

Kommunikations- handbuch

MC 5010

MC 5005

MC 5004

MC 5004 P STO

MCS

MC 3603

EtherCAT[®] 

Impressum

Version:
7. Auflage, 14.09.2023

Copyright
by Dr. Fritz Faulhaber GmbH & Co. KG
Faulhaberstraße 1 · 71101 Schönaich

Alle Rechte, auch die der Übersetzung, vorbehalten.
Ohne vorherige ausdrückliche schriftliche Genehmigung
der Dr. Fritz Faulhaber GmbH & Co. KG darf kein Teil
dieser Beschreibung vervielfältigt, reproduziert, in einem
Informationssystem gespeichert oder verarbeitet oder in
anderer Form weiter übertragen werden.

Dieses Dokument wurde mit Sorgfalt erstellt.
Die Dr. Fritz Faulhaber GmbH & Co. KG übernimmt jedoch
für eventuelle Irrtümer in diesem Dokument und
deren Folgen keine Haftung. Ebenso wird keine Haftung
für direkte Schäden oder Folgeschäden übernommen,
die sich aus einem unsachgemäßen Gebrauch der Geräte
ergeben.

Bei der Anwendung der Geräte sind die einschlägigen
Vorschriften bezüglich Sicherheitstechnik und Funkentstörung
sowie die Vorgaben dieses Dokuments zu beachten.

Änderungen vorbehalten.

Die jeweils aktuelle Version dieses Dokuments
finden Sie auf der Internetseite von FAULHABER:
www.faulhaber.com

Inhalt

1	Zu diesem Dokument	5
1.1	Gültigkeit dieses Dokuments	5
1.2	Mitgeltende Dokumente	5
1.3	Umgang mit diesem Dokument	5
1.4	Abkürzungsverzeichnis	6
1.5	Symbole und Kennzeichnungen	7
2	Überblick	8
2.1	Grundaufbau eines EtherCAT-Geräts	8
2.2	FAULHABER Motion Manager	9
2.3	Voraussetzungen für die Kommunikation (Physical Layer)	10
2.4	ESI-Datei	11
2.5	Identifikation eines Slaves	11
3	EtherCAT-Kommunikation	12
3.1	Einführung	12
3.2	Data Link Layer	12
3.2.1	EtherCAT-Frames und Datagramme	13
3.2.2	SyncManager-Management	14
3.2.3	Adressierung	15
3.2.4	Schnittstellen zum Application Layer	15
3.3	Application Layer	16
3.4	PDO (Prozessdatenobjekt)	17
3.4.1	PDO-Konfiguration	17
3.4.2	PDO-Mapping in der Standardkonfiguration	17
3.5	SDO (Servicedatenobjekt)	19
3.5.1	SDO-Fehlerbeschreibung	19
3.6	Emergency-Objekt (Fehlermeldung)	20
3.7	Synchronisation	22
3.7.1	Synchronisation über Distributed Clocks (DC-Sync)	23
3.7.2	Synchronisation über ein SyncManager-Ereignis (SM-Sync)	24
3.8	Layer management	25
3.8.1	Steuerung der EtherCAT-Statemachine	25
3.8.2	Slave Information Interface (SII)	26
3.9	Einträge im Objektverzeichnis	26
3.10	Fehlerbehandlung	27
3.10.1	Gerätefehler	27
3.10.2	Kommunikationsfehler	28
3.10.2.1	Fehlerprüfung über EtherCAT-Frame-Einträge	28
3.10.2.2	Fehlerreaktion	29
3.10.2.3	Analyse des Netzwerkverkehrs	30
3.10.2.4	EtherCAT AL-Status-Codes und Fehlerbehebung	30
3.11	Parameter speichern und wiederherstellen	32
3.11.1	Parameter speichern	33
3.11.2	Einstellungen wiederherstellen	33
3.11.3	Parametersatz wechseln	34

Inhalt

4	Parameterbeschreibung	36
4.1	Kommunikationsobjekte nach CiA 301	36
4.2	Herstellerspezifische Objekte	44

Zu diesem Dokument

1 Zu diesem Dokument

1.1 Gültigkeit dieses Dokuments

Dieses Dokument beschreibt:

- Kommunikation mit dem Antrieb über EtherCAT
- Basisdienste der Kommunikationsstruktur
- Methoden für den Parameterzugriff
- Antrieb aus Kommunikationssicht

Dieses Dokument richtet sich an Softwareentwickler mit EtherCAT-Erfahrung und an EtherCAT-Projektingenieure.

Alle Angaben in diesem Dokument beziehen sich auf Standardausführungen der Antriebe. Änderungen aufgrund kundenspezifischer Ausführungen dem entsprechenden Datenblatt entnehmen.

Alle Angaben in diesem Dokument beziehen sich auf die Firmware-Revision M.

1.2 Mitgeltende Dokumente

Für bestimmte Handlungsschritte bei der Inbetriebnahme und Bedienung der FAULHABER Produkte sind zusätzliche Informationen aus folgenden Handbüchern hilfreich:

Handbuch	Beschreibung
Motion Manager 6	Bedienungsanleitung zur FAULHABER Motion Manager PC Software
Schnellstartanleitung	Beschreibung der ersten Schritte zur Inbetriebnahme und Bedienung des FAULHABER Motion Controllers
Antriebsfunktionen	Beschreibung der Betriebsarten und Funktionen des Antriebs
Gerätehandbuch	Anleitung zur Installation und zum Gebrauch des FAULHABER Motion Controllers
CiA 301	CANopen application layer and communication profile
CiA 402	CANopen device profile for drives and motion control

Diese Handbücher können im PDF-Format von der Internetseite www.faulhaber.com/manuals heruntergeladen werden.

1.3 Umgang mit diesem Dokument

- ▶ Dokument vor der Konfiguration aufmerksam lesen.
- ▶ Dokument während der Lebensdauer des Produkts aufbewahren.
- ▶ Dokument dem Bedienpersonal jederzeit zugänglich halten.
- ▶ Dokument an jeden nachfolgenden Besitzer oder Benutzer des Produkts weitergeben.

Zu diesem Dokument

1.4 Abkürzungsverzeichnis

Abkürzung	Bedeutung
AL	Application Layer
Attr.	Attribut
CAN	Controller Area Network
CSP	Cyclic Synchronous Position
CSV	Comma-Separated Values
DC	Distributed Clocks
DL	Data Link Layer
EEPROM	Electrically Erasable Programmable Read-Only Memory
EMCY	Emergency
ESC	EtherCAT Slave Controller
ESI	EtherCAT Slave Information
ETG	EtherCAT Technology Group
EtherCAT	Ethernet for Control Automation Technology
FCS	Frame Check Sequence
FMMU	Fieldbus Memory Management Unit
HB	High Byte
HHB	Higher High Byte
HLB	Higher Low Byte
LB	Low Byte
LHB	Lower High Byte
LLB	Lower Low Byte
LSB	Least Significant Byte
LSS	Layer Setting Service
MSB	Most Significant Byte
OD	Objektverzeichnis
PDO	Prozessdatenobjekt
PP	Profile Position
PV	Profile Velocity
ro	read only
RTR	Remote Request
rw	read-write
RxPDO	Receive-Prozessdatenobjekt (vom Antrieb empfangenes PDO)
SDO	Servicedatenobjekt
SII	Slave Information Interface
PLC	Speicherprogrammierbare Steuerung
Sxx	Datentyp Signed (negative und positive Zahlen) mit Bitgröße xx
TxPDO	Transmit-Prozessdatenobjekt (vom Antrieb gesendetes PDO)
Uxx	Datentyp Unsigned (positive Zahlen) mit Bitgröße xx

1.5 Symbole und Kennzeichnungen



HINWEIS!

Gefahr von Sachschäden.

- ▶ Maßnahme zur Vermeidung



Hinweise zum Verständnis oder zum Optimieren der Arbeitsabläufe

- ✓ Voraussetzung zu einer Handlungsaufforderung
- 1. Erster Schritt einer Handlungsaufforderung
 - ↪ Resultat eines Schritts
- 2. Zweiter Schritt einer Handlungsaufforderung
 - ↪ Resultat einer Handlung
- ▶ Einschrittige Handlungsaufforderung

Überblick

2 Überblick

EtherCAT ist eine eingetragene Marke und patentierte Technologie lizenziert durch die Beckhoff Automation GmbH, Deutschland.

2.1 Grundaufbau eines EtherCAT-Geräts

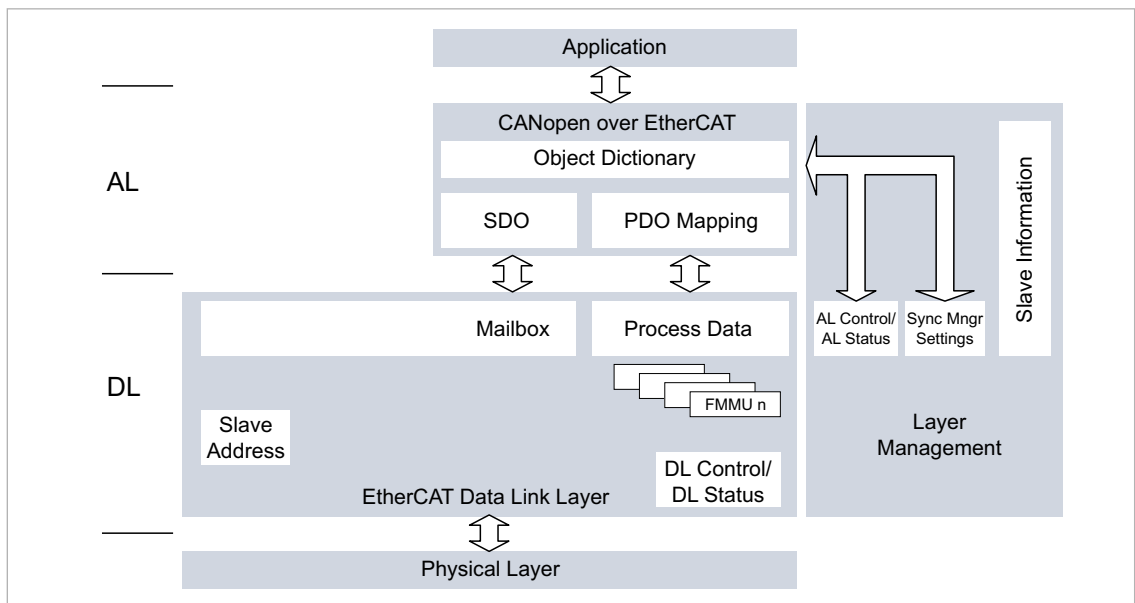


Abb. 1: Grundaufbau eines EtherCAT-Geräts

Physical Layer

Der EtherCAT Physical Layer ist gemäß den Spezifikationen der für Ethernet erstellten Norm IEEE 802.3 mit dem Standard 100Base-TX aufgebaut. Er stellt das Bindeglied zwischen dem EtherCAT-Master und den EtherCAT-Slaves dar. Der Physical Layer tauscht Datenpakete mit dem Data Link Layer aus und kodiert bzw. dekodiert diese Datenpakete, indem er Framing-Informationen hinzufügt bzw. entfernt.

Data Link Layer

Der Data Link Layer entnimmt aus dem durchlaufenden EtherCAT-Frame Daten und/oder legt sie dort ab. Außerdem überprüft er den EtherCAT-Frame auf Vollständigkeit. Dabei geht der Data Link Layer nach Regeln vor, die in Data Link Layer Parametern gespeichert sind. Die Daten werden entweder als Mailbox-Daten oder als Prozessdaten in den zugehörigen Speicherabschnitten dem EtherCAT-Slave zur Verfügung gestellt (siehe Kap. 3.2, S. 12).

Application Layer

Der Application Layer enthält alle Dienste und Objekte, die für die Kommunikation zwischen dem Data Layer und dem Antrieb notwendig sind. Die Dienste sind in Anlehnung an CANopen gestaltet (siehe Kap. 3.2, S. 12).

Application

Der Anwendungsteil enthält Antriebsfunktionen entsprechend CiA 402. Die Antriebsfunktionen lesen Parameter aus dem Objektverzeichnis, erhalten vom Objektverzeichnis Sollwerte und geben Istwerte zurück. Die Parameter aus dem Objektverzeichnis bestimmen das Antriebsverhalten.

i Auf den Anwendungsteil wird in diesem Dokument nicht näher eingegangen. Die Kommunikation mit dem Antrieb und die zugehörigen Betriebsarten sind im separaten Handbuch „Antriebsfunktionen“ beschrieben.

2.2 FAULHABER Motion Manager

Es wird empfohlen, die erste Inbetriebnahme eines FAULHABER-Antriebs über die USB-Schnittstelle oder die serielle COM-Schnittstelle (je nach Verfügbarkeit) des Motion Controllers mit der Software „FAULHABER Motion Manager“ durchzuführen.

i Beim gleichzeitigen Betrieb mehrerer Schnittstellen können unzulässige Übergangszustände auftreten.

Vor dem Konfigurieren des FAULHABER-Antriebs über die USB- oder RS232-Schnittstelle den Motion Controller vom EtherCAT-Netzwerk trennen.

Der FAULHABER Motion Manager ermöglicht einen einfachen Zugriff auf die Einstellungen und Parameter der angeschlossenen Motorsteuerungen. Über die grafische Benutzeroberfläche können Konfigurationen ausgelesen, verändert und wieder eingespielt werden. Einzelne Befehle oder komplette Parametersätze und Programmsequenzen können eingegeben und zur Steuerung übertragen werden.

Assistenzfunktionen unterstützen den Bediener bei der Inbetriebnahme von Antriebssteuerungen. Die Assistenzfunktionen sind auf der Benutzeroberfläche so angeordnet wie sie im Normalfall verwendet werden:

- Verbindungsassistent: Unterstützt den Bediener beim Einrichten der Verbindung zur angeschlossenen Steuerung
- Motorassistent: Unterstützt den Bediener beim Anpassen einer externen Steuerung an den angeschlossenen Motor durch Auswahl des jeweiligen FAULHABER Motors
- Reglereinstellungsassistent: Unterstützt den Bediener bei der Optimierung der Reglerparameter.

Die Software kann kostenlos von der FAULHABER Internet-Seite heruntergeladen werden: <https://www.faulhaber.com/motionmanager>

i Es wird empfohlen, immer die neueste Version des FAULHABER Motion Managers zu verwenden.

Der FAULHABER Motion Manager ist im separaten Handbuch „Motion Manager 6“ beschrieben. Der Inhalt des Handbuchs steht zusätzlich als kontext-sensitive Online-Hilfe des FAULHABER Motion Managers zur Verfügung.

2.3 Voraussetzungen für die Kommunikation (Physical Layer)

i Für die Netzwerkverbindungen können Ethernet-Patchkabel oder Crossoverkabel der Kategorie 5e (Cat5e gemäß EN 50288) oder höher bis maximal 100 m Länge verwendet werden.

In einem physikalischen Netzwerk niemals EtherCAT und Standard-Ethernet zusammen verwenden. Die Kommunikation kann beeinträchtigt werden.

i Beim gleichzeitigen Betrieb mehrerer Schnittstellen können unzulässige Übergangszustände auftreten.

Vor dem Anschließen im EtherCAT-Netzwerk sicherstellen, dass der Motion Controller über keine weiteren Schnittstellen (z. B. USB, RS232) angeschlossen ist.

1. Controller an Spannungsversorgung (mindestens Elektronikversorgung) anschließen.
2. EtherCAT-Anschluss IN mit dem Anschluss auf Master-Seite verbinden (siehe Abb. 2).
3. Bei Anschluss mehrerer Controller den EtherCAT-Anschluss OUT mit dem EtherCAT-Anschluss IN des nächsten Controllers verbinden.
 - ↗ Der EtherCAT-Anschluss OUT des letzten Controllers (Slave) in der Kette bleibt frei. Das vom Master kommende Telegramm wird nach dem Durchlaufen aller Slaves durch dieselbe Leitung an den Master zurückgesendet.
4. Spannung einschalten.
5. Über die Konfigurationsanwendung Verbindung herstellen (siehe Kap. 2.2, S. 9).
6. EtherCAT-Slave Informationen definieren (siehe Kap. 2.4, S. 11).

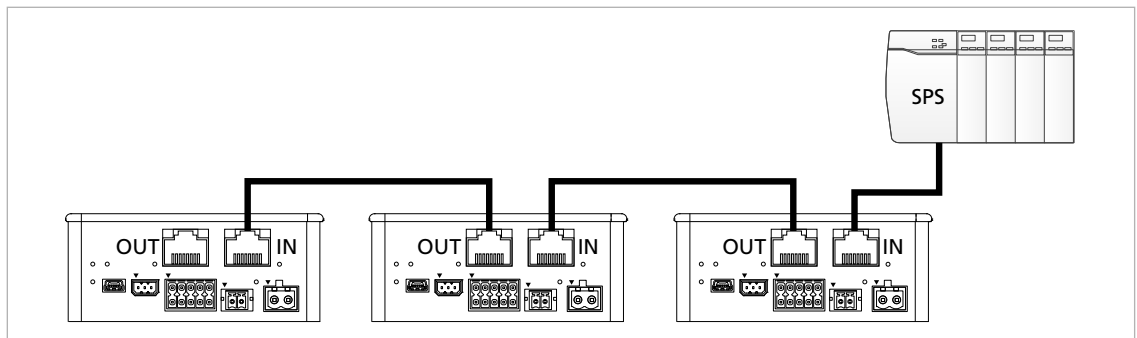


Abb. 2: Anschluss im EtherCAT-Netzwerk

Nach dem Einschalten und der Initialisierung befindet sich der Motion Controller zunächst im Zustand *Init*. Um Antriebsfunktionen ausführen zu können, muss der Motion Controller in den Zustand *Operational* gebracht werden.

2.4 ESI-Datei

Die ESI-Datei (EtherCAT Slave Information) enthält Informationen über den angeschlossenen Antrieb und sein Verhalten. Diese Informationen sind für die Kommunikation des EtherCAT-Masters mit dem Slave notwendig.

Die zum FAULHABER Motion Controller gehörende ESI-Datei liegt an folgenden Stellen vor:

- Als XML-Datei, die in einem Unterverzeichnis des Motion Manager Installationsverzeichnisses gespeichert ist
- In leicht vereinfachter Form auf dem EtherCAT-EEPROM des Motion Controllers (Slave Information Interface, siehe Kap. 3.8.2, S. 26)


Der entsprechend konfigurierte EtherCAT-Master kann die Informationen aus der ESI-Datei auslesen.

Der Master vergleicht die im Netzwerk gefundenen Antriebe mit den ihm zur Verfügung stehenden ESI-Dateien. Stimmen die Hersteller-Nummer (0147), der Produkt-Code und ggf. auch die Revisionsnummer überein, hat er die zum Antrieb passende ESI-Datei gefunden und kann den Antrieb mit den Einstellungen der ESI-Datei konfigurieren. In einer ESI-Datei können zur Abdeckung mehrerer Firmware-Versionen mehrere Revisionsnummern enthalten sein.

2.5 Identifikation eines Slaves

An einem Netzwerk hat der EtherCAT-Master folgende Möglichkeiten, einen Slave zu identifizieren:

- Identifikation über die Positionsnummer:
Jeder Slave besitzt durch seine Anordnung innerhalb des logischen Ethernet-Segments eine Nummer, über die er identifiziert werden kann. Die Nummerierung erfolgt ausgehend vom EtherCAT-Master in aufsteigender Reihenfolge (der 1. Slave nach dem EtherCAT-Master hat die Nummer 1, der darauffolgende Slave hat die Nummer 2 usw.)
- Identifikation über die Explicit Device ID:
Während der Konfigurationsphase setzt der Benutzer den Inhalt des Objekts 2400.08 (Explicit Device ID) auf einen beliebigen Wert und speichert ihn mit dem SAVE-Befehl im Anwendungs-EEPROM (siehe Kap. 3.11, S. 32). Während des Betriebs kann der EtherCAT-Master diese ID über EtherCAT-Mechanismen auslesen und mit einer vorher gespeicherten Version vergleichen. So können ausgetauschte Geräte oder vertauschte Kabel erkannt werden.

 Über den FAULHABER Motion Manager kann die Explicit Device ID anwenderfreundlich eingegeben und gespeichert werden.

EtherCAT-Kommunikation

3 EtherCAT-Kommunikation

3.1 Einführung

EtherCAT

EtherCAT ist eine Ethernet-basierende Kommunikationstechnologie. Für die Kommunikation mit EtherCAT wird ein EtherCAT-Master benötigt. Der EtherCAT-Master kontrolliert das Netzwerk und die Kommunikation mit den angeschlossenen EtherCAT-Slaves. Innerhalb eines EtherCAT-Netzwerks können mehr als 65000 Geräte in einem Segment adressiert werden. Da EtherCAT das Full-Duplex-Verfahren nutzt, werden Übertragungsgeschwindigkeiten von bis zu 100 MBit/s erreicht.

EtherCAT-Spezifikationen

Die für die FAULHABER-Antriebe wichtigen ETG-Spezifikationen definieren folgende Aspekte:

- ETG1000-Serie: EtherCAT-Technologie und Kommunikationsstruktur
- ETG2000-Serie: Spezifikation der EtherCAT Slave Information (ESI)
- ETG6010: Implementierung der CiA 402 Antriebsprofile

Für eine Reihe von Geräteklassen wurden CANopen-Geräteprofile definiert, wie zum Beispiel:

- CiA 402 für Antriebe
- CiA 401 für Ein- und Ausgabegeräte

3.2 Data Link Layer

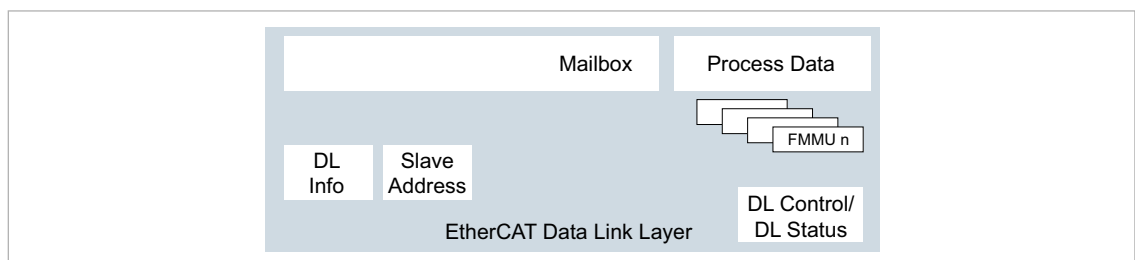


Abb. 3: Data Link Layer

Der Data Link Layer verbindet den Physical Layer mit dem Application Layer. Der Data Link Layer beinhaltet im Wesentlichen folgende Kontroll- und Kommunikationsdienste:

- Schnittstellenmanagement zum Physical Layer (siehe Kap. 3.2.1, S. 13)
- Schnittstellenmanagement zum Application Layer (siehe Kap. 3.2.4, S. 15)
- Zugang zum EtherCAT-EEPROM
- ESC-Konfiguration
- Distributed Clock (siehe Kap. 3.7.1, S. 23)
- Adressierung des EtherCAT-Slaves (siehe Kap. 3.2.3, S. 15)
- SyncManager-Management (siehe Kap. 3.2.2, S. 14)

EtherCAT-Kommunikation

3.2.1 EtherCAT-Frames und Datagramme

Frame-Struktur

Ein EtherCAT-Frame besteht aus bis zu 1518 Byte und kann bis zu 1450 Byte Nutzdaten beinhalten.

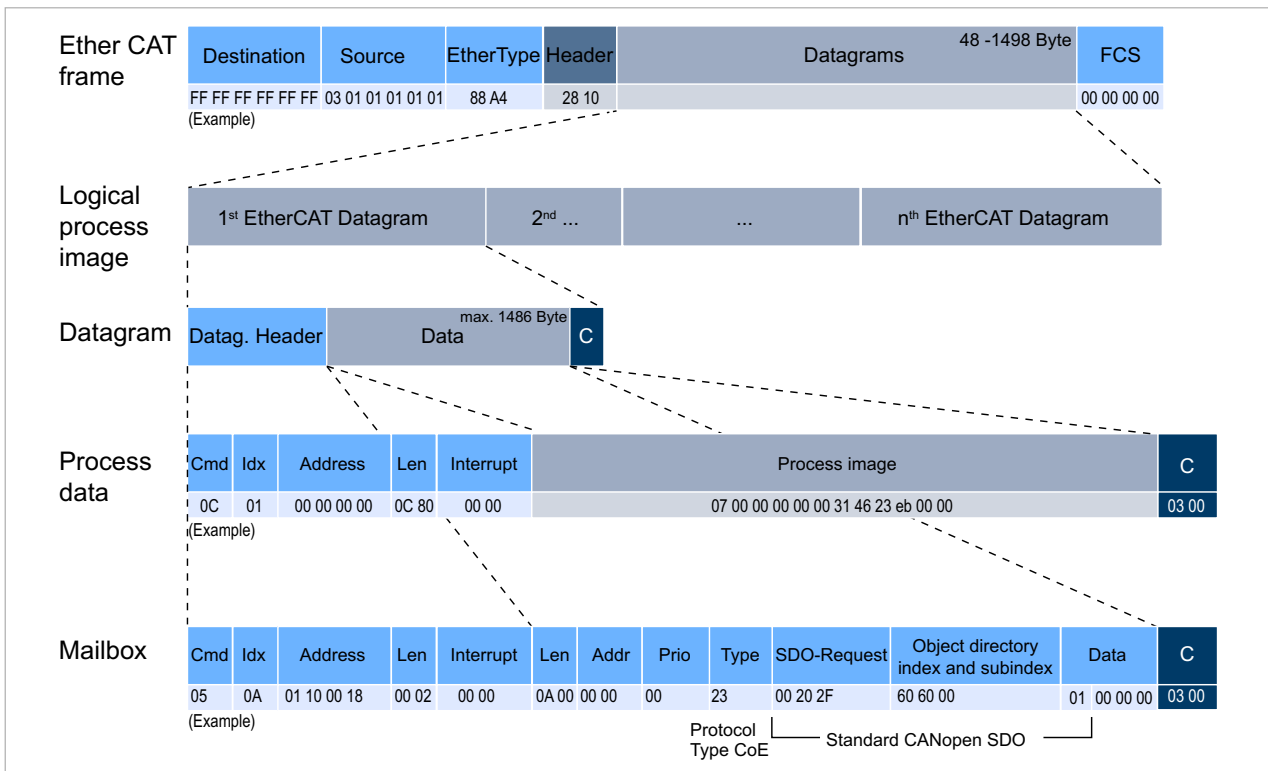


Abb. 4: Frame-Struktur

Ein EtherCAT-Frame setzt sich aus folgenden Datenbereichen zusammen:

- Ethernet-Header: Der Ethernet-Header enthält die Quell- und Zieladresse des Frames und die verwendete Protokollart.
- Datagramm(e): Ein oder mehrere Datagramme (siehe unten).
- Frame Check Sequence: Anhand dieser Daten wird die Fehlerfreiheit der Daten geprüft (siehe Kap. 3.10.2, S. 28).

Jedes Datagramm setzt sich aus folgenden Datenbereichen zusammen:

- Datagram-Header: Der Datagram-Header enthält Informationen über Kommunikationsart, Speicherzugriffsrechte, Adressen und Länge der Nutzdaten.
- Nutzdaten: Die Nutzdaten sind für Mailbox- und Prozessdaten-Kommunikation unterschiedlich aufgebaut und enthalten die bei CANopen verwendeten Servicedatenobjekte (SDOs) oder Prozessdatenobjekte (PDOs).
- Working Counter: Mit dem Working Counter werden Fehler im Datenaustausch erkannt (siehe Kap. 3.10.2, S. 28).

Die Prozessdatengröße pro EtherCAT-Slave ist nahezu beliebig groß und kann gegebenenfalls auf mehrere Datagramme verteilt (segmentiert) werden. Die Zusammensetzung der Prozessdaten kann sich in jedem Zyklus ändern.

EtherCAT-Kommunikation

Der Weg eines EtherCAT-Frames

Der EtherCAT-Master sendet auf einem Leitungspaar den EtherCAT-Frame zum ersten EtherCAT-Slave. Dieser verarbeitet den Frame und schickt ihn weiter zum nächsten EtherCAT-Slave. Auf diese Weise wird die Nachricht durch das ganze Netz geschickt und passiert jeden EtherCAT-Slave. Die EtherCAT-Slave-Controller (ESC) entnehmen dem EtherCAT-Frame im Durchlauf Daten und fügen ihre Daten ein. Der letzte EtherCAT-Slave im Netzwerk sendet den EtherCAT-Frame auf einem zweiten Leitungspaar zurück zum EtherCAT-Master.

3.2.2 SyncManager-Management

Datenübertragung durch den SyncManager

Die PDOs und SDOs werden vom SyncManager aus dem EtherCAT-Frame ausgelesen (Receive Parameter) bzw. in den EtherCAT-Frame eingebunden (Transmit Parameter). Zur Datenübertragung stehen vier Sync-Kanäle zur Verfügung:

SyncManager-Kanal	Funktion
0	Übertragung der Servicedaten aus dem EtherCAT-Frame in die Mailbox (Receive SDO)
1	Übertragung der Servicedaten aus der Mailbox in den EtherCAT-Frame (Transmit SDO)
2	Übertragung der Prozessdaten aus dem EtherCAT-Frame (Receive PDO 1/2/3/4)
3	Übertragung der Prozessdaten in den EtherCAT-Frame (Transmit PDO 1/2/3/4)

Für die Prozessdaten-Übertragung stehen die SyncManager-Objekte 0x1C12 und 0x1C13 zur Verfügung (siehe Kap. 4.1, S. 36).

Überwachung des Lese- und Schreibzugriffs

Der SyncManager schützt den Speicher für den Datenaustausch zwischen EtherCAT-Master und EtherCAT-Slave vor gleichzeitigem Zugriff. Damit wird verhindert, dass während des Lesens eines Speicherbereichs ein anderer Speicherbereich bereits überschrieben wird und damit die ausgelesenen Daten inkonsistent sind.

Für den Datenaustausch stehen 2 Typen von Speichern zur Verfügung:

Speichertyp	Beschreibung
Mailboxspeicher	<p>Der Mailboxspeicher besteht aus einem Speicherbereich.</p> <p>Der SyncManager führt folgende Funktionen aus:</p> <ul style="list-style-type: none"> Das Lesen des Speichers wird verhindert, während in den Speicher geschrieben wird. Das Schreiben des Speichers wird verhindert, während der Speicher gelesen wird. Der Speicher wird vor Überlaufen geschützt. <p>Dieser Speichertyp ist nicht für Echtzeitdaten geeignet und wird deshalb nur für Servicedaten verwendet.</p>
Pufferspeicher für Prozessdaten	<p>Der Pufferspeicher ist in 3 Pufferbereiche eingeteilt.</p> <p>Der SyncManager führt folgende Funktionen aus:</p> <ul style="list-style-type: none"> Zum Schreiben wird ein Pufferbereich gewählt, der zu diesem Zeitpunkt nicht gelesen wird. Der Lesezugriff auf den Speicher wird während des Schreibens gesperrt. Ein beschriebener Pufferbereich wird zum Lesen freigegeben. Der Schreibzugriff wird während des Lesens gesperrt. <p>So stehen zu jeder Zeit folgende Pufferbereiche zur Verfügung:</p> <ul style="list-style-type: none"> Pufferbereich 1, der gerade beschrieben wird Pufferbereich 2, der gerade gelesen wird Pufferbereich 3, der beschrieben ist und zum Auslesen bereitsteht (für den Fall, dass der Lesevorgang in Pufferbereich 2 vor dem Schreibvorgang in Pufferbereich 1 beendet ist) <p>Wenn der Schreibvorgang in Pufferbereich 1 vor dem Lesen des Pufferbereichs 2 beendet ist, werden die Daten in Pufferbereich 1 zum Lesen freigegeben und die Daten in Pufferbereich 3 werden verworfen.</p>

EtherCAT-Kommunikation

3.2.3 Adressierung

Das EtherCAT-Protokoll erlaubt folgende Adressierverfahren:

- **Positionsadressierung:** Die physikalische Positionen der EtherCAT-Slaves im Netzwerk dienen als Adressen. In jedem EtherCAT-Slave wird ein bestimmter Speicherbereich für die Adresse reserviert.
- **Knotenadressierung:** Konfigurierte Knotenadressen, die der EtherCAT-Master bei der Inbetriebnahme den EtherCAT-Slaves zuweist, dienen als Adressen. In jedem EtherCAT-Slave wird ein bestimmter Speicherbereich für die Adresse reserviert.
- **Logische Adressierung:** Der gesamte Speicherbereich des Netzwerks, d. h. die Speicherbereiche des EtherCAT-Masters und aller EtherCAT-Slaves, wird auf einen logischen Speicher abgebildet, der mit einem Parameter adressiert werden kann. Die Zuordnung der physikalischen Adressen der EtherCAT-Slaves zu den logischen Adressen wird im EtherCAT-Master hinterlegt. Sie wird in der Startphase an die Fieldbus Memory Management Units (FMMU) des Data Link Layers übergeben. Die FMMU wandelt die logische Adresse in eine physikalische Adresse um.

3.2.4 Schnittstellen zum Application Layer

Tab. 1: Data Link Schnittstellen zum Application Layer

Schnittstelle	Beschreibung
Mailbox	Über die Mailbox werden ausschließlich Daten ausgetauscht, die nicht zeitkritisch sind. Dazu zählen die Servicedaten. Die Servicedaten werden in Anlehnung an CANopen (CiA 301) als Servicedatenobjekt-Frames (SDO-Frames) übertragen (siehe Kap. 3.5, S. 19). Die Übertragung der Servicedaten erfolgt zyklisch.
Prozessdaten	Die Prozessdaten sind Echtzeitdaten. Dies bedeutet, dass immer die zuletzt gespeicherten Daten berücksichtigt werden. Nicht abgerufene Daten (z. B. bei Zykluszeiten, bei denen der EtherCAT-Slave die Daten nicht schnell genug verarbeiten kann) werden verworfen. Die Prozessdaten werden in Anlehnung an CANopen (CiA 301) als Prozessdatenobjekt-Frames (PDO-Frames) übertragen (siehe Kap. 3.4, S. 17). Die Übertragung der Prozessdaten erfolgt zyklisch.

EtherCAT-Kommunikation

3.3 Application Layer

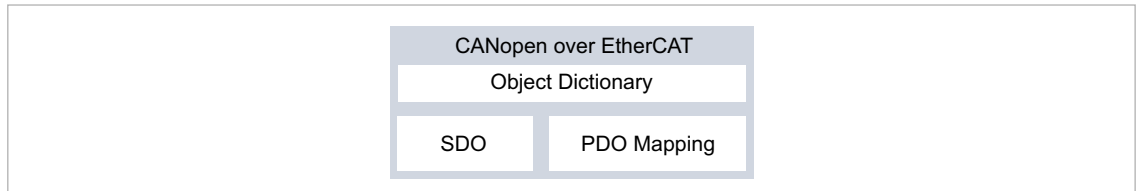


Abb. 5: Application Layer

CANopen over EtherCAT

Der FAULHABER Motion Controller unterstützt das CANopen over EtherCAT (CoE) Protokoll mit dem CANopen-Kommunikationsprofil gemäß CiA 301:

- 4 Sende-PDOs (TxPDOs)
- 4 Empfangs-PDOs (RxPDOs)
- 2 SDOs

Die CANopen-Telegramme können bis zu 250 Bytes lang sein und sind somit weit leistungsfähiger als das ursprüngliche CAN-Telegramm mit 8 Bytes.

Die CANopen-Antriebsprofile gemäß CiA 402 können für EtherCAT unverändert übernommen werden (siehe Funktionshandbuch).

Objektverzeichnis

Das Objektverzeichnis enthält Parameter, Soll- und Istwerte eines Antriebs. Das Objektverzeichnis ist das Bindeglied zwischen der Anwendung (Antriebsfunktionen) und den Kommunikationsdiensten. Alle Objekte im Objektverzeichnis sind über eine 16-Bit-Indexnummer (0x1000 bis 0x6FFF) und einen 8-Bit-Subindex (0x00 bis 0xFF) ansprechbar.

Index	Zuordnung der Objekte
0x1000 bis 0x1FFF	Kommunikationsobjekte
0x2000 bis 0x5FFF	Herstellerspezifische Objekte
0x6000 bis 0x6FFF	Objekte des Antriebsprofils nach CiA 402

Die Werte der Parameter können von Kommunikationsseite sowie von der Antriebsseite geändert werden.

Der Kommunikationsteil enthält Kommunikationsdienste entsprechend CiA 301.

Die Datenbelegung der PDOs ist entsprechend dem "PDO set for servo drive" nach CiA 402 voreingestellt.

EtherCAT-Kommunikation

3.4 PDO (Prozessdatenobjekt)

PDOs enthalten Prozessdaten zur Steuerung und Überwachung des Geräteverhaltens. Aus Sicht des Antriebs werden PDOs in Empfangs- und Sende-PDOs unterschieden.

- Empfangs-PDO (RxPDO): wird von einem Antrieb empfangen und enthält z. B. Steuerdaten
- Sende-PDO (TxPDO): wird von einem Antrieb gesendet und enthält z. B. Überwachungsdaten

PDOs werden nur ausgewertet oder übertragen, wenn sich das Gerät im NMT-Zustand *Operational* befindet (siehe Kap. 3.8.1, S. 25).

3.4.1 PDO-Konfiguration

- Maximal 4 Parameter können in einem PDO gemappt werden.
- Über die Objekte 0x1600 bis 0x1603 und 0x1A00 bis 0x1A03 kann die Datenbelegung der PDOs geändert werden. Die dafür notwendige Mapping-Prozedur ist in der CiA 301 beschrieben. Zur Durchführung der Mapping-Prozedur ist ein geeignetes Tool notwendig (z. B. FAULHABER Motion Manager oder Konfigurationswerkzeug der verwendeten SPS-Steuerung).

3.4.2 PDO-Mapping in der Standardkonfiguration

RxPDO1: Controlword

2 Byte Nutzdaten

LB	HB
----	----

Das RxPDO1 enthält das 16-Bit-Controlword nach CiA DSP402. Das Controlword steuert die State-Machine der Antriebseinheit und verweist auf den Objektindex 0x6040 im Objektverzeichnis. Die Bitaufteilung ist in der Dokumentation der Antriebsfunktionen beschrieben.

TxPDO1: Statusword

2 Byte Nutzdaten

LB	HB
----	----

Das TxPDO1 enthält das 16-Bit-Statusword nach CiA 402. Das Statusword zeigt den Zustand der Antriebseinheit an und verweist auf den Objektindex 0x6041 im Objektverzeichnis. Die Bitaufteilung ist in der Dokumentation der Antriebsfunktionen beschrieben.

RxPDO2: Controlword, Target Position (PP und CSP)

6 Byte Nutzdaten

LB	HB	LLB	LHB	HLB	HHB
----	----	-----	-----	-----	-----

Das RxPDO2 enthält das 16-Bit-Controlword und den 32-Bit-Wert der Zielposition (Objekt 0x607A) für den Profile Position Mode (PP)

TxPDO2: Statusword, Position Actual Value

6 Byte Nutzdaten

LB	HB	LLB	LHB	HLB	HHB
----	----	-----	-----	-----	-----

Das TxPDO2 enthält das 16-Bit-Statusword und den 32-Bit-Wert der Istposition (Objekt 0x6064).

EtherCAT-Kommunikation

RxPDO3: Controlword, Target Velocity (PV und CSV)

6 Byte Nutzdaten

LB	HB	LLB	LHB	HLB	HHB
----	----	-----	-----	-----	-----

Das RxPDO3 enthält das 16-Bit-Controlword und den 32-Bit-Wert der Soll Drehzahl (Objekt 0x60FF) für den Profile Velocity Mode (PV).

TxPDO3: Statusword, Velocity Actual Value

6 Byte Nutzdaten

LB	HB	LLB	LHB	HLB	HHB
----	----	-----	-----	-----	-----

Das TxPDO3 enthält das 16-Bit-Statusword und den 32-Bit-Wert der Ist Drehzahl (Objekt 0x606C).

RxPDO4: Controlword, Target Torque (PV und CSV)

6 Byte Nutzdaten

LB	HB	LLB	LHB	HLB	HHB
----	----	-----	-----	-----	-----

Das RxPDO4 enthält das 16-Bit-Controlword und den 16-Bit Wert des Soll-Drehmoments (Objekt 0x6071) für den Cyclic Torque Modus (CST).

TxPDO4: Statusword, Torque Actual Value

6 Byte Nutzdaten

LB	HB	LLB	LHB	HLB	HHB
----	----	-----	-----	-----	-----

Das TxPDO4 enthält das 16-Bit-Statusword und den 16-Bit Wert des Ist-Drehmoments (Objekt 0x6077) für den Cyclic Torque Modus (CST)

EtherCAT-Kommunikation

3.5 SDO (Servicedatenobjekt)

Das SDO liest und beschreibt Parameter im OV (Objektverzeichnis). Über den 16-Bit-Index und den 8-Bit-Subindex greift das SDO auf das Objektverzeichnis zu. Der Motion Controller stellt auf Anforderung des EtherCAT-Masters Daten zur Verfügung (Upload) bzw. empfängt Daten vom Master (Download).

Tab. 2: Allgemeine Strukturierung der SDO-Nutzdaten

Byte 0	Byte 1 bis 2	Byte 3	Byte 4 bis 7
Command-Specifier	16-Bit-Index	8-Bit-Subindex	4 Byte-Parameter-Data

Tab. 3: Einteilung der SDO-Übertragungsarten

Übertragungsart	Byteanzahl	Verwendungszweck
Expedited Transfer	maximal 250 Byte	–
Segmented Transfer	beliebige Größe	Übertragung von Datenblöcken (z. B. Trace-Puffer)

Die Transfer-Arten sind in der CiA 301 beschrieben.

3.5.1 SDO-Fehlerbeschreibung

Kann das SDO-Protokoll nicht verarbeitet werden, wird ein SDO-Abort-Telegramm versendet. Die Fehlerarten sind wie folgt codiert (siehe Tab. 2):

- Byte 4 + 5: Zusätzlicher Fehlercode LB + HB
- Byte 6: Fehlercode
- Byte 7: Fehlerklasse

Fehlerklasse	Fehlercode	Zusatzcode	Beschreibung
0x05	0x03	0x0000	Toggle-Bit nicht geändert
0x05	0x04	0x0001	SDO-Command-Specifier ungültig oder unbekannt
0x06	0x01	0x0000	Zugriff auf dieses Objekt wird nicht unterstützt
0x06	0x01	0x0001	Versuch, einen Write-Only-Parameter zu lesen
0x06	0x01	0x0002	Versuch, auf einen Read-Only-Parameter zu schreiben
0x06	0x02	0x0000	Objekt nicht im Objektverzeichnis vorhanden
0x06	0x04	0x0041	Objekt kann nicht in PDO gemappt werden
0x06	0x04	0x0042	Anzahl und/oder Länge der gemappten Objekte würde PDO-Länge überschreiten
0x06	0x04	0x0043	Allgemeine Parameter-Inkompatibilität
0x06	0x04	0x0047	Allgemeiner interner Inkompatibilitätsfehler im Gerät
0x06	0x07	0x0010	Datentyp oder Parameterlänge stimmen nicht überein oder sind unbekannt
0x06	0x07	0x0012	Datentypen stimmen nicht überein, Parameterlänge zu groß
0x06	0x07	0x0013	Datentypen stimmen nicht überein, Parameterlänge zu klein
0x06	0x09	0x0011	Subindex nicht vorhanden
0x06	0x09	0x0030	Allgemeiner Wertebereichfehler
0x06	0x09	0x0031	Wertebereichfehler: Parameterwert zu groß
0x06	0x09	0x0032	Wertebereichfehler: Parameterwert zu klein

EtherCAT-Kommunikation

Fehlerklasse	Fehlercode	Zusatzcode	Beschreibung
0x06	0x09	0x0036	Wertebereichfehler: Maximumwert kleiner als Minimumwert
0x08	0x00	0x0000	Allgemeiner SDO-Fehler
0x08	0x00	0x0020	Zugriff nicht möglich
0x08	0x00	0x0022	Zugriff bei aktuellem Gerätestatus nicht möglich

3.6 Emergency-Objekt (Fehlermeldung)

Emergency-Meldungen werden nicht wie bei CANopen autark vom Slave gesendet, sondern müssen vom EtherCAT-Master über das Mailbox-Protokoll abgefragt werden. Da dies ein langsamer Vorgang ist, wird von der Verwendung von Emergency abgeraten. Sinnvoller ist es, das Error-Register 1001h oder das Faulhaber-Fehlerregister 2320h in ein PDO zu mappen. Dann bekommt der Master Fehlerinformationen in kürzest möglicher Zeit.

Das Emergency-Object ist immer 8 Byte groß:

8 Byte Nutzdaten							
Error0(LB)	Error1(HB)	Error-Reg	FE0 (LB)	FE1 (HB)	0	0	0

Belegung der Nutzdaten:

- Error0(LB)/Error1(HB): 16-Bit-Error-Code
- Error-Reg: Error-Register (Inhalt von Objekt 0x1001, siehe Kap. 4.2, S. 44)
- FE0(LB)/FE1(HB): 16-Bit FAULHABER Fehlerregister (Inhalt von Objekt 0x2320, siehe Tab. 7)
- Bytes 5 bis 7: unbenutzt (0)

Das Error Register kennzeichnet die Fehlerart. Die einzelnen Fehlerarten sind bitcodiert und den jeweiligen Error Codes zugeordnet. Über das Objekt 0x1001 kann der letzte Wert des Error Registers abgefragt werden.

Maximal 3 Emergency können gespeichert werden. Wenn der EtherCAT-Master keine Emergency abfragt, werden die 3 ältesten gespeichert und die darauf folgenden verworfen. Dadurch lassen sich die Fehler feststellen, die zu weiteren Fehlern führten.

Tab. 4 listet alle Fehler auf, die über Emergency-Nachrichten gemeldet werden, sofern der entsprechende Fehler in der Emergency-Mask für das FAULHABER Fehlerregister gesetzt ist (Tab. 8).

EtherCAT-Kommunikation

Tab. 4: Emergency-Error-Codes

Emergency-Nachricht		FAULHABER-Fehlerregister 0x2320			Error Register 0x1001	
Error Code	Bezeichnung	Error Mask 0x2321	Bit	Bezeichnung	Bit	Bezeichnung
0x0000	No error (wird verschickt, wenn ein Fehler nicht mehr vorliegt bzw. bestätigt wurde)	–	–	–	–	–
–	–	–	–	–	0	Generic error (wird gesetzt, wenn eines der Fehlerbits 1 bis 7 gesetzt wird)
0x3210	Overvoltage	0x0004	2	OverVoltageError	2	Spannungsfehler
0x3220	Undervoltage	0x0008	3	UnderVoltageError	2	Spannungsfehler
0x43F0	Temperature Warning	0x0010	4	TempWarning	1	Strom-Fehler ^{a)}
0x4310	Temperature Error	0x0020	5	TempError	3	Temperatur-Fehler
0x5410	Output Stages	0x0080	7	IntHWError	7	Herstellerspezifischer Fehler
0x5530	EEPROM Fault	0x0400	10	MemError	–	–
0x6100	Software Error	0x1000	12	CalcError	7	Herstellerspezifischer Fehler
0x7200	Measurement Circuit: Current Measurement	0x0200	9	CurrentMeasError	7	Herstellerspezifischer Fehler
0x7300	Sensor Fault (Encoder)	0x0040	6	EncoderError	7	Herstellerspezifischer Fehler
0x7400	Computation Circuit: Module Fault	0x0100	8	ModuleError	7	Herstellerspezifischer Fehler
0x8110	CAN Overrun	0x0800	11	ComError	4	Kommunikations-Fehler
0x8130	CAN Guarding Failed					
0x8140	CAN Recovered From Bus-Off					
0x8310	RS232 overrun					
0x84F0	Deviation Error (Velocity Controller)	0x0001	0	SpeedDeviationError	5	Antriebsspezifischer Fehler
0x84FF	Max Speed Error	0x2000	13	DynamicError	7	Herstellerspezifischer Fehler
0x8611	FollowingError (Position Controller)	0x0002	1	FollowingError	5	Antriebsspezifischer Fehler

a) Der Stromregler hält den Motorstrom immer unter der eingestellten Grenze. Das Überstromfehler-Bit wird bei Überschreiten der Warnungstemperatur gesetzt und der zulässige Motorstrom wird vom Spitzenstrom-Wert auf den Dauerstrom-Wert reduziert.

EtherCAT-Kommunikation

Beispiel:

Eine Emergency-Nachricht mit der Nutzdatenbelegung in Tab. 5 wird in folgendem Fall versendet:

- In der Error Mask 0x2321 ist unter Subindex 1 (Emergency Mask) Bit 1 (Schleppfehler) gesetzt (siehe Tab. 9).
- Der in Objekt 0x6065.00 eingestellte Korridor für die Regelabweichung des Positionsreglers wurde für einen längeren Zeitraum überschritten, als der in Objekt 0x6066.00 eingestellte Wert für die Fehlerverzögerungszeit (siehe Dokumentation der Antriebsfunktionen).

Tab. 5: Beispielhafte Nutzdatenbelegung einer Emergency-Nachricht

8 Byte Nutzdaten							
0x11	0x86	0x20	0x02	0x00	0x00	0x00	0x00

3.7 Synchronisation

Der FAULHABER Motion Controller unterstützt die Synchronisation über Distributed Clocks und über ein SyncManager-Ereignis. Der von Ihnen verwendete EtherCAT-Master stellt die Synchronisation im FAULHABER Antrieb über dessen EtherCAT-Konfiguration ein. Diese Konfiguration geschieht normalerweise, ohne dass der Anwender dies bemerkt. Die folgenden Arten sind möglich:

- SM-Synchron: Synchronisation über ein SyncManager-Ereignis (siehe Kap. 3.7.2, S. 24)
- DC-Synchron: Synchronisation über Distributed Clocks (siehe Kap. 3.7.1, S. 23)

Konkret erkennt der FAULHABER Antrieb am Inhalt des EtherCAT-Registers 0x0980, welche Synchronisationsart der Master eingestellt hat:

- 0x0300 bedeutet DC-Synchron
- 0x0000 bedeutet SM-Synchron

Diese Werte sind in der ESI-Datei als Eintrag von <AssignActivate> definiert. Sie werden automatisch vom EtherCAT-Master eingestellt. Ein direkter Zugriff auf die EtherCAT-Register ist deshalb nicht erforderlich.

Die durch den Master vorgegebene Zykluszeit muss immer ein Vielfaches von 1 ms betragen. Die Grenzwerte der Zykluszeit sind wie folgt:

Modus	Minimale Zykluszeit	Maximale Zykluszeit
SM-Synchron	1 ms	100 ms
DC-Synchron	500 µs	50 ms

EtherCAT-Kommunikation

3.7.1 Synchronisation über Distributed Clocks (DC-Sync)

Jeder EtherCAT-Slave hat eine eigene Uhr, die vom ESC verwaltet wird. Die Uhrzeit des ersten EtherCAT-Slaves (Referenz-Slave) dient als Referenzzeit für das gesamte Netzwerk. Die Uhren aller anderen EtherCAT-Slaves und des EtherCAT-Masters gleichen ihre Zeit auf diese Referenzzeit ab.

Zur Synchronisation der Uhren sendet der EtherCAT-Master in kurzen Zeitabständen ein spezielles Datagramm, in das der EtherCAT-Slave mit der Referenzuhr seine aktuelle Uhrzeit einträgt. Alle anderen EtherCAT-Slaves und der EtherCAT-Master lesen diese Uhrzeit aus dem Datagramm. Da die EtherCAT-Teilnehmer eines Netzwerks in einer logischen Ringstruktur angeordnet sind, ist der erste EtherCAT-Slave nach dem EtherCAT-Master der Referenz-Slave.

Die von den EtherCAT-Teilnehmern eingelesene Referenzzeit wird jeweils um die Laufzeit des Datagramms von der Referenzuhr zum entsprechenden EtherCAT-Teilnehmer korrigiert. Zur Ermittlung dieser Laufzeiten sendet der EtherCAT-Master ein spezielles Datagramm an die EtherCAT-Slaves. Die ESCs schreiben bei Erhalt des Datagramms den Empfangszeitpunkt in ein Datagramm. Der EtherCAT-Master liest diese Empfangszeiten ein und verrechnet sie entsprechend.

Der ESC des Antriebs besitzt einen internen Taktgeber, der mit dem Taktgeber des Referenz-Slaves synchronisiert wird. Bei der Synchronisation wird die Telegramm-Laufzeit kompensiert. Der interne Taktgeber erzeugt ein Sync0-Signal, das den lokalen Zyklus des Antriebs startet.

Der lokale Zyklus benötigt Prozess-Daten aus einem EtherCAT-Telegramm, das bereits vorher empfangen und vorläufig gespeichert wurde. Wenn der lokale Zyklus durch das Sync0-Signal gestartet wird, liest er die gespeicherten Daten und führt die Regel-Schleife aus. Zum Schluss schreibt er die Eingangsdaten wieder in das Prozessabbild, damit sie dem Master zur Verfügung stehen.

Der Master sollte seine Telegramme im gleichen Takt wie die Zykluszeit des Slaves schicken, damit immer die aktuellen Daten vom Slave verarbeitet werden können. Wird durch ein Jitter im Takt des Masters ein Paket zu spät gesendet, kann es nicht mehr im aktuellen sondern erst im nächsten Reglerzyklus verarbeitet werden. Der aktuelle Reglerzyklus verwendet in diesem Fall die Daten des vorherigen Telegramms.

Die DC-Zykluszeit wird nicht über das Objekt 0x1C32.02 gesetzt, sondern der Master setzt sie direkt in den ESC-Registern. Die DC-Zykluszeit muss mindestens 500 µs oder ein Vielfaches von 1 ms betragen. Sie darf höchstens 50 ms betragen.

Einen Sonderfall stellt die Verwendung der Bahnkurven-Modi Profile Velocity (PV) oder Profile Position (PP) bei gleichzeitiger Synchronisation über Distributed Clocks (DC-Synchronisation) dar. Wird dabei eine Zykluszeit von 500 µs verwendet, kann es zu Kommunikationsfehlern kommen. Es wird dringend geraten, im Master mindestens eine Zykluszeit von 1 ms zu verwenden. Dies ist auch sinnvoll, da die PP/PV-Modi ihre Bahnkurven selbst erzeugen und deshalb nur auf sporadische Interaktion mit dem Master angewiesen sind.

3.7.2 Synchronisation über ein SyncManager-Ereignis (SM-Sync)

Der lokale Zyklus des EtherCAT-Slaves wird bei Empfang eines Prozessdaten-Telegramms (SyncManager-Ereignis) gestartet. Wenn RxPDOs übertragen werden, wird der EtherCAT-Slave auf das SyncManager2-Ereignis (SM2-Ereignis) synchronisiert. Wenn nur TxPDOs übertragen werden, wird der EtherCAT-Slave auf das SyncManager3-Ereignis (SM3-Ereignis) synchronisiert.

Überwachung des Prozessdateneingangs

Der Eintrag in 0x1C32.02 (Cycle Time) dient zur Überwachung der vom Master gesendeten Telegramme. Er muss im Zustand *Pre-Operational* eingestellt werden (siehe Kap. 4.1, S. 36).

Die Prozessdaten müssen im angegebenen Zeitraster beim Slave eintreffen. Tritt ein Fehler auf (z. B. Leitungsbruch) und es kommen keine Daten beim Slave an und ist der Slave entsprechend konfiguriert, sendet er ein Emergency und geht in den Fehlerzustand. Mit einer Zykluszeit von Null wird dieser Überwachungsmechanismus deaktiviert.

Die ESI-Datei beinhaltet auch den Eintrag *Cycle Time* mit dem Attribut *AdaptAutomatically*, das gemäß ETG2000 bewirkt, dass der EtherCAT-Master an dieser Stelle die Zykluszeit einträgt.

Es kann sein, dass der verwendete EtherCAT-Master das Attribut *AdaptAutomatically* nicht versteht. Er wird dann anstelle der tatsächlichen Zykluszeit den Wert 0 übertragen, was „keine Überwachung“ bedeutet. Der Antrieb kann auch mit dieser Einstellung problemlos betrieben werden.

Soll allerdings auch in diesem Fall überwacht werden, dass immer rechtzeitig Daten beim Slave ankommen, muss der Eintrag in der ESI-Datei manuell geändert werden. Der Standardwert 0 muss dabei durch die Zykluszeit des Masters in Nanosekunden ersetzt werden. Bei einer Zykluszeit von 4 ms muss also der Wert 4000000 als „little-endian“ Hexadezimal-Zahl eingetragen werden. Dazu müssen Sie die ESI mit einem Texteditor ändern. Die Anleitung dazu finden Sie im ESI unter dem Eintrag „Important Hint!“.

EtherCAT-Kommunikation

3.8 Layer management

Das Layer Management stellt folgende Dienste zur Verfügung:

- Steuerung der EtherCAT-Statemachine (Kap. 3.8.1, S. 25)
- Auslesen und Schreiben des Slave Information Interface (siehe Kap. 3.8.2, S. 26)

Zur Ausübung dieser Dienste kommuniziert der EtherCAT-Master direkt mit dem ESC.

3.8.1 Steuerung der EtherCAT-Statemachine

Nach dem Einschalten und der Initialisierung befindet sich der Motion Controller automatisch im Zustand *Pre-Operational*. Im Zustand *Pre-Operational* kann nur per Mailbox-Kommunikation mit dem Gerät kommuniziert werden.

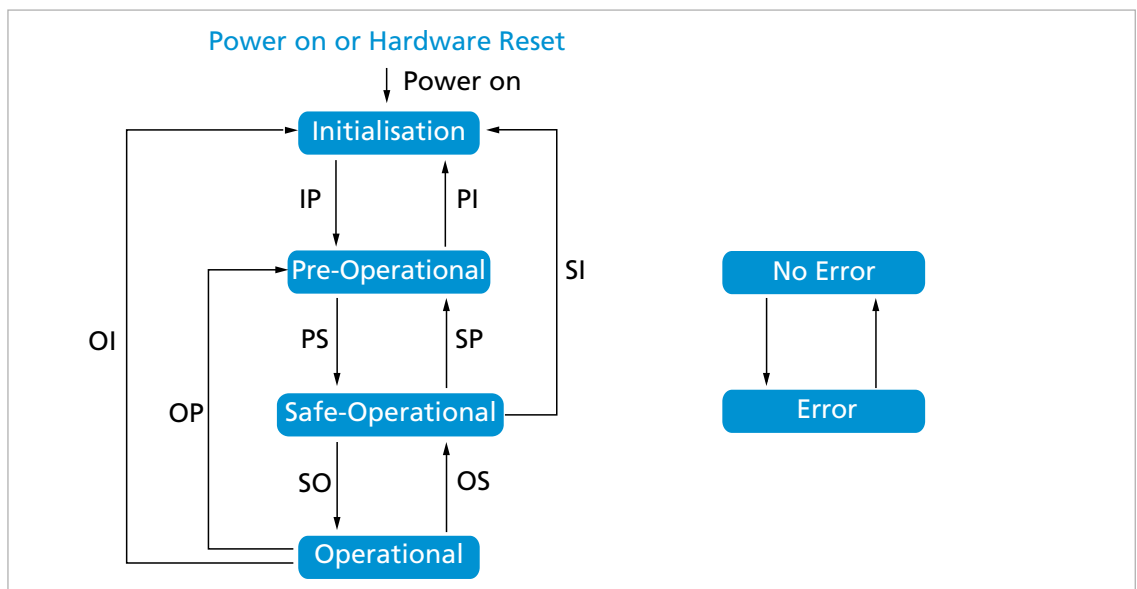


Abb. 6: EtherCAT-Statemachine

Tab. 6: Zustandsänderungen

Übergang	Aktionen
Power on	<ul style="list-style-type: none"> ■ Der Initialisierungs-Status wird beim Einschalten selbsttätig erreicht. ■ Mailbox-Kommunikation und Prozessdatenkommunikation sind nicht möglich. ■ Der EtherCAT-Master initialisiert die SyncManager-Kanäle für die Mailbox-Kommunikation.
IP	<ul style="list-style-type: none"> ■ Der EtherCAT-Master synchronisiert den EtherCAT-Feldbus. ■ Der EtherCAT-Master initialisiert die SyncManager-Kanäle für die Prozessdaten-Kommunikation, die FMMU-Kanäle und das SyncManager-PDO-Assignment. ■ Die Mailbox-Kommunikation wird zwischen dem EtherCAT-Master und den EtherCAT-Slaves aufgebaut. ■ Einstellungen für die Prozessdatenübertragung werden übertragen.
PS	<ul style="list-style-type: none"> ■ Der EtherCAT-Slave prüft, ob die SyncManager-Kanäle für die Prozessdaten-Kommunikation und die Einstellungen für die Distributed Clocks korrekt sind. ■ Der EtherCAT-Slave kopiert aktuelle Eingangsdaten in die Speicherbereiche des ESC. ■ Mailbox- und Prozessdaten-Kommunikation sind möglich. Die Ausgänge des EtherCAT-Slaves bleiben in einem sicheren Zustand und werden nicht ausgegeben. Die Eingangsdaten werden zyklisch aktualisiert.
SO	<ul style="list-style-type: none"> ■ Der EtherCAT-Master überträgt gültige Ausgangsdaten an den EtherCAT-Slave. ■ Der EtherCAT-Master schaltet den EtherCAT-Slave in den Zustand <i>Operational</i>. ■ Im Zustand <i>Operational</i> kopiert der EtherCAT-Slave die Eingangsdaten auf seine Ausgänge. ■ Mailbox- und Prozessdaten-Kommunikation sind möglich.

EtherCAT-Kommunikation

Die ESI-Datei für die FAULHABER Motion Controller enthält die Standardkonfiguration für alle Objekte (siehe Kap. 2.4, S. 11). Im Regelfall ist keine weitere Parametrisierung beim Systemstart notwendig.

Notwendige Parametereinstellungen können mit dem FAULHABER Motion Manager über die USB-Schnittstelle durchgeführt und dauerhaft in den EEPROM gespeichert werden (siehe Kap. 3.11, S. 32). Einstellungen im EEPROM sind nach dem Systemstart sofort verfügbar.

i Im Zustand *Init* werden alle Werte des Antriebs auf die Einschalt-Werte zurückgesetzt. Vorher in einem anderen Zustand vom Benutzer gesetzte Werte werden überschrieben, wenn diese nicht per Speichern-Kommando 1010h gesichert wurden. Ist dieses Verhalten unerwünscht, darf nicht in den Zustand *Init* gewechselt werden, sondern der Antrieb muss mindestens im Zustand *Pre-Operational* bleiben.

i Der Antrieb wird über Objekte des Antriebsprofils (Controlword, Statusword) gesteuert. Die Kommunikation mit dem Antrieb und die zugehörigen Betriebsarten sind im separaten Handbuch „Funktionshandbuch“ beschrieben.

Der Wechsel in den Zustand *Pre-Operational* benötigt einige Millisekunden. Der Master muss das AL-Register (130h) abfragen und warten, bis der Zustand erfolgreich gewechselt ist. Vorher ist keine SDO-Kommunikation möglich.

3.8.2 Slave Information Interface (SII)

Das Slave Information Interface beinhaltet Daten, die für den EtherCAT-Slave und den angeschlossenen Antrieb spezifisch sind (z. B. Werte des Objekts 0x1018) sowie die Mailbox-SyncManager-Konfiguration.

Diese Daten sind im EtherCAT-EEPROM gespeichert, das bei der Inbetriebnahme des Netzwerks ausgelesen wird (siehe Kap. 2.4, S. 11).

3.9 Einträge im Objektverzeichnis

Das Objektverzeichnis verwaltet die Konfigurationsparameter. Das Objektverzeichnis ist in drei Bereiche unterteilt. Jedes Objekt kann über seinen Index und Subindex referenziert werden (SDO-Protokoll).

- Kommunikationsparameter (Index 0x1000 bis 0x1FFF, enthält Kommunikationsobjekte nach CiA 301, siehe Kap. 4.1, S. 36)
- Herstellerspezifischer Bereich (Index 0x2000 bis 0x5FFF, enthält herstellerspezifische Objekte, siehe Kap. 4.2, S. 44)
- Standardisierte Geräteprofile (0x6000 bis 0x9FFF, enthält die vom Motion Controller unterstützten Objekte, siehe Dokumentation der Antriebsfunktionen)

EtherCAT-Kommunikation

3.10 Fehlerbehandlung

3.10.1 Gerätefehler

Tab. 7: FAULHABER Fehlerregister (0x2320)

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x2320	0x00	Fault Register	U16	ro	–	FAULHABER Fehlerregister

Das FAULHABER Fehlerregister enthält bitcodiert die zuletzt aufgetretenen Fehler. Die Fehler können durch Selektion der gewünschten Fehlerarten über das Objekt Error Mask (0x2321) maskiert werden.

Tab. 8: Fehlercodierung

Error-Bit	Fehlermeldung	Beschreibung
0x0001	SpeedDeviationError	Geschwindigkeitsabweichung zu groß
0x0002	FollowingError	Schleppfehler
0x0004	OverVoltageError	Überspannung detektiert
0x0008	UnderVoltageError	Unterspannung detektiert
0x0010	TempWarning	Temperatur überschritten, bei der eine Warnung ausgegeben wird
0x0020	TempError	Temperatur überschritten, bei der eine Fehlermeldung ausgegeben wird
0x0040	EncoderError	Fehler am Encoder detektiert
0x0080	IntHWError	Interner Hardwarefehler
0x0100	ModuleError	Fehler am externen Modul
0x0200	CurrentMeasError	Strommessfehler
0x0400	MemError	Speicherfehler (EEPROM)
0x0800	ComError	Kommunikationsfehler
0x1000	CalcError	Interner Softwarefehler
0x2000	DynamicError	Aktuelle Geschwindigkeit ist größer als die eingestellte Maximalgeschwindigkeit des Motors.
0x4000	–	Nicht verwendet, Wert = 0
0x8000	–	Nicht verwendet, Wert = 0

Jeder dieser Fehler entspricht auch einem Emergency Error Code. (siehe Kap. 3.6, S. 20).

Die Error Mask beschreibt die Behandlung interner Fehler entsprechend der Fehlercodierung (siehe Tab. 8).

EtherCAT-Kommunikation

Tab. 9: Error Mask (0x2321)

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x2321	0x00	Number of Entries	U8	ro	6	Anzahl Objekteinträge
	0x01	Emergency Mask	U16	rw	0xFFFF	Fehler, für die eine Fehlermeldung verschickt werden
	0x02	Fault Mask	U16	rw	0x0000	Fehler, für die die Zustandsmaschine des Antriebs in den Zustand <i>Fault Reaction Active</i> geht
	0x03	Error Out Mask	U16	rw	0x0000	Fehler, für die der Fehler-Ausgangspin gesetzt wird
	0x04	Disable Voltage Mask	U16	ro	0x4024	Fehler, die den Antrieb abschalten (nicht konfigurierbar)
	0x05	Disable Voltage User Mask	U16	rw	0x0000	Fehler, die den Antrieb abschalten (konfigurierbar)
	0x06	Quick Stop Mask	U16	rw	0x0000	Fehler, für die die Zustandsmaschine des Antriebs in den Zustand <i>Quick Stop Active</i> geht

Beispiele:

- Beim Setzen der Fault Mask (Subindex 2) von Objekt 0x2321 auf 0x0001 wird der Antrieb bei Überstrom ausgeschaltet und dessen Zustandsmaschine wird in den Zustand *Fault Reaction Active* versetzt.
- Wenn der Subindex 3 von Objekt 0x2321 auf 0 gesetzt ist, zeigt der Fehlerausgang (Fault-Pin) keine Fehler an. Wenn der Subindex 3 von Objekt 0x2321 auf 0xFFFF gesetzt ist, zeigt der Fehlerausgang (Fault-Pin) alle Fehler an.

3.10.2 Kommunikationsfehler

Das Netzwerk wird sowohl auf fehlerhafte Kommunikationsdaten als auch auf fehlende Daten überwacht. Beim Auftreten eines Fehlers können somit die Antriebe in einen sicheren Zustand gebracht und Fehlermeldungen angezeigt werden. Zur Lokalisierung und Behebung eines Fehlers muss der Netzwerkverkehr analysiert werden.

3.10.2.1 Fehlerprüfung über EtherCAT-Frame-Einträge

Da der EtherCAT-Slave nicht direkt mit dem EtherCAT-Master kommunizieren kann, erfolgt die Überwachung auf fehlerhafte Daten über Einträge im EtherCAT-Frame.

- Frame Check Sequence (FCS): Der ESC prüft mit Hilfe einer Prüfsumme den durchlaufenden EtherCAT-Frame auf Fehler. Nur wenn die Prüfung positiv ist, werden die Informationen aus dem EtherCAT-Frame verwertet. Wenn die Prüfung negativ ist, wird der EtherCAT-Frame durch das Erhöhen des Zählerwerts für die nachfolgenden EtherCAT-Slaves und den EtherCAT-Master als fehlerhaft markiert.
- Working Counter: Der Working Counter ist Teile des Datagramms. Der EtherCAT-Slave erhöht nach erfolgreichem Datenaustausch den Zählerwert um 1. Der EtherCAT-Master vergleicht den Zählerwert des zurückgekommenen EtherCAT-Datagramms mit dem erwarteten Zählerwert und kann dadurch Fehler im Datenaustausch erkennen.

EtherCAT-Kommunikation

3.10.2.2 Fehlerreaktion

Der Antrieb muss vom EtherCAT-Master zum passenden Zeitpunkt Ausgabedaten bekommen und regelmäßig seinen Zustand an den EtherCAT-Master senden können. Dazu muss der EtherCAT-Slave in regelmäßigen Zeitabständen Prozessdaten erhalten. Schwankungen beim Datenempfang müssen innerhalb bestimmter Grenzen bleiben.

Der Austausch des Prozessabbilds wird von zwei prinzipiell verschiedenen Mechanismen im Antrieb überwacht:

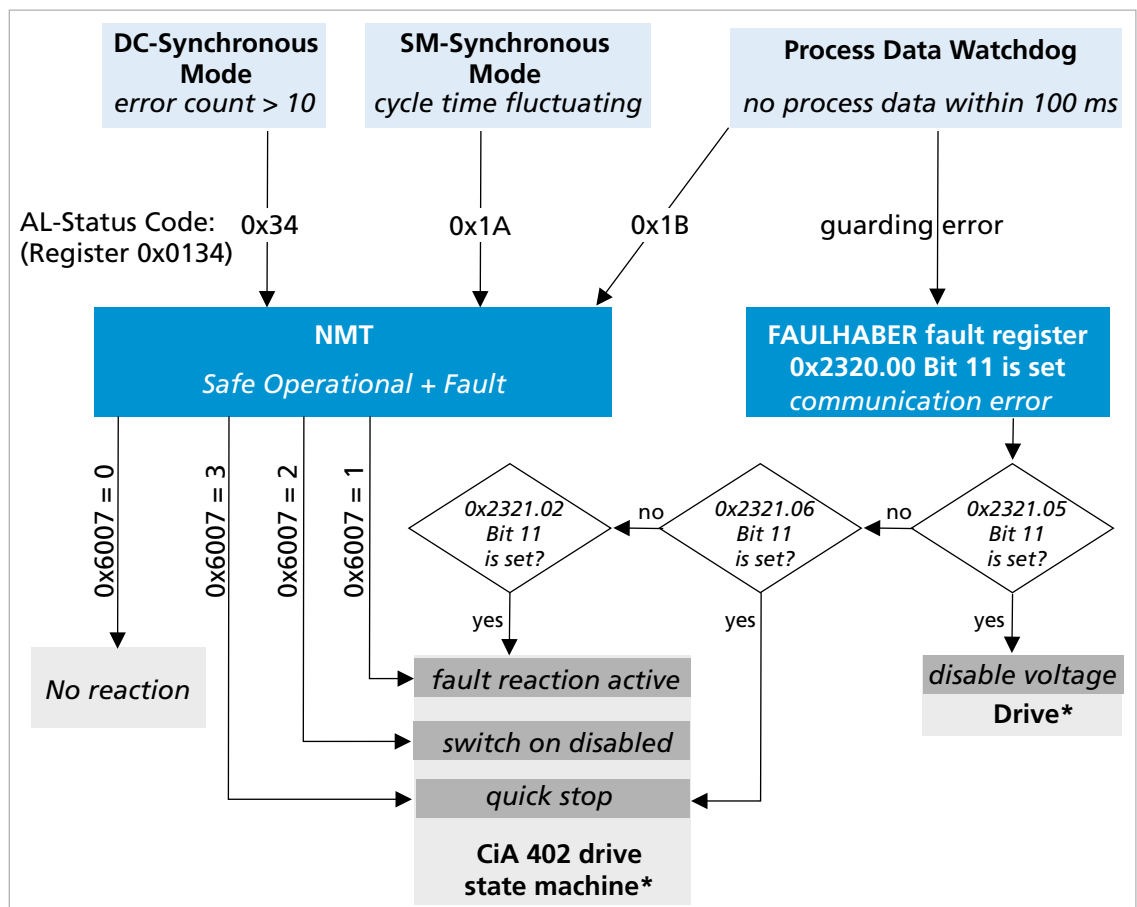


Abb. 7: Mechanismen der Fehlerüberwachung

* Die Objekte des Antriebsprofils nach CiA 402 sind ausführlich in der Dokumentation der Antriebsfunktionen beschrieben

Überwachung des Eintreffzeitpunkts der Prozessdaten

Beim Eintreffen der Prozessdaten beim Antrieb wird geprüft, ob der Zeitpunkt des Eintreffens mit dem erwarteten Zeitpunkt übereinstimmt. Ist die Abweichung mehrmals zu hoch, geht die EtherCAT-Zustandsmaschine in die Zustände *Safe Operational* und *Error* über. Abhängig von der eingestellten Fehlerreaktion in Objekt 0x6007 bremsst der Antrieb auf Stillstand ab (siehe Dokumentation der Antriebsfunktionen).

Diese Art der Überwachung ist abhängig von der Synchronisationsart unterschiedlich benannt:

- DC-Synchronisation: SYNC0-Überwachung
- SM-Synchronisation: SyncManager-Überwachung

EtherCAT-Kommunikation

Überwachung des Prozessdatenempfangs

Wenn beim Antrieb keine Prozessdaten ankommen (z. B. wegen Kabelbruch), wird die Überwachung des Eintreffzeitpunkts nicht aktiviert. Wenn länger als 100 ms keine Prozessdaten empfangen werden, löst der Prozessdaten-Watchdog eine Fehlerreaktion aus. Die Fehlerreaktion hängt vom Zustand von Bit 11 im FAULHABER Fehlerregister 0x2320.00 ab:

- Bit 11 nicht gesetzt:
Die EtherCAT-Zustandsmaschine geht in die Zustände *Safe Operational* und *Error* über. Abhängig von der eingestellten Fehlerreaktion in Objekt 0x6007 bremsst der Antrieb auf Stillstand ab (siehe Dokumentation der Antriebsfunktionen).
- Bit 11 gesetzt:
Der Antrieb hält an, wie es in den Objekten 0x2321.02, 0x2321.04, 0x2321.05 und 0x2321.06 definiert ist (siehe Kap. 4.2, S. 44).

3.10.2.3 Analyse des Netzwerkverkehrs

Der Netzwerkverkehr kann mit Software-Tools (z. B. *Wireshark*) analysiert werden. Das Software-Tool kann entweder auf einem separaten PC, der an das Netzwerk angeschlossen wird, oder direkt auf dem EtherCAT-Master installiert sein. Die Analyse des Netzwerkverkehrs besteht darin, dass der vom EtherCAT-Master gesendete Frame und der empfangene Frame ausgelesen und miteinander verglichen werden. Besonders markante Stellen für die Fehleranalyse sind die oben erwähnten EtherCAT-Frame-Einträge (FCS, Working Counter).

3.10.2.4 EtherCAT AL-Status-Codes und Fehlerbehebung

Bei Kommunikationsfehler wird das AL-Status-Code-Register (0x0134) mit einem Fehlercode geladen. In der folgenden Tabelle sind die möglichen Codes beschrieben und Maßnahmen zur Fehlerbehebung aufgezeigt.

AL-Status-Code	Beschreibung	Fehlerbehebung
0x0001	EtherCAT-System (Hardware oder Software) konnte nicht korrekt initialisiert werden.	FAULHABER-Support informieren.
0x0002	Für einen internen Puffer eines SyncManager (Mapping oder Pufferung) steht kein Speicher zur Verfügung.	FAULHABER-Support informieren.
0x0011	Der angeforderte Ziel-Zustand kann nicht direkt erreicht werden bzw. die Transition ist nicht erlaubt.	Die AL-Zustandsmaschine nur in gültigen Schritten schalten. Ungültig sind zum Beispiel: <ul style="list-style-type: none"> ▪ Init → Safe-Operational ▪ Init → Operational ▪ Pre-Operational → Operational
0x0012	Den angeforderten Ziel-Zustand gibt es nicht.	Gültige AL-Zustands-Codes verwenden. Gültige Codes sind: <ul style="list-style-type: none"> ▪ Init: 1 ▪ Pre-Operational: 2 ▪ Safe-Operational: 4 ▪ Operational: 8
0x0013	Der Zustand <i>Boot State</i> ist in diesem Produkt nicht implementiert.	Nicht in den Boot-Zustand wechseln.
0x0015	Beim Wechsel in den Boot-Zustand wurde ein Fehler bei der Konfiguration der Mailbox (SDO-Kommunikation) erkannt.	Siehe 0x0016.

EtherCAT-Kommunikation

AL-Status-Code	Beschreibung	Fehlerbehebung
0x0016	Beim Wechsel in den Zustand <i>Pre-Operational</i> wurde ein Fehler bei der Konfiguration der Mailbox (SDO-Kommunikation) erkannt.	Konfiguration prüfen: <ul style="list-style-type: none"> SM0 und SM1 müssen eingeschaltet sein Die Control Bytes der SM müssen stimmen Die physikalischen und konfigurierten Adressen der SM müssen stimmen Die Länge der SM muss innerhalb der erlaubten Grenzen sein.
0x0017	Beim Mapping eines SM wurde ein Fehler festgestellt.	Konfiguration prüfen: <ul style="list-style-type: none"> Die Länge der gemappten Objekte darf die Länge des SM nicht übersteigen Maximal 8 SM können verwendet werden SM dürfen nicht überlappen
0x001A	Synchronisations-Fehler: Die Fehler-Schwelle ist > 0 gesetzt und der interne Fehlerzähler hat diese Schwelle überschritten, weil das Prozessabbild zu schnell kam. Nur im SM-synchronen Modus.	Prozessabbild langsamer schicken. oder: Die in 0x1C32.02 gesetzte Zykluszeit prüfen. Sie muss der tatsächlichen Zykluszeit entsprechen.
0x001B	Synchronisations-Fehler: Der SM-Watchdog hat eine Zeitüberschreitung des Prozessabbaus festgestellt, weil für mehr als 100 ms keine Prozessdaten geschickt wurden.	Prozessabbild schicken.
0x001D	Ein Output-SM ist falsch konfiguriert.	SM richtig konfigurieren: <ul style="list-style-type: none"> SM mit Länge 0 müssen ausgeschaltet sein, die anderen müssen eingeschaltet werden Das Kontroll-Byte des SM muss stimmen Physikalische und konfigurierte Adresse des SM müssen übereinstimmen Die Länge des SM muss innerhalb der erlaubten Grenzen sein SM dürfen nicht überlappen
0x001E	Ein Input-SM ist falsch konfiguriert.	SM richtig konfigurieren: <ul style="list-style-type: none"> SM mit Länge 0 müssen ausgeschaltet sein, die anderen müssen eingeschaltet werden Das Kontroll-Byte des SM muss stimmen Physikalische und konfigurierte Adresse des SM müssen übereinstimmen Die Länge des SM muss innerhalb der erlaubten Grenzen sein SM dürfen nicht überlappen
0x0026	Ungültige Einstellungen für SyncManager.	Das ECAT-Register 0x0980 muss vom EtherCAT-Master richtig gesetzt werden, siehe Kap. 3.7, S. 22. Das EtherCAT-Register 0x0980 muss entweder 0x0000 oder 0x0300 sein.
0x002C	Im DC wird kein SYNC0-Signal mehr empfangen.	Konfiguration des SYNC0 prüfen: <ul style="list-style-type: none"> SM: Zykluszeit mindestens 1 ms, maximal 100 ms DC: Zykluszeit mindestens 500 µs, maximal 50 ms
0x002E	Zykluszeit ist zu klein.	Die Zykluszeit muss immer ein Vielfaches von 1 ms sein. Nur im Modus DC-Synchron ist daneben noch eine Zykluszeit von 0,5 ms erlaubt.
0x0030	DC ist nicht richtig konfiguriert.	Konfiguration für DC prüfen: Das ECAT-Register 0x0980 muss vom EtherCAT-Master richtig gesetzt werden, siehe Kap. 3.7, S. 22. Das EtherCAT-Register 0x0980 muss 0x0300 sein.

EtherCAT-Kommunikation

AL-Status-Code	Beschreibung	Fehlerbehebung
0x0034	Bevor Prozessdaten verarbeitet werden können, muss der Slave sie erhalten. Der Master hatte diese Daten zu oft nicht rechtzeitig gesendet. Nur im DC-Modus.	Prozessabbild rechtzeitig schicken, damit es beim SYNC0-Impuls verwendet werden kann.
0x0036	Die DC-Zeit ist zu klein.	Die Zykluszeit für DC muss mindestens 500 µs betragen.
0xB001	Für den SyncManager-Kanal wurden keine PDOs gefunden.	FAULHABER-Support informieren.
0xB003	PDO-Mapping gibt es nicht.	FAULHABER-Support informieren.
0xB004	PDO-Index ist nicht im erforderlichen Bereich.	FAULHABER-Support informieren.
0xB007	PDO-Eintrag existiert nicht im Objektverzeichnis.	FAULHABER-Support informieren.

3.11 Parameter speichern und wiederherstellen

Damit geänderte Parameter im OV auch nach erneutem Einschalten des Controllers erhalten bleiben, müssen sie mit dem Save-Befehl dauerhaft in den nicht-flüchtigen Speicher (Anwendungs-EEPROM) gespeichert werden (siehe Kap. 4.1, S. 36). Beim Einschalten des Motors werden die Parameter automatisch aus dem nicht-flüchtigen Speicher in den flüchtigen Speicher (RAM) geladen.

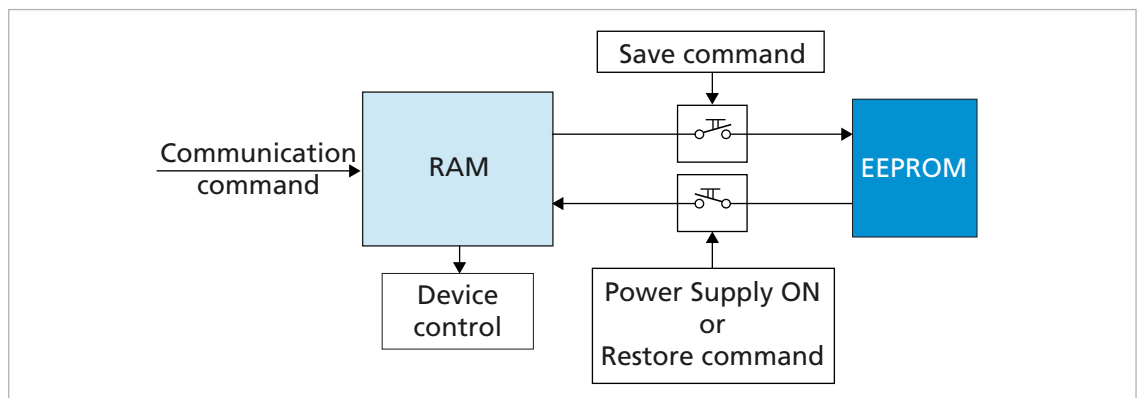


Abb. 8: Parameter speichern und wiederherstellen

Folgende Parameter können mit dem Restore-Befehl geladen werden (siehe Kap. 4.1, S. 36):

- Werkseinstellungen
- Mit dem Save-Befehl gespeicherte Parameter


EtherCAT-Kommunikation

3.11.1 Parameter speichern

Die aktuellen Parametereinstellungen können komplett oder für einzelne Bereiche im internen EEPROM gespeichert werden (SAVE) (siehe Tab. 10).


- ▶ Auf Subindex 01 bis 05 des Objekts 0x1010 die Signatur "save" schreiben (siehe Tab. 11).

3.11.2 Einstellungen wiederherstellen

 Abgespeicherte Parameter werden beim Einschalten des Antriebs automatisch geladen.

Werkseinstellungen oder zuletzt gespeicherte Parametereinstellungen können jederzeit komplett oder für einzelne Bereiche aus dem internen EEPROM geladen werden (RESTORE) (siehe Tab. 12).

1. Auf Subindex 01 bis 06 des Objekts 0x1011 die Signatur "load" schreiben (siehe Tab. 13).
 - ↪ Nach Restore Factory (01), Restore Communication (02) und Restore Application (03) muss der Antrieb zurückgesetzt werden. Nur dann werden die Parameter aktualisiert.
2. Applikationsparameter (04) sowie Satz 1 und Satz 2 der speziellen Applikationsparameter (05/06) mit dem Reload-Befehl aktualisieren.
 - ↪ Der Reload-Befehl überschreibt die zuletzt als Anwenderparameter gespeicherten Werte.

 Sollen die aktuell geladenen Werte auch nach einem Restore zur Verfügung stehen, müssen diese mit einem geeigneten Programm (z. B. FAULHABER Motion Manager) auf dem PC gesichert werden.

EtherCAT-Kommunikation

3.11.3 Parametersatz wechseln

Die Ablage der Applikationsparameter (Motordaten, I/O-Konfiguration, Reglerparameter, Betriebsart etc.) umfasst einen gemeinsamen Basissatz von Parametern (App) und daneben einen Speicherbereich für Parameter, die häufig an unterschiedliche Lastsituationen angepasst werden müssen (App1/App2):

Drehzahlregler und Filter

Index	Subindex	Name	Typ	Attr.	Bedeutung
0x2344	0x01	Gain K_p	U32	rw	Reglerverstärkung [As $1e^{-6}$]
	0x02	Integral Time T_N	U16	rw	Reglernachstellzeit [100 μ s]
0x2346	0x01	Set Point Velocity Filter Time T_F	U16	rw	Filterzeit T_F [100 μ s]
	0x02	Setpoint Filter Enable	U8	rw	0: inactive 1: active
0x2347	0x01	Gain Factor	U8	rw	Gain Faktor (wird bei der Geschwindigkeitsregelung im PP Mode auf K_p angewendet) 0: Verstärkung des Geschwindigkeitsreglers wird im Ziel auf 0 reduziert 128: keine Variable Verstärkung 255: Verstärkung des Geschwindigkeitsreglers wird im Ziel verdoppelt

Positionsregler

Index	Subindex	Name	Typ	Attr.	Bedeutung
0x2348	0x00	Number of Entries	U8	ro	Anzahl Objekteinträge
	0x01	K_v [1/s]	U8	rw	Range: 1-250

Vorsteuerungen

Index	Subindex	Name	Typ	Attr.	Bedeutung
0x2349	0x01	Torque/Force FeedForward Factor	U8	rw	Faktor der Drehmoment- bzw. Kraftvorsteuerung 0: 0% Aufschaltung des Vorsteuerwerts 128: 100% Vorsteuerung
	0x02	Torque/Force FeedForward Delay	U8	rw	Sollwertverzögerung: 0: unverzögerte Aufschaltung 1: Aufschaltung um eine Abtastung verzögert
0x234A	0x01	Velocity Feedforward Factor	U8	rw	Faktor der Drehmoment- bzw. Kraftvorsteuerung 0: 0% Vorsteuerung 128: 100% Vorsteuerung
	0x02	Velocity FeedForward Delay	U8	rw	Sollwertverzögerung: 0: unverzögerte Aufschaltung 1: Aufschaltung um eine Abtastung verzögert

EtherCAT-Kommunikation

Allgemeine Einstellungen

Index	Subindex	Name	Typ	Attr.	Bedeutung
0x6060	0x00	Modes of Operation	S8	rw	Auswahl der Betriebsart -4: ATC -3: AVC -2: APC -1: Volt Mode 0: Regler nicht aktiviert 1: PP 3: PV 6: Homing 8: CSP 9: CSV 10: CST
0x6081	0x00	Profile Velocity	U32	rw	Profile Velocity [in benutzerdefinierten Einheiten]
0x6083	0x00	Profile Acceleration	U32	rw	Profile Acceleration [1/s ²]
0x6084	0x00	Profile Deceleration	U32	rw	Profile Deceleration [1/s ²]
0x6086	0x00	Motion Profile Type	S16	rw	Bewegungsprofiltyp: 0: Lineares Profil 1: Sin ² Geschwindigkeit
0x60E0	0x00	Positive Torque Limit Value	U16	rw	Betrag des oberen Begrenzungswerts [in bezogener Darstellung]
0x60E1	0x00	Negative Torque Limit Value	U16	rw	Betrag des unteren Begrenzungswerts [in bezogener Darstellung]

Diese Parameter sind doppelt abgelegt. Im Betrieb kann schnell zwischen diesen unterschiedlichen Voreinstellungen gewechselt werden.

Einen Anwendungssatz anlegen

- ▶ Save Application Parameters 1: Auf Subindex 04 des Objekts 0x1010 die Signatur "save" schreiben.
 - ↪ Aktuelle Daten sind als Anwendungsparametersatz 1 gespeichert.
- ▶ Save Application Parameters 2: Auf Subindex 05 des Objekts 0x1010 die Signatur "save" schreiben.
 - ↪ Aktuelle Daten sind als Datensatz Anwendungsparametersatz 2 gespeichert.

Einen Anwendungssatz aktivieren

- ▶ Reload Application Parameters 1: Auf Subindex 05 des Objekts 0x1011 die Signatur "load" schreiben.
 - ↪ Aktuelle Daten aus dem Anwendungsparametersatz 1 werden direkt aktiviert.
- ▶ Reload Application Parameters 2: Auf Subindex 06 des Objekts 0x1011 die Signatur "load" schreiben.
 - ↪ Aktuelle Daten aus dem Anwendungsparametersatz 2 werden direkt aktiviert.

Parameterbeschreibung

4 Parameterbeschreibung

4.1 Kommunikationsobjekte nach CiA 301

Device Type

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1000	0x00	Device Type	U32	ro	0x00420192	Angabe des Gerätetyps

Enthält Informationen zum Gerätetyp, aufgeteilt in zwei 16-Bit-Feldern:

- Byte MSB (Most Significant Byte): Additional Information = 0x42 (Servo drive, type specific PDO mapping)
- Byte LSB (Least Significant Byte): Device Profile Number = 0x192 (402d)

Error Register

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1001	0x00	Error Register	U8	ro	ja	Fehlerregister

Das Error Register beinhaltet bitcodiert die zuletzt aufgetretenen Fehlerarten.

Dieser Parameter kann in ein PDO gemappt werden.

Predefined Error Field (Fehlerspeicher)

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1003	0x00	Number of Errors	U8	rw	–	Anzahl gespeicherter Fehler
	0x01– 0x08	Standard Error Field	U32	ro	–	Zuletzt aufgetretene Fehlercodes

Der Fehlerspeicher enthält die Codierung der zuletzt aufgetretenen Fehler.

- Byte MSB: Error Register
- Byte LSB: Error Code

Die Bedeutung der Fehlercodes ist in Kap. 3.6, S. 20 beschrieben.

Durch Schreiben einer 0 auf Subindex 0 wird der Fehlerspeicher gelöscht.

Manufacturer Device Name

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1008	0x00	Manufacturer Device Name	Vis-String	const	–	Gerätename

Manufacturer Hardware Version

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1009	0x00	Manufacturer Hardware Version	Vis-String	const	–	Hardwareversion

Manufacturer Software Version

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x100A	0x00	Manufacturer Software Version	Vis-String	const	–	Softwareversion

Parameterbeschreibung

Store Parameters

Tab. 10: Parameter speichern

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1010	0x00	Number of Entries	U8	ro	9	Anzahl Objekteinträge
	0x01	Save All Parameters	U32	rw	1	Speichert alle Parameter
	0x02	Save Comm Parameters	U32	rw	1	Kommunikationsparameter speichern (Objektverzeichnis-Einträge 0x0000 bis 0x1FFF)
	0x03	Save App Parameters	U32	rw	1	Anwendungsparameter speichern (Objektverzeichnis-Einträge 0x2000 bis 0x6FFF)
	0x04	Save App Parameters 1	U32	rw	1	Anwendungsparameter für den direkten Wechsel (Satz 1) speichern
	0x05	Save App Parameters 2	U32	rw	1	Anwendungsparameter für den direkten Wechsel (Satz 2) speichern

Das Objekt Store Parameters speichert Konfigurationsparameter in den Flash-Speicher. Ein Lesezugriff liefert Informationen über die Speichermöglichkeiten. Das Schreiben der Signatur "save" auf den entsprechenden Subindex leitet den Speichervorgang ein.

Tab. 11: Signatur "save"

Signature	ISO 8 859 ("ASCII")	hex
MSB	e	65h
	v	76h
	a	61h
LSB	s	73h



HINWEIS!

Der Flash-Speicher ist für 10 000 Schreibzyklen ausgelegt. Wird dieser Befehl mehr als 10 000 mal ausgeführt, ist die Funktion des Flash-Speichers nicht mehr gewährleistet.

- ▶ Häufiges Speichern vermeiden.
- ▶ Nach 10 000 Speicherzyklen Gerät wechseln.

Restore Default Parameters

Tab. 12: Wiederherstellen von Parametern

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1011	0x00	Number of Entries	U8	ro	6	Anzahl Objekteinträge
	0x01	Restore all Default Parameters	U32	rw	1	Alle Werkseinstellungen wiederherstellen
	0x02	Restore Comm Default Parameters	U32	rw	1	Werkseinstellungen für Kommunikations-Parameter (0x0000 bis 0x1FFF) wiederherstellen
	0x03	Restore App Default Parameters	U32	rw	1	Werkseinstellungen für Anwendungs-Parameter (ab 0x2000) wiederherstellen
	0x04	Reload User Parameters	U32	rw	1	Vom Benutzer zuletzt gespeicherte Anwendungsparameter (ab 0x2000) wiederherstellen

Parameterbeschreibung

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
	0x05	Reload Application Parameters 1	U32	rw	1	Anwendungsparametersatz 1 für den direkten Wechsel
	0x06	Reload Application Parameters 2	U32	rw	1	Anwendungsparametersatz 2 für den direkten Wechsel

Das Objekt Restore Default Parameters lädt Standardkonfigurationsparameter. Die Standardkonfigurationsparameter sind entweder der Auslieferungszustand oder der letzte gespeicherte Zustand. Ein Lesezugriff liefert Informationen über die Restoremöglichkeit. Das Schreiben der Signatur "load" auf den entsprechenden Subindex leitet den Restorevorgang ein:

Tab. 13: Signatur "load"

Signature	ISO 8859 ("ASCII")	hex
MSB	d	64h
	a	61h
	o	6Fh
LSB	l	6Ch



Der Auslieferungszustand darf nur bei abgeschalteter Endstufe geladen werden.



Um die durch einen **Restore Factory** wiederhergestellten Parameter zu aktivieren, muss der Antrieb aus- und wieder eingeschaltet werden.

Identity Object

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1018	0x00	Number of Entries	U8	ro	4	Anzahl Objekteinträge
	0x01	Vendor ID	U32	ro	327	Herstellernummer (FAULHABER: 327)
	0x02	Product Code	U32	ro	48	Produktkennnummer
	0x03	Revision Number	U32	ro	–	Versionsnummer
	0x04	Serial Number	U32	ro	–	Seriennummer

Receive PDO1 Mapping

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1600	0x00	Number of Mapped Objects	U8	ro	1	Anzahl gemappter Objekte
	0x01	RxPDO1 Mapping Entry 1	U32	rw	0x60400010	Verweis auf 16-Bit Controlword (0x6040)
	0x02	RxPDO1 Mapping Entry 2	U32	rw	0	
	0x03	RxPDO1 Mapping Entry 3	U32	rw	0	
	0x04	RxPDO1 Mapping Entry 4	U32	rw	0	

Parameterbeschreibung

Receive PDO2 Mapping

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1601	0x00	Number of Mapped Objects	U8	ro	2	Anzahl gemappter Objekte
	0x01	RxPDO2 Mapping Entry 1	U32	rw	0x60400010	Verweis auf 16-Bit Controlword (0x6040)
	0x02	RxPDO2 Mapping Entry 2	U32	rw	0x607A0020	Verweis auf 32-Bit Target Position (0x607A)
	0x03	RxPDO2 Mapping Entry 3	U32	rw	0	
	0x04	RxPDO2 Mapping Entry 4	U32	rw	0	

Receive PDO3 Mapping

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1602	0x00	Number of Mapped Objects	U8	ro	2	Anzahl gemappter Objekte
	0x01	RxPDO3 Mapping Entry 1	U32	rw	0x60400010	Verweis auf 16-Bit Controlword (0x6040)
	0x02	RxPDO3 Mapping Entry 2	U32	rw	0x60FF0020	Verweis auf 32-Bit Target Velocity (0x60FF)
	0x03	RxPDO3 Mapping Entry 3	U32	rw	0	
	0x04	RxPDO3 Mapping Entry 4	U32	rw	0	

Receive PDO4 Mapping

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1603	0x00	Number of Mapped Objects	U8	ro	2	Anzahl gemappter Objekte
	0x01	RxPDO4 Mapping Entry 1	U32	rw	0x60400010	Verweis auf 16-Bit Controlword (0x6040)
	0x02	RxPDO4 Mapping Entry 2	U32	rw	0x60710010	Verweis auf 16-Bit Target Torque (0x6071)
	0x03	RxPDO4 Mapping Entry 3	U32	rw	0	
	0x04	RxPDO4 Mapping Entry 4	U32	rw	0	

Parameterbeschreibung

Transmit PDO1 Mapping

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1A00	0x00	Number of Mapped Objects	U8	rw	1	Anzahl gemappter Objekte
	0x01	TxPDO1 Mapping Entry 1	U32	rw	0x60410010	Verweis auf 16-Bit Statusword (0x6041)
	0x02	TxPDO1 Mapping Entry 2	U32	rw	0	
	0x03	TxPDO1 Mapping Entry 3	U32	rw	0	
	0x04	TxPDO1 Mapping Entry 4	U32	rw	0	

Transmit PDO2 Mapping

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1A01	0x00	Number of Mapped Objects	U8	rw	2	Anzahl gemappter Objekte
	0x01	TxPDO2 Mapping Entry 1	U32	rw	0x60410010	Verweis auf 16-Bit Statusword (0x6041)
	0x02	TxPDO2 Mapping Entry 2	U32	rw	0x60640020	Verweis auf 32-Bit Position Actual Value (0x6064)
	0x03	TxPDO2 Mapping Entry 3	U32	rw	0	
	0x04	TxPDO2 Mapping Entry 4	U32	rw	0	

Transmit PDO3 Mapping

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1A02	0x00	Number of Mapped Objects	U8	rw	2	Anzahl gemappter Objekte
	0x01	TxPDO3 Mapping Entry 1	U32	rw	0x60410010	Verweis auf 16-Bit Statusword (0x6041)
	0x02	TxPDO3 Mapping Entry 2	U32	rw	0x606C0020	Verweis auf 32-Bit Velocity Actual Value (0x606C)
	0x03	TxPDO3 Mapping Entry 3	U32	rw	0	
	0x04	TxPDO3 Mapping Entry 4	U32	rw	0	

Parameterbeschreibung

Transmit PDO4 Mapping

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1A03	0x00	Number of Mapped Objects	U8	rw	2	Anzahl gemappter Objekte
	0x01	TxPDO4 Mapping Entry 1	U32	rw	0x60410010	Verweis auf 32-Bit Position Actual Value (0x6064)
	0x02	TxPDO4 Mapping Entry 2	U32	rw	0x60770010	Verweis auf 16-Bit Torque Actual Value (0x6077)
	0x03	TxPDO4 Mapping Entry 3	U32	rw	0	
	0x04	TxPDO4 Mapping Entry 4	U32	rw	0	

SyncManager Communication Type

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1C00	0x00	Number of Objects	U8	ro	4	Anzahl der Objekte
	0x01	SM0 Communication Type	U8	ro	0	0: SyncManager not in use 1: mailbox receive (master to slave)
	0x02	SM1 Communication Type	U8	ro	0	2: mailbox send (slave to master) 3: process data output (master to slave)
	0x03	SM2 Communication Type	U8	ro	0	4: process data input (slave to master)
	0x04	SM3 Communication Type	U8	ro	0	

SyncManager 2 (RxPDO, Master zum Antrieb): PDO Mapping

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1C12	0x00	Number of Objects	U8	rw	4	Anzahl der Objekte
	0x01	SM2: 1st RxPDO Assignment	U16	rw	0x1600	Zuordnung des SyncManager-Kanals 2 zum Empfangs-PDO 1 Mögliche Werte: 0x1600...0x1603
	0x02	SM2: 2nd RxPDO Assignment	U16	rw	0x1601	Zuordnung des SyncManager-Kanals 2 zum Empfangs-PDO 2 Mögliche Werte: 0x1600...0x1603
	0x03	SM2: 3rd RxPDO Assignment	U16	rw	0x1602	Zuordnung des SyncManager-Kanals 2 zum Empfangs-PDO 3 Mögliche Werte: 0x1600...0x1603
	0x04	SM2: 4th RxPDO Assignment	U16	rw	0x1603	Zuordnung des SyncManager-Kanals 2 zum Empfangs-PDO 4 Mögliche Werte: 0x1600...0x1603

Parameterbeschreibung

SyncManager 3 (TxPDO, Master zum Antrieb): PDO Mapping

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1C13	0x00	Number of Objects	U8	rw	4	Anzahl der Objekte
	0x01	SM3: 1st TxPDO Assignment	U16	rw	0x1A00	Zuordnung des SyncManager-Kanals 3 zum Sende-PDO 1 Mögliche Werte: 0x1A00...0x1A03
	0x02	SM3: 2nd TxPDO Assignment	U16	rw	0x1A01	Zuordnung des SyncManager-Kanals 3 zum Sende-PDO 2 Mögliche Werte: 0x1A00...0x1A03
	0x03	SM3: 3rd TxPDO Assignment	U16	rw	0x1A02	Zuordnung des SyncManager-Kanals 3 zum Sende-PDO 3 Mögliche Werte: 0x1A00...0x1A03
	0x04	SM3: 4th TxPDO Assignment	U16	rw	0x1A03	Zuordnung des SyncManager-Kanals 3 zum Sende-PDO 4 Mögliche Werte: 0x1A00...0x1A03

SyncManager 2 (RxPDO, Master zum Antrieb): Parameter

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1C32	0x00	Number of Objects	U8	ro	12	Sync-Manager-Parameter für Eingangs-PDOs
	0x01	SM2: Synchronization Type	U16	ro	2	Synchronisationsart: <ul style="list-style-type: none"> ▪ 1: SM-Sync ▪ 2: DC-Sync
	0x02	SM2: Cycle Time	U32	rw	500000	Zykluszeit (Wert muss ein Vielfaches von 1000000 ns sein)
	0x04	SM2: Synchronization Types Supported	U16	ro	0	Unterstützte Synchronisationsarten
	0x05	SM2: Minimum Cycle Time	U32	ro	0	Minimale Zykluszeit (nur im DC-Sync-Modus)
	0x06	SM2: Calc and Copy Time	U32	ro	0	Zeit in ns, nach der das nächste SyncManager-Ereignis frühestens kommen darf (nur im DC-Sync-Modus)
	0x09	SM2: Delay Time	U32	ro	0	Hardware-Verzögerungszeit bis zur Ausgabe der Ausgänge (nur im DC-Sync-Modus)
	0x0B	SM2: SM-Event Missed Counter	U16	ro	0	Anzahl der ausgefallenen SyncManager-Ereignisse (nur im DC-Sync-Modus)
	0x0C	SM2: Cycle Time Too Short Counter	U16	ro	0	Störungszähler, der um 1 erhöht wird, wenn Prozess-Eingangsdaten nicht aktualisiert worden sind, bevor das nächste SM2-Ereignis eintritt

Parameterbeschreibung

SyncManager 3 (TxPDO, Antrieb zum Master): Parameter

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x1C33	0x00	Sync Manager 3(TxPDO): Parameter	U8	ro	12	SyncManager-Parameter für Sende-PDOs
	0x01	SM3: Synchronization Type	U16	ro	2	Synchronisationsart: <ul style="list-style-type: none"> 1: SM-Sync 2: DC-Sync
	0x02	SM3: Cycle Time	U32	ro	0	Kopie des Werts von 0x1C32.02
	0x04	SM3: Synchronization Types Supported	U16	ro	0	Unterstützte Synchronisationsarten
	0x05	SM3: Minimum Cycle Time	U32	ro	0	Minimale Zykluszeit (nur im DC-Sync-Modus)
	0x06	SM3: Calc and Copy Time	U32	ro	0	Zeit in ns zwischen dem Einlesen der Eingänge und der Verfügbarkeit der Eingänge für den Master (nur im DC-Sync-Modus)
	0x0B	SM3: SM-Event Missed Counter	U16	ro	0	Anzahl der ausgefallenen SyncManager-Ereignisse (nur im DC-Sync-Modus)
	0x0C	SM3: Cycle Time Too Short Counter	U16	ro	0	Störungszähler, der um 1 erhöht wird, wenn Prozess-Eingangsdaten nicht aktualisiert worden sind, bevor das nächste SM2-Ereignis eintritt

Parameterbeschreibung

4.2 Herstellerspezifische Objekte

FAULHABER Fehlerregister (0x2320)

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x2320	0x00	Fault Register	U16	ro	–	FAULHABER Fehlerregister

Error Mask (0x2321)

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x2321	0x00	Number of Entries	U8	ro	6	Anzahl Objekteinträge
	0x01	Emergency Mask	U16	rw	0xFFFF	Fehler, für die eine Fehlermeldung verschickt werden
	0x02	Fault Mask	U16	rw	0x0000	Fehler, für die die Zustandsmaschine des Antriebs in den Zustand <i>Fault Reaction Active</i> geht
	0x03	Error Out Mask	U16	rw	0x0000	Fehler, für die der Fehler-Ausgangspin gesetzt wird
	0x04	Disable Voltage Mask	U16	ro	0x4024	Fehler, die den Antrieb abschalten (nicht konfigurierbar)
	0x05	Disable Voltage User Mask	U16	rw	0x0000	Fehler, die den Antrieb abschalten (konfigurierbar)
	0x06	Quick Stop Mask	U16	rw	0x0000	Fehler, für die die Zustandsmaschine des Antriebs in den Zustand <i>Quick Stop Active</i> geht

Die Zustände der Antriebs-Zustandsmaschine sind in der Dokumentation der Antriebsfunktionen beschrieben.

Trace Configuration

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x2370	0x00	Number of Entries	U8	ro	10	Anzahl Objekteinträge
	0x01	Trigger Source	U32	wo	0	Triggerquelle
	0x02	Trigger Threshold	S32	rw	0	Triggerschwelle
	0x03	Trigger Delay Offset	S16	rw	0	Triggerverzögerung
	0x04	Trigger Mode	U16	rw	0	Triggermodus
	0x05	Buffer Length	U16	rw	100	Pufferlänge
	0x06	Sample Time	U8	rw	1	Abtastrate der Aufzeichnung 1: in jedem Abtastschritt
	0x07	Trace Source of Channel 1	U32	wo	0	Tracequelle am Kanal 1
	0x08	Trace Source of Channel 2	U32	wo	0	Tracequelle am Kanal 2
	0x09	Trace Source of Channel 3	U32	wo	0	Tracequelle am Kanal 3
	0x0A	Trace Source of Channel 4	U32	wo	0	Tracequelle am Kanal 4

Parameterbeschreibung

Trace Buffer

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x2371	0x00	Number of Entries	U8	ro	5	Anzahl Objekteinträge
	0x01	Trace State	U16	ro	0	Triggerstatus
	0x02	Trace Value of Channel 1	Vis-String	ro	–	Signalpuffer Kanal 1
	0x03	Trace Value of Channel 2	Vis-String	ro	–	Signalpuffer Kanal 2
	0x04	Trace Value of Channel 3	Vis-String	ro	–	Signalpuffer Kanal 3
	0x05	Trace Value of Channel 4	Vis-String	ro	–	Signalpuffer Kanal 4

RS232 Baudrate Index und Knotennummer

Index	Subindex	Name	Typ	Attr.	Standardwert	Bedeutung
0x2400	0x00	Number of Entries	U8	ro	8	Anzahl Objekteinträge
	0x02	RS232 Rate	U8	rw	3	Index der Baudrate
	0x03	Node ID	U8	rw	1	Knotennummer
	0x08	Explicit Device ID	U16	rw	0	Identifikation des Antriebs
	0x09	Measured Cycle Times	U16	ro	0	Nur im SM-Modus: Tatsächlich gemessene Zykluszeit des Prozessabbildes. Im DC-Modus führt die Abfrage zu einem Fehler und sollte deshalb auch nicht in ein PDO gemappt werden. Die Zykluszeit 1C32.02 darf nicht auf 0 stehen.

