöht den Datendurchsatz. Eine PDO-Nachricht kann bis zu 8 Byte Daten übertragen. Sind nur 2 Byte belegt, werden auch nur 2 Datenbyte übertragen.

Die Länge einer PDO-Nachricht und Belegung der Datenfelder wird über das PDO-Mapping festgelegt. Informationen hierzu finden Sie im Kapitel 3.4.2.1 "Dynamisches und statisches PDO-Mapping".

PDO-Nachrichten können zwischen Teilnehmern ausgetauscht werden, die Prozessdaten erzeugen oder verarbeiten.

Für das Senden und Empfangen einer PDO-Nachricht steht jeweils ein PDO zur Verfügung:

- Das T_PDO um PDO-Nachrichten zu senden (T: "Transmit"),
- Das R_PDO, um PDO-Nachrichten zu empfangen (R: "Receive").

0198441113585, V2.01, 11.2008

3.4.2 PDO-Datenaustausch

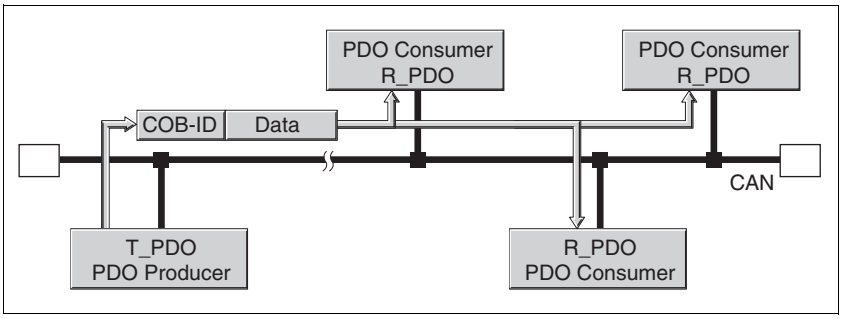


Bild 3.16 PDO-Nachrichtenaustausch

Der Datenaustausch mit PDOs folgt der Producer-Consumer-Beziehung und kann auf 3 Arten ausgelöst werden

- synchronisiert
- ereignisgesteuert, asynchron
- auf Anforderung eines Consumers, asynchron

Die Steuerung der synchronisierten Datenbearbeitung übernimmt das SYNC-Objekt. Synchrone PDO-Nachrichten werden wie die übrigen PDO-Nachrichten sofort übermittelt, werden aber erst mit dem nächsten SYNC ausgewertet. Durch synchronisierten Datenaustausch können z.B. mehrere Antriebe gleichzeitig gestartet werden.

PDO-Nachrichten, die auf Anforderung oder ereignisgesteuert abgerufen werden, wertet der Teilnehmer sofort aus.

Die Übertragungsart kann für jedes PDO separat über Subindex 02_h (transmission type) der PDO Kommunikationsparameter eingestellt werden. Die Objekte sind in 8 "Objektverzeichnis" angegeben.

Event Driven – Ereignisgesteuert

Das "Ereignis" ist die Änderung der PDO-Daten. Die Daten werden in diesem Mode sofort nach einer Änderung gesendet. Hierbei ist zu beachten, dass Beispielsweise bei einer Positionierung sich die Istposition ständig ändert und somit sehr viele PDOs gesendet werden. Eine solche Vielzahl von PDOs kann auf zwei Arten verhindert werden:

- A) Es gibt die Möglichkeit einen "Inhibit Timer" (Objekt 1803_h Subindex 3) einzustellen. Das PDO wird dann erst nach Ablauf dieser Inhibit Zeit übertragen.
- B) Mit Hilfe einer Bitmaske können Sie die Überprüfung auf Änderungen (=Ereignis) einschränken. Beschreibung siehe unten im Abschnitt "Bitmaske für T_PDO4".

Eine weitere Möglichkeit zur "Erzeugung" eines Ereignisses ist die Aktivierung eines "Event Timers" (Objekt 1803_h Subindex 5). Dieser Zähler wird aktiviert, indem ein Wert ungleich Null eingetragen wird. Der Ablauf des Zählers stellt dann zusätzlich ein Ereignis dar. Das heißt das PDO wird bei Wertänderung oder bei Zählerereignis übertragen.

Synchronisiert

Bei dieser Übertragungsart wird die Übertragung eines PDO in Relation zu einem SYNC Objekt durchgeführt. Eine genaue Beschreibung folgt im 3.5 "Synchronisation".

0198441113585, V2.01, 11.2008

Remotely requested – auf externe Anforderung

Die Übertragung einer asynchronen PDO wird ausgelöst durch den Empfang einer externen Anforderung. Ein solcher "Remote Request" wird durch ein spezielles Bit im CAN Übertragungsrahmen angezeigt und hat den gleichen COB-ID (communication object identifier) wie das angeforderte Kommunikationsobjekt.

Eine Übersicht über die einzelnen Übertragungsarten findet sich im Objektverzeichnis bei den PDO Parametern.

Bitmaske für T_PDO4

Über die Objekte `CAN.pdo4msk1` (30:9) und `CAN.pdo4msk2` (30:10) kann eine Bitmaske über alle Bit in der T_PDO4 definiert werden. Alle Bitpositionen die eine "Null" enthalten werden dann bei der Überprüfung auf eine Änderung (=Ereignis) nicht berücksichtigt. Dadurch kann man gezielt z.B. nur Änderungen der Information driveStat berücksichtigen.

NameIdx:Sub dez. (hex.)	BedeutungBit-Belegung	Datentyp-Wertebereich (dez.)	EinheitDefault (dez.)	R/W/ rem.Info
30:9 (1E:09h)	Der Defaultwert 4294967295 entspricht 0xFFFFFFFF.	UINT32	-4294967295	R/W/-
30:10 (1E:0Ah)	Beschreibung siehe Objekt pdo4msk1.	UINT32	-0	R/W/-

Tabelle 3.7 Parameter für den CAN-Bus

Beispiel

In diesem Beispiel wird durch Nullsetzen des Objektes CAN.pdo4msk2 verhindert, dass Änderungen der aktuellen Position einen Event auslösen.

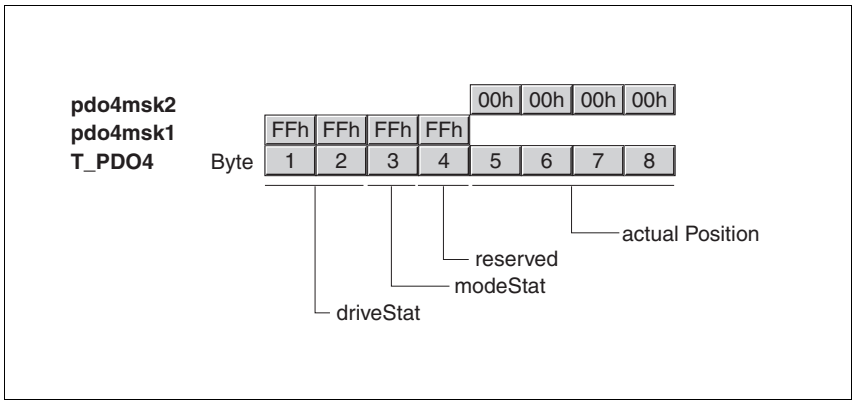


Bild 3.17 Nullsetzen des Objektes `CAN.pdo4msk2`

0198441113585, V2.01, 11.2008

Prozessdaten anfordern

Ein oder mehrere Netzwerkteilnehmer mit Consumer-Funktion können PDO-Nachrichten von einem Producer anfordern. Der Producer wird über die COB-Id der Anforderung identifiziert und antwortet mit der geforderten PDO.

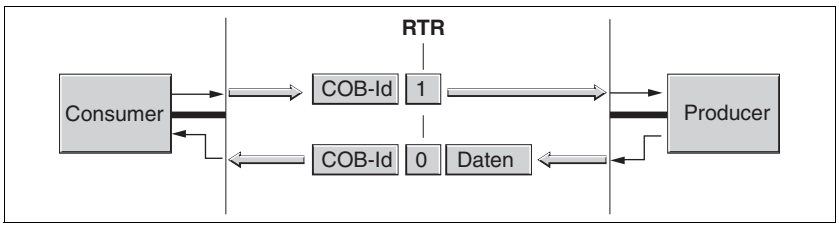


Bild 3.18 Anforderung einer Nachricht mit RTR = 1

Zur Erkennung einer Anforderung wird das RTR-Bit einer CAN-Nachricht benutzt (RTR: **R**emote **T**ransmission **R**equest). Die COB-Id bleibt für beide Nachrichten gleich:RTR = 0: Übertragung von DatenRTR = 1: Anforderung von Daten.

RTR-Anforderung einstellen

Für jedes PDO kann separat eingestellt werden, ob es auf RTR-Anforderungen reagieren soll. Die Kennung wird über Subindex 01, Bit 30_h jedes PDO ein- oder ausgeschaltet. Subindex 02_h (Transmission Type) der Objekte legt den Übertragungstyp fest. Nur wenn die RTR-Übertragung für ein PDO eingeschaltet ist, antwortet das PDO auf eine Anforderung über das Bit RTR. Die Subindex-Werte zur Nutzung des Bits RTR sind:

Objekte 1403_h, 1803_h Subindex 02_h, "transmission type"	**Bedeutung**
252	RTR aktiviert, synchron
253	RTR aktiviert, asynchron

Tabelle 3.8 Subindexe zur Nutzung des Bits

Eine Übersicht aller Werte zum Subindex 02_h finden Sie im Objektverzeichnis beim jeweiligen Objekt angegeben.

Der integrierte Antrieb kann keine PDO anfordern, kann jedoch auf die Anforderung von PDO reagieren.

0198441113585, V2.01, 11.2008

3.4.2.1 Dynamisches und statisches PDO-Mapping

Dynamisches PDO-Mapping

Die Einstellungen für das PDO-Mapping werden für jedes PDO in einem zugeordneten Kommunikationsobjekt definiert. Sind die Einstellungen zum PDO-Mapping für ein PDO änderbar, spricht man von dynamischen PDO-Mapping für das PDO. Das dynamische PDO-Mapping ermöglicht eine flexible Zusammenstellung verschiedener Prozessdaten während des Betriebs.

Statisches PDO-Mapping

Beim statischen PDO-Mapping werden alle Objekte entsprechend einer festgelegten, nicht änderbaren Einstellung im jeweiligen PDO abgebildet.

Eigenschaften des integrierte Antriebs.

Der integrierte Antrieb unterstützt 2 PDO, die Kommunikationsobjekte `T_PD04` und `R_PD04`. Diese beiden PDO4 sind per Default aktiviert.

Diese PDO sind statisch gemappt, können also nicht konfiguriert, sondern nur gelesen werden. Die Indizes der fest eingetragenen Objekte können aus dem Objektbereich PDO-Mapping ausgelesen werden:

- Objekt 1403_h: receive PDO4 communication parameter
- Objekt 1603_h: receive PDO4 mapping
- Objekt 1803_h: transmit PDO4 communication parameter
- Objekt $1A03_h$: transmit PDO4 mapping

3.4.2.2 Empfangs-PDO `R_PD04` (Master -> Slave)

Über den PDO4-Kanal zum Slave kann der Master folgende Aktionen ausführen:

- Steuerung der Zustandsmaschine des Slaves
  - Endstufe aktivieren/deaktivieren
  - "Quick Stop" auslösen und zurücksetzen
  - Fehler zurücksetzen
- Aktivierung der Betriebsarten
  - Betriebsart Punkt-zu-Punkt (absolut und relativ)
  - Geschwindigkeitsprofil
  - Referenzfahrt
  - Maßsetzen
- Sollwerte übergeben
  - Soll-Position
  - Soll-Geschwindigkeit
  - Art der Referenzfahrt

0198441113585, V2.01, 11.2008

Aufbau des R_PDO4:

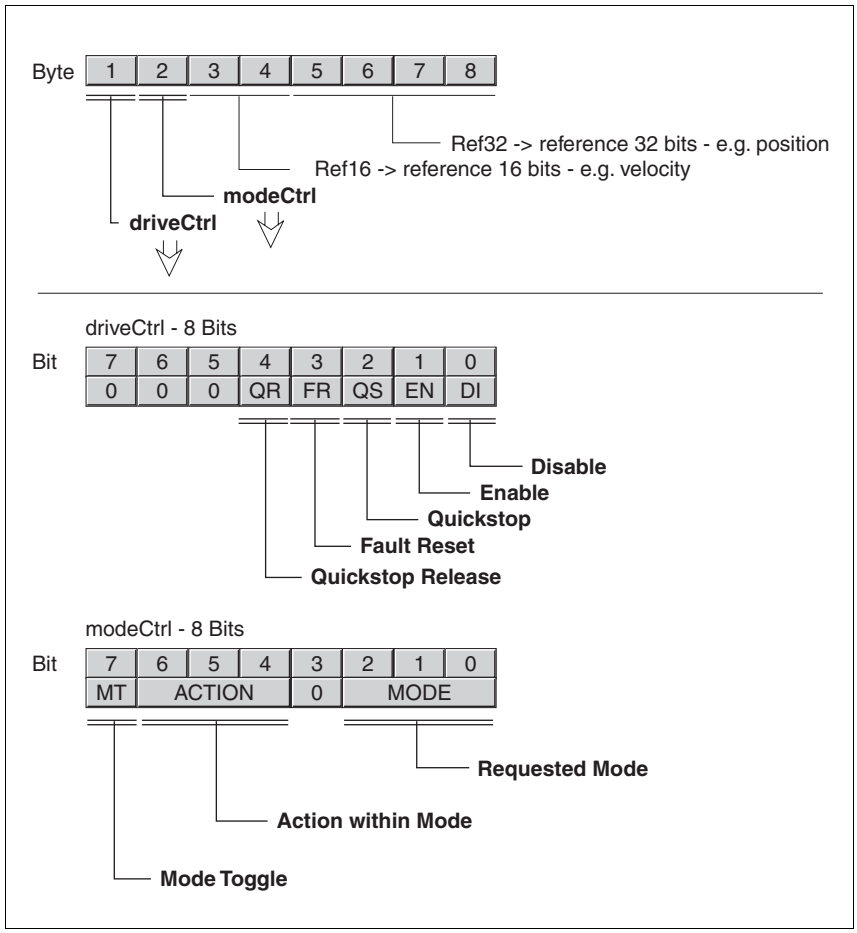


Bild 3.19 Aufbau der R_PDO4

Zustandsmaschine – `drivectrl`

Die Steuerung der Zustandsmaschine erfolgt über PDO4 oder das SDO-Objekt `drivectrl`, 28:1, jeweils über die Bits 0 ... 4.

Im PDO-Betrieb wird mit einem wechsel von 0 auf 1 die jeweilige Funktion ausgelöst.

Beim Zugriff mittels SDO genügt ein Schreibzugriff mit gesetztem Bitwert, es muss also kein Flankenwechsel durchgeführt werden.

Steuerung derZustandsmaschine	PDO4Bits 0 ... 4	SDO-Objekt drivectrl, 28:1Bits 0 ... 4
Bit 0: Endstufe Disable	Bearbeitung wird bei einem wechsel von 0 auf 1 durchgeführt	Bearbeitung wird bei Schreibzugriff durchgeführt wenn Bitwert=1
Bit 1: Endstufe Enable		
Bit 2: Quickstop		
Bit 3: Fault Reset		
Bit 4: Quickstop Release		

Einen Sonderfall stellt der Wert "0" dar: Wenn bei der Übertragung alle Bits 0 ... 7 "Null" sind, wird dies vom Slave als Befehl "Disable" interpretiert und die Endstufe deaktiviert. Dies gilt sowohl für PDO- als auch für SDO-Zugriffe.

0198441113585, V2.01, 11.2008

Fehlerbearbeitung Wenn Anforderungen zur Steuerung der Zustandsmaschine vom Slave nicht umgesetzt werden können, ignoriert der Slave diese Anforderungen. Eine Fehlerreaktion erfolgt nicht.

Betriebsarten – `modeCtrl`

Im PDO-Betrieb erfolgt die Steuerung der Betriebsarten über das Objekt `modeCtrl`. Zum Auslösen einer Betriebsart oder Ändern von Sollwerten muss der Master folgende Werte eintragen:

- Sollwerte in die Felder "Ref16" und "Ref32"
- Betriebsart mit `modeCtrl`, Bits 0 ... 2 wählen (MODE)
- Aktion für diese Betriebsart mit `modeCtrl`, Bits 4 ... 6 wählen (ACTION)
- `modeCtrl`, Bit 7 toggeln (MT)

Nachfolgende Tabelle zeigt die möglichen Betriebsarten und die dazugehörigen Sollwerte

Mode Bits 0... 2	Action Bits 4 ... 6	modeCtrl [1] Bits 0 ... 6	Beschreibung	entspricht Objekt [2]	Sollwert Ref16	Sollwert Ref32
1 (JOG)	0	01h	Manuellfahrt	41:3	Start (wie Objekt 41:1)	-
2 (REF)	0	02h	Maßsetzen	40:3	-	Maßsetz-Position
	1	12h	Referenzfahrt	40:1	Typ (wie Objekt 40:1)	-
3 (PTP)	0	03h	Absolute Positionierung	35:1	Soll-Geschwindigkeit	Soll-Position
	1	13h	Relative Positionierung	35:3	Soll-Geschwindigkeit	Soll-Position
	2	23h	Fortsetzen der Positionierung	35:4	Soll-Geschwindigkeit	-
4 (VEL)	0	04h	Geschwindigkeitsprofil	36:1	Soll-Geschwindigkeit	-

1) Spalte entspricht dem in Byte `modeCtrl` einzutragenden Wert, jedoch ohne ModeToggle (Bit 7).

2) Spalte zeigt Index:Subindex (dezimal) der entsprechenden Betriebsarten-Objekte, die in der Gerätedokumentation ausführlicher beschrieben sind.

0198441113585, V2.01, 11.2008

Soll-Positionen werden in Inkrementen, Soll-Geschwindigkeiten in [min^{-1}] eingetragen.

⚠ WARNUNG

UNBEABSICHTIGTER BETRIEB

- Berücksichtigen Sie, dass Eingaben in diese Parameter sofort nach Empfang des Datensatzes von der Antriebssteuerung ausgeführt werden.
- Vergewissern Sie sich, dass die Anlage frei und bereit für Bewegung ist, bevor Sie diese Parameter ändern.

Nichtbeachtung dieser Vorkehrungen kann zu Tod, schwerwiegenden Verletzungen oder Materialschäden führen.

Bei gleichzeitiger Übertragung von Betriebsart, Soll-Position und Soll-Geschwindigkeit in einem PDO muss die Datenkonsistenz gewährleistet sein. Deshalb wertet der Slave die Betriebsarten-Daten nur aus wenn Bit 7 getoggelt wurde. Togglen bedeutet, dass seit der letzen Übertragung bei diesem Bit ein "0 -> 1"- oder ein "1 -> 0"-Flankenwechsel erkannt wurde.

Bit 7 wird im Antwort-PDO4 vom Slave gespiegelt, sodass über die PDO4 ein synchronisierter Betrieb möglich ist.

Fehlerbearbeitung

Durch das Toggeln des Bits 7 werden Anforderungen zur Betriebsart ausgelöst. Wenn diese Anforderungen nicht umgesetzt werden können, führt der Slave eine Fehlerreaktion aus. wie im Abschnitt Transmit-PDO4 - Fehlerbearbeitung beschrieben.

3.4.2.3 Sende-PDO T_PD04 (Slave zu Master)

Mit den Default-Einstellungen des Slaves wird die Sende-PDO ereignisgesteuert asynchron verschickt "Event Driven", die Einstellung der Zeit "Inhibit Time" ist möglich.

Über den PDO4-Kanal liefert der Slave folgende Informationen zum Master:

- Zustand der Zustandsmaschine
- Fehler und Warnungen
- Aktive Betriebsart
- Status der aktiven Betriebsart
  - Betriebsart beendet
  - Fehler aufgetreten
  - Soll-Geschwindigkeit bzw. Soll-Position erreicht
  - Ist-Position
- Slave referenziert
- Quittierung von Betriebsarten-Anforderungen
- Zustand der 24V Ein- bzw. Ausgänge

0198441113585, V2.01, 11.2008

Aufbau des T_PDO4:

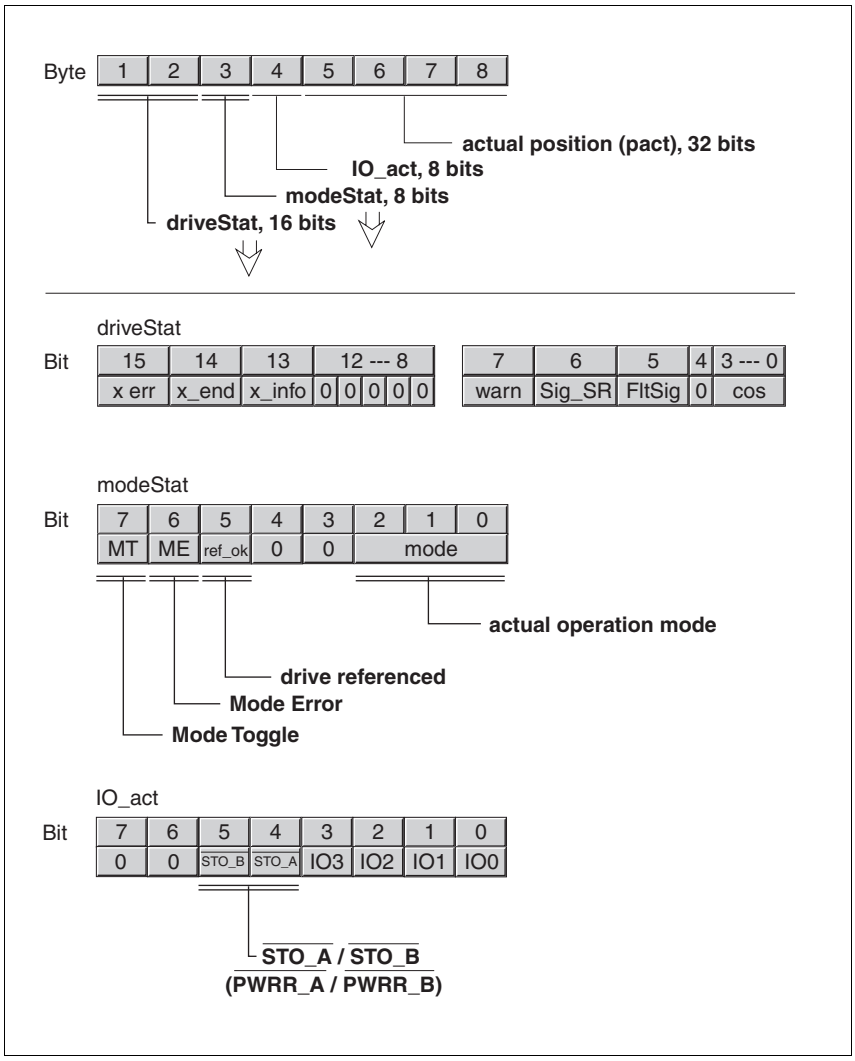


Bild 3.20 Aufbau des T_PDO4

Statuswort `driveStat`

Die Informationen im Statuswort `driveStat` entsprechen den Bits 0 ...15 des Objektes `Status.driveStat`, 28:2.

Inhalt der Informationen:

- Zustand der Zustandsmaschine
- Warnungs- und Fehlerbits
- Status der aktuellen Achsbetriebsart.

0198441113585, V2.01, 11.2008

Betriebsart `modeStat`

Dieses Feld entspricht den Bits 0 ... 2 des Objektes `Status.xMode_act`. Die Bits 6 und 7 sind zusätzliche Informationen die dazu benutzt werden können über die PDO eine synchronisierte Betriebsarten-Steuerung durchzuführen.

Das Feld enthält folgende Informationen:

Bit	Name	Beschreibung
0...2	mode	aktuell eingestellte Betriebsart wie bei R_PDO4
5	ref_ok	Ist gesetzt, wenn der Slave durch Referenzfahrt oder Maßsetzen erfolgreich referenziert wurde.
6	ME, ModeError	Ist gesetzt wenn eine Anforderung des Masters über R_PDO4 vom Slave abgelehnt wurde.
7	MT, ModeToggle	Spiegelung des Bits 7 (Mode Toggle) vom R_PDO4

3.4.2.4 Handshake mit Mode Toggle Bit

Mode Toggle

Mit den Sendedaten `modeCtrl`, Bit 7 (MT), und den Empfangsdaten `modeStat`, Bit 6 (ME) und 7 (MT), kann eine synchronisierte Bearbeitung durchgeführt werden. Synchronisierte Bearbeitung bedeutet, dass der Master Rückmeldungen vom Slave abwartet und korrekt darauf reagieren kann.

Beispiel Positionierung

Der Master startet eine Positionierung zum Zeitpunkt t_0. An den Zeitpunkten t_1, t_2 ... prüft der Master die Rückmeldungen vom Slave. Er wartet auf das Ende der Positionierung, indem er das Input Assembly auf Bit `x_end = 1` (Positionierende) prüft.

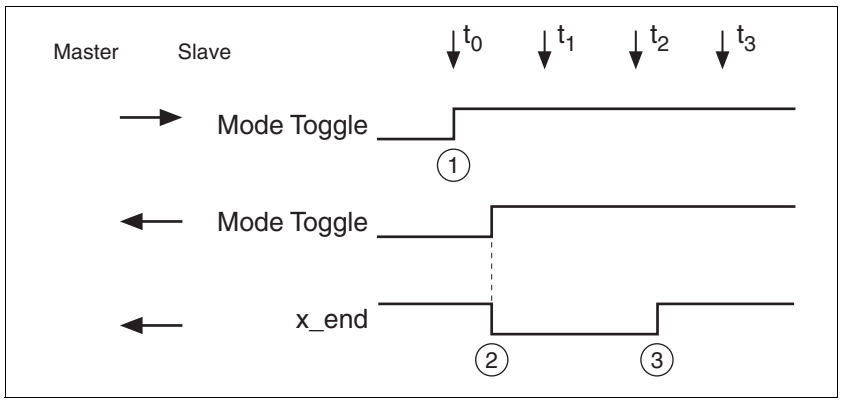


Bild 3.21 Mode Toggle Handshake

(1) Master startet Positionierung durch MT = 1 im Byte `modeCtrl`

(2) Slave meldet Positionierung läuft mit MT = 1 in `modeStat` und gleichzeitig `x_end` = 0 in `driveStat`

(4) Slave meldet Positionierung beendet mit `x_end` = 1 in `driveStat`

0198441113585, V2.01, 11.2008

Beispiel kurze Bewegung

Der Master startet eine Positionierung, die nur eine sehr kurze Zeit in Anspruch nimmt. Die Dauer ist kürzer als der Abfragezyklus des Masters. Zum Zeitpunkt t_1 ist die Bewegung schon beendet. Anhand des `x_end` Bits kann der Master nicht entscheiden, ob die Bewegung schon beendet ist oder noch gar nicht gestartet wurde. Dies erkennt er jedoch zusammen mit dem MT-Bit vom Slave:

Master MT	Slave MT	Slave x_end	
1	0	1	Slave hat Befehl noch nicht erkannt
1	1	0	Slave hat Befehl erkannt, Positionierung läuft
1	1	1	Slave meldet Positionierung beendet

Generell darf der Master nur Daten auswerten, bei denen das empfangene MT-Bit identisch ist mit dem zuletzt von ihm gesendeten.

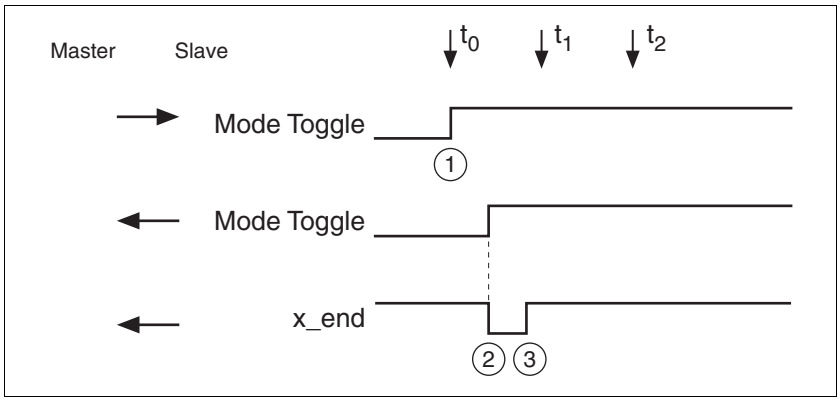


Bild 3.22 Mode Toggle Handshake, kurze Bewegung

(1) Master startet Positionierung durch MT = 1 im Byte `modeCtrl`

(2+3) Slave meldet Positionierung läuft mit MT = 1 in `modeStat` und gleichzeitig `x_end` = 0 in `driveStat`

(4) Slave meldet Positionierung beendet mit `x_end` = 1 in `driveStat`

0198441113585, V2.01, 11.2008

Fehlerbearbeitung

Wenn der Master das Bit 7 (MT) toggelt, gilt dies als Anforderung an den Slave, eine Betriebsart zu starten oder Daten der aktuellen Betriebsart zu ändern. Wenn die Anforderung nicht bearbeitet werden kann, ändert sich die eingestellte Betriebsart nicht und der Slave setzt in `modeStat` das Bit 6 (ME = `ModeError`).

Die aktive Betriebsart wird also nicht beeinflusst und es findet auch kein Zustandswechsel statt.

Das Bit 6 (ME) bleibt gesetzt, bis der Master in `modeCtrl` das Bit 7 (MT) erneut toggelt, und damit einen neuen Befehl auslöst.

Der Master kann den entsprechenden Fehlercode über einen Lesezugriff auf Parameter `ModeError` auslesen.

Mögliche Gründe für eine fehlgeschlagene Betriebsarten-Anforderung:

- Sollwerte außerhalb des Wertebereichs
- Umschaltung der Betriebsart bei laufender Bearbeitung (nicht möglich)
- Ungültige Betriebsart angefordert
- Das Gerät befindet sich nicht im Zustand 6 (`Operation Enable`) des Zustandsdiagramms

Weitere Informationen finden Sie im Produkthandbuch.

3.4.2.5 Emergency-Dienst

Der Emergency-Dienst (emergency: Notfall) meldet interne Gerätefehler über den CAN-Bus. Die Fehlermeldung wird mit einem Objekt `EMCY` entsprechend der "Consumer-Producer"-Beziehung an alle Netzwerkteilnehmer gesandt.

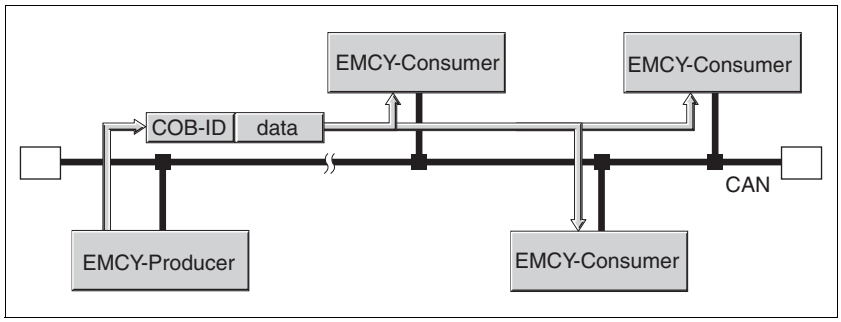


Bild 3.23 Fehlermeldung über das Objekt `EMCY`

0198441113585, V2.01, 11.2008

EMCY-Nachricht Ursachen für eine EMCY-Nachricht sind:

- asynchrone Fehler, Error-Code = 1000_hWenn ein interner Gerätefehler auftritt, wechselt der Slave entsprechend der Geräte-Zustandsmaschine in den Fehlerzustand. Gleichzeitig sendet der Slave eine EMCY-Nachricht mit Fehlerregister und Fehlercode.
- PDO4-Fehler bei der Betriebsarten-Steuerung, Error-Code = 8200_hWenn die Anforderung für eine Betriebsart mittels PDO4 fehlschlägt sendet der Slave ebenfalls eine EMCY-Nachricht.
- CAN- Komunikationsfehler, Error-Code = 8100_h

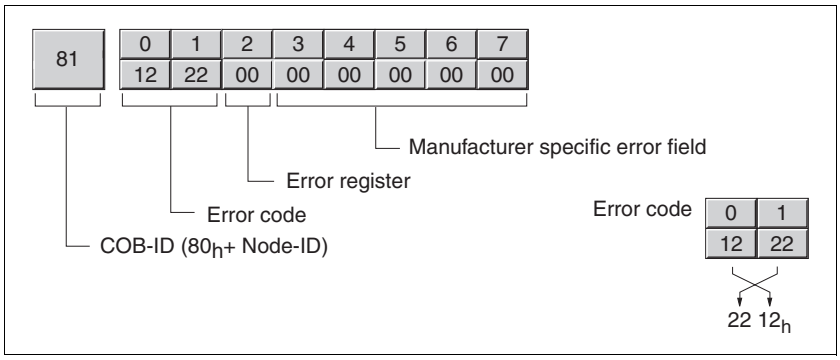


Bild 3.24 EMCY-Nachricht

- Byte 0, 1(Error Code): CANopen Error-CodeDieser Wert ist beim Slave je nach Fehlerursache 1000h, 8200_h oder 8100_h.
- Byte 2 (Error Register): FehlerregisterDer Wert ist auch im Objekt Error register, 1001_h gespeichert.
- Byte 3 (Manufacturer Specific Error Field):Herstellerspezifische Fehler, Fehlerklasse

  Byte 6 und 7 sind 0. In Byte 4,5 ist eine herstellerspezifische Fehlernummer abgelegt.Eine Liste der Fehlernummern finden Sie im Gerätehandbuch.

COB-Id Für jeden Netzwerkteilnehmer, der ein Objekt EMCY unterstützt, errechnet sich die COB-Id aus der Knotenadresse:

COB-Id = Funktionskode des Objektes EMCY, 80_h + Node-Id

0198441113585, V2.01, 11.2008

3.5 Synchronisation

Das Synchronisationsobjekt SYNC steuert den synchronen Nachrichtenaustausch zwischen Netzwerkteilnehmern, um z.B. den gleichzeitigen Start mehrerer Antriebe zu ermöglichen.

Der Datenaustausch folgt der Producer-Consumer-Beziehung. Das SYNC-Objekt wird von einem Netzwerkteilnehmer an alle Teilnehmer verschickt und kann von allen Teilnehmern ausgewertet werden, die synchrone PDOs unterstützen.

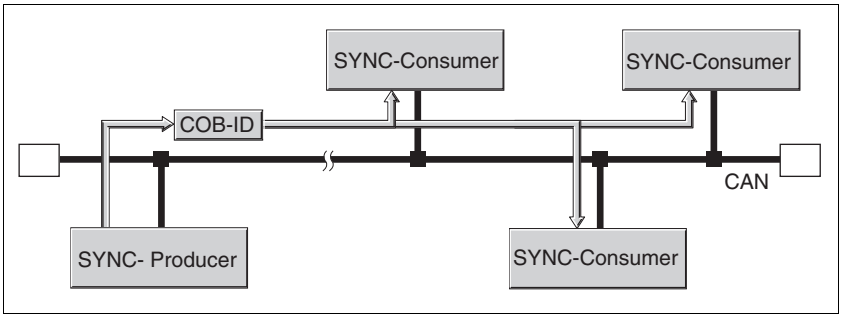


Bild 3.25 SYNC-Nachricht

Zeitwerte zur Synchronisation

2 Zeitwerte definieren das Verhalten der synchronen Datenübertragung:

- Die Zykluszeit" gibt die Zeitspanne zwischen 2 SYNC-Nachrichten an. Sie wird mit dem Objekt `Communication cycle period(1006h)` eingestellt.
- Das synchrone Zeitfenster" legt die Zeitspanne fest, in der synchrone PDO-Nachrichten empfangen und gesendet werden müssen. Das Zeitfenster wird über das Objekt `Synchronous window length (1007h)` definiert.

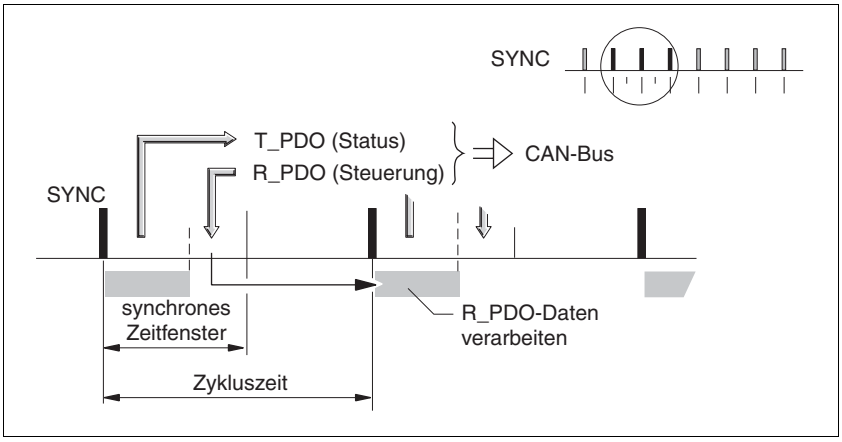


Bild 3.26 Synchronisationszeiten

0198441113585, V2.01, 11.2008

Synchrone Datenübertragung

Aus Sicht eines SYNC-Empfängers werden in einem Zeitfenster zuerst die Statusdaten in einem T_PDO verschickt, anschließend neue Steuerdaten über ein R_PDO empfangen. Die Steuerdaten werden aber erst bei Eintreffen der nächsten SYNC-Nachricht verarbeitet. Das SYNC-Objekt selbst überträgt keine Daten.

Zyklische und azyklische Datenübertragung

Der synchrone Nachrichtenaustausch kann zyklisch oder azyklisch ausgeführt werden.

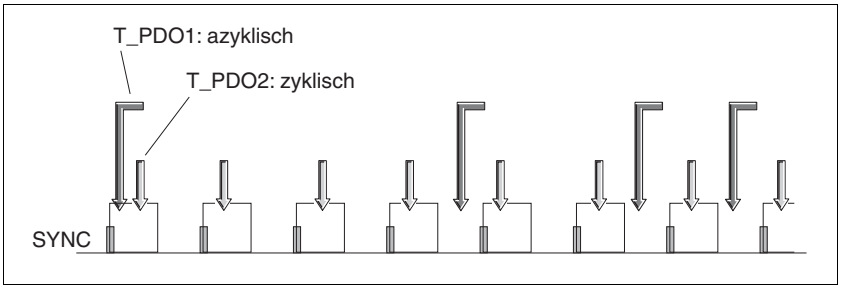


Bild 3.27 Zyklische und azyklische Übertragung

Mit der zyklischen Übertragung werden PDO-Nachrichten kontinuierlich in einem festgelegten Takt, z.B. mit jeder SYNC-Nachricht ausgetauscht.

Wird eine synchrone PDO-Nachricht azyklisch übertragen, kann sie zu einem beliebigen Zeitpunkt gesendet oder empfangen werden, wird aber erst mit der nächsten SYNC-Nachricht gültig.

Zyklisches oder azyklisches Verhalten eines PDOs wird im Subindex `transmission type` (02_h) des jeweiligen PDO-Parameters abgelegt, für R_PDO1 z.B. im Objekt `1st receive PDO parameter` ($1400_h:02_h$).

COB-ID, SYNC-Objekt

Zur schnellen Übermittlung wird das SYNC-Objekt mit hoher Priorität und unbestätigt übertragen.

Die COB-ID des SYNC-Objekts ist standardmäßig auf den Wert 128 (80_h) eingestellt. Der Wert kann nach Initialisierung des Netzwerks über das Objekt `COB-ID SYNC Message` (1005_h) geändert werden.

0198441113585, V2.01, 11.2008

3.6 Netzwerk-Management-Dienste

Das Netzwerk-Management (NMT) ist Teil des CANopen-Kommunikationsprofils und wird eingesetzt, um das Netzwerk und die Netzwerkteilnehmer zu initialisieren und die Teilnehmer im Netzwerkbetrieb zu starten, zu stoppen und zu überwachen.

NMT-Dienste werden in einer Master-Slave-Beziehung ausgeführt. Der NMT-Master spricht einzelne NMT-Slaves über ihre Knotenadresse an. Eine Nachricht mit Knotenadresse "0" richtet sich an alle NMT-Slaves gleichzeitig.

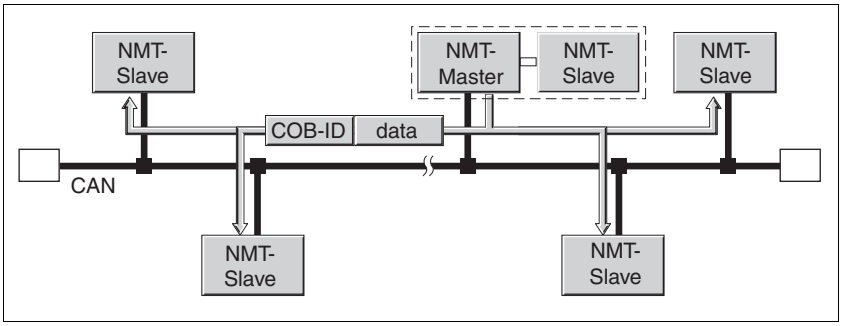


Bild 3.28 NMT-Dienste über die Master-Slave-Beziehung

Das Gerät kann nur die Funktion eines NMT-Slaves übernehmen.

NMT-Dienste

NMT-Dienste können in 2 Gruppen unterteilt werden:

- Dienste zur Gerätekontrolle, um Teilnehmer für die CANopen-Kommunikation zu initialisieren und das Verhalten der Teilnehmer im Netzwerkbetrieb zu steuern
- Dienste zur Verbindungsüberwachung

0198441113585, V2.01, 11.2008

3.6.1 NMT-Dienste zur Gerätekontrolle

NMT-Zustandsmaschine

Die NMT-Zustandsmaschine beschreibt die Initialisierung und die Zustände eines NMT-Slaves im Netzwerkbetrieb.

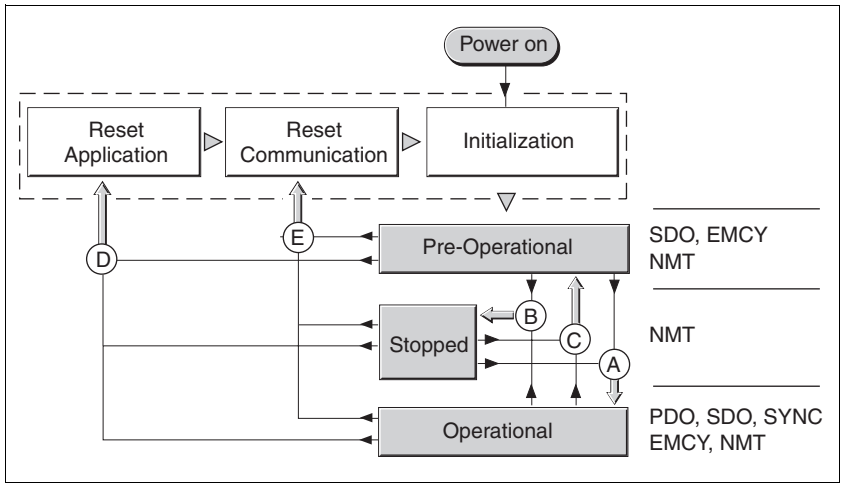


Bild 3.29 NMT-Zustandsmaschine und einsetzbare Kommunikationsobjekte

In der Grafik sind auf der rechten Seite alle Kommunikationsobjekte angegeben, die im jeweiligen Netzwerkzustand eingesetzt werden können.

Initialisierung

Ein NMT-Slave durchläuft nach Einschalten der Versorgungsspannung (Power on) automatisch eine Initialisierungsphase, die ihn für den CAN-Busbetrieb vorbereitet. Nach Abschluss der Initialisierung wechselt der Slave in den Zustand "Pre Operational" und sendet eine Boot-Up-Nachricht. Ab jetzt kann ein NMT-Master das Betriebsverhalten eines NMT-Slaves im Netzwerk über 5 NMT-Dienste steuern, in obiger Grafik dargestellt mit den Buchstaben A bis E.

NMT-Dienst	Übergang	Bedeutung
Start remote node (Netzknoten starten)	A	Wechsel auf Zustand "Operational" Normalen Netzbetrieb zu allen Teilnehmern starten
Stop remote node (Netzknoten stoppen)	B	Wechsel auf Zustand "Stopped" Kommunikation des Teilnehmers im Netzwerk stoppen. Ist eine Verbindungsüberwachung aktiviert, bleibt sie eingeschaltet. Bei aktiver Endstufe (Zustand "Operation Enabled" oder "QuickStop") wird ein Fehler der Fehlerklasse 2 ausgelöst. Der Antrieb wird gestoppt und ausgeschaltet.
Enter Pre-Operational (Auf "Pre-Operational" wechseln)	C	Wechsel auf Zustand "Pre-Operational" Alle Kommunikationsobjekte außer PDOs können eingesetzt werden. Der Zustand "Pre-Operational" kann zur Konfiguration per SDOs genutzt werden: - PDO-Mapping - Start der Synchronisation - Start der Verbindungsüberwachung
Reset node (Knoten rücksetzen)	D	Wechsel auf Zustand "Reset application" Gespeicherte Daten der Geräteprofile laden und automatisch über Zustand "Reset communication" nach "Pre-Operational" wechseln
Reset communication (Kommunikationsdaten rücksetzen)	E	Wechsel auf Zustand "Reset communication" Gespeicherte Daten des Kommunikationsprofils laden und automatisch in Zustand "Pre-Operational" wechseln. Bei aktiver Endstufe (Zustand "Operation Enabled" oder "QuickStop") wird ein Fehler der Fehlerklasse 2 ausgelöst. Der Antrieb wird gestoppt und ausgeschaltet.

0198441113585, V2.01, 11.2008

Persistent gespeicherte Daten

Wird die Versorgungsspannung eingeschaltet (Power on), lädt das Gerät die persistent gespeicherten Objektdaten aus dem EEPROM in das RAM.

NMT-Nachricht

Die NMT-Dienste zur Gerätekontrolle werden als unbestätigte Nachrichten mit der COB-ID = 0 übertragen. Sie erhalten damit standardmäßig die höchste Übertragungspriorität auf dem CAN-Bus.

Der Datenrahmen des NMT-Gerätedienstes besteht aus 2 Byte.

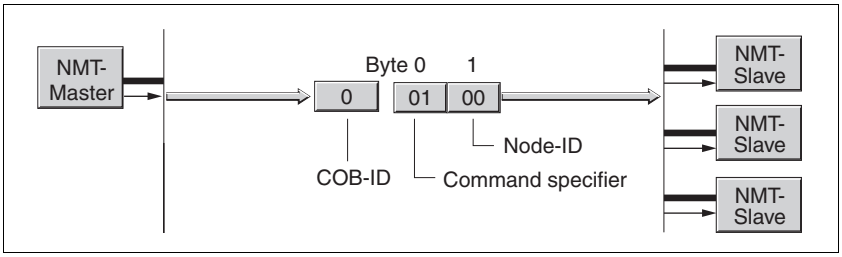


Bild 3.30 NMT-Nachricht

Das erste Byte, der "Command specifier", gibt den benutzten NMT-Dienst an.

Command Specifier	NMT-Dienst	Übergang
1 (01_h)	Start remote node	A
2 (02_h)	Stop remote node	B
128 (80_h)	Enter Pre-Operational	C
129 (81_h)	Reset node	D
130 (82_h)	Reset communication	E

Das zweite Byte adressiert mit einer Knotenadresse zwischen 1 und 127 ($7F_h$) den Empfänger der NMT-Nachricht. Eine Nachricht mit der Knotenadresse "0" richtet sich an alle NMT-Slaves.

3.6.2 NMT-Dienste zur Verbindungsüberwachung

Die Verbindungsüberwachung kontrolliert den Kommunikationsstatus von Netzwerkteilnehmern, so dass auf den Ausfall eines Teilnehmers oder auf eine Unterbrechung im Netzwerk reagiert werden kann.

Es stehen 3 NMT-Dienste zur Verbindungsüberwachung bereit:

- "Node guarding" (engl.: Knotenüberwachung) zur Verbindungsüberwachung eines NMT-Slaves
- "Life guarding" (engl.: Überwachung auf Lebenszeichen) zur Verbindungsüberwachung eines NMT-Masters

0198441113585, V2.01, 11.2008

3.6.2.1 Node/Life guarding

COB-Id

Eine Verbindungsüberwachung wird über das Kommunikationsobjekt `NMT error control (700`$_h$`+node-Id)` ausgeführt. Für jeden NMT-Slave errechnet sich die COB-Id aus der Knotenadresse:

COB-Id = Funktionscode `NMTerror control (700`$_h$`) + node-Id .`

Aufbau der NMT-Nachricht

Nach Anforderung durch den NMT-Master antwortet der NMT-Slave mit einem Datenbyte.

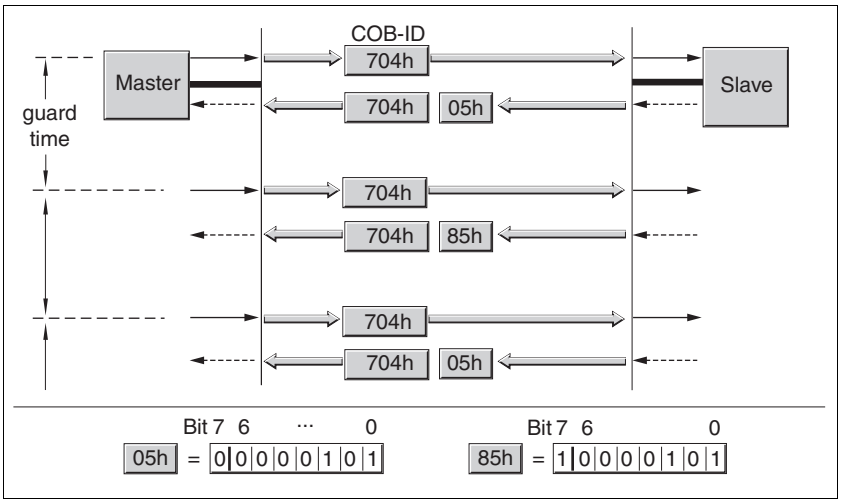


Bild 3.31 Rückmeldung des NMT-Slaves

Bit 0 bis 6 markieren den NMT-Zustand des Slaves:

- 4 (04_h): "Stopped"
- 5 (05_h): "Operational"
- 127 ($7F_h$) : "Pre-Operational"

Bit 7 wechselt nach jedem "guard time"-Intervall seinen Zustand zwischen "0" und "1", so dass der NMT-Master eine zweite Rückmeldung innerhalb der Intervallzeit "guard time" erkennen und ignorieren kann. Die erste Anforderung bei Start der Verbindungsüberwachung beginnt mit Bit 7 = 0.

Während der Initialisierungsphase eines Teilnehmers darf die Verbindungsüberwachung nicht aktiviert sein. Der Zustand von Bit 7 wird zurückgesetzt, sobald der Teilnehmer den NMT-Zustand "Reset communication" durchläuft.

Im NMT-Zustand "Stopped" läuft die Verbindungsüberwachung weiter.

Konfiguration

Node/Life Guarding wird konfiguriert über:

- guard time ($100C_h$)
- life time factor ($100D_h$)

0198441113585, V2.01, 11.2008

Verbindungsfehler

Der NMT-Master meldet an das übergeordnete Masterprogramm einen Verbindungsfehler, wenn:

- sich der Slave nicht innerhalb der Zeitspanne "guard time" zurückmeldet
- sich der NMT-Zustand des Slaves ohne Veranlassung durch den NMT-Master geändert hat.

Bild 3.32 zeigt eine Fehlermeldung nach Ablauf des dritten Zyklus wegen fehlender Antwort eines NMT-Slaves.

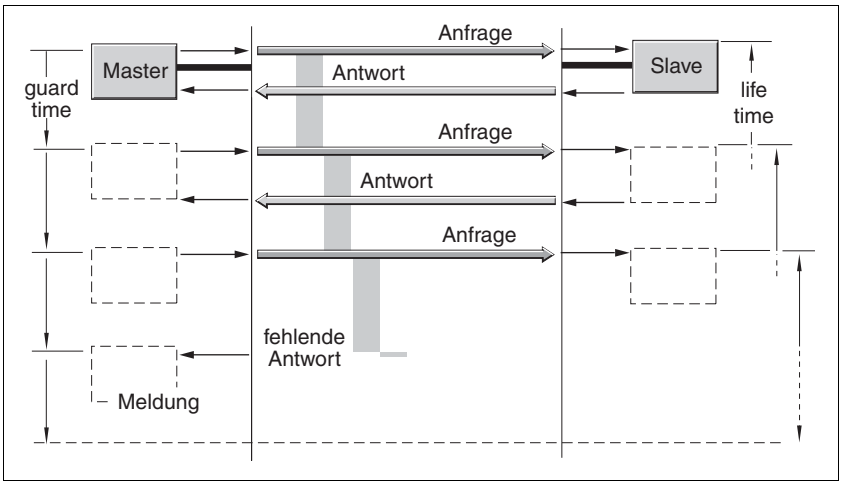


Bild 3.32 "Node guarding" und "Life guarding" mit Zeitintervallen

Boot-Up-Nachricht

Das Kommunikationsprofil DS301, Version 4.0 definiert eine zusätzliche Aufgabe für die NMT-Dienste: Das Senden einer Boot-Up-Nachricht.

Mit einer Boot-Up-Nachricht informiert ein Netzwerkteilnehmer alle anderen Netzwerkteilnehmer, dass er betriebsbereit ist.

Eine Boot-Up-Nachricht besteht aus der COB-Id des Objekts `NMT Error Control` und wird ohne Daten übertragen. Standardeinstellung der COB-Id ist `1792 (700h) + node-Id`

0198441113585, V2.01, 11.2008

4 Installation

⚠ WARNUNG

VERLUST DER STEUERUNGSKONTROLLE

- Bei der Entwicklung des Steuerungskonzeptes muss der Anlagenhersteller die potentiellen Ausfallmöglichkeiten der Steuerungspfade berücksichtigen und für bestimmte kritische Funktionen Mittel bereitstellen, mit denen während und nach dem Ausfall eines Steuerungspfades sichere Zustände erreicht werden. Beispiele für kritische Steuerungsfunktionen sind: NOT-HALT, Endlagen-Begrenzung, Spannungsausfall und Wiederanlauf.
- Für kritische Funktionen müssen separate oder redundante Steuerungspfade vorhanden sein.
- Die Anlagensteuerung kann Kommunikationsverbindungen umfassen. Der Anlagenhersteller muss die Folgen unerwarteter Zeitverzögerungen oder Ausfälle der Kommunikationsverbindung berücksichtigen.
- Beachten Sie die Unfallverhütungsvorschriften sowie alle geltenden Sicherheitsbestimmungen. [1)]
- Jede Anlage, in der das in diesem Handbuch beschriebene Produkt verwendet wird, muss vor dem Betrieb einzeln und gründlich auf korrekte Funktion überprüft werden.

Nichtbeachtung dieser Vorkehrungen kann zu Tod oder schwerwiegenden Verletzungen führen.

1) Für USA: siehe NEMA ICS 1.1 (neueste Ausgabe), Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control sowie NEMA ICS 7.1 (neueste Ausgabe), Safety Standards for Construction and Guide for Selection, Installation for Construction and Operation of Adjustable-Speed Drive Systems.

⚠ WARNUNG

STÖRUNG VON SIGNALEN UND GERÄTEN

Gestörte Signale können unvorhergesehene Gerätereaktionen hervorrufen.

- Führen Sie die Verdrahtung gemäß den EMV-Maßnahmen durch.
- Überprüfen Sie die korrekte Ausführung der EMV-Maßnahmen.

Nichtbeachtung dieser Vorkehrungen kann zu Tod, schwerwiegenden Verletzungen oder Materialschäden führen.

Angaben zur Geräteinstallation und zum Anschluss des Gerätes an den Feldbus finden Sie im Produkthandbuch.

Slave mit DIP-Schaltern

Vor dem Einbau des Slaves in die Anlage müssen Sie die Netzwerkadresse und die Baudrate über die DIP-Schalter im Steckergehäuse einstellen.

Informationen zur Belegung der DIP-Schalter finden Sie im Gerätehandbuch, Kapitel "Installation".

0198441113585, V2.01, 11.2008

0198441113585, V2.01, 11.2008

5 Inbetriebnahme

⚠ GEFAHR

UNBEABSICHTIGTE FOLGEN DES BETRIEBS

Beim Start der Anlage sind die angeschlossenen Antriebe in der Regel außer Sichtweite des Anwenders und können nicht unmittelbar überwacht werden.

- Starten Sie die Anlage nur, wenn sich keine Personen oder Hindernisse im Gefahrenbereich befinden.

Nichtbeachtung dieser Vorkehrungen führt zu Tod oder schweren Verletzungen.

⚠ WARNUNG

UNBEABSICHTIGTER BETRIEB

- Schreiben Sie nicht in reservierte Parameter.
- Schreiben Sie nicht in Parameter bevor Sie die Funktion nicht verstanden haben. Weitere Informationen finden Sie im Produkthandbuch.
- Führen Sie erste Tests ohne angekoppelte Lasten durch.
- Vergewissern Sie sich, dass die Anlage frei und bereit für die Bewegung ist, bevor Sie Parameter ändern.
- Überprüfen Sie bei der Feldbus-Kommunikation die Verwendung der Bits: Bit 0 steht ganz rechts (least significant). Bit 15 steht ganz links (most significant).
- Überprüfen Sie bei der Feldbus-Kommunikation die Verwendung der Wortfolge.
- Stellen Sie keine Feldbus-Verbindung her, bevor Sie nicht die Kommunikations-Prinzipien verstanden haben.

Nichtbeachtung dieser Vorkehrungen kann zu Tod, schwerwiegenden Verletzungen oder Materialschäden führen.

5.1 Geräteinbetriebnahme

Voraussetzung für die Installation im Netzwerk ist die korrekte mechanische und elektrische Installation des Gerätes sowie die erfolgreiche Durchführung der Geräte-Inbetriebnahme.

Führen Sie die Geräte-Inbetriebnahme entsprechend dem Produkthandbuch durch. Das Gerät ist damit für den Einsatz im Netzwerk vorbereitet.

0198441113585, V2.01, 11.2008

5.2 Adresse und Baudrate

Es können bis zu 32 Geräte in einem CAN-Bus-Netzwerkzweig und bis zu 127 Geräte im erweiterten Netzwerk adressiert werden. Jedes Gerät wird über eine eindeutige Adresse identifiziert. Die Knotenadresse für ein Gerät ist auf 127 voreingestellt.

Die Baudrate ist auf 125 KBaud voreingestellt.

Jedes Gerät muss eine eigene Knotenadresse erhalten, das heißt jede Knotenadresse darf nur einmal im Netzwerk vergeben sein.

Adresse und Baudrate einstellen

Die Adresse wird lokal am Gerät in den Parameter `canAddr` und die Baudrate in den Parameter `canBaud` eingetragen.

Die Baudrate muss für alle Geräte im Feldbus gleich eingestellt sein.

5.3 Inbetriebnahme der Feldbusverbindung

5.3.1 Feldbusbetrieb starten

Konfiguration mit SyCon

Hinweis zur Verwendung der Hilscher Konfigurationssoftware SyCon:

Im Dialog `Knotenkonfiguration` die Einstellung `Geräteprofil` (Wert = 0) **nicht ändern**!

Sofern Sie diesen Wert geändert haben, funktioniert die Kommunikation mit dem Antrieb nicht mehr. Es ist jedoch nicht mehr möglich die Einstellung in den Originalzustand zurückzusetzen.

Um die Kommunikation mit dem Produkt wiederherzustellen müssen Sie:

- Die Schaltfläche `Knoten BootUp` im Dialog `Knotenkonfiguration` anklicken.
- Klicken Sie auf `Prüfe Knoten Type and Profile` im Dialog `Knoten Aufschaltreihenfolge` um diesen Schritt zu umgehen.

Feldbusbetrieb testen

Testen Sie nach korrekter Einstellung der Übertragungsdaten den Feldbusbetrieb.

Dazu muss ein CAN-Konfigurationswerkzeug installiert sein, das CAN-Nachrichten anzeigt. Erfasst wird die Rückmeldung vom Antrieb durch eine Boot-Up-Nachricht:

- Schalten Sie die Spannungsversorgung des Antriebs aus und wieder ein.
- Beobachten Sie die Netzwerk-Nachrichten kurz nach Einschalten des Geräts. Der Positioniersteuerung sendet nach der Bus-Initialisieung eine 1 Byte lange Boot-Up-Nachricht: 128 (80_h)+node-Id.

Mit Werkseinstellung der Knotenadresse auf 127 ($7F_h$) wird die Boot-Up-Nachricht 255 (FF_h) über den Bus gesendet. Der Antrieb kann anschließend über NMT-Dienste in Betrieb genommen werden.

0198441113585, V2.01, 11.2008

5.3.2 Fehlersuche

Wenn der Slave nicht antwortet, kontrollieren Sie folgende Einstellungen:

- ▶ Ist der Slave gestartet und der Master eingeschaltet ?
- ▶ Sind die Kabelverbindungen mechanisch und elektrisch in Ordnung ?
- ▶ Ist die richtige Adresse am Slave eingestellt. Überprüfen Sie die Einstellungen der DIP-Schalter und des HEX-Schalters. Die Einstellungen sind im Gerätehandbuch beschrieben.Slaves ohne DIP-Schalter sind standardmäßig auf die CAN-Adresse 127 ($7F_h$) und die Baud-Rate 125 [kBit/s] eingestellt. Diese Einstellungen können Sie über den CAN selbst oder über die RS485-Schnittstelle mit dem PC-Inbetriebnahmetool ändern.
- ▶ Sind an Master und Slave die gleiche Baudrate und die gleichen Schnittstellenparameter eingestellt.

Wenn der Slave noch immer nicht antwortet, gehen Sie wie folgt vor:

- ▶ Öffnen Sie den Steckergehäusedeckel.
- ▶ Bei einem einwandfrei funktionierenden Slave mit deaktivierter Endstufe blinkt die LED im Steckergehäuse konstant mit 0,5 Hz (1 Sekunde ein, 1 Sekunde aus). Wenn dies nicht der Fall ist, liegt ein Betriebsfehler am Slave vor. Informationen zu Fehlern und zur Fehlerbehebung finden Sie im Gerätehandbuch.
- ▶ Vergleichen Sie die Anzeige der LED mit den Angaben in der folgenden Tabelle.

Fehler	Fehlerklasse	Fehlerursache	Fehlerbehebung
LED dunkel	–	Versorgungsspannung fehlt.	Versorgungsspannung und Sicherungen prüfen.
LED blinkt mit 0,5 Hz(1 s ein, 1 s aus)	.–	Firmware arbeitet fehlerfrei, Endstufe inaktiv.	Kabelverbindungen überprüfen. Einstellungen der DIP-Schalter überprüfen.
LED blinkt mit 6 Hz.	4	Flash-Prüfsumme falsch.	Firmware neu aufspielen.Slave austauschen.
LED blinkt mit 10 Hz.Watchdog	4	Hardware-Fehler	Slave aus- und wieder einschalten.Slave austauschen.

Weitere Informationen zu Fehlerursachen und Fehlerbehebung finden Sie im Gerätehandbuch.

5.4 SyCon CANopen Konfigurationssoftware

Mit der Konfigurationssoftware "SyCon" kann das CANopen Netzwerk konfiguriert werden. Auf der Produkt CD ist ein weiteres EDS File im Unterverzeichnis SYCON vorhanden.

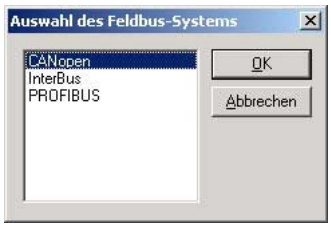
- ▶ Führen Sie hierzu folgende Schritte aus:

0198441113585, V2.01, 11.2008

5.4.1 Neues Netzwerk erzeugen

Über den Menüpunkt "Datei - Neu" wird ein neues Netzwerk erzeugt.

- ▶ Wählen Sie CANopen als Feldbus Netzwerk aus.
- ▶ Bestätigen Sie ihre Auswahl mit "OK".

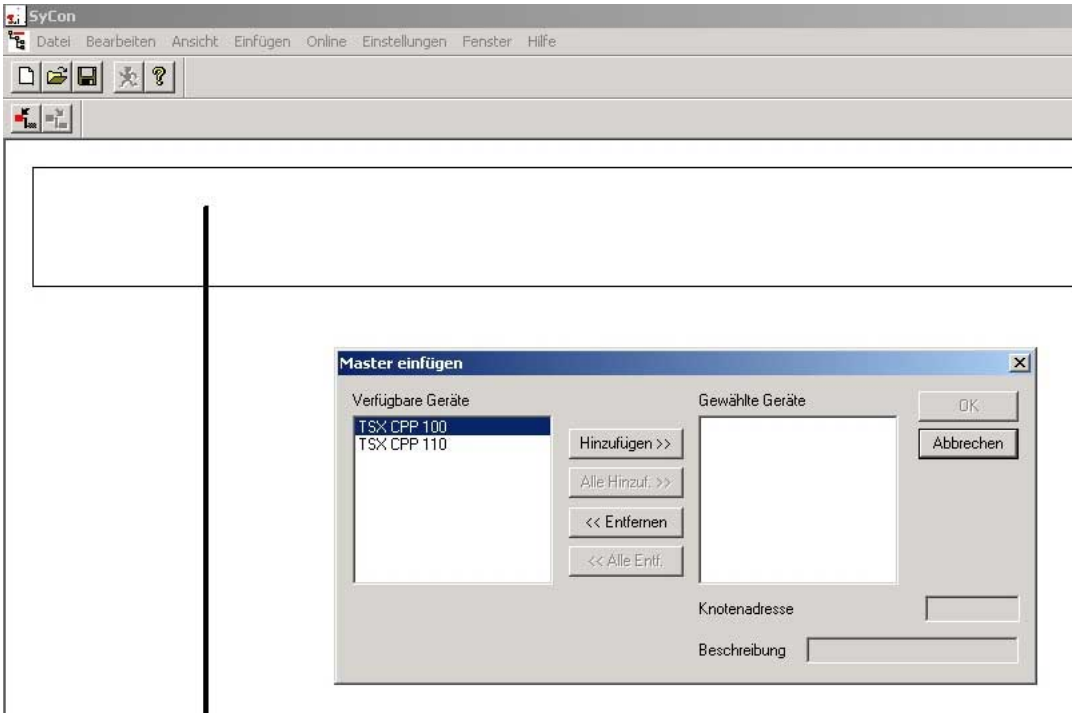


5.4.2 Auswahl des CANopen Master

Über den Menüpunkt "Einfügen - Master" wählen Sie den Netzwerkmaster aus. In dem Beispiel wird die TSX CPP 110 Karte einer Premium SPS verwendet.

Knotenadresse und eine Kurzbeschreibung können direkt eingegeben werden.

- ▶ Bestätigen Sie ihre Auswahl mit "OK".

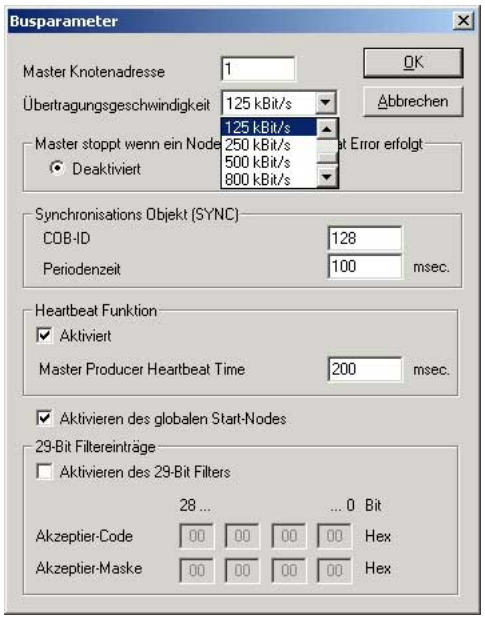


0198441113585, V2.01, 11.2008

5.4.3 Einstellen der Busparameter

Über den Menüpunkt "Einstellungen - Busparameter..." werden die CA-Nopen Kommunikationsparameter eingestellt. Bitte lesen Sie hierzu auch die Bedienungsanleitung der Konfigurationssoftware SyCon.

▶ Bestätigen Sie ihre Auswahl mit "OK".

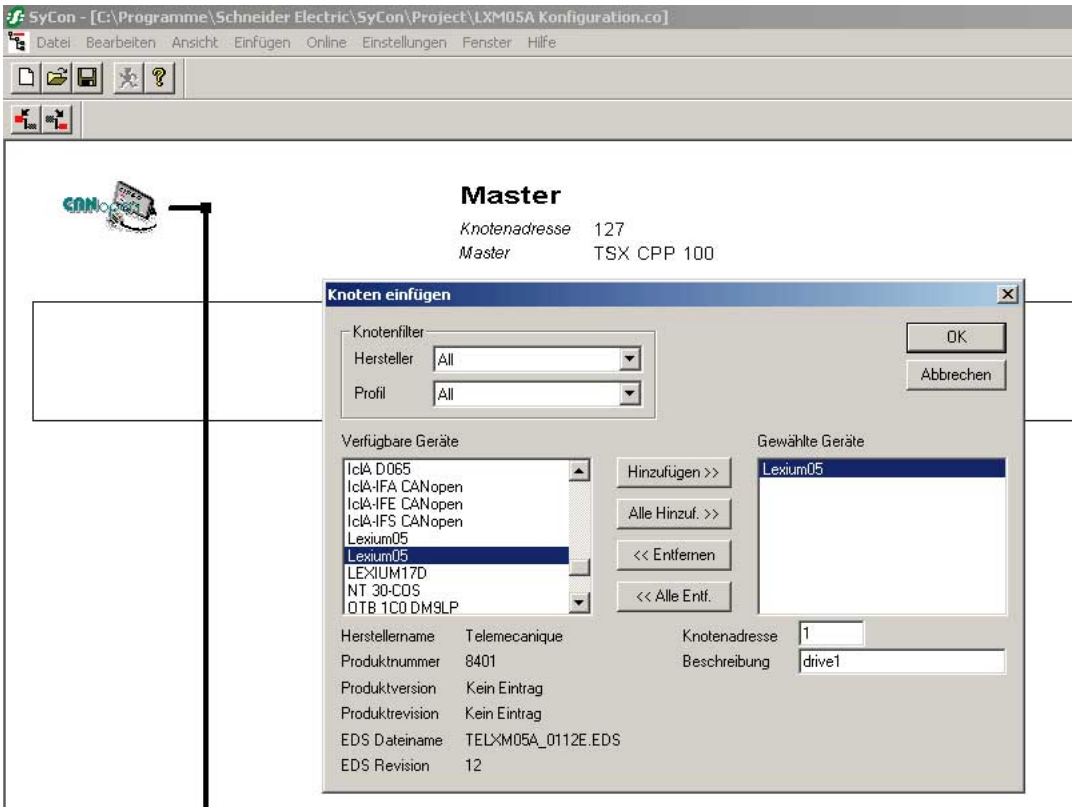


0198441113585, V2.01, 11.2008

5.4.4 Auswahl und Einfügen der Knoten

Über den Menüpunkt "Einfügen - Knoten" wählen Sie die Netzwerkknoten aus. In dem Beispiel wird ein Lexium05 verwendet.

▶ Bestätigen Sie ihre Auswahl mit "OK".



0198441113585, V2.01, 11.2008

6 Betrieb

⚠ WARNUNG

UNBEABSICHTIGTER BETRIEB

- Schreiben Sie nicht in reservierte Parameter.
- Schreiben Sie nicht in Parameter bevor Sie die Funktion nicht verstanden haben. Weitere Informationen finden Sie im Produkthandbuch.
- Führen Sie erste Tests ohne angekoppelte Lasten durch.
- Vergewissern Sie sich, dass die Anlage frei und bereit für die Bewegung ist, bevor Sie Parameter ändern.
- Überprüfen Sie bei der Feldbus-Kommunikation die Verwendung der Bits: Bit 0 steht ganz rechts (least significant). Bit 15 steht ganz links (most significant).
- Überprüfen Sie bei der Feldbus-Kommunikation die Verwendung der Wortfolge.
- Stellen Sie keine Feldbus-Verbindung her, bevor Sie nicht die Kommunikations-Prinzipien verstanden haben.

Nichtbeachtung dieser Vorkehrungen kann zu Tod, schwerwiegenden Verletzungen oder Materialschäden führen.

6.1 Übersicht

Die Programmbeispiele zeigen praktische Anwendungen für den Netzwerkeinsatz von Slaves. Generell gibt es über den CANopen-Feldbus 2 Zugriffsmethoden: SDO "Service Data Objects" und PDO "Process Data Objects".

Verwendung von SDO

Ein SDO-Zugriff ist ein Schreib- oder Lesezugriff auf einen einzelnes Objekt. Die verfügbaren Objekte sind im Gerätehandbuch beschrieben und dort auch als Tabelle im Kapitel "Parameter" zusammengefasst. Die Verwendung von SDO wird in diesem Kapitel nur für wenige Objekte exemplarisch beschrieben, da diese Art der Kommunikation einheitlich für alle vorhandenen Anwenderobjekte verwendet werden kann und sehr ähnlich aufgebaut ist.

Verwendung von PDO

Für den eigentlichen Positionierbetrieb wird empfohlen PDOs zu verwenden, da hier die Informationen wesentlich effektiver übertragen werden. Für die Anwendung der vom Slave unterstützten PDO4 werden verschiedene praxisnahe Beispiele gezeigt und das generelle Vorgehen beschrieben.

- die PDO vom Master zum Slave ist mit "R_PDO" bezeichnet
- die PDO vom Slave zum Master ist mit "T_PDO" bezeichnet.

0198441113585, V2.01, 11.2008

Aufbau der Beispiele

Bei der Beschreibung der PDOs muss beachtet werden dass diese aus Sicht des Slaves beschrieben sind:

Dargestellt werden:

- Aufgabenbeschreibung
- Startbedingungen
- Erforderliche Befehle im Sendedatenrahmen
- Reaktion des Slaves im Empfangsdatenrahmen
- Mögliche Einschränkungen für die Befehlsausführung.

Um die Beispiele nachvollziehen zu können, sollte Ihnen folgendes bekannt sein:

- Bedienkonzept und Funktionsumfang des Slaves. Informationen darüber finden Sie in Ihrer Gerätedokumentation.
- Feldbusprotokoll und Anbindung an die Mastersteuerung
- Funktionsumfang des Feldbusprofils.

Gerätehandbuch

Die Beispiele sind als Ergänzung zu den Funktionsbeschreibungen der Gerätehandbücher angelegt. Die grundlegende Funktionsweise der Betriebsarten und Funktionen finden Sie im Gerätehandbuch beschrieben.

Ebenso finden Sie dort alle Parameter zu den Betriebsarten und Funktionen aufgelistet.

Das Zahlenformat der Parameterwerte in einem Feldbusbefehl ist in Tabelle 9.2, Seite 9-1 des Gerätehandbuchs beschrieben.

0198441113585, V2.01, 11.2008

6.2 Verwendung von SDO-Befehle

6.2.1 Parameter schreiben

Aufgabe Es soll der Parameter `Motion.acc`, 29:26 (Beschleunigung) auf den Wert 10.000 gesetzt werden.

Dazu sind Index und Subindex in die hexadezimale Darstellung umzurechnen und für den SDO-Zugriff zum Index die Konstante 3000_h zu addieren:

- Index:29 = $1D_h$ + 3000_h = $301D_h$
- Subindex: 26 = $1A_h$
- Wert:10000 = 00002710_h

Als CCD (Client Command Specifier) ist der Wert 23_h einzutragen, da der Parameter einen 32-Bit-Datentyp besitzt.

Sendedaten

Objekt	COB-ID	CCD	Idx	Sdx	Daten	Beschreibung
Tx $301D_h$:$1A_h$ Motion.acc	600_h+ID	23_h	$1D_h$ 30_h	$1A_h$	10_h 27_h 00 00	Setzen der Beschleunigung auf 10000 min^{-1}*s = 2710_h als 32-Bit-Wert

Der Datentyp des zu schreibenden Wertes können Sie der Spalte "Datentyp" in der Parameterbeschreibung des Gerätehandbuchs entnehmen. Beim verwendeten CAN-Protokoll werden 16-Bit-Werte und 32-Bit-Werte im Format "niederwertigstes Byte zuerst – höchstwertigstes Byte zuletzt" übertragen. Bei der Übergabe eines INT16- oder UINT16-Wertes muss der dem Datentyp entsprechende CCD eingetragen werden. Der Wert muss in den ersten beiden Datenbytes abgelegt werden, die letzten beiden Datenbytes sind mit "0" zu beschreiben.

Empfangsdaten

Objekt	COB-ID	CCD	Idx	Sdx	Daten	Beschreibung
Rx $301D_h$:$1A_h$ Motion.acc	580_h+ID	60_h	$1D_h$ 30_h	$1A_h$	XX XX XX XX	Die Antwortdaten haben keine Bedeutung.

0198441113585, V2.01, 11.2008

6.2.2 Parameter lesen

Aufgabe

Es soll der Parameter `Status.n_act, 31:9` (Ist-Drehzahl) gelesen werden.

Dazu sind Index und Subindex in hexadezimale Darstellung umzurechnen und für den SDO-Zugriff zum Index die Konstante 3000_h zu addieren:

- Index:31 = $1F_h + 3000_h = 301F_h$
- Subindex 9 = 09_h

Als CCD ist der Wert "40_h" einzutragen. Dieser kennzeichnet eine Leseanforderung "Read Request".

Sendedaten

Objekt	COB-ID	CCD	Idx	Sdx	Daten	Beschreibung
Tx $301F_h$:09_h Status.n_actT	600_h+ID	40_h	$1F_h$ 30_h	09_h	XX XX XX XX	Lesen der Ist-Drehzahl. Die Daten haben keine Bedeutung.

Die 4 Datenbytes haben bei einer Leseanforderung keine Bedeutung.

Empfangsdaten

Objekt	COB-ID	CCD	Idx	Sdx	Daten	Beschreibung
Rx $301D_h$:09_h Status.n_act	580_h+ID	43_h	$1F_h$ 30_h	09_h	E8 03 00 00	Die Daten 000003E8 entsprechen 1000 min^{-1}.

Der Slave sendet Daten als 32-Bit-Werte an den Master zurück (CCD ist "43_h"). Er sendet auch Daten, die im Gerätehandbuch als INT16- oder UINT16-Datentypen beschrieben sind, als 32-Bit-Werte zurück. Beim Lesen eines INT16- oder UINT16-Wertes können daher alle 4 Datenbytes ausgewertet werden. Bei 16-Bit-Daten ist es jedoch auch korrekt, nur die ersten beiden Datenbytes auszuwerten und die letzen beiden Datenbytes zu ignorieren.

6.2.3 Synchrone Fehler

Empfangsdaten mit Fehlerdatenrahmen "Error Response"

Wenn ein SDO-Schreib- oder Lesebefehl fehlschlägt, antwortet der Slave mit einem Fehlerdatenrahmen "Error Response". Eine Fehlerquelle kann z. B. sein, dass Sie versuchen, ein nicht existierendes Objekt zu lesen oder zu schreiben. Die übertragene Fehlernummer gibt Auskunft über die genaue Ursache.

Objekt	COB-ID	CCD	Index	Sub	Daten	Beschreibung
Rx 3028_h:20_h	580_h+ID	80_h	28_h 30_h	20_h	00 00 02 06	Fehlernummer 06020000h bedeutet "Objekt im Objektverzeichnis nicht vorhanden"

Das Beispiel zeigt die Antwort auf eine Schreib- oder Leseanforderung für ein nicht existierendes Objekt `40:32`.

Die Fehlernummer einer synchronen Fehlermeldung wird als UINT16-Wert abgelegt und der entsprechende CCD (Error Response) der Wert 80_h übergeben. Die Tabelle der Fehlernummern finden Sie im Kapitel 7 "Diagnose und Fehlerbehebung".

0198441113585, V2.01, 11.2008

6.3 Betriebszustände wechseln mit PDO4

Der Slave kennt verschiedene Betriebszustände. Den einzelnen Betriebszuständen sind Nummern von 1 bis 9 zugeordnet. Die Betriebszustände und die Übergangsbedingungen sind im Gerätehandbuch, Kapitel "Grundlagen" und "Betrieb" genauer beschrieben.

Betriebs-zustand	Name	Endstufe	Beschreibung
4	Ready To Switch On	aus	passiver Betriebszustand, Motor stromlos
6	Operation Enable	ein	aktiver Betriebszustand, Motor bestromt
7	Quick-stop Active	ein	Fehlerzustand, Endstufe bleibt aktiviert
9	Fault	aus	Fehlerzustand, Endstufe wird deaktiviert

Tabelle 6.1 Wichtige Betriebszustände

Anforderungen für den Wechsel von Betriebszuständen werden im R_PD04 im Feld `drivectrl` dem Slave übergeben. Dieser meldet den aktuellen Betriebszustand im T_PDO4, Feld driveStat, zum Master zurück.

Tabelle 6.2 zeigt die Bit-Belegung des Feldes `drivectrl` im Objekt R_PD04:

Bit Nr.	Wertigkeit	Bedeutung
0	01_h	Disable
1	02_h	Enable
2	04_h	Quickstop
3	08_h	Fault-Reset
4	10_h	Quickstop-Release

Tabelle 6.2 R_PDO4, drivectrl, Bit-Belegung

6.3.1 Endstufe aktivieren und deaktivieren

Durch den Übergang von Betriebszustand 4 nach 6 wird die Endstufe aktiviert. Dazu sind im R_PD04 die beiden Bits `Enable` und `Disable` vorhanden.

Endstufe einschalten

Vorbedingung: Slave ist im Betriebszustand 4.

Zum Einschalten der Endstufe muss in `drivectrl`, Bit 1 (Enable) eine "0 -> 1"-Flanke erzeugt werden. Dies kann durch Löschen von Bit 0 (Disable) und Setzen von Bit 1 durchgeführt werden. Daraufhin wartet der Master, bis der Slave den Betriebszustand 6 meldet. Dies kann einige Zeit dauern (ca. 1 Sekunde), da beim Einschalten der Endstufe im Slave verschiedene Tests durchgeführt werden.

0198441113585, V2.01, 11.2008

Beispiel

	Master <---> Slave	
Disable ist angefordert	--->	drivectrl01_h
Slave meldet Betriebszustand 4	<---	driveStat $XXX4_h$
Enable anfordern	--->	drivectrl02_h
Slave meldet Betriebszustand 5	<---	driveStat $XXX5_h$
Slave meldet Betriebszustand 6	<---	driveStat $XXX6_h$

Endstufe ausschalten

Vorbedingung: Slave ist im Betriebszustand 6 oder 7.

Zum Ausschalten der Endstufe muss in drivectrl, Bit 0 (Disable) eine "0 -> 1"-Flanke erzeugt werden. Dies kann durch Setzen von `Bit 0` (Disable) und Löschen von Bit 1 (Enable) durchgeführt werden. Der Slave wechselt daraufhin in den Betriebszustand 4.

Beispiel

	Master <---> Slave	
Enable ist angefordert	--->	drivectrl 02_h
Slave meldet Betriebszustand 6	<---	driveStat $XXX6_h$
Disable anfordern	--->	drivectrl 01_h
Slave meldet Betriebszustand 4	<---	driveStat $XXX4_h$

0198441113585, V2.01, 11.2008

6.3.2 "Quick Stop" auslösen

Einen laufenden Fahrauftrag können Sie jederzeit über den Feldbus dem Befehl Quick Stop abrechen. Dieses wird durch eine "0 -> 1"-Flanke in drivectrl, Bit 2 ausgelöst. Beim Wechsel in den Betriebszustand 7 (Quick Stop) bremst der Slave mit der eingestellten NOT-HALT-Rampe ab und kommt zum Stehen.

Um einen neuen Fahrauftrag zu starten, müssen Sie den Slave in den Betriebszustand 6 bringen. Führen Sie dazu einen der folgenden Schritte aus:

- Fault Reset"0 -> 1"-Flanke in `drivectrl`, Bit 3
- Quick Stop Release"0 -> 1"-Flanke in `drivectrl`, Bit 4

Beispiel

	Master <---> Slave	
"Enable" ist angefordert	--->	drivectrl 02_h
Slave meldet Betriebszustand 6	<---	driveStat $XXX6_h$
"Quick Stop" und "Enable" anfordern	--->	drivectrl 06_h
Slave meldet Betriebszustand 7	<---	driveStat $XXX7_h$
Warten bis Slave steht und Anlage weiterlaufen soll	--->	
"Quick Stop Release" und "Enable" anfordern	<---	drivectrl 12_h
Slave meldet Betriebszustand 6	--->	driveStat $XXX6_h$
"Quick Stop Release" zurücknehmen	<---	drivectrl 02_h

6.3.3 Fehler zurücksetzen

Tritt im Betrieb ein Fehler auf, so wechselt der Slave in Abhängigkeit vom aufgetretenen Fehler in Betriebszustand 7 "Quick Stop" oder in Betriebszustand 9 "Fault".

Nach der Fehlerbehebung können Sie den Fehlerzustand zurücksetzen, in dem Sie ein Fault Reset ("0 -> 1"-Flanke in `drivectrl`, Bit 3) durchführen.

Befand sich der Slave ursprünglich in Betriebszustand 7, wechselt er nach dem "Fault Reset" in Betriebszustand 6.

Befand sich der Slave ursprünglich in Betriebszustand 9, wechselt er nach dem "Fault Reset" in Betriebszustand 4. Anschließend müssen Sie eine "0 -> 1"-Flanke in `drivectrl`, Bit 1 "Enable" senden, um die Endstufe zu aktivieren.

0198441113585, V2.01, 11.2008

Beispiel

	Master <---> Slave	
Enable anfordern	--->	drivectrl 02_h
Slave meldet Betriebszustand 9 (Fault)	<---	driveStat $XXX9_h$
Fehler beheben		
"Fault Reset" anfordern	--->	drivectrl 08_h
Slave meldet Betriebszustand 4	<---	driveStat $XXX4_h$
"Enable" anfordern	--->	drivectrl 02_h
Slave meldet Betriebszustand 5	<---	driveStat $XXX5_h$
Slave meldet Betriebszustand 6	<---	driveStat$XXXX6_h$

Tabelle 6.3 Endstufe ausschalten

Anmerkung: In diesem Beispiel löscht der Master beim "Fault Reset" das `Bit 1` "Enable", um danach implizit eine "0 -> 1"-Flanke auf `Bit 1` durchführen zu können. Dadurch wechselt der Slave wieder in Betriebszustand 6.

6.4 Beispiele zu den Betriebsarten mit PDO4

R_PD04

Mithilfe des `R_PD04` können Sie Fahrbefehle starten und während der laufenden Bearbeitung ändern.

Im `R_PD04` stehen Ihnen dafür 3 Felder zur Verfügung:

- `modeCtrl` Betriebsart starten und ändern
- "Ref16" und "Ref32" Betriebsartenabhängige Vorgabewerte

Die Vorgabewerte dieser 3 Felder werden vom Slave erst übernommen, wenn `modeCtrl`, Bit 7 (`ModeToggle`) gewechselt wurde.

Für die Übergabe von Werten an den Slave müssen Sie wie folgt vorgehen:

▶ Tragen Sie die gewünschte Betriebsart und die zugehörigen Vorgabewerte in die Felder `modeCtrl`, "Ref16" bzw. "Ref32" ein.

▶ Wechseln Sie `modeCtrl`, Bit 7 (`ModeToggle`)

Damit werden Konsistenzprobleme innerhalb der R_PDO4 umgangen.

T_PD04

Mithilfe des `T_PD04` überwachen Sie Fahraufträge.

Im T_PDO4 stehen Ihnen dafür 3 Felder zur Verfügung:

- `modeStat` Für Handshake-Zwecke
- `driveStat` Meldet Fahrzustand und Fehler
- `p_act`Ist-Position des Slaves

0198441113585, V2.01, 11.2008

ModeToggle Im R_PD04 und im T_PD04 gibt es jeweils das Bit ModeToggle. Der Master gibt dieses Bit im R_PD04 vor und der Slave spiegelt es im T_PD04. Durch dieses Verfahren kann der Master erkennen, ob die vom Slave übermittelten Daten aktuell sind.

Beispiel Der Master startet eine Positionierung, die nur eine sehr kurze Zeit in Anspruch nimmt. Der Master wartet auf das Ende der Positionierung, indem er T_PD04 auf Bit x_end = 1 (kennzeichnet Positionierende) prüft.

Dabei kann es vorkommen, dass der Master vom Slave Daten erhält, die noch aus der Zeit vor dem Start der Positionierung stammen. Diese enthalten ebenfalls x_end = 1. Jetzt erkennt der Master, dass diese Daten veraltet sind, da das erhaltene Bit ModeToggle nicht mit dem seines Positionierauftrags übereinstimmt.

Generell sollte der Master nur Daten auswerten, bei denen das empfangene Bit ModeToggle identisch ist mit dem zuletzt von ihm gesendeten.

Beschleunigung Vor einer Positionierung können Sie zuerst die gewünschte Beschleunigung über einen SDO-Zugriff einstellen (Objekt Motion.acc, 29:26). Beachten Sie, dass Sie die Beschleunigung nur bei Stillstand des Slaves verändern können.

Annahmen Den Beispielen dieses Kapitels liegen folgende Annahmen zugrunde:

- Betriebszustand 6 "Operation Enable"
- Slave wurde noch nicht referenziert (Bit ref_ok = 0)
- p_act = 0 (Ist-Position)
- R_PD04: modeCtrl, Bit 7 = 0 (ModeToggle)

6.4.1 Betriebsart Punkt-zu-Punkt – Absolut-Positionierung

Zum Starten einer Absolut-Positionierung müssen Sie im R_PD04 folgende Einstellungen vornehmen:

▶ Tragen Sie in "Ref16" die Soll-Geschwindigkeit und in "Ref32" die Ziel-Position ein.

▶ Tragen Sie im Feld modeCtrl die Betriebsart 03_h (Betriebsart Punkt-zu-Punkt, Absolut-Positionierung) ein.

▶ Wechseln Sie modeCtrl, Bit 7, damit die Daten vom Slave übernommen werden.

0198441113585, V2.01, 11.2008

Beispiel Absolut-Positionierung auf Position 100.000 ($000186A0_h$)
mit einer Soll-Geschwindigkeit von 1000 min^{-1} ($03E8_h$)

			Master <---> Slave			
Positionierung auslösen	R_PDO4	--->	drivectrl 02_h	modeCtrl 83_h	Ref16 $03E8_h$	Ref32 $000186A0_h$
Positionierung läuftx_err = 0, x_end = 0	T_PDO4	<---	driveStat 0006_h	modeStat 83_h		p_act $XXXXXXXX_h$
Positionierung beendetx_err = 0, x_end = 1, x_info = 1	T_PDO4	<---	driveStat 6006_h	modeStat 83_h		p_act $000186A0_h$

Tabelle 6.4 Betriebsart Punkt-zu-Punkt, Absolut-Positionierung mit konstanter Soll-Geschwindigkeit

Anmerkung: Der Datenrahmen "Positionierung läuft" kann auch mehrfach übertragen werden, im Feld `p_act` steht jeweils die aktuelle Ist-Position.

Beispiel Wie vorhergehendes Beispiel, jedoch wird die Soll-Geschwindigkeit während der Fahrt auf 2000 min^{-1} ($07D0_h$) geändert.

			Master <---> Slave			
Positionierung auslösen	R_PDO4	--->	drivectrl 02_h	modeCtrl 83_h	Ref16 $03E8_h$	Ref32 $000186A0_h$
Positionierung läuftx_err = 0, x_end = 0	T_PDO4	<---	driveStat 0006_h	modeStat 83_h		p_act $XXXXXXXX_h$
Soll-Geschwindigkeit ändern	R_PDO4	--->	drivectrl 02_h	modeCtrl 03_h	Ref16 $07D0_h$	Ref32 $000186A0_h$
Positionierung läuftx_err = 0, x_end = 0	T_PDO4	<---	driveStat 0006_h	modeStat 03_h		p_act $XXXXXXXX_h$
Positionierung beendet xerr=0, x_end = 1, x_info = 1	T_PDO4	<---	driveStat 6006_h	modeStat 03_h		p_act $000186A0_h$

Tabelle 6.5 Betriebsart Punkt-zu-Punkt, Absolut-Positionierung mit Änderung der Soll-Geschwindigkeit

Anmerkung: Der Datenrahmen "Positionierung läuft" kann auch mehrfach übertragen werden. Im Feld `p_act` steht jeweils die Ist-Position. Bei Änderung der Soll-Geschwindigkeit wird die gleiche Ziel-Position übergeben, da diese sich in diesem Beispiel nicht ändert.

0198441113585, V2.01, 11.2008

6.4.2 Betriebsart Punkt-zu-Punkt – Relativ-Positionierung

Eine Relativ-Positionierung läuft ähnlich wie die Absolut-Positionierung ab. Sie müssen lediglich in Feld `modeCtrl` den Wert 13_h (Betriebsart Punkt-zu-Punkt, Relativ-Positionierung) eintragen. Außerdem müssen Sie beachten, dass mehrfach nacheinander übergebene relative Ziel-Positionen addiert werden.

Beispiel: Relativ-Positionierung um 100.000 ($000186A0_h$) Inkremente mit einer Geschwindigkeit von 1000 min^{-1} ($03E8_h$)

Die Geschwindigkeit soll während der Fahrt auf 2000 min^{-1} ($07D0_h$) geändert werden.

		Master <---> Slave				
Positionierung auslösen	R_PDO4	--->	drivectrl 02_h	modeCtrl 93_h	Ref16 $03E8_h$	Ref32 $000186A0_h$
Positionierung läuftx_err = 0, x_end = 0	T_PDO4	<---	driveStat 0006_h	modeStat 83_h		p_act $XXXXXXXX_h$
Soll-Geschwindigkeit ändern Relativ-Position "0" übergeben	R_PDO4	--->	drivectrl 02_h	modeCtrl 13_h	Ref16 $07D0_h$	Ref32 00000000_h
Positionierung läuftx_err = 0, x_end = 0	T_PDO4	<---	driveStat 0006_h	modeStat 03_h		p_act $XXXXXXXX_h$
Positionierung beendetx_err = 0, x_end = 1, x_info = 1	T_PDO4	<---	driveStat 6006_h	modeStat 03_h		p_act $000186A0_h$

Tabelle 6.6 Betriebsart Punkt-zu-Punkt, Relativ-Positionierung mit Änderung der Soll-Geschwindigkeit

Anmerkungen: Der Datenrahmen "Positionierung läuft" kann auch mehrfach übertragen werden, im Feld `p_act` steht jeweils die Ist-Position. Bei Änderung der Soll-Geschwindigkeit muss hier als neue Zielposition der Wert "0" übergeben werden, da der neue Wert zur bereits berechneten Ziel-Position addiert wird.

0198441113585, V2.01, 11.2008

6.4.3 Betriebsart Geschwindigkeitsprofil

In der Betriebsart Geschwindigkeitsprofil wird dem Motor eine Soll-Geschwindigkeit vorgegeben und eine Bewegung ohne Ziel-Position gestartet.

Zum Starten des Geschwindigkeitsprofils oder zum Ändern der Soll-Geschwindigkeit bei aktiver Betriebsart Geschwindigkeitsprofils müssen Sie im `R_PDO4` folgende Einstellungen vornehmen:

- Tragen Sie in `Ref16` die Soll-Geschwindigkeit. (`Ref32` ist hier ohne Bedeutung)
- Tragen Sie in `modeCtrl` die Betriebsart 04_h (Betriebsart Geschwindigkeitsprofil) ein
- Wechseln Sie `modeCtrl`, Bit 7, damit die Daten vom Slave übernommen werden

Beispiel Das Geschwindigkeitsprofil wird mit einer Soll-Geschwindigkeit von 1000 min^{-1} ($03E8_h$) gestartet.

Die Soll-Geschwindigkeit wird während der Fahrt auf 2000 min^{-1} ($07D0_h$) geändert.

			Master <---> Slave			
Geschwindigkeitsprofil mit 1000 min^{-1} starten	R_PDO4	--->	drivectrl 02_h	modeCtrl 84_h	Ref16 $03E8_h$	Ref32 $XXXXXXXX_h$
Slave beschleunigt xerr=0, xend=0, xinfo=0	T_PDO4	<---	driveStat 0006_h	modeStat 84_h		p_act $XXXXXXXX_h$
Sollgeschwindigkeit erreicht xerr=0, xend=0, xinfo=1	T_PDO4	<---	driveStat 2006_h	modeStat 84_h		p_act $XXXXXXXX_h$
Geschwindigkeit auf 2000 min^{-1} ändern	R_PDO4	--->	drivectrl 02_h	modeCtrl 04_h	Ref16 $07D0_h$	Ref32 $XXXXXXXX_h$
Slave beschleunigt xerr=0, xend=0, xinfo=0	T_PDO4	<---	driveStat 0006_h	modeStat 04_h		p_act $XXXXXXXX_h$
Sollgeschw. erreicht xerr=0, xend=0, xinfo=1	T_PDO4	<---	driveStat 2006_h	modeStat 04_h		p_act $XXXXXXXX_h$
Geschwindigkeit auf 0 min^{-1} ändern	R_PDO4	--->	drivectrl 02_h	modeCtrl 84_h	Ref16 0000_h	Ref32 $XXXXXXXX_h$
Slave verzögert xerr=0, xend=0, xinfo=0	T_PDO4	<---	driveStat 0006_h	modeStat 84_h		p_act $XXXXXXXX_h$
Geschw.betrieb beendet xerr=0, xend=1, xinfo=1	T_PDO4	<---	driveStat 6006_h	modeStat 84_h		p_act $XXXXXXXX_h$

Das Geschwindigkeitsprofil wird durch Übergabe der Soll-Geschwindigkeit "0" beendet und der Stillstand des Slaves abgewartet.

Anmerkung: Im Feld `p_act` des `T_PDO4` steht die aktuelle Position des Antriebs in Inkrementen.

0198441113585, V2.01, 11.2008

6.4.4 Maßsetzen

Beim Maßsetzen wird der aktuellen Motorposition eine neue Position zugewiesen. Dabei wird lediglich das Koordinatensystem verschoben, der Motor bewegt sich nicht.

Zum Maßsetzen müssen Sie im R_PD04 folgende Einstellungen vornehmen:

- Tragen Sie in Ref32 in die neue Position ein. (Ref16 ist hier ohne Bedeutung)
- Tragen Sie in modeCtrl die Betriebsart 02_h ("Referenzierung", "Maßsetzen") ein.
- Wechseln Sie modeCtrl, Bit 7, damit die Daten vom Slave übernommen werden

Beispiel: Der Motor steht auf Position -100.000 ($FFFE7960_h$).

Dem Motor wird die Position 200.000 zugewiesen ($00030D40_h$).

			Master <---> Slave			
Slave meldet Position-100.000	T_PDO4	<---	driveStat $XXXX_h$	modeStat XX_h		p_act $FFFE7960_h$
Maßsetzen auf 200.000	R_PDO4	--->	drivectrl 02_h	modeCtrl 82_h	Ref16 $XXXX_h$	Ref32 $00030D40_h$
Position übernommen x_err = 0, x_end = 1, x_info = 0	T_PDO4	<---	driveStat 4006_h	modeStat $A2_h$		p_act $00030D40_h$

Tabelle 6.7 Maßsetzen

0198441113585, V2.01, 11.2008

6.4.5 Betriebsart Referenzierung

Bei der Referenzfahrt wird ein End- oder Referenzschalter angefahren und anschließend dieser Position ein neuer Wert zugewiesen.

Vor dem Starten einer Referenzfahrt müssen die Parameter durch SDO-Schreibzugriffe den Anforderungen entsprechend eingestellt werden. Weitere Informationen zur Parametrierung sowie zum Ablauf einer Referenzfahrt finden Sie im Gerätehandbuch.

Zum Starten einer Referenzfahrt müssen Sie im `R_PDO4` folgende Einstellungen vornehmen:

- Tragen Sie in `Ref16` den Typ der Referenzfahrt ein(`Ref32` ist hier ohne Bedeutung).

  Die Typen der Referenzfahrt sind im Gerätehandbuch beschrieben.

- Tragen Sie in `modeCtrl` die Betriebsart 12_h "Referenzierung" ein.
- Wechseln Sie `modeCtrl`, Bit 7, damit die Daten vom Slave übernommen werden

Beispiel Es soll eine Referenzfahrt auf den negativen Endschalter (LIMN) durchgeführt werden, dies ist Referenzfahrt Typ 2.

			Master <---> Slave			
Referenzfahrt auslösen	R_PDO4	--->	drivectrl 02_h	modeCtrl 92_h	Ref16 0002_h	Ref32 $XXXXXXXX_h$
Referenzfahrt läuftxerr=0, xend=0	T_PDO4	<---	driveStat 0006_h	modeStat8 2_h		p_act $XXXXXXXX_h$
Referenzfahrt beendet xerr=0, xend=1	T_PDO4	<---	driveStat 4006_h	modeStat $A2_h$		p_act 00000000_h

Tabelle 6.8 Referenzfahrt

0198441113585, V2.01, 11.2008

6.5 Fehlersignalisierung über PDO4

6.5.1 Synchrone Fehler

Kann eine über R_PDO4 gesendete Anforderung einer Betriebsart vom Slave nicht verarbeiten werden, lehnt der Slave die Bearbeitung ab und setzt im T_PD04 `modeStat`, Bit 6 ("ModeError"). Die laufende Bearbeitung wird dabei nicht unterbrochen. Um die Ursache des Fehlers zu ermitteln kann der Master nun aus dem Objekt `CAN.modeError`, 30:11 mit einem SDO-Zugriff die Fehlernummer auslesen.

Beispiel Der Slave dreht im Geschwindigkeitsprofil.Es wird versucht, ein Maßsetzen durchzuführen.

			Master <---> Slave			
Geschwindigkeitsprofilx_end = 0	T_PDO4	<---	driveStat 0006_h	modeStat 04_h		p_act $XXXXXXXX_h$
Anforderung: Maßsetzen auf 0	R_PDO4	--->	drivectrl 02_h	modeCtrl 82_h	Ref16 $XXXX_h$	Ref32 00000000_h
Anforderung. abgelehnt "Mode-Error" = 1	T_PDO4	<---	driveStat 0006_h	modeStat $C4_h$		p_act XXXXXXXXh

Tabelle 6.9 Synchroner Fehler, Ungültige Anforderung einer Betriebsart

Anmerkung: Beim Ablehnen der Maßsetzen-Anforderung dreht der Slave im Geschwindigkeitsprofil unverändert weiter.

Der Slave sendet jedoch eine EMCY-Nachricht mit der entsprechenden Fehlernummer zum Master.

0198441113585, V2.01, 11.2008

6.5.2 Asynchrone Fehler

Asynchrone Fehler werden durch die interne Überwachung (z. B. Temperatur) oder durch die externe Überwachung (z. B. Endschalter) ausgelöst. Beim Auftreten eines asynchronen Fehlers reagiert der Slave durch Abbremsen bzw. durch Ausschalten der Endstufe.

Asynchrone Fehler werden wie folgt angezeigt:

- Wechsel in Betriebszustand 7 "Quick Stop" oder in Betriebszustand 9 "Fault".

  Der Wechsel wird in T_PDO4, `driveStat`, Bits 0 ... 3 angezeigt.

- Setzen von `driveStat`, Bit 5 (Störung durch interne Überwachung) oder `driveStat`, Bit 6 (Störung durch externe Überwachung)
- Bei einer Störungsmeldung durch die interne Überwachung:

  Setzen des der Störung entsprechenden Bits im Objekt `Status.FltSig_SR`, 28:18.

  Bei einer Störungsmeldung durch die externe Überwachung:Setzen des der Störung entsprechenden Bits im Objekt `Status.Sign_SR`, 28:15 das der Störung entsprechende Bit gesetzt

- Zusätzlich ist jedem Fehler auch eine Fehlernummer zugeordnet. Bei einem asynchronen Fehler kann die entsprechende Fehlernummer aus dem Objekt `Status.StopFault` (32:7) ausgelesen werden.

Beispiel: Auslösen einer Störungsmeldung durch die externe Überwachung; Fahrt auf den positiven Endschalter "LIMP".

			Master <---> Slave		
Positionierung läuft xerr=0, xend=0	T_PDO4	<---	driveStat 0006_h	modeStat 03_h	p_act $XXXXXXXX_h$
Endschalter erkannt xerr=1, xend=0	T_PDO4	<---	driveStat 8047_h	modeStat 03_h	p_act $XXXXXXXX_h$
Motor steht xerr=1, xend=1	T_PDO4	<---	driveStat $C047_h$	modeStat 03_h	p_act $XXXXXXXX_h$

Tabelle 6.10 Asynchroner Fehler, Auslösen eines externen

Anmerkung: Beim Erkennen des Endschalters wird der Motor mit der NOT-HALT-Rampe bis zum Stillstand abgebremst und das Bit `x_err` gesetzt. Nach Stillstand des Motors wird Bit `x_end` gesetzt.

0198441113585, V2.01, 11.2008

7 Diagnose und Fehlerbehebung

7.1 Fehlerdiagnose Feldbus-Kommunikation

Um Betriebs- und Fehlermeldungen auswerten zu können, ist es erforderlich, dass der Feldbusbetrieb funktioniert.

Anschlüsse zum Feldbusbetrieb

Kann das Gerät über den Feldbus nicht angesprochen werden, prüfen Sie zuerst die Anschlüsse. Im Produkthandbuch finden Sie die technischen Daten zum Gerät und Informationen zur Netzwerk- und Geräteinstallation. Prüfen Sie:

- $24V_{dc}$ Spannungsversorgung
- Versorgungsanschlüsse zum Gerät
- Feldbuskabel und Feldbusverdrahtung
- Netzwerkanschluss zum Gerät

Zur Fehlerdiagnose können Sie auch die Inbetriebnahmesoftware einsetzen.

Baudrate und Adresse

Wenn die Verbindung zu einem Teilnehmer nicht aufgenommen werden kann, prüfen Sie Baudrate und Knotenadresse.

- Die Baudrate aller Netzwerkteilnehmer muss gleich eingestellt sein
- Die Knotenadresse jedes Teilnehmers muss zwischen 1 und 127 liegen und für jeden Teilnehmer anders eingestellt sein

Zum Einstellen der Baudrate und Knotenadresse siehe Kapitel 5.2 "Adresse und Baudrate".

Funktionstest im Feldbus

Testen Sie nach korrekter Einstellung der Übertragungsdaten den Feldbusbetrieb. Dazu muss ein CAN-Konfigurationswerkzeug installiert sein, das CAN-Nachrichten anzeigt. Erfasst wird die Rückmeldung vom Gerät durch eine Boot-Up-Nachricht:

- Schalten Sie die Spannungsversorgung aus und wieder ein.
- Beobachten Sie die Netzwerk-Nachrichten nach dem Einschalten. Das Gerät sendet nach der Bus-Initialisierung eine Boot-Up-Nachricht (COB-Id 700_h + Node-ID und 1 Daten-Byte mit Inhalt 00_h).
- Mit Werkseinstellung der Knotenadresse auf 127 ($7F_h$) wird die Boot-Up-Nachricht über den Bus gesendet. Das Gerät kann anschließend über NMT-Dienste in Betrieb genommen werden.

Kann der Netzwerkbetrieb nicht aufgenommen werden, muss die Netzwerkfunktion des Gerätes von Ihrem lokalen Vertriebspartner geprüft werden. Setzen Sie sich mit Ihrem lokalen Vertriebspartner in Verbindung!

0198441113585, V2.01, 11.2008

7.2 Fehlerdiagnose über Feldbus

7.2.1 Meldungsobjekte

Mehrere Objekte informieren über den Betriebs- und Fehlerzustand:

- Objekt `Statusword (6041h)`
  Betriebszustände, siehe Produkthandbuch
- Objekt `EMCY (80h+ Node-ID)`
  Fehlermeldung eines Teilnehmers mit Fehlerzustand und Fehlercode, siehe Kapitel 3.4.2.5 "Emergency-Dienst"
- Objekt `Error register (1001h)`
  Fehlerzustand
- Objekt `Error code (603Fh)`
  Fehlercode des zuletzt aufgetretenen Fehlers
- Über die spezielle SDO-Fehlernachricht ABORT melden Teilnehmer den fehlerhaften Nachrichtenaustausch per SDO (engl. to abort, abbrechen)

7.2.2 Meldungen über den Gerätezustand

Zur Fehlerauswertung und -behandlung wird zwischen synchronen und asynchronen Fehlern unterschieden.

Synchrone Fehler

Das Gerät meldet einen synchronen Fehler direkt als Antwort auf eine nicht auswertbare Nachricht. Ursachen können z.B. eine fehlerhafte Übermittlung oder nicht zulässige Daten sein. Eine Liste der synchronen Fehler finden Sie im Kapitel 7.3.1 "Fehlerregister".

Asynchrone Fehler

Asynchrone Fehler werden von den Überwachungseinrichtungen des Gerätes gemeldet, sobald ein Gerätefehler auftritt. Ein asynchroner Fehler wird über Bit 3, Fault" des Objekts `statusword (6041h)` gemeldet. Bei Fehlern, die zu einer Fahrtunterbrechung führen, sendet das Gerät eine EMCY-Nachricht.

Asynchrone Fehler werden zusätzlich über die Bits 5..7 des Objekts `driveStat (2041h)` gemeldet.

0198441113585, V2.01, 11.2008

7.3 CANopen-Fehlermeldungen

CANopen-Fehlermeldungen werden über eine EMCY-Nachricht angezeigt. Ausgewertet werden sie über die Objekt `Error register` (1001_h) und `Error code` ($603F_h$). Informationen zum Objekt `EMCY` finden Sie im Kapitel 3.4.2.5 "Emergency-Dienst".

Fehler, die beim Datenaustausch per SDO auftreten, meldet CANopen durch die spezielle SDO-Fehlernachricht ABORT.

7.3.1 Fehlerregister

Das Objekt `Error register`(1001_h) zeigt den Fehlerzustand eines Teilnehmers bitcodiert an. Die genaue Fehlerursache muss über die Fehlercode-Tabelle ermittelt werden. Bit 0 wird gesetzt, sobald ein Fehler auftritt.

Bit	Meldung	Bedeutung
0	generic error	Ein Fehler ist aufgetreten
1	-	reserved
2	-	reserved
3	-	reserved
4	Communication	Fehler in der Netzwerk-Kommunikation
5	Device profile specific	Fehler bei Ausführung nach Geräteprofil
6	-	reserved
7	Manufacturer specific	Herstellerabhängige Fehlermeldung

7.3.2 Fehlercode-Tabelle

Der Fehlercode wird über das Objekt `error code` ($603F_h$), ein Objekt des Geräteprofils DSP402 ausgewertet und als vierstellige Hexadezimalzahl angegeben. Der Fehlercode gibt die Ursache der letzten Fahrtunterbrechung an. Die Bedeutung des Fehlercodes ist im Produkthandbuch im Kapitel zur Fehlerdiagnose und -behebung angegeben.

0198441113585, V2.01, 11.2008

7.3.3 SDO-Fehlernachricht ABORT

Eine SDO-Fehlernachricht wird als Antwort auf eine fehlerhafte SDO-Übertragung ausgegeben. Die Fehlerursache ist im `error code`, Byte 4 bis Byte 7 angegeben.

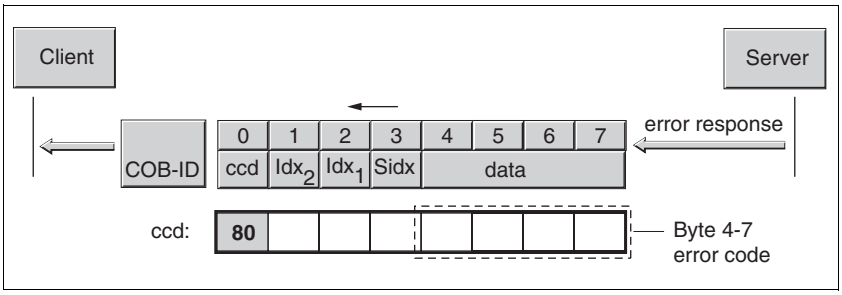


Bild 7.1 SDO-Fehlermeldung als Antwort auf eine SOD-Nachricht

Die folgende Tabelle zeigt alle Fehlermeldungen, die beim Datenaustausch mit dem Gerät auftreten können.

Fehler-Code	Bedeutung
0504 0000_h	Time-Out bei SDO-Transfer
0504 0001_h	Command specifier CS nicht korrekt oder unbekannt
0601 0000_h	Kein Zugriff auf Objekt möglich
0601 0001_h	Kein Lesezugriff, da Nur-Schreib-Objekt (wo)
0601 0002_h	Kein Schreibzugriff, da Lese-Objekt (ro)
0602 0000_h	Objekt nicht im Objektverzeichnis vorhanden
0604 0041_h	Objekt unterstützt kein PDO-Mapping
0604 0042_h	PDO-Mapping: Anzahl oder Länge der Objekte überschreiten die Byte-Länge des PDOs
0607 0010_h	Datentyp und Parameterlänge stimmen nicht überein
0607 0012_h	Datentyp stimmt nicht überein, Parameter zu lang
0607 0013_h	Datentyp stimmt nicht überein, Parameter zu kurz
0609 0011_h	Subindex nicht unterstützt
0609 0030_h	Wertebereich des Parameters zu groß (nur für Schreibzugriff relevant)
0609 0031_h	Parameterwerte zu groß
0609 0032_h	Parameterwerte zu klein
0800 0000_h	Genereller Fehler
0800 0022_h	Gerätestatus verhindert das Übertragen und Speichern der Daten.

0198441113585, V2.01, 11.2008

8 Objektverzeichnis

8.1 Übersicht

Dieses Objektverzeichnis beschreibt lediglich das Protokoll zum Slave laut CANopen DS301. Die Objekte zur Steuerung der Betriebsarten, Funktionen sowie alle Parameter finden Sie im Gerätehandbuch zum Slave.

8.1.1 Spezifikationen zu den Objekten

Index Der Index gibt die Position des Objekts im Objektverzeichnis an. Der Indexwert wird hexadezimal angegeben.

Objektcode Der Objektcode gibt die Datenstruktur des Objekts an.

Objektcode	Bedeutung	Codierung
VAR	Ein einfacher Wert, der z. B. vom Typ Integer8, Unsigned32 oder Visible String8 ist.	7
ARR (ARRAY)	Ein Datenfeld, bei dem jeder Eintrag vom gleichen Datentyp ist.	8
REC (RECORD)	Ein Datenfeld, das Einträge enthält, die eine Kombination einfacher Datentypen sind.	9

Datentyp	Wertebereich	Datenlänge
Boolean	0 = false, 1 = true	1 Byte
INT8	-128 .. +127	1 Byte
INT16	-32768 .. +32767	2 Byte
INT32	-2147483648 .. +2147483647	4 Byte
UINT8	0 .. 255	1 Byte
UINT16	0 .. 65535	2 Byte
UINT32	0 .. 4294967295	4 Byte
Visible String8	ASCII Zeichen	8 Byte
Visible String16	ASCII Zeichen	16 Byte

Zugriff **ro**: "Read Only"Wert kann nur gelesen werden

rw: "Read Write"Wert kann gelesen und geschrieben werden

wo: "Write Only"Wert kann nur geschrieben werden

PDO **R_PDO**: Mapping für R_PDO möglich

T_PDO: Mapping für T_PDO möglich

keine Angabe: PDP-Mapping mit dem Objekt nicht möglich

Wertebereich Gibt den zulässigen Bereich an, in dem der Objektwert definiert und gültig ist.

Defaultwert Durch Laden und Initialisierung mit der gespeicherten Werkseinstellung sind die Defaultwerte einstellbar.

0198441113585, V2.01, 11.2008

speicherbar ja :Werte können im Speicher des Slaves gesichert werden und stehen nach dem Einschalten wieder zur Verfügung.

–: Werte gehen mit Ausschalten des Slaves verloren.

8.1.2 Objekte , Übersicht

Index	Subindex	Bezeichnung	Obj.code	Datentyp	Zugriff
1000_h		device type	VAR	UINT32	ro
1001_h		error register	VAR	UINT8	ro
1008_h		manufacturer device name	VAR	String	ro
$100C_h$		guard time	VAR	UINT16	rw
$100D_h$		life time faktor	VAR	UINT8	rw
1015_h		inhibit time EMCY	VAR	UINT16	rw
1018_h		identity object	RECORD	Identity	ro
1018_h	0	number of elements	VAR	UINT8	ro
1018_h	1	vendor id	VAR	UINT32	ro
1018_h	2	product code	VAR	UINT8	ro
1403_h		receive PDO4 communication parameter	RECORD	PDO_Com	ro
1403_h	0	number of elements	VAR	UINT8	ro
1403_h	1	COB-ID used by R_PDO4	VAR	UINT32	ro
1403_h	2	transmission type R_PDO4	VAR	UINT8	rw
1403_h	3	inhibit time R_PDO4	VAR	UINT16	rw
1403_h	4	compatibility entry R_PDO4	VAR	UINT8	rw
1403_h	5	event timer R_PDO4	VAR	UINT16	rw
1603_h		receive PDO4 mapping	RECORD	PDO_Map	ro
1603_h	0	number of elements	VAR	UINT8	ro
1603_h	1	1st mapped object R_PDO4	VAR	UINT32	ro
1603_h	2	2nd mapped object R_PDO4	VAR	UINT32	ro
1603_h	3	3rd mapped object R_PDO4	VAR	UINT32	ro
1603_h	4	4th mapped object R_PDO4	VAR	UINT32	ro
1803_h		transmit PDO4 communication parameter	RECORD	PDO_Com	ro
1803_h	0	number of elements	VAR	UINT8	ro
1803_h	1	COB-ID used by T_PDO4	VAR	UINT32	ro
1803_h	2	transmission type T_PDO4	VAR	UINT8	rw
1803_h	3	inhibit time T_PDO4	VAR	UINT16	rw
1803_h	4	reserved T_PDO4	VAR	UINT8	rw
1803_h	5	event timer T_PDO4	VAR	UINT16	rw
$1A03_h$		transmit PDO4 mapping	RECORD	PDO_Map	ro
$1A03_h$	0	number of elements	VAR	UINT8	ro
$1A03_h$	1	1st mapped object T_PDO4	VAR	UINT32	ro
$1A03_h$	2	2nd mapped object T_PDO4	VAR	UINT32	ro
$1A03_h$	3	3rd mapped object T_PDO4	VAR	UINT32	ro
$1A03_h$	4	4th mapped object T_PDO4	VAR	UINT32	ro

0198441113585, V2.01, 11.2008

8.2 Objekte des Slaves

1000h **Device type**

Das Objekt gibt das eingesetzte Geräteprofil und den Gerätetyp an.

Objektbeschreibung

Index	1000_h
Objektname	device type
Objektcode	VAR
Datentyp	Unsigned32

Wertebeschreibung

Subindex	00_h, device type
Bedeutung	Gerätetyp und -profil
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	0
speicherbar	–

1001h **Error register**

Das Objekt zeigt den Fehlerzustand des Slaves an. Die detaillierte Fehlerursache kann über das herstellerspezifische Objekt `Status.StopFault 32:7` ermittelt werden.

Fehler werden im Moment ihres Auftretens durch eine EMCY-Nachricht signalisiert.

Objektbeschreibung

Index	1001h
Objektname	error register
Objektcode	VAR
Datentyp	Unsigned8

Wertebeschreibung

Subindex	00_h, error register
Bedeutung	Fehlerregister
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	–
speicherbar	–

0198441113585, V2.01, 11.2008

Bitcodierung, Subindex 00h

Bit	Zugriff	Wert	Bedeutung
0	ro	–	Fehler! (generic error)
1	ro	–	Strom
2	ro	–	Spannung
3	ro	–	Temperatur
4	ro	–	Kommunikationsprofil (communication error)
5	ro	–	Geräteprofil (device profile error)
6	ro	–	reserviert
7	ro	–	herstellerspezifisch (manufacturer specific)

1008h **Manufacturer device name**

Das Objekt gibt den Gerätenamen an (z.B. “IFS ”)

Objektbeschreibung

Index	1008_h
Objektname	manufacturer device name
Objektcode	VAR
Datentyp	String

Wertebeschreibung

Subindex	00_h, manufacturer device name
Bedeutung	Herstellername
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	–
speicherbar	–

100Ch **Guard time**

Das Objekt gibt die Zeitspanne zur Verbindungsüberwachung (node guarding) eines NMT-Slaves an.

Objektbeschreibung

Index	$100C_h$
Objektname	guard time
Objektcode	VAR
Datentyp	Unsigned16

0198441113585, V2.01, 11.2008

Wertebeschreibung

Subindex	00_h, guard time
Bedeutung	Zeitspanne für das Node-Guarding [ms]
Zugriff	read-write
PDO-Mapping	–
Wertebereich	0...65535
Default-Wert	0
speicherbar	–

Die Zeitspanne zur Verbindungsüberwachung eines NMT-Masters ergibt sich aus der Zeitspanne "guard time" multipliziert mit dem Faktor "life time", Objekt `Life time factor` ($100D_h$).

Die Zeitspanne kann im NMT-Zustand "Pre-Operational" geändert werden.

100Dh Life time factor

Das Objekt gibt den Faktor an, der zusammen mit der Zeitspanne "guard time" das Zeitintervall zur Verbindungsüberwachung eines NMT-Masters ergibt. Innerhalb dieser Zeitspanne erwartet der NMT-Slave eine Überwachungsanfrage per Node-guarding vom NMT-Master.

life time = guard time * life time factor

Der Wert "0" deaktiviert die Überwachung des NMT-Masters.

Objektbeschreibung

Index	$100D_h$
Objektname	life time faktor
Objektcode	VAR
Datentyp	Unsigned8

Wertebeschreibung

Subindex	00_h, life time faktor
Bedeutung	Zeitfaktor für das Node-Guarding Protokoll.
Zugriff	read-write
PDO-Mapping	–
Wertebereich	0...255
Default-Wert	0
speicherbar	–

Bleibt die Verbindungsüberwachung von Seiten des NMT-Masters während des Zeitintervalls "life time" aus, meldet die #Variable:device-name# einen Fehler und wechselt in den Fehlerzustand.

Der Zeitfaktor kann im NMT-Zustand "pre-operational" geändert werden.

Die Zeitspanne "guard time" wird mit dem Objekt `Guard time` ($100C_h$) eingestellt.

1015h Inhibit time emergency message

Das Objekt legt die Wartezeit für das wiederholte Senden von EMCY-Nachrichten als vielfaches von 100µs fest.

0198441113585, V2.01, 11.2008

Objektbeschreibung

Index	1015_h
Objektname	inhibit time EMCY
Objektcode	VAR
Datentyp	Unsigned16

Wertebeschreibung

Subindex	00h, inhibit time EMCY
Bedeutung	Wartezeit zum wiederholten Senden einer EMCY
Zugriff	read-write
PDO-Mapping	–
Wertebereich	0...65535
Default-Wert	0
speicherbar	–

1018h Identity Object

Das Objekt gibt Informationen über das Gerät an. Subindex 01_h (vendor Id) enthält die Identifikationskennung des Herstellers, Subindex 02_h (product Id) gibt den herstellerspezifischen Produktcode an.

Wertebeschreibung

Index	1018_h
Objektname	Identity Object
Objektcode	RECORD
Datentyp	Identity

Subindex	00h, number of elements
Bedeutung	Anzahl Subindices
Zugriff	read-only
PDO-Mapping	–
Wertebereich	1...4
Default-Wert	2
speicherbar	–

Subindex	01_h, vendor id
Bedeutung	Verkäufer-ID
Zugriff	read-only
PDO-Mapping	–
Wertebereich	0...4294967295
Default-Wert	0x0100002E
speicherbar	–

0198441113585, V2.01, 11.2008

Subindex	02_h, product code
Bedeutung	Produkt-Kennzeichnung
Zugriff	read-only
PDO-Mapping	–
Wertebereich	0...4294967295
Default-Wert	0x01
speicherbar	–

1403h Receive PDO4 communication parameter

Das Objekt speichert Einstellungen für das vierte Empfangs-PDO R_PDO4.

Objektbeschreibung

Index	1403_h
Objektname	receive PDO4 communication parameter
Objektcode	RECORD
Datentyp	PDO Communication parameter

Wertebeschreibung

Subindex	00_h, number of elements
Bedeutung	Anzahl Subindices
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	5
speicherbar	–

Bedeutung	Identifier des R_PDO4
Subindex	01_h, COB-ID R_PDO4
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	0x40000500+nodeID
speicherbar	–

Subindex	02_h, transmission type R_PDO4
Bedeutung	Übertragungstyp
Zugriff	read-write
PDO-Mapping	–
Wertebereich	–
Default-Wert	254
speicherbar	–

0198441113585, V2.01, 11.2008

Subindex	03_h, inhibit time R_PDO4
Bedeutung	Verzögerungszeit für wiederholte Übertragungen (1=100 µsec)
Zugriff	read-write
PDO-Mapping	–
Wertebereich	0...65535
Default-Wert	0
speicherbar	–

Subindex	04_h, compatibility entry R_PDO4
Bedeutung	nur für Kompatibilitätszwecke
Zugriff	read-write
PDO-Mapping	–
Wertebereich	–
Default-Wert	–
speicherbar	–

Subindex	05_h, event timer R_PDO4
Bedeutung	Zeiteinstellung für Ereignisauslösung
Zugriff	read-write
PDO-Mapping	–
Wertebereich	–
Default-Wert	0
speicherbar	–

Bitbelegung Subindex 01h

Bit	Zugriff	Wert	Bedeutung
31	rw	0_b	0: PDO ist aktiv 1: PDO ist inaktiv
30	ro	0_b	0: RTR (s. u.) ist möglich 1: RTR nicht erlaubt
29	ro	0_b	0: 11-Bit-Identifier (CAN 2.0A) 1: 29.Bit-Identifier (CAN 2.0B)
28-11	ro	0000_h	Nur relevant, wenn Bit 29=1, von Slave nicht genutzt.
10-7	rw	0100_h	Funktions-Code, Bit 10-7 der COB-Id
6-0	ro	–	Knotenadresse, Bit 6-0 der COB-Id

Bit 31 Ein R_PDO kann nur eingesetzt werden, wenn Bit 31="0" ist.

Bit 30 RTR-Bit Unterstützt ein Teilnehmer R_PDOs mit RTR (remote transmission request), kann er entsprechend der Producer-Consumer-Beziehung von einem PDO--Producer mit RTR = "0" ein PDO anfordern.

Der Slave kann keine PDOs anfordern, kann jedoch auf die Anforderung einer PDO reagieren, siehe RTR-Bit für T_PDO1-Einstellungen (1800h).

0198441113585, V2.01, 11.2008

Bitcodierung, Subindex 02h Die Steuerung zur Auswertung von R_PDO-Daten wird über Subindex 02h festgelegt. Die Werte 241..251 sind reserviert.

Übertragungsart	zyklisch	azyklisch	synchron	asynchron	RTR-gesteuert
0	–	X	X	–	–
1-240	X	–	X	–	–
252	–	–	X	–	X
253	–	–	–	X	X
254	–	–	–	X	–
255	–	–	–	X	–

Wird ein R_PDO synchron übertragen (Übertragungsart=0..252), wertet das Gerät empfangene Daten abhängig vom SYNC-Objekt aus.

- Bei azyklischer Übertragung (Übertragungsart=0) ist die Auswertung an das SYNC-Objekt gebunden, nicht jedoch die Übertragung des PDO. Eine empfangene PDO-Nachricht wird mit dem folgenden SYNC ausgewertet.

  Ein Wert zwischen 1 und 240 gibt die Anzahl SYNC-Zyklen an, nach denen ein empfangenes PDO ausgewertet wird.

Die Werte 252 bis 254 sind für die Aktualisierung – nicht jedoch für das Senden – von T_PDOs relevant.

- 252: Aktualisierung von Sendedaten mit Empfang des nächsten SYNC
- 253 Aktualisierung von Sendedaten mit Empfang einer Anforderung von einem PDO-Consumer
- 254: Aktualisierung der Daten ereignisgesteuert, das auslösende Ereignis ist herstellerspezifisch festgelegt

R_PDOs mit dem Wert 255 werden sofort mit Empfang des PDOs aktualisiert. Auslösendes Ereignis sind die Daten, die entsprechend der Definition des Geräteprofils im PDO übertragen werden.

Subindex 03h Das Zeitintervall "Inhibit time" ist nur für T_PDOs relevant.

Ein T_PDO wird frühestens nach Ablauf des Zeitintervalls "Inhibit time" erneut übertragen. Der Wert wird als Vielfaches von 100 [µs] angegeben, jedoch vom Antrieb auf ganze Millisekunden abgerundet.

Subindex 04h Der Wert ist reserviert und wird nicht benutzt. Schreib- oder Lesezugriff löst eine SDO-Fehlernachricht aus.

Subindex 05h Das Zeitintervall "event timer" ist nur für T_PDOs relevant. Ein T_PDO wird nach Ablauf des Zeitintervalls "event timer" übertragen. Gleichzeitig wird das Zeitintervall erneut gestartet. Die Übertragungsart "transmission type" muss über Subindex 02h auf den Wert 254 oder 255 eingestellt sein.

0198441113585, V2.01, 11.2008

Einstellungen

R_PDO4 wird asynchron und ereignisgesteuert verarbeitet.

Die Byte-Belegung des R_PDO4 wird über das PDO-Mapping mit dem Objekt `Receive PDO4 mapping` (1603_h) festgelegt und kann nicht verändert werden. Die Belegung ist im 3.4.2.2 "Empfangs-PDO R_PDO4 (Master -> Slave)" beschrieben.

Die COB-Id des Objekts kann im NMT-Zustand "pre-operational" geändert werden.

1603h

Receive PDO4 mapping

Das Objekt gibt an, welche Objekte im R_PDO4 abgebildet sind und mit dem PDO übertragen werden. Mit Lesen des Objekts, Subindex 00_h wird die Anzahl der abgebildeten Objekte angegeben.

Objektbeschreibung

Index	1603h
Objektname	receive PDO4 mapping
Objektcode	RECORD
Datentyp	PDO Mapping

Wertebeschreibung

Subindex	00h, number of elements
Bedeutung	Anzahl Subindices
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	4
speicherbar	–

Subindex	01_h, 1st mapped object R_PDO4
Bedeutung	Erstes Objekt für das Mapping in R_PDO4
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	0x301E0108
speicherbar	–

Subindex	02_h, 2nd mapped object R_PDO4
Bedeutung	Zweites Objekt für das Mapping in R_PDO4
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	0x301E0208
speicherbar	–

0198441113585, V2.01, 11.2008

Subindex	03_h, 3rd mapped object R_PDO4
Bedeutung	Drittes Objekt für das Mapping in R_PDO4
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	0x301E0510
speicherbar	–

Subindex	04_h, 4th mapped object R_PDO4
Bedeutung	Viertes Objekt für das Mapping in R_PDO4
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	0x301E0620
speicherbar	–

Bitcodierung ab Subindex 01h

Jeder Subindex-Eintrag ab Subindex 01h gibt das Objekt und die Bytelänge des Objekts an. Identifiziert wird das Objekt über Index und Subindex, die sich auf das Objektverzeichnis des Geräts beziehen.

Bit	Bedeutung
31..16	Index
15..8	Subindex
7..0	Objektlänge in Byte

Einstellungen

Die Belegung des R_PDO4 ist fest eingestellt und kann nicht verändert werden.

Die Belegung ist im 3.4.2.2 "Empfangs-PDO R_PDO4 (Master -> Slave)" beschrieben.

1803h

Transmit PDO4 communication parameter

Das Objekt speichert Einstellungen für das vierte Sende-PDO T_PDO4.

Objektbeschreibung

Index	1803_h
Objektname	Transmit PDO4 communication parameter
Objektcode	RECORD
Datentyp	PDO Communication Parameter

Wertebeschreibung

Subindex	00_h, number of elements
Bedeutung	Anzahl Subindices
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	5
speicherbar	–

0198441113585, V2.01, 11.2008

Subindex	01_h, COB-ID used by T_PDO4
Bedeutung	Identifier des T_PDO4
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	0x00000480+nodeID
speicherbar	–

Subindex	02_h, transmission type T_PDO4
Bedeutung	Übertragungstyp
Zugriff	read-write
PDO-Mapping	–
Wertebereich	–
Default-Wert	254
speicherbar	–

Subindex	03_h, inhibit time T_PDO4
Bedeutung	Verzögerungszeit für wiederholte Übertragungen (in [100µsec]). Der Wert wird vom Antrieb auf ganze Millisekunden abgerundet.
Zugriff	read-write
PDO-Mapping	–
Wertebereich	0...65535
Default-Wert	0
speicherbar	–

Subindex	04_h, reserved T_PDO4
Bedeutung	reserviert (nur für Kompatibilitätszwecke)
Zugriff	read-write
PDO-Mapping	–
Wertebereich	–
Default-Wert	–
speicherbar	–

Subindex	05h, event timer T_PDO4
Bedeutung	Zeiteinstellung für Ereignisauslösung
Zugriff	read-write
PDO-Mapping	–
Wertebereich	–
Default-Wert	0
speicherbar	–

Die Bedeutung der Bitzustände und Subindexwerte wird mit dem Objekt `receive PDO4 communication parameter` (1403_h) beschrieben.

0198441113585, V2.01, 11.2008

Einstellungen

T_PDO4 wird asynchron und ereignisgesteuert übermittelt.

Die Byte-Belegung des T_PDO4 wird über das PDO-Mapping mit dem Objekt `transmit PDO4 mapping` ($1A03_h$) festgelegt und kann nicht verändert werden. Die Belegung ist im 3.4.2.3 "Sende-PDO T_PDO4 (Slave zu Master)" beschrieben.

Die COB-Id des Objekts kann im NMT-Zustand "pre-operational" geändert werden.

1A03h

Transmit PDO4 mapping

Das Objekt gibt an, welche Objekte im T_PDO4 abgebildet sind und mit dem PDO übertragen werden. Mit Lesen des Objekts, Subindex 00_h wird die Anzahl der abgebildeten Objekte angegeben.

Objektbeschreibung

Index	$1A03_h$
Objektname	transmit PDO4 mapping
Objektcode	RECORD
Datentyp	PDO Mapping

Wertebeschreibung

Subindex	00_h, number of elements
Bedeutung	Anzahl Subindices
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	4
speicherbar	–

Subindex	01_h, 1st mapped object T_PDO4
Bedeutung	Erstes Objekt für das Mapping in T_PDO4
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	0x301E0410
speicherbar	–

Subindex	02_h, 2nd mapped object T_PDO4
Bedeutung	Zweites Objekt für das Mapping in T_PDO4
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	0x301E0308
speicherbar	–

Subindex	03_h, 3rd mapped object T_PDO4
Bedeutung	Drittes Objekt für das Mapping in T_PDO4
Zugriff	read-only

0198441113585, V2.01, 11.2008

PDO-Mapping	–
Wertebereich	–
Default-Wert	0x301E0708
speicherbar	–

Subindex	04_h, 4th mapped object T_PDO4
Bedeutung	Viertes Objekt für das Mapping in T_PDO4
Zugriff	read-only
PDO-Mapping	–
Wertebereich	–
Default-Wert	0x301E0820
speicherbar	–

Die Bedeutung der Bitzustände wird mit dem Objekt `receive PDO4 mapping` (1603_h) beschrieben..

Einstellungen Die PDO-Belegung für T_PDO4 kann nicht verändert werden. Die Belegung ist im 3.4.2.3 "Sende-PDO T_PDO4 (Slave zu Master)" beschrieben.

0198441113585, V2.01, 11.2008

9 Glossar

9.1 Einheiten und Umrechnungstabellen

Der Wert in der gegebenen Einheit (linke Spalte) wird mit der Formel (im Feld) für die gesuchte Einheit (obere Zeile) berechnet.

Beispiel: Umrechnung von 5 Meter [m] nach Yard [yd]
5 m / 0,9144 = 5,468 yd

9.1.1 Länge

	in	ft	yd	m	cm	mm
in	-	/ 12	/ 36	* 0,0254	* 2,54	* 25,4
ft	* 12	-	/ 3	* 0,30479	* 30,479	* 304,79
yd	* 36	* 3	-	* 0,9144	* 91,44	* 914,4
m	/ 0,0254	/ 0,30479	/ 0,9144	-	* 100	* 1000
cm	/ 2,54	/ 30,479	/ 91,44	/ 100	-	* 10
mm	/ 25,4	/ 304,79	/ 914,4	/ 1000	/ 10	-

9.1.2 Masse

	lb	oz	slug	kg	g
lb	-	* 16	* 0,03108095	* 0,4535924	* 453,5924
oz	/ 16	-	* $1{,}942559 \cdot 10^{-3}$	* 0,02834952	* 28,34952
slug	/ 0,03108095	/ $1{,}942559 \cdot 10^{-3}$	-	* 14,5939	* 14593,9
kg	/ 0,453592370	/ 0,02834952	/ 14,5939	-	* 1000
g	/ 453,592370	/ 28,34952	/ 14593,9	/ 1000	-

9.1.3 Kraft

	lb	oz	p	dyne	N
lb	-	* 16	* 453,55358	* 444822,2	* 4,448222
oz	/ 16	-	* 28,349524	* 27801	* 0,27801
p	/ 453,55358	/ 28,349524	-	* 980,7	* $9{,}807 \cdot 10^{-3}$
dyne	/ 444822,2	/ 27801	/ 980,7	-	/ $100 \cdot 10^{3}$
N	/ 4,448222	/ 0,27801	/ $9{,}807 \cdot 10^{-3}$	* $100 \cdot 10^{3}$	-

9.1.4 Leistung

	HP	W
HP	-	* 746
W	/ 746	-

0198441113585, V2.01, 11.2008

9.1.5 Rotation

	min^{-1} (RPM)	**rad/s**	**deg./s**
min^{-1} (RPM)	-	* π / 30	* 6
rad/s	* 30 / π	-	* 57,295
deg./s	/ 6	/ 57,295	-

9.1.6 Drehmoment

	lb·in	**lb·ft**	**oz·in**	**Nm**	**kp·m**	**kp·cm**	**dyne·cm**
lb·in	-	/ 12	* 16	* 0,112985	* 0,011521	* 1,1521	* $1,129*10^6$
lb·ft	* 12	-	* 192	* 1,355822	* 0,138255	* 13,8255	* $13,558*10^6$
oz·in	/ 16	/ 192	-	* $7,0616*10^{-3}$	* $720,07*10^{-6}$	* $72,007*10^{-3}$	* 70615,5
Nm	/ 0,112985	/ 1,355822	/ $7,0616*10^{-3}$	-	* 0,101972	* 10,1972	* $10*10^6$
kp·m	/ 0,011521	/ 0,138255	/ $720,07*10^{-6}$	/ 0,101972	-	* 100	* $98,066*10^6$
kp·cm	/ 1,1521	/ 13,8255	/ $72,007*10^{-3}$	/ 10,1972	/ 100	-	* $0,9806*10^6$
dyne·cm	/ $1,129*10^6$	/ $13,558*10^6$	/ 70615,5	/ $10*10^6$	/ $98,066*10^6$	/ $0,9806*10^6$	-

9.1.7 Trägheitsmoment

	lb·in^2	**lb·ft^2**	**kg·m^2**	**kg·cm^2**	**kp·cm·s^2**	**oz·in^2**
lb·in^2	-	/ 144	/ 3417,16	/ 0,341716	/ 335,109	* 16
lb·ft^2	* 144	-	* 0,04214	* 421,4	* 0,429711	* 2304
kg·m^2	* 3417,16	/ 0,04214	-	* $10*10^3$	* 10,1972	* 54674
kg·cm^2	* 0,341716	/ 421,4	/ $10*10^3$	-	/ 980,665	* 5,46
kp·cm·s^2	* 335,109	/ 0,429711	/ 10,1972	* 980,665	-	* 5361,74
oz·in^2	/ 16	/ 2304	/ 54674	/ 5,46	/ 5361,74	-

9.1.8 Temperatur

	°F	**°C**	**K**
°F	-	(°F - 32) * 5/9	(°F - 32) * 5/9 + 273,15
°C	°C * 9/5 + 32	-	°C + 273,15
K	(K - 273,15) * 9/5 + 32	K - 273,15	-

9.1.9 Leiterquerschnitt

AWG	1	2	3	4	5	6	7	8	9	10	11	12	13
mm^2	42,4	33,6	26,7	21,2	16,8	13,3	10,5	8,4	6,6	5,3	4,2	3,3	2,6

AWG	14	15	16	17	18	19	20	21	22	23	24	25	26
mm^2	2,1	1,7	1,3	1,0	0,82	0,65	0,52	0,41	0,33	0,26	0,20	0,16	0,13

0198441113585, V2.01, 11.2008

9.2 Begriffe und Abkürzungen

AC	Alternating current (engl.), Wechselstrom
CAN	(**C**ontroller **A**rea **N**etwork), standardisierter offener Feldbus nach ISO 11898, über den Antriebe und andere Geräte unterschiedlicher Hersteller miteinander kommunizieren.
CANopen	geräte- und herstellerunabhängige Beschreibungssprache zur Kommunikation im CAN-Bus
CiA	**C**AN **i**n **A**utomation, CAN Interessengemeinschaft, legt Standards für CAN und CANopen fest.
COB-ID	(**C**ommunication **OB**ject**-Id**entifier) identifiziert eindeutig jedes Kommunikationsobjekt in einem CAN Netzwerk
DC	Direct current (engl.), Gleichstrom
Defaultwert	Werkseinstellung.
DriveCom	Spezifikation der DSP402 Zustandsmaschine wurde entsprechend der DriveCom Spezifikation erstellt.
DS301	standardisiert das CANopen Kommunikationsprofil
DSP402	standardisiert das CANopen Geräteprofil für Antriebsverstärker
E	Encoder (engl.)
E/A	Ein-/Ausgänge
EDS	(**E**lectronic **D**ata **S**heet) elektronisches Datenblatt, das spezifische Merkmale eines Produkts enthält.
Eingabegerät	Ein an die RS232-Schnittstelle anschließbares Gerät zur Inbetriebnahme; entweder das Handbediengerät HMI oder ein PC mit der Inbetriebnahmesoftware.
Elektronisches Getriebe	Im Antriebssystem erfolgende Umrechnung einer Eingangsdrehzahl mit den Werten eines einstellbaren Getriebefaktors zu einer neuen Ausgangsdrehzahl für die Motorbewegung.
EMCY-Objekt	Emergency Objekt
EMV	Elektromagnetische Verträglichkeit.
Encoder	Sensor zur Erfassung der Winkelposition eines rotierenden Elements. Im Motor eingebaut gibt der Encoder die Winkellage des Rotors an.
Endschalter	Schalter, die das Verlassen des zulässigen Verfahrbereichs melden.
Endstufe	Hierüber wird der Motor angesteuert. Die Endstufe erzeugt entsprechend den Positioniersignalen der Steuerung Ströme zur Ansteuerung des Motors.
Error	Diskrepanz zwischen einem erkannten (berechneten, gemessenen oder per Signal übermittelten) Wert oder Zustand und dem vorgesehenen oder theoretisch korrekten Wert beziehungsweise Zustand.
Fataler Fehler	Bei einem fatalen Fehler ist das Produkt nicht mehr in der Lage, den Motor anzusteuern, so dass ein sofortiges Deaktivieren der Endstufe erforderlich wird.

0198441113585, V2.01, 11.2008

Fault	Fault ist ein Betriebszustand des Antriebs. In den Betriebszustand Fault wird gewechselt, wenn eine Diskrepanz zwischen einem erkannten (berechneten, gemessenen oder per Signal übermittelten) Wert oder Zustand einerseits und einem vorgesehenen oder theoretisch korrekten Wert oder Zustand andererseits auftritt.
Fault reset	Eine Funktion, mit der ein Antrieb nach einem erkannten Fehler wieder in den regulären Betriebszustand versetzt wird, nachdem die Fehlerursache beseitigt worden ist und der Fehler nicht mehr ansteht (Zustandswechsel von "Fault" zu "Operation Enable").
Fehlerklasse	Klassifizierung von Fehlern in Gruppen. Die Einteilung in unterschiedliche Fehlerklassen ermöglicht gezielte Reaktionen auf die Fehler einer Klasse, zum Beispiel nach Schwere eines Fehlers.
Heartbeat	(engl.: Herzschlag) dient zur unbestätigten Verbindungsmeldung von Netzwerkteilnehmern.
HMI	Human Machine Interface (engl.): Mensch-Maschine-Schnittstelle, Handbediengerät.
Leistungsteil	siehe Endstufe
Life-Guarding	(engl.: Überwachung auf Lebenszeichen) zur Verbindungsüberwachung eines NMT-Masters
Mapping	Zuordnung von Objektverzeichniseinträgen an PDOs
node-ID	Knotenadresse, die ein Teilnehmer am Netzwerk belegt.
NMT	Netzwerk-Management (NMT), Teil des CANopen-Kommunikationsprofils, Aufgaben: Netzwerk und Teilnehmer initialisieren, Teilnehmer starten, stoppen, überwachen
Node Guarding	(engl.: Knotenüberwachung), Verbindungsüberwachung mit dem Slave an einer Schnittstelle auf zyklischen Datenverkehr.
Objektverzeichnis	Liste aller im Gerät verfügbaren Parameter, Werte und Funktionen. Jeder Eintrag wird über Index (16 bit) und Subindex (8 bit) eindeutig referenziert.
Parameter	Vom Anwender einstellbare Gerätedaten und -werte.
PDO	Prozessdatenobjekt
Persistent	Kennzeichnung, ob der Wert des Parameters nach Abschalten des Gerätes im Speicher erhalten bleibt.
Quick Stop	Schnell-Stopp, Funktion wird bei Störung oder über einen Befehl zum schnellen Abbremsen des Motors eingesetzt.
R_PDO	(engl. receive: empfangen) Empfangs-PDO
SDO	Servicedatenobjekt
SYNC-Objekt	Synchronisations-Objekt
T_PDO	(engl. transmit: senden) Sende-PDO
Warnung	Bei einer Warnung außerhalb des Kontextes von Sicherheitshinweisen handelt es sich um einen Hinweis auf ein potentielles Problem, das durch eine Überwachungsfunktion erkannt wurde. Eine Warnung ist kein Fehler und bewirkt keinen Wechsel des Betriebszustands.

0198441113585, V2.01, 11.2008

10 Stichwortverzeichnis

A

Abkürzungen 97
ABORT 78, 80
Adresse 56
    prüfen 77
Antwort
    auf SDO Fehler 32
Asynchrone Fehler 78
Aufgaben
    der COB-Id 23
Azyklische Datenübertragung 47

B

Baudrate 56
    prüfen 77
Begriffe 97
Beispiel
    SDO Nachricht 29
    Wahl einer COB-Id 24
Betrieb 61
Betriebsart
    Geschwindigkeitsprofil 72
    Referenzierung 74
Bevor Sie beginnen
    Sicherheitsinformationen 13
Bitfelder
    Data 22
    Identifier 22
Boot-Up
    Nachricht 49
Busarbitrierung 23

C

CAN
    Nachricht 22
CAN 3.0A 23
CANopen
    Fehlermeldungen 79
    Kommunikationsprofil, NMT 48
    Nachricht 22
    Standards 11
ccd
    siehe Kommando-Code
Client-Server 26
    SDO Datenaustausch 28
COB-Id 23
    Aufgaben 23
    bei Node-guarding 51
    Busarbitrierung 23
    der Kommunikationsobjekte 23
    Identifikation von Kommunikationsobjekten 23

0198441113585, V2.01, 11.2008

SDO 29
SYNC-Objekt 47
Codierung
Kommando-Code 31
Command specifier 50
command-code
siehe Kommando-Code

D

Data Bitfeld 22
Daten
lesen 31
persistent gespeicherte 50
schreiben 30
SDO 29
Datenlänge
flexible 33
Datenrahmen 24
des NMT-Gerätedienstes 50
SDO 29
Datenübertragung
azyklische 47
synchrone 47
zyklische 47
Diagnose 77
Dienste
NMT 21, 48
zur Gerätekontrolle 48
zur Verbindungsüberwachung 48
Dokumentation und Literaturhinweise 11
DS 301
Kommunikationsprofil 18
DSP 402
Geräteprofil 18

E

Echtzeit-Datenaustausch 33
Einführung 9
Einheiten und Umrechnungstabellen 95
EMCY
Objekt 21
Emergency object
siehe EMCY-Objekt
Empfänger
einer NMT-Nachricht 50

F

Fahrtunterbrechung
Ursache 79
Fehler
Antwort bei SDO 32
Behebung 77
Meldungen zu CANopen 79
Fehlercode

0198441113585, V2.01, 11.2008

Tabelle 79
Fehlerdiagnose
Anschlüsse zum Feldbusbetrieb 77
Funktionstest im Feldbus 77
Fehlerregister 79
function-code
siehe Funktionscode
Funktionscode 24
Funktionstest
im Feldbus 77

G

Geräteinbetriebnahme 55
Geräteprofil
DSP 402 18
Geschwindigkeitsprofil 72
Glossar 95
Grundlagen 15

H

herstellerspezifisch
Profile 19

I

Identifier Bitfeld 22
Identifikation
von Kommunikationsobjekten 23
Inbetriebnahme 55
Index
SDO 29
Installation 53

K

Knotenadresse 23, 24, 50
Kodierung
Kommando-Code 30
Kommando-Code
Lesewert 31
Schreibwert 30
SDO 29
Kommunikationsbeziehung
Client-Server 25
Master-Slave 25
Producer-Consumer 25
Kommunikationsobjekte
COB-Ids 23
Identifikation 23
Steuerung von 23
Übersicht 21
Kommunikationsprofil
DS 301 18

L

Life guarding 50

0198441113585, V2.01, 11.2008

M

Master-Slave 25
Meldungen
asynchrone Fehler 78
Error code (603Fh) 79
Error register(1001h) 79
synchrone Fehler 78
über den Gerätezustand 78
Meldungsobjekte 78
EMCY(80h+ node-Id) 78
Error code(603Fh) 78
Error register (1001h) 78
Statusword (6041h) 78
Mode Toggle 42
Multi-Master-Fähigkeit 9

N

Nachricht 22
CANopen 22
NMT 50, 51
SDO 29
Nachrichtenorientierte Kommunikation 9
Netzwerk-Management
siehe NMT
NMT
Aufbau einer Nachricht 51
Dienste 21, 48
Initialisierung 49
zur Gerätekontrolle 49
zur Verbindungsüberwachung 50
Empfänger einer Nachricht 50
Nachricht 50
Netzwerkdienste 48
Zustand des Slaves 51
Zustandsmaschine 49
Node guarding 50
Verbindungsfehler 52
Node-guarding
COB-Id 51
node-Id 23

O

Objektgruppen
Übersicht 16

P

PDO 21, 33
Producer-Consumer 34
Priorisierung von Nachrichten 9
Process Data Object
siehe PDO
Producer-Consumer 27
PDO 34
SYNC 46

0198441113585, V2.01, 11.2008

Profile
herstellerspezifische 19
standardisierte 18
Prozessdatenobjekte
siehe PDO
Prüfen
Adresse 77
Baudrate 77

R

Referenzierung 74
Restfehlerwahrscheinlichkeit 9

S

Schichtenmodell
Application Layer 15
Data Link Layer 15
Physical Layer 15
SDO 21, 28
Antwort 31
COB-Id 29
Daten 29
Datenrahmen 29
Fehler-Antwort 32
fehlerhafte Übertragung 80
Fehlernachricht 78, 80
Index, Subindex 29
Kommando-Code 29
Nachricht 29
Nachrichtentypen 28
Service Data Object
siehe SDO
Servicedatenobjekte 21
siehe SDO
Spezifikation
CAN 3.0A 23
Subindex
SDO 29
synchrone
Datenübertragung 47
Fehler 78
Synchronisation 46
Zeitwerte 46
Synchronisation object
siehe SYNC-Objekt
Synchronisationsobjekt
siehe SYNC-Objekt
SYNC-Objekt 21, 46
bei PDO 34
COB-Id 47

0198441113585, V2.01, 11.2008

U

Übersicht
Kommunikationsobjekte 21
Objektgruppen 16

V

Verbindungsfehler
node guarding 52
Verbindungsüberwachung
NMT-Dienste 50

Z

Zeitwerte
zur Synchronisation 46
Zustandsmaschine
NMT 49
Zyklische Datenübertragung 47

0198441113585, V2.01, 11.2008