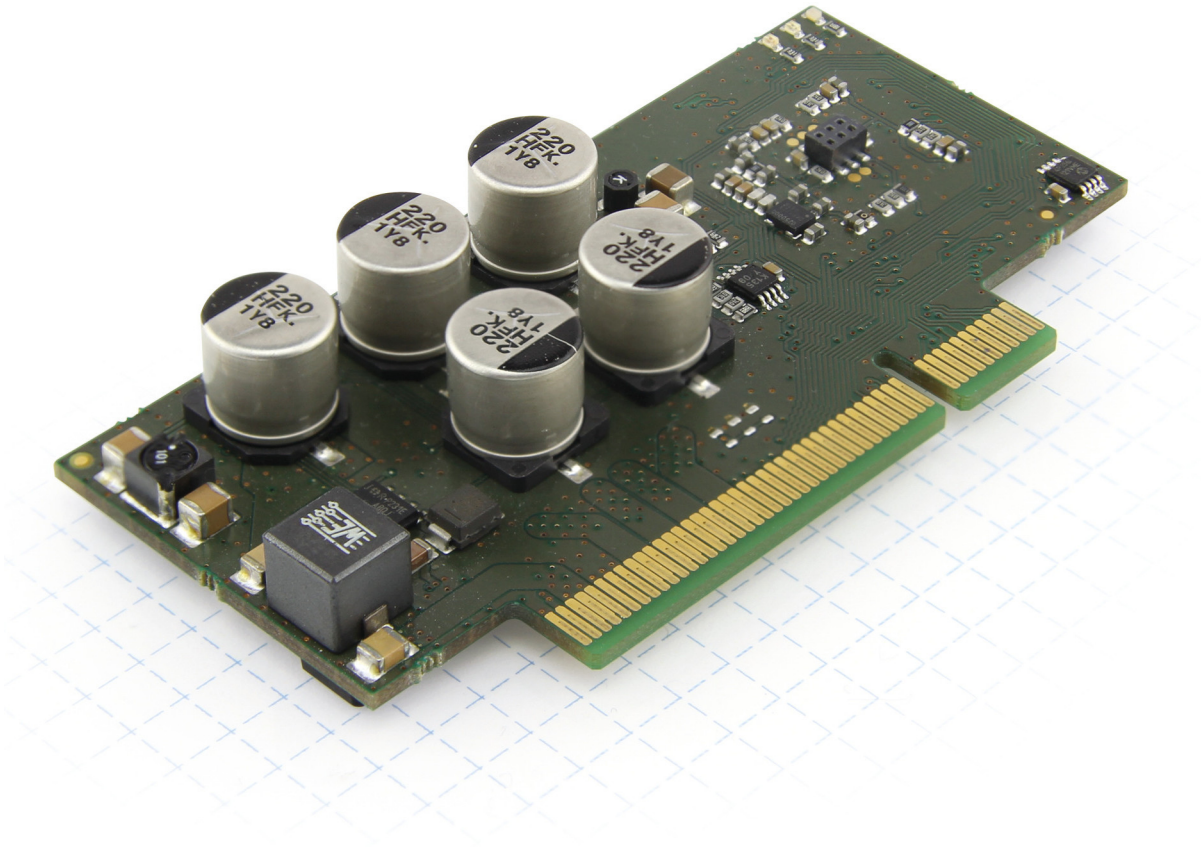
Technisches Handbuch NP5-40

Feldbus: SPI



Gültig ab Firmware-Version FIR-v1650
und ab Hardware-Version W003a

Technisches Handbuch Version: 1.0.1

Inhalt

1 Einleitung

Die NP5 ist eine Steuerung für BLDC- und Schrittmotoren im Steckmodulformat (Steckleiste im PCI-Format) zur Integration in Ihre eigenen Entwicklungen.

Hinweis

Die Steckleiste im PCI-Format ist nicht elektrisch kompatibel zu PCI-Express. Keinesfalls in PC-Mainboard einstecken.

Dieses Handbuch beschreibt die Integration der *NP5* in Ihr Motherboard und die Funktionen der Steuerung. Weiterhin wird gezeigt, wie Sie die Steuerung über die Kommunikationsschnittstelle ansprechen und programmieren können.

Weitere Informationen zum Gerät finden Sie auf der Nanotec-Homepage **www.nanotec.de**.

1.1 Versionshinweise

Version Handbuch	Datum	Änderungen	Version Firmware	Version Hardware
1.0.0	10/2017	erste Veröffentlichung	FIR-v1650-B472161	W003a
1.0.1	04/2018	Ergänzungen und Fehlerkorrekturen	FIR-v1650-B527540	W003a

1.2 Urheberrecht, Kennzeichnung und Kontakt

CE

Nanotec® Electronic GmbH & Co. KG

Kapellenstraße 6

D-85622 Feldkirchen bei München

Tel.: +49 (0)89-900 686-0

Fax: +49 (0)89-900 686-50

Internet: **www.nanotec.de**

1.3 Bestimmungsgemäßer Gebrauch

Die *NP5* dient der Steuerung von Schritt- und BLDC-Motoren und ist für den Einsatz unter den freigegebenen **Umgebungsbedingungen** konzipiert.

Die Steuerung muss über eine Steckleiste im PCI-Format und ein geeignetes Motherboard an Motoren angeschlossen werden. Die Systemgrenze der Steuerung endet an der PCI-Steckleiste.

Ein anderer Gebrauch gilt als nicht bestimmungsgemäß.

Hinweis

Änderungen oder Umbauten der Steuerung sind nicht zulässig.

1.4 Gewährleistung und Haftungsausschluss

Nanotec produziert Komponententeile, die ihren Einsatz in vielfältigen Industrieanwendungen finden. Die Auswahl und Anwendung von Nanotec-Produkten liegt im Verantwortungsbereich des Anlagenkonstrukteurs bzw. Endnutzers. Nanotec übernimmt keinerlei Verantwortung für die Integration der Produkte in das Endsystem.

Unter keinen Umständen darf ein Nanotec-Produkt als Sicherheitssteuerung in ein Produkt oder eine Konstruktion integriert werden. Alle Produkte, in denen ein von Nanotec hergestelltes Komponententeil enthalten ist, müssen bei der Übergabe an den Endnutzer entsprechende Warnhinweise und Anweisungen für eine sichere Verwendung und einen sicheren Betrieb aufweisen. Alle von Nanotec bereitgestellten Warnhinweise müssen unmittelbar an den Endnutzer weitergegeben werden.

Es gelten unsere Allgemeinen Geschäftsbedingungen: **de.nanotec.com/service/agb/**.

1.5 Fachkräfte

Nur Fachkräfte dürfen das Gerät installieren, programmieren und in Betrieb nehmen:

- Personen, die eine entsprechende Ausbildung und Erfahrung im Umgang mit Motoren und deren Steuerung haben.
- Personen, die den Inhalt dieses technischen Handbuchs kennen und verstehen.
- Personen, die die geltenden Vorschriften kennen.

1.6 Mitgeltende Vorschriften

Neben diesem technischen Handbuch sind folgende Vorschriften zu beachten:

- Unfallverhütungsvorschriften
- örtliche Vorschriften zur Arbeitssicherheit

1.7 EU-Richtlinien zur Produktsicherheit

Folgende EU-Richtlinien wurden beachtet:

- RoHS-Richtlinie (2011/65/EU, 2015/863/EU)

1.8 Verwendete Symbole

Alle Hinweise sind in einheitlicher Form. Der Grad der Gefährdung wird in die nachfolgenden Klassen eingeteilt.

VORSICHT

- Der Hinweis VORSICHT verweist auf eine eventuell gefährliche Situation.
- Die Missachtung des Hinweises führt **möglicherweise** zu mittelschweren Verletzungen.
- Beschriebt, wie Sie der Gefährdung entgehen können.

Hinweis

- Weist auf eine Fehlerquelle oder Verwechslungsgefahr hin.
- Die Missachtung des Hinweises führt **möglicherweise** zu Beschädigungen an diesem Gerät oder anderen Geräten.
- Beschreibt, wie Sie Geräteschäden vermeiden können.

Tipp

Zeigt einen Tipp zur Anwendung oder Aufgabe.

1.9 Hervorhebungen im Text

Im Dokument gelten folgende Konventionen:

Ein **fett** hervorgehobener Text markiert Querverweise und Hyperlinks:

- Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:
- Eine Liste verfügbarer Systemcalls findet sich im Kapitel **Systemcalls im NanoJ-Programm**.

Ein *kursiv* hervorgehobener Text markiert benannte Objekte:

- Lesen Sie das *Installationshandbuch*.
- Benutzen Sie die Software *Plug & Drive Studio*, um das Auto-Setup durchzuführen.
- Für Software: Im Tab *Operation* finden Sie die entsprechenden Informationen.
- Für Hardware: Benutzen Sie den *EIN/AUS*-Schalter, um das Gerät einzuschalten.

Ein Text in `courier` markiert einen Code-Abschnitt oder Programmierbefehl:

- Die Zeile mit dem Befehl `od_write(0x6040, 0x00, 5 );` ist wirkungslos.
- Die NMT-Nachricht baut sich wie folgt auf: `000 | 81 2A`

Ein Text in "Anführungszeichen" markiert Benutzereingaben:

- NanoJ-Programm starten durch Beschreiben von Objekt 2300$_h$, Bit 0 = "1".
- Wird in diesem Zustand bereits Haltemoment benötigt, muss in das 3212$_h$:01$_h$ der Wert "1" geschrieben werden.

1.10 Zahlenwerte

Zahlenwerte werden grundsätzlich in dezimaler Schreibweise angegeben. Sollte eine hexadezimale Notation verwendet werden, wird das mit einem tiefgestellten *h* am Ende der Zahl markiert.

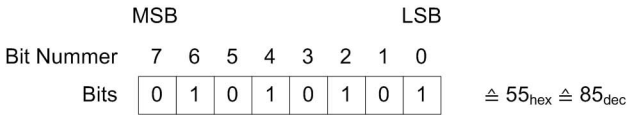
Die Objekte im Objektverzeichnis werden mit Index und Subindex folgendermaßen notiert: `<Index>:<Subindex>`

Sowohl der Index als auch der Subindex werden in hexadezimaler Schreibweise angegeben. Sollte kein Subindex notiert sein, gilt der Subindex 00$_h$.

Beispiel: Der Subindex 5 des Objekts 1003$_h$ wird adressiert mit `1003h:05h`, der Subindex 00 des Objekts 6040$_h$ mit `6040h`.

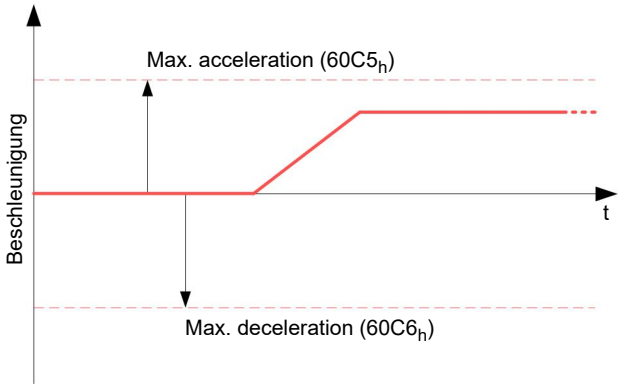
1.11 Bits

Einzelne Bits in einem Objekt beginnen bei der Nummerierung immer bei dem LSB (Bitnummer 0). Siehe nachfolgende Abbildung am Beispiel des Datentyps *UNSIGNED8*.



1.12 Zählrichtung (Pfeile)

In Abbildungen gilt die Zählrichtung immer in Richtung eines Pfeiles. Die in der nachfolgenden Abbildung beispielhaft dargestellten Objekte 60C5$_h$ und 60C6$_h$ werden beide positiv angegeben.



2 Sicherheits- und Warnhinweise

Hinweis

- Beschädigung der Steuerung.
- Ein Wechsel der Verdrahtung im Betrieb kann die Steuerung beschädigen.
- Ändern Sie die Verdrahtung nur im spannungsfreien Zustand und warten Sie nach dem Abschalten, bis sich die Kondensatoren entladen haben.

Hinweis

- Störung der Steuerung durch Erregerspannung des Motors.
- Während des Betriebs können Spannungsspitzen die Steuerung beschädigen.
- Verbauen Sie geeignete Schaltungen (z. B. Stützkondensator), die Spannungsspitzen abbauen.

Hinweis

- Ein Verpolungsschutz ist nicht gegeben.
- Bei Verpolung entsteht ein Kurzschluss zwischen Versorgungsspannung und GND (Masse) über die Leistungsdiode.
- Installieren Sie eine Leitungsschutzeinrichtung (Sicherung) in der Zuleitung.

Hinweis

- Das Gerät enthält Bauteile, die empfindlich gegen elektrostatische Entladung sind.
- Unsachgemäßer Umgang kann das Gerät beschädigen.
- Beachten Sie die Grundprinzipien des ESD-Schutzes beim Umgang mit dem Gerät.

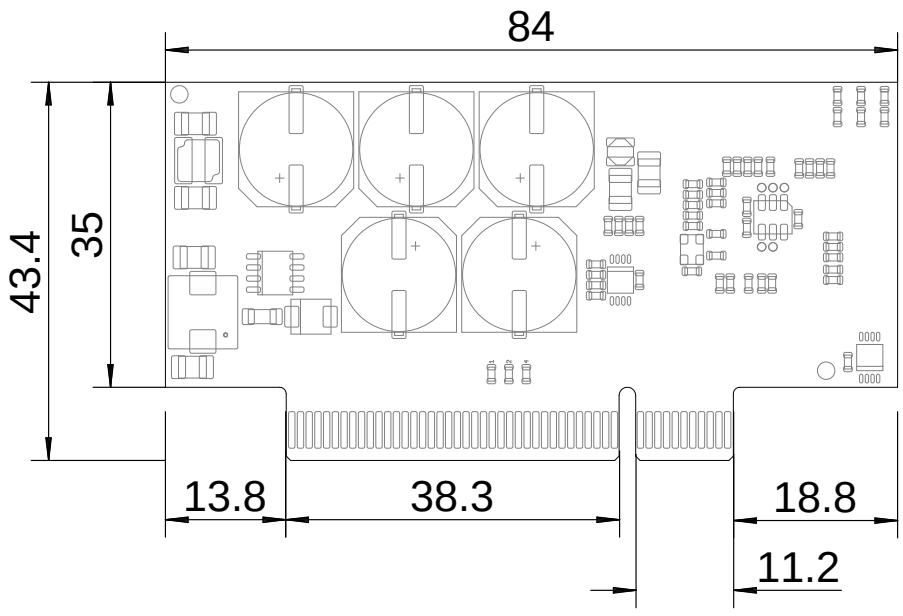
3 Technische Daten und Anschlussbelegung

3.1 Umgebungsbedingungen

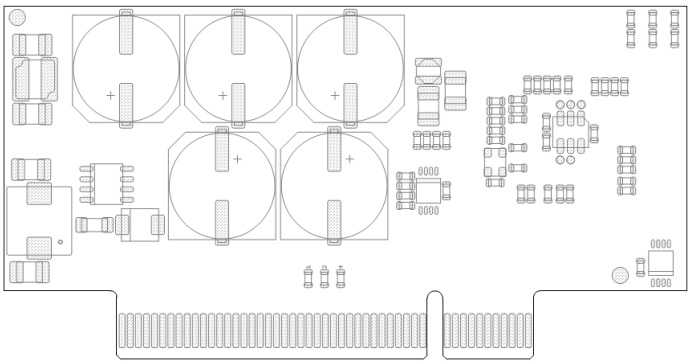
Umgebungsbedingung	Wert
Schutzklasse	kein IP-Schutz
Umgebungstemperatur (Betrieb)	-10 … +40°C
Luftfeuchtigkeit (nicht kondensierend)	0 … 95 %
Aufstellhöhe über *NN* (ohne Leistungsbeschränkung)	1500 m
Umgebungstemperatur (Lagerung)	-25 … +85°C

3.2 Maßzeichnungen

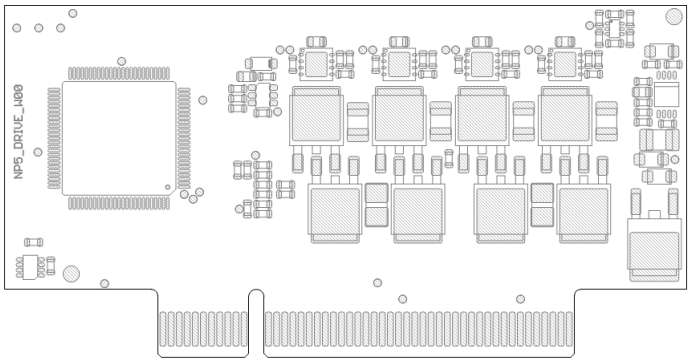
Die Maße sind in [mm].



Folgende Abbildungen zeigen das Platinenlayout.



Seite A



Seite B

3.3 Elektrische Eigenschaften und technische Daten

Eigenschaft	Beschreibung/Wert
Betriebsspannung	12 - 48 V DC ±4%
Nennstrom	6 A_{eff}
Spitzenstrom	10 A_{eff} (für 1 Sekunde)
Kommutierung	Schrittmotor *Open Loop*, Schrittmotor *Closed Loop* mit Encoder, BLDC sinuskommutiert über Hallsensor, BLDC sinuskommutiert über Encoder **Anmerkung:** Für Encoder und Hallsensor ist eine externe Beschaltung erforderlich!
Betriebsmodi	*Profile Position Mode*, *Profile Velocity Mode*, *Profile Torque Mode*, *Velocity Mode*, *Homing Mode*, *Interpolated Position Mode*, *Cyclic Sync Position Mode*, *Cyclic Sync Velocity Mode*, *Cyclic Synchronous Torque Mode*, *Takt-Richtung-Modus*
Sollwertvorgabe/ Programmierung	*Takt-Richtung*, *Analog*, *NanoJ-Programm*
Schnittstellen	2x SPI, 1x I^2C oder CANopen **Anmerkung:** Für CANopen ist eine externe Beschaltung erforderlich!
Encoder/Hall	2x Encoder und 1x Hallsensor

Eigenschaft	Beschreibung/Wert
	Anmerkung: Für Encoder und Hallsensor ist eine externe Beschaltung erforderlich!
I/O	6x General I/O , 2x Analogeingang, 1x Ausgang für die externe Bremse (Open-Drain), 1x Ausgang für die externe Ballast-Schaltung
Steckverbinder	PCI Express 8x, 1,0 mm RM, 2×49 Kontakte
Übertemperatur	Schutzschaltung bei Temperatur > 70°C
Verpolungsschutz	Verpolungsschutz durch Leistungsdiode (Kurzschluss zwischen +UB und GND, Sicherung in Zuleitung nötig)
Sicherungsgröße für Verpolungsschutz:	I_{max} (Steuerung) < I (Auslösestrom Sicherung) < I_{max} (Spannungsversorgung)
Stützkondensator	Nanotec empfiehlt pro Ampere Nennstrom am Motor eine Kapazität von ca. 1000 µF.

Hinweis

- Für die digitalen Eingänge liegt die Einschaltschwelle bei 1,8 V, die Ausschaltschwelle liegt bei 1,2 V.
- Für die digitalen Eingänge liegt die maximale Abtastfrequenz bei 1 MHz.
- Der Bereich der Analogeingänge ist 0 … 3,3 V.

Tipp

Falls der Sicherungswert (I Auslösestrom Sicherung) sehr nahe an der maximalen Stromaufnahme der Steuerung (I_{max} Steuerung) liegt, sollte eine Auslösecharakteristik *mittel/träge* eingesetzt werden.

3.4 Übertemperaturschutz

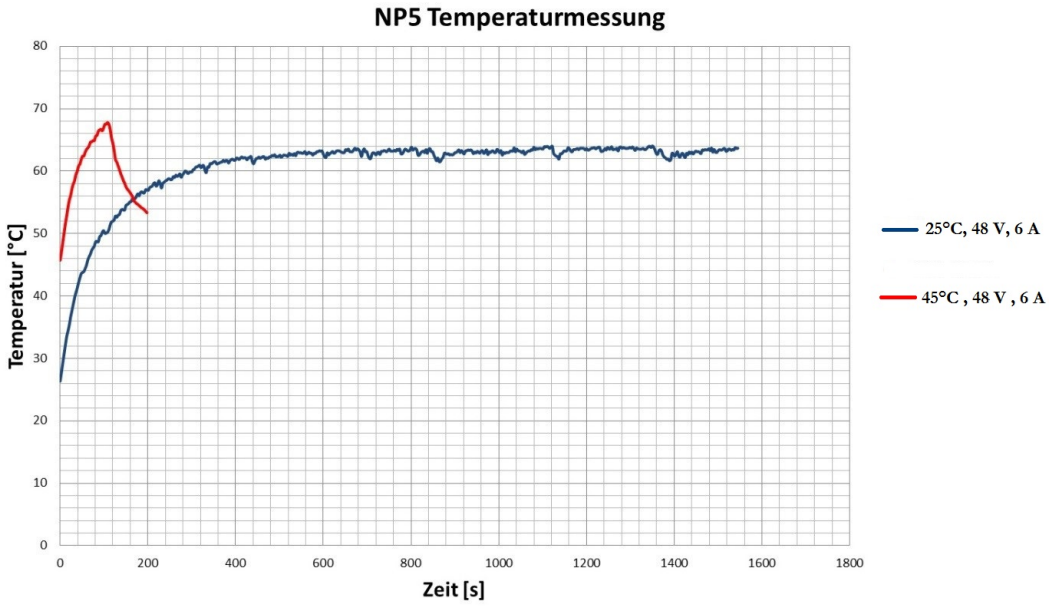
Ab einer Temperatur von ca. 70 °C auf der Leistungsplatine wird das Leistungsteil der Steuerung abgeschaltet und das Fehlerbit gesetzt (siehe Objekt **1001**$_h$ und **1003**$_h$). Nach Abkühlung und dem Bestätigen des Fehlers (siehe **Tabelle für das Contolword**, "Fault reset") funktioniert die Steuerung wieder normal.

Die folgenden Ergebnisse von Temperaturtests geben einen Hinweis auf das Temperaturverhalten dieser Steuerung.

Es wurden Temperaturtests unter folgenden Bedingungen durchgeführt:

- Betriebsspannung: 48 V DC
- Motorstrom: 6 A effektiv
- Operationsmodus: Drehzahlmodus Vollschritt, 30 U/min
- Umgebungstemperatur: 25 °C / 45 °C
- Aufstellhöhe: 500 m über NN
- keine externe Kühlung im Klimaschrank, z.B. über Lüfter

Die folgende Grafik zeigt die Ergebnisse der Temperaturtests:



Zusammenfassung:

Bei 25°C (+48V, 6A effektiv, Drehzahlmodus 30 U/min) ist die Steuerung länger als 2 Stunden in Betrieb gewesen ohne Abschaltung. Die Temperatur war stabil bei ca. 62°C.

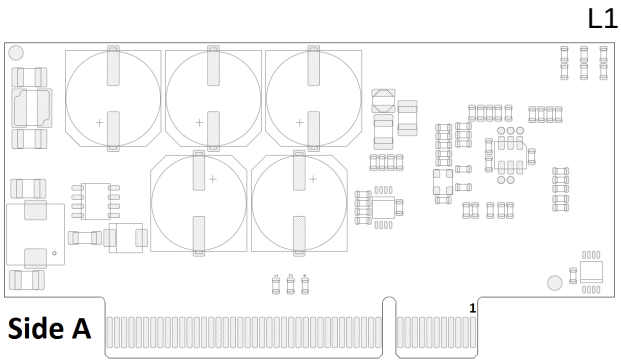
Bei 45°C (+48V, 6A effektiv, Drehzahlmodus 30 U/min) hat der Temperaturschutz die Steuerung in weniger als 2 Minuten abgeschaltet.

Hinweis

Da das genaue Temperaturverhalten jedoch außer vom Motor auch wesentlich von der Anflanschung und dem dortigen Wärmeübergang sowie von der Konvektion in der Maschine abhängt, empfehlen wir bei Applikationen, die hinsichtlich Stromhöhe und Umgebungstemperatur problematisch sind, immer einen Dauertest in der realen Umgebung.

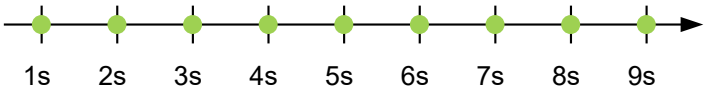
3.5 LED-Signalisierung

3.5.1 Betriebs-LED



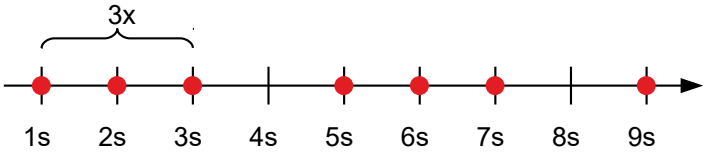
Normaler Betrieb

Im normalen Betrieb blinkt die grüne Betriebs-LED einmal in der Sekunde sehr kurz auf.



Fehlerfall

Liegt ein Fehler vor, schaltet die LED auf Rot um und signalisiert eine Fehlernummer. In der folgenden Darstellung wird der Fehler mit der Nummer 3 signalisiert.



Folgende Tabelle zeigt die Bedeutung der Fehlernummern.

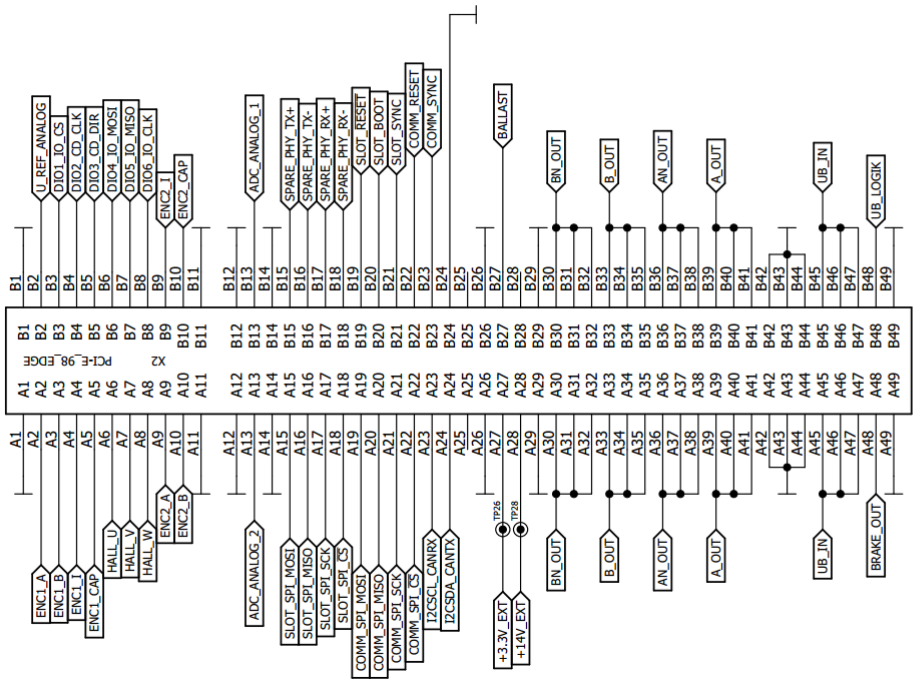
Blinktakt	Fehler
1	Allgemein
2	Spannung
3	Temperatur
4	Überstrom
5	Regler
6	Watchdog-Reset

Hinweis

Für jeden aufgetretenen Fehler wird im Objekt **1003_h** ein genauerer Fehlercode hinterlegt.

3.6 Anschlussbelegung

Anschlussbelegung der PCI-Steckleiste



Hinweis

- Für die digitalen Eingänge 1 bis 6 liegt die Einschaltschwelle bei 1,8 V, die Ausschaltschwelle liegt bei 1,2 V DC. Die maximale Abtastfrequenz liegt bei 1 MHz. Wenn die I/O PINs als Ausgang verwendet werden (siehe **Ein- und Ausgangsbelegung festlegen**), ist die Strombelastbarkeit ca. 10 mA bei 3,3 V DC.
- Der Bereich der Analogeingänge ist 0 … 3,3 V DC.
- Das Encoder-Signal ist single-ended, die Einschaltschwelle liegt bei 1,8 V, die Ausschaltschwelle bei 1,2 V DC. Die maximale Abtastfrequenz ist 1 MHz.
- Die Stromaufnahme der Logik-Versorgung UB_LOGIK beträgt ca. 30 mA bei 24 V DC.

PCI-Pin-Belegung:

Pin	Name	Beschreibung/Funktion
A1	GND	
A2	ENC1_A	Encoder 1, A
A3	ENC1_B	Encoder 1, B
A4	ENC1_I	Encoder 1, Index
A5	ENC1_CAP	nicht benutzt
A6	HALL_U (H1)	Hallsensor 1 (U)
A7	HALL_V (H2)	Hallsensor 2 (V)
A8	HALL_W (H3)	Hallsensor 3 (W)
A9	ENC2_A	Encoder 2, A
A10	ENC2_B	Encoder 2, B

Pin	Name	Beschreibung/Funktion
A11	GND	
A12	GND	
A13	ADC_ANALOG_2	Analog Eingang 2: 0 … 3,3 V
A14	GND	
A15	SLOT_SPI_MOSI	*SLOT_SPI*, siehe **Anschluss SPI**
A16	SLOT_SPI_MISO	*SLOT_SPI* , siehe **Anschluss SPI**
A17	SLOT_SPI_SCK	*SLOT_SPI* , siehe **Anschluss SPI**
A18	SLOT_SPI_$\overline{CS}$	*SLOT_SPI_$\overline{CS}$*, siehe **Anschluss SPI**
A19	COMM_SPI_MOSI	*COMM_SPI* , siehe **Anschluss SPI**
A20	COMM_SPI_MISO	*COMM_SPI*, siehe **Anschluss SPI**
A21	COMM_SPI_SCK	*COMM_SPI* , siehe **Anschluss SPI**
A22	COMM_SPI_$\overline{CS}$	*COMM_SPI* , siehe **Anschluss SPI**
A23	I2CSCL_CANRX	
A24	I2CSDA_CANTX	
A25	n.c.	reserviert
A26	GND	
A27	+3.3V_EXT	nicht benutzt
A28	+14V_EXT	nicht benutzt
A29	GND	
A30	BN_OUT	B\ (Schrittmotor)
A31		
A32		
A33	B_OUT	B\(Schrittmotor) oder W (BLDC)
A34		
A35		
A36	AN_OUT	A\ (Schrittmotor) oder V (BLDC)
A37		
A38		
A39	A_OUT	A (Schrittmotor) oder U (BLDC)
A40		
A41		
A42	GND	
A43		
A44		
A45	UB_IN	12 … 48 V DC ±4%
A46		
A47		

Pin	Name	Beschreibung/Funktion
A48	BRAKE_OUT	Ansteuerung der externen Bremse, Open-Drain Output, max. 1 A
A49	GND	
B1	GND	
B2	U_REF_ANALOG	3,3 V DC, Referenzspannung für die Analogeingänge
B3	DIO1_IO_CS	General I/O
B4	DIO2_CD_CLK	General I/O (Takt-Eingang in Takt-Richtung-Modus)
B5	DIO3_CD_DIR	General I/O (Richtungseingang in Takt-Richtung-Modus)
B6	DIO4_IO_MOSI	General I/O
B7	DIO5_IO_MISO	General I/O
B8	DIO6_IO_CLK	General I/O
B9	ENC2_I	Encoder 2, Index
B10	ENC2_CAP	nicht benutzt
B11	GND	
B12	GND	
B13	ADC_ANALOG_1	Analog Eingang 1: 0 … 3,3 V
B14	GND	
B15	SPARE_PHY_TX+	reserviert
B16	SPARE_PHY_TX-	reserviert
B17	SPARE_PHY_RX+	reserviert
B18	SPARE_PHY_RX-	reserviert
B19	SLOT_$\overline{\text{RESET}}$	Systemfunktion, reserviert
B20	SLOT_BOOT	Systemfunktion, reserviert
B21	SLOT_SYNC	Systemfunktion, reserviert
B22	COMM_RESET	
B23	COMM_SYNC	
B24	GND	
B25	n.c.	reserviert
B26	GND	
B27	BALLAST	zur Ansteuerung der externen Ballast-Schaltung
B28	n.c.	reserviert
B29	GND	
B30	BN_OUT	B\ (Schrittmotor)
B31		
B32		
B33	B_OUT	B (Schrittmotor) oder W (BLDC)
B34		
B35		
B36	AN_OUT	A\ (Schrittmotor) oder V (BLDC)
B37		
B38		

Pin	Name	Beschreibung/Funktion
B39	A_OUT	A (Schrittmotor) oder U (BLDC)
B40		
B41		
B42	GND	
B43		
B44		
B45	UB_IN	12 … 48 V DC ±4%
B46		
B47		
B48	UB_LOGIK	Externe Logikversorgung, 24 V DC
B49	GND	

4 Hardware-Installation

Hinweis

Beachten Sie, dass alle Bauteile spannungsfrei sind.

Hinweis

- Das Gerät enthält Bauteile, die empfindlich gegen elektrostatische Entladung sind.
- Unsachgemäßer Umgang kann das Gerät beschädigen.
- Beachten Sie die Grundprinzipien des ESD-Schutzes beim Umgang mit dem Gerät.

4.1 Anschließen der Steuerung

Zum einfachen Anschluss empfiehlt Nanotec das *Discovery Board DK-NP5-48* . Falls Sie die Steuerung über dieses *Discovery Board* betreiben, lesen Sie das Kapitel **Anschließen der Steuerung NP5 über das Discovery Board**.

4.1.1 Integrieren der NP5

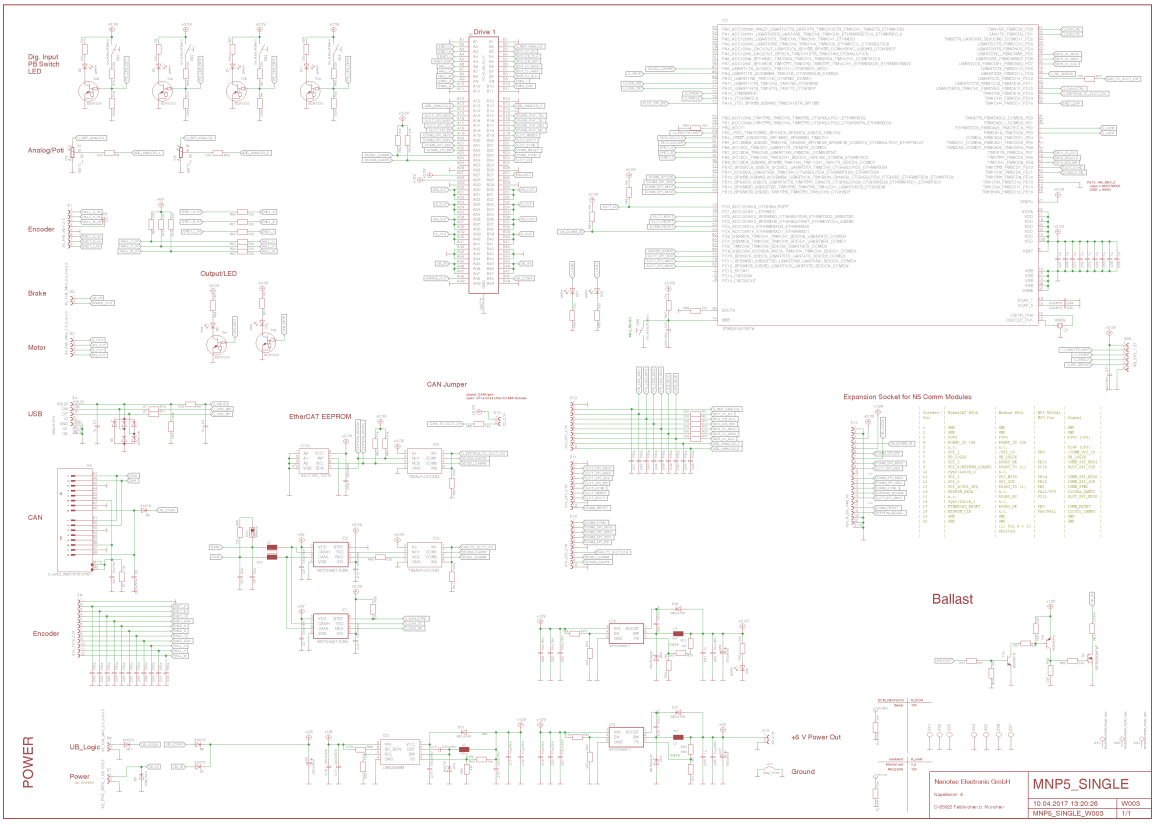
Hinweis

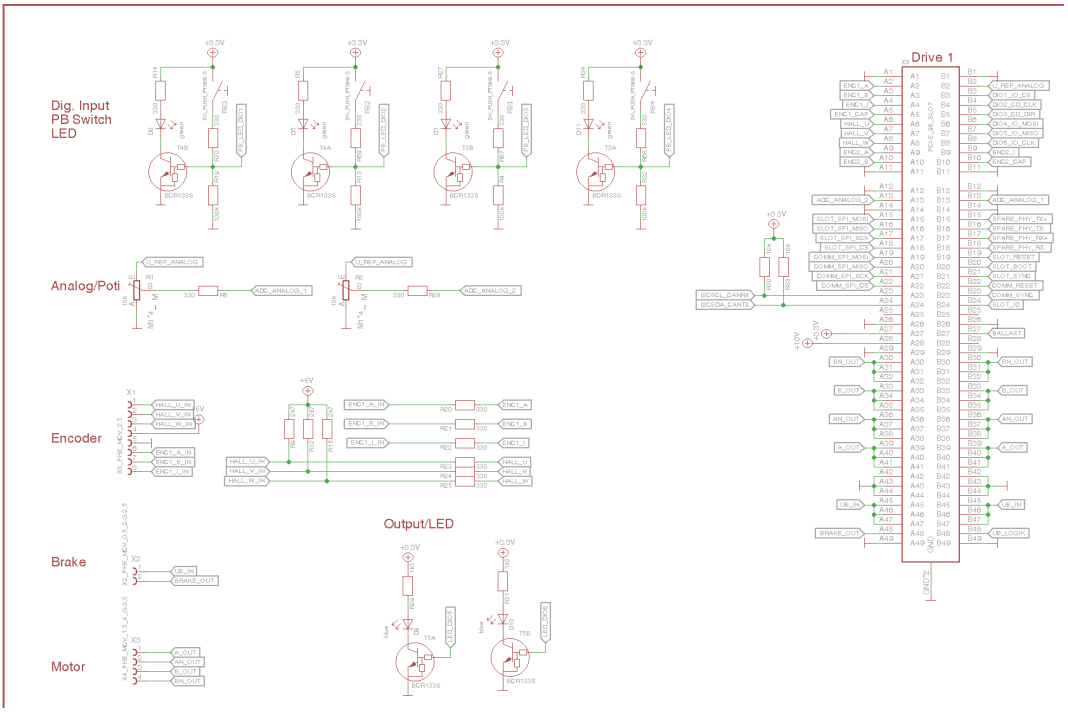
- EMV: Stromführende Leitungen – insbesondere um Versorgungs- und Motorenleitungen – erzeugen elektromagnetische Wechselfelder.
- Diese können den Motor und andere Geräte stören. Nanotec empfiehlt folgende Maßnahmen:
- Geschirmte Leitungen verwenden und den Leitungsschirm beidseitig auf kurzem Weg erden.
- Kabel mit paarweise verdrillten Adern verwenden.
- Stromversorgungs- und Motorleitungen so kurz wie möglich halten.
- Motorgehäuse großflächig auf kurzem Weg erden.
- Versorgungs-, Motor- und Steuerleitungen räumlich getrennt verlegen.

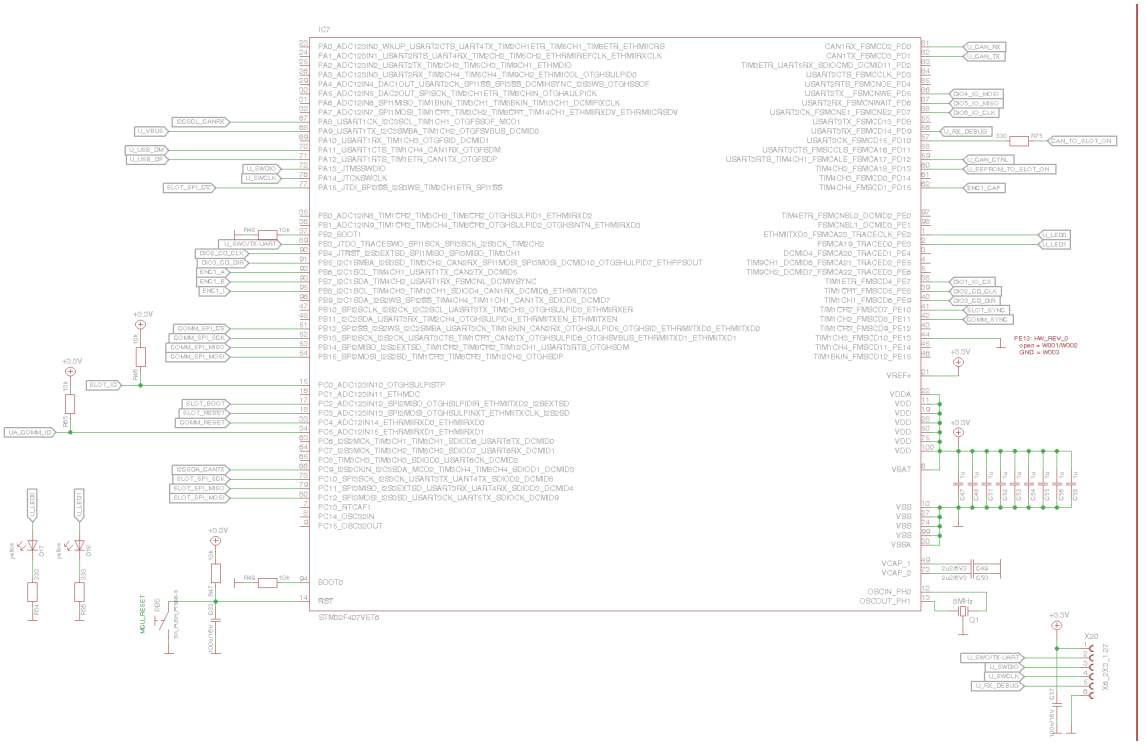
In den nachfolgenden Abbildungen sehen Sie den Schaltplan des *Discovery Board NP5*, der als Referenz für die Entwicklung Ihres eigenen Motherboards dienen kann. Die Pin-Belegung der PCI-Steckleiste finden Sie im Kapitel **Anschlussbelegung**.

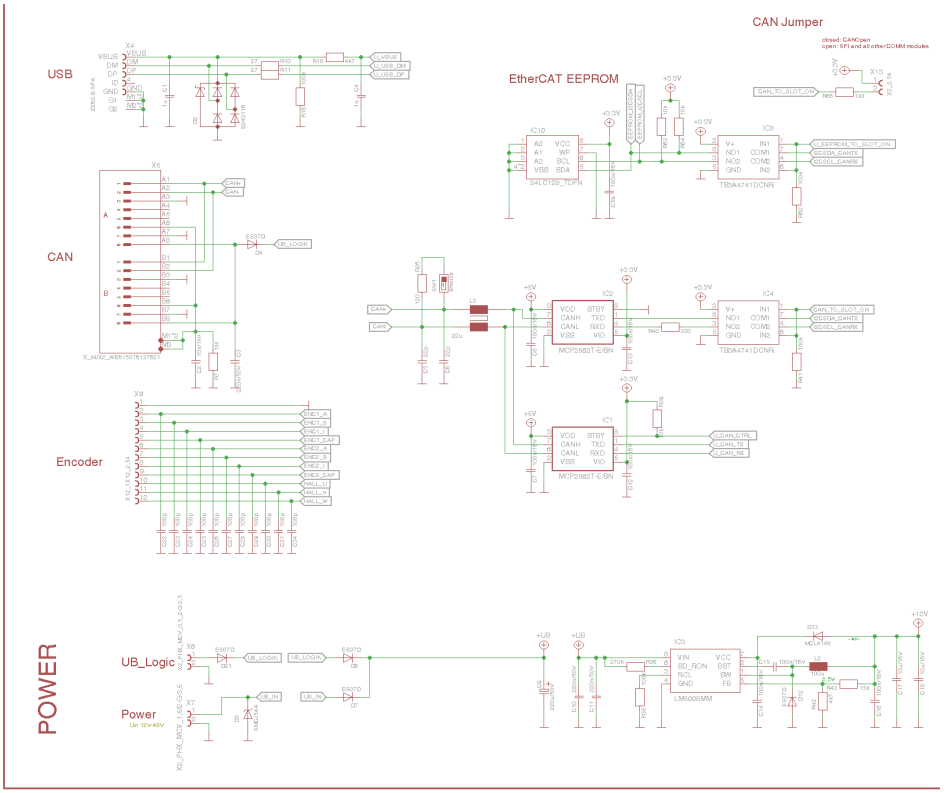
1. Bereiten Sie Ihr Motherboard vor.

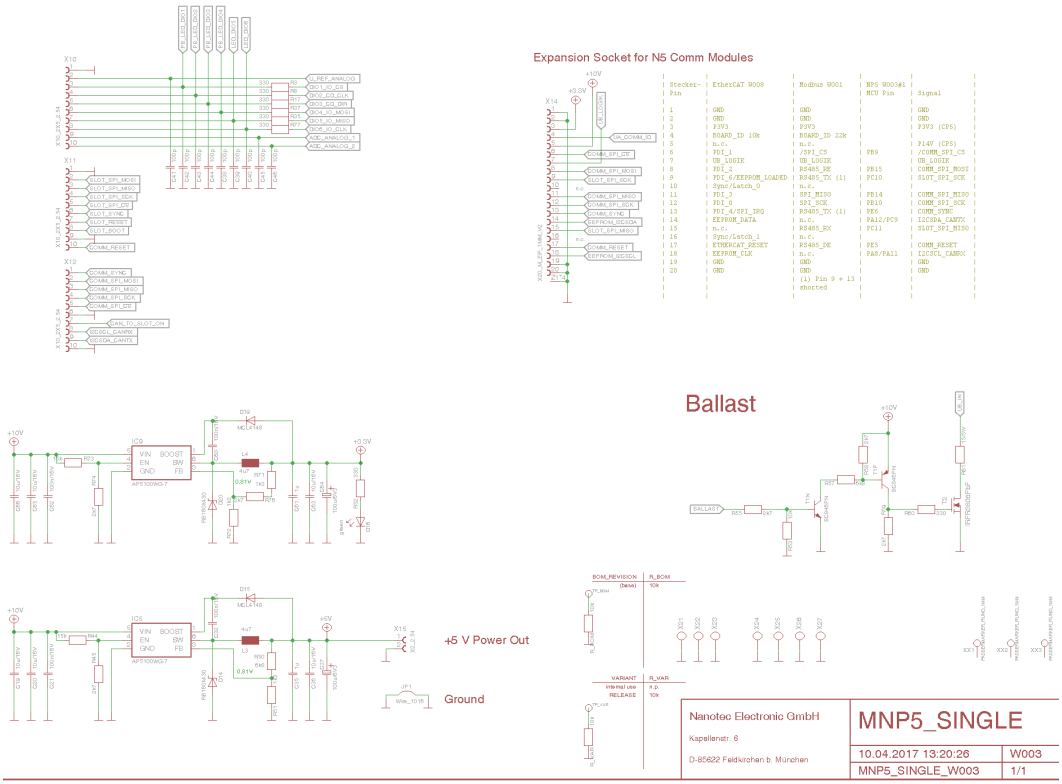
   Die minimale Beschaltung variiert je nach Motortyp und vorhandener Rückführung (Schritt- oder BLDC-Motor, Hallsensoren/Encoder). Zur Inbetriebnahme ist der Anschluss der Spannungsversorgung (*POWER*), des Motors und der SPI-Leitungen (siehe auch **Anschluss SPI**) ausreichend.


Dig. Input
PB Switch
LED
Analog/Poti
Encoder
Brake
Motor
Output/LED
Drive 1
CAN Jumper
USB
EtherCAT EEPROM
Expansion Socket for NS Comm Modules
CAN
Encoder
Ballast
POWER
UB_Logic
Power
+5 V Power Out
Ground
Nanotec Electronic GmbH
MNP5_SINGLE
10.04.2017 13:20:26
W003
MNP5_SINGLE_W003
1/1


Dig. Input
PB Switch
LED
Analog/Poti
Encoder
Brake
Motor
Output/LED
Drive 1




CAN Jumper
USB
EtherCAT EEPROM
CAN
Encoder
POWER
UB_Logic
Power



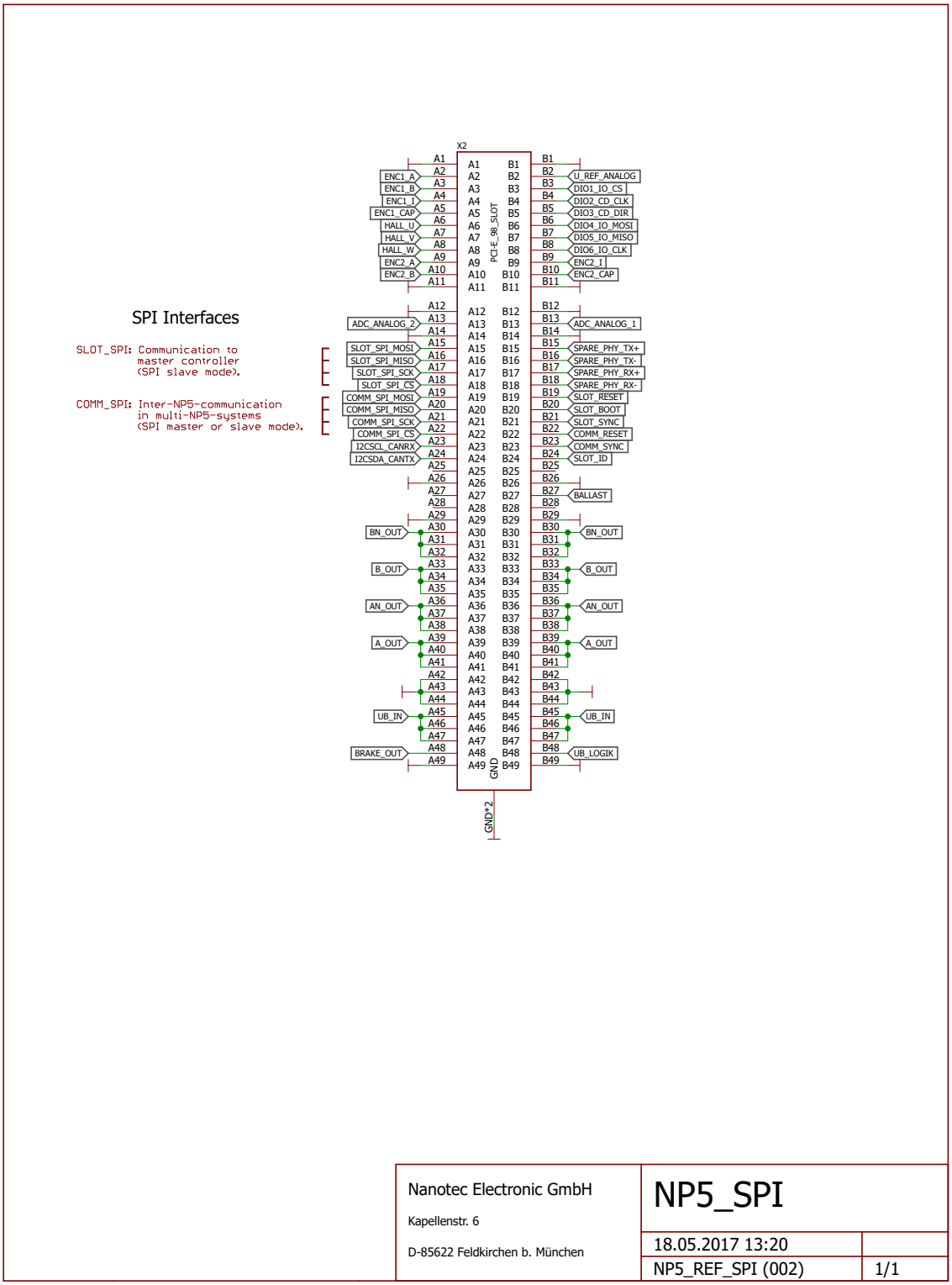
2. Stecken Sie die *NP5* in die *PCI*-Steckverbindung.

4.1.2 Anschluss SPI

Die folgende Abbildung zeigt eine Referenzschaltung für den Anschluss der NP5 SPI

Hinweis

Für die Standardbelegung der Anschlüsse, siehe **Anschlussbelegung**.



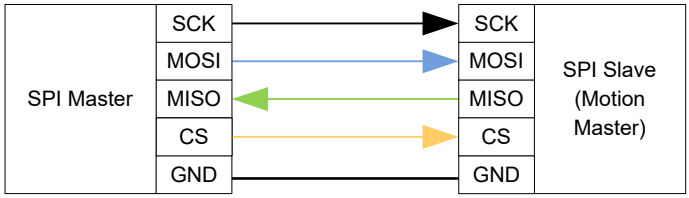
PCI spezielle Pin-Belegung für SPI :

Pin	Name	Beschreibung/Funktion
A15	SLOT_SPI_MOSI	*SLOT_SPI*
A16	SLOT_SPI_MISO	*SLOT_SPI*

Pin	Name	Beschreibung/Funktion
A17	SLOT_SPI_SCK	*SLOT_SPI*
A18	SLOT_SPI_$\overline{CS}$	*SLOT_SPI*
A19	COMM_SPI_MOSI	*COMM_SPI*
A20	COMM_SPI_MISO	*COMM_SPI*
A21	COMM_SPI_SCK	*COMM_SPI*
A22	COMM_SPI_$\overline{CS}$	*COMM_SPI*

Bus-Topologie

Der SPI Bus verwendet die Leitungen *SCK* (source clock), *MOSI* (master out, slave in), *MISO* (master in, slave out) und *CS* (chip select).



4.1.3 Anschließen der Steuerung NP5 über das *Discovery Board*

Das *Discovery Board NP5* hilft Ihnen bei Tests und bei der Evaluierung der *NP5* Steuerung.

Die notwendigen Stecker für das Board werden bereits montiert geliefert.

Der **Jumper X13** muss nur gesetzt sein, wenn CANopen (*NP5-08*) verwendet wird, sonst müssen Sie ihn entfernen.

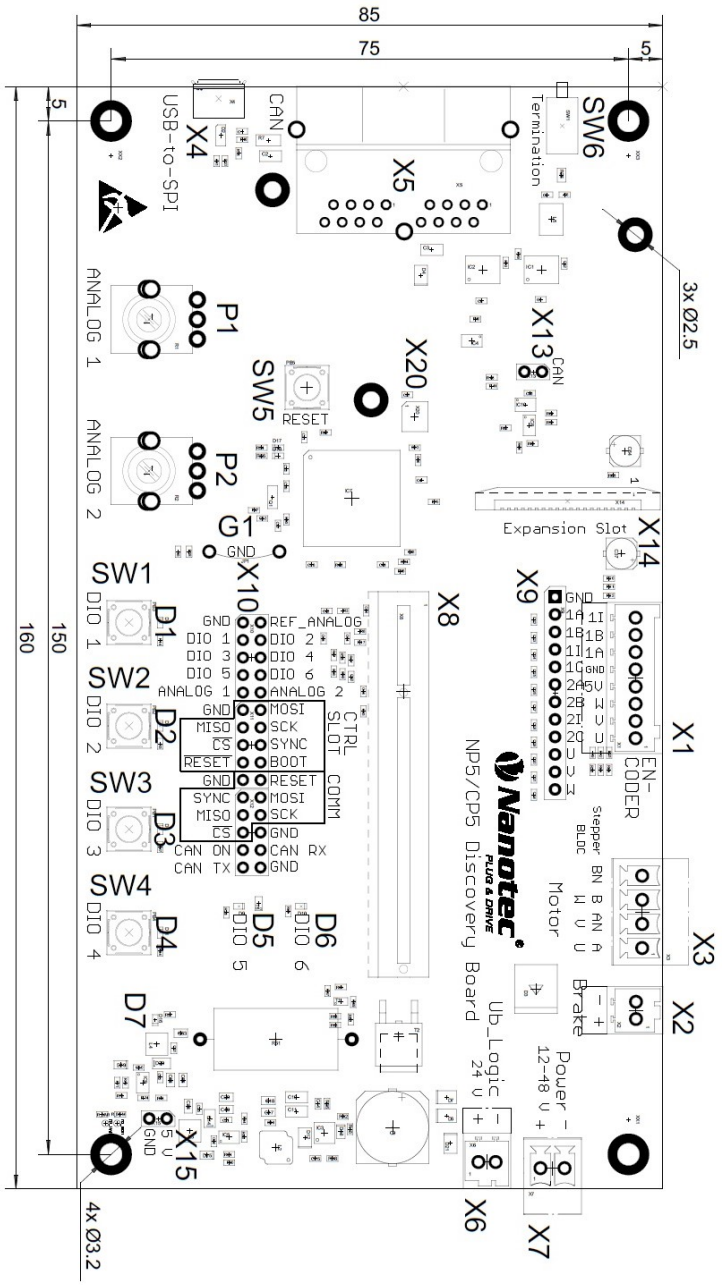
Technische Daten - *Discovery Board NP5*

Eigenschaft	Beschreibung/Wert
Betriebsspannung +UB:	12 … 48 V DC ±5%
Logik-Spannung +UB_Logic:	24 V DC ±5%
Stromaufnahme +UB:	max. 100 mA (ohne angeschlossene NP5)
Stromaufnahme +UB_Logic:	max. 100 mA (ohne angeschlossene NP5)
Kommunikationsschnittstelle:	SPI, CANopen
Analog-Referenzspannung:	3,3 V DC ±5%, max. 10 mA
Digital-Eingangsspannung:	max. 3,3 V DC
DC-Ausgangsspannung:	5 V DC ±3%, max. 300 mA
Statusanzeige:	4x LED grün für GPIO 1 bis 4
	2x LED blau für GPIO 5 und GPIO 6
	1x LED grün für Discovery Board (+3,3 V DC)

Eigenschaft	Beschreibung/Wert
Ballast-Widerstand:	15 Ω/5 W
Befestigungslöcher:	4× Ø 3,2 mm für Discovery Board
Gewicht:	0,12 kg

Maßzeichnungen - *Discovery Board NP5*

Die Maße sind in [mm].



Anschlussbelegung - *Discovery Board NP5*

Stecker	Funktion
X1	Encoder 1 und Hallsensor
X2	Bremse

Stecker	Funktion
X3	Motor
X5	CANopen
X4	SPI über USB (Virtual COM-Port)
X6	Logik-Spannung
X7	Spannungsversorgung
X8	Steckplatz für NP5 Steuerung, siehe auch **Maßzeichnungen** und **Anschlussbelegung**
X9	Encoder 1/2 und Hallsensor
X10	GPIO und Kommunikationsschnittstelle
X13	Jumper zum Aktivieren/Deaktivieren der CANopen-Kommunikation
X15	+5V DC-Ausgang
P1	Potenziometer für den Analogeingang 1
P2	Potenziometer für den Analogeingang 2
SW1 bis SW4	Taster für GPIO 1 bis GPIO 4
SW5	Reset-Taster für das *Discovery Board*
SW6	Schalter für 120 Ohm Terminierungswiderstand (CANopen)
D1 bis D6	Statusanzeige für GPIO 1 bis GPIO 6
D7	Statusanzeige für das *Discovery Board* (+3,3 V DC)
G1	Erdungsanschluss

Stecker X1 - Encoder 1 und Hallsensor

Der Stecker X1 hat folgende Eigenschaften:

- Stecker-Typ: Phönix Grundleiste, MCV-0,5/8-G-2,5
- Spannungspegel: +5 V Logikpegel
- Strombelastbarkeit: max. 300 mA (zusammen mit +5 V DC Ausgangsspannung auf der Stiftleiste X15)
- Hall-Eingänge: intern durch 2,7 kΩ Pull-up Widerstand an +5 V DC angeschlossen

Pin	Name/Funktion
1	Hall_U (H1)
2	Hall_V (H2)
3	Hall_W (H3)
4	+5 V DC
5	GND
6	ENC1_A
7	ENC1_B
8	ENC1_I

Stecker X2 - Bremse

Der Stecker X2 hat folgende Eigenschaften:

- Stecker-Typ: Phönix Grundleiste, MCV-0,5/2-G-2,5

Pin	Name/Funktion
1	Bremse + (mit +UB verbunden)
2	Bremse - (PWM-gesteuerter Open-Drain-Ausgang, max. 1,5 A)

Stecker X3 - Motor

Der Stecker X3 hat folgende Eigenschaften:

- Stecker-Typ: Phönix Grundleiste, MCV-1,5/4-G-3,5
- max. Nennstrom 6A RMS
- max. Spitzenstrom 10A RMS (für 1s)

Pin	Schrittmotor	BLDC-Motor
1	A	U
2	A\	V
3	B	W
4	B\	

Stecker X4 - SPI über USB

Für diesen USB-Anschluss wird ein Kabel des Typs "Micro-USB" benötigt.

Den dazugehörigen Treiber *Nanotec_ComToSPI* finden Sie auf der Website **www.nanotec.de**.

Stecker X5 - CANopen

Der Stecker X5 hat folgende Eigenschaften:

- Stecker-Typ: RJ45 Duo Port, liegend

Pin	Name/Funktion
1	CAN+
2	CAN-
3	GND
4	N.C
5	N.C
6	CAN_Shield
7	GND
8	+UB_Logic (24 V DC ±5%)

Stecker X6 - Logik-Spannung

Der Stecker X6 hat folgende Eigenschaften:

- Stecker-Typ: Phönix Grundleiste, MCV-0,5/2-G-2,5

Pin	Name/Funktion
1	+UB_Logic (24 V DC ±5%)
2	GND

Stecker X7 - Betriebsspannung

Der Stecker X7 hat folgende Eigenschaften:

- Stecker-Typ: Phönix Grundleiste, MCV-1,5/2-G-3,5

Pin	Name/Funktion
1	+UB (12…48 V DC ±5%)

Pin	Name/Funktion
2	GND

Stecker X9 - Encoder und Hallsensoren

Der Stecker X9 hat folgende Eigenschaften:

- Stecker-Typ: Stiftleiste, einreihig, RM 2.54 mm, 12-polig, stehend
- Spannungspegel: +3,3 V DC Logikpegel

Pin	Name/Funktion
1	GND
2	ENC1_A
3	ENC1_B
4	ENC1_I
5	ENC1_CAP
6	ENC2_A
7	ENC2_B
8	ENC2_I
9	ENC2_CAP
10	Hall_U (H1)
11	Hall_V (H2)
12	Hall_W (H3)

Stecker X10 - I/O und Kommunikationsschnittstelle

Der Stecker X10 hat folgende Eigenschaften:

- Stecker-Typ: Stiftleiste, zweireihig, RM 2.54mm, 2x15 polig, stehend

Pin	Name	Typ	Anmerkung
1	GND	Masse	
2	U_REF_ANALOG	Out	Analog-Referenzspannung
3	DIO1_IO_CS	I/O	General I/O
4	DIO2_CD_CLK	I/O	General I/O
5	DIO3_CD_DIR	I/O	General I/O
6	DIO4_IO_MOSI	I/O	General I/O
7	DIO5_IO_MISO	I/O	General I/O
8	DIO6_IO_CLK	I/O	General I/O
9	ADC_ANALOG_1	In	AD-Wandler 1
10	ADC_ANALOG_2	In	AD-Wandler 2
11	GND	Masse	
12	SLOT_SPI_MOSI	-	SPI 1
13	SLOT_SPI_MISO	-	SPI 1
14	SLOT_SPI_SCK	-	SPI 1
15	SLOT_SPI_$\overline{CS}$	-	SPI 1
16	SLOT_SYNC	-	Systemfunktion, reserviert
17	SLOT_$\overline{RESET}$	-	Systemfunktion, reserviert

Pin	Name	Typ	Anmerkung
18	SLOT_BOOT	-	Systemfunktion, reserviert
19	GND	Masse	
20	COMM_RESET	-	Systemfunktion, reserviert
21	COMM_SYNC	-	Systemfunktion, reserviert
22	COMM_SPI_MOSI	-	SPI 2
23	COMM_SPI_MISO	-	SPI 2
24	COMM_SPI_SCK	-	SPI 2
25	COMM_SPI_$\overline{CS}$	-	SPI 2
26	GND	Masse	
27	CANopen ON	-	CANopen ON
28	I2CSCL_CANRX	-	I^2C Clock oder CANopen RX
29	I2CSDA_CANTX	-	I^2C Data oder CANopen TX
30	GND	Masse	

Stecker X13 - Jumper zum Aktivieren/Deaktivieren der CANopen-Kommunikation

Der Stecker X13 hat folgende Eigenschaften:

- Stecker-Typ: Stiftleiste, RM 2.54mm, 2 polig, stehend
- Mit Jumper gebrückt: CANopen aktiviert
- Mit Jumper nicht gebrückt: CANopen deaktiviert, SPI aktiviert

Pin	Name/Funktion
1	+3,3V
2	CANopen ON

Stecker X15 - +5V DC Ausgang

Der Stecker X15 hat folgende Eigenschaften:

- Stecker-Typ: Stiftleiste, RM 2.54 mm, 2 polig, stehend
- Strombelastbarkeit: max. 300 mA (zusammen mit +5 V DC Ausgangsspannung auf der Stiftleiste X1)

Pin	Name/Funktion
1	+5 V DC
2	GND

Inbetriebnahme SPI über das *Discovery Board*

Um Verbindung mit der *NP5-40* herzustellen, gehen Sie wie folgt vor:

1. Stecken Sie die *NP5-40* an X8 ein.
2. Stecken Sie den Jumper X13 ab.
3. Falls Sie die Steuerung über USB (Virtual COM-Port) ansprechen möchten, installieren Sie den Treiber *Nanotec_ComToSPI* und schließen Sie das USB-Kabel an X4 an.
   Falls Sie die Steuerung direkt über SPI ansprechen möchten, verbinden Sie den SPI-Master mit der Steuerung über die Leitungen SCK (source clock), MOSI (master out, slave in), MISO (master in, slave out) und CS (chip select). Überprüfen Sie, dass die Masse (GND) vom Master mit der Masse der Steuerung verbunden ist.

4. Schließen Sie Ihre Versorgungsspannung an X7 an.

5 Inbetriebnahme

In diesem Kapitel wird beschrieben, wie Sie die Kommunikation zur Steuerung aufbauen und die notwendigen Parameter einstellen, damit der Motor betriebsbereit ist.

Die Software *Plug & Drive Studio* bietet eine komfortable Möglichkeit, die Konfiguration vorzunehmen und die Steuerung an den angeschlossenen Motor anzupassen. Weiterführende Informationen finden Sie im Dokument *Plug & Drive Studio: Quick Start Guide* auf **http://www.nanotec.de**.

5.1 Kommunikationseinstellungen

5.1.1 SPI

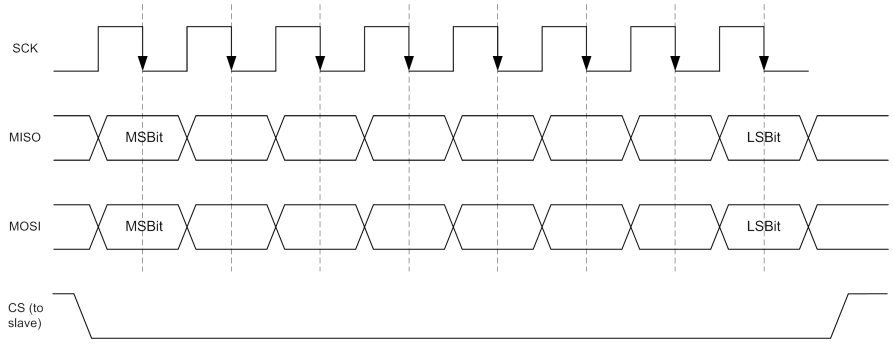
5.1.2 SPI-Einstellungen

Die SPI-Parameter sind folgendermaßen einzustellen (siehe auch nachfolgende Abbildung):

- Der Ruhepegel des Clock-Signals ist *low*.
- Die Bereitstellung eines Bitwertes (*MISO* und *MOSI*) geschieht auf der steigenden Flanke des Clock-Signals.
- Der Abtastzeitpunkt ist die fallende Flanke des Clock-Signals.
- Die Daten werden mit dem *Most Significant Bit* zuerst versendet und empfangen.
- Das *CS*-Signal ist *low* aktiv.
- Solange sich der SPI-Slave nicht auf den Millisekundentakt des SPI-Masters aufsynchronisiert hat, darf der SPI-Master nur alle zwei Millisekunden eine Nachricht übertragen.
  Wenn der SPI-Slave synchron zum Millisekundentakt des SPI-Masters läuft, darf der SPI-Master jede Millisekunde eine Nachricht übertragen.

Der *SPI-Slave* kann mit einer Frequenz von maximal 20 MHz angesteuert werden.

Folgende Abbildung zeigt den SPI-Signalverlauf:



5.1.3 Bus-Initialisierung

Die Slaves senden erst gültige Inhalte, nachdem einmalig eine korrekte Nachricht vom Master empfangen wurde. Die Bus-Initialisierung ist mit der ersten korrekt empfangenen Nachricht abgeschlossen.

5.2 Kommunikation aufbauen

5.2.1 SPI

Vor Beginn der Inbetriebnahme wird empfohlen die Kapitel **Anschließen der Steuerung** und Konfiguration **SPI** durchzulesen.

1. Verbinden Sie den SPI Master mit der Steuerung über die Leitungen *SCK* (source clock), *MOSI* (master out, slave in), *MISO* (master in, slave out) und *CS* (chip select). Überprüfen Sie dass die Masse (GND) vom Master mit der Masse der Steuerung verbunden ist.
2. Versorgen Sie die Steuerung mit Spannung.
3. Ändern Sie ggf. die Konfigurationswerte, siehe Konfiguration **SPI**.
4. Zum Testen der Schnittstelle senden Sie die Bytes `01 40 41 60 00 00 00 00 00 D4` an die Steuerung und nach Empfang des ersten Response (`02 00 00 00 00 00 00 00 00 51`) die Bytes `02 00 00 00 00 00 00 00 00 51`. (eine detaillierte Beschreibung der Nachrichten finden Sie im Kapitel **SPI-Nachricht**).
   Das Statusword (6041_h) wurde ausgelesen, Sie erhalten diese Response: `01 4B 41 60 00 XX XX 00 00 0A`

5.3 Motordaten einstellen

Die Steuerung benötigt vor der Inbetriebnahme des Motors einige Werte aus dem Motordatenblatt.

- Polpaarzahl: Objekt **2030**$_h$:00_h (Pole pair count) Hier ist die Anzahl der Motorpolpaare einzutragen. Bei einem Schrittmotor wird die Polpaarzahl über den Schrittwinkel berechnet, z.B. 1,8° = 50 Polpaare, 0,9° = 100 Polpaare (siehe Schrittwinkel im Motordatenblatt). Bei BLDC-Motoren ist die Polpaarzahl direkt im Motordatenblatt angegeben.
- Motorstrom/Motortyp einstellen:
  - Nur Schrittmotor: Objekt **2031**$_h$:00_h: Nennstrom (Bipolar) in mA (siehe Motordatenblatt)
    - Objekt **2031**$_h$:00_h: Nennstrom (Bipolar) in mA (siehe Motordatenblatt)
    - Objekt **3202**$_h$:00_h (Motor Drive Submode Select): Definiert den Motortyp Schrittmotor, aktiviert die Stromabsenkung bei Stillstand des Motors: 0000008h. Siehe auch Kapitel **Inbetriebnahme *Open Loop***.
  - Nur BLDC-Motor:
    - Objekt **2031**$_h$:00_h Spitzenstrom in mA (siehe Motordatenblatt)
    - Objekt **203B**$_h$:01_h Nennstrom in mA (siehe Motordatenblatt)
    - Objekt **203B**$_h$:02_h Maximale Dauer des Spitzenstroms in ms (für eine Erstinbetriebnahme wird ein Wert von 100ms empfohlen; dieser Wert ist später an die konkrete Applikation anzupassen).
    - Objekt **3202**$_h$:00_h (Motor Drive Submode Select): Definiert den Motortyp BLDC: 00000041h
- Motor mit Encoder: Objekt **2059**$_h$:00_h (Encoder Configuration): Je nach Encoderausführung ist einer der folgenden Werte einzutragen (siehe Motordatenblatt):
  - Versorgungsspannung 5V, differentiell: 00000000h
  - Versorgungsspannung 5V, single-ended: 00000002h
- Motor mit Bremse: Objekt **3202**$_h$:00_h (Motor Drive Submode Select): Für die Erstinbetriebnahme wird die Bremsensteuerung aktiviert. Abhängig von der konkreten Applikation kann diese Konfiguration bei Bedarf später wieder deaktiviert werden. Je nach Motortyp ist einer der folgenden Werte einzutragen:
  - Schrittmotor, Bremsensteuerung (und **Stromabsenkung** im Stillstand) aktiviert: 0000000Ch
  - BLDC-Motor, Bremsensteuerung aktiviert: 00000044h

5.4 Motor anschließen

Nach der Einstellung der Motorparameter, siehe **Motordaten einstellen**, schließen Sie den Motor und ggf. die vorhandenen Sensoren (Encoder/Hallsensoren) und die Bremse an.

- Motor anschließen:
  - an die entsprechenden Pins der PCI-Steckleiste, siehe **Anschlussbelegung**
  - an X3 des Discovery Boards, falls es verwendet wird, siehe **Stecker X3 - Motor**
- Encoder/Hallsensoren anschließen:

- an die entsprechenden Pins der PCI-Steckleiste, siehe **Anschlussbelegung**
- an X1 des Discovery Boards, falls es verwendet wird, siehe **Stecker X1 - Encoder 1 und Hallsensor**

- Bremse anschließen:
  - Minus an Pin A48 der PCI-Steckleiste, siehe **Anschlussbelegung**
  - Plus an UB_IN der PCI-Steckleiste oder direkt an die Spannungsversorgung, siehe **Anschlussbelegung**
  - an X2 des Discovery Boards, falls es verwendet wird, siehe **Stecker X2 - Bremse**

  Im Kapitel **Automatische Bremsensteuerung** wird beschrieben, wie die automatische Bremsensteuerung aktiviert werden kann.

5.5 Auto-Setup

Um einige Parameter im Bezug zum Motor und den angeschlossenen Sensoren (Encoder/ Hallsensoren) zu ermitteln, wird ein Auto-Setup durchgeführt. Der **Closed Loop**-Betrieb setzt ein erfolgreich abgeschlossenes Auto-Setup voraus.

Hinweis

- Beachten Sie die folgenden Voraussetzungen für das Durchführen des Auto-Setups:
- Der Motor muss lastfrei sein.
- Der Motor darf nicht berührt werden.
- Der Motor muss sich frei in beliebige Richtungen drehen können.
- Es darf kein NanoJ-Programm laufen (Objekt 2300_h:00_h Bit 0 = "0", siehe **2300h NanoJ Control**).

Tipp

Die Ausführung des Auto-Setups benötigt relativ viel Prozessorrechenleistung. Während des Auto-Setups können dadurch eventuell die Feldbusse nicht zeitgerecht bedient werden.

Hinweis

In diesem Modus sind die Endschalter und damit die Toleranzbänder aktiv. Für weitere Information zu den Endschaltern, siehe **Begrenzung des Bewegungsbereichs**.

Tipp

Solange sich der an der Steuerung angeschlossene Motor oder die Sensoren für die Rückführung (Encoder/Hallsensoren) nicht ändern, ist das Auto-Setup nur einmal bei der Erstinbetriebnahme durchzuführen.

5.5.1 Parameter-Ermittlung

Das Auto-Setup ermittelt über mehrere Test- und Messläufe verschiedene Parameter des angeschlossenen Motors und der vorhandenen Sensoren. Art und Anzahl der Parameter sind teilweise von der jeweiligen Motorkonfiguration abhängig.

Parameter	Alle Motoren unabhängig von der Konfiguration
Motortyp (Schrittmotor oder BLDC-Motor)	X

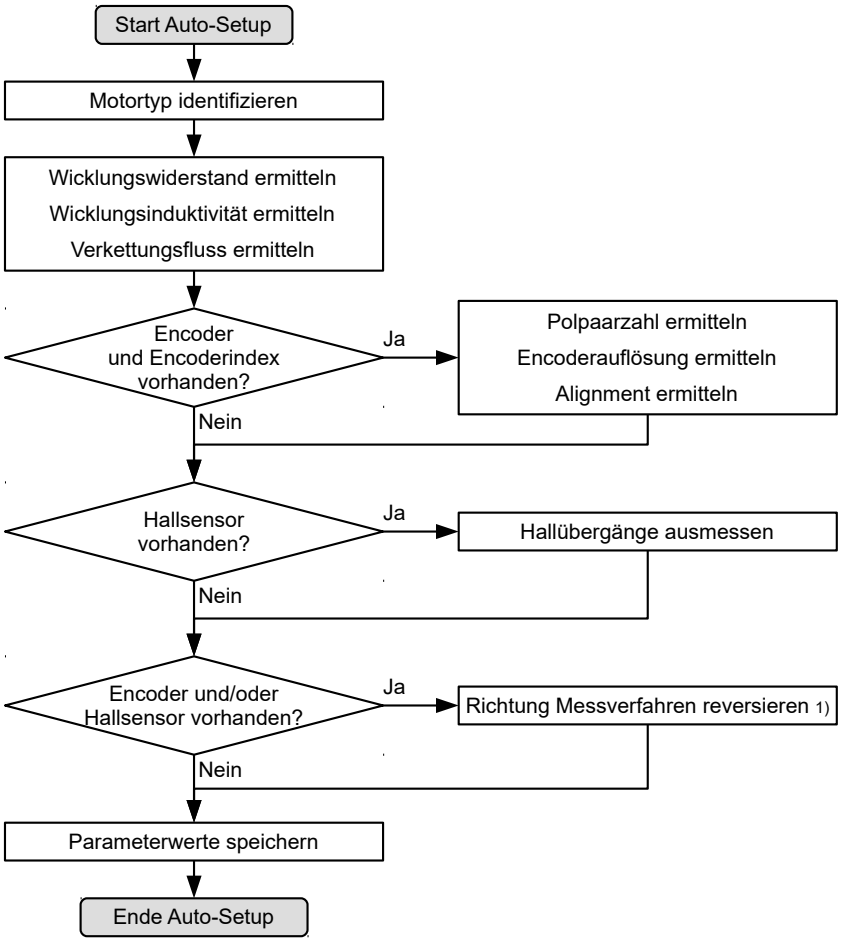
Parameter	Alle Motoren unabhängig von der Konfiguration
Wicklungswiderstand	X
Wicklungsinduktivität	X
Verkettungsfluss	X

Parameter	Motor ohne Encoder	Motor mit Encoder und Index	Motor mit Encoder ohne Index
Encoderauflösung	-	X	---
Alignment (Verschiebung des elektrischen Nullpunkts zum Index.)	-	X	---

Parameter	Motor ohne Hallsensor	Motor mit Hallsensor
Hallübergänge	-	X

5.5.2 Durchführung

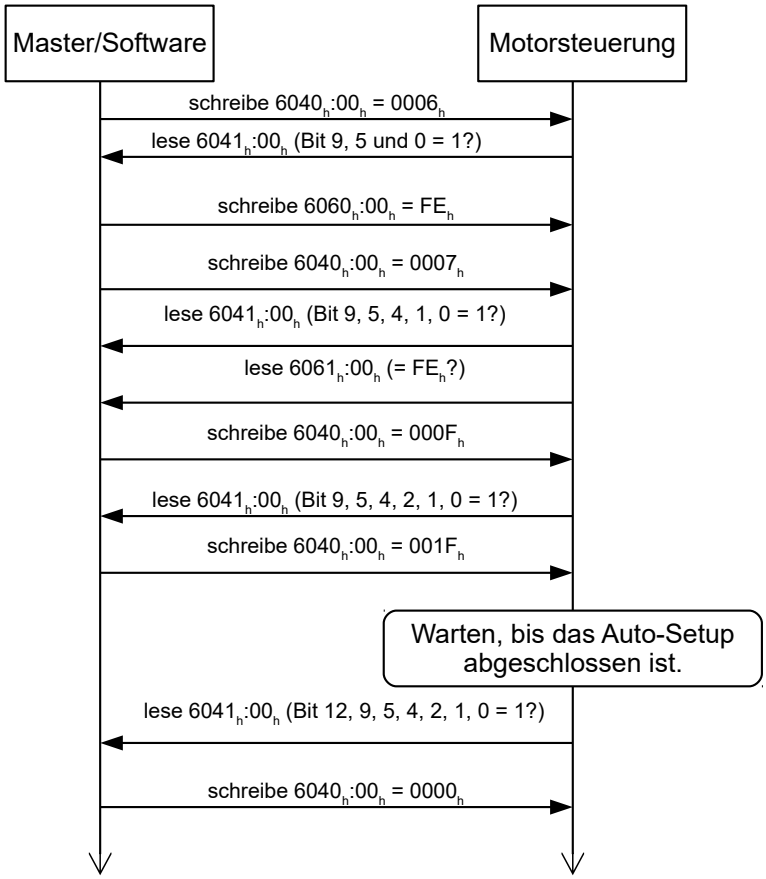
1. Zum Vorwählen des Betriebsmodus *Auto-Setup* tragen Sie in das Objekt 6060_h:00_h den Wert "-2" (="FE_h") ein.
   Die *Power state machine* muss nun in den Zustand *Operation enabled* versetzt werden, siehe **CiA 402 Power State Machine**.
2. Starten Sie das *Auto-Setup* mit Setzten von Bit 4 *OMS* im Objekt 6040_h:00_h (Controlword).



Während der Ausführung des Auto-Setups werden nacheinander folgende Tests und Messungen durchgeführt:

1) Zum Ermitteln der Werte wird die Richtung des Messverfahrens reversiert und die Flankenerkennung erneut ausgewertet.

Der Wert 1 im Bit 12 *OMS* im Objekt 6041_h:00_h (Statusword) zeigt an, dass das Auto-Setup vollständig durchgeführt und beendet wurde. Zusätzlich kann über das Bit 10 *TARG* im Objekt 6041_h:00_h abgefragt werden, ob ein Encoder-Index gefunden wurde (= "1") oder nicht (= "0").



5.5.3 Parameterspeicherung

Nach erfolgreichem *Auto-Setup* werden die ermittelten Parameterwerte automatisch in die zugehörigen Objekte übernommen und mit dem Speichermechanismus gespeichert, siehe **Objekte speichern** und **1010h Store Parameters**. Benutzt werden die Kategorien *Drive* $1010_h:05_h$ und *Tuning* $1010_h:06_h$.

VORSICHT

- Nach der Durchführung des Auto-Setup Modes ist das interne Koordinatensystem nicht mehr gültig.
- *Homing* alleine genügt nicht! Wird die Steuerung nicht neu gestartet, kann es zu unvorhersehbaren Reaktionen kommen.
- Starten Sie das Gerät nach einem Auto-Setup neu!

6 Generelle Konzepte

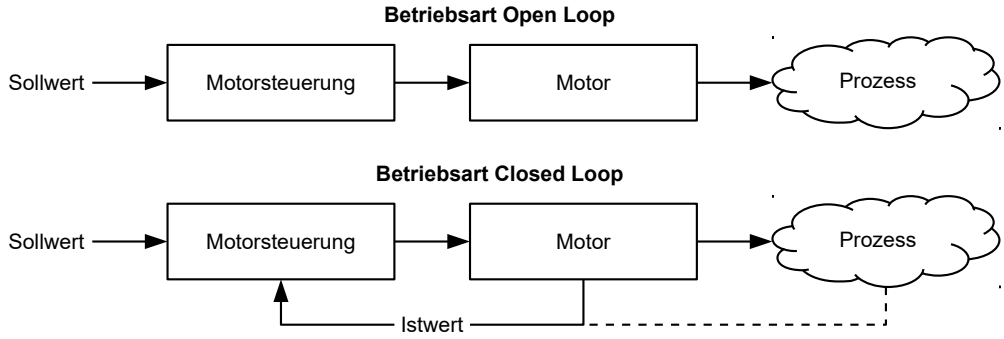
6.1 Betriebsarten

6.1.1 Allgemein

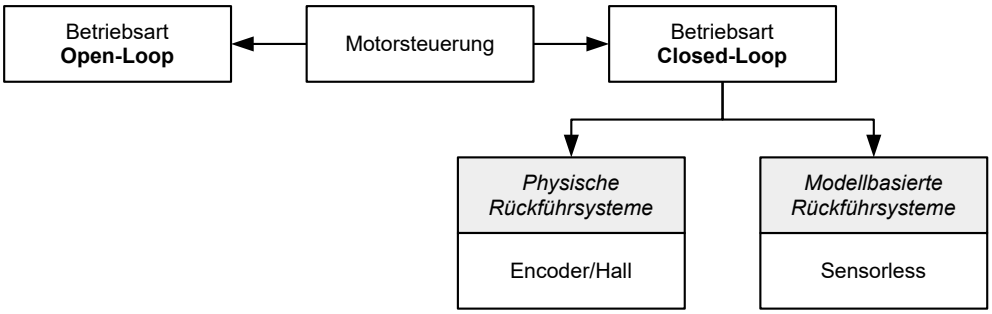
Die Betriebsart von Systemen ohne Rückführung wird als *Open Loop*, die mit Rückführung als *Closed Loop* bezeichnet. In der Betriebsart *Closed Loop* ist es zunächst unerheblich, ob die zurückgeführten Signale vom Motor selbst oder aus dem beeinflussten Prozess kommen.

Bei Steuerungen mit Rückführung wird die gemessene Regelgröße (Istwert) permanent mit einer Führungsgröße (Sollwert) verglichen. Bei Abweichungen zwischen diesen Größen regelt die Steuerung entsprechend den vorgegebenen Regelparametern nach.

Dagegen fehlt den reinen Steuerungen die Rückführung der zu regelnden Größe. Die Führungsgröße (Sollwert) wird lediglich vorgegeben.



Neben den physischen Rückführsystemen (beispielsweise über Encoder oder Hallsensoren) kommen auch modellbasierte Rückführsysteme, die alle unter dem Überbegriff Sensorless bekannt sind, zum Einsatz. Beide Rückführsystemen können auch in Kombination eingesetzt werden, um die Qualität der Regelung weiter zu verbessern.



Nachfolgend werden alle möglichen Kombinationen von Betriebsarten und Rückführsysteme im Bezug auf die Motorentechnik zusammengefasst. Die Unterstützung der jeweiligen Betriebsart und Rückführung ist steuerungsspezifisch und in den Kapiteln **Anschlussbelegung** und **Betriebsmodi** nachzulesen.

Betriebsart	Schrittmotor	BLDC-Motor
Open Loop	ja	nein
Closed Loop	ja	ja

Rückführung	Schrittmotor	BLDC-Motor
Hall	nein	ja
Encoder	ja	ja
Sensorless	ja	ja

In Abhängigkeit der Betriebsart können verschiedene Betriebsmodi angewendet werden. Die nachfolgende Liste fasst alle Betriebsmodi, die in den verschiedenen Betriebsarten möglich sind, zusammen.

Betriebsmodus	Betriebsart	
	Open Loop	Closed Loop
Profile Position	ja	ja
Velocity	ja	ja
Profile Velocity	ja	ja
Profile Torque	nein[1)]	ja
Homing	ja[2)]	ja
Interpolated Position Mode	ja[3)]	ja
Cyclic Synchronous Position	ja[3)]	ja
Cyclic Synchronous Velocity	ja[3)]	ja
Cyclic Synchronous Torque	nein[1)]	ja
Takt-Richtung	ja	ja

1) Die Drehmoment-Betriebsmodi **Profile Torque** und **Cyclic Synchronous Torque** sind in der Betriebsart *Open Loop* aufgrund einer fehlenden Rückführung nicht möglich.

2) Ausnahme: Homing auf Block ist aufgrund einer fehlenden Rückführung nicht möglich.

3) Da sich Rampen und Geschwindigkeiten in den Betriebsmodi **Cyclic Synchronous Position** und **Cyclic Synchronous Velocity** aus den vorgegeben Punkten des Masters ergeben, ist es normalerweise nicht möglich, diese Parameter so vorzuwählen und zu erproben, dass ein Schrittverlust ausgeschlossen werden kann. Es wird deshalb davon abgeraten, diese Betriebsmodi in Verbindung mit der Betriebsart *Open Loop* zu verwenden.

6.1.2 Open Loop

Einführung

Die Betriebsart *Open Loop* wird nur bei Schrittmotoren angewendet und ist ein reiner Stellbetrieb. Die Felddrehung im Stator wird durch die Steuerung vorgegeben. Der Rotor folgt der magnetischen Felddrehung ohne Schrittverluste unmittelbar, solange keine Grenzparameter - wie beispielsweise das maximal mögliche Drehmoment - überschritten werden. Im Vergleich zum *Closed Loop* werden keine komplexen internen Regelungsprozesse in der Steuerung benötigt. Dadurch sind die Anforderungen an die Steuerungshardware wie auch an die Steuerungslogik sehr gering. Im Besonderen bei preissensitiven Anwendungen und einfachen Bewegungsaufgaben wird deshalb die Betriebsart *Open Loop* vorwiegend eingesetzt.

Da es im Gegensatz zu *Closed Loop* keine Rückkopplung über die aktuelle Rotorposition gibt, kann auch kein Rückschluss auf das an der Abtriebsseite der Motorwelle anstehende Gegenmoment gezogen werden. Um eventuell an der Abtriebswelle des Motors auftretende Drehmomentschwankungen auszugleichen, liefert die Steuerung in der Betriebsart *Open Loop* über den gesamten Drehzahlbereich immer den maximal möglichen (bzw. durch Parameter vorgegebenen) eingestellten Strom an die Statorwicklungen. Die dadurch erzeugte hohe magnetische Feldstärke zwingt den Rotor, in kürzester Zeit den neuen Beharrungszustand einzunehmen. Diesem Moment

steht jedoch das Trägheitsmoment des Rotors entgegen. Unter bestimmten Betriebsbedingungen neigt diese Kombination zu Resonanzen, vergleichbar einem Feder-Masse-System.

Inbetriebnahme

Um die Betriebsart *Open Loop* anzuwenden, sind folgende Einstellungen notwendig:

- Im Objekt **2030**$_h$ (Pole Pair Count) die Polpaarzahl eingeben (siehe Motordatenblatt: Ein Schrittwinkel von 1,8° entspricht bei einem Schrittmotor mit 2 Phasen 50 Polpaaren und von 0,9° entspricht 100 Polpaaren).
- Im Objekt **2031**$_h$ (Max Current) den Maximalstrom in mA eingeben (siehe Motordatenblatt).
- Im Objekt **3202**$_h$ (Motor Drive Submode Select) das Bit 0 (CL/OL) mit dem Wert "0" belegen.
- Soll der Takt-Richtungs-Modus angewendet werden, dann Kapitel **Takt-Richtungs-Modus** berücksichtigen.

Bei Bedarf sollte die Stromabsenkung bei Stillstand des Motors aktiviert werden, um die Verlustleistung und Wärmeentwicklung zu reduzieren. Um die Stromabsenkung zu aktivieren, sind folgende Einstellungen notwendig:

- Im Objekt **3202**$_h$ (Motor Drive Submode Select) das Bit 3 (CurRed) auf "1" setzen.
- Im Objekt **2036**$_h$ (Open Loop Current Reduction Idle Time) wird die Zeit in Millisekunden angegeben, die sich der Motor im Stillstand befinden muss, bis die Stromabsenkung aktiviert wird.
- Im Objekt **2037**$_h$ (Open Loop Current Reduction Value/factor) wird der Effektivwert angegeben, auf den der Nennstrom reduziert werden soll, wenn die Stromabsenkung im *Open Loop* aktiviert wird und sich der Motor im Stillstand befindet.

Optimierungen

Systembedingt können in der Betriebsart *Open Loop* Resonanzen auftreten, besonders bei geringer Belastung ist die Resonanzneigung hoch. Aus praktischen Erfahrungen heraus haben sich in Abhängigkeit der Applikation verschiedene Maßnahmen bewährt, um Resonanzen weitgehend zu reduzieren:

- Strom reduzieren oder erhöhen, siehe Objekt **2031**$_h$ (Max Current). Zu hohe Drehmomentreserve begünstigt Resonanzen.
- Die Betriebsspannung unter Berücksichtigung der produktspezifisch zugelassenen Bereiche reduzieren (bei genügender Drehmomentreserve) oder erhöhen. Der zulässige Betriebsspannungsbereich kann dem Produktdatenblatt entnommen werden.
- Die Regelparameter des Stromreglers über die Objekte **3210**$_h$:09$_h$ (I_P) und **3210**$_h$:0A$_h$ (I_I) optimieren.
- Anpassen der Beschleunigung, Verzögerung und/oder Zielgeschwindigkeit in Abhängigkeit des gewählten Betriebsmodus:

  Betriebsmodus Profile Position

  Objekte **6083**$_h$ (Profile Acceleration), **6084**$_h$ (Profile Deceleration) und **6081**$_h$ (Profile Velocity).

  Betriebsmodus Velocity

  Objekte **6048**$_h$ (Velocity Acceleration), **6049**$_h$ (Velocity Deceleration) und **6042**$_h$ (Target Velocity).

  Betriebsmodus Profile Velocity

  Objekte **6083**$_h$ (Profile Acceleration), **6084**$_h$ (Profile Deceleration) und **6081**$_h$ (Profile Velocity).

  Betriebsmodus Homing

  Objekte **609A**$_h$ (Homing Acceleration), **6099**$_h$:01$_h$ (Speed During Search For Switch) und **6099**$_h$:02$_h$ (Speed During Search For Zero).

Betriebsmodus Interpolated Position Mode

Mit der übergeordneten Steuerung können die Beschleunigungs- und Verzögerungsrampen beeinflusst werden.

Betriebsmodus Cycle Synchronous Position

Über die externen Zielvorgaben "Positionsvorgabe/Zeiteinheit" können die Beschleunigungs- und Verzögerungsrampen beeinflusst werden.

Betriebsmodus Cycle Synchronous Velocity

Über die externen Zielvorgaben "Positionsvorgabe/Zeiteinheit" können die Beschleunigungs- und Verzögerungsrampen beeinflusst werden.

Betriebsmodus Takt-Richtung

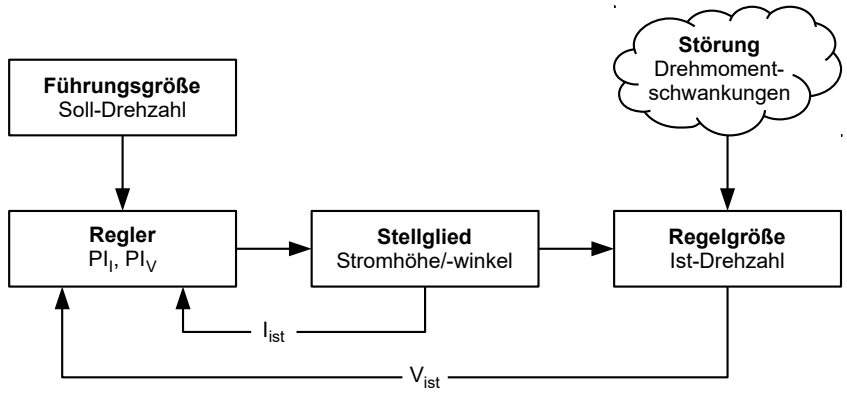
Änderung der Schrittauflösung über die Objekte **2057**$_h$ (Clock Direction Multiplier) und **2058**$_h$ (Clock Direction Divider). Beschleunigungs-/Verzögerungsrampen durch Anpassen der Impulsfrequenz optimieren, um den Resonanzbereich möglichst schnell zu durchlaufen.

6.1.3 Closed Loop

Einführung

Die *Closed Loop*-Theorie geht auf die Vorstellung eines Regelkreises zurück. Eine am System einwirkende Störgröße soll möglichst schnell und ohne bleibende Abweichung ausgeregelt werden, um die Regelgröße wieder an die Führungsgröße anzugleichen.

Closed Loop am Beispiel einer Drehzahlregelung:



PI$_I$	=	Proportional-/Integralregler Stromregelkreis
PI$_V$	=	Proportional-/Integralregler Drehzahlregelkreis
I$_{ist}$	=	Aktueller Strom
V$_{ist}$	=	Aktuelle Drehzahl

Das *Closed Loop*-Verfahren wird auch als "Sinuskommutierung über Encoder mit feldorientierter Regelung" bezeichnet. Kern der *Closed Loop*-Technologie ist die leistungsangepasste Stromregelung sowie die Rückführung der Istwerte des Prozesses. Über die Signale des Encoders wird die Rotorlage erfasst und es werden in den Motorwicklungen sinusförmige Phasenströme erzeugt. Durch die Vektorregelung des Magnetfelds ist gewährleistet, dass das Statormagnetfeld immer senkrecht zum Rotormagnetfeld steht und die Feldstärke genau dem gewünschten Drehmoment entspricht. Der in den Wicklungen so gesteuerte Strom sorgt für eine gleichmäßige Motorkraft und führt zu einem besonders ruhig laufenden Motor, der sich genau regeln lässt.

Die für die Betriebsart *Closed Loop* notwendige Rückführung der Regelgrößen kann mit verschiedenen Technologien realisiert werden. Neben der physischen Rückführung mit Encoder oder Hall-Sensoren, ist auch eine virtuelle Erfassung der Motorparameter durch softwarebasierte Modellberechnung möglich. Physikalische Größen, wie Geschwindigkeit oder Gegen-EMK, können mit Hilfe eines sogenannten "Beobachters" aus den Daten des Stromreglers rekonstruiert werden. Mit dieser Sensorless-Technologie erhält man einen"virtuellen Drehgeber", der ab einer gewissen Minimalgeschwindigkeit die Positions- und Drehzahlinformation mit der gleichen Präzision liefert wie ein realer optischer oder magnetischer Drehgeber.

Alle Steuerungen von Nanotec, welche die Betriebsart *Closed Loop* unterstützen, implementieren eine feldorientierte Regelung mit einer sinuskommutierten Stromregelung. Die Schrittmotoren und BLDC-Motoren werden also genauso geregelt wie ein Servomotor. Mit der Betriebsart *Closed Loop* können Schrittwinkelfehler während der Fahrt kompensiert und Lastwinkelfehler innerhalb eines Vollschritts korrigiert werden.

Inbetriebnahme

Vor dem Anwenden der Betriebsart *Closed Loop* muss ein Auto-Setup durchgeführt werden. Der Betriebsmodus Auto-Setup ermittelt automatisch die notwendigen Parameter (z.B. Motorkenndaten, Rückführsysteme), welche für eine optimale Arbeitsweise der feldorientierten Regelung notwendig sind. Alle Informationen zur Durchführung des Auto-Setups sind im Kapitel **Auto-Setup** beschrieben.

Um die Betriebsart *Closed Loop* anzuwenden, sind je nach Motortyp und Rückführung bestimmte Einstellungen notwendig, siehe Kapitel **Motordaten einstellen**. Das Bit 0 im **3202**$_h$ muss gesetzt sein. Wenn der Encoder für die Kommutierung verwendet wird, muss der Index des Encoders mindestens einmal nach dem Einschalten überfahren werden (das Bit 15 im **6041h Statusword** wird gesetzt).

6.2 CiA 402 Power State Machine

6.2.1 Zustandsmaschine

CiA 402

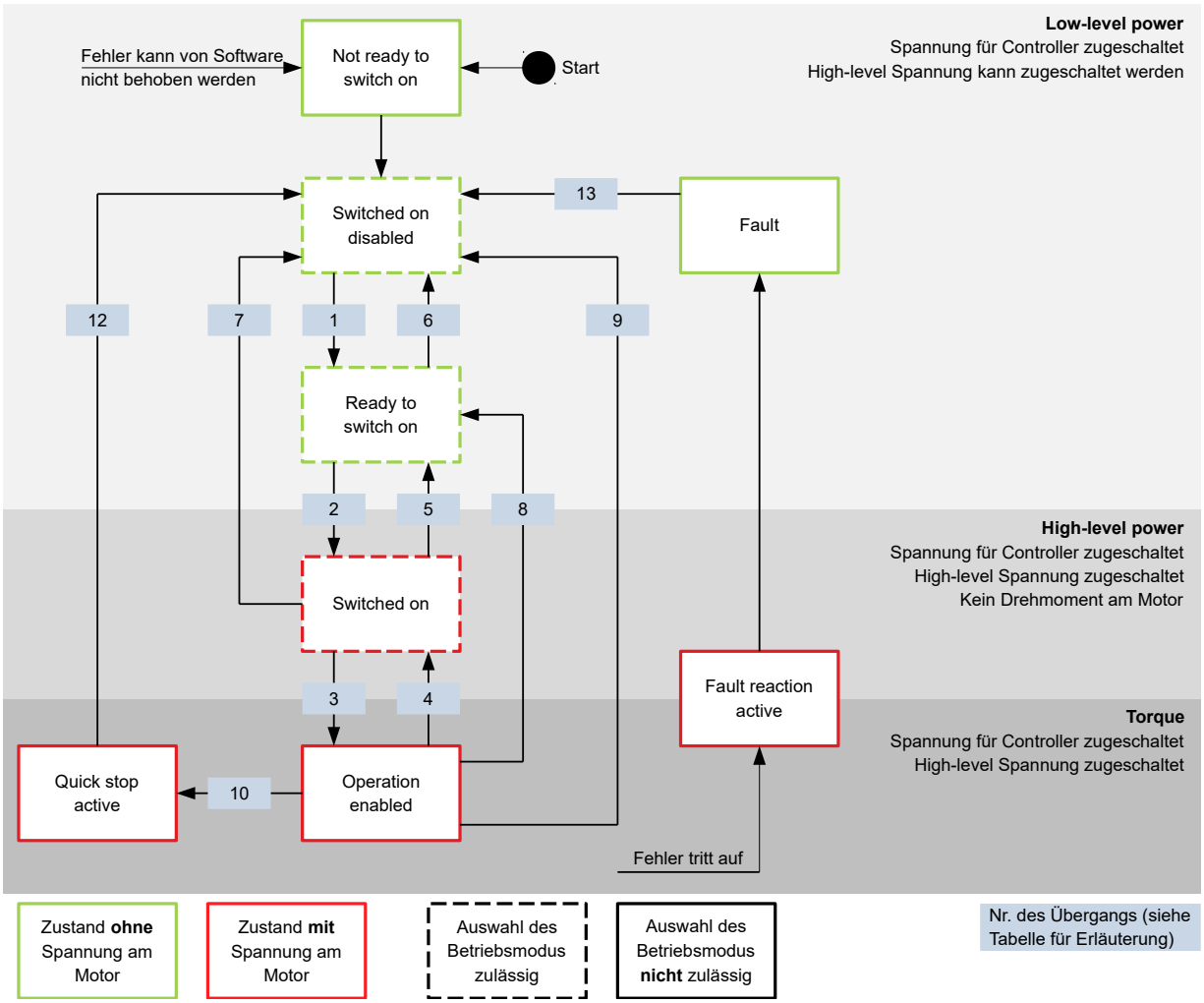
Um die Steuerung betriebsbereit zu schalten, ist es notwendig, eine Zustandsmaschine (*State Machine*) zu durchlaufen. Diese ist im *CANopen-Standard 402* definiert. Zustandsänderungen werden im Objekt **6040**$_h$ (Controlword) angefordert. Der tatsächliche Zustand der Zustandsmaschine lässt sich aus dem Objekt **6041**$_h$ (Statusword) entnehmen.

Controlword

Zustandsänderungen werden über Objekt **6040**$_h$ (Controlword) angefordert.

Zustandsübergänge

Das Diagramm zeigt die möglichen Zustandsübergänge.



In der nachfolgenden Tabelle sind die Bit-Kombinationen für das Controlword aufgelistet, die zu den entsprechenden Zustandsübergängen führen. Ein X entspricht dabei einem nicht weiter zu berücksichtigenden Bit-Zustand. Einzige Ausnahme ist das Rücksetzen des Fehlers (Fault reset): Der Übergang wird nur durch steigende Flanke des Bits angefordert.

Kommando	**Bit im Objekt 6040_h**					**Übergang**
	Bit 7	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	
Shutdown	0	X	1	1	0	1, 5, 8
Switch on	0	0	1	1	1	2
Disable voltage	0	X	X	0	X	6, 7, 9, 12
Quick stop	0	X	0	1	X	10
Disable operation	0	0	1	1	1	4
Enable operation	0	1	1	1	1	3
Fault reset		X	X	X	X	13

Haltemoment im Zustand *Switched On*

Im Status *Switched On* wird ab Werk kein Haltemoment aufgebaut. Wird in diesem Zustand bereits Haltemoment benötigt, muss in das **3212**$_h$:01$_h$ der Wert "1" geschrieben werden.

Hinweis

Ist die Option *Haltemoment im Zustand Switched on* aktiv, kann es beim Umschalten der Betriebsmodi dazu führen, dass der Motor ruckt.

Statusword

In der nachfolgenden Tabelle sind die Bitmasken aufgelistet, die den Zustand der Steuerung aufschlüsseln.

Statusword (6041$_h$)	Zustand
xxxx xxxx x0xx 0000	Not ready to switch on
xxxx xxxx x1xx 0000	Switch on disabled
xxxx xxxx x01x 0001	Ready to switch on
xxxx xxxx x01x 0011	Switched on
xxxx xxxx x01x 0111	Operation enabled
xxxx xxxx x00x 0111	Quick stop active
xxxx xxxx x0xx 1111	Fault reaction active
xxxx xxxx x0xx 1000	Fault

Die Steuerung erreicht nach Einschalten und erfolgreichem Selbsttest den Zustand *Switch on disabled*.

Betriebsmodus

Der eingestellte Betriebsmodus (**6060**$_h$) wird erst im Zustand *Operation enabled* aktiv. Der tatsächlich aktive Betriebsmodus wird im **6061**$_h$ angezeigt.

Die Einstellung oder Änderung des Betriebsmodus ist nur in folgenden Zuständen möglich (siehe gestrichelt umrahmte Zustände im Diagramm):

- Switch on disabled
- Ready to switch on
- Switched on

Im laufenden Betrieb (*Operation enabled*) ist es nicht möglich, den Betriebsmodus zu wechseln. Der Zustand *Fault* wird verlassen, wenn das Bit 7 in Objekt **6040**$_h$ (Controlword) von "0" auf "1" gesetzt wird (steigende Flanke).

Hinweis

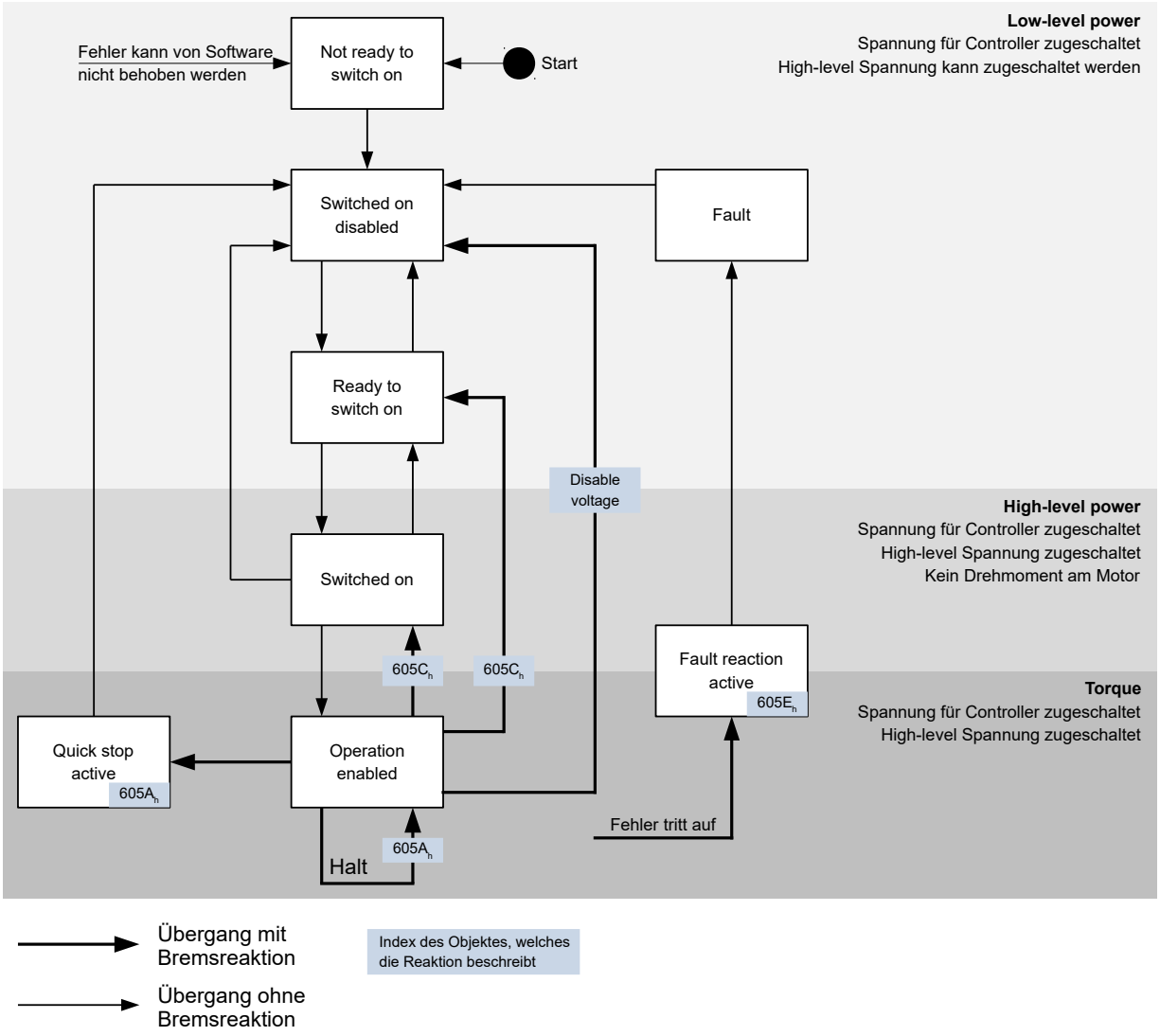
Tritt ein nicht behebbarer Fehler auf, wechselt die Steuerung in den Zustand *Not ready to switch on* und verbleibt dort.

6.2.2 Verhalten beim Verlassen des Zustands *Operation enabled*

Bremsreaktionen

Beim Verlassen des Zustands *Operation enabled* lassen sich unterschiedliche Bremsreaktionen programmieren.

Die nachfolgende Grafik zeigt eine Übersicht der Bremsreaktionen.



Quick stop active

Übergang in den Zustand *Quick stop active* (quick stop option):

In diesem Fall wird die in Objekt **605A$_h$** hinterlegte Aktion ausgeführt (siehe nachfolgende Tabelle).

Wert in Objekt 605A$_h$	Beschreibung
-32768 bis -1	Reserviert
0	Soforthalt
1	Abbremsen mit *slow down ramp* (Bremsbeschleunigung je nach Betriebsmodus) und anschließendem Zustandswechsel in *Switch on disabled*
2	Abbremsen mit *quick stop ramp* und anschließendem Zustandswechsel in *Switch on disabled*
3 bis 32767	Reserviert

Ready to switch on

Übergang in den Zustand *Ready to switch on* (shutdown option):

In diesem Fall wird die in Objekt **605B$_h$** hinterlegte Aktion ausgeführt (siehe nachfolgende Tabelle).

Wert in Objekt 605B$_h$	Beschreibung
-32768 bis -1	Reserviert
0	Soforthalt
1	Abbremsen mit *slow down ramp* (Bremsbeschleunigung je nach Betriebsmodus) und anschließendem Zustandswechsel in *Switch on disabled*
2 bis 32767	Reserviert

Switched on

Übergang in den Zustand *Switched on* (disable operation option):

In diesem Fall wird die in Objekt **605C$_h$** hinterlegte Aktion ausgeführt (siehe nachfolgende Tabelle).

Wert in Objekt 605C$_h$	Beschreibung
-32768 bis -1	Reserviert
0	Soforthalt
1	Abbremsen mit *slow down ramp* (Bremsbeschleunigung je nach Betriebsmodus) und anschließendem Zustandswechsel in *Switch on disabled*
2 bis 32767	Reserviert

Halt

Das Bit ist gültig in folgenden Modi:

- **Profile Position**
- **Velocity**
- **Profile Velocity**
- **Profile Torque**
- **Interpolated Position Mode**

Beim Setzen des Bit 8 in Objekt **6040$_h$** (Controlword) wird die in **605D$_h$** hinterlegte Reaktion ausgeführt (siehe nachfolgende Tabelle):

Wert in Objekt 605D$_h$	Beschreibung
-32768 bis 0	Reserviert
1	Abbremsen mit *slow down ramp* (Bremsbeschleunigung je nach Betriebsmodus)
2	Abbremsen mit *quick stop ramp* (Bremsbeschleunigung je nach Betriebsmodus)
3 bis 32767	Reserviert

Fault

Fehlerfall (fault):

Sollte ein Fehler auftreten, wird der Motor abgebremst, wie es in Objekt **605E$_h$** hinterlegt ist.

Wert in Objekt $605E_h$	Beschreibung
-32768 bis -1	Reserviert
0	Soforthalt
1	Abbremsen mit *slow down ramp* (Bremsbeschleunigung je nach Betriebsart)
2	Abbremsen mit *quick stop ramp* (Bremsbeschleunigung je nach Betriebsart)
3 bis 32767	Reserviert

Schleppfehler

Sollte ein Schleppfehler auftreten, wird der Motor abgebremst, wie es in Objekt **3700**$_h$ hinterlegt ist.

Wert	Beschreibung
-32768 bis -1	Reserviert
0	Soforthalt
1	Abbremsen mit *slow down ramp* (Bremsbeschleunigung je nach Betriebsart)
2	Abbremsen mit *quick stop ramp* (Bremsbeschleunigung je nach Betriebsart)
3 bis 32767	Reserviert

Die Schleppfehlerüberwachung kann deaktiviert werden, indem das Objekt **6065**$_h$ auf den Wert "-1" (FFFFFFFF$_h$) gesetzt wird.

6.3 Benutzerdefinierte Einheiten

Die Steuerung unterstützt die Möglichkeit, benutzerdefinierte Einheiten einzustellen. Damit lassen sich die entsprechenden Parameter z. B. direkt in Grad [°], [mm], usw. setzen und auslesen.

6.3.1 Berechnungsformeln für Benutzereinheiten

Positionsangaben

Alle Positionswerte im *Open Loop* und im *Closed Loop*-Betrieb werden in der Auflösung des virtuellen Positionsencoders angegeben. Diese berechnet sich aus den virtuellen Encoder-Inkrementen (**608F**$_h$:1$_h$ (Encoder Increments)) pro Motorumdrehungen (**608F**$_h$:2$_h$ (Motor Revolutions)) :

$$\text{Auflösung virtueller Positionsencoder} = \frac{\text{Encoder-Inkremente (608F}_h\text{:01)}}{\text{Motorumdrehungen (608F}_h\text{:02)}}$$

Sollte der Wert **608F**$_h$:1$_h$ oder der Wert **608F**$_h$:2$_h$ auf "0" gesetzt werden, rechnet die Steuerung intern mit einer "1" weiter. Die Werkseinstellungen sind:

- Encoder-Inkremente **608F**$_h$:1 = "2000"
- Motorumdrehungen **608F**$_h$:2 = "1"

Beispiel

608F$_h$:2$_h$ ist auf dem Wert "1", **608F**$_h$:1$_h$ auf dem Wert "2000" (Default). Somit ist die Benutzereinheit 2000 Inkremente pro Umdrehung. Das entspricht bei einem Schrittmotor mit 1,8° Schrittwinkel dem Schrittmodus *Zehntelschritt* .

Bei einer Zielposition (**607A**$_h$) von 2000 fährt der Motor genau eine mechanische Umdrehung

Die physikalische Auflösung des angeschlossenen Positionsencoders (der vorhandenen Rückführung allgemein) wird in Objekt **2052**$_h$ eingestellt bzw. vom **Auto-Setup** ermittelt.

Getriebeübersetzung

Die Getriebeübersetzung berechnet sich aus Motorumdrehung (**6091**$_h$:1 (Motor Revolutions)) pro Achsenumdrehung (**6091**$_h$:2 (Shaft Revolutions)) wie folgt:

$$\text{Getriebeübersetzung} = \frac{\text{Motorumdrehung } (6091_h{:}1)}{\text{Achsenumdrehung } (6091_h{:}2)}$$

Sollten Objekt **6091**$_h$:1 oder Objekt **6091**$_h$:2 auf "0" gesetzt werden, setzt die Firmware den Wert auf "1".

Vorschubkonstante

Die Vorschubkonstante wird aus dem Vorschub (**6092**$_h$:1 (Feed Constant) pro Umdrehung der Antriebsachse (**6092**$_h$:2 (Shaft Revolutions) wie folgt berechnet:

$$\text{Vorschubkonstante} = \frac{\text{Vorschub } (6092_h{:}1)}{\text{Umdrehung der Antriebsachse } (6092_h{:}2)}$$

Dies ist zur Angabe der Spindelsteigung bei einer Linearachse nützlich.

Sollte Objekt **6092**$_h$:1 oder Objekt **6092**$_h$:2 auf "0" gesetzt werden, setzt die Firmware den Wert auf "1".

Position

Die aktuelle Position in Benutzereinheiten (**6064**$_h$) und die Zielposition (**607A**$_h$) berechnen sich wie folgt:

$$\text{Position} = \frac{608F_h{:}01 \times \text{Vorschubkonstante } (6092_h)}{608F_h{:}02 \times \text{Getriebeübersetzung } (6091_h)}$$

Geschwindigkeit

Die Geschwindigkeitsvorgaben der nachfolgenden Objekte können ebenfalls in Benutzereinheiten angegeben werden:

Objekt	Modus	Bedeutung
606B$_h$	Profile Velocity Mode	Ausgabewert des Rampengenerators
60FF$_h$	Profile Velocity Mode	Geschwindigkeitsvorgabe
6099$_h$	Homing Mode	Geschwindigkeit zum Suchen des Index / Schalters
6081$_h$	Profile Position Mode	Zielgeschwindigkeit
6082$_h$	Profile Position Mode	Endgeschwindigkeit
2032$_h$	Profile Torque	Maximale Geschwindigkeit

Die interne Einheit ist Umdrehungen pro Sekunde (U/s).

Der Faktor n für die Geschwindigkeit errechnet sich aus Faktor für Zähler (**2061**$_h$) geteilt durch Faktor für Nenner (**2062**$_h$).

$$n_{Geschwindigkeit} = \frac{2061_h}{2062_h}$$

Bei der Eingabe von Werten gilt entsprechend: Interner Wert = $n_{Geschwindigkeit}$ x Eingabewert

Bei der Ausgabe von Werten gilt entsprechend: Ausgabewert = Interner Wert / $n_{Geschwindigkeit}$

Beispiel

2061$_h$ ist auf dem Wert "1", **2062**$_h$ auf dem Wert "60" (Default). Somit ist die Benutzereinheit "Umdrehung pro Minute" und $n_{Geschwindigkeit}$ = 1/60.

Wird das **60FF**$_h$ mit dem Wert "300" beschrieben, wird der interne Wert auf 300 U/min x 1/60 = 5 U/s gestellt.

Dreht der Motor mit einer internen Geschwindigkeit von 5 U/s, dann wird das Objekt **606B**$_h$ auf einer Geschwindigkeit von 5 / 1/60 = 300 U/min stehen.

Beschleunigung

Die Beschleunigung kann ebenfalls in Benutzereinheiten angegeben werden:

Objekt	Modus	Bedeutung
609A$_h$	Homing Mode	Beschleunigung
6083$_h$	Profile Position Mode	Beschleunigung
6084$_h$	Profile Position Mode	Bremsbeschleunigung
60C5$_h$	Profile Velocity Mode	Beschleunigung
60C6$_h$	Profile Position Mode	Bremsbeschleunigung
6085$_h$	Zustand "Quick stop active" (**CiA 402 Power State Machine**)	Bremsbeschleunigung

Die interne Einheit ist Umdrehungen pro Sekunde^2 (U/s^2).

Der Faktor n für die Beschleunigung errechnet sich aus Skalierungswert für Zähler (**2063**$_h$) geteilt durch Skalierungswert für Nenner (**2064**$_h$).

$$n_{Beschleunigung} = \frac{2063_h}{2064_h}$$

Bei der Eingabe von Werten gilt entsprechend: Interner Wert = $n_{Beschleunigung}$ x Eingabewert

Beispiel

2063$_h$ ist auf dem Wert "1", **2064**$_h$ auf dem Wert "60". Somit ist die Benutzereinheit *Umdrehung pro Minute pro Sekunde* und $n_{Beschleunigung}$ = 1/60.

Wird das **60C5**$_h$ mit dem Wert "600" beschrieben, wird der interne Wert auf 600 U/(s*min) x 1/60 = 10 U/s^2 gestellt.

Sollte Objekt **2063**$_h$ oder Objekt **2064**$_h$ auf "0" gesetzt werden, setzt die Firmware den Wert auf "1".

Ruck

Für den Ruck lassen sich die Objekte **60A4**$_h$:1$_h$ bis **60A4**$_h$:4$_h$ in Benutzereinheiten angeben. Diese Objekte betreffen nur den *Profile Position Mode* und den *Profile Velocity Mode*.

Die interne Einheit ist Umdrehungen pro Sekunde^3 (U/s^3).

Der Faktor n für die Beschleunigung errechnet sich aus Faktor für Zähler (**2065**$_h$) geteilt durch Faktor für Nenner (**2066**$_h$).

$$n_{Ruck} = \frac{2065_h}{2066_h}$$

Bei der Eingabe von Werten gilt entsprechend: Interner Wert = n_{Ruck} x Eingabewert

Beispiel

2063$_h$ ist auf dem Wert "1", **2064**$_h$ auf dem Wert "60". Somit ist die Benutzereinheit "Umdrehung pro Minute pro Sekunde hoch 2" und n_{Ruck} = 1/60.

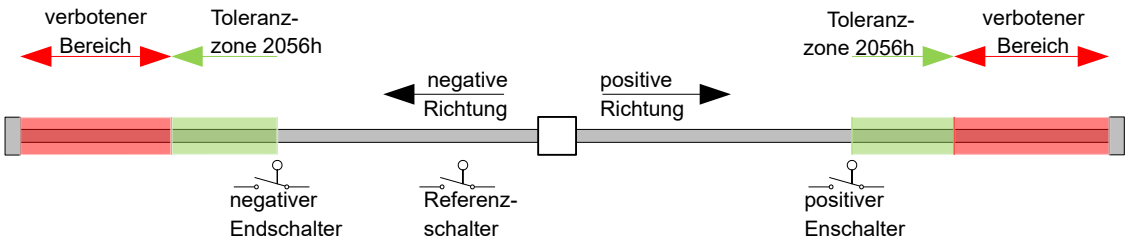
Wird das **60A4**$_h$ mit dem Wert "500" beschrieben, wird der interne Wert auf 500 U/(min * s^2) x 1/60 = 8,3 U/s^3 gestellt.

Wird Objekt **2065**$_h$ oder Objekt **2066**$_h$ auf "0" gesetzt, setzt die Firmware den Wert auf "1".

6.4 Begrenzung des Bewegungsbereichs

Die digitalen Eingänge können als Endschalter verwendet werden, im Kapitel **Digitale Eingänge** wird beschrieben, wie Sie diese Funktion der Eingänge aktivieren. Die Steuerung unterstützt auch Software-Endschalter.

6.4.1 Toleranzbänder der Endschalter



Das vorherige Bild stellt die Aufteilung der Toleranzbänder neben den Endschaltern dar:

- Die Toleranzzone beginnt unmittelbar nach dem Endschalter. In dieser Zone kann frei gefahren werden. Die Länge der Zone kann in dem Objekt **2056**$_h$ eingestellt werden.
- Falls der Motor in den verbotenen Bereich fährt, löst die Steuerung einen Soforthalt aus und es wird in den Zustand *Fault* gewechselt, siehe auch **Zustandsübergänge**.

6.4.2 Software-Endschalter

Die Steuerung berücksichtigt Software-Endschalter (**607D**$_h$ (Software Position Limit)). Zielpositionen (**607A**$_h$) werden durch **607D**$_h$ limitiert, die Sollposition (**6062**$_h$) darf nicht größer sein als die Grenzen in **607D**$_h$. Sollte sich der Motor beim Einrichten der Endschalter außerhalb des zulässigen Bereichs befinden, werden nur Fahrbefehle in Richtung des zulässigen Bereichs angenommen.

6.5 Zykluszeiten

Die Steuerung arbeitet mit einer Zykluszeit vom 1 ms. Das bedeutet, dass Daten jeweils alle 1 ms verarbeitet werden, mehrfache Änderungen eines Wertes (z.B. Wert eines Objektes oder Pegel an einem digitalen Eingang) innerhalb einer ms können nicht erfasst werden.

In der nachfolgenden Tabelle finden Sie eine Übersicht der Zykluszeiten der verschiedenen Prozesse.

Task	Zykluszeit
Applikation	1 ms
NanoJ Applikation	1 ms
Stromregler	31,25 µs (32 KHz)
Geschwindigkeitsregler	31,25 µs (32 KHz)
Positionsregler	31,25 µs (32 KHz)

7 Betriebsmodi

7.1 Profile Position

7.1.1 Übersicht

Beschreibung

Der *Profile Position Mode* dient dazu, Positionen relativ zur letzten Zielposition oder absolut zur letzten Referenzposition anzufahren. Während der Bewegung werden Grenzwerte für die Geschwindigkeit, Anfahr- und Bremsbeschleunigung und Rucke berücksichtigt.

Hinweis

In diesem Modus sind die Endschalter und damit die Toleranzbänder aktiv. Für weitere Informationen zu den Endschaltern, siehe **Begrenzung des Bewegungsbereichs**.

Aktivierung

Um den Modus zu aktivieren, muss im Objekt **6060**$_h$ (Modes Of Operation) der Wert "1" gesetzt werden (siehe "**CiA 402 Power State Machine**").

Controlword

Folgende Bits im Objekt **6040**$_h$ (Controlword) haben eine gesonderte Funktion:

- Bit 4 startet einen Fahrauftrag. Dieser wird bei einem Übergang von "0" nach "1" übernommen.
- Bit 5: Ist dieses Bit auf "1" gesetzt, wird ein durch Bit 4 ausgelöster Fahrauftrag sofort ausgeführt. Ist es auf "0" gesetzt, wird der gerade ausgeführte Fahrauftrag zu Ende gefahren und erst im Anschluss der nächste Fahrauftrag gestartet.
- Bit 6: Bei "0" ist die Zielposition (**607A**$_h$) absolut und bei "1" ist die Zielposition relativ. Die Referenzposition ist abhängig von den Bits 0 und 1 des Objekts **60F2**$_h$.
- Bit 8 (Halt): Ist dieses Bit auf "1" gesetzt, bleibt der Motor stehen. Bei einem Übergang von "1" auf "0" beschleunigt der Motor mit der eingestellten Startrampe bis zur Zielgeschwindigkeit. Bei einem Übergang von "0" auf "1" bremst der Motor ab und bleibt stehen. Die Bremsbeschleunigung ist dabei abhängig von der Einstellung des "Halt Option Code" im Objekt **605D**$_h$.
- Bit 9 (Change on setpoint): Ist dieses Bit gesetzt, wird die Geschwindigkeit erst beim Erreichen der ersten Zielposition geändert. Das bedeutet, dass vor Erreichen des ersten Ziels keine Bremsung durchgeführt wird, da der Motor auf dieser Position nicht stehen bleiben soll.

Controlword 6040$_h$		
Bit 9	**Bit 5**	**Definition**
X	1	Die neue Zielposition wird sofort angefahren.
0	0	Das Positionieren wird erst vollständig abgeschlossen, bevor die nächste Zielposition mit den neuen Limitierungen angefahren wird.
1	0	Die momentane Zielposition wird nur durchfahren, danach wird die neue Zielposition mit den neuen Werten angefahren.

Siehe dazu das Bild in "**Setzen von Fahrbefehlen**".

Hinweis

Das Bit 9 im Controlword wird ignoriert, wenn die Rampengeschwindigkeit im Zielpunkt unterschritten wird. In diesem Fall müsste die Steuerung zurücksetzen und Anlauf nehmen, um die Vorgabe zu erreichen.

Statusword

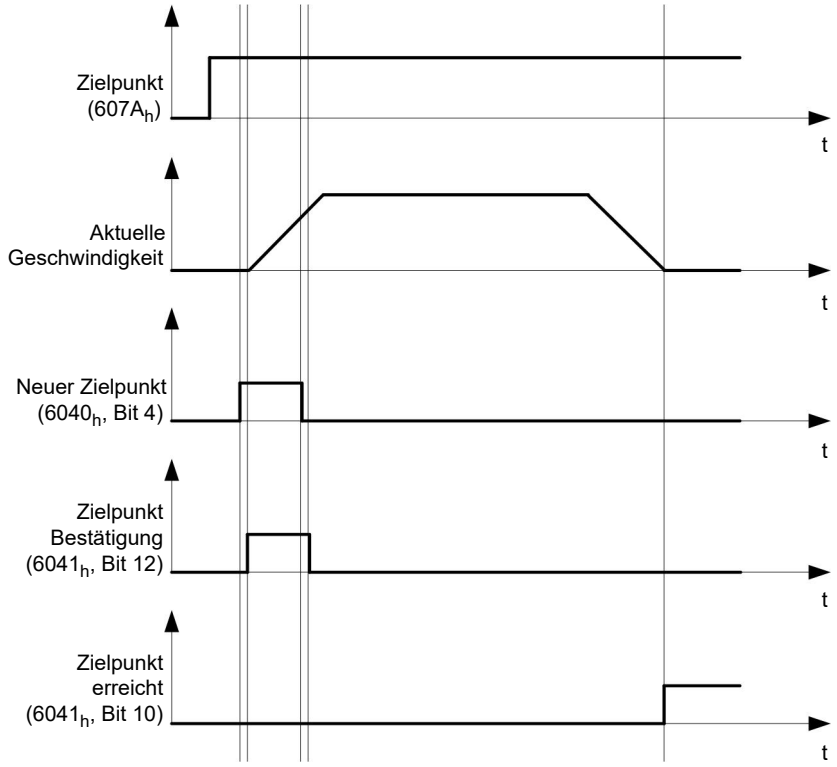
Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:

- Bit 10 (Target Reached): Dieses Bit ist auf "1" gesetzt, wenn das letzte Ziel erreicht wurde und der Motor eine vorgegebene Zeit (**6068**$_h$) innerhalb eines Toleranzfensters (**6067**$_h$) steht.
- Bit 11: Limit überschritten: Die Sollposition über- oder unterschreitet die in **607D**$_h$ eingegebenen Grenzwerte.
- Bit 12 (Set-point acknowledge): Dieses Bit bestätigt den Erhalt eines neuen und gültigen Zielpunktes. Es wird synchron zu dem Bit "New set-point" im Controlword gesetzt und zurückgesetzt.
  Eine Ausnahme besteht, wenn eine neue Fahrt gestartet wird, während eine andere noch nicht abgeschlossen ist, und die nächste Fahrt erst nach dem Abschluss der ersten Fahrt ausgeführt werden soll. In diesem Fall wird das Bit erst zurückgesetzt, wenn der Befehl angenommen wurde und die Steuerung bereit ist, neue Fahrbefehle auszuführen. Wird ein neuer Fahrauftrag gesendet, obwohl dieses Bit noch gesetzt ist, wird der neueste Fahrauftrag ignoriert.
  Das Bit wird nicht gesetzt, wenn eine der folgenden Bedingungen erfüllt ist:
  - Die neue Zielposition kann unter Einhaltung aller Randbedingungen nicht mehr erreicht werden.
  - Es wird bereits eine Zielposition angefahren und zudem ist bereits eine Zielposition vorgegeben. Eine neue Zielposition lässt sich erst vorgeben, nachdem die aktuelle Positionierung abgeschlossen ist.
- Bit 13 (Following Error): Dieses Bit wird im *Closed Loop*-Betrieb gesetzt, wenn der Schleppfehler größer als die eingestellten Grenzen ist (**6065**$_h$ (Following Error Window) und **6066**$_h$ (Following Error Time Out)).

7.1.2 Setzen von Fahrbefehlen

Fahrbefehl

In Objekt **607A**$_h$ (Target Position) wird die neue Zielposition in Benutzereinheiten angegeben (siehe "**Benutzerdefinierte Einheiten**"). Anschließend wird mit dem Setzen von Bit 4 im Objekt **6040**$_h$ (Controlword) der Fahrbefehl ausgelöst. Wenn die Zielposition gültig ist, antwortet die Steuerung mit Bit 12 im Objekt **6041**$_h$ (Statusword) und beginnt die Positionierfahrt. Sobald die Position erreicht ist, wird im Statusword das Bit 10 auf "1" gesetzt.



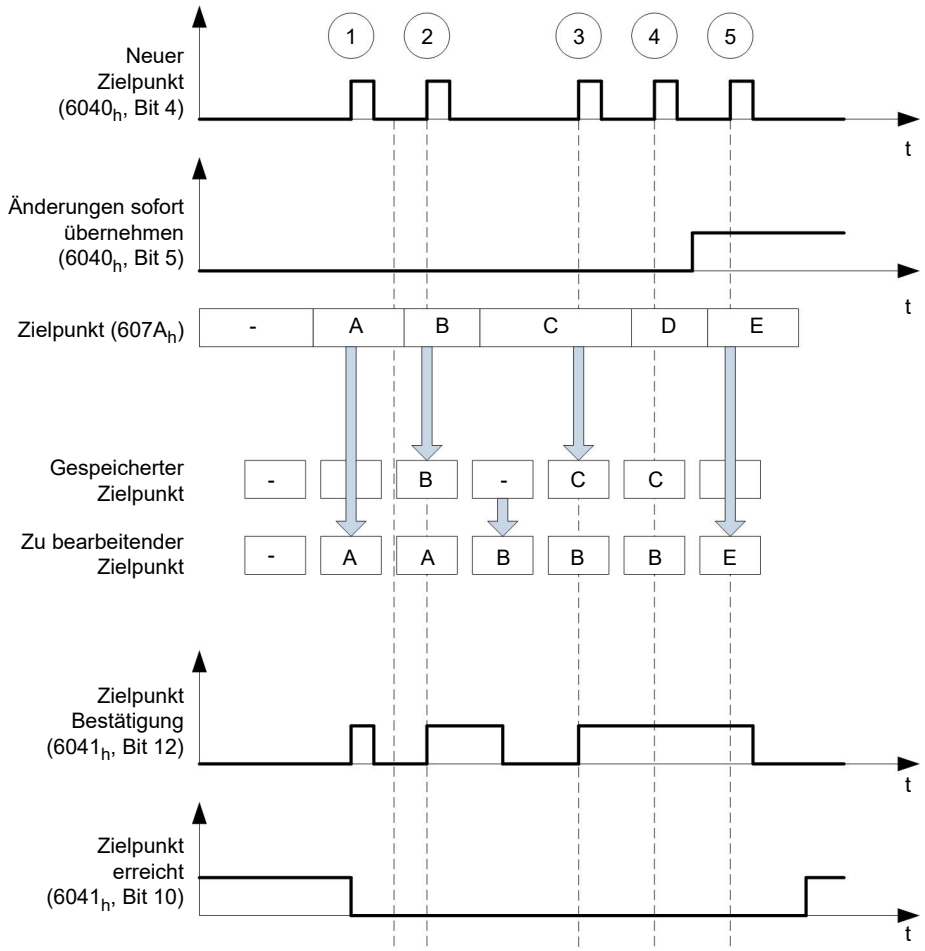
Die Steuerung kann das Bit 4 im Objekt **6040**$_h$ (Controlword) auch selbstständig zurücksetzen. Das wird mit den Bits 4 und 5 des Objektes **60F2**$_h$ eingestellt.

Weitere Fahrbefehle

Bit 12 im Objekt **6041**$_h$ (Statusword, Set-point acknowledge) fällt auf "0", falls ein weiterer Fahrbefehl zwischengespeichert werden kann (siehe Zeitpunkt 1 im nachfolgenden Bild). Solange eine Zielposition angefahren wird, lässt sich eine zweite Zielposition vorbereitend an die Steuerung übergeben. Dabei können alle Parameter - wie Geschwindigkeit, Beschleunigung, Bremsbeschleunigung usw. - neu gesetzt werden (Zeitpunkt 2). Ist der Zwischenspeicher wieder leer, lässt sich der nächste Zeitpunkt einreihen (Zeitpunkt 3).

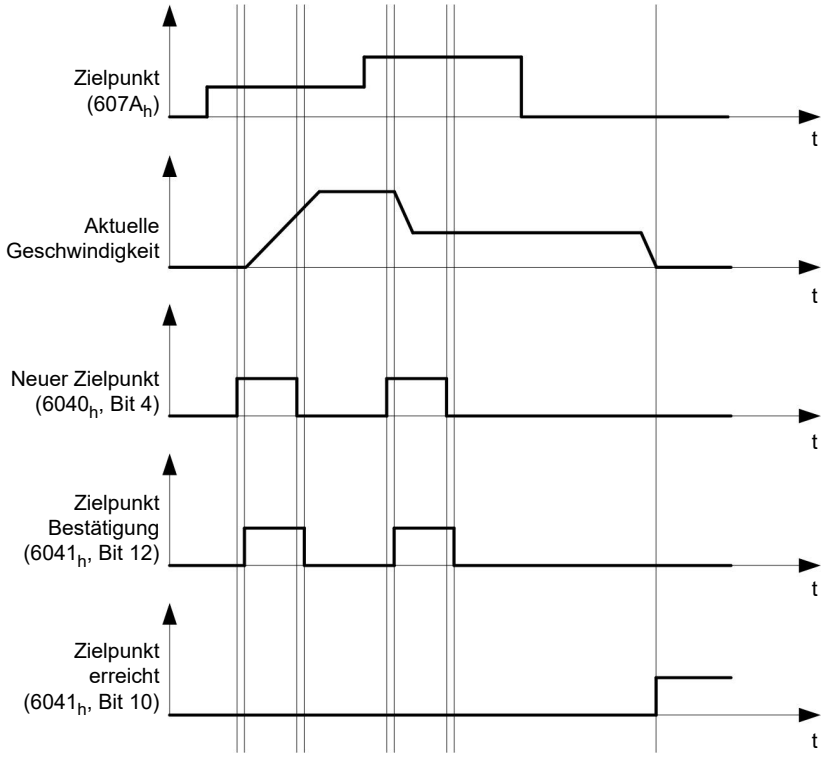
Sollte der Zwischenspeicher schon voll sein, wird ein neuer Zielpunkt ignoriert (Zeitpunkt 4). Wird Bit 5 im Objekt **6040**$_h$ (Controlword, Bit: "Change Set-Point Immediately") gesetzt, arbeitet die Steuerung ohne den Zwischenspeicher, neue Fahrbefehle werden direkt umgesetzt (Zeitpunkt 5).

Zeitpunkte



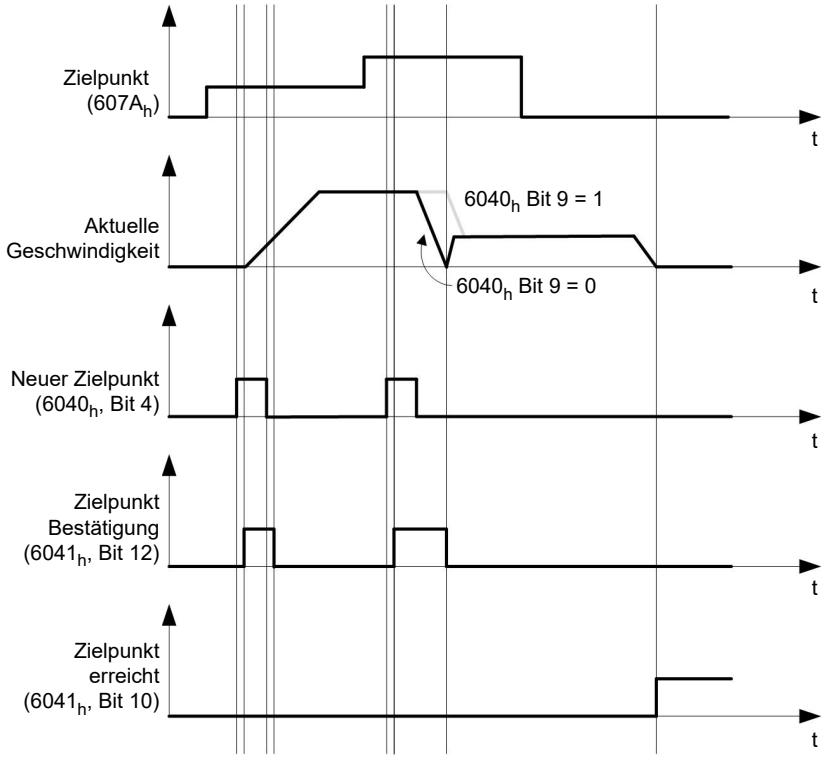
Übergangsprozedur für zweite Zielposition

Die folgende Grafik zeigt die Übergangsprozedur für die zweite Zielposition, während die erste Zielposition angefahren wird. In dieser Abbildung ist Bit 5 von Objekt **6040**$_h$ (Controlword) auf "1" gesetzt, der neue Zielwert wird demnach sofort übernommen.



Möglichkeiten zum Anfahren einer Zielposition

Ist Bit 9 in Objekt **6040**$_h$ (Controlword) gleich "0", wird die momentane Zielposition erst vollständig angefahren. In diesem Beispiel ist die Endgeschwindigkeit (**6082**$_h$) der ersten Zielposition gleich Null. Wird Bit 9 auf "1" gesetzt, wird die Profilgeschwindigkeit (**6081**$_h$) gehalten, bis die Zielposition erreicht wurde; erst ab dann gelten die neuen Randbedingungen.



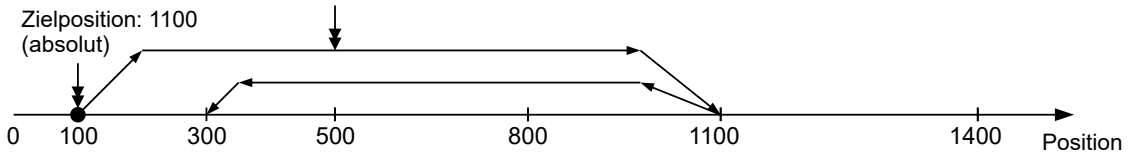
Mögliche Kombinationen von Fahrbefehlen

Um eine bessere Übersicht für die Fahrbefehle zu bekommen, werden in diesem Kapitel Kombinationen von Fahrbefehlen aufgelistet und dargestellt.

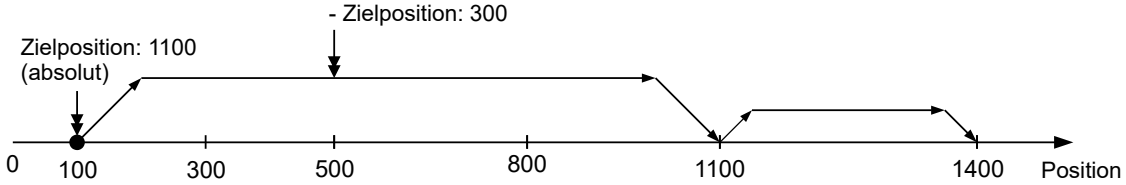
Die nachfolgenden Bilder setzen voraus:

- Ein Doppelpfeil markiert einen neuen Fahrbefehl.
- Der erste Fahrbefehl am Start ist immer ein absoluter Fahrbefehl auf die Position 1100.
- Die zweite Bewegung wird mit einer niedrigeren Geschwindigkeit durchgeführt, um einen übersichtlicher dargestellten Graphen zu erhalten.

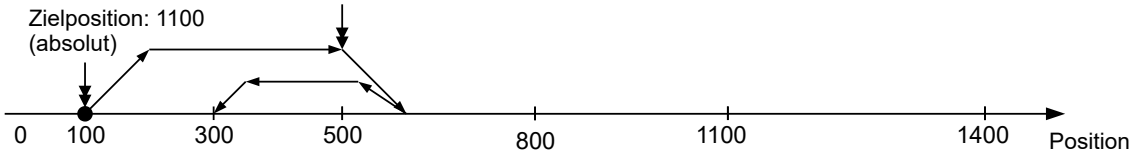
- Änderung im Zielpunkt übernehmen (6040_h:00 Bit 5 = 0)
- Positionierung absolut (6040_h:00 Bit 6 = 0)
- Zielposition: 300



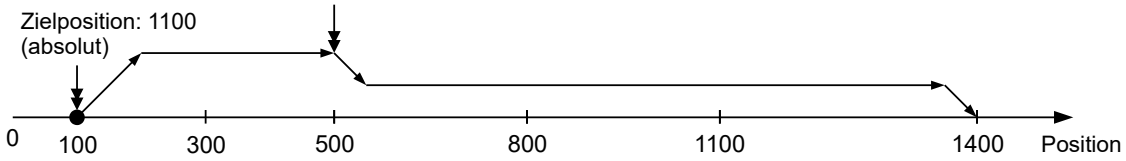
- Relativ zu der vorhergehenden Zielposition ($60F2_h$:00 = 0)
- Änderung im Zielpunkt übernehmen (6040_h:00 Bit 5 = 0)
- Positionierung relativ (6040_h:00 Bit 6 = 1)
- Zielposition: 300



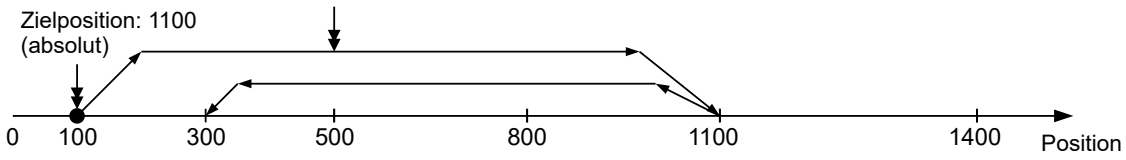
- Änderung sofort übernehmen (6040_h:00 Bit 5 = 1)
- Positionierung absolut (6040_h:00 Bit 6 = 0)
- Zielposition: 300



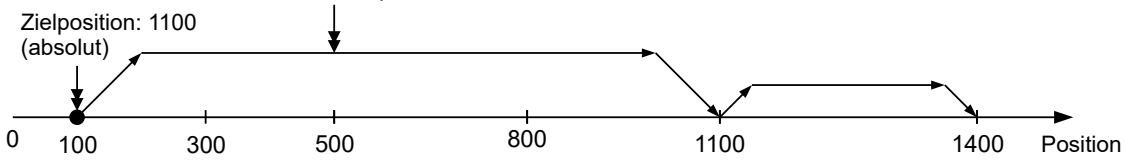
- Relativ zu der vorhergehenden Zielposition ($60F2_h$:00 = 0)
- Änderung sofort übernehmen (6040_h:00 Bit 5 = 1)
- Positionierung relativ (6040_h:00 Bit 6 = 1)
- Zielposition: 300



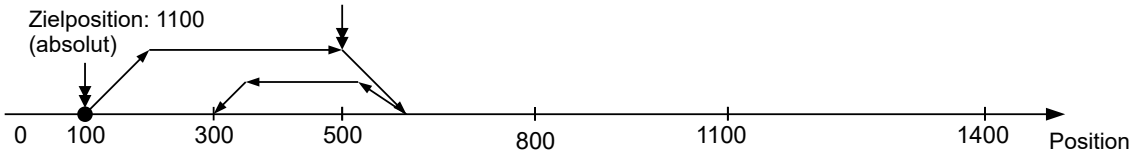
- Änderung im Zielpunkt übernehmen (6040$_h$:00 Bit 5 = 0)
- Positionierung absolut (6040$_h$:00 Bit 6 = 0)
- Zielposition: 300



- Relativ zu der aktuellen Position (60F2$_h$:00 = 1)
- Änderung im Zielpunkt übernehmen (6040$_h$:00 Bit 5 = 0)
- Positionierung relativ (6040$_h$:00 Bit 6 = 1)
- Zielposition: 300

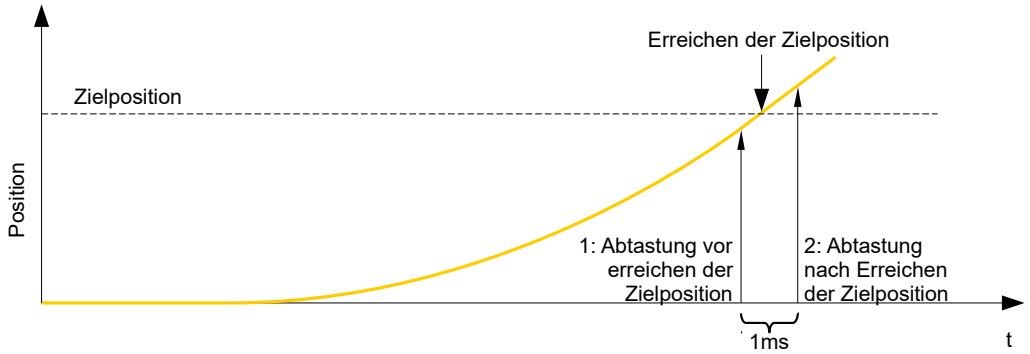


- Änderung sofort übernehmen (6040$_h$:00 Bit 5 = 1)
- Positionierung absolut (6040$_h$:00 Bit 6 = 0)
- Zielposition: 300



7.1.3 Genauigkeitsverlust bei Relativbewegungen

Beim Verketten von relativen Bewegungen kann es zu einem Verlust an Genauigkeit kommen, sollte die Endgeschwindigkeit nicht auf Null gesetzt sein. Die folgende Grafik zeigt, aus welchen Grund.



Die aktuelle Position wird einmal pro Millisekunde abgetastet. Es kann passieren, dass die Zielposition zwischen zwei Abtastungen erreicht wird. Im Falle einer Endgeschwindigkeit ungleich Null wird die Abtastung nach Erreichen der Zielposition als Grundlage für die nachfolgende Bewegung als Offset herangezogen. Demzufolge kann die nachfolgende Bewegung etwas weiter gehen, als erwartet.

7.1.4 Randbedingungen für eine Positionierfahrt

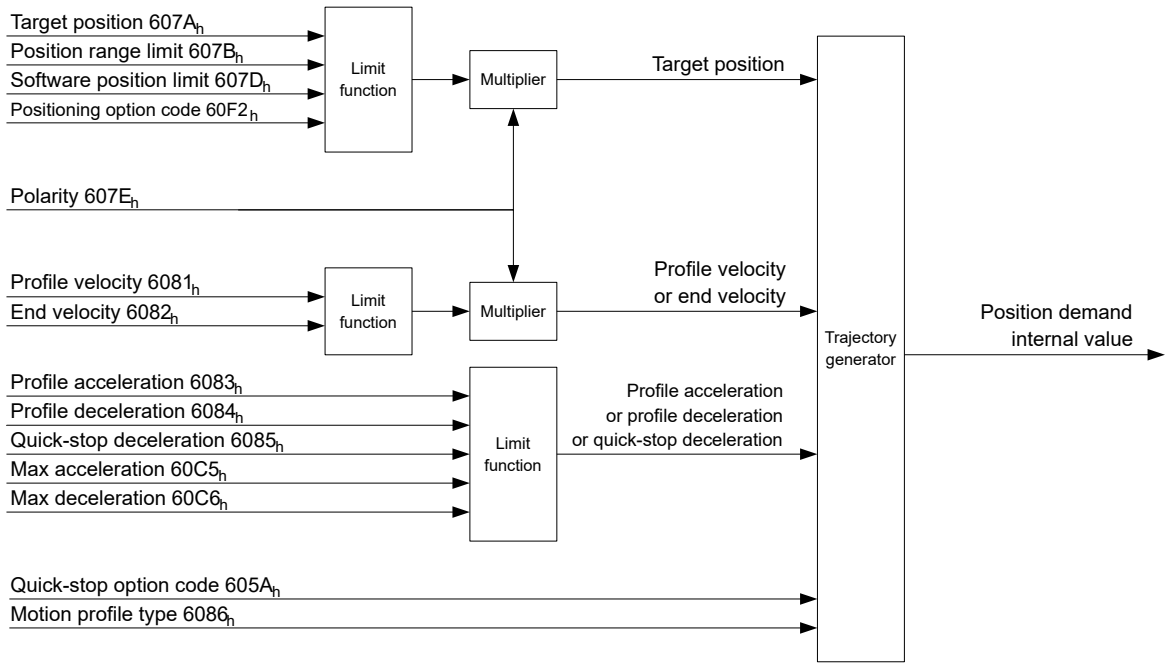
Objekteinträge

Die Randbedingungen für die gefahrene Position lassen sich in folgenden Einträgen des Objektverzeichnisses einstellen:

- **607A**$_h$ (Target Position): vorgesehene Zielposition
- **607D**$_h$ (Software Position Limit): Definition der Endanschläge (siehe Kapitel **Software-Endschalter**)
- **607C**$_h$ (Home Offset): Gibt die Differenz zwischen Null-Position der Steuerung und dem Referenzpunkt der Maschine in **benutzerdefinierten Einheiten** an. (siehe "**Homing**")
- **607B**$_h$ (Position Range Limit): Grenzen einer Modulo-Operation zur Nachbildung einer endlosen Rotationsachse
- **607E**$_h$ (Polarity): Drehrichtung
- **6081**$_h$ (Profile Velocity): maximale Geschwindigkeit, mit der die Position angefahren werden soll
- **6082**$_h$ (End Velocity): Geschwindigkeit beim Erreichen der Zielposition
- **6083**$_h$ (Profile Acceleration): gewünschte Anfahrbeschleunigung
- **6084**$_h$ (Profile deceleration): gewünschte Bremsbeschleunigung
- **6085**$_h$ (Quick Stop Deceleration): Nothalt-Bremsbeschleunigung im Falle des Zustandes "Quick stop active" der "CiA 402 Power State machine"
- **6086**$_h$ (Motion Profile Type): Typ der zu fahrenden Rampe; ist der Wert "0", wird der Ruck nicht limitiert, ist der Wert "3", werden die Werte von 60A4$_h$:1$_h$- 4$_h$ als Limitierungen des Rucks gesetzt.
- **60C5**$_h$ (Max Acceleration): die maximale Beschleunigung, die beim Anfahren der Endposition nicht überschritten werden darf
- **60C6**$_h$ (Max Deceleration): die maximale Bremsbeschleunigung, die beim Anfahren der Endposition nicht überschritten werden darf
- **60A4**$_h$ (Profile Jerk), Subindex 01$_h$ bis 04$_h$: Objekte zur Beschreibung der Grenzwerte für den Ruck.
- **60F2**$_h$ (Positioning Option Code): definiert das Positionierverhalten

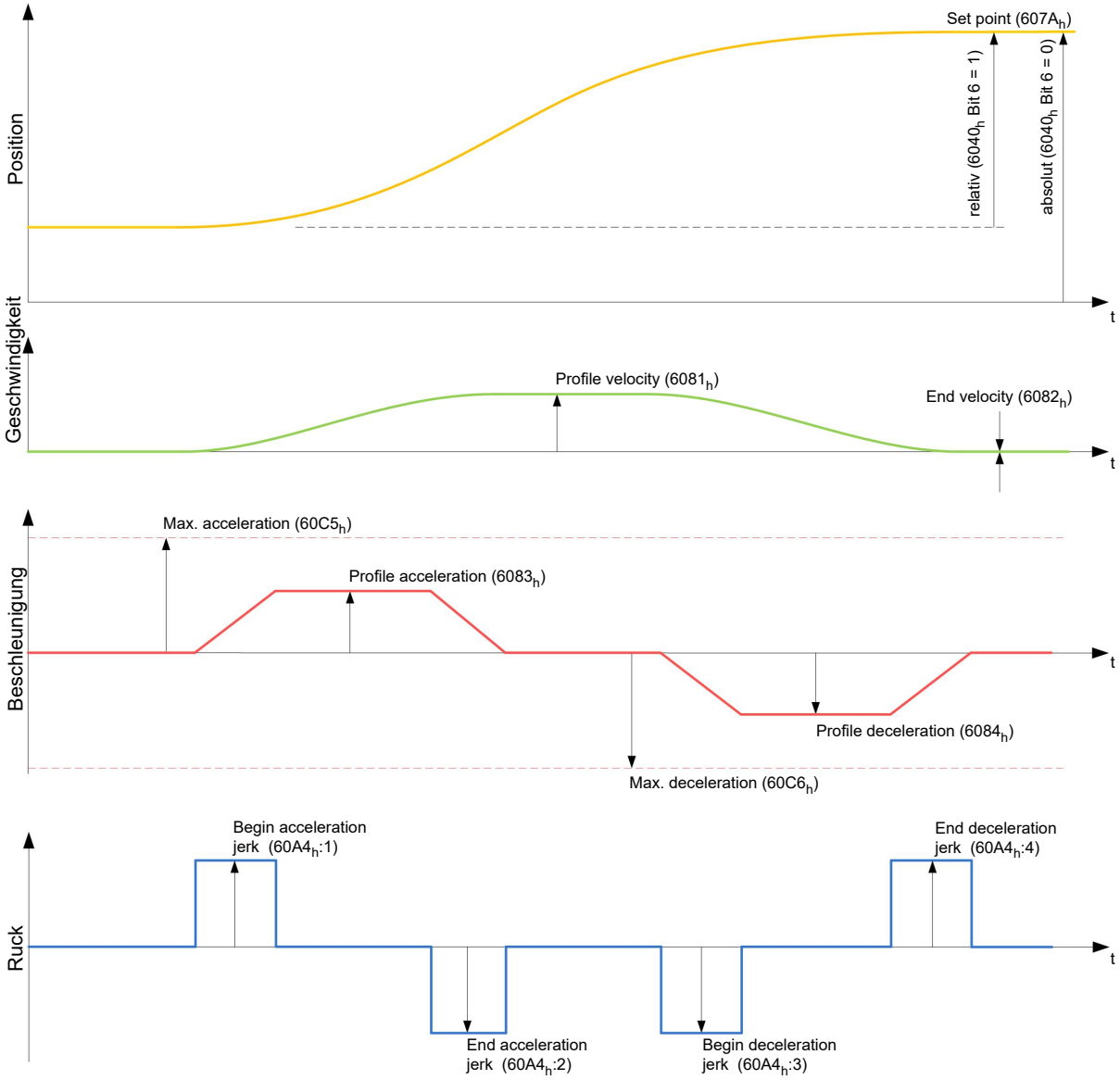
Objekte für die Positionierfahrt

Die nachfolgende Grafik zeigt die beteiligten Objekte für die Randbedingungen der Positionierfahrt.



Parameter für die Zielposition

Nachfolgende Grafik zeigt eine Übersicht über die Parameter, die für das Anfahren einer Zielposition angewendet werden (Abbildung nicht maßstabsgerecht).



7.1.5 Ruck-begrenzter und nicht ruck-begrenzter Modus

Beschreibung

Es wird grundsätzlich zwischen den Modi "ruck-begrenzt" und "nicht ruck-begrenzt" unterschieden.

Ruck-begrenzter Modus

Eine ruck-begrenzte Positionierung lässt sich erreichen, indem das Objekt **6086**$_h$ auf "3" gesetzt wird. Damit werden die Einträge für die Rucke im Subindex :1$_h$ - 4$_h$ vom Objekt **60A4** gültig.

Nicht ruck-begrenzter Modus

Eine "nicht ruck-begrenzte" Rampe wird gefahren wenn der Eintrag im Objekt **6086**$_h$ auf "0" gesetzt wird (Standard-Einstellung).

7.2 Velocity

7.2.1 Beschreibung

Dieser Modus betreibt den Motor unter Vorgabe einer Zielgeschwindigkeit ähnlich einem Frequenzumrichter. Im Gegensatz zum *Profile Velocity Mode* erlaubt dieser Modus nicht, ruckbegrenzte Rampen auszuwählen.

Hinweis

In diesem Modus sind die Endschalter und damit die Toleranzbänder aktiv. Für weitere Informationen zu den Endschaltern, siehe **Begrenzung des Bewegungsbereichs**.

7.2.2 Aktivierung

Um den Modus zu aktivieren, muss im Objekt **6060**$_h$ (Modes Of Operation) der Wert "2" gesetzt werden (siehe **CiA 402 Power State Machine**).

7.2.3 Controlword

Folgende Bits im Objekt **6040**$_h$ (Controlword) haben eine gesonderte Funktion:

- Bit 8 (Halt): Ist dieses Bit auf "1" gesetzt bleibt der Motor stehen. Bei einem Übergang von "1" auf "0" beschleunigt der Motor mit der eingestellten Beschleunigungsrampe bis zur Zielgeschwindigkeit. Bei einem Übergang von "0" auf "1" bremst der Motor entsprechend der Bremsrampe ab und bleibt stehen.

7.2.4 Statusword

Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:

- Bit 11: Limit überschritten: Die Zielgeschwindigkeit über- oder unterschreitet die eingegebenen Grenzwerte.

7.2.5 Objekteinträge

Folgende Objekte sind zur Steuerung dieses Modus erforderlich:

- **604C**$_h$ (Dimension Factor):
  Hier wird die Einheit der Geschwindigkeitsangaben für die nachfolgenden Objekte festgelegt. Werden die Subindizes 1 und 2 auf den Wert "1" eingestellt, erfolgt die Geschwindigkeitsangabe in Umdrehungen pro Minute.
  Sonst enthält der Subindex 1 den Multiplikator und der Subindex 2 den Divisor des Bruches, mit dem Geschwindigkeitsangaben in Umdrehungen pro Sekunde multipliziert werden, um auf die gewünschte Benutzereinheit zu kommen, siehe **Benutzerdefinierte Einheiten**. Über das Objekt **2060**$_h$ wird ausgewählt, ob es sich um elektrische (**2060**$_h$ = 0) oder mechanische (**2060**$_h$ = 1) Umdrehungen handelt.
- **6042**$_h$: Target Velocity.
  Hier wird die Zielgeschwindigkeit in benutzerdefinierten Einheiten eingestellt.
- **6048**$_h$: Velocity Acceleration
  Dieses Objekt definiert die Beschleunigung. Der Subindex 1 enthält dabei die Geschwindigkeitsänderung, der Subindex 2 die zugehörige Zeit in Sekunden. Beides zusammen wird als Beschleunigung verrechnet:

$$\text{VL velocity acceleration} = \frac{\text{Delta speed } (6048_h{:}1)}{\text{Delta time } (6048_h{:}2)}$$

- **6049**$_h$ (Velocity Deceleration):

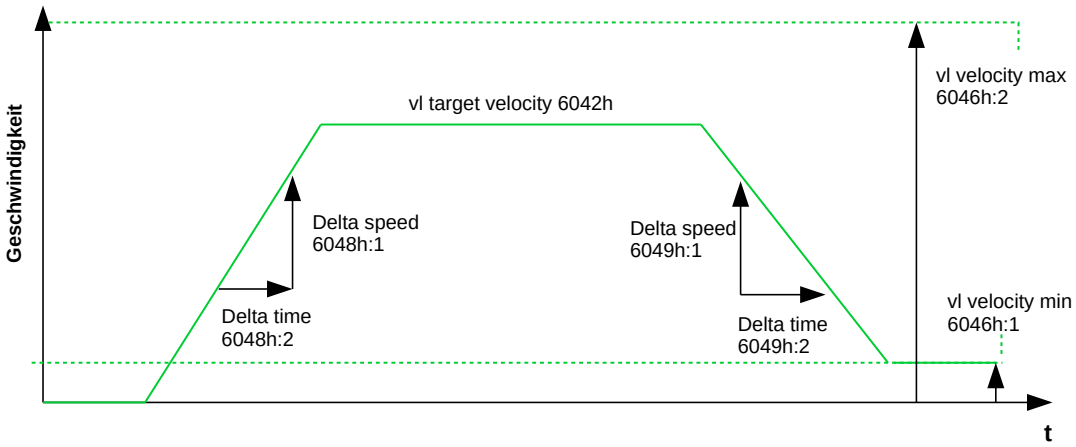
Dieses Objekt definiert die Verzögerung (Bremsrampe). Die Subindizes sind dabei so aufgebaut, wie im Objekt **6048**$_h$ beschrieben, die Geschwindigkeitsänderung ist mit positiven Vorzeichen anzugeben.

- **6046**$_h$ (Velocity Min Max Amount):
  In diesem Objekt werden die Limitierungen der Zielgeschwindigkeiten angegeben.
  In **6046**$_h$:1$_h$ wird die minimale Geschwindigkeit eingestellt. Unterschreitet die Zielgeschwindigkeit (**6042**$_h$) die Minimalgeschwindigkeit, wird der Wert auf die Minimalgeschwindigkeit **6046**$_h$:1$_h$ begrenzt.
  In **6046**$_h$:2$_h$ wird die maximale Geschwindigkeit eingestellt. Überschreitet die Zielgeschwindigkeit (**6042**$_h$) die Maximalgeschwindigkeit, wird der Wert auf die Maximalgeschwindigkeit **6046**$_h$:2$_h$ begrenzt.
- **604A**$_h$ (Velocity Quick Stop):
  Mit diesem Objekt kann die Schnellstopp-Rampe eingestellt werden. Die Subindizes 1 und 2 sind dabei identisch wie bei Objekt **6048**$_h$ beschrieben.

Folgende Objekte können zur Kontrolle der Funktion genutzt werden:

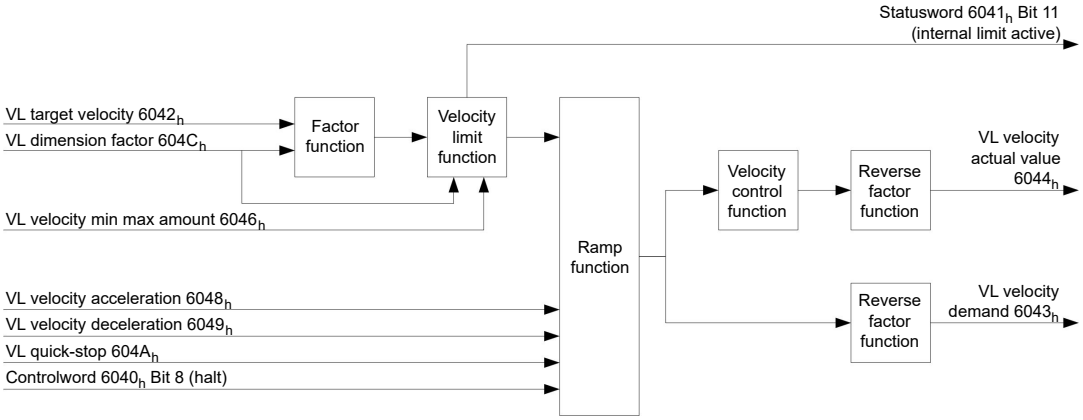
- **6043**$_h$ (Vl Velocity Demand)
- **6044**$_h$ (Vl Velocity Actual Value)

Geschwindigkeiten im Velocity Mode



Objekte für den Velocity Mode

Der Rampengenerator folgt der Zielgeschwindigkeit unter Einhaltung der eingestellten Geschwindigkeits- und Beschleunigungsgrenzen. Solange eine Begrenzung aktiv ist, wird das Bit 11 im Objekt **6041**$_h$ gesetzt (internal limit active).



7.3 Profile Velocity

7.3.1 Beschreibung

Dieser Modus betreibt den Motor im Geschwindigkeitsmodus mit erweiterten (ruck-limitierten) Rampen.

Hinweis

In diesem Modus sind die Endschalter und damit die Toleranzbänder aktiv. Für weitere Informationen zu den Endschaltern, siehe **Begrenzung des Bewegungsbereichs**.

7.3.2 Aktivierung

Um den Modus zu aktivieren, muss im Objekt **6060**$_h$ (Modes Of Operation) der Wert "3" gesetzt werden (siehe "**CiA 402 Power State Machine**").

7.3.3 Controlword

Folgende Bits im Objekt **6040**$_h$ (Controlword) haben eine gesonderte Funktion:

- Bit 8 (Halt): Ist dieses Bit auf "1" gesetzt, bleibt der Motor stehen. Bei einem Übergang von "1" auf "0" beschleunigt der Motor mit der eingestellten Startrampe bis zur Zielgeschwindigkeit. Bei einem Übergang von "0" auf "1" bremst der Motor ab und bleibt stehen.

7.3.4 Statusword

Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:

- Bit 10 (Zielgeschwindigkeit erreicht; Target Reached): Dieses Bit gibt in Kombination mit dem Bit 8 im Controlword an, ob die Zielgeschwindigkeit erreicht ist, gebremst wird oder der Motor steht (siehe Tabelle).

6041$_h$ Bit 10	**6040$_h$ Bit 8**	**Beschreibung**
0	0	Zielgeschwindigkeit nicht erreicht
0	1	Achse bremst
1	0	Zielgeschwindigkeit innerhalb Zielfenster (definiert in **606D**$_h$ und **606E**$_h$)
1	1	Geschwindigkeit der Achse ist 0

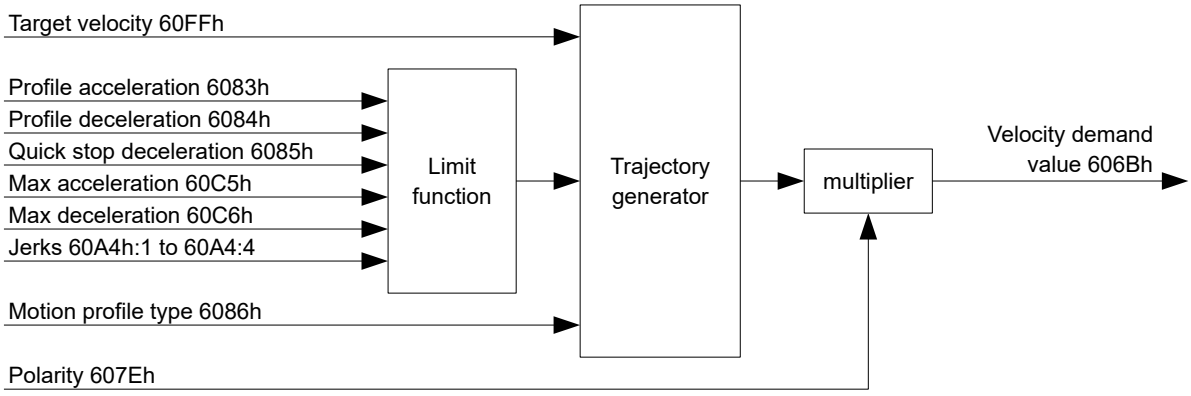
7.3.5 Objekteinträge

Folgende Objekte sind zur Steuerung dieses Modus erforderlich:

- **606B**$_h$ (Velocity Demand Value):
  Dieses Objekt enthält die Ausgabe des Rampengenerators, die gleichzeitig der Vorgabewert für den Geschwindigkeitsregler ist.
- **606C**$_h$ (Velocity Actual Value):
  Gibt die aktuelle Istgeschwindigkeit an.
- **606D**$_h$ (Velocity Window):
  Dieser Wert gibt an, wie stark die tatsächliche Geschwindigkeit von der Sollgeschwindigkeit abweichen darf, damit das Bit 10 (Zielgeschwindigkeit erreicht; Target Reached") im Objekt **6041**$_h$ (Statusword) auf "1" gesetzt ist.
- **606E**$_h$ (Velocity Window Time):
  Dieses Objekt gibt an, wie lange die reale Geschwindigkeit und die Sollgeschwindigkeit nahe beieinander liegen müssen (siehe **606D**$_h$ "Velocity Window"), damit Bit 10 "Zielgeschwindigkeit erreicht" im Objekt **6041**$_h$ (Statusword) auf "1" gesetzt wird.

- **607E**$_h$ (Polarity):
  Wird hier Bit 6 auf "1" gestellt, wird das Vorzeichen der Zielgeschwindigkeit umgekehrt.
- **6083**$_h$ (Profile acceleration):
  Setzt den Wert für die Beschleunigungsrampe im Velocity Mode.
- **6084**$_h$ (Profile Deceleration):
  Setzt den Wert für die Bremsrampe im Velocity-Mode.
- **6085**$_h$ (Quick Stop Deceleration):
  Setzt den Wert für die Bremsrampe für die Schnellbremsung im Velocity Mode.
- **6086**$_h$ (Motion Profile Type):
  Hier kann der Rampentyp ausgewählt werden ("0" = Trapez-Rampe, "3" = ruck-begrenzte Rampe).
- **60FF**$_h$ (Target Velocity):
  Gibt die zu erreichende Zielgeschwindigkeit an.

Objekte im Profile Velocity Mode

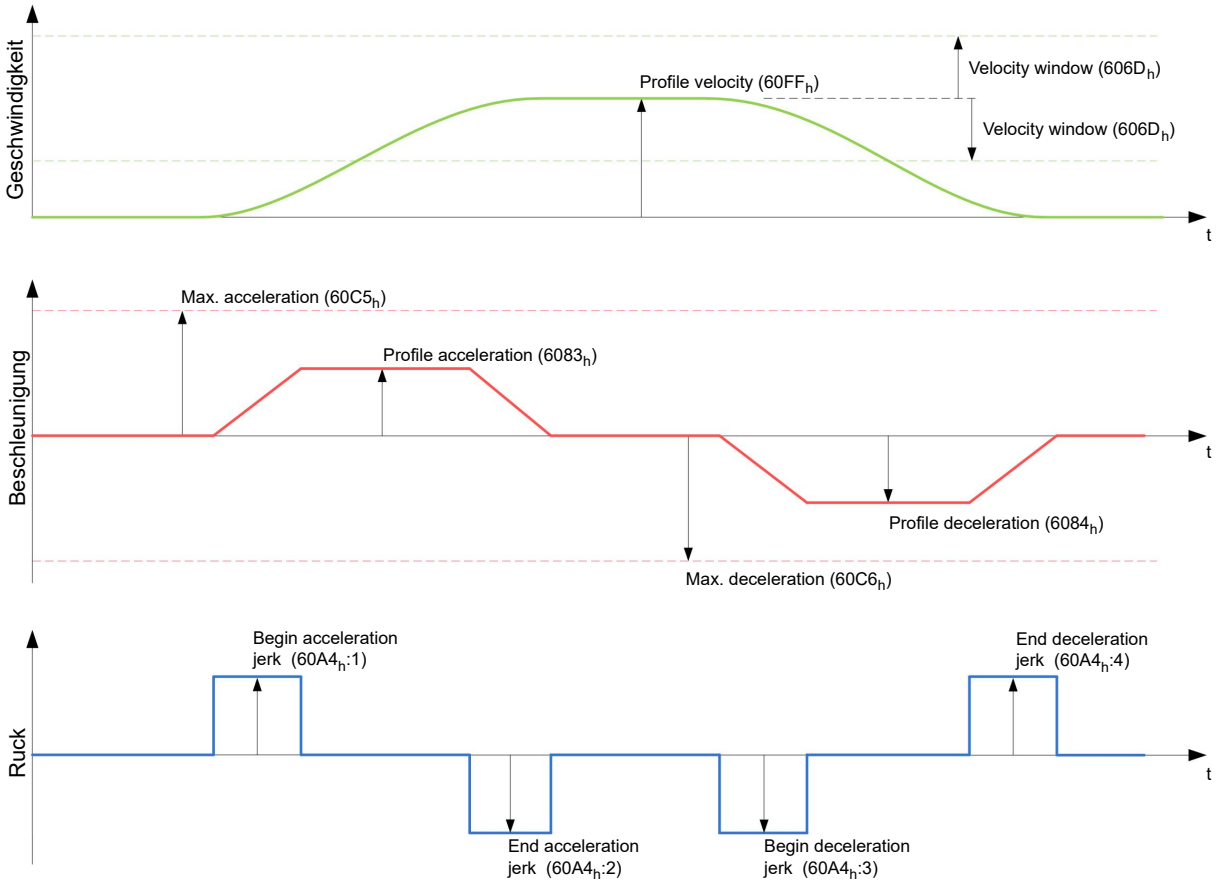


Aktivierung

Nachdem der Modus im Objekt **6060**$_h$ (Modes Of Operation) ausgewählt wurde und die "Power State machine" (siehe "**CiA 402 Power State Machine**") auf *Operation enabled* geschaltet wurde, wird der Motor auf die Zielgeschwindigkeit im Objekt **60FF**$_h$ beschleunigt (siehe nachfolgende Bilder). Dabei werden die Geschwindigkeits-, Beschleunigungs- und bei ruck-begrenzten Rampen auch die Ruckgrenzwerte berücksichtigt.

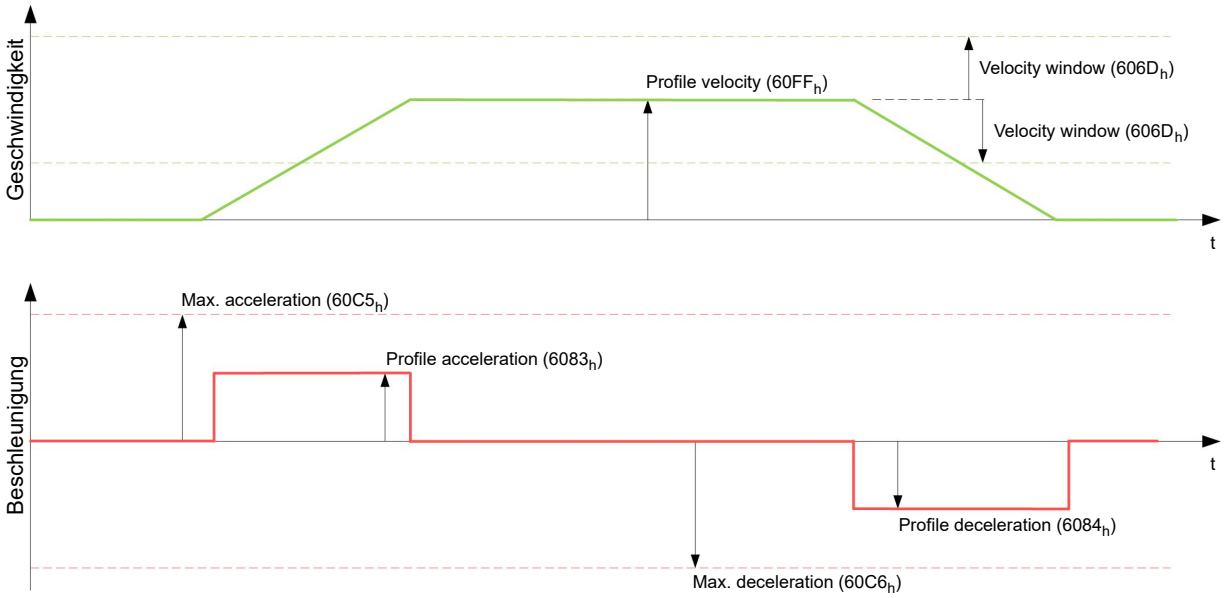
Limitierungen im ruck-limitierten Fall

Das folgende Bild zeigt die einstellbaren Limitierungen im ruck-limitierten Fall (**6086**$_h$ = 3).



Limitierungen im Trapez-Fall

Dieses Bild zeigt die einstellbaren Limitierungen für den Trapez-Fall (**6086**$_h$ = 0).



7.4 Profile Torque

7.4.1 Beschreibung

In diesem Modus wird das Drehmoment als Sollwert vorgegeben und über eine Rampenfunktion angefahren.

Hinweis

Dieser Modus funktioniert, nur wenn der **Closed Loop** aktiviert ist, siehe auch **Inbetriebnahme Closed Loop**.

Hinweis

In diesem Modus sind die Endschalter und damit die Toleranzbänder aktiv. Für weitere Informationen zu den Endschaltern, siehe **Begrenzung des Bewegungsbereichs**.

7.4.2 Aktivierung

Um den Modus zu aktivieren, muss im Objekt **6060**$_h$ (Modes Of Operation) der Wert "4" gesetzt werden (siehe "**CiA 402 Power State Machine**").

7.4.3 Controlword

Folgende Bits im Objekt **6040**$_h$ (Controlword) haben eine gesonderte Funktion:

- Bit 8 (Halt): Ist dieses Bit auf "1" gesetzt, bleibt der Motor stehen. Wird dieses Bit von "1" auf "0" gesetzt, wird der Motor den Vorgaben entsprechend angefahren. Beim Setzen von "0" auf "1" wird der Motor unter Berücksichtigung der Vorgabewerte wieder zum Stillstand gebracht.

7.4.4 Statusword

Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:

- Bit 10 (Target Reached): Dieses Bit gibt in Kombination mit dem Bit 8 des Objekts **6040**$_h$ (Controlword) an, ob das vorgegebene Drehmoment erreicht ist (siehe nachfolgende Tabelle). Das Ziel gilt als erreicht wenn das Istdrehmoment (**6077h Torque Actual Value**) eine vorgegebene Zeit (**203Eh Torque Window Time**) innerhalb eines Toleranzfensters (**203Dh Torque Window**) ist.

6040$_h$ Bit 8	**6041$_h$ Bit 10**	**Beschreibung**
0	0	Vorgegebenes Drehmoment nicht erreicht
0	1	Vorgegebenes Drehmoment erreicht
1	0	Achse beschleunigt
1	1	Geschwindigkeit der Achse ist 0

- Bit 11: Limit überschritten: Das Zieldrehmoment (**6071**$_h$) überschreitet das in **6072**$_h$ eingegebene maximalen Drehmoment.

7.4.5 Objekteinträge

Alle Werte der folgenden Einträge im Objektverzeichnis sind als Tausendstel des maximalen Drehmoments anzugeben, welches dem Nennstrom (**203B**$_h$:01$_h$) entspricht. Dazu zählen die Objekte:

- **6071**$_h$ (Target Torque):
  Zielvorgabe des Drehmomentes
- **6072**$_h$ (Max Torque):
  Maximales Drehmoment während der gesamten Rampe (Beschleunigen, Drehmoment halten, Abbremsen)
- **6074**$_h$ (Torque Demand):
  Momentaner Ausgabewert des Rampengenerators (Drehmoment) für den Regler
- **6087**$_h$ (Torque Slope):
  Max. Änderung des Drehmoments pro Sekunde

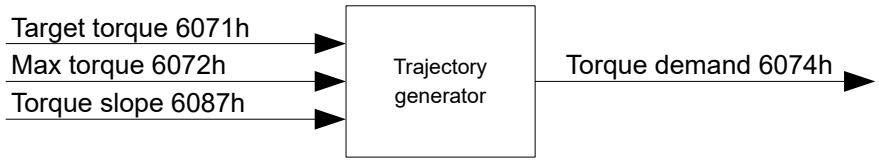
Hinweis

Diese Werte sind nicht limitiert auf 100% des Nennstroms (**203B**$_h$:01$_h$). Drehmomentwerte höher als das Nenndrehmoment (generiert von dem Nennstrom) können erreicht werden, wenn die Maximaldauer des Spitzenstroms (**203B**$_h$:02$_h$) gesetzt wird (siehe **I2t Motor-Überlastungsschutz**). Alle Drehmoment-Objekte werden von dem Spitzenstrom limitiert.

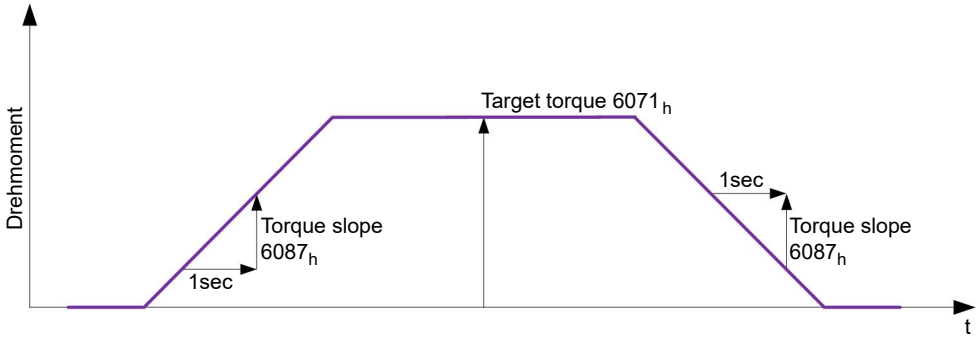
Die folgenden Objekte werden zudem für diesen Operationsmodus benötigt:

- **3202**$_h$ Bit 5 (Motor Drive Submode Select):
  Ist dieses Bit auf "0" gesetzt, wird der Antriebsregler im Drehmoment-begrenzten Velocity Mode betrieben, d.h. die maximale Geschwindigkeit kann in Objekt **2032**$_h$ begrenzt werden und der Regler kann im Feldschwächebetrieb arbeiten.
  Wird dieses Bit auf "1" gesetzt, arbeitet der Regler im ("Real") Torque Mode, die maximale Geschwindigkeit kann hier nicht begrenzt werden und der Feldschwächebetrieb ist nicht möglich.

Objekte des Rampengenerators



Torque-Verlauf



7.5 Homing

7.5.1 Übersicht

Beschreibung

Aufgabe der Referenzfahrt (Homing Method) ist es, den Positionsnullpunkt der Steuerung auf einen Encoder-Index bzw. Positionsschalter auszurichten.

Aktivierung

Um den Modus zu aktivieren, muss im Objekt **6060**$_h$ (Modes Of Operation) der Wert "6" gesetzt werden (siehe "**CiA 402 Power State Machine**").

Werden Referenz- und/oder Endschalter verwendet, müssen diese Spezialfunktionen erst in der E/A-Konfiguration aktiviert werden (siehe "**Digitale Ein- und Ausgänge**").

Controlword

Folgende Bits im Objekt **6040**$_h$ (Controlword) haben eine gesonderte Funktion:

- Bit 4: Wird das Bit auf "1" gesetzt, wird die Referenzierung gestartet. Diese wird solange ausgeführt, bis entweder die Referenzposition erreicht wurde oder Bit 4 wieder auf "0" gesetzt wird.

Statusword

Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:

Bit 13	Bit 12	Bit 10	Beschreibung
0	0	0	Referenzfahrt wird ausgeführt
0	0	1	Referenzfahrt ist unterbrochen oder nicht gestartet
0	1	0	Referenzfahrt bestätigt, aber Ziel wurde noch nicht erreicht
0	1	1	Referenzfahrt vollständig abgeschlossen
1	0	0	Fehler während der Referenzfahrt, Motor dreht sich noch
1	0	1	Fehler während der Referenzfahrt, Motor im Stillstand

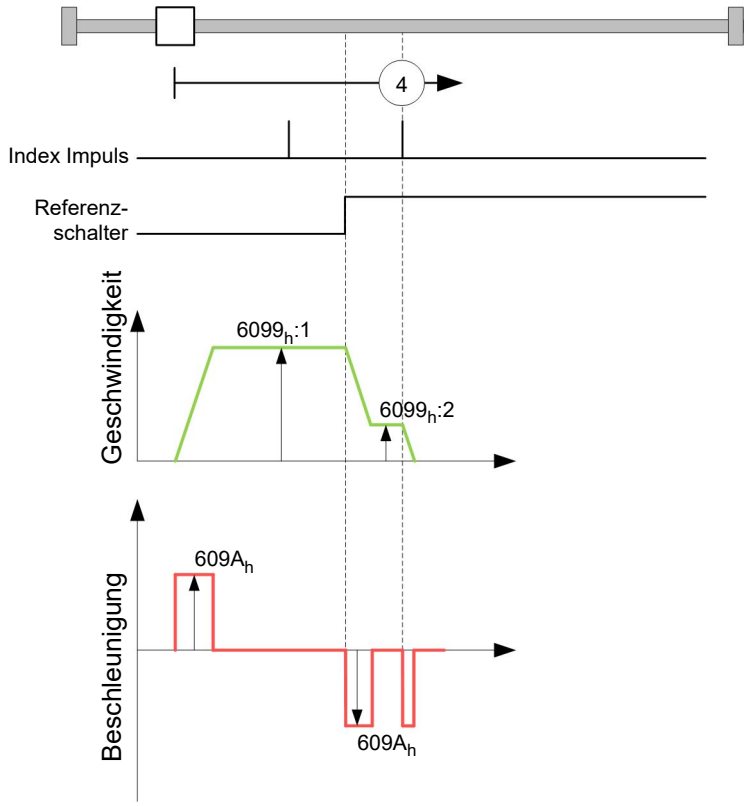
Objekteinträge

Folgende Objekte sind zur Steuerung dieses Modus erforderlich:

- **607C**$_h$ (Home Offset): Gibt die Differenz zwischen Null-Position der Steuerung und dem Referenzpunkt der Maschine in **benutzerdefinierten Einheiten** an.
- **6098**$_h$ (Homing Method):
  Methode, mit der referenziert werden soll (siehe "**Referenzfahrt-Methode**")
- **6099**$_h$:01$_h$ (Speed During Search For Switch):
  Geschwindigkeit für die Suche nach dem Schalter
- **6099**$_h$:02$_h$ (Speed During Search For Zero):
  Geschwindigkeit für die Suche nach dem Index
- **609A**$_h$ (Homing Acceleration):
  Anfahr- und Bremsbeschleunigung für die Referenzfahrt
- **2056**$_h$ (Limit Switch Tolerance Band):
  Die Steuerung lässt nach dem Auffahren auf den positiven oder negativen Endschalter einen Toleranzbereich zu, den der Motor noch zusätzlich weiter fahren darf. Wird dieser Toleranzbereich überschritten, stoppt der Motor und die Steuerung wechselt in den Zustand "Fault". Falls während der Referenzfahrt Endschalter betätigt werden können, sollte der Toleranzbereich ausreichend gewählt werden, so dass der Motor beim Abbremsen den Toleranzbereich nicht verlässt. Andernfalls kann die Referenzfahrt nicht erfolgreich ausgeführt werden. Nach Abschluss der Referenzfahrt kann der Toleranzbereich, wenn dies die Anwendung erfordert, wieder auf "0" gesetzt werden.
- **203A**$_h$:01$_h$ (Minimum Current For Block Detection):
  Minimale Stromschwelle, durch deren Überschreiten, das Blockieren des Motors an einem Block erkannt werden soll.
- **203A**$_h$:02$_h$ (Period Of Blocking):
  Gibt die Zeit in ms an, die der Motor nach der Blockdetektion trotzdem noch gegen den Block fahren soll.

Geschwindigkeiten der Referenzfahrt

Das Bild zeigt die Geschwindigkeiten der Referenzfahrt am Beispiel der Methode 4:



7.5.2 Referenzfahrt-Methode

Beschreibung

Die Referenzfahrt-Methode wird als Zahl in das Objekt **6098**$_h$ geschrieben und entscheidet darüber, ob auf eine Schalterflanke (steigend/fallend), eine Stromschwelle für Blockdetektion bzw. einen Index-Impuls referenziert wird oder in welche Richtung die Referenzfahrt startet. Methoden, die den Index-Impuls des Encoders benutzen, liegen im Zahlenbereich 1 bis 14, 33 und 34. Methoden, die den Index-Impuls des Encoders nicht benutzen, liegen zwischen 17 und 30, sind in den Fahrprofilen aber identisch mit den Methoden 1 bis 14. Diese Zahlen sind in den nachfolgenden Abbildungen eingekreist dargestellt. Methoden, bei denen keine Endschalter eingesetzt werden und stattdessen das Fahren gegen einen Block erkannt werden soll, müssen mit einem Minus vor der Methodenzahl aufgerufen werden.

Für die nachfolgenden Grafiken gilt die negative Bewegungsrichtung nach links. Der Endschalter (*limit switch*) liegt jeweils vor der mechanischen Blockierung, der Referenzschalter (*home switch*) liegt zwischen den beiden Endschaltern. Die Index-Impulse kommen vom angeschlossenen Encoder.

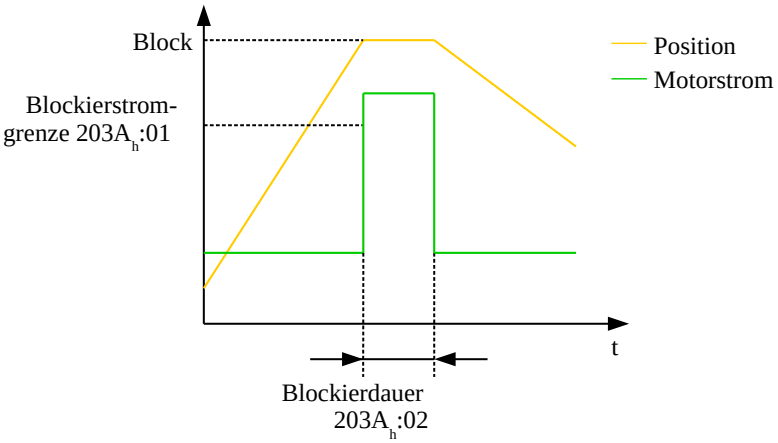
Bei Methoden, die Homing auf Block benutzen, gelten die gleichen Abbildungen wie für die Methoden mit Endschalter. Da sich außer den fehlenden Endschaltern nichts ändert, wurde auf neue Abbildungen verzichtet. Hier gilt für die Abbildungen, dass die Endschalter durch eine mechanische Blockierung ersetzt werden müssen.

Homing auf Block

Homing auf Block funktioniert derzeit nur im *Closed Loop*-Betrieb.

"Homing auf Block" funktioniert wie jede Homing-Methode mit dem Unterschied, dass zur Positionierung - anstelle auf einen Endschalter - auf einen Block (Endanschlag) gefahren wird. Dabei sind zwei Einstellungen vorzunehmen:

1. Stromhöhe: im Objekt **203A**$_h$:01 wird die Stomhöhe definiert, ab der ein Fahren gegen den Block erkannt wird.
2. Blockierdauer: im Objekt **203A**$_h$:02 wird die Dauer, während der Motor gegen den Block fährt, eingestellt.



Methoden-Überblick

Die Methoden 1 bis 14, sowie 33 und 34 benutzen den Index-Impuls des Encoders.

Die Methoden 17 bis 32 sind identisch mit den Methoden 1 bis 14, mit dem Unterschied, dass nur noch auf den End- oder Referenzschalter referenziert wird und nicht auf den Index-Impuls.

- Methoden 1 bis 14 verwenden einen Index-Impuls.
- Methoden 17 bis 30 verwenden keinen Index-Impuls.
- Methoden 33 und 34 referenzieren nur auf den nächsten Index-Impuls.
- Methode 35 referenziert auf die aktuelle Position.

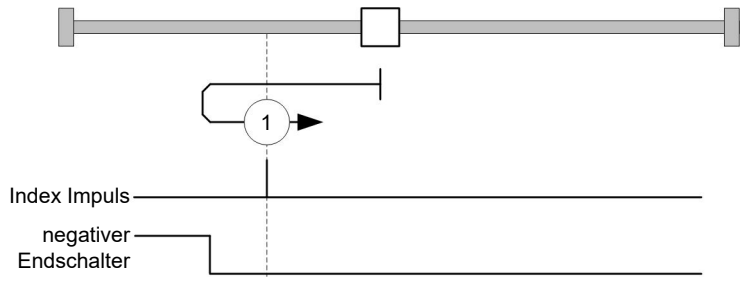
Folgende Methoden können für Homing auf Block benutzt werden:

- Methoden -1 bis -2 und -7 bis -14 enthalten einen Index-Impuls
- Methoden -17 bis -18 und -23 bis -30 haben keinen Index-Impuls

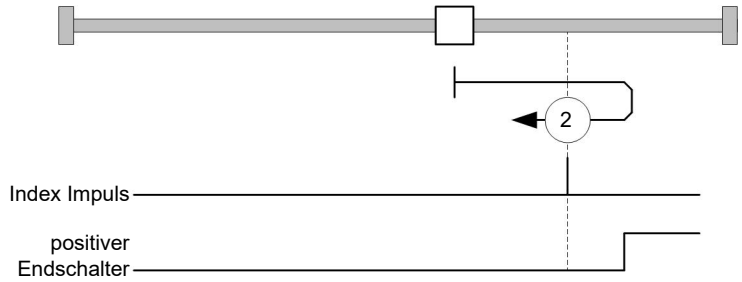
Methoden 1 und 2

Referenzieren auf Endschalter und Index-Impuls.

Methode 1 referenziert auf negativen Endschalter und Index-Impuls:



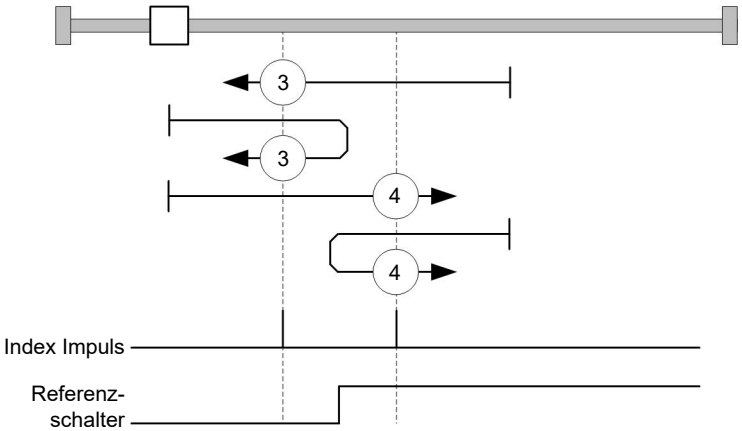
Methode 2 referenziert auf positiven Endschalter und Index-Impuls:



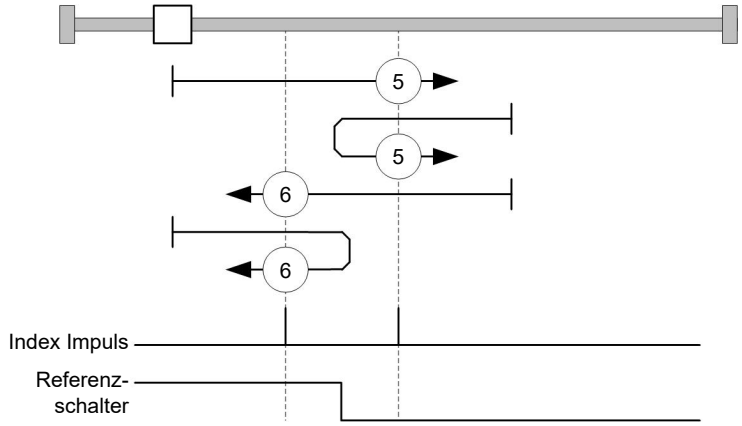
Methoden 3 bis 6

Referenzieren auf die Schaltflanke des Referenzschalters und Index-Impuls.

Bei den Methoden 3 und 4 wird die linke Schaltflanke des Referenzschalters als Referenz verwendet:



Bei den Methoden 5 und 6 wird die rechte Schaltflanke des Referenzschalters als Referenz verwendet:

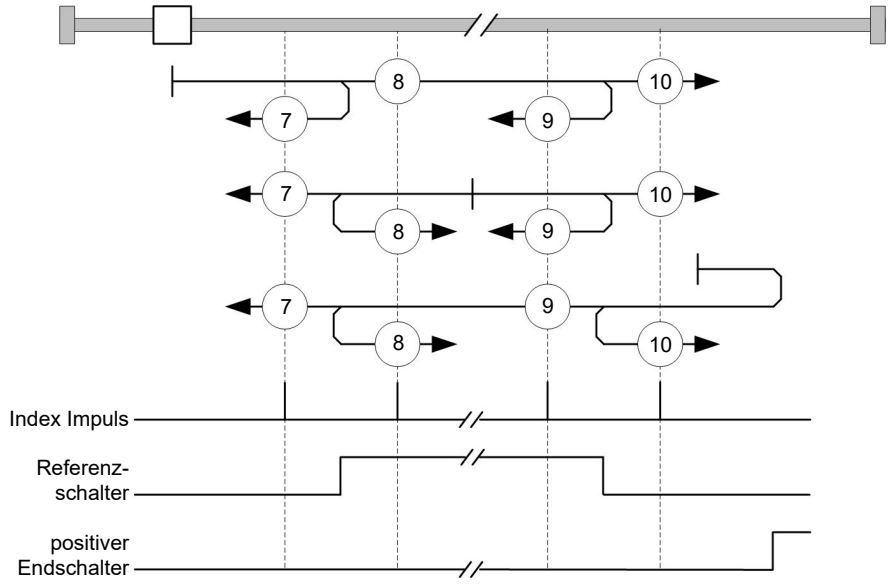


Methoden 7 bis 14

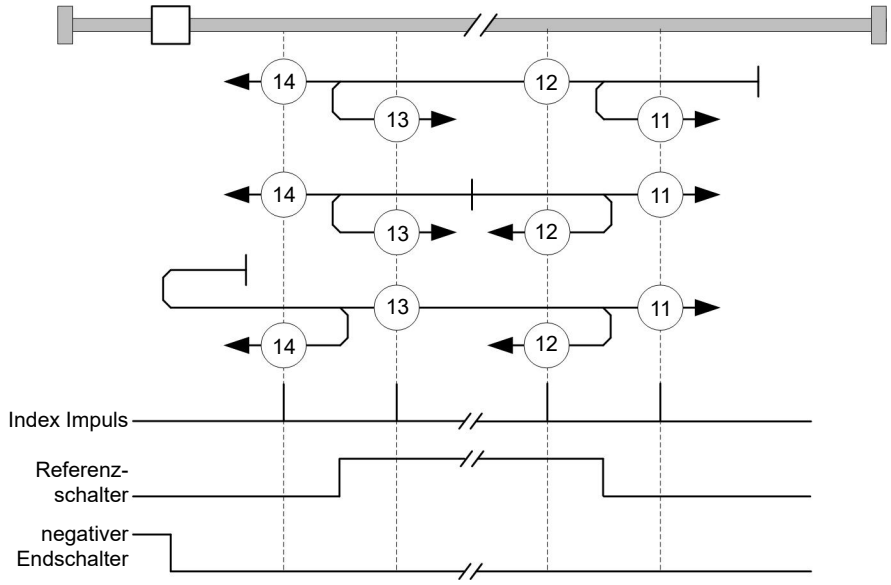
Referenzieren auf Referenzschalter und Index-Impuls (mit Endschaltern).

Bei diesen Methoden ist die derzeitige Position relativ zum Referenzschalter unwichtig. Mit der Methode 10 wird beispielsweise immer auf den Index-Impuls rechts neben der rechten Flanke des Referenzschalters referenziert.

Die Methoden 7 bis 10 berücksichtigen den positiven Endschalter:



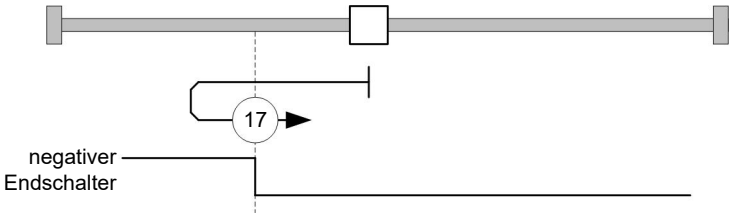
Die Methoden 11 bis 14 berücksichtigen den negativen Endschalter:



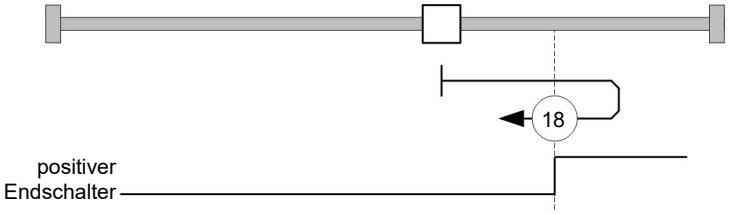
Methoden 17 und 18

Referenzieren auf den Endschalter ohne den Index-Impuls.

Methode 17 referenziert auf den negativen Endschalter:



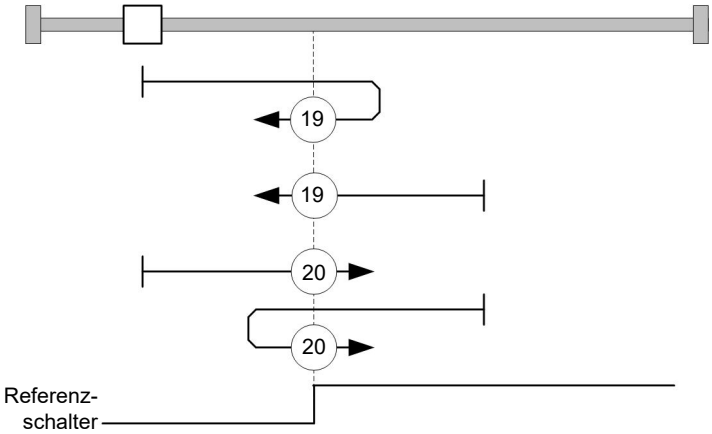
Methode 18 referenziert auf den positiven Endschalter:



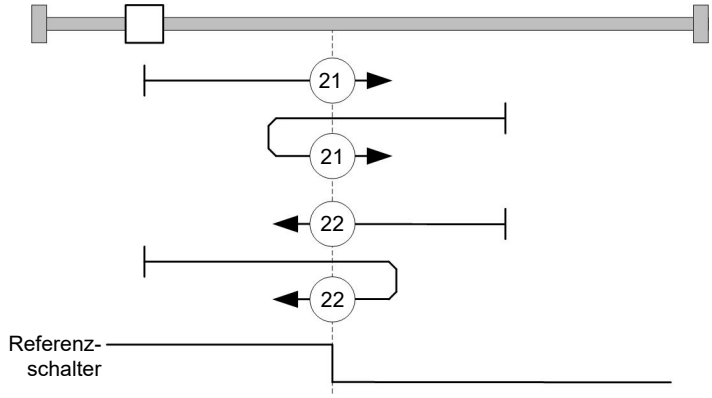
Methoden 19 bis 22

Referenzieren auf die Schaltflanke des Referenzschalters ohne den Index-Impuls.

Bei den Methoden 19 und 20 (äquivalent zu Methoden 3 und 4) wird die linke Schaltflanke des Referenzschalters als Referenz verwendet:



Bei den Methoden 21 und 22 (äquivalent zu Methoden 5 und 6) wird die rechte Schaltflanke des Referenzschalters als Referenz verwendet:

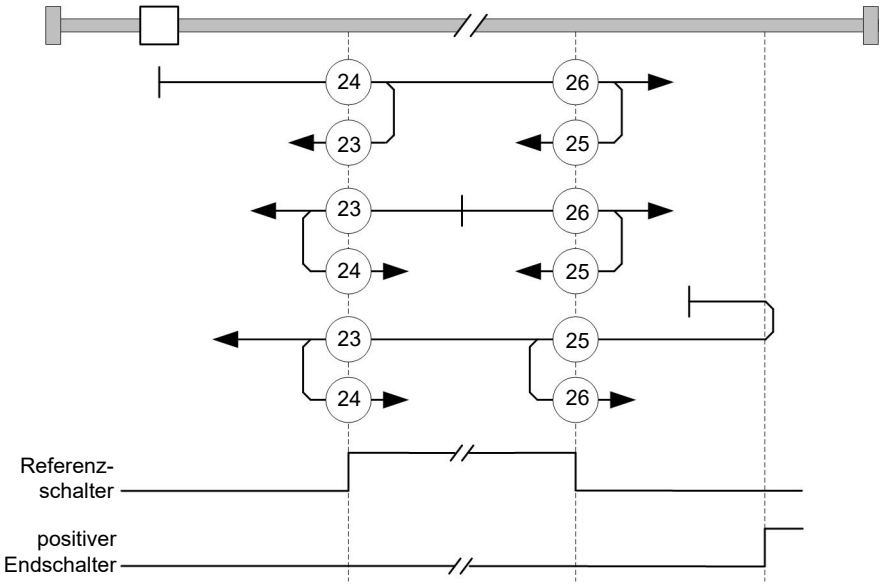


Methoden 23 bis 30

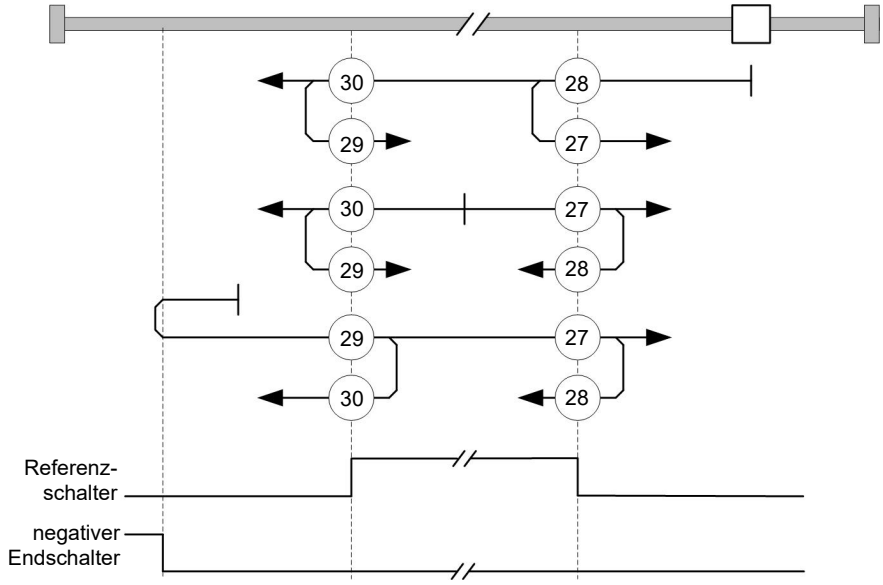
Referenzieren auf Referenzschalter ohne den Index-Impuls (mit Endschaltern).

Bei diesen Methoden ist die derzeitige Position relativ zum Referenzschalter unwichtig. Mit der Methode 26 wird beispielsweise immer auf den Index-Impuls rechts neben der rechten Flanke des Referenzschalters referenziert.

Die Methoden 23 bis 26 berücksichtigen den positiven Referenzschalter:



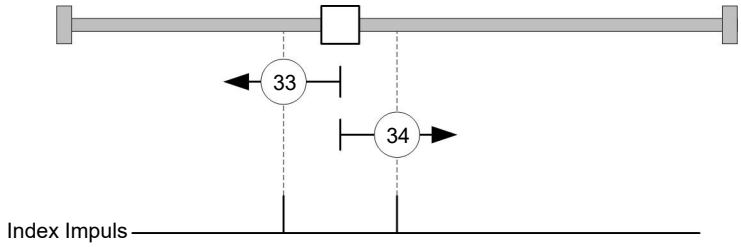
Die Methoden 27 bis 30 berücksichtigen den negativen Referenzschalter:



Methoden 33 und 34

Referenzieren auf den nächsten Index-Impuls.

Bei diesen Methoden wird nur auf den jeweils folgenden Index-Impuls referenziert:



Methode 35

Referenziert auf die aktuelle Position.

Hinweis

Für den Homing Mode 35 ist es nicht notwendig, die **CiA 402 Power State Machine** in den Status "Operation Enabled" zu schalten. Auf diese Weise kann vermieden werden, dass durch eine Bestromung der Motorwicklungen im *Open Loop*-Betrieb, die aktuelle Position nach dem Homing Mode 35 nicht genau 0 ist.

7.6 Interpolated Position Mode

7.6.1 Übersicht

Beschreibung

Der *Interpolated Position Mode* dient zum Synchronisieren mehrerer Achsen. Hierzu übernimmt eine übergeordnete Steuerung die Rampen- bzw. Bahnberechnung und überträgt die jeweilige Sollposition, bei der sich die Achse zu einem bestimmten Zeitpunkt befinden soll, zur Steuerung. Zwischen diesen Positions-Stützstellen interpoliert die Steuerung.

Hinweis

In diesem Modus sind die Endschalter und damit die Toleranzbänder aktiv. Für weitere Informationen zu den Endschaltern, siehe **Begrenzung des Bewegungsbereichs**.

Synchronisierung zum SYNC-Objekt

Für den Interpolated Position Mode ist es notwendig, dass sich die Steuerung auf das SYNC-Objekt (abhängig vom Feldbus) aufsynchronisiert. Dieses SYNC-Objekt ist in regelmäßigen Zeitabständen von der übergeordneten Steuerung zu senden. Die Synchronisation erfolgt, sobald die Steuerung in den NMT-Modus *Operational* geschaltet wird.

Hinweis

Es wird empfohlen, wenn möglich ein Zeitintervall des *SYNC-Objekts* zu nutzen.

7.6.2 Aktivierung

Um den Modus zu aktivieren, muss im Objekt **6060**$_h$ (Modes Of Operation) der Wert "7" gesetzt werden (siehe "**CiA 402 Power State Machine**").

7.6.3 Controlword

Folgende Bits im Objekt **6040**$_h$ (Controlword) haben eine gesonderte Funktion:

- Bit 4 aktiviert die Interpolation, wenn es auf "1" gesetzt wird.
- Bit 8 (Halt): Ist dieses Bit auf "1" gesetzt, bleibt der Motor stehen. Bei einem Übergang von "1" auf "0" beschleunigt der Motor mit der eingestellten Startrampe bis zur Zielgeschwindigkeit. Bei einem Übergang von "0" auf "1" bremst der Motor ab und bleibt stehen. Die Bremsbeschleunigung ist dabei abhängig von der Einstellung des "Halt Option Code" im Objekt **605D**$_h$.

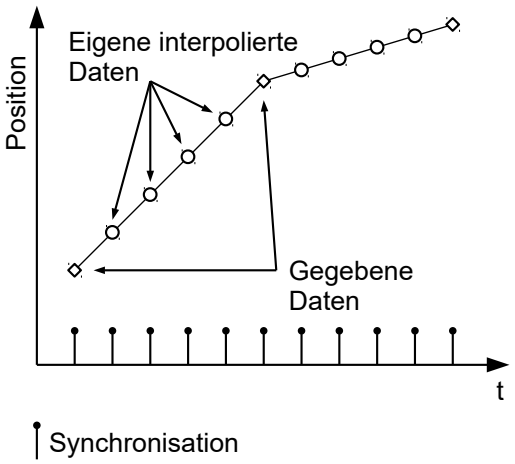
7.6.4 Statusword

Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:

- Bit 10: Zielposition erreicht: Dieses Bit ist auf "1" gesetzt, wenn die Zielposition erreicht wurde (sollte das Halt-Bit im Controlword "0" sein) oder die Achse hat die Geschwindigkeit 0 (falls das Halt-Bit im letzten Controlword "1" war).
- Bit 11: Limit überschritten: Die Sollposition über- oder unterschreitet die in **607D**$_h$ eingegebenen Grenzwerte.
- Bit 12 (IP Modus aktiv): Dieses Bit wird auf "1" gesetzt, wenn die Interpolation aktiv ist.

7.6.5 Benutzung

Die Steuerung folgt einem linear interpolierten Pfad zwischen der aktuellen und der vorgegebenen Zielposition. Die (nächste) Zielposition muss in das Datensatz **60C1**$_h$:01$_h$ geschrieben werden.



In der derzeitigen Implementation wird nur

- lineare Interpolation
- und eine Zielposition

unterstützt.

7.6.6 Setup

Das folgende Setup ist nötig:

- **60C2**$_h$:01$_h$: Zeit zwischen zwei übergebenen Zielpositionen in ms.
- **60C4**$_h$:06$_h$: dieses Objekt ist auf "1" zu setzen um die Zielposition im Objekt **60C1**$_h$:01$_h$ modifizieren zu dürfen.
- Um den Motor drehen zu können, ist die *Power state machine* auf den Status *Operation enabled* zu setzen (siehe **CiA 402 Power State Machine**)

7.6.7 Operation

Nach dem Setup ist die Aufgabe der übergerodeten Steuerung, die Zielpositionen rechtzeitig in das Objekt **60C1**$_h$:01$_h$ zu schreiben.

7.7 Cyclic Synchronous Position

7.7.1 Übersicht

Beschreibung

In diesem Modus wird der Steuerung in festen Zeitabständen (im Folgenden *Zyklus* genannt) über den Feldbus eine absolute Positionsvorgabe übergeben. Die Steuerung berechnet dabei keine Rampen mehr, sondern folgt nur noch den Vorgaben.

Die Zielposition wird zyklisch (per *PDO*) übertragen. Das Bit 4 im Controlword muss nicht gesetzt werden (im Gegensatz zum **Profile Position** Modus).

Hinweis

Die Zielvorgabe ist absolut und damit unabhängig davon, wie oft sie pro *Zyklus* versendet wurde.

Hinweis

In diesem Modus sind die Endschalter und damit die Toleranzbänder aktiv. Für weitere Informationen zu den Endschaltern, siehe **Begrenzung des Bewegungsbereichs**.

Aktivierung

Um den Modus zu aktivieren, muss im Objekt **6060**$_h$ (Modes Of Operation) der Wert "8" gesetzt werden (siehe "**CiA 402 Power State Machine**").

Controlword

In diesem Modus haben die Bits des Controlword **6040**$_h$ keine gesonderte Funktion.

Statusword

Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:

Bit	Wert	Beschreibung
8	0	Steuerung ist nicht synchron zum Feldbus
8	1	Steuerung ist synchron zum Feldbus
10	0	Reserviert
10	1	Reserviert
12	0	Steuerung folgt nicht der Zielvorgabe, die Vorgabe des **607A**$_h$ (Target Position) wird ignoriert
12	1	Steuerung folgt der Zielvorgabe, das Objekt **607A**$_h$ (Target Position) wird als Eingabe für die Positionsregelung genutzt.
13	0	Kein Schleppfehler
13	1	Schleppfehler

Bit 11: Limit überschritten: Die Sollposition über- oder unterschreitet die in **607D**$_h$ eingegebenen Grenzwerte.

7.7.2 Objekteinträge

Folgende Objekte sind zur Steuerung dieses Modus erforderlich:

- **607A**$_h$ (Target Position): Dieses Objekt muss zyklisch mit dem Positions-Sollwert beschrieben werden.
- **607B**$_h$ (Position Range Limit): Dieses Objekt enthält die Vorgabe für einen Über- oder Unterlauf der Positionsangabe.
- **607D**$_h$ (Software Position Limit): Dieses Objekt legt die Limitierungen fest, innerhalb deren sich die Positionsvorgabe (607A$_h$) befinden muss.
- **6065**$_h$ (Following Error Window): Dieses Objekt gibt einen Toleranz-Korridor in positiver wie negativer Richtung von der Sollvorgabe vor. Befindet sich die Ist-Position länger als die vorgegebene Zeit (**6066**$_h$) außerhalb dieses Korridors, wird ein Schleppfehler gemeldet.
- **6066**$_h$ (Following Error Time Out): Dieses Objekt gibt den Zeitbereich in Millisekunden vor. Sollte sich die Ist-Position länger als dieser Zeitbereich außerhalb des Positions-Korridors (**6065**$_h$) befinden, wird ein Schleppfehler ausgelöst.
- **6085**$_h$ (Quick-Stop Deceleration): Dieses Objekt hält die Bremsbeschleunigung für den Fall, dass ein Quick-Stop ausgelöst wird.
- **605A**$_h$ (Quick-Stop Option Code): Dieses Objekt enthält die Option, die im Falle eines Quick-Stops ausgeführt werden soll.
- **6086**$_h$ (Motion Profile Type):

- **60C2**$_h$:01$_h$ (Interpolation Time Period): Dieses Objekt gibt die Zeit eines *Zyklus* vor, in diesen Zeitabständen muss ein neuer Sollwert in das **607A**$_h$ geschrieben werden.
  Es gilt dabei: Zykluszeit = Wert des **60C2**$_h$:01$_h$ * $10^{\text{Wert des 60C2:02}}$ Sekunden.
- **60C2**$_h$:02$_h$ (Interpolation Time Index): Dieses Objekt gibt die Zeitbasis der Zyklen an. Derzeit wird nur der Wert **60C2**$_h$:02$_h$=-3 unterstützt, das ergibt eine Zeitbasis von 1 Millisekunde.

Folgende Objekte können in dem Modus ausgelesen werden:

- **6064**$_h$ (Position Actual Value)
- **606C**$_h$ (Velocity Actual Value)
- **60F4**$_h$ (Following Error Actual Value)

7.8 Cyclic Synchronous Velocity

7.8.1 Übersicht

Beschreibung

In diesem Modus wird der Steuerung in festen Zeitabständen (im Folgenden *Zyklus* genannt) über den Feldbus eine Geschwindigkeitsvorgabe übergeben. Die Steuerung berechnet dabei keine Rampen mehr, sondern folgt nur noch den Vorgaben.

Hinweis

In diesem Modus sind die Endschalter und damit die Toleranzbänder aktiv. Für weitere Informationen zu den Endschaltern, siehe **Begrenzung des Bewegungsbereichs**.

Aktivierung

Um den Modus zu aktivieren, muss im Objekt **6060**$_h$ (Modes Of Operation) der Wert "9" gesetzt werden (siehe "**CiA 402 Power State Machine**").

Controlword

In diesem Modus haben die Bits des Controlword **6040**$_h$ keine gesonderte Funktion.

Statusword

Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:

Bit	Wert	Beschreibung
8	0	Steuerung ist nicht synchron zum Feldbus
8	1	Steuerung ist synchron zum Feldbus
10	0	Reserviert
10	1	Reserviert
12	0	Steuerung folgt nicht der Zielvorgabe, die Vorgabe des **60FF**$_h$ (Target Velocity) wird ignoriert
12	1	Steuerung folgt der Zielvorgabe, das Objekt **60FF**$_h$ (Target Velocity) wird als Eingabe für die Positionsregelung genutzt.
13	0	Reserviert
13	1	Reserviert

7.8.2 Objekteinträge

Folgende Objekte sind zur Steuerung dieses Modus erforderlich:

- **60FF**$_h$ (Target Velocity): Dieses Objekt muss zyklisch mit dem Geschwindigkeits-Sollwert beschrieben werden.
- **6085**$_h$ (Quick-Stop Deceleration): Dieses Objekt hält die Bremsbeschleunigung für den Fall, dass ein Quick-Stop ausgelöst wird (siehe "**CiA 402 Power State Machine**").
- **605A**$_h$ (Quick-Stop Option Code): Dieses Objekt enthält die Option, die im Falle eines Quick-Stops ausgeführt werden soll (siehe "**CiA 402 Power State Machine**").
- **60C2**$_h$:01$_h$ (Interpolation Time Period): Dieses Objekt gibt die Zeit eines *Zyklus* vor, in diesen Zeitabständen muss ein neuer Sollwert in das **60FF**$_h$ geschrieben werden.
  Es gilt dabei: Zykluszeit = Wert des **60C2**$_h$:01$_h$ * $10^{\text{Wert des 60C2:02}}$ Sekunden.
- **60C2**$_h$:02$_h$ (Interpolation Time Index): Dieses Objekt gibt die Zeitbasis der Zyklen an. Derzeit wird nur der Wert **60C2**$_h$:02$_h$=-3 unterstützt, das ergibt eine Zeitbasis von 1 Millisekunde.

Folgende Objekte können in dem Modus ausgelesen werden:

- **606C**$_h$ (Velocity Actual Value)
- **607E**$_h$ (Polarity)

7.9 Cyclic Synchronous Torque

7.9.1 Übersicht

Beschreibung

In diesem Modus wird der Steuerung in festen Zeitabständen (im Folgenden *Zyklus* genannt) über den Feldbus eine absolute Drehmomentsvorgabe übergeben. Die Steuerung berechnet dabei keine Rampen mehr, sondern folgt nur noch den Vorgaben.

Hinweis

Dieser Modus funktioniert nur wenn der **Closed Loop** aktiviert ist, siehe auch **Inbetriebnahme Closed Loop**.

Hinweis

In diesem Modus sind die Endschalter und damit die Toleranzbänder aktiv. Für weitere Informationen zu den Endschaltern, siehe **Begrenzung des Bewegungsbereichs**.

Aktivierung

Um den Modus zu aktivieren, muss im Objekt **6060**$_h$ (Modes Of Operation) der Wert "10" gesetzt werden (siehe "**CiA 402 Power State Machine**").

Controlword

In diesem Modus haben die Bits des Controlword **6040**$_h$ keine gesonderte Funktion.

Statusword

Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:

Bit	Wert	Beschreibung
8	0	Steuerung ist nicht synchron zum Feldbus
8	1	Steuerung ist synchron zum Feldbus
10	0	Reserviert
10	1	Reserviert

Bit	Wert	Beschreibung
12	0	Steuerung folgt nicht der Zielvorgabe, die Vorgabe des **6071**$_h$ (Target Torque) wird ignoriert
12	1	Steuerung folgt der Zielvorgabe, das Objekt **6071**$_h$ (Target Torque) wird als Eingabe für die Positionsregelung genutzt.
13	0	Reserviert
13	1	Reserviert

7.9.2 Objekteinträge

Folgende Objekte sind zur Steuerung dieses Modus erforderlich:

- **6071**$_h$ (Target Torque): Dieses Objekt muss zyklisch mit dem Drehmoment-Sollwert beschrieben werden und ist relativ zu **6072**$_h$ einzustellen.
- **6072**$_h$ (Max Torque): Beschreibt das maximal zulässige Drehmoment.
- **60C2**$_h$:01$_h$ (Interpolation Time Period): Dieses Objekt gibt die Zeit eines *Zyklus* vor, in diesen Zeitabständen muss ein neuer Sollwert in das **60FF**$_h$ geschrieben werden.
  Es gilt dabei: Zykluszeit = Wert des **60C2**$_h$:01$_h$ * $10^{\text{Wert des 60C2:02}}$ Sekunden.
- **60C2**$_h$:02$_h$ (Interpolation Time Index): Dieses Objekt gibt die Zeitbasis der Zyklen an. Derzeit wird nur der Wert **60C2**$_h$:02$_h$=-3 unterstützt, das ergibt eine Zeitbasis von 1 Millisekunde.

Folgende Objekte können in dem Modus ausgelesen werden:

- **606C**$_h$ (Velocity Actual Value)

7.10 Takt-Richtungs-Modus

7.10.1 Beschreibung

Im Takt-Richtungs-Modus wird der Motor über zwei Eingänge durch eine übergeordnete Positioniersteuerung mit einem Takt- und einem Richtungssignal betrieben. Bei jedem Takt führt der Motor einen Schritt in die dem Richtungssignal entsprechende Richtung aus.

Hinweis

In diesem Modus sind die Endschalter und damit die Toleranzbänder aktiv. Für weitere Informationen zu den Endschaltern, siehe **Begrenzung des Bewegungsbereichs**.

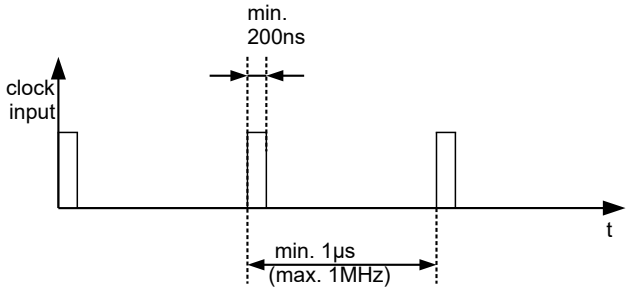
7.10.2 Aktivierung

Um den Modus zu aktivieren, muss im Objekt **6060**$_h$ (Modes Of Operation) der Wert "-1" (bzw. "FFh" gesetzt werden (siehe "**CiA 402 Power State Machine**").

7.10.3 Generelles

Folgende Daten gelten für jede Unterart des Takt-Richtungs-Modus:

- Die maximale Frequenz der Eingangspulse liegt bei 1MHz, der ON-Puls sollte dabei nicht kleiner als 200 ns werden.



- Die Skalierung der Schritte erfolgt über die Objekte **2057**$_h$ und **2058**$_h$. Dabei gilt die folgende Formel:

$$\text{Schrittweite pro Puls} = \frac{2057_h}{2058_h}$$

Ab Werk ist der Wert "Schrittweite pro Puls" = 128 (**2057**$_h$=128 und **2058**$_h$=1) eingestellt, was einem Viertelschritt pro Puls entspricht. Ein Vollschritt ist der Wert "512", ein Halbschritt pro Puls entsprechend "256" usw.

Hinweis

Bei einem Schrittmotor mit 50 Polpaaren entsprechen 200 Vollschritte einer mechanischen Umdrehung der Motorwelle.
Die BLDC-Motoren werden von der Steuerung im *Takt-Richtungs-Modus* auch als Schrittmotoren behandelt. Das bedeutet, dass ,bei einem BLDC-Motor mit z.B. 3 Polpaaren, 12 (=4*3) Vollschritte einer Umdrehung entsprechen.

Hinweis

Bei einem Richtungswechsel ist es nötig, mindestens eine Zeit von 35µs verstreichen zu lassen, bevor der neue Takt angelegt wird.

7.10.4 Statusword

Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:

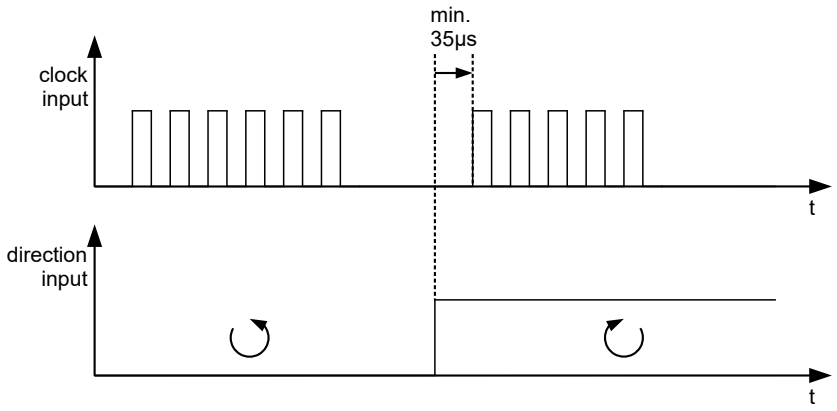
- Bit 13 (Following Error): Dieses Bit wird im *Closed Loop*-Betrieb gesetzt, wenn der Schleppfehler größer als die eingestellten Grenzen ist (**6065**$_h$ (Following Error Window) und **6066**$_h$ (Following Error Time Out)).

7.10.5 Unterarten des Takt-Richtungs-Modus

Takt-Richtungs-Modus (TR-Modus)

Um den Modus zu aktivieren muss das Objekt **205B**$_h$ auf den Wert "0" gesetzt sein (Werkseinstellung).

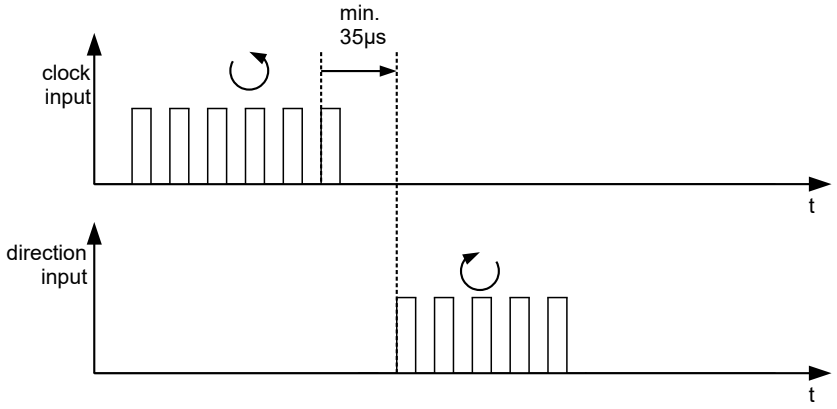
In diesem Modus müssen über den Takteingang die Pulse vorgegeben werden, das Signal des Richtungseingang gibt dabei die Drehrichtung vor (siehe nachfolgende Grafik).



Rechts-/Linkslauf-Modus (CW/CCW-Modus)

Um den Modus zu aktivieren muss das Objekt **205B**$_h$ auf den Wert "1" gesetzt sein.

In diesem Modus entscheidet der verwendete Eingang über die Drehrichtung (siehe nachfolgende Grafik).



7.11 Auto-Setup

7.11.1 Beschreibung

Um einige Parameter im Bezug zum Motor und den angeschlossenen Sensoren (Encoder/ Hallsensoren) zu ermitteln, wird ein Auto-Setup durchgeführt. Der **Closed Loop** Betrieb setzt ein erfolgreich abgeschlossenes *Auto-Setup* voraus. Das *Auto-Setup* ist nur einmal bei der Inbetriebnahme durchzuführen, solange sich der an der Steuerung angeschlossene Motor nicht ändert.

Für Details siehe entsprechenden Abschnitt im Kapitel **Inbetriebnahme**.

Hinweis

In diesem Modus sind die Endschalter und damit die Toleranzbänder aktiv. Für weitere Informationen zu den Endschaltern, siehe **Begrenzung des Bewegungsbereichs**.

7.11.2 Aktivierung

Um den Modus zu aktivieren, muss im Objekt **6060**$_h$ (Modes Of Operation) der Wert "-2" (="FE$_h$") gesetzt werden (siehe **CiA 402 Power State Machine**).

7.11.3 Controlword

Folgende Bits im Objekt **6040**$_h$ (Controlword) haben eine gesonderte Funktion:

- Bit 4 startet einen Fahrauftrag. Dieser wird bei einem Übergang von "0" nach "1" übernommen.

7.11.4 Statusword

Folgende Bits im Objekt **6041**$_h$ (Statusword) haben eine gesonderte Funktion:

- Bit 10: Indexed: zeigt an, ob ein Encoder-Index gefunden wurde (= "1") oder nicht (= "0").
- Bit 12: Aligned: dieses Bit wird auf "1" gesetzt, nachdem das *Auto-Setup* beendet ist

8 Spezielle Funktionen

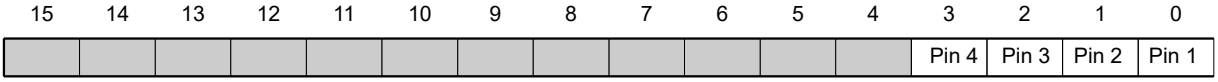
8.1 Digitale Ein- und Ausgänge

Diese Steuerung verfügt über 6 digitale I/O Pins. Davon können 4 wahlweise als Eingang oder Ausgang konfiguriert werden. Die Pins *DIO5_IO_MISO* und *DIO6_IO_CLK* sind als Eingänge vorgegeben.

8.1.1 Ein- und Ausgangsbelegung festlegen

Die digitalen Ein- /Ausgänge 1 … 4 an der PCI-Steckleiste des Geräts können frei belegt werden, siehe auch **Anschlussbelegung** und **3231h Flex IO Configuration**.

- Pin 1: DIO1_IO_CS
- Pin 2: DIO2_CD_CLK
- Pin 3: DIO3_CD_DIR
- Pin 4: DIO4_IO_MOSI



- Subindex 01_h *Output Mask*: Diese Bitmaske legt fest, ob der Pin als Eingang oder Ausgang verwendet wird:
  - Bit = "0:" Pin ist Eingang (Standard)
  - Bit = "1": Pin ist Ausgang
- Subindex 02_h *Pullup Mask*: Diese Bitmaske legt fest, ob der Pin ein *Pullup* oder *Pulldown* ist:
  - Bit = "0": Pin ist *Pulldown* (Standard)
  - Bit = "1": Pin ist *Pullup*

Tipp

Subindex 02_h ist für den Pin nur aktiv, wenn er über Subindex 01_h als Eingang definiert ist.

Beispiel für Subindex 01_h: Pin 2 und Pin 3 sollen Ausgänge sein, Wert ="6" (=0110_b)

1. Prüfen Sie welche Pins Sie als Ein- oder Ausgang definieren möchten.
2. Prüfen Sie welche Eingänge als *Pulldown* oder *Pullup* definieren möchten.
3. Setzen Sie die Werte in **3321**$_h$:01_h und **3321**$_h$:02_h passend.

8.1.2 Bitzuordnung

Die Software der Steuerung ordnet jedem Eingang und Ausgang zwei Bits im jeweiligen Objekt (z.B. **60FDh Digital Inputs** bzw. **60FEh Digital Outputs**) zu:

1. Das erste Bit entspricht der Spezialfunktion eines Ausgangs oder Eingangs. Diese Funktionen sind immer verfügbar auf den Bits 0 bis einschließlich 15 des jeweiligen Objekts. Darunter fallen die Endschalter und der Referenzschalter bei den digitalen Eingängen und die Bremsensteuerung bei den Ausgängen.
2. Das zweite Bit zeigt den Aus-/Eingang an sich als Pegel, diese sind auf Bit 16 bis 31 verfügbar.

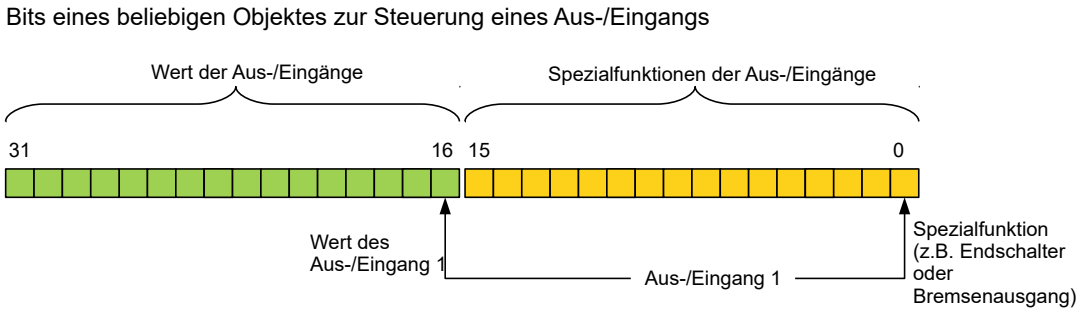
Beispiel

Um den Wert des Ausgangs 2 zu manipulieren, ist immer Bit 17 in **60FE**$_h$ zu benutzen.

Um die Spezialfunktion "Negativer Endschalter" des Eingangs 1 zu aktivieren, ist Bit 0 in **3240**$_h$:01_h zu setzen, und um den Zustand des Eingangs abzufragen ist Bit 0 in **60FD**$_h$ zu

lesen. Das Bit 16 in **60FD**$_h$ zeigt ebenfalls den Zustand des Eingangs 1 (unabhängig davon, ob die Spezialfunktion des Eingangs aktiviert wurde oder nicht).

In der nachfolgenden Zeichnung ist diese Zuordnung graphisch dargestellt.



Tipp

Die ersten 4 I/O Pins können auch als Ausgänge konfiguriert werden, siehe **Ein- und Ausgangsbelegung festlegen**. Sind diese als Ausgänge konfiguriert, kann der aktuelle Zustand immer noch in den Bits 16 bis 19 des Objekts **60FD**$_h$ zurückgelesen werden. Die Zuordnung der Bits im 60FD$_h$ bleibt somit unverändert, Bit 20 entspricht dem Eingang 5 und Bit 21 dem Eingang 6.

8.1.3 Digitale Eingänge

Übersicht

Hinweis

Bei Digitaleingängen mit 5 V darf die Länge der Zuleitungen 3 Meter nicht überschreiten.

Hinweis

Die digitalen Eingänge werden einmal pro Millisekunde erfasst. Signaländerungen am Eingang kürzer als eine Millisekunde werden nicht verarbeitet.

Folgende Eingänge stehen zur Verfügung:

PIN/Eingang	Name für Input Routing	Auslieferungszustand
B3/DIO1_IO_CS	physikalischer Eingang 1	keine
B4/DIO2_CD_CLK	physikalischer Eingang 2	Takteingang im Takt-Richtungs Modus
B5/DIO3_CD_DIR	physikalischer Eingang 3	Referenzschalter / Richtungseingang im Takt-Richtungs Modus
B6/DIO4_IO_MOSI	physikalischer Eingang 4	keine
B7/DIO5_IO_MISO	physikalischer Eingang 5	keine
B8/DIO6_IO_CLK	physikalischer Eingang 6	keine

Objekteinträge

Über die folgenden OD-Einstellungen kann der Wert eines Eingangs manipuliert werden, wobei hier immer nur das entsprechende Bit auf den Eingang wirkt.

- **3240**$_h$:01$_h$ (Special Function Enable): Dieses Bit erlaubt Sonderfunktionen eines Eingangs aus- (Wert "0") oder einzuschalten (Wert "1"). Soll Eingang 1 z.B. nicht als negativer Endschalter verwendet werden, so muss die Sonderfunktion abgeschaltet werden, damit nicht fälschlicherweise auf den Signalgeber reagiert wird. Auf die Bits 16 bis 31 hat das Objekt keine Auswirkungen.
  Die Firmware wertet folgende Bits aus:
  - Bit 0: Negativer Endschalter
  - Bit 1: Positiver Endschalter
  - Bit 2: Referenzschalter

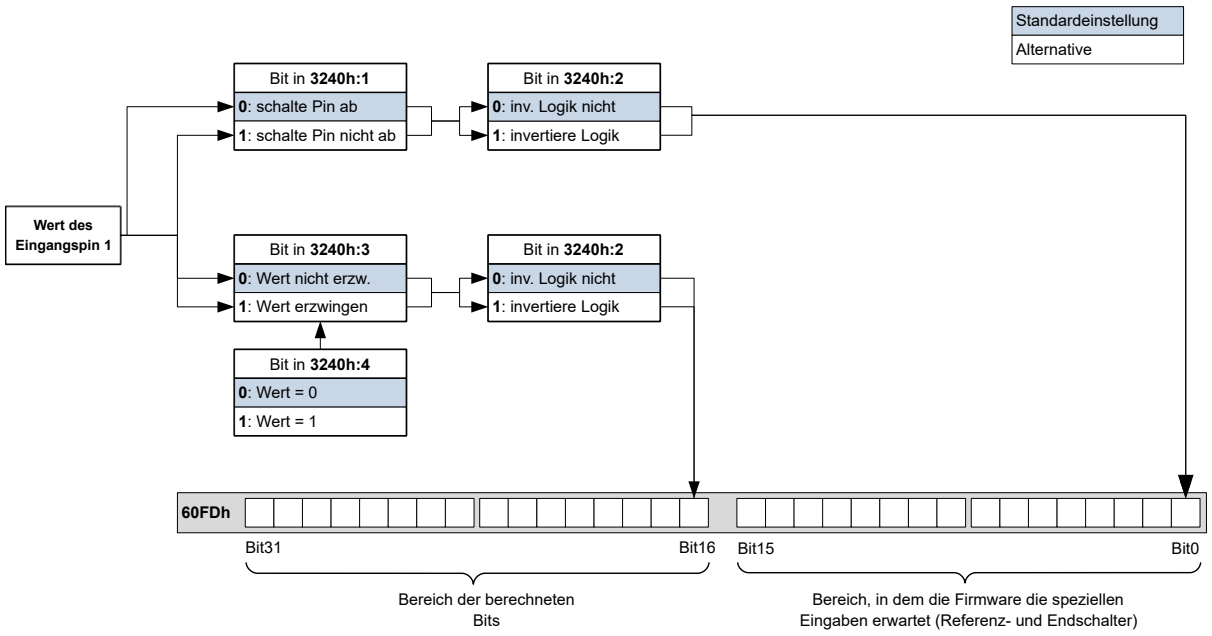
  Sollen z.B. zwei Endschalter und ein Referenzschalter verwendet werden, müssen Bits 0-2 in **3240**$_h$:01$_h$ auf "1" gesetzt werden

- **3240**$_h$:02$_h$ (Function Inverted): Dieses Bit wechselt von Schließer-Logik (ein logischer High-Pegel am Eingang ergibt den Wert "1" im Objekt **60FD**$_h$) auf Öffner-Logik (der logische High-Pegel am Eingang ergibt den Wert "0"). Das gilt für die Sonderfunktionen (außer den Takt- und Richtungseingängen) und für die normalen Eingänge.
  Hat das Bit den Wert "0" gilt Schließer-Logik, entsprechend bei dem Wert "1" die Öffner-Logik. Bit 0 entspricht dabei dem Eingang 1, Bit 1 dem Eingang 2 usw. .
- **3240**$_h$:03$_h$ (Force Enable): Dieses Bit schaltet die Softwaresimulation von Eingangswerten ein, wenn es auf "1" gesetzt ist. Dann werden nicht mehr die tatsächlichen sondern die in Objekt **3240**$_h$:04$_h$ eingestellten Werte für den jeweiligen Eingang verwendet.
- **3240**$_h$:04$_h$ (Force Value): Dieses Bit gibt den Wert vor, der als Eingangswert eingelesen werden soll, wenn das gleiche Bit im Objekt **3240**$_h$:03$_h$ gesetzt wurde.
- **3240**$_h$:05$_h$ (Raw Value): Dieses Objekt beinhaltet den unmodifizierten Eingabewert.
- **60FD**$_h$ (Digital Inputs): Dieses Objekt enthält eine Zusammenfassung der Eingänge und den Spezialfunktionen.

Verrechnung der Eingänge

Verrechnung des Eingangssignals am Beispiel von Eingang 1:

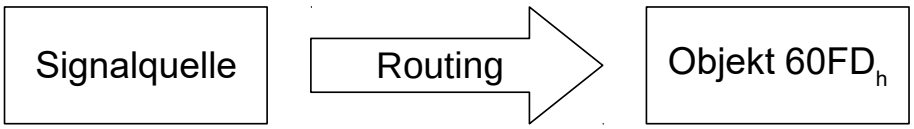
Der Wert an Bit 0 des Objekts **60FD**$_h$ wird von der Firmware als negativer Endschalter interpretiert, das Ergebnis der vollständigen Verrechnung wird in Bit 16 abgelegt.



Input Routing

Prinzip

Um die Zuordnung der Eingänge flexibler vornehmen zu können, existiert der sogenannte *Input Routing Modus*. Dieser weist ein Signal einer Quelle auf ein Bit in dem Objekt **60FD**$_h$ zu.



Aktivierung

Dieser Modus wird aktiviert, indem das Objekt **3240**$_h$:08$_h$ (Routing Enable) auf 1 gesetzt wird.

Hinweis

Die Einträge **3240**$_h$:01$_h$ bis **3240**:04$_h$ haben dann **keine** Funktion mehr, bis das Eingangsrouting wieder abgeschaltet wird.

Hinweis

Wird das *Input Routing* eingeschaltet, werden initial die Werte des **3242**$_h$ geändert und entsprechen der Funktion der Inputs, wie diese vor der Aktivierung des *Input Routing* war. Die Eingänge der Steuerung verhalten sich mit der Aktivierung des *Input Routing* gleich. Es sollte daher nicht zwischen dem normalen Modus und dem *Input Routing* hin- und her geschalten werden.

Routing

Das Objekt 3242$_h$ bestimmt, welche Signalquelle auf welches Bit des **60FD**$_h$ geroutet wird. Der Subindex 01$_h$ des 3242$_h$ bestimmt Bit 0, Subindex 02$_h$ das Bit 1, und so weiter. Die Signalquellen und deren Nummern finden Sie in den nachfolgenden Listen.

Nummer		
dec	**hex**	**Signalquelle**
00	00	Signal ist immer 0
01	01	Physikalischer Eingang 1
02	02	Physikalischer Eingang 2
03	03	Physikalischer Eingang 3
04	04	Physikalischer Eingang 4
05	05	Physikalischer Eingang 5
06	06	Physikalischer Eingang 6
07	07	Physikalischer Eingang 7
08	08	Physikalischer Eingang 8
09	09	Physikalischer Eingang 9
10	0A	Physikalischer Eingang 10
11	0B	Physikalischer Eingang 11
12	0C	Physikalischer Eingang 12
13	0D	Physikalischer Eingang 13
14	0E	Physikalischer Eingang 14

Nummer		
dec	**hex**	**Signalquelle**
15	0F	Physikalischer Eingang 15
16	10	Physikalischer Eingang 16
65	41	Hall Eingang "U"
66	42	Hall Eingang "V"
67	43	Hall Eingang "W"
68	44	Encoder Eingang "A"
69	45	Encoder Eingang "B"
70	46	Encoder Eingang "Index"

Die nachfolgende Tabelle beschreibt die invertierten Signale der vorherigen Tabelle.

Nummer		
dec	**hex**	**Signalquelle**
128	80	Signal ist immer 1
129	81	Invertierter physikalischer Eingang 1
130	82	Invertierter physikalischer Eingang 2
131	83	Invertierter physikalischer Eingang 3
132	84	Invertierter physikalischer Eingang 4
133	85	Invertierter physikalischer Eingang 5
134	86	Invertierter physikalischer Eingang 6
135	87	Invertierter physikalischer Eingang 7
136	88	Invertierter physikalischer Eingang 8
137	89	Invertierter physikalischer Eingang 9
138	8A	Invertierter physikalischer Eingang 10
139	8B	Invertierter physikalischer Eingang 11
140	8C	Invertierter physikalischer Eingang 12
141	8D	Invertierter physikalischer Eingang 13
142	8E	Invertierter physikalischer Eingang 14
143	8F	Invertierter physikalischer Eingang 15
144	90	Invertierter physikalischer Eingang 16
193	C1	Invertierter Hall Eingang "U"
194	C2	Invertierter Hall Eingang "V"
195	C3	Invertierter Hall Eingang "W"
196	C4	Invertierter Encoder Eingang "A"
197	C5	Invertierter Encoder Eingang "B"
198	C6	Invertierter Encoder Eingang "Index"

Beispiel

Es soll der Eingang 1 auf Bit 16 des Objekts **60FD**$_h$ geroutet werden:

Die Nummer der Signalquelle für Eingang 1 ist die "1". Das Routing für Bit 16 wird in das 3242_h:11_h geschrieben.

Demnach muss das Objekt 3242_h:11_h auf den Wert "1" gesetzt werden.

8.1.4 Digitale Ausgänge

Ausgänge

Die Ausgänge werden über das Objekt **60FE**$_h$ gesteuert. Dabei entspricht Ausgang 1 dem Bit 16 im Objekt **60FE**$_h$, Ausgang 2 dem Bit 17 usw. wie bei den Eingängen. Die ersten 4 I/O Pins können als Ausgänge konfiguriert werden, siehe **Ein- und Ausgangsbelegung festlegen**. Die Ausgänge mit Sonderfunktionen sind in der Firmware wieder in den unteren Bits 0 bis 15 eingetragen. Im Moment ist nur Bit 0 belegt, das die Motorbremse steuert.

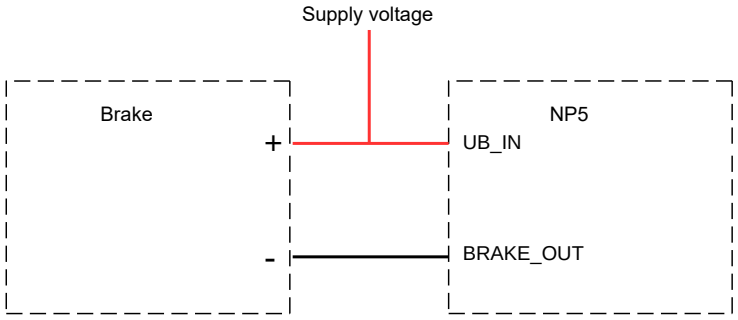
Beschaltung

Hinweis

Beachten Sie immer die maximale Belastbarkeit des Ausgangs (siehe **Anschlussbelegung**).

Die digitalen Ausgänge, mit der Ausnahme des Bremsenausgangs, haben einen digitalen Pegel von 3,3 V DC. Die Strombelastbarkeit liegt bei 10mA.

Der Bremsenausgang ist als *Open Drain* realisiert. Demzufolge ist immer eine externe Spannungsversorgung nötig, wie in der folgenden Abbildung zu sehen. Siehe auch **Automatische Bremsensteuerung**.



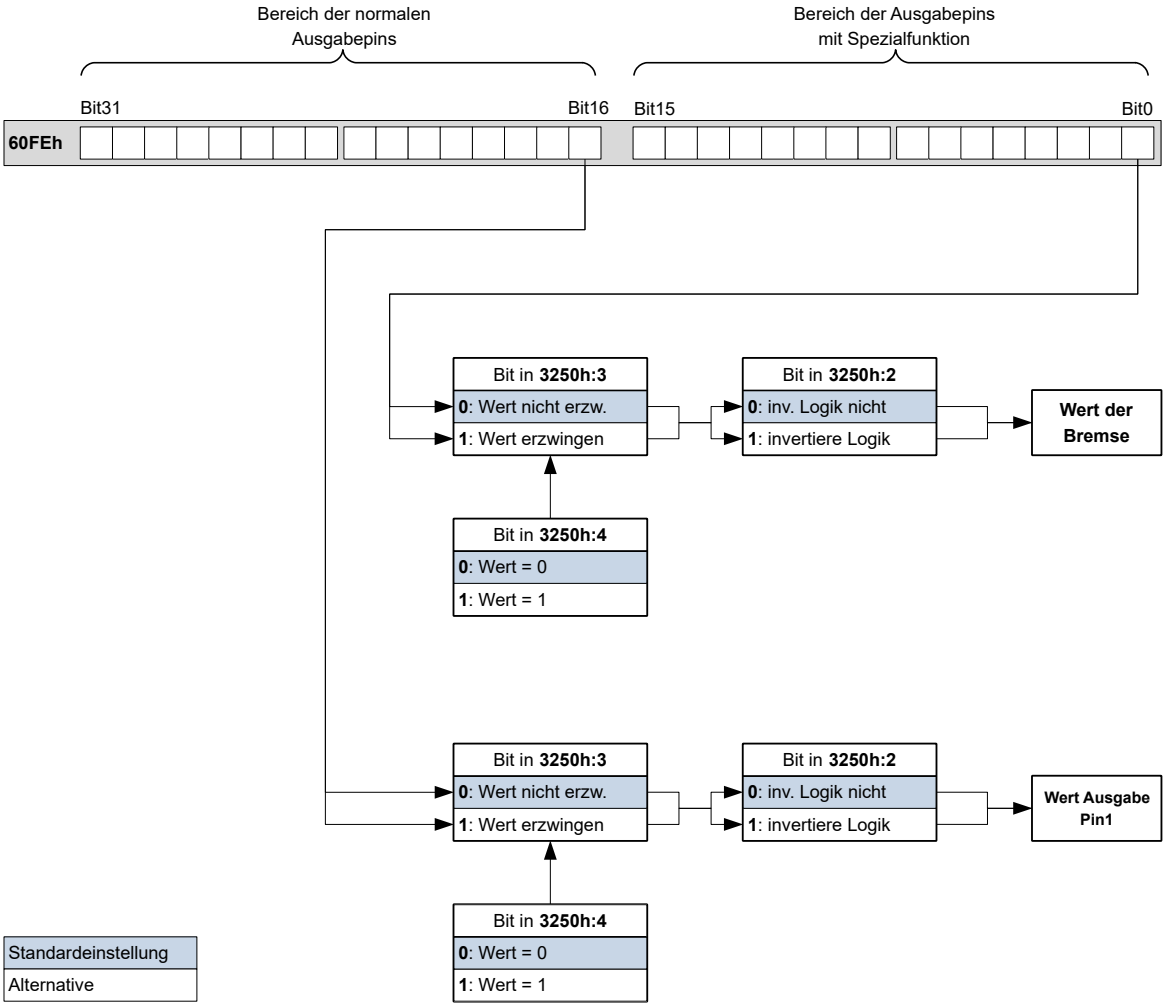
Objekteinträge

Es existieren zusätzliche OD-Einträge, um den Wert der Ausgänge zu manipulieren (siehe dazu das nachfolgende Beispiel). Ähnlich wie bei den Eingängen wirkt immer nur das Bit an der entsprechenden Stelle auf den jeweiligen Ausgang:

- **3250**$_h$:01$_h$: Keine Funktion.
- **3250**$_h$:02$_h$: Damit lässt sich die Logik von *Schließer* auf *Öffner* umstellen. Als *Schließer* konfiguriert, gibt der Eingang einen logischen High-Pegel ab, sollte das Bit "1" sein. Bei der *Öffner* -Konfiguration wird bei einer "1" im Objekt **60FE**$_h$ entsprechend ein logischer Low-Pegel ausgegeben.
- **3250**$_h$:03$_h$: Ist hier ein Bit gesetzt, wird der Ausgang manuell gesteuert. Der Wert für den Ausgang steht dann in Objekt **3250**$_h$:4$_h$, dies ist auch für den Bremsenausgang möglich.
- **3250**$_h$:04$_h$: Die Bits in diesem Objekt geben den Ausgabewert vor, welcher am Ausgang angelegt sein soll, wenn die manuelle Steuerung des Ausgangs über das Objekt **3250**$_h$:03$_h$ aktiviert ist.
- **3250**$_h$:05$_h$: Dieses Objekt besitzt keine Funktion und ist aus Gründen der Kompatibilität enthalten.

Verrechnung der Ausgänge

Beispiel für die Verrechnung der Bits für die Ausgänge:

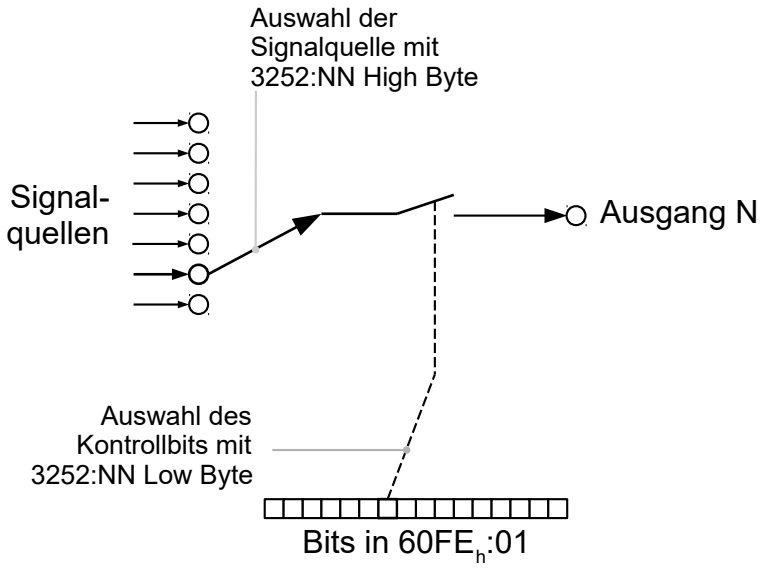


Output Routing

Prinzip

Der "Output Routing Mode" weist einem Ausgang eine Signalquelle zu, ein Kontrollbit im Objekt **60FE**$_h$:01$_h$ schaltet das Signal ein oder aus.

Die Auswahl der Quelle wird mit **3252**$_h$:01 bis 05 im "High Byte" (Bit 15 bis Bit 8) gemacht. Die Zuordnung eines Kontrollbit aus dem Objekt **60FE**$_h$:01$_h$ erfolgt im "Low Byte" (Bit 7 bis Bit 0) des **3252**$_h$:01$_h$ bis 05 (siehe nachfolgende Abbildung).



Aktivierung

Dieser Modus wird aktiviert, indem das Objekt **3250**$_h$:08$_h$ (Routing Enable) auf 1 gesetzt wird.

Hinweis

Die Einträge **3250**$_h$:01$_h$ bis **3250**:04$_h$ haben dann **keine** Funktion mehr, bis das "Ausgangsrouting" wieder abgeschaltet wird.

Routing

Der Subindex des Objekts **3252**$_h$ bestimmt, welche Signalquelle auf welchen Ausgang geroutet wird. Die Zuordnung der Ausgänge ist nachfolgend gelistet:

Subindex 3252$_h$	Output Pin
01$_h$	Konfiguration des Bremsenausgangs (falls verfügbar)
02$_h$	Konfiguration des Ausgangs 1
03$_h$	Konfiguration des Ausgangs 2 (falls verfügbar)
04$_h$	Konfiguration des Ausgangs 3 (falls verfügbar)
05$_h$	Konfiguration des Ausgangs 4 (falls verfügbar)

Hinweis

Die maximale Ausgangsfrequenz des Bremsenausgangs, Ausgang 1 und Ausgang 2 ist 10kHz. Alle anderen Ausgänge können nur bis zu 500Hz Signale erzeugen.

Die Subindizes **3252**$_h$:01$_h$ bis 05$_h$ sind 16 Bit breit, wobei das High Byte die Signalquelle auswählt (z.B. den PWM-Generator) und das Low Byte bestimmt das Kontrollbit im Objekt **60FE**$_h$:01.

Bit 7 von **3252**$_h$:01$_h$ bis 05 invertiert die Steuerung aus dem Objekt **60FE**$_h$:01. Normalerweise schaltet der Wert "1" im Objekt **60FE**$_h$:01 das Signal "ein", ist das Bit 7 gesetzt, schaltet der Wert "0" das Signal ein.

Nummer in 3252:01 bis 05	
$00XX_h$	Ausgang ist immer "1"
$01XX_h$	Ausgang ist immer "0"
$02XX_h$	Encodersignal (**6063_h**) mit Frequenzteiler 1
$03XX_h$	Encodersignal (**6063_h**) mit Frequenzteiler 2
$04XX_h$	Encodersignal (**6063_h**) mit Frequenzteiler 4
$05XX_h$	Encodersignal (**6063_h**) mit Frequenzteiler 8
$06XX_h$	Encodersignal (**6063_h**) mit Frequenzteiler 16
$07XX_h$	Encodersignal (**6063_h**) mit Frequenzteiler 32
$08XX_h$	Encodersignal (**6063_h**) mit Frequenzteiler 64
$09XX_h$	Position Actual Value (**6064_h**) mit Frequenzteiler 1
$0AXX_h$	Position Actual Value (**6064_h**) mit Frequenzteiler 2
$0BXX_h$	Position Actual Value (**6064_h**) mit Frequenzteiler 4
$0CXX_h$	Position Actual Value (**6064_h**) mit Frequenzteiler 8
$0DXX_h$	Position Actual Value (**6064_h**) mit Frequenzteiler 16
$0EXX_h$	Position Actual Value (**6064_h**) mit Frequenzteiler 32
$0FXX_h$	Position Actual Value (**6064_h**) mit Frequenzteiler 64
$10XX_h$	PWM-Signal, das mit Objekt **2038_h**:05_h und 06_h konfiguriert wird
$11XX_h$	Invertiertes PWM-Signal, das mit Objekt **2038_h**:05_h und 06_h konfiguriert wird

Beispiel

Das Encodersignal (**6063_h**) soll auf Ausgang 1 mit einem Frequenzteiler 4 gelegt werden. Der Ausgang soll mit Bit 5 des Objektes **60FE**:01 gesteuert werden.

- **3250_h**:08_h = 1 (Routing aktivieren)
- **3252_h**:02_h = 0405_h ($04XX_h$ + 0005_h) Dabei ist:
- $04XX_h$: Encodersignal mit Frequenzteiler 4
- 0005_h: Auswahl von Bit 5 des **60FE**:01

Das Einschalten des Ausgangs wird mit dem Setzen des Bit 5 in Objekt **60FE**:01 erledigt.

Beispiel

Das PWM-Signal soll auf Ausgang 2 gelegt werden. Das Bit 0 des **60FE**:01_h soll als Kontrollbit benutzt werden.

- **3250_h**:08_h = 1 (Routing aktivieren)
- **3252_h**:03_h = 1080_h (=$10XX_h$ + 0080_h). Dabei gilt:
  - $10XX_h$: PWM-Signal
  - 0080_h: Auswahl des invertierten Bits 0 des Objekts **60FE**:01

8.2 Automatische Bremsensteuerung

8.2.1 Beschreibung

Die automatische Bremsensteuerung wird aktiv, wenn die Steuerung in den Zustand *Operation Enabled* der **CiA 402 Power State Machine** gebracht wird, sonst bleibt die Bremse immer geschlossen.

Der Bremsen-Ausgang der Steuerung resultiert in einem PWM-Signal, welches sich in der Frequenz und in dem Tastverhältnis einstellen lässt.

Für das Zusammenspiel der Bremse mit dem Motorstoppverhalten, lesen Sie auch das Kapitel **Power State machine - Bremsreaktionen**.

8.2.2 Aktivierung und Anschluss

Die Bremse kann entweder automatisch oder manuell gesteuert werden:

- Automatisch: Bit 2 des Objekts **3202**$_h$ auf "1" setzen aktiviert die Bremsensteuerung.
- Manuell: Bit 2 des Objekts **3202**$_h$ auf "0" setzen deaktiviert die Bremsensteuerung, die Bremse lässt sich jetzt mit dem Bit 0 im Objekt **60FE**$_h$:01$_h$ kontrollieren.

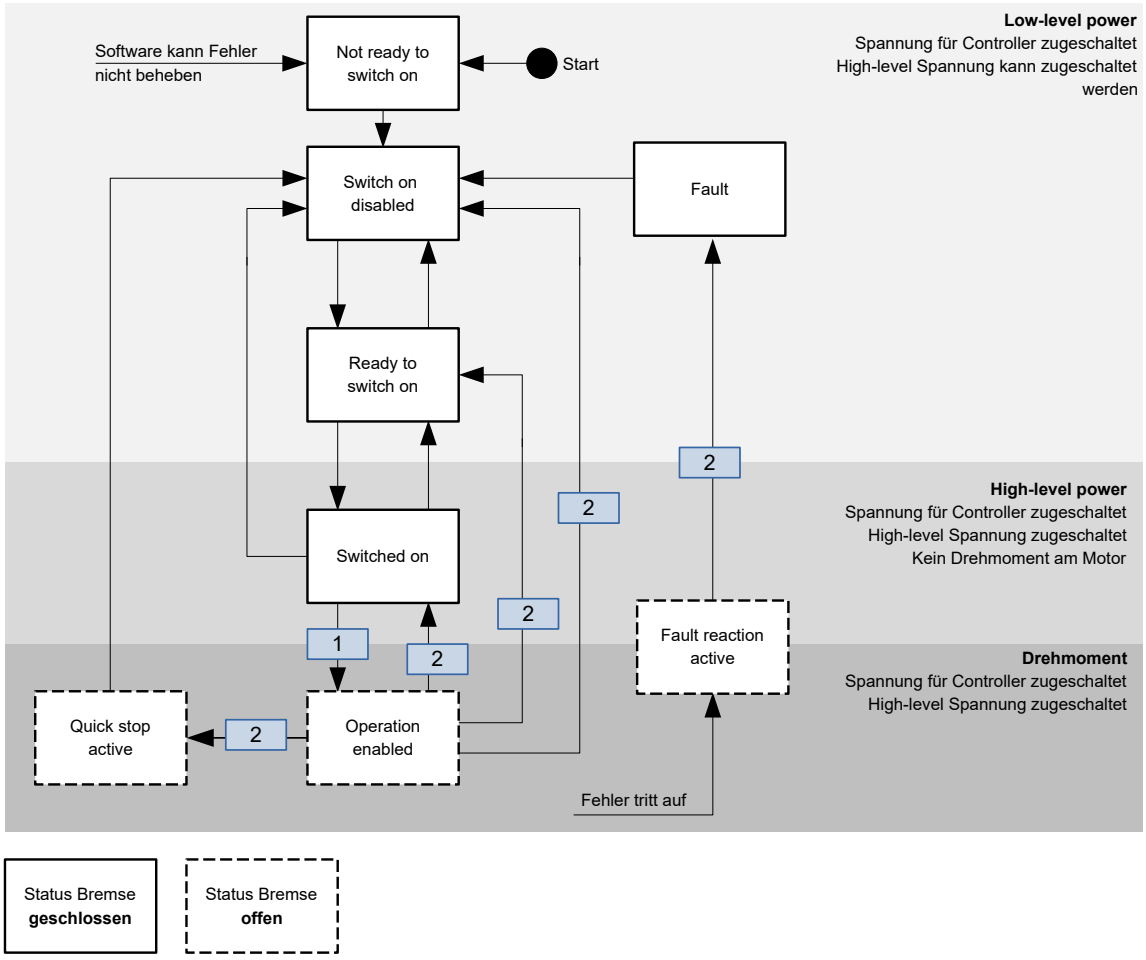
Anschluss

Der Bremsenausgang befindet sich:

- am Pin A48 der PCI Steckleiste , siehe **Anschlussbelegung** und **Beschaltung der Ausgänge**
- am Stecker X2 des Discovery Boards, falls dieses verwendet wird, siehe **Stecker X2 - Bremse**

8.2.3 Steuerung der Bremse

Die nachfolgende Grafik zeigt die Zustände der **CiA 402 Power State Machine** zusammen mit den Zuständen der Bremse für den automatischen Modus.



Bei dem Übergang, welcher mit 1 markiert ist, werden folgende Schritte durchgeführt:

1. Der Motorstrom wird eingeschaltet.
2. Die Zeit, welche in **2038**$_h$:3$_h$ hinterlegt wird, wird abgewartet.
3. Die Bremse löst sich.
4. Die Zeit, welche in **2038**$_h$:4$_h$ hinterlegt wird, wird abgewartet.
5. Der Zustand *Operation Enabled* wird erreicht, die Motorsteuerung kann Fahrbefehle umsetzen.

Bei allen Übergängen, welche mit 2 markiert sind, werden folgende Schritte durchgeführt:

1. Der Motor wird zum Stillstand gebracht.
2. Die Zeit, welche in **2038**$_h$:1$_h$ hinterlegt wird, wird abgewartet.
3. Die Bremse wird aktiviert.
4. Die Zeit, welche in **2038**$_h$:2$_h$ hinterlegt wird, wird abgewartet.
5. Der Motorstrom wird abgeschaltet.

8.2.4 Bremsen-PWM

Die eingeschaltete Bremse erzeugt am Ausgang der Steuerung ein PWM-Signal, welches im Tastgrad und der Frequenz eingestellt werden kann. Sollte ein Ausgangspin ohne PWM benötigt werden, lässt sich ein Tastgrad von 100 Prozent einstellen.

Hinweis

Der PIN *Bremse +* des Bremsenausgangs ist mit der Spannungsversorgung der Steuerung verbunden.

Wenn die Betriebsspannung der Bremse größer als die Versorgungsspannung der Steuerung ist, können Sie den Bremsenausgang der Steuerung nicht nutzen, Sie müssen die Bremse extern versorgen.

Wenn die Versorgungsspannung der Steuerung größer als die Betriebsspannung der Bremse ist (und bis 48 V DC), wird empfohlen, den PWM-Regler von Nanotec mit der Bestellbezeichnung *EB-BRAKE-48V* zu verwenden und den Tastgrad des Bremsenausgangs der Steuerung auf "100" zu setzen.

Frequenz

Die Frequenz der Bremsen-PWM kann im Objekt **2038**$_h$:5$_h$ eingestellt werden. Die Einheit ist Hertz, ein Wert größer 2000 ist nicht möglich.

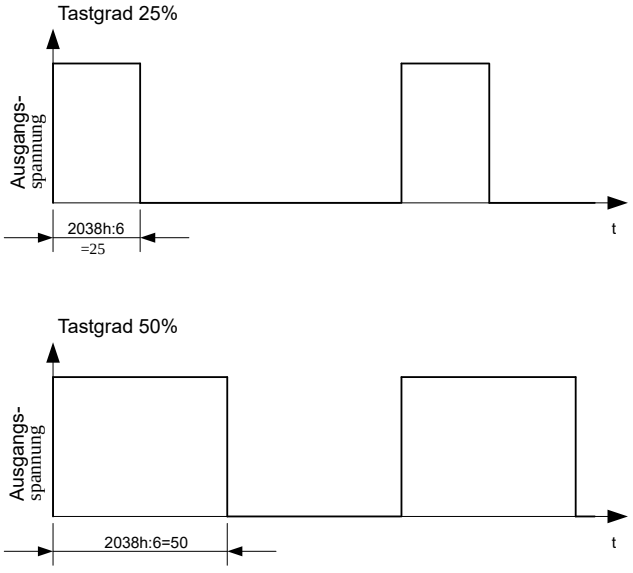
Hinweis

Sollte das PWM-Signal der Bremse störende Geräusche verursachen, so kann dies durch Parallelschaltung eines 47 µF … 100 µF Kondensators am Bremsenausgang behoben werden.

Tastgrad

Der Tastgrad - das Verhältnis Impuls- zu Periodendauer - wird im **2038**$_h$:6$_h$ eingestellt. Der Wert wird als Prozentzahl angesehen und kann zwischen 2 und 100 gewählt werden. Bei einem Wert von 100 ist der Ausgangspin dauerhaft eingeschaltet.

In nachfolgender Abbildung ist beispielhaft ein Tastgrad von 25 und 50 Prozent eingezeichnet, wobei die Frequenz beibehalten wurde.



8.3 I^2t Motor-Überlastungsschutz

8.3.1 Beschreibung

Hinweis

Für Schrittmotoren wird nur der Nennstrom und kein Maximalstrom angegeben. Daher erfolgt die Nutzung von I^2t mit Schrittmotoren ohne Gewähr.

Das Ziel des I^2t Motor-Überlastungsschutz ist es, den Motor vor einem Schaden zu bewahren und gleichzeitig, ihn normal bis zu seinem thermischen Limit zu betreiben.

Diese Funktion ist nur verfügbar, wenn sich die Steuerung in der **Closed Loop-Betriebsart** befindet (Bit 0 des Objekts **3202**$_h$ muss auf "1" gesetzt sein).

Es gibt eine Ausnahme: Sollte I^2t im *Open Loop*-Betrieb aktiviert sein, wird der Strom auf den eingestellten Nennstrom begrenzt, auch wenn der eingestellte Maximalstrom größer ist. Diese Funktion wurde aus Sicherheitsgründen implementiert, damit man auch aus dem *Closed Loop*-Betrieb mit sehr hohem kurzzeitigem Maximalstrom in den *Open Loop*-Betrieb wechseln kann, ohne den Motor zu schädigen.

8.3.2 Objekteinträge

Folgende Objekte haben Einfluss auf den I^2t Motor-Überlastungsschutz:

- **2031**$_h$: Peak Current - Gibt den Maximalstrom in mA an.
- **203B**$_h$:1$_h$ Nominal Current - Gibt den Nennstrom in mA an.
- **203B**$_h$:2$_h$ Maximum Duration Of Peak Current - Gibt die maximale Dauer des Maximalstroms in ms an.

Folgende Objekte zeigen den gegenwärtigen Zustand von I^2t an:

- **203B**$_h$:3$_h$ Threshold - Gibt die Grenze in mAs an, von der abhängt, ob auf Maximalstrom oder Nennstrom geschaltet wird.
- **203B**$_h$:4$_h$ CalcValue - Gibt den berechneten Wert an, welcher mit Threshold verglichen wird, um den Strom einzustellen.
- **203B**$_h$:5$_h$ LimitedCurrent - Zeigt den gegenwärtigen Stromwert an, der von I^2t eingestellt wurde.
- **203B**$_h$:6$_h$ Status:
  - Wert = "0": I^2t deaktiviert
  - Wert = "1": I^2t aktiviert

8.3.3 Aktivierung

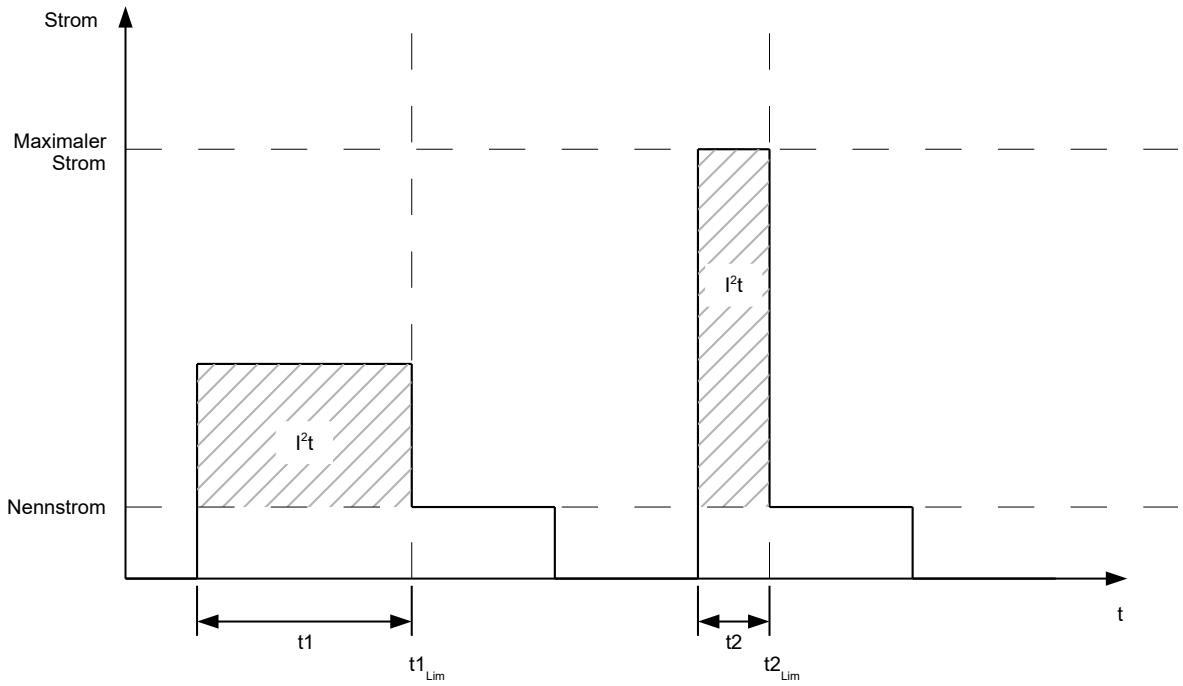
Der *Closed Loop* muss aktiviert sein (Bit 0 des Objekts **3202**$_h$ auf "1" gesetzt, siehe auch Kapitel **Closed Loop**). Zum Aktivieren des Modus müssen die drei oben genannten Objekteinträge (**2031**$_h$, **203B**$_h$:1,$_h$ **203B**$_h$:2$_h$) sinnvoll beschrieben worden sein. Das bedeutet, dass der Maximalstrom größer als der Nennstrom sein muss und ein Zeitwert für die maximale Dauer des Maximalstroms eingetragen sein muss. Wenn diese Bedingungen nicht erfüllt sind, bleibt die I^2t Funktionalität deaktiviert.

8.3.4 Funktion von I^2t

Durch die Angabe von Nennstrom, Maximalstrom und maximaler Dauer des Maximalstromes wird ein I^2T_{Lim} berechnet.

Der Motor kann solange mit Maximalstrom laufen, bis das berechnete I^2T_{Lim} erreicht wird. Darauffolgend wird der Strom sofort auf Nennstrom gesenkt.

Im folgenden Diagramm sind die Zusammenhänge noch einmal dargestellt.



Im ersten Abschnitt t1 ist der Stromwert höher als der Nennstrom. Am Zeitpunkt $t1_{Lim}$ wird I^2t_{Lim} erreicht und der Strom wird auf Nennstrom begrenzt. Danach kommt während der Dauer t2 ein Strom, der dem Maximalstrom entspricht. Dementsprechend ist der Wert für I^2t_{Lim} schneller erreicht, als im Zeitraum t1.

8.4 Objekte speichern

Hinweis

Die unsachgemäße Anwendung dieser Funktion kann dazu führen, dass die Steuerung sich nicht mehr starten lässt. Lesen Sie daher vor der Benutzung der Funktion das Kapitel vollständig durch.

8.4.1 Allgemeines

Viele Objekte im Objektverzeichnis lassen sich speichern und werden beim nächsten Einschalten/ Reset automatisch wieder geladen. Zudem bleiben die gespeicherten Werte auch bei einem Firmware-Update erhalten.

Es lassen sich immer nur ganze Sammlungen von Objekten (im Folgenden *Kategorien* genannt) zusammen abspeichern, einzelne Objekte können nicht gespeichert werden.

Ein Objekt kann einer der folgenden *Kategorien* zugeordnet sein:

- Kommunikation: Parameter mit Bezug auf externe Schnittstellen, wie PDO-Konfiguration etc.
- Applikation: Parameter mit Bezug auf Betriebsmodi.
- Benutzer: Parameter, die ausschließlich vom Kunden/Benutzer geschrieben und gelesen, und von der Steuerungsfirmware ignoriert werden.
- Bewegung: Parameter mit Bezug auf den Motor und die Sensoren (BLDC/Stepper, *Closed/Open Loop*...). Einige werden vom Auto-Setup gesetzt und gespeichert.
- Tuning: Parameter mit Bezug auf Motor und Encoder, die entweder vom Auto-Setup gesetzt werden, oder den Datenblättern entnommen werden können, zum Beispiel Polpaare und Maximum Current.

Wenn ein Objekt keiner dieser *Kategorien* zugeordnet ist, kann es nicht gespeichert werden, zum Beispiel Statusword und alle Objekte, deren Wert abhängig vom aktuellen Zustand der Steuerung ist.

Die Objekte in jeder *Kategorie* werden unten aufgelistet. Im Kapitel **Objektverzeichnis Beschreibung** wird ebenfalls für jedes Objekt die zugehörige *Kategorie* angegeben.

8.4.2 Kategorie: Kommunikation

- **1600**$_h$: Receive PDO 1 Mapping Parameter
- **1601**$_h$: Receive PDO 2 Mapping Parameter
- **1602**$_h$: Receive PDO 3 Mapping Parameter
- **1603**$_h$: Receive PDO 4 Mapping Parameter
- **1A00**$_h$: Transmit PDO 1 Mapping Parameter
- **1A01**$_h$: Transmit PDO 2 Mapping Parameter
- **1A02**$_h$: Transmit PDO 3 Mapping Parameter
- **1A03**$_h$: Transmit PDO 4 Mapping Parameter
- **2102**$_h$: Fieldbus Module Control
- **3400**$_h$: NanoSPI Comm Rx PDO Assignment
- **3401**$_h$: NanoSPI Comm Tx PDO Assignment
- **3402**$_h$: NanoSPI Ctrl Rx PDO Assignment
- **3403**$_h$: NanoSPI Ctrl Tx PDO Assignment
- **3410**$_h$: NanoSPI Comm Controlword
- **3412**$_h$: NanoSPI SDO Control
- **3413**$_h$: NanoSPI SDO Request
- **3414**$_h$: NanoSPI SDO Raw Request
- **3416**$_h$: NanoSPI Slave Rx PDO Data
- **3417**$_h$: NanoSPI Slave Tx PDO Data
- **3500**$_h$: NanoSPI Rx PDO Mapping
- **3600**$_h$: NanoSPI Tx PDO Mapping

8.4.3 Kategorie: Applikation

- **2033**$_h$: Plunger Block
- **2034**$_h$: Upper Voltage Warning Level
- **2035**$_h$: Lower Voltage Warning Level
- **2036**$_h$: Open Loop Current Reduction Idle Time
- **2037**$_h$: Open Loop Current Reduction Value/factor
- **2038**$_h$: Brake Controller Timing
- **203A**$_h$: Homing On Block Configuration
- **203D**$_h$: Torque Window
- **203E**$_h$: Torque Window Time
- **2056**$_h$: Limit Switch Tolerance Band
- **2057**$_h$: Clock Direction Multiplier
- **2058**$_h$: Clock Direction Divider
- **205B**$_h$: Clock Direction Or Clockwise/Counter Clockwise Mode
- **2060**$_h$: Compensate Polepair Count
- **2061**$_h$: Velocity Numerator
- **2062**$_h$: Velocity Denominator
- **2063**$_h$: Acceleration Numerator
- **2064**$_h$: Acceleration Denominator
- **2065**$_h$: Jerk Numerator
- **2066**$_h$: Jerk Denominator
- **2084**$_h$: Bootup Delay
- **2300**$_h$: NanoJ Control
- **2410**$_h$: NanoJ Init Parameters
- **2800**$_h$: Bootloader And Reboot Settings
- **320A**$_h$: Motor Drive Sensor Display Open Loop

- **320B**$_h$: Motor Drive Sensor Display Closed Loop
- **3210**$_h$: Motor Drive Parameter Set
- **3212**$_h$: Motor Drive Flags
- **3221**$_h$: Analogue Inputs Control
- **3231**$_h$: Flex IO Configuration
- **3240**$_h$: Digital Inputs Control
- **3242**$_h$: Digital Input Routing
- **3250**$_h$: Digital Outputs Control
- **3252**$_h$: Digital Output Routing
- **3321**$_h$: Analogue Input Offset
- **3322**$_h$: Analogue Input Pre-scaling
- **3700**$_h$: Following Error Option Code
- **4013**$_h$: HW Configuration
- **6040**$_h$: Controlword
- **6042**$_h$: Vl Target Velocity
- **6046**$_h$: Vl Velocity Min Max Amount
- **6048**$_h$: Vl Velocity Acceleration
- **6049**$_h$: Vl Velocity Deceleration
- **604A**$_h$: Vl Velocity Quick Stop
- **604C**$_h$: Vl Dimension Factor
- **605A**$_h$: Quick Stop Option Code
- **605B**$_h$: Shutdown Option Code
- **605C**$_h$: Disable Option Code
- **605D**$_h$: Halt Option Code
- **605E**$_h$: Fault Option Code
- **6060**$_h$: Modes Of Operation
- **6065**$_h$: Following Error Window
- **6066**$_h$: Following Error Time Out
- **6067**$_h$: Position Window
- **6068**$_h$: Position Window Time
- **606D**$_h$: Velocity Window
- **606E**$_h$: Velocity Window Time
- **6071**$_h$: Target Torque
- **6072**$_h$: Max Torque
- **607A**$_h$: Target Position
- **607B**$_h$: Position Range Limit
- **607C**$_h$: Home Offset
- **607D**$_h$: Software Position Limit
- **607E**$_h$: Polarity
- **6081**$_h$: Profile Velocity
- **6082**$_h$: End Velocity
- **6083**$_h$: Profile Acceleration
- **6084**$_h$: Profile Deceleration
- **6085**$_h$: Quick Stop Deceleration
- **6086**$_h$: Motion Profile Type
- **6087**$_h$: Torque Slope
- **608F**$_h$: Position Encoder Resolution
- **6091**$_h$: Gear Ratio
- **6092**$_h$: Feed Constant
- **6098**$_h$: Homing Method
- **6099**$_h$: Homing Speed
- **609A**$_h$: Homing Acceleration
- **60A4**$_h$: Profile Jerk

- **60C1**$_h$: Interpolation Data Record
- **60C2**$_h$: Interpolation Time Period
- **60C4**$_h$: Interpolation Data Configuration
- **60C5**$_h$: Max Acceleration
- **60C6**$_h$: Max Deceleration
- **60F2**$_h$: Positioning Option Code
- **60FE**$_h$: Digital Outputs
- **60FF**$_h$: Target Velocity

8.4.4 Kategorie: Benutzer

- **2701**$_h$: Customer Storage Area

8.4.5 Kategorie: Bewegung

- **3202**$_h$: Motor Drive Submode Select

8.4.6 Kategorie: Tuning

- **2030**$_h$: Pole Pair Count
- **2031**$_h$: Maximum Current
- **2032**$_h$: Maximum Speed
- **203B**$_h$: I2t Parameters
- **2050**$_h$: Encoder Alignment
- **2051**$_h$: Encoder Optimization
- **2052**$_h$: Encoder Resolution
- **2059**$_h$: Encoder Configuration

8.4.7 Speichervorgang starten

Hinweis

- Das Speichern kann einige Sekunden dauern. Unterbrechen Sie während des Speicherns keinesfalls die Spannungsversorgung. Andernfalls ist der Stand der gespeicherten Objekte undefiniert.
- Warten Sie immer, dass die Steuerung das erfolgreiche Speichern mit dem Wert "1" in dem entsprechenden Subindex im Objekt **1010**$_h$ signalisiert.

Für jede *Kategorie* gibt es einen Subindex im Objekt **1010**$_h$. Um alle Objekte dieser *Kategorie* zu speichern, muss der Wert "65766173$_h$" [1] in den Subindex geschrieben werden. Das Ende des Speichervorgangs wird signalisiert, indem der Wert von der Steuerung durch eine "1" überschrieben wird.

Nachfolgende Tabelle listet auf, welcher Subindex des Objektes **1010**$_h$ für welche *Kategorie* zuständig ist.

Subindex	Kategorie
01$_h$	Alle Kategorien
02$_h$	Kommunikation
03$_h$	Applikation
04$_h$	Benutzer

[1] Das entspricht dezimal der `1702257011`$_d$ bzw. dem ASCII String `save`.

Subindex	Kategorie
05_h	Bewegung
06_h	Tuning

8.4.8 Speicherung verwerfen

Falls alle Objekte oder eine *Kategorie* an gespeicherten Objekten gelöscht werden sollen, muss in das Objekt **1011_h** der Wert "$64616F6C_h$" [2] geschrieben werden. Folgende Subindizes entsprechen dabei einer *Kategorie*:

Subindex	Kategorie
01_h	Alle Kategorien (Zurücksetzen auf Werkseinstellung) mit der Ausnahme der Kategorie 06_h (Tuning)
02_h	Kommunikation
03_h	Applikation
04_h	Benutzer
05_h	Bewegung
06_h	Tuning

Die gespeicherten Objekte werden daraufhin verworfen. Nachdem die Daten gelöscht wurden, startet die Steuerung selbstständig neu.

Hinweis

Die Objekte der *Kategorie* 06_h (Tuning) werden vom **Auto-Setup** ermittelt und werden beim Zurücksetzen auf Werkseinstellungen mittels Subindex 01_h nicht zurückgesetzt (damit eine erneutes Auto-Setup nicht notwendig wird). Sie können diese Objekte mit Subindex 06_h zurücksetzen.

8.4.9 Konfiguration verifizieren

Das Objekt **1020_h** kann herangezogen werden, um die Konfiguration zu verifizieren. Es agiert wie ein Modifkationsmarker in üblichen Text-Editoren: Sobald eine Datei in dem Editor modifiziert wird, wird ein Marker (normalerweise ein Stern) hinzugefügt.

Die Einträge des Objektes **1020_h** können mit einem Datum und einer Uhrzeit beschrieben und anschließend über **1010_h**:01 zusammen mit allen anderen speicherbaren Objekten gespeichert werden.

Die Einträge von **1020_h** werden auf "0" zurückgesetzt, sobald ein beliebiges speicherbares Objekt (einschließlich **1010_h**:$0x,_h$ außer **1010_h**:01_h und **1020_h**) beschrieben wird.

Die folgende Reihenfolge macht die Verifikation möglich:

1. Ein externes Tool oder Master konfiguriert die Steuerung.
2. Das Tool oder der Master setzt den Wert in das Objekt **1020_h**.
3. Das Tool oder der Master aktiviert das Speichern aller Objekte **1010_h**:01_h = 65766173_h. Das Datum und die Uhrzeit im Objekt **1020_h** werden ebenfalls abgespeichert.

Nach einem Neustart der Steuerung kann der Master den Wert in **1020_h**:01_h und **1020**:01_h prüfen. Sollte einer der Werte "0" sein, wurde das Objektverzeichnis verändert, nachdem die gespeicherten Werte geladen wurden. Sollten das Datum oder die Uhrzeit in **1020** nicht den erwarteten Werten entsprechen, wurden Objekte wahrscheinlich mit anderen als den erwarteten Werten gespeichert.

[2] Das entspricht dezimal der 1684107116_d bzw. dem ASCII String `load`.

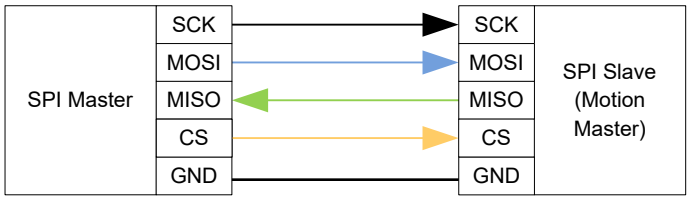
9 NanoSPI

Das Serial Peripheral Interface (kurz SPI) ist ein Bus-System für einen synchronen, seriellen Datenbus (Synchronous Serial Port), mit dem digitale Schaltungen nach dem Master-Slave-Prinzip miteinander verbunden werden können.

Im diesem Kapitel wird das von Nanotec entwickelte Protokoll beschrieben, dass Ihnen beispielsweise CANopen-SDO-Zugriffe via SPI ermöglicht. Das Protokoll ist eine Kombination aus EtherCAT und CANopen und dementsprechend ein Single-Master-Protokoll.

9.1 Bus-Topologie

Der SPI-Bus verwendet die Leitungen *SCK* (source clock), *MOSI* (master out, slave in), *MISO* (master in, slave out) und *CS* (chip select). Nachdem keine differentiellen Signale verwendet werden, ist der Anschluss von GND notwendig. Die folgende Grafik zeigt die Topologie im einfachen Fall eines einzigen Slaves.



Je nach Ausbaustufe lassen sich mehrere Slaves von einem Master aus steuern, siehe Kapitel **SPI-Sub-Master**.

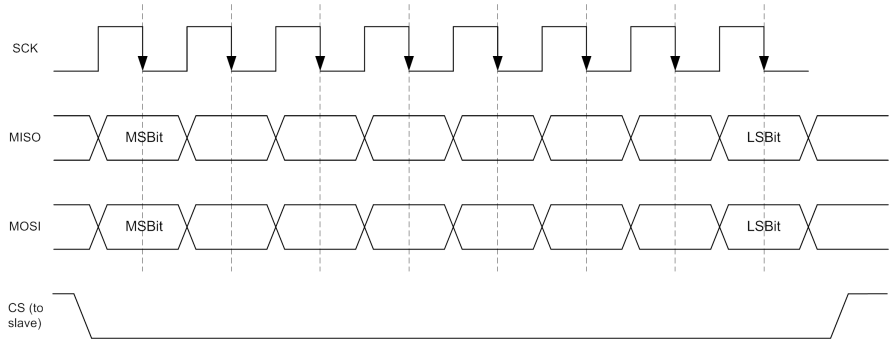
9.2 SPI-Einstellungen

Die SPI-Parameter sind folgendermaßen einzustellen (siehe auch nachfolgende Abbildung):

- Der Ruhepegel des Clock-Signals ist *low*.
- Die Bereitstellung eines Bitwertes (*MISO* und *MOSI*) geschieht auf der steigenden Flanke des Clock-Signals.
- Der Abtastzeitpunkt ist die fallende Flanke des Clock-Signals.
- Die Daten werden mit dem *Most Significant Bit* zuerst versendet und empfangen.
- Das *CS*-Signal ist *low* aktiv.
- Solange sich der SPI-Slave nicht auf den Millisekundentakt des SPI-Masters aufsynchronisiert hat, darf der SPI-Master nur alle zwei Millisekunden eine Nachricht übertragen.
  Wenn der SPI-Slave synchron zum Millisekundentakt des SPI-Masters läuft, darf der SPI-Master jede Millisekunde eine Nachricht übertragen.

Der *SPI-Slave* kann mit einer Frequenz von maximal 20 MHz angesteuert werden.

Folgende Abbildung zeigt den SPI-Signalverlauf:



9.3 Bus-Initialisierung

Die Slaves senden erst gültige Inhalte, nachdem einmalig eine korrekte Nachricht vom Master empfangen wurde. Die Bus-Initialisierung ist mit der ersten korrekt empfangenen Nachricht abgeschlossen.

9.4 Allgemeines zum Protokoll

Im Folgenden werden folgende Ausdrücke benutzt:

- *Nachricht* (engl. message) bedeutet, Daten werden an einen einzelnen Teilnehmer gesendet.
- *Übertragung* : mehrere logisch zusammengehörige *Nachrichten* sind eine *Übertragung*.
- *Mailbox* ist ein Datenbereich innerhalb einer *Nachricht*, der als Container Daten eines bestimmten Protokolls enthält (z. B. SDO-Protokoll). Die verfügbaren Protokolle sind festgelegt, aufeinanderfolgende Nachrichten müssen nicht immer das gleiche Protokoll in der *Mailbox* enthalten.
- *Abbild* ist ein Datenbereich in der *Nachricht*, der ausgewählte Daten aus dem Objektverzeichnis überträgt oder ausgewählte Daten in das Objektverzeichnis schreibt. Falls aktiv, wird dieses *Abbild* mit jeder Nachricht übertragen. Damit lassen sich wichtige Objekte aus dem Objektverzeichnis sehr gut überwachen.
  Die Auswahl der Daten geschieht vor Aktivierung des Abbilds mittels Protokolls aus der Mailbox und kann nur unter bestimmten Bedingungen wieder geändert werden.
- *Mapping* bedeutet eine Zuordnung der Daten innerhalb eines *Abbilds*.

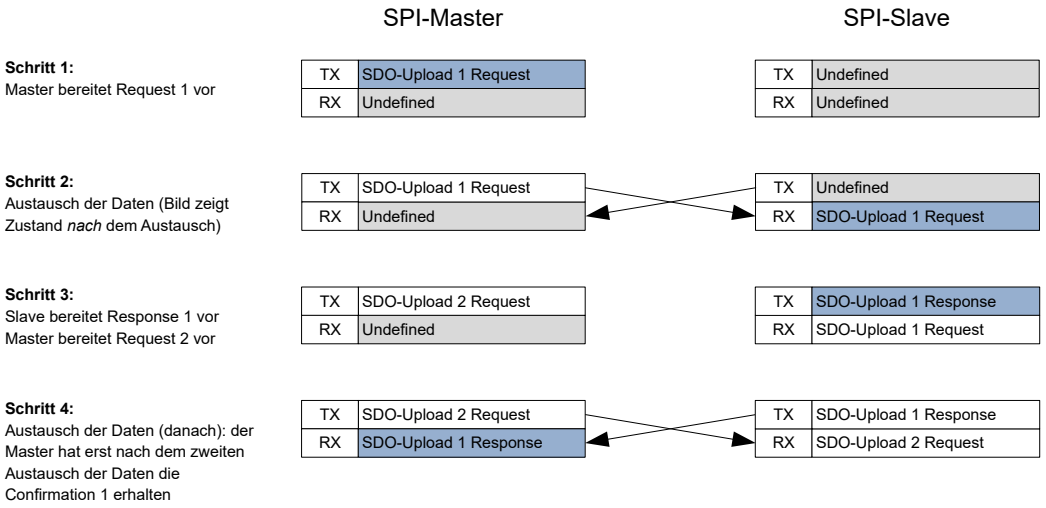
9.5 SPI-Nachricht

In einer SPI-Nachricht kann eine oder keine Mailbox eingebettet werden. Die möglichen Mailboxen werden nachfolgend beschrieben.

9.5.1 Mailbox Datenaustausch

Um eine Antwort auf eine Mailbox zu erhalten, muss der SPI-Master zwei Nachrichten übertragen. Folgende Abbildung zeigt den Sende- und Empfangsspeicherverlauf des Masters und des Slaves. Während der Übertragung der allerersten Nachricht am Bus sind dabei einige dieser Puffer inhaltlich nicht definiert.

Für die Antwort auf *Request 1* müssen zwei Nachrichten verschickt werden. Die zweite Nachricht kann dabei wieder einen neuen Request enthalten.



9.5.2 Nachrichtenhäufigkeit und Synchronisierung

Die Nachrichten können in folgender Häufigkeit ausgetauscht werden:

- Asynchroner Betrieb: höchstens alle zwei Millisekunden eine Nachricht
- Synchroner Betrieb: eine Nachricht pro Millisekunde

Die Synchronisation erfolgt im Zustand *Operational* des Slaves auf die Nachrichten des Masters. Dieser Vorgang kann initial bis zu 100 Millisekunden dauern. Erst bei aktiver Synchronisation werden die Abbilder der Nachrichten ausgewertet. Der Zustand *Operational* des Slaves wird erst angezeigt, wenn dieser sich synchronisiert hat. Bis dahin bleibt der Slave im Zustand *Init* und dem Master ist es nur gestattet, alle zwei Millisekunden eine Nachricht zu übertragen.

Wenn der Slave eine Sekunde lang keine Nachricht mehr vom Master empfangen hat, ist er wieder asynchron und schaltet in den *Init*-Status zurück.

Sollten die Nachrichten des Masters nicht in einem sauberen Millisekundenraster übertragen werden (zu großer Jitter), dann kann sich der Slave nicht aufsynchronisieren bzw. fällt frühestens nach 64 Nachrichten in den *Init*-Status zurück und ist dann wieder asynchron.

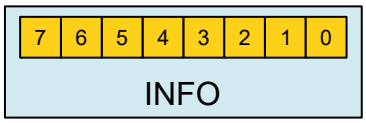
9.5.3 Aufbau einer SPI-Nachricht

Eine Nachricht besteht aus folgenden Bestandteilen:

- *INFO*-Byte: Beschreibt das benutzte Protokoll in der *Mailbox* und gibt den Bus-Status des Senders der Nachricht an (für Details: siehe **INFO-Byte**).
- *Mailbox* entsprechend dem *INFO*-Byte: siehe **CANopen Mailbox**
- *Abbild*: wenn aktiv, siehe **Abbild**
- *CRC*-Byte: siehe **CRC**

9.5.4 INFO-Byte

Das *INFO*-Byte setzt sich folgendermaßen zusammen:



Hinweis

Bit 5 bis 2 sind reserviert.

Bit 7-6	Bedeutung
0b00	Betriebszustand *Init*: • keine Tx/Rx-Abbilder zulässig
0b01	Betriebszustand *Operational* (sync): • Tx/Rx-Abbilder aktiv • CANopen *Mailbox* möglich • Synchroner Betrieb des Slaves
0b10	Betriebszustand *Operational* (async): • Tx/Rx-Abbilder aktiv • CANopen *Mailbox* möglich • Asynchroner Betrieb des Slaves
0b11	Betriebszustand *Error* • keine Tx/Rx-Abbilder zulässig • nur CANopen *Mailbox* möglich

Bit 1-0	Bedeutung (siehe auch CANopen Mailbox)
0b00	Keine *Mailbox*
0b01	CANopen Mailbox mit SDO-Protokoll (siehe Unterkapitel **CANopen SDO-Protokoll**)
0b10	CANopen *Mailbox* mit ungültigen 8 Datenbytes (Details: siehe Unterkapitel **CANopen Ungültige Daten**)
0b11	NanoSPI *Mailbox* (Details: siehe Unterkapitel **NanoSPI-Mailbox**)

9.5.5 CANopen Mailbox

CANopen SDO-Protokoll

Mittels dieser *Mailbox* wird das *SDO-Protokoll* des CANopen-Standards verwendet. Da keine anderen Services adressiert werden können, wird die *COB-ID* nicht mitgeschickt. Die Mailbox enthält demnach 8 Bytes einer SDO-Nachricht.

CANopen Ungültige Daten

Um die Bestätigung (*Confirmation*) auf eine Anfrage (*Request*) zu erhalten, müssen zwei SPI-Nachrichten verschickt werden, die erste mit dem *Request* und eine zweite um die *Response* zu

transportieren (siehe auch **Mailbox Datenaustausch**). Falls kein weiterer *Request* zu senden ist und nur die *Response* abgeholt werden soll, kann die Mailbox der zweiten Nachricht diesen Typ haben.

Die Daten innerhalb der *Mailbox* sind nicht relevant, auf diese Nachricht wird inhaltlich nicht reagiert.

9.5.6 NanoSPI-Mailbox

Über die NanoSPI-Mailbox können NanoJ-Programme übertragen werden. Pro Nachricht können dabei maximal 1024 Bytes Nutzdaten verschickt werden. Mehrere Nachrichten lassen sich zu einer Übertragung zusammenfassen. Eine *Mailbox* besteht aus den folgenden vier Teilen:

Byte Position	Name	Beschreibung
0	**Indication**	zur Anzeige der Inhalte, der letzten Nachricht der Übertragung, etc.
1	**Counter**	zum Nummerieren der Nachrichten innerhalb einer Übertragung. Der Überlauf des Zählers wird im Indication-Byte mit einem Wechsel des Werts des "Toggle Bits" bestätigt.
3-2	**Length**	Beinhaltet die Länge der im Datenbereich hinterlegten Daten (Einheit: Byte).
4 bis 1028	**Data**	Enthält die Daten (bis zu 1024 Bytes).

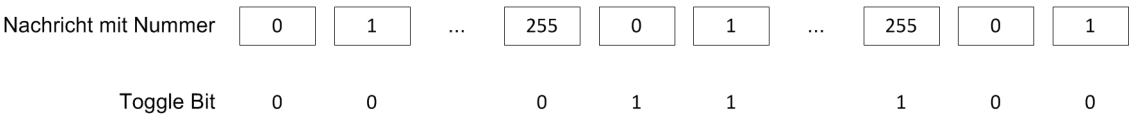
Indication

Das *Indication* Byte macht Angaben über den Inhalt und über die Übertragung. Die Bits sind in der nachfolgenden Tabelle aufgelistet.

Bit-Postion	Name	Beschreibung
1-0	DataType	Art der Daten: • Wert 1: NanoJ-Programm
2	Toggle	Jede Übertragung startet mit diesem Bit auf dem Wert "0". Bei jedem Überlauf des Counter-Bytes von "255" auf "0" muss der Zustand des Bits wechseln.
3	Last message	Zeigt die letzte Nachricht der momentanen Übertragung an.
4	Reset Comm	Setzt die Übertragung zurück.
7-5	Reserviert	Diese Bits müssen 0 sein.

Counter

Das *Counter* Byte nummeriert die Nachrichten. Bei jeder neuen Übertragung beginnt der Zähler mit `0`. Bei einem Überlauf von `255` auf `0` muss das *Toggle* Bit im *Indication* Byte den Zustand wechseln (siehe nachfolgende Abbildung).



Length

Length definiert die Länge des Datenbereichs (*Data*) in Bytes. Die Länge der Daten ist maximal 1024 Bytes.

Data

Data beinhaltet die Daten, die maximal übertragbare Datenmenge ist 1024 Bytes.

Beispiel

Im folgenden Beispiel soll ein NanoJ-Programm übertragen werden, welches aus 3204 Bytes besteht. Die Bytes mit den Wert `XX` sind für das Beispiel nicht relevant.

1. Senden der ersten 1024 Bytes eines NanoJ-Programms; Header: Mailboxtyp NanoSPI, Busstatus Init:
   Die erste Nachricht besteht aus folgenden Bytes:

   ```
   03 01 00 00 04 XX XX ... XX XX
   ```

   Die Bytes dieser Nachricht haben folgende Bedeutung:

   - Byte 0 = `0x03` (*Info* Byte): Es wird die NanoSPI-Mailbox genutzt, Busstatus ist *Init*.
   - Byte 1 = `0x01` (*Indication* Byte):
     - *Datentyp* ist NanoJ-Programm.
     - *Toggle*-Bit ist auf "0" gesetzt, da eine neue Übertragung erfolgt.
     - *LastFrame*-Bit ist auf "0" gesetzt, da noch weitere Datenpakete folgen werden.
     - *Reset Comm*-Bit ist auf "0" gesetzt.
   - Byte 2 = `0` (*Counter*): Es ist die erste Nachricht der Übertragung.
   - Byte 3 / 4 = `0x0400` (*Length*-Bytes): Byte 4 = `0x04`, Byte 3 = `0x00` was zusammengesetzt die Datenlänge von 1024 Bytes in der Mailbox bedeutet.
   - Byte 5 bis einschließlich Byte 1028: Das sind die ersten 1024 Bytes des NanoJ-Programms.
   - Byte 1029 = `0xXX` (*CRC* Byte)

2. Senden der zweiten 1024 Bytes eines NanoJ-Programms; Header: Mailboxtyp NanoSPI, Busstatus *Init*:

   ```
   03 01 01 00 04 XX XX ... XX XX
   ```

   Im Gegensatz zur ersten Nachricht hat sich nur das *Counter*-Byte auf `1` erhöht und die Daten sind mit den nächsten 1024 Bytes des NanoJ-Programms gefüllt.

3. Senden der dritten 1024 Bytes eines NanoJ-Programms; Header: Mailboxtyp NanoSPI, Busstatus *Init*:

   ```
   03 01 02 00 04 XX XX ... XX XX
   ```

   Im Gegensatz zur zweiten Nachricht ist nur der *Counter* erhöht worden, zudem sind die NanoJ-Daten die dritten 1024 Bytes des NanoJ-Programms.

4. Senden der letzten 132 Bytes eines NanoJ-Programms; Header: Mailboxtyp NanoSPI, Busstatus *Init*:

   ```
   03 09 03 84 00 XX XX ... XX XX
   ```

   Die Bytes der obigen Nachricht haben folgende Bedeutung:

   - Byte 0 = `0x03` (*Info* Byte): Es wird die Mailbox NanoSPI genutzt, Busstatus ist *Init*.
   - Byte 1 = `0x09` (*Indication* Byte):
     - *Datentyp* ist NanoJ-Programm.
     - *Toggle*-Bit auf "0" gesetzt.
     - *LastFrame*-Bit auf "1" gesetzt, da dies letzte Nachricht der Übertragung ist.
     - *Reset Comm*-Bit ist auf "0" gesetzt
   - Byte 2 = `3` (*Counter*): Es ist die vierte Nachricht der Übertragung.
   - Byte 3 / 4 = `0x0084` (*Length*-Bytes): Byte 4 = `0x00`, Byte 3 = `0x84`, was zusammengesetzt die Datenlänge von 132 Byte in der Mailbox bedeutet.

- Byte 5 bis einschließlich Byte 136: Sind die letzten 132 Bytes des NanoJ-Programms.
- Byte 137 = `0xXX` (*CRC* Byte)

9.5.7 Abbild

Um wichtige Objekte im Objektverzeichnis mit jeder Nachricht austauschen zu können, kann das *Abbild* verwendet werden. Das *Abbild* besteht nur noch aus Daten für das oder aus dem Objektverzeichnis. Meta-Informationen für die übermittelten Daten (also die Information *Index*, *Subindex* und *Länge*) für das Abbild werden vorab definiert und nicht mitgeschickt.

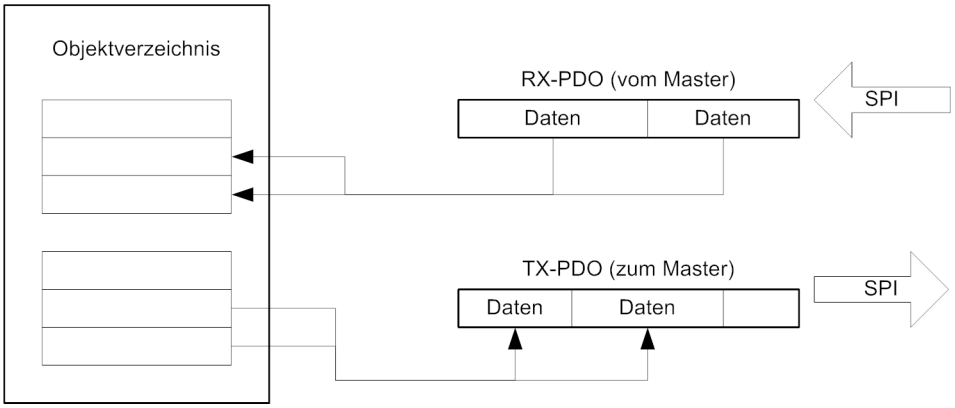
Das *Abbild* wird intern jede Millisekunde aktualisiert, beim Abholen der Daten sind alle Werte aktuell.

Generelles Prinzip

Grundsätzlich wird immer zwischen einem Abbild zum Empfang (*RX*) und einem zum Senden (*TX*) unterschieden.

- *RX* bezeichnet die Daten, die zyklisch von der jeweiligen Steuerung vom SPI-Bus empfangen werden und damit in das Objektverzeichnis des Geräts geschrieben werden.
- *TX* bezeichnet die Daten, die aus dem Objektverzeichnis der Steuerung gelesen und an den Master verschickt werden.

Die ankommenden Daten werden ins Objektverzeichnis kopiert, wie in der nachfolgenden Abbildung dargestellt. Anschließend wird das TX-Abbild zusammengestellt, das bei der nächsten Nachricht verschickt wird.



Die Zuordnung der Daten zu Objekten (das Mapping) wird in speziellen Objekten abgelegt.

Die Zuordnungen für den Empfang von Daten sind in den Objekten **1600**$_h$ bis **1603**$_h$ und **3500**$_h$ einzutragen.

Die Zuordnungen für das Senden von Daten sind in den Objekten **1A00**$_h$ bis **1A03**$_h$ und **3600**$_h$ einzutragen.

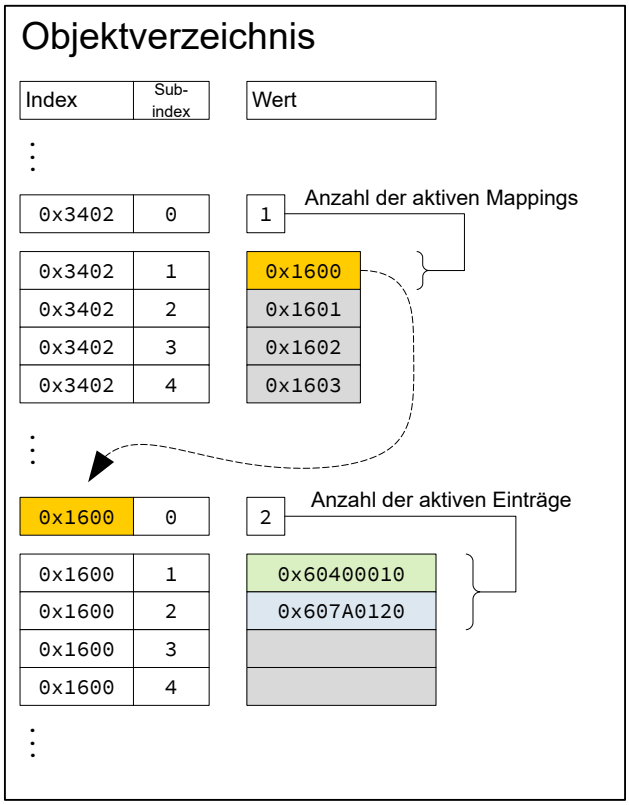
Das Mapping wird aktiv, sobald der SPI-Bus von *Init* auf *Operational* geschaltet wird. Für Änderungen muss der Bus zurück auf *Init* gesetzt werden, die Änderungen durchgeführt und im Anschluss der Bus wieder auf *Operational* geschaltet werden.

Anlegen eines Abbilds

Vier Objekte im Objektverzeichnis definieren in welchen Objekten das Mapping definiert ist :

- Zwei Objekte für den *RX*-Abbilder: Objekt **3402**$_h$:01$_h$ … **3402**$_h$:04$_h$ für die Schnittstelle *NanoSPI Ctrl (SLOT_SPI)*) bzw. Objekt **3400**$_h$:01$_h$ … **3400**$_h$:04$_h$ für die Schnittstelle *NanoSPI Comm (COMM_SPI)* definieren die aktiven *Mappings*.
  Die Objekte **1600**$_h$ bis **1603**$_h$ oder **3500**$_h$ beinhalten das *Mapping*.

- Zwei Objekte für den *TX*-Abbilder: Objekt **3403**$_h$:01$_h$ … **3403**$_h$:04$_h$ für die Schnittstelle *NanoSPI Ctrl (SLOT_SPI)* bzw. Objekt **3401**$_h$:01$_h$ … **3401**$_h$:04$_h$ für die Schnittstelle *NanoSPI Comm (COMM_SPI)* definieren die aktiven *Mappings*.
  Die Objekte **1A00**$_h$ bis **1A03**$_h$ oder **3600**$_h$ beinhalten das *Mapping*.



Beispiel:

Die Folgende Abbildung zeigt einen Ausschnitt aus dem Objektverzeichnis. Dabei sind alle relevanten Objekte für das *RX*-Abbild des *NanoSPI Ctrl (SLOT_SPI)* aufgezeichnet.

Objekt **3402**$_h$:00$_h$ definiert die Anzahl der aktiven Subeinträge. Im obigen Beispiel = `1`. d. h. nur der Subindex 01$_h$ ist aktiv.

Objekt **3402**$_h$:01$_h$ bis **3402**$_h$:04 definiert, wo das *Mapping* im Objektverzeichnis hinterlegt ist. In dem Beispiel ist nur der Subindex 01$_h$ aktiv, somit nur das Objekt **1600**$_h$.

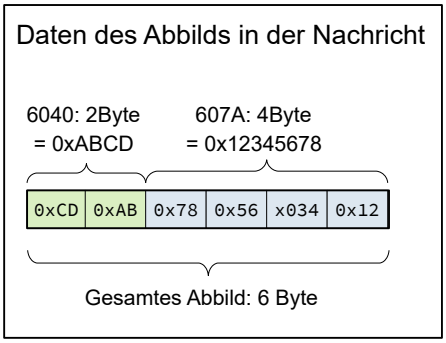
Das aktive Objekt bei **1600**$_h$:00 gibt wiederum an, wie viele der Subeinträge als aktiv gelten. In dem Beispiel sind die Einträge **1600**$_h$:01$_h$ und **1600**$_h$:02$_h$ aktiv. Dort die Informationen `60400010`$_h$ und `607A00120`$_h$ hinterlegt. So ein Mapping-Eintrag wird wie folgt aufgebaut:

- Die oberen beiden Bytes des Eintrags entsprechen dem Index des abzubildenden Objekts
- Das folgende Byte gibt den Subindex des abzubildenden Objekts vor
- Das untere Byte gibt die Bit-Größe des abzubildenden Objekts an

Der Zahlenwert `60400010`$_h$ in einem Mapping ergibt demnach

Index	Subindex	Länge in Bit
6040	00	10
2Byte	1Byte	1Byte

Das zu dem Beispiel aus der vorherigen Abbildung zugehörige Datenpaket ist nachfolgend dargestellt, die Zahlenwerte wie `0xABCD` sind nur Beispiele.



Vorbelegung

Die in den nachfolgenden Tabellen aufgelisteten Werte sind beim Start der Steuerung vorbelegt.

Index	Subindex	aktives Rx Mapping
3400_h	01_h	1600_h
3400_h	02_h	1601_h
3402_h	01_h	1600_h
3402_h	02_h	1601_h

Index	Subindex	Ziel
1600_h	01_h	**6060h Modes Of Operation**
1600_h	02_h	**6040h Controlword**
1601_h	01_h	**607Ah Target Position**
1601_h	02_h	**6042h Vl Target Velocity**
1601_h	03_h	**60FFh Target Velocity**
1601_h	04_h	**6071h Target Torque**
1601_h	05_h	**6098h Homing Method**

Index	Subindex	Ziel
3500_h	01_h	**3416h NanoSPI Slave Rx PDO Data**:01_h
3500_h	02_h	**3416h NanoSPI Slave Rx PDO Data**:02_h
3500_h	03_h	**3416h NanoSPI Slave Rx PDO Data**:03_h
3500_h	04_h	**3416h NanoSPI Slave Rx PDO Data**:04_h
3500_h	05_h	**3416h NanoSPI Slave Rx PDO Data**:05_h
3500_h	06_h	**3416h NanoSPI Slave Rx PDO Data**:06_h
3500_h	07_h	**3416h NanoSPI Slave Rx PDO Data**:07_h
3500_h	08_h	**3416h NanoSPI Slave Rx PDO Data**:08_h
3500_h	09_h	**3416h NanoSPI Slave Rx PDO Data**:09_h
3500_h	$0A_h$	**3416h NanoSPI Slave Rx PDO Data**:$0A_h$
3500_h	$0B_h$	**3416h NanoSPI Slave Rx PDO Data**:$0B_h$

Index	Subindex	aktives Tx Mapping
3401_h	01_h	$1A00_h$
3401_h	02_h	$1A01_h$
3403_h	01_h	$1A00_h$
3403_h	02_h	$1A01_h$

Index	Subindex	Ziel
$1A00_h$	01_h	**6061h Modes Of Operation Display**
$1A00_h$	02_h	**6041h Statusword**
$1A00_h$	03_h	**1001h Error Register**
$1A01_h$	01_h	**6062h Position Demand Value**
$1A01_h$	02_h	**6064h Position Actual Value**
$1A01_h$	03_h	**60F4h Following Error Actual Value**
$1A01_h$	04_h	**6043h Vl Velocity Demand**
$1A01_h$	05_h	**6044h Vl Velocity Actual Value**
$1A01_h$	06_h	**606Bh Velocity Demand Value**
$1A01_h$	07_h	**606Ch Velocity Actual Value**
$1A01_h$	08_h	**6077h Torque Actual Value**

Index	Subindex	Ziel
3600_h	01_h	**3417h NanoSPI Slave Tx PDO Data**:01_h
3600_h	02_h	**3417h NanoSPI Slave Tx PDO Data**:02_h
3600_h	03_h	**3417h NanoSPI Slave Tx PDO Data**:03_h
3600_h	04_h	**3417h NanoSPI Slave Tx PDO Data**:04_h
3600_h	05_h	**3417h NanoSPI Slave Tx PDO Data**:05_h
3600_h	06_h	**3417h NanoSPI Slave Tx PDO Data**:06_h
3600_h	07_h	**3417h NanoSPI Slave Tx PDO Data**:07_h

Beispiel

In diesem Beispiel wird folgendes Szenario verwendet:

- Der Anwender will mehrere geschwindigkeitsgesteuerte Fahrten im *Profile Velocity Mode* durchführen.
- Alle nachfolgenden Befehle sind vom *Master* aus gesehen.

Das Beispiel gliedert sich in zwei Punkte:

1. Vorbereitung: Hier wird das Mapping des Slaves angelegt, die Steuerung in den *Profile Velocity Mode* geschaltet und anschließend die *Power State Machine* aktiviert, siehe **CiA 402 Power State Machine**.
2. Benutzung: Hier wird der laufende Betrieb erläutert.

Vorbereitung

Für den *Profile Velocity Mode* ist es sinnvoll, dass der *Master* per *Abbild* folgende Daten erhält und versendet:

- *TX-Mapping* (Daten, die von dem Master an den Slave verschickt werden): *Controlword* 6040_h:00_h) zur Steuerung des Slaves und die *Target Velocity* ($60FF_h$:00_h) zum Vorgeben einer Zielgeschwindigkeit.

- *RX-Mapping* (Daten, die von dem Slave an den Master verschickt werden): *Statusword* (6041_h:00_h) zur Überwachung des Slaves und die aktuelle Geschwindigkeit (*Velocity actual value*, $606C_h$:00_h).

TX Mapping des Masters

Daten, die der Master an den Slave verschickt, müssen in das *RX-Mapping* des Slaves eingetragen werden.

Das *RX-Mapping* wird im Objekt 1600_h hinterlegt (die Objekte 1601_h bis 1603_h werden in diesem Beispiel nicht genutzt).

- Setzen des 1600_h:00_h auf den Wert "02_h" (Anzahl der Mappings = "2"); Header: Mailboxtyp CANopen, Busstatus *Init*, deshalb kein Mapping:
  - Nachricht Master an Slave: `01 2F 00 16 00 02 00 00 00 18`
  - Nachricht Slave an Master: `01 60 00 16 00 00 00 00 00 AC`

Hinweis

Für den Erhalt einer Antwort muss eine weitere Nachricht verschickt werden, siehe **SPI-Nachricht**! Diese wird in den Beispielen nicht aufgelistet.

- Setzen des 1600_h:01_h auf den Wert "60400010_h" (Mapping: *Controlword*); Header: Mailboxtyp CANopen, Busstatus *Init*, deshalb kein Abbild
  - Nachricht Master an Slave: `01 23 00 16 01 10 00 40 60 2B`
  - Antwort Slave an Master: `01 60 00 16 01 00 00 00 00 61`

- Setzen des 1600_h:02_h auf den Wert "$60FF0020_h$" (Mapping: *Target Velocity*); Header: Mailboxtyp CANopen, Busstatus *Init*, deshalb kein Abbild
  - Nachricht Master an Slave: `01 23 00 16 02 20 00 FF 60 37`
  - Antwort Slave an Master: `01 60 00 16 02 00 00 00 00 2F`

- Setzen des 3402_h:00_h auf den Wert "01_h" (Anzahl aktiver Mappings = "1"); Header: Mailboxtyp CANopen, Busstatus *Init*, deshalb kein Abbild
  - Nachricht Master an Slave: `01 2F 02 34 00 01 00 00 00 32`
  - Antwort Slave an Master: `01 60 00 16 00 00 00 00 00 AC`

- Setzen des 3402_h:01_h auf den Wert "1600_h" (Aktives Mapping-Objekt = 1600_h); Header: Mailboxtyp CANopen, Busstatus *Init*, deshalb kein Abbild
  - Nachricht Master an Slave: `01 2B 02 34 01 00 16 00 00 FE`
  - Antwort Slave an Master: `01 60 02 34 01 00 00 00 00 00`

RX Mapping des Masters

Daten, die vom Slave an den Master verschickt werden, müssen in das TX-Mapping des Slaves eingetragen werden.

Das TX-Mapping wird im Objekt $1A00_h$ hinterlegt (die Objekte $1A01_h$ bis $1A03_h$ werden in diesem Beispiel nicht genutzt).

- Setzen des $1A00_h$:00_h auf den Wert "02_h" (Anzahl der Mappings = "2"); Header: Mailboxtyp CANopen, Busstatus *Init*, deshalb kein Abbild
  - Nachricht Master an Slave: `01 2F 00 1A 00 02 00 00 00 65`
  - Antwort Slave an Master: `01 60 00 1A 00 00 00 00 00 D1`

- Setzen des $1A00_h$:01_h auf den Wert "60410010_h" (Mapping: Statusword); Header: Mailboxtyp CANopen, Busstatus *Init*, deshalb kein Abbild
  - Nachricht Master an Slave: `01 23 00 1A 01 10 00 41 60 92`
  - Antwort Slave an Master: `01 60 00 1A 01 00 00 00 00 1C`

- Setzen des $1A00_h$:02_h auf den Wert "$606C0020_h$" (Mapping: Velocity actual value); Header: Mailboxtyp CANopen, Busstatus *Init*, deshalb kein Abbild
  - Nachricht Master an Slave: `01 23 00 1A 02 20 00 6C 60 DC`
  - Antwort Slave an Master: `01 60 00 1A 02 00 00 00 00 52`

- Setzen des 3403_h:00_h auf den Wert "01_h" (Anzahl aktiver Mappings = "1"); Header: Mailboxtyp CANopen, Busstatus *Init*, deshalb kein Abbild
  - Nachricht Master an Slave: `01 2F 03 34 00 01 00 00 00 0F`
  - Antwort Slave an Master: `01 60 03 34 00 00 00 00 00 33`

Sonstige Einstellung und Aktivierung

An dieser Stelle wird das Objekt *Mode of operation* (**6060_h**:00_h) auf den Wert "03_h" gesetzt, um den *Profile Velocity Mode* auszuwählen, siehe **Profile Velocity**.

Setzen des 6060_h:00 auf den Wert "03_h" (*Mode of operation* = *Profile Velocity*); Header: Mailboxtyp CANopen, Busstatus Init, deshalb kein Abbild

- Nachricht Master an Slave: `01 2F 60 60 00 03 00 00 00 95`
- Antwort Slave an Master: `01 60 60 60 00 00 00 00 00 AE`

Das Mapping wird aktiv, sobald der SPI-Bus von *Init* auf *Operational* geschaltet wird. Für Änderungen muss der Bus zurück auf *Init* gesetzt werden, die Änderungen durchgeführt und im Anschluss der Bus wieder auf *Operational* geschaltet werden.

Betrieb

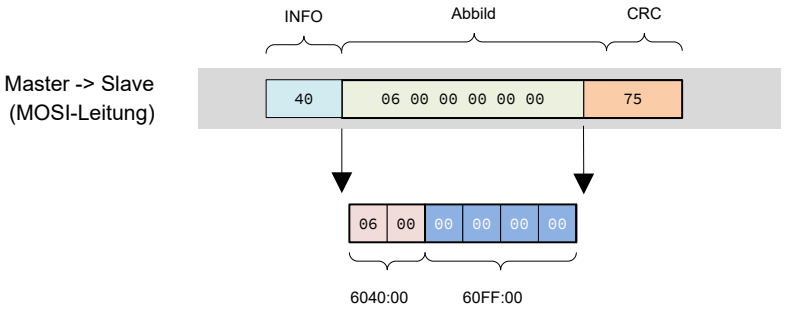
Der Steuerung lässt sich jetzt mit dem Abbild direkt Werte vorgeben. Um den Motor einzuschalten, ist es nötig, das *Controlword* erst auf den Wert "6", dann auf "7" und zuletzt auf "15" zu setzen.

- Schalten der Controlwords 6040_h:00_h auf "06_h"; Header: keine Mailbox, Busstatus *Operational*, Mapping vorhanden: 6040_h:00_h = 06_h, $60FF_h$:00_h = 0000_h

  Nachricht Master an Slave:

  `40 06 00 00 00 00 00 75`

  Diese Nachricht enthält ein *Abbild*, die nachfolgende Abbildung zeigt die einzelnen Bytes.



- Controlword 6040_h:00_h auf "07_h" schalten; Header: keine Mailbox, Busstatus *Operational*, Mapping vorhanden: 6040_h:00_h = 07_h, $60FF_h$:00_h = 0000_h
  Nachricht Master an Slave: `40 07 00 00 00 00 00 42`

- Controlword 6040_h:00_h auf "$0F_h$" schalten; Header: keine Mailbox, Busstatus *Operational*, Mapping vorhanden: 6040_h:00_h = $0F_h$, $60FF_h$:00_h = 0000_h
  Nachricht Master an Slave: `40 0F 00 00 00 00 00 E3`

Im folgendem Beispiel wird die Geschwindigkeit auf "200" gesetzt:

Controlword 6040_h:00_h auf "$0F_h$" und $60FF_h$:00_h auf "200" (="$1F4_h$") schalten; Header: keine Mailbox, Busstatus *Operational*, Mapping vorhanden:

Nachricht Master an Slave: `40 0F 00 F4 01 00 00 37`

9.5.8 CRC

Für den **C**yclic **R**edundancy **C**heck (CRC) wird das Polynom x^8+x^5+x^4+x^0 verwendet. Der Startwert ist 0 (siehe auch Maxim 1-Wire 8-Bit CRC). Die CRC wird über das *INFO*-Byte, die *Mailbox*-Daten und *Abbild*-Daten berechnet.

Die CRC kann auch mit dem Codeabschnitt aus nachfolgendem Listing berechnet werden.

```
uint8_t crc_array[256] = { 0x00, 0x5e, 0xbc, 0xe2, 0x61, 0x3f, 0xdd, 0x83,
0xc2, 0x9c, 0x7e, 0x20, 0xa3, 0xfd, 0x1f, 0x41, 0x9d, 0xc3, 0x21, 0x7f,
0xfc, 0xa2, 0x40, 0x1e, 0x5f, 0x01, 0xe3, 0xbd, 0x3e, 0x60, 0x82, 0xdc,
0x23, 0x7d, 0x9f, 0xc1, 0x42, 0x1c, 0xfe, 0xa0, 0xe1, 0xbf, 0x5d, 0x03,
0x80, 0xde, 0x3c, 0x62, 0xbe, 0xe0, 0x02, 0x5c, 0xdf, 0x81, 0x63, 0x3d,
0x7c, 0x22, 0xc0, 0x9e, 0x1d, 0x43, 0xa1, 0xff, 0x46, 0x18, 0xfa, 0xa4,
0x27, 0x79, 0x9b, 0xc5, 0x84, 0xda, 0x38, 0x66, 0xe5, 0xbb, 0x59, 0x07,
0xdb, 0x85, 0x67, 0x39, 0xba, 0xe4, 0x06, 0x58, 0x19, 0x47, 0xa5, 0xfb,
0x78, 0x26, 0xc4, 0x9a, 0x65, 0x3b, 0xd9, 0x87, 0x04, 0x5a, 0xb8, 0xe6,
0xa7, 0xf9, 0x1b, 0x45, 0xc6, 0x98, 0x7a, 0x24, 0xf8, 0xa6, 0x44, 0x1a,
0x99, 0xc7, 0x25, 0x7b, 0x3a, 0x64, 0x86, 0xd8, 0x5b, 0x05, 0xe7, 0xb9,
0x8c, 0xd2, 0x30, 0x6e, 0xed, 0xb3, 0x51, 0x0f, 0x4e, 0x10, 0xf2, 0xac,
0x2f, 0x71, 0x93, 0xcd, 0x11, 0x4f, 0xad, 0xf3, 0x70, 0x2e, 0xcc, 0x92,
0xd3, 0x8d, 0x6f, 0x31, 0xb2, 0xec, 0x0e, 0x50, 0xaf, 0xf1, 0x13, 0x4d,
0xce, 0x90, 0x72, 0x2c, 0x6d, 0x33, 0xd1, 0x8f, 0x0c, 0x52, 0xb0, 0xee,
0x32, 0x6c, 0x8e, 0xd0, 0x53, 0x0d, 0xef, 0xb1, 0xf0, 0xae, 0x4c, 0x12,
0x91, 0xcf, 0x2d, 0x73, 0xca, 0x94, 0x76, 0x28, 0xab, 0xf5, 0x17, 0x49,
0x08, 0x56, 0xb4, 0xea, 0x69, 0x37, 0xd5, 0x8b, 0x57, 0x09, 0xeb, 0xb5,
0x36, 0x68, 0x8a, 0xd4, 0x95, 0xcb, 0x29, 0x77, 0xf4, 0xaa, 0x48, 0x16,
0xe9, 0xb7, 0x55, 0x0b, 0x88, 0xd6, 0x34, 0x6a, 0x2b, 0x75, 0x97, 0xc9,
0x4a, 0x14, 0xf6, 0xa8, 0x74, 0x2a, 0xc8, 0x96, 0x15, 0x4b, 0xa9, 0xf7,
0xb6, 0xe8, 0x0a, 0x54, 0xd7, 0x89, 0x6b, 0x35, };

uint8_t Calculate8BitBlockCrc( uint8_t *data, uint16_t length )
```

```
{
 uint8_t initValue = 0;
 uint8_t i;
 for( i=0; i<length; ++i )
 {
  initValue = crc_array[data[i] ^ initValue];
 }
 return initValue;
}
```

9.6 SPI-Slave Verhalten im Fehlerfall

Falls der *Master* an den *Slave* einen *Error State* sendet, geht der *Slave* in den *Init*-Status.

Sollte der *Slave* einen Fehler in der Nachricht erkennen (beispielsweise ein CRC-Fehler), wird der *Slave* in seiner nächsten Antwortnachricht im Info-Byte den *Error-State* mit einer CANopen Mailbox signalisieren, die dann eine SDO-Abort-Nachricht enthält, und in den *Init-State* umschalten. Mit der nächsten Nachricht vom *Master* wird er dessen Vorgaben wieder folgen.

9.7 SPI-Sub-Master

Mittels des *SPI-Sub-Master-Betriebs* können Sie zwei Steuerungen kaskadiert an einem Master betreiben. Der Master steuert den *Sub-Master* direkt und den *Sub-Slave* indirekt.

9.7.1 Statusword und Controlword

Der *Sub-Master* besitzt ein *Statusword* und ein *Controlword*. Mit dem *Controlword* lässt sich der *Sub-Master* ein- und ausschalten sowie in einen der Zustände *Init* oder *Operational* bringen. Im *Statusword* lässt sich der Status des *Sub-Masters* und des *Sub-Slaves* auslesen.

9.7.2 Zustände des Sub-Masters

Der Sub-Master kann sich in einem von drei verschiedenen Zuständen befinden:

- **Init**:
  - *Sub-Slave* kann mit CANopen Nachrichten versorgt werden.
  - Das Abbild wird nicht verschickt und kann konfiguriert werden.
  - Keine Synchronisation
- **Operational**:
  - *Sub-Slave* kann mit CANopen Nachrichten versorgt werden.
  - Das Abbild wird verschickt.
  - Synchronisation zwischen Sub-Master und Sub-Slave

Der *Master* kann selbstständig auf den Status *Operational* schalten, dazu muss das Bit 1 *Managed Slave* des *Controlwords* 3410_h:00_h auf `1` gesetzt werden (siehe **3410h NanoSPI Comm Controlword**).

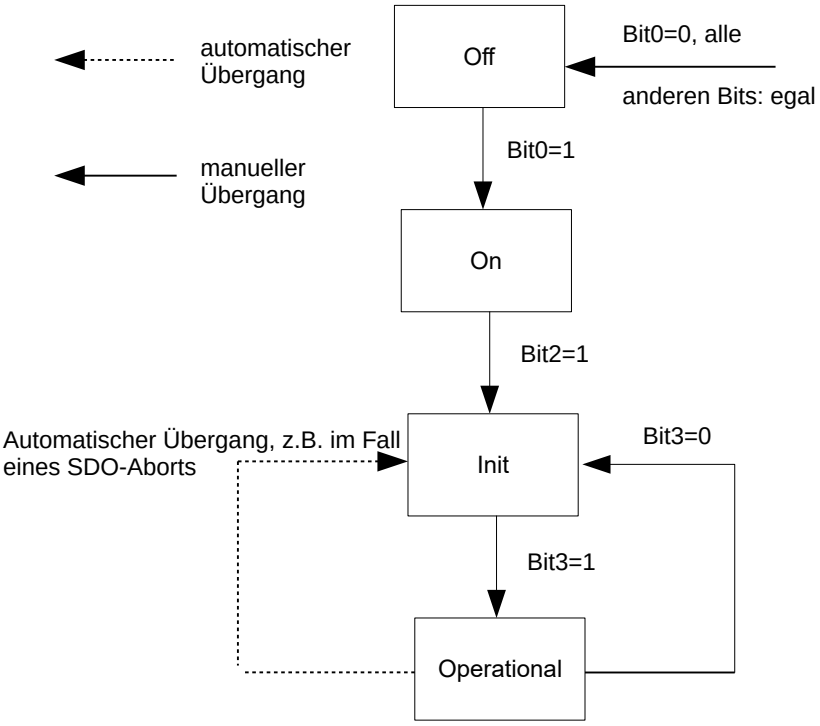
9.7.3 Controlword

Das *Controlword* liegt im Objektverzeichnis im Eintrag 3410_h:00_h (siehe **3410h NanoSPI Comm Controlword**).

Nach Einschalten des Mikrocontrollers ist der *Sub-Master* standardmäßig deaktiviert. Bevor er genutzt werden kann, muss er eingeschalten werden (Bit 0 = "1").

Zudem kann festgelegt werden, ob der *Master* die Zustände selbständig bis zum Status *Operational* durchläuft (Bit 1 = "1") oder ob der *Sub-Master* über weitere Bits von außen in die entsprechenden Zustände geführt werden soll (Bit 1 = "0"). Bei einem selbständigen Durchlauf ist es nicht möglich, das Mapping des Sub-Slaves noch zu konfigurieren.

Die Bits 2 und 3 schalten den *Sub-Master* in den jeweiligen Zustand *Init* und *Operational*. Folgende Abbildung zeigt die Übergänge mit den jeweiligen Bits des *Controlwords*.



9.7.4 Statusword

Das *Statusword* 3411_h (**3411h NanoSPI Comm Statusword**) zeigt den jeweiligen Status des *Sub-Masters* und des *Sub-Slaves* an. Das *Statusword* ist zweigeteilt: das LSB hält den Status des *Sub-Masters*, das MSB enthält den Status des *Sub-Slaves*.

9.8 Sub-Slave Kommunikation

Befehle an den *Sub-Slave* werden über die Objekte 3410_h bis 3417_h übertragen, siehe **3410h NanoSPI Comm Controlword** bis **3417h NanoSPI Slave Tx PDO Data**.

9.8.1 Senden

Zum Senden einer Nachricht, muss die CANopen-Mailbox des *Sub-Masters* genutzt werden. Dieser muss aktiviert sein.

Die Zusammenstellung der Nachricht lässt sich auf zwei Arten erreichen:

- Objekt 3413_h wird mit allen Informationen gefüllt (Index, Subindex, Länge, Wert) und Bit 1 des Objekts 3412_h wird auf "0" zum Lesen und "1" zum Schreiben gesetzt, siehe **3413h NanoSPI SDO Request** und **3412h NanoSPI SDO Control**.

- Eine fertige SDO-Nachricht mit 8 Bytes wird in 3414_h eingetragen , siehe **3414h NanoSPI SDO Raw Request**. Das reduziert die Anzahl der OD-Zugriffe, allerdings muss der Anwender die Bits und Bytes der CANopen Nachricht selbst zusammenstellen.

Verschickt wird die Nachricht indem Bit 0 im Objekt 3412_h:00 auf "1" gesetzt wird, wobei Bit 2 definiert, ob die Nachricht aus 3413_h:00 (Bit 2 ist "0") oder 3414_h:00 (Bit 2 ist "1") verschickt wird, siehe **3412h NanoSPI SDO Control**.

Der *Sub-Master* erledigt den Versand der Nachricht und setzt das Bit 0 in 3412_h zurück, die Antwort liegt im Objekt 3415_h sobald das Bit 3 des Objekts 3412_h auf "1" gewechselt hat, siehe **3415h NanoSPI SDO Response** und **3412h NanoSPI SDO Control**.

9.8.2 Ausfüllen einer SDO-Nachricht

Objekt 3413_h enthält alle Speicherplätze für eine vollständige SDO-Nachricht, siehe **3413h NanoSPI SDO Request**. Folgende Informationen sind beim Versenden wichtig:

- $3413_h:01_h$ (1 Byte, rw): SDO Header, wird beim Versenden automatisch ausgefüllt, sollte nicht beschrieben werden
- $3413_h:02_h$ (2 Byte, rw): Index des zu schreibenden Objekts
- $3413_h:03_h$ (1 Byte, rw): Subindex des zu schreibenden Objekts
- $3413_h:04_h$ (1 Byte, rw): Länge der Daten in Bytes
- $3413_h:05_h$ (4 Byte, rw): Daten

Anschließend kann das Objekt verschickt werden, siehe **Senden einer vorbereiteten Nachricht**.

9.8.3 Senden einer vorbereiteten Nachricht

Existiert eine fertige SDO-Nachricht , lässt sie sich in die beiden Subindizes des Objekts $3414_h:01_h$ und $3414_h:02_h$ schreiben, siehe **3414h NanoSPI SDO Raw Request**. Anschließend kann die Nachricht versendet werden.

Tipp

Objekt $3414_h:01_h$ enthält dabei die MSBs der Nachricht, Objekt $3414_h:02_h$ die LSBs.

10 Programmierung mit *NanoJ*

NanoJ ist eine *C*- bzw. *C*++-nahe Programmiersprache. NanoJ ist in der Software *Plug & Drive Studio* integriert. Weiterführende Informationen finden Sie im Dokument *Plug & Drive Studio: Quick Start Guide* auf **www.nanotec.de**.

10.1 NanoJ-Programm

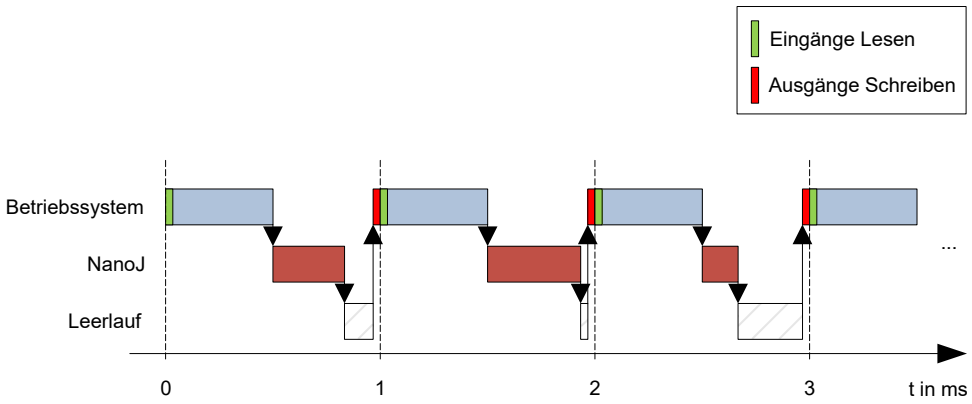
Ein *NanoJ-Programm* stellt eine geschützte Ausführungsumgebung innerhalb der Firmware zur Verfügung. In dieser kann der Anwender eigene Abläufe anlegen. Diese können dann Funktionen in der Steuerung auslösen, indem beispielsweise Einträge im Objektverzeichnis gelesen oder geschrieben werden.

Durch Verwendung von Schutzmechanismen wird verhindert, dass ein *NanoJ-Programm* die Firmware zum Absturz bringt. Im schlimmsten Fall wird die Ausführung mit einem im Objektverzeichnis hinterlegten Fehlercode abgebrochen.

Wenn das *NanoJ-Programm* auf die Steuerung geladen wurde, wird es nach dem Einschalten oder Neustarten der Steuerung automatisch ausgeführt.

10.1.1 Verfügbare Rechenzeit

Ein *NanoJ-Programm* erhält zyklisch im 1 ms-Takt Rechenzeit (siehe folgende Abbildung). Da durch Interrupts und Systemfunktionen der Firmware Rechenzeit verloren geht, stehen dem Benutzerprogramm (abhängig von Betriebsart und Anwendungsfall) nur ca. 30% … 50% Rechenzeit zur Verfügung. In dieser Zeit muss das Benutzerprogramm den Zyklus durchlaufen und entweder beenden oder durch Aufruf der Funktion `yield()` die Rechenzeit abgeben. Bei Ersterem wird das Benutzerprogramm mit dem Beginn des nächsten 1 ms-Zyklus wieder neu gestartet, letzteres bewirkt eine Fortsetzung des Programms an dem der Funktion `yield()` nachfolgenden Befehl beim nächsten 1 ms-Zyklus.



Falls das *NanoJ-Programm* mehr als die ihm zugeteilte Zeit benötigt, wird es beendet und im Objektverzeichnis ein Fehlercode gesetzt.

Tipp

Bei der Entwicklung von Benutzerprogrammen ist speziell bei zeitintensiveren Aufgaben eine sorgfältige Überprüfung des Laufzeitverhaltens durchzuführen. So empfiehlt sich beispielsweise die Verwendung von Tabellen, anstatt einen Sinuswert über eine `sin` Funktion zu berechnen.

Hinweis

Sollte das *NanoJ-Programm* zu lange die Rechenzeit nicht abgeben, wird es vom Betriebssystem beendet. In diesem Fall wird in das Statusword bei Objekt 2301_h die Ziffer `4` eingetragen, im Fehlerregister bei Objekt 2302_h wird die Ziffer `5` (Timeout) notiert, siehe **2301h NanoJ Status** und **2302h NanoJ Error Code**.

10.1.2 Sandbox

Durch prozessorspezifische Eigenschaften wird eine sogenannte *Sandbox* generiert. Ein Benutzerprogramm in der Sandbox hat nur die Möglichkeit, auf speziell zugewiesene Speicherbereiche und Systemressourcen zuzugreifen. Beispielsweise wird ein Versuch, auf ein Prozessor-IO-Register direkt zu schreiben, mit einem *MPU Fault* quittiert und das Benutzerprogramm wird mit dem entsprechenden Fehlercode im Objektverzeichnis abgebrochen.

10.1.3 NanoJ-Programm - Kommunikationsmöglichkeiten

Ein *NanoJ-Programm* hat mehrere Möglichkeiten, mit der Steuerung zu kommunizieren:

- Lesen und Schreiben von OD-Werten per PDO-Mapping
- direktes Lesen und Schreiben von OD-Werten über Systemcalls
- Aufruf sonstiger Systemcalls (z. B. Debug-Ausgabe schreiben)

Über ein *PDO-Mapping* werden dem Benutzerprogramm OD-Werte in Form von Variablen zur Verfügung gestellt. Bevor ein Benutzerprogramm die 1 ms-Zeitscheibe erhält, werden dazu von der Firmware die Werte aus dem Objektverzeichnis in die Variablen des Benutzerprogramms übertragen. Sobald das Benutzerprogramm Rechenzeit erhält, kann es diese Variablen wie gewöhnliche C-Variablen manipulieren. Am Ende der Zeitscheibe werden letztendlich die neuen Werte von der Firmware wieder automatisch in die jeweiligen OD-Einträge kopiert.

Um die Performance zu optimieren, werden dabei drei Arten von Mappings definiert: Input, Output und Input/Output (In, Out, InOut).

- *Input Mappings* lassen sich nur lesen und werden nicht zurück ins Objektverzeichnis übertragen.
- *Output Mappings* lassen sich nur schreiben.
- *Input/Output Mappings* erlauben hingegen Lesen und Schreiben.

Die gesetzten Mappings können über die GUI bei den Objekten 2310_h, 2320_h, und 2330_h ausgelesen und überprüft werden. Für jedes Mapping sind maximal 16 Einträge erlaubt.

Über die Angabe der *Linker-Section* wird in *NanoJEasy* gesteuert, ob eine Variable im Input-, Output- oder Datenbereich abgelegt wird.

10.1.4 NanoJ-Programm ausführen

Zusammengefasst besteht das *NanoJ-Programm* bei der Ausführung eines Zyklus hinsichtlich des PDO-Mappings aus folgenden drei Schritten:

1. Werte aus dem Objektverzeichnis lesen und in die Bereiche Inputs und Outputs kopieren
2. Benutzerprogramm ausführen
3. Werte aus den Bereichen Outputs und Inputs zurück in das Objektverzeichnis kopieren

Die Konfiguration der Kopiervorgänge ist dem CANopen-Standard angelehnt.

Zusätzlich kann über Systemcalls auf Werte des Objektverzeichnisses zugegriffen werden. Dies ist im Allgemeinen deutlich langsamer und daher sind Mappings vorzuziehen. Die Anzahl an Mappings ist begrenzt (jeweils 16 Einträge in In/Out/InOut).

Tipp

Nanotec empfiehlt: Häufig genutzte und veränderte OD-Einträge mappen und auf weniger häufig genutzte OD-Einträge per Systemcall zuzugreifen.

Eine Liste verfügbarer Systemcalls findet sich im Kapitel **Systemcalls im NanoJ-Programm**.

Tipp

Nanotec empfiehlt, entweder per Mapping oder Systemcall mit `od_write()` auf ein und denselben OD-Wert zuzugreifen. Wird beides gleichzeitig verwendet, so hat der Systemcall keine Auswirkung.

10.1.5 NanoJ-Programm OD-Einträge

Das *NanoJ-Programm* wird durch OD-Einträge im Objekt-Bereich 2300_h bis 2330_h gesteuert und konfiguriert (siehe **2300h NanoJ Control**).

OD-Index	Name und Beschreibung
2300_h	**2300h NanoJ Control**
2301_h	**2301h NanoJ Status**
2302_h	**2302h NanoJ Error Code**
2310_h	**2310h NanoJ Input Data Selection**
2320_h	**2320h NanoJ Output Data Selection**
2330_h	**2330h NanoJ In/output Data Selection**

Beispiel:

Um das Benutzerprogramm *TEST1.USR* zu starten, kann z. B. folgende Sequenz benutzt werden:

- Überprüfen des Eintrags **2302_h** auf Fehlercode.
- Wenn kein Fehler:
  NanoJ-Programm starten durch Beschreiben von Objekt **2300_h**, Bit 0 = "1".

Hinweis

Das Starten des NanoJ Programms kann bis zu 200 ms dauern.

- Überprüfen des Eintrags **2302_h** auf Fehlercode und des Objekts **2301_h**, Bit 0 = "1".

Um ein laufendes Programm anzuhalten: Beschreiben des Eintrags **2300_h** mit dem Bit 0 Wert = "0".

10.1.6 Aufbau NanoJ-Programm

Ein Benutzerprogramm besteht aus mindestens zwei Anweisungen:

- der Präprozessoranweisung `#include "wrapper.h"`
- der Funktion `void user(){}`

In der Funktion `void user()` lässt sich der auszuführende Code hinterlegen.

Hinweis

Die Dateinamen der Benutzerprogramme dürfen nicht länger als acht Zeichen sein und drei Zeichen im Suffix enthalten; Dateiname `main.cpp` ist zulässig, Dateiname `einLangerDateiname.cpp` ist nicht zulässig.

Hinweis

Im *NanoJ-Programm*dürfen nur globale Variablen und ausschließlich innerhalb von Code initialisieren. Daraus folgt:

- kein `new` Operator
- keine Konstruktoren
- keine Initialisierung von globalen Variablen außerhalb von Code

Beispiele:

Die globale Variable soll erst innerhalb der Funktion `void user()` initialisiert werden:

```
unsigned int i;
void user(){
 i = 1;
 i += 1;
}
```

Folgende Zuweisung ist nicht korrekt :

```
unsigned int i = 1;
 void user() {
 i += 1;
}
```

10.1.7 NanoJ-Programmbeispiel

Das Beispiel zeigt das Programmieren eines Rechtecksignals in das Objekt 2500_h:01_h.

```
// file main.cpp
map S32 outputReg1 as inout 0x2500:1
#include "wrapper.h"

// user program
void user()
{
  U16 counter = 0;
  while( 1 )
  {
    ++counter;

    if( counter < 100 )
     InOut.outputReg1 = 0;
    else if( counter < 200 )
      InOut.outputReg1 = 1;
    else
      counter = 0;

    // yield() 5 times (delay 5ms)
    for(U08 i = 0; i < 5; ++i )
      yield();
  }
}// eof
```

Weitere Beispiele finden Sie auf **www.nanotec.de**

10.2 Mapping im NanoJ-Programm

Mit dieser Methode wird eine Variable im *NanoJ-Programm* direkt mit einem Eintrag im Objektverzeichnis verknüpft. Das Anlegen des Mappings muss dabei am Anfang der Datei stehen - noch vor der `#include "wrapper.h"`-Anweisung. Ein Kommentar oberhalb des Mappings ist erlaubt.

Tipp

Nanotec empfiehlt:

- Benutzen Sie das Mapping, falls Sie den Zugriff auf ein Objekt im Objektverzeichnis häufiger benötigen, z. B. das *Controlword* 6040_h oder das *Statusword* 6041_h.
- Für den einzelnen Zugriff auf Objekte bieten sich eher die Funktionen `od_write()` und `od_read()` an, siehe **Zugriff auf das Objektverzeichnis**.

10.2.1 Deklaration des Mappings

Die Deklaration des Mappings gliedert sich dabei folgendermaßen:

```
map <TYPE> <NAME> as <input|output|inout> <INDEX>:<SUBINDEX>
```

Dabei gilt:

- `<TYPE>`

  Der Datentyp der Variable; U32, U16, U08, S32, S16 oder S08.
- `<NAME>`
  Der Name der Variable; wie sie im Benutzerprogramm verwendet wird.
- `<input|output|inout>`

  Die Schreib- und Leseberechtigung einer Variable: Eine Variable kann entweder als `input`, `output` oder `inout` deklariert werden. Damit wird festgelegt, ob eine Variable lesbar (`input`), schreibbar (`output`) oder beides ist (`inout`) und über welche Struktur sie im Programm angesprochen werden muss.
- `<INDEX>:<SUBINDEX>`

  Index und Subindex des zu mappenden Objekts im Objektverzeichnis.

Jede deklarierte Variable wird im Benutzerprogramm über eine der drei Strukturen *In*, *Out* oder *InOut* angesprochen, je nach definierter Schreib- und Leserichtung.

10.2.2 Beispiel eines Mappings

Beispiel eines Mappings und der zugehörigen Variablenzugriffe:

```
map U16 controlWord as output 0x6040:00
map U08 statusWord as input 0x6041:00
map U08 modeOfOperation as inout 0x6060:00

#include "wrapper.h"

void user()
{
  [...]
  Out.controlWord = 1;
```

```
  U08 tmpVar = In.statusword;
  InOut.modeOfOperation = tmpVar;
  [...]
}
```

10.2.3 Möglicher Fehler bei `od_write()`

Eine mögliche Fehlerquelle ist ein schreibender Zugriff mittels der Funktion `od_write()` (siehe **Systemcalls im NanoJ-Programm**) auf ein Objekt im Objektverzeichnis, welches gleichzeitig als Mapping angelegt wurde. Nachfolgend aufgelisteter Code ist fehlerhaft:

```
map U16 controlWord as output 0x6040:00
#include " wrapper.h"
void user()
{
 [...]
  Out.controlWord = 1;
  [...]
  od_write(0x6040, 0x00, 5 ); // der Wert wird durch das Mapping
 überschrieben
  [...]
}
```

Die Zeile mit dem Befehl `od_write(0x6040, 0x00, 5 );` ist wirkungslos. Wie in der Einleitung beschrieben, werden alle Mappings am Ende jeder Millisekunde in das Objektverzeichnis kopiert.

Damit ergibt sich folgender Ablauf:

1. Die Funktion `od_write` schreibt den Wert `5` in das Objekt 6040_h:00_h.
2. Am Ende des 1 ms-Zyklus wird das Mapping geschrieben, welches ebenfalls das Objekt 6040_h:00_h beschreibt, allerdings mit dem Wert `1`.
3. Somit wird - aus Sicht des Benutzers - der `od_write`-Befehl wirkungslos.

10.3 Systemcalls im NanoJ-Programm

Mit Systemcalls ist es möglich, in der Firmware eingebaute Funktionen direkt aus einem Benutzerprogramm aufzurufen. Eine direkte Code-Ausführung ist nur in dem geschützten Bereich der Sandbox möglich und wird über sogenannte *Cortex-Supervisor-Calls* (Svc Calls) realisiert. Dabei wird mit dem Aufruf der Funktion ein Interrupt ausgelöst und die Firmware hat so die Möglichkeit, temporär eine Code-Ausführung außerhalb der Sandbox zuzulassen. Der Entwickler des Benutzerprogramms muss sich jedoch um diesen Mechanismus nicht kümmern - für ihn sind die Systemcalls wie ganz normale C-Funktionen aufrufbar. Lediglich die Datei *wrapper.h* muss - wie üblich - eingebunden werden.

10.3.1 Zugriff auf das Objektverzeichnis

void **od_write** (U32 index, U32 subindex, U32 value)

Diese Funktion schreibt den übergebenen Wert an die angegebene Stelle in das Objektverzeichnis.

index	Index des zu schreibenden Objekts im Objektverzeichnis
subindex	Subindex des zu schreibenden Objekts im Objektverzeichnis
value	zu schreibender Wert

Hinweis

Es wird dringend empfohlen, nach dem Aufruf eines `od_write()` die Prozessorzeit mit `yield()` abzugeben. Der Wert wird zwar sofort ins OD geschrieben. Damit die Firmware jedoch davon abhängige Aktionen auslösen kann, muss diese Rechenzeit erhalten und somit das Benutzerprogramm beendet oder mit `yield()` unterbrochen worden sein.

U32 **od_read** (U32 index, U32 subindex)

Diese Funktion liest den Wert an der angegebenen Stelle aus dem Objektverzeichnis und gibt ihn zurück.

index	Index des zu lesenden Objekts im Objektverzeichnis
subindex	Subindex des zu lesenden Objekts im Objektverzeichnis
Rückgabewert	Inhalt des OD-Eintrags

Hinweis

Aktives Warten auf einen Wert im Objektverzeichnis sollte immer mit einem `yield()` verbunden werden.

Beispiel

```
while (od_read(2400,2) != 0) // wait until 2400:2 is set
{ yield(); }
```

10.3.2 Prozesssteuerung

```
void yield()
```

Diese Funktion gibt die Prozessorzeit wieder an das Betriebssystem ab. Das Programm wird in der nächsten Zeitscheibe wieder an der Stelle nach dem Aufruf fortgesetzt.

```
void sleep (U32 ms)
```

Diese Funktion gibt die Prozessorzeit für die angegebene Zahl an Millisekunden an das Betriebssystem ab. Das Benutzerprogramm wird anschließend an der Stelle nach dem Aufruf fortgesetzt.

ms	Zu wartende Zeit in Millisekunden

11 Objektverzeichnis Beschreibung

11.1 Übersicht

In diesem Kapitel finden Sie eine Beschreibung aller Objekte.

Sie finden hier Angaben zu:

- Funktionen
- Objektbeschreibungen ("Index")
- Wertebeschreibungen ("Subindices")
- Beschreibungen von Bits
- Beschreibung des Objekts

11.2 Aufbau der Objektbeschreibung

Die Beschreibung der Objekteinträge ist immer gleich aufgebaut und besteht im Normalfall aus folgenden Abschnitten:

Funktion

In diesem Abschnitt wird kurz die Funktion des Objektverzeichnisses beschrieben.

Objektbeschreibung

Diese Tabelle gibt detailliert Auskunft über den Datentyp, Vorgabewerte und dergleichen. Eine genaue Beschreibung findet sich im Abschnitt "**Objektbeschreibung**"

Wertebeschreibung

Diese Tabelle ist nur bei dem Datentyp "Array" oder "Record" verfügbar und gibt genaue Auskunft über die Untereinträge. Eine genauere Beschreibung der Einträge findet sich im Abschnitt "**Wertebeschreibung**"

Beschreibung

Hier werden genauere Angaben zu den einzelnen Bits eines Eintrags gemacht oder eventuelle Zusammensetzungen erläutert. Eine genauere Beschreibung findet sich im Abschnitt "**Beschreibung**"

11.3 Objektbeschreibung

Die Objektbeschreibung besteht aus einer Tabelle, welche folgende Einträge enthält:

Index

Benennt den Index des Objekts in Hexadezimalschreibweise.

Objektname

Der Name des Objekts.

Object Code

Der Typ des Objekts. Das kann einer der folgenden Einträge sein:

- VARIABLE: In dem Fall besteht das Objekt nur aus einer Variable, die mit dem Subindex 0 indiziert wird.
- ARRAY: Diese Objekte bestehen immer aus einem Subindex 0 - welcher die Menge der Untereinträge angibt - und den Untereinträgen selber ab dem Index 1. Der Datentyp innerhalb eines Arrays ändert sich nie, das heißt, Untereintrag 1 und alle folgenden Einträge haben immer den gleichen Datentyp.
- RECORD: Diese Objekte bestehen immer aus einem Untereintrag mit dem Subindex 0 - welcher die Menge der Untereinträge angibt - und den Untereinträgen selber ab dem Index 1. Im Gegensatz zu einem ARRAY kann der Datentyp der Subeinträge variieren, das

bedeutet, dass beispielsweise Untereintrag 1 einen anderen Datentyp als Untereintrag 2 haben kann.

- VISIBLE_STRING: Das Objekt beschreibt eine in ASCII codierte Zeichenkette. Die Länge des Strings wird in Subindex 0 angegeben, die einzelnen Zeichen sind ab Subindex 1 gespeichert. Diese Zeichenketten sind **nicht** durch ein Null-Zeichen terminiert.

Datentyp

Hier wird die Größe und die Interpretation des Objekts angegeben. Für den Object Code "VARIABLE" gilt folgende Schreibweise:

- Es wird unterschieden zwischen Einträgen die vorzeichenbehaftet sind, das wird mit dem Präfix "SIGNED" bezeichnet. Für die vorzeichenunbehafteten Einträge wird das Präfix "UNSIGNED" benutzt.
- Die Größe der Variable in Bit wird an das Präfix angestellt und kann entweder 8, 16 oder 32 sein.

Speicherbar

Hier wird beschreiben ob dieses Objekt speicherbar ist und wenn ja, unter welcher Kategorie.

Firmware Version

Hier ist die Firmwareversion eingetragen, ab der das Objekt verfügbar ist.

Änderungshistorie (ChangeLog)

Hier werden eventuelle Änderungen an dem Objekt notiert.

Zudem gibt es noch die Einträge für den Datentyp "VARIABLE" folgende Tabelleneinträge:

Zugriff

Hier wird die Zugriffsbeschränkung eingetragen. Folgende Beschränkungen gibt es:

- "lesen/schreiben": Das Objekt kann sowohl gelesen, als auch geschrieben werden
- "nur lesen": Das Objekt kann nur aus dem Objektverzeichnis gelesen werden. Setzen eines Werte ist nicht möglich.

PDO-Mapping

Einige Bussysteme, wie CANopen oder EtherCAT unterstützen ein PDO-Mapping. In diesem Tabelleneintrag wird beschrieben, ob das Objekt in ein Mapping eingefügt werden darf und in welches. Dabei gibt es folgende Bezeichnungen:

- "no": Das Objekt darf in kein Mapping eingetragen werden.
- "TX-PDO": Das Objekt darf in ein RX Mapping eingetragen werden.
- "RX-PDO": Das Objekt dar in ein TX Mapping eingetragen werden.

Zulässige Werte

In einigen Fällen ist es nur erlaubt, bestimmte Werte in das Objekt zu schreiben. Sollte das der Fall sein, werden diese Werte hier aufgelistet. Besteht keine Beschränkung bleibt das Feld leer.

Vorgabewert

Um die Steuerung beim Einschalten in einen gesicherten Zustand zu bringen ist es nötig, einige Objekte mit Werten vorzubelegen. Der Wert, der beim Start der Steuerung in das Objekt geschrieben wird, wird in diesem Tabelleneintrag notiert.

11.4 Wertebeschreibung

Hinweis

Der Übersichtlichkeit halber werden einige Subindizes zusammengefasst, wenn die Einträge alle den gleichen Namen haben.

In der Tabelle mit der Überschrift "Wertebeschreibung" werden alle Daten für Untereinträge mit Subindex 1 oder höher aufgelistet. Die Tabelle beinhaltet folgende Einträge:

Subindex

Nummer des aktuell beschriebenen Untereintrages.

Name

Der Name des Untereintrages.

Datentyp

Hier wird die Größe und die Interpretation des Untereintrages angegeben. Hier gilt immer folgende Schreibweise:

- Es wird unterschieden zwischen Einträgen die vorzeichenbehaftet sind, das wird mit dem Präfix "SIGNED" bezeichnet. Für die vorzeichenunbehafteten Einträge wird das Präfix "UNSIGNED" benutzt.
- Die Größe der Variable in Bit wird an das Präfix angestellt und kann entweder 8, 16 oder 32 sein.

Zugriff

Hier wird die Zugriffsbeschränkung für den Untereintrag eingetragen. Folgende Beschränkungen gibt es:

- "lesen/schreiben": Das Objekt kann sowohl gelesen, als auch geschrieben werden
- "nur lesen": Das Objekt kann nur aus dem Objektverzeichnis gelesen werden. Setzen eines Wertes ist nicht möglich.

PDO-Mapping

Einige Bussysteme, wie CANopen oder EtherCAT unterstützen ein PDO-Mapping. In diesem Tabelleneintrag wird beschrieben, ob der Untereintrag in ein Mapping eingefügt werden darf und in welches. Dabei gibt es folgende Bezeichnungen:

- "no": Das Objekt darf in kein Mapping eingetragen werden.
- "TX-PDO": Das Objekt darf in ein RX Mapping eingetragen werden.
- "RX-PDO": Das Objekt darf in ein TX Mapping eingetragen werden.

Zulässige Werte

In einigen Fällen ist es nur erlaubt, bestimmte Werte in den Untereintrag zu schreiben. Sollte das der Fall sein, werden diese Werte hier aufgelistet. Besteht keine Beschränkung, bleibt das Feld leer.

Vorgabewert

Um die Steuerung beim Einschalten in einen gesicherten Zustand zu bringen ist es nötig, einige Untereinträge mit Werten vor zu belegen. Der Wert, welcher beim Start der Steuerung in den Untereintrag geschrieben wird, wird in diesem Tabelleneintrag notiert.

11.5 Beschreibung

Dieser Abschnitt kann vorhanden sein, wenn die Benutzung zusätzliche Information verlangt. Sollten einzelne Bits eines Objekts oder Untereintrags unterschiedliche Bedeutung haben, so werden Diagramme wie im nachfolgenden Beispiel verwendet.

Beispiel: Das Objekt ist 8 Bit groß, Bit 0 und 1 haben separat eine Funktion. Bit 2 und 3 sind zu einer Funktion zusammengefasst, für Bit 4 bis 7 gilt das gleiche.

7	6	5	4	3	2	1	0
Beispiel [4]				Beispiel [2]		B	A

Beispiel [4]

Beschreibung der Bits 4 bis einschließlich 7, diese Bits gehören logisch zusammen. Die 4 in den eckigen Klammern gibt die Anzahl der zusammengehörigen Bits an. Oftmals wird an der Stelle noch eine Liste mit möglichen Werten und deren Beschreibung angehängt.

Beispiel [2]

Beschreibung der Bits 3 und 2, diese Bits gehören logisch zusammen. Die 2 in den eckigen Klammern gibt die Anzahl der zusammengehörigen Bits an.

- Wert 00_b: Die Beschreibung an dieser Stelle gilt, wenn Bit 2 und Bit 3 auf "0" sind.
- Wert 01_b: Die Beschreibung an dieser Stelle gilt, wenn Bit 2 auf "0" und Bit 3 auf "1" ist.
- Wert 10_b: Die Beschreibung an dieser Stelle gilt, wenn Bit 2 auf "1" und Bit 3 auf "0" ist.
- Wert 11_b: Die Beschreibung an dieser Stelle gilt, wenn Bit 2 und Bit 3 auf "1" sind.

B

Beschreibung des Bits B, auf die Längenangabe wird bei einem einzelnen Bit verzichtet.

A

Beschreibung des Bits A, Bits mit grauen Hintergrund bleiben ungenutzt.

1000h Device Type

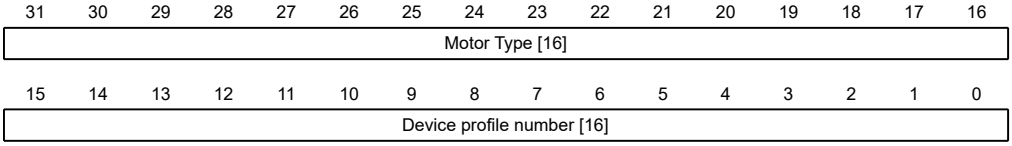
Funktion

Beschreibt den Steuerungstyp.

Objektbeschreibung

Index	1000_h
Objektname	Device Type
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00060192_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung



Motor Type[16]

Beschreibt den unterstützten Motor-Typ. Die folgenden Werte sind möglich:

- Bit 23 bis Bit 16: Wert "1": Servoantrieb
- Bit 23 bis Bit 16: Wert "2": Schrittmotor

Device profile number[16]

Beschreibt den unterstützten CANopen-Standard.

Werte:

0192_h bzw. 0402_d (Vorgabewert): Der CiA 402-Standard wird unterstützt.

1001h Error Register

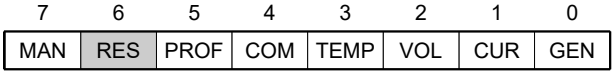
Funktion

Fehlerregister: Im Fehlerfall wird das entsprechende Fehlerbit gesetzt. Sollte der Fehler nicht mehr bestehen, wird es automatisch wieder gelöscht.

Objektbeschreibung

Index	1001_h
Objektname	Error Register
Object Code	VARIABLE
Datentyp	UNSIGNED8
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung



GEN

Genereller Fehler

CUR

Strom

VOL
Spannung

TEMP
Temperatur

COM
Kommunikation

PROF
Betrifft das Geräteprofil

RES
Reserviert, immer "0"

MAN
Hersteller spezifisch: Der Motor drehte sich in die falsche Richtung.

1003h Pre-defined Error Field

Funktion

Dieses Objekt beinhaltet einen Fehlerstapel mit bis zu acht Einträgen.

Objektbeschreibung

Index	1003_h
Objektname	Pre-defined Error Field
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	nein
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Number Of Errors
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	01_h
Name	Standard Error Field
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein

Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Standard Error Field
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	Standard Error Field
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	04_h
Name	Standard Error Field
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	05_h
Name	Standard Error Field
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	06_h
Name	Standard Error Field
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	07_h
Name	Standard Error Field
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	08_h
Name	Standard Error Field
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

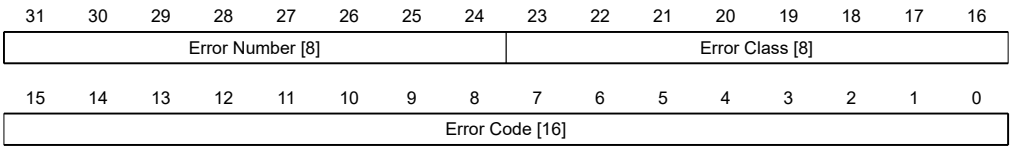
Beschreibung

Allgemeine Funktionsweise

Tritt ein neuer Fehler auf, wird dieser in Subindex 1 eingetragen. Die bereits vorhandenen Einträge in den Subindizes 1 bis 7 werden um eine Stelle nach hinten verschoben. Der Fehler auf Subindex 7 wird dabei entfernt.

Die Anzahl der bereits aufgetreten Fehler lässt sich aus dem Objekt mit dem Subindex 0 ablesen. Ist im Fehlerstapel zur Zeit kein Fehler eingetragen, dann ist das Auslesen eines der acht Subindizes 1-8 nicht möglich und wird mit einem Fehler (Abort-Code=08000024_h) beantwortet. Wird in den Subindex 0 eine "0" geschrieben, beginnt die Zählung von neuem.

Bitbeschreibung



Error Number [8]

Damit lässt sich der Grund des Fehlers genau eingrenzen. Die Bedeutung der Zahl lässt sich aus nachfolgender Tabelle entnehmen.

Fehlernummer	Beschreibung
0	Watchdog-Reset
1	Eingangsspannung zu hoch
2	Ausgangsstrom zu hoch
3	Eingangsspannung zu niedrig
4	Fehler am Feldbus
5	Motor dreht - trotz aktivierter Sperre - in die falsche Richtung
6	Nur CANopen: NMT-Master braucht zu lange, um Nodeguarding-Anforderung zu schicken
7	Encoderfehler durch elektrische Störung oder defekte Hardware

Fehlernummer	Beschreibung
8	Encoderfehler; Index während des Auto-Setups nicht gefunden
9	Fehler in der AB-Spur
10	Positiver Endschalter und Toleranzzone überschritten
11	Negativer Endschalter und Toleranzzone überschritten
12	Temperatur des Gerätes oberhalb 80°C
13	Die Werte des Objekts **6065**$_h$ (Following Error Window) und des Objekts **6066**$_h$ (Following Error Time Out) wurden überschritten, es wurde ein Fault ausgelöst.
14	Nichtflüchtiger Speicher voll, Neustart der Steuerung erforderlich für Aufräumarbeiten.
15	Motor blockiert
16	Nichtflüchtiger Speicher beschädigt, Neustart der Steuerung erforderlich für Aufräumarbeiten.
17	Nur CANopen: Slave brauchte zu lange um PDO Nachrichten zu Senden.
18	Hallsensor fehlerhaft
19	Nur CANopen: PDO aufgrund eines Längenfehlers nicht verarbeitet
20	Nur CANopen: PDO Länge überschritten
21	Nichtflüchtiger Speicher voll, Neustart der Steuerung erforderlich für Aufräumarbeiten.
22	Nennstrom muss gesetzt werden (203B$_h$:01$_h$)
23	Encoderauflösung, Polpaarzahl und einige andere Werte sind falsch.
24	Motorstrom ist zu hoch, passen Sie die PI-Parameter an.
25	Interner Softwarefehler, generisch
26	Zu hoher Strom am digitalen Ausgang
27	Nur CANopen: Unerwartete Sync-Länge
28	Nur EtherCAT: Der Motor wurde gestoppt, da von EtherCAT Zustand OP nach SafeOP, oder PreOP geschalten wurde ohne vorher den Motor zu stoppen.

Error Class[8]

Dieses Byte ist identisch mit dem Objekt **1001**$_h$

Error Code[16]

Die Bedeutung der beiden Bytes lässt sich aus der nachfolgenden Tabelle entnehmen.

Error Code	Beschreibung
1000$_h$	Allgemeiner Fehler
2300$_h$	Strom am Ausgang der Steuerung zu groß
3100$_h$	Über-/ Unterspannung am Eingang der Steuerung
4200$_h$	Temperaturfehler innerhalb der Steuerung
6010$_h$	Software reset (watchdog)
6100$_h$	Interner Softwarefehler, generisch
6320$_h$	Nennstrom muss gesetzt werden (203B$_h$:01$_h$)
7121$_h$	Motor blockiert
7305$_h$	Inkrementaler oder Hallsensor fehlerhaft

Error Code	Beschreibung
7600_h	Nichtflüchtiger Speicher voll oder korrupt, Neustart der Steuerung für Aufräumarbeiten
8000_h	Fehler bei der Feldbusüberwachung
8130_h	Nur CANopen: "Life Guard" - Fehler oder "Heartbeat" - Fehler
8200_h	Nur CANopen: Slave brauchte zu lange um PDO Nachrichten zu Senden.
8210_h	Nur CANopen: PDO wurde nicht verarbeitet aufgrund eines Längen-Fehlers
8220_h	Nur CANopen: PDO Länge überschritten
8611_h	Fehler in der Positionsüberwachung: Schleppfehler zu groß
8612_h	Fehler in der Positionsüberwachung: Endschalter und Toleranzzone überschritten
9000_h	EtherCAT: Motor fährt während Ethercat wechselt von OP -> SafeOp, PreOP usw.

1008h Manufacturer Device Name

Funktion

Enthält den Gerätenamen als Zeichenkette.

Objektbeschreibung

Index	1008_h
Objektname	Manufacturer Device Name
Object Code	VARIABLE
Datentyp	VISIBLE_STRING
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	NP5-40
Firmware Version	FIR-v1426
Änderungshistorie	

1009h Manufacturer Hardware Version

Funktion

Dieses Objekt enthält die Hardware-Version als Zeichenkette.

Objektbeschreibung

Index	1009_h
Objektname	Manufacturer Hardware Version
Object Code	VARIABLE
Datentyp	VISIBLE_STRING
Speicherbar	nein

Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0
Firmware Version	FIR-v1426
Änderungshistorie	

100Ah Manufacturer Software Version

Funktion

Dieses Objekt enthält die Software-Version als Zeichenkette.

Objektbeschreibung

Index	$100A_h$
Objektname	Manufacturer Software Version
Object Code	VARIABLE
Datentyp	VISIBLE_STRING
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	FIR-v1650-B527540
Firmware Version	FIR-v1426
Änderungshistorie	

1010h Store Parameters

Funktion

Mit diesem Objekt lässt sich das Speichern von Objekten starten.

Objektbeschreibung

Index	1010_h
Objektname	Store Parameters
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	nein
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1436: Eintrag "Objektname" geändert von "Store Parameter" auf "Store Parameters". Firmware Version FIR-v1436: Die Anzahl der Einträge haben sich geändert von 3 auf 4. Firmware Version FIR-v1512: Die Anzahl der Einträge haben sich geändert von 4 auf 5.

	Firmware Version FIR-v1540: Die Anzahl der Einträge haben sich geändert von 5 auf 7.

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	06_h

Subindex	01_h
Name	Save All Parameters To Non-volatile Memory
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	02_h
Name	Save Communication Parameters To Non-volatile Memory
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	03_h
Name	Save Application Parameters To Non-volatile Memory
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	04_h
Name	Save Customer Parameters To Non-volatile Memory
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	00000001_h

Subindex	05_h
Name	Save Drive Parameters To Non-volatile Memory
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	06_h
Name	Save Tuning Parameters To Non-volatile Memory
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Beschreibung

Jeder Subindex des Objekts steht für eine bestimmte Speicherklasse. Durch Auslesen eines Eintrages kann festgestellt werden, ob diese Speicherkategorie abgespeichert (Wert "1") werden kann oder nicht (Wert="0").

Um den Speichervorgang einer Speicherkategorie zu starten, muss der Wert "65766173_h" in den jeweiligen Subindex geschrieben werden. Das entspricht dezimal der 1702257011_d bzw. dem ASCII String " `save`. Sobald der Speichervorgang abgeschlossen wurde, wird der Speicherbefehl wieder durch den Wert "1" überschrieben, da ein Speichern wieder möglich ist.

Für eine detaillierte Beschreibung siehe Kapitel **Objekte speichern**.

1011h Restore Default Parameters

Funktion

Mit diesem Objekt kann das gesamte oder Teile des Objektverzeichnis auf die Defaultwerte zurückgesetzt werden.

Objektbeschreibung

Index	1011_h
Objektname	Restore Default Parameters
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	nein
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1436: Eintrag "Object Name" geändert von "Restore Default Parameter" auf "Restore Default Parameters".

	Firmware Version FIR-v1436: Die Anzahl der Einträge haben sich geändert von 2 auf 4. Firmware Version FIR-v1512: Die Anzahl der Einträge haben sich geändert von 4 auf 5. Firmware Version FIR-v1512: Eintrag "Name" geändert von "Restore The Comm Default Parameters" auf "Restore Communication Default Parameters". Firmware Version FIR-v1512: Eintrag "Name" geändert von "Restore The Application Default Parameters" auf "Restore Application Default Parameters". Firmware Version FIR-v1540: Die Anzahl der Einträge haben sich geändert von 5 auf 7.

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	06_h

Subindex	01_h
Name	Restore All Default Parameters
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	02_h
Name	Restore Communication Default Parameters
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	03_h
Name	Restore Application Default Parameters
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	00000001_h

Subindex	04_h
Name	Restore Customer Default Parameters
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	05_h
Name	Restore Drive Default Parameters
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	06_h
Name	Restore Tuning Default Parameters
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Wird der Wert $64616F6C_h$ (bzw. 1684107116_d oder ASCII `load`) in dieses Objekt geschrieben, werden Teile oder das gesamte Objektverzeichnis auf die Defaultwerte zurückgesetzt. Der verwendete Subindex entscheidet darüber, welcher Bereich zurück gesetzt wird.

Für eine detaillierte Beschreibung siehe Kapitel **Speicherung verwerfen**.

1018h Identity Object

Funktion

Dieses Objekt liefert generelle Informationen zu dem Gerät wie Hersteller, Produktcode, Revision und Seriennummer.

Tipp

Halten Sie diese Werte bei Serviceanfragen bereit.

Objektbeschreibung

Index	1018_h
Objektname	Identity Object
Object Code	RECORD
Datentyp	IDENTITY
Speicherbar	nein
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	04_h

Subindex	01_h
Name	Vendor-ID
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$0000026C_h$

Subindex	02_h
Name	Product Code
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$0000000C_h$

Subindex	03_h
Name	Revision Number
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	06720000_h

Subindex	04_h
Name	Serial Number
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

1020h Verify Configuration

Funktion

Dieses Objekt zeigt den Tag und die Zeit der abgespeicherten Konfiguration an.

Ein Konfigurationstool oder ein Master kann dieses Objekt nutzen, um die Konfiguration nach einem Reset zu verifizieren und gegebenenfalls eine Neukonfiguration durchzuführen.

Das Tool muss das Datum und die Uhrzeit setzen, bevor der Speichermechanismus gestartet wird (siehe Kapitel **Objekte speichern**).

Objektbeschreibung

Index	1020_h
Objektname	Verify Configuration
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Prüfung
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1540
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Configuration Date
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben

PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Configuration Time
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Subindex 01_h (Konfigurationsdatum) soll die Anzahl der Tage seit dem 1. Januar 1984 enthalten.

Subindex 02_h (Konfigurationszeit) soll die Nummer der Millisekunden seit Mitternacht enthalten.

1600h Receive PDO 1 Mapping Parameter

Funktion

Dieses Objekt enthält die Mapping-Parameter für PDOs, welche die Steuerung empfangen kann (RX-PDO 1).

Objektbeschreibung

Index	1600_h
Objektname	Receive PDO 1 Mapping Parameter
Object Code	RECORD
Datentyp	PDO_MAPPING
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1426: Eintrag "Überschrift" geändert von "1600h Drive Control" auf "1600h Receive PDO 1 Mapping Parameter". Firmware Version FIR-v1426: Eintrag "Object Name" geändert von "Drive Control" auf "Receive PDO 1 Mapping Parameter".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	02_h

Subindex	01_h
Name	1st Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60600008_h

Subindex	02_h
Name	2nd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60400010_h

Subindex	03_h
Name	3rd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	04_h
Name	4th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	05_h
Name	5th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	06_h

Name	6th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

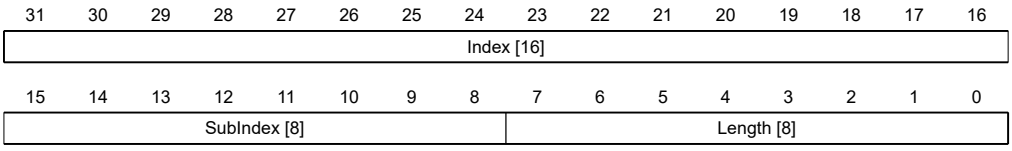
Subindex	07_h
Name	7th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	08_h
Name	8th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Jeder Subindex (1-8) beschreibt jeweils ein gemapptes Objekt.

Ein Mapping-Eintrag besteht aus vier Bytes, die sich nach folgender Grafik zusammensetzen.



Index [16]

Darin ist der Index des zu mappenden Objektes enthalten.

Subindex [8]

Darin ist der Subindex des zu mappenden Objektes enthalten.

Length [8]

Darin ist die Länge des zu mappenden Objektes in der Einheit Bit enthalten.

1601h Receive PDO 2 Mapping Parameter

Funktion

Dieses Objekt enthält die Mapping-Parameter für PDOs, welche die Steuerung empfangen kann (RX-PDO 2).

Objektbeschreibung

Index	1601_h
Objektname	Receive PDO 2 Mapping Parameter
Object Code	RECORD
Datentyp	PDO_MAPPING
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1426: Eintrag "Überschrift" geändert von "1601h Positioning Control" auf "1601h Receive PDO 2 Mapping Parameter". Firmware Version FIR-v1426: Eintrag "Object Name" geändert von "Positioning Control" auf "Receive PDO 2 Mapping Parameter".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	05_h

Subindex	01_h
Name	1st Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$607A0020_h$

Subindex	02_h
Name	2nd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60420010_h

Subindex	03_h
Name	3rd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben

PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$60FF0020_h$

Subindex	04_h
Name	4th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60710010_h

Subindex	05_h
Name	5th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60980008_h

Subindex	06_h
Name	6th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

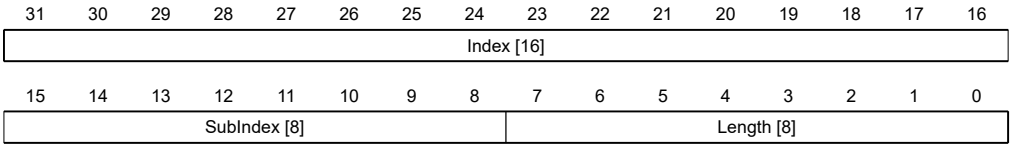
Subindex	07_h
Name	7th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	08_h
Name	8th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Jeder Subindex (1-8) beschreibt jeweils ein gemapptes Objekt.

Ein Mapping-Eintrag besteht aus vier Bytes, die sich nach folgender Grafik zusammensetzen.



Index [16]

Darin ist der Index des zu mappenden Objektes enthalten.

Subindex [8]

Darin ist der Subindex des zu mappenden Objektes enthalten.

Length [8]

Darin ist die Länge des zu mappenden Objektes in der Einheit Bit enthalten.

1602h Receive PDO 3 Mapping Parameter

Funktion

Dieses Objekt enthält die Mapping-Parameter für PDOs, welche die Steuerung empfangen kann (RX-PDO 3).

Objektbeschreibung

Index	1602_h
Objektname	Receive PDO 3 Mapping Parameter
Object Code	RECORD
Datentyp	PDO_MAPPING
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1426: Eintrag "Überschrift" geändert von "1602h Velocity Control" auf "1602h Receive PDO 3 Mapping Parameter". Firmware Version FIR-v1426: Eintrag "Object Name" geändert von "Velocity Control" auf "Receive PDO 3 Mapping Parameter".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	01_h
Name	1st Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	2nd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	3rd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	04_h
Name	4th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	05_h
Name	5th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	06_h
Name	6th Object To Be Mapped
Datentyp	UNSIGNED32

Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	07_h
Name	7th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	08_h
Name	8th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

1603h Receive PDO 4 Mapping Parameter

Funktion

Dieses Objekt enthält die Mapping-Parameter für PDOs, welche die Steuerung empfangen kann (RX-PDO 4).

Objektbeschreibung

Index	1603_h
Objektname	Receive PDO 4 Mapping Parameter
Object Code	RECORD
Datentyp	PDO_MAPPING
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1426: Eintrag "Überschrift" geändert von "1603h Output Control" auf "1603h Receive PDO 4 Mapping Parameter". Firmware Version FIR-v1426: Eintrag "Object Name" geändert von "Output Control" auf "Receive PDO 4 Mapping Parameter".

Wertebeschreibung

Subindex	00_h

Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	01_h
Name	1st Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	2nd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	3rd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	04_h
Name	4th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	05_h
Name	5th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben

PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	06_h
Name	6th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	07_h
Name	7th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	08_h
Name	8th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

1A00h Transmit PDO 1 Mapping Parameter

Funktion

Dieses Objekt enthält die Mapping-Parameter für PDOs, welche die Steuerung senden kann (TX-PDO 1).

Objektbeschreibung

Index	$1A00_h$
Objektname	Transmit PDO 1 Mapping Parameter
Object Code	RECORD
Datentyp	PDO_MAPPING
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1426: Eintrag "Überschrift" geändert von "1A00h Drive Status" auf "1A00h Transmit PDO 1 Mapping Parameter".

	Firmware Version FIR-v1426: Eintrag "Object Name" geändert von "Drive Status" auf "Transmit PDO 1 Mapping Parameter".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	03_h

Subindex	01_h
Name	1st Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60610008_h

Subindex	02_h
Name	2nd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60410010_h

Subindex	03_h
Name	3rd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	10010008_h

Subindex	04_h
Name	4th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	00000000_h

Subindex	05_h
Name	5th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	06_h
Name	6th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

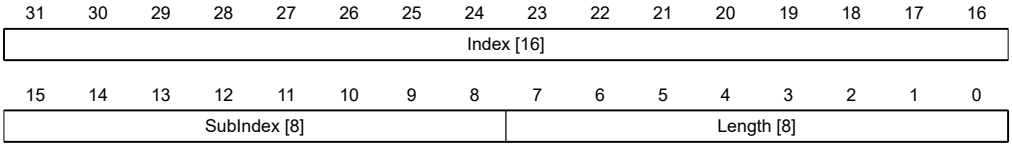
Subindex	07_h
Name	7th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	08_h
Name	8th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Jeder Subindex (1-8) beschreibt jeweils ein gemapptes Objekt.

Ein Mapping-Eintrag besteht aus vier Bytes, die sich nach folgender Grafik zusammensetzen.



Index [16]

Darin ist der Index des zu mappenden Objektes enthalten.

Subindex [8]

Darin ist der Subindex des zu mappenden Objektes enthalten.

Length [8]

Darin ist die Länge des zu mappenden Objektes in der Einheit Bit enthalten.

1A01h Transmit PDO 2 Mapping Parameter

Funktion

Dieses Objekt enthält die Mapping-Parameter für PDOs, welche die Steuerung senden kann (TX-PDO 2).

Objektbeschreibung

Index	$1A01_h$
Objektname	Transmit PDO 2 Mapping Parameter
Object Code	RECORD
Datentyp	PDO_MAPPING
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1426: Eintrag "Überschrift" geändert von "1A01h Positioning Status" auf "1A01h Transmit PDO 2 Mapping Parameter". Firmware Version FIR-v1426: Eintrag "Object Name" geändert von "Positioning Status" auf "Transmit PDO 2 Mapping Parameter".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	08_h

Subindex	01_h
Name	1st Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60620020_h

Subindex	02_h
Name	2nd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60640020_h

Subindex	03_h
Name	3rd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$60F40020_h$

Subindex	04_h
Name	4th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60430010_h

Subindex	05_h
Name	5th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60440010_h

Subindex	06_h
Name	6th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$606B0020_h$

Subindex	07_h
Name	7th Object To Be Mapped
Datentyp	UNSIGNED32

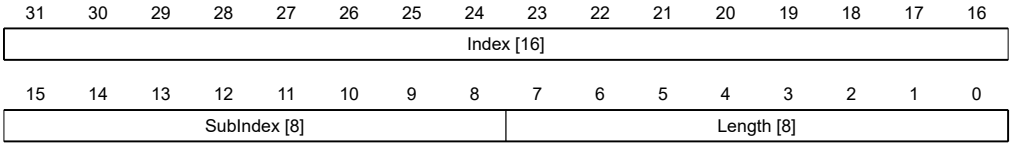
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$606C0020_h$

Subindex	08_h
Name	8th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	60770010_h

Beschreibung

Jeder Subindex (1-8) beschreibt jeweils ein gemapptes Objekt.

Ein Mapping-Eintrag besteht aus vier Bytes, die sich nach folgender Grafik zusammensetzen.



Index [16]

Darin ist der Index des zu mappenden Objektes enthalten.

Subindex [8]

Darin ist der Subindex des zu mappenden Objektes enthalten.

Length [8]

Darin ist die Länge des zu mappenden Objektes in der Einheit Bit enthalten.

1A02h Transmit PDO 3 Mapping Parameter

Funktion

Dieses Objekt enthält die Mapping-Parameter für PDOs, welche die Steuerung senden kann (TX-PDO 3).

Objektbeschreibung

Index	$1A02_h$
Objektname	Transmit PDO 3 Mapping Parameter
Object Code	RECORD
Datentyp	PDO_MAPPING
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426

Änderungshistorie	Firmware Version FIR-v1426: Eintrag "Überschrift" geändert von "1A02h Velocity Status" auf "1A02h Transmit PDO 3 Mapping Parameter". Firmware Version FIR-v1426: Eintrag "Object Name" geändert von "Velocity Status" auf "Transmit PDO 3 Mapping Parameter".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	01_h
Name	1st Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	2nd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	3rd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	04_h
Name	4th Object To Be Mapped
Datentyp	UNSIGNED32

Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	05_h
Name	5th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	06_h
Name	6th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

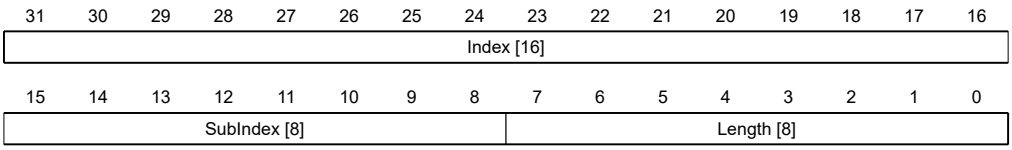
Subindex	07_h
Name	7th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	08_h
Name	8th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Jeder Subindex (1-8) beschreibt jeweils ein gemapptes Objekt.

Ein Mapping-Eintrag besteht aus vier Bytes, die sich nach folgender Grafik zusammensetzen.



Index [16]

Darin ist der Index des zu mappenden Objektes enthalten.

Subindex [8]

Darin ist der Subindex des zu mappenden Objektes enthalten.

Length [8]

Darin ist die Länge des zu mappenden Objektes in der Einheit Bit enthalten.

1A03h Transmit PDO 4 Mapping Parameter

Funktion

Dieses Objekt enthält die Mapping-Parameter für PDOs, welche die Steuerung senden kann (TX-PDO 4).

Objektbeschreibung

Index	$1A03_h$
Objektname	Transmit PDO 4 Mapping Parameter
Object Code	RECORD
Datentyp	PDO_MAPPING
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1426: Eintrag "Überschrift" geändert von "1A03h Input Status" auf "1A03h Transmit PDO 4 Mapping Parameter". Firmware Version FIR-v1426: Eintrag "Object Name" geändert von "Input Status" auf "Transmit PDO 4 Mapping Parameter".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	01_h
Name	1st Object To Be Mapped
Datentyp	UNSIGNED32

Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	2nd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	3rd Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	04_h
Name	4th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	05_h
Name	5th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	06_h
Name	6th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	00000000_h

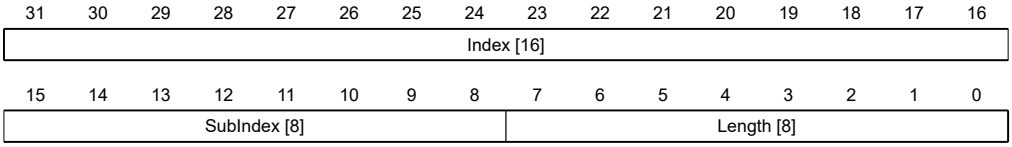
Subindex	07_h
Name	7th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	08_h
Name	8th Object To Be Mapped
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Jeder Subindex (1-8) beschreibt jeweils ein gemapptes Objekt.

Ein Mapping-Eintrag besteht aus vier Bytes, die sich nach folgender Grafik zusammensetzen.



Index [16]

Darin ist der Index des zu mappenden Objektes enthalten.

Subindex [8]

Darin ist der Subindex des zu mappenden Objektes enthalten.

Length [8]

Darin ist die Länge des zu mappenden Objektes in der Einheit Bit enthalten.

1F50h Program Data

Funktion

Dieses Objekt wird zum Programmieren von Speicherbereichen der Steuerung verwendet. Jeder Eintrag steht für einen bestimmten Speicherbereich.

Objektbeschreibung

Index	$1F50_h$
Objektname	Program Data

Object Code	ARRAY
Datentyp	DOMAIN
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1540
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	03_h

Subindex	01_h
Name	Program Data Bootloader/firmware
Datentyp	DOMAIN
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0

Subindex	02_h
Name	Program Data NanoJ
Datentyp	DOMAIN
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0

Subindex	03_h
Name	Program Data DataFlash
Datentyp	DOMAIN
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0

Beschreibung

1F51h Program Control

Funktion

Dieses Objekt wird zum Steuern des Programmierens von Speicherbereichen der Steuerung verwendet. Jeder Eintrag steht für einen bestimmten Speicherbereich.

Objektbeschreibung

Index	$1F51_h$
Objektname	Program Control
Object Code	ARRAY
Datentyp	UNSIGNED8
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1540
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	03_h

Subindex	01_h
Name	Program Control Bootloader/firmware
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	02_h
Name	Program Control NanoJ
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein

Zulässige Werte	
Vorgabewert	00_h

Subindex	03_h
Name	Program Control DataFlash
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Beschreibung

1F57h Program Status

Funktion

Dieses Objekt zeigt den Programmierstatus während dem Programmieren von Speicherbereichen der Steuerung an. Jeder Eintrag steht für einen bestimmten Speicherbereich.

Objektbeschreibung

Index	$1F57_h$
Objektname	Program Status
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1540
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	03_h

Subindex	01_h

Name	Program Status Bootloader/firmware
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Program Status NanoJ
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	Program Status DataFlash
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

2030h Pole Pair Count

Funktion

Enthält die Polpaarzahl des angeschlossenen Motors.

Objektbeschreibung

Index	2030_h
Objektname	Pole Pair Count
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Tuning
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000032_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Saveable" geändert von "nein" auf "ja, Kategorie: Tuning".

2031h Maximum Current

Funktion

Ist die **I^2t-Überwachung** nicht aktiv, wird hier der im Motordatenblatt angegebene Effektivstrom in mA eingetragen. Wird die **Closed Loop** Betriebsart verwendet oder ist die **I^2t-Überwachung** aktiviert, wird hier der Maximalstromwert in mA angegeben.

Steuerungsintern wird der eingegebene Wert immer als Effektivwert interpretiert.

Objektbeschreibung

Index	2031_h
Objektname	Maximum Current
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Tuning
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000258_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1614: Eintrag "Speicherbar" geändert von "ja, Kategorie: Applikation" auf "ja, Kategorie: Tuning". Firmware Version FIR-v1614: Eintrag "Object Name" geändert von "Peak Current" auf "Max Current".

2032h Maximum Speed

Funktion

Gibt die maximal zulässige Geschwindigkeit des Motors in **benutzerdefinierten Einheiten** an.

Objektbeschreibung

Index	2032_h
Objektname	Maximum Speed
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Tuning
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$00030D40_h$
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1614: Eintrag "Speicherbar" geändert von "ja, Kategorie: Applikation" auf "ja, Kategorie: Tuning".

Beschreibung

Hinweis
Das Objekt wird in den Betriebsmodi **Cyclic Synchronous Velocity** und **Homing** nicht berücksichtigt. In den Betriebsmodi **Velocity** und **Profile Velocity** wird es berücksichtigt nur, wenn eine S-Rampe (Positionsrampe, siehe **3202h Motor Drive Submode Select**) verwendet wird.

2033h Plunger Block

Funktion

Dieses Objekt verhindert ein zu weites Fahren in eine unerwünschte Richtung.

Objektbeschreibung

Index	2033_h
Objektname	Plunger Block
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Damit wird ein elektronischer Sperr-Riegel realisiert.

Der Wert 0 schaltet die Überwachung ab.

Der Wert 100 bedeutet beispielsweise, dass sich der Antrieb beliebig weit in die negative Richtung drehen darf, sobald er sich jedoch um mehr als 100 Schritte in die positive Richtung bewegt, wird der Motor sofort gestoppt und ein Fehler ausgelöst.

Dadurch kann z. B. beim Aufwickeln von Fäden ein versehentliches Abwickeln unterbunden werden.

2034h Upper Voltage Warning Level

Funktion

Dieses Objekt enthält den Schwellenwert für den Fehler "Überspannung" in Millivolt.

Objektbeschreibung

Index	2034_h
Objektname	Upper Voltage Warning Level
Object Code	VARIABLE
Datentyp	UNSIGNED32

Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$0000D2F0_h$
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Steigt die Eingangsspannung der Steuerung über diesen Schwellenwert, wird der Motor abgeschaltet und ein Fehler ausgelöst. Dieser Fehler setzt sich automatisch zurück, wenn die Eingangsspannung kleiner als (Spannung des Objekts 2034_h minus 2 Volt) ist.

2035h Lower Voltage Warning Level

Funktion

Dieses Objekt enthält den Schwellenwert für den Fehler "Unterspannung" in Millivolt.

Objektbeschreibung

Index	2035_h
Objektname	Lower Voltage Warning Level
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00002710_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Fällt die Eingangsspannung der Steuerung unter diesen Schwellenwert, wird der Motor abgeschaltet und ein Fehler ausgelöst. Der Fehler setzt sich automatisch zurück, wenn die Eingangsspannung größer als die Spannung des Objekts **2035_h** plus 2 Volt ist.

2036h Open Loop Current Reduction Idle Time

Funktion

Dieses Objekt beschreibt die Zeit in Millisekunden, die sich der Motor im Stillstand befinden muss, bis die Stromabsenkung aktiviert wird.

Objektbeschreibung

Index	2036_h

Objektname	Open Loop Current Reduction Idle Time
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$000003E8_h$
Firmware Version	FIR-v1426
Änderungshistorie	

2037h Open Loop Current Reduction Value/factor

Funktion

Dieses Objekt beschreibt den Effektivstrom, auf den der Motorstrom reduziert werden soll, wenn die Stromabsenkung im Open Loop aktiviert wird (Bit 3 in **3202_h** = "1") und sich der Motor im Stillstand befindet.

Objektbeschreibung

Index	2037_h
Objektname	Open Loop Current Reduction Value/factor
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$FFFFFFCE_h$
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Wert von 2037_h größer/gleich 0 und kleiner als Wert 2031_h

Strom wird auf den hier eingetragenen Wert reduziert. Der Wert wird in mA und als Effektivwert interpretiert.

Wert von 2037_h im Bereich von -1 bis -100

Der eingetragene Wert wird als eine Prozentzahl interpretiert und bestimmt die Reduktion des Nennstroms in **2037_h**. Für die Berechnung wird der Wert in **2031_h** herangezogen.

Beispiel: Das Objekt **2031_h** hat den Wert 4200 mA. Der Wert -60 in **2037_h** senkt den Strom um 60% von **2031_h** ab, somit ergibt sich eine Stromabsenkung auf einen Effektivwert von **2031_h** * (**2037_h** + 100) / 100 = 1680 mA.

Die Angabe -100 in **2037_h** würde z.B. bedeuten, dass eine Stromabsenkung auf einen Effektivwert von 0 mA eingestellt wird.

Hinweis
Falls ein Nennstrom größer 0 in **203B**$_h$:01 eingetragen ist, wird der kleinere Wert von **2031**$_h$ und **203B**$_h$:01 als Nennstrom zur Berechnung der Stromreduzierung herangezogen.

2038h Brake Controller Timing

Funktion

Dieses Objekt enthält die Zeiten für die *Bremsensteuerung* in Millisekunden sowie die PWM-Frequenz und den Tastgrad.

Objektbeschreibung

Index	2038$_h$
Objektname	Brake Controller Timing
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00$_h$
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	06$_h$

Subindex	01$_h$
Name	Close Brake Idle Time
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	000003E8$_h$

Subindex	02$_h$
Name	Shutdown Power Idle Time
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	$000003E8_h$

Subindex	03_h
Name	Open Brake Delay Time
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$000003E8_h$

Subindex	04_h
Name	Start Operation Delay Time
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	05_h
Name	PWM Frequency
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	zwischen 0 und 2000 ($7D0_h$)
Vorgabewert	00000000_h

Subindex	06_h
Name	PWM Duty Cycle
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	0, zwischen 2 und 100 (64_h)
Vorgabewert	00000000_h

Beschreibung

Die Subindizes haben folgende Funktionen:

- 01_h: Zeit zwischen dem Motorstillstand und dem Schließen der Bremse.
- 02_h: Zeit zwischen dem Schließen der Bremse und dem Abschalten des Motorstroms.
- 03_h: Zeit zwischen dem Einschalten des Motorstroms und dem Öffnen der Bremse.
- 04_h: Zeit zwischen dem Öffnen der Bremse und dem Erreichen des Zustands *Operation enabled* der **CiA 402 Power State Machine**.
- 05_h: Frequenz der Bremsen-PWM in Hertz.
- 06_h: Tastgrad der Bremsen-PWM in Prozent.

2039h Motor Currents

Funktion

Dieses Objekt enthält die gemessenen Motorströme in mA.

Objektbeschreibung

Index	2039_h
Objektname	Motor Currents
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	nein
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1504: Tabellen-Eintrag "PDO-Mapping" bei Subindex 01 geändert von "nein" auf "TX-PDO". Firmware Version FIR-v1504: Tabellen-Eintrag "PDO-Mapping" bei Subindex 02 geändert von "nein" auf "TX-PDO". Firmware Version FIR-v1504: Tabellen-Eintrag "PDO-Mapping" bei Subindex 03 geändert von "nein" auf "TX-PDO". Firmware Version FIR-v1504: Tabellen-Eintrag "PDO-Mapping" bei Subindex 04 geändert von "nein" auf "TX-PDO".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	04_h

Subindex	01_h
Name	I_d
Datentyp	INTEGER32
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	I_q
Datentyp	INTEGER32
Zugriff	nur lesen
PDO-Mapping	TX-PDO

Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	I_a
Datentyp	INTEGER32
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	04_h
Name	I_b
Datentyp	INTEGER32
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

203Ah Homing On Block Configuration

Funktion

Dieses Objekt enthält die Parameter für das *Homing auf Block* (siehe Kapitel **Homing**)

Objektbeschreibung

Index	$203A_h$
Objektname	Homing On Block Configuration
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Zugriff	
PDO-Mapping	
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Die Anzahl der Einträge haben sich geändert von 4 auf 3. Firmware Version FIR-v1540: Eintrag "Name" geändert von "Period Of Blocking" auf "Block Detection Time". Firmware Version FIR-v1614: Eintrag "Data Type" geändert von "UNSIGNED32" auf "INTEGER32". Firmware Version FIR-v1614: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

	Firmware Version FIR-v1614: Eintrag "Data type" geändert von "UNSIGNED32" auf "INTEGER32". Firmware Version FIR-v1614: Eintrag "Data type" geändert von "UNSIGNED32" auf "INTEGER32".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Minimum Current For Block Detection
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$FFFFFFBA_h$

Subindex	02_h
Name	Block Detection Time
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$000000C8_h$

Beschreibung

Die Subindizes haben folgende Funktion:

- 01_h: Gibt den Stromgrenzwert an, ab dem ein Blockieren detektiert werden soll. Positive Zahlenwerte geben die Stromgrenze in mA an, negative Zahlen einen Prozentwert von Objekt **2031**$_h$:01_h. Beispiel: der Wert "1000" entspricht 1000 mA (=1 A), der Wert "-70" entspricht 70% von **2031**$_h$.
- 02_h: Gibt die Zeit in ms an, die der Motor nach der Blockdetektion trotzdem noch gegen den Block fahren soll.

203Bh I2t Parameters

Funktion

Dieses Objekt hält die Parameter für die I^2t-Überwachung.

Die I^2t-Überwachung wird aktiviert, in dem in **$203B_h$**:01 und **$203B_h$**:02 ein Wert größer 0 eingetragen wird (siehe **I2t Motor-Überlastungsschutz**).

I^2t kann nur für den *Closed Loop*-Betrieb verwendet werden, mit einer Ausnahme: Wenn I^2t im *Open Loop*-Betrieb aktiviert ist, wird der Strom auf den kleineren der beiden Werte von **$203B_h$** und **2031_h** begrenzt.

Objektbeschreibung

Index	$203B_h$
Objektname	I2t Parameters
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Tuning
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1512: Eintrag "Savable" geändert von "nein" auf "ja, Kategorie: Applikation". Firmware Version FIR-v1512: Die Anzahl der Einträge haben sich geändert von 7 auf 8. Firmware Version FIR-v1614: Eintrag "Speicherbar" geändert von "ja, Kategorie: Applikation" auf "ja, Kategorie: Tuning".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	07_h

Subindex	01_h
Name	Nominal Current
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Maximum Duration Of Peak Current
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	00000000_h

Subindex	03_h
Name	Threshold
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	04_h
Name	CalcValue
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	05_h
Name	LimitedCurrent
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	06_h
Name	Status
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	07_h
Name	ActualResistance
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Die Subindizes sind in zwei Gruppen geteilt: Subindex 01_h und 02_h enthalten Parameter zur Steuerung, Subindex 03_h bis 06_h sind Statuswerte. Die Funktionen sind wie folgt:

- 01_h: Hier wird der im Motordatenblatt angegebene Nennstrom in mA eingetragen. Dieser muss kleiner als der eingegebene Strom in Objekt **2031**$_h$ sein, sonst wird die Überwachung nicht aktiviert. Der angegebene Wert wird als Effektivwert interpretiert.
- 02_h: Gibt die maximale Dauer des Spitzenstroms in ms an.
- 03_h: Threshold, gibt die Grenze in mA an, von der abhängt, ob auf Maximalstrom oder Nennstrom geschalten wird.
- 04_h: CalcValue, gibt den berechneten Wert an, welcher mit Threshold verglichen wird, um den Strom einzustellen.
- 05_h: LimitedCurrent, zeigt den gegenwärtigen Strom als Effektivwert an, der von I^2t eingestellt wurde.
- 06_h: aktueller Status. Ist der Subentry-Wert "0", ist I^2t deaktiviert, ist der Wert "1", wird I^2t aktiviert.

203Dh Torque Window

Funktion

Gibt relativ zum Zieldrehmoment einen symmetrischen Bereich an, innerhalb dessen das Ziel als erreicht gilt.

Wird der Wert auf "FFFFFFFF"$_h$ gesetzt, wird die Überwachung abgeschaltet, das Bit "Target reached" im Objekt **6041**$_h$ (Statusword) wird nie gesetzt.

Objektbeschreibung

Index	$203D_h$
Objektname	Torque Window
Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1540
Änderungshistorie	Firmware Version FIR-v1614: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

203Eh Torque Window Time

Funktion

Das Istdrehmoment muss sich für diese Zeit (in Millisekunden) innerhalb des "Torque Window" (**203D**$_h$) befinden, damit das Zieldrehmoment als erreicht gilt.

Objektbeschreibung

Index	$203E_h$
Objektname	Torque Window Time

Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1540
Änderungshistorie	Firmware Version FIR-v1614: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

2050h Encoder Alignment

Funktion

Dieser Wert gibt den Versatz zwischen dem Index des Encoders und dem elektrischen Feld an.

Objektbeschreibung

Index	2050_h
Objektname	Encoder Alignment
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Tuning
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Saveable" geändert von "nein" auf "ja, Kategorie: Tuning".

Beschreibung

Die exakte Bestimmung ist nur über das **Auto-Setup** möglich. Das Vorhandensein dieses Wertes ist für den *Closed Loop*-Betrieb mit Encoder erforderlich.

2051h Encoder Optimization

Funktion

Enthält Kompensationswerte, um einen besseren Rundlauf im *Closed Loop*-Betrieb zu erreichen.

Objektbeschreibung

Index	2051_h
Objektname	Encoder Optimization
Object Code	ARRAY

Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Tuning
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Saveable" geändert von "nein" auf "ja, Kategorie: Tuning".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	03_h

Subindex	01_h
Name	Parameter 1
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Parameter 2
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	Parameter 3
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Die exakte Bestimmung ist nur über das **Auto-Setup** möglich.

2052h Encoder Resolution

Funktion

Beinhaltet die physikalische Auflösung des Encoders, der zur Kommutierung verwendet wird.

Objektbeschreibung

Index	2052_h
Objektname	Encoder Resolution
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Tuning
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Saveable" geändert von "nein" auf "ja, Kategorie: Tuning".

Beschreibung

Ein negativer Wert bedeutet, dass der Encoder gegensinnig zum Motor angetrieben wird. Dies lässt sich durch Umpolen einer Motorwicklung korrigieren.

Tipp

Die Einheit ist "Flanken pro Umdrehung" (ppr), welche dem vierfachen der Auflösung in "Inkrementen pro Umdrehung" (cpr) entspricht (Quadratur). Das bedeutet, dass bei einem Encoder, dessen Auflösung beispielsweise 1000 Inkremente pro Umdrehung ist, der Wert im 2052_h 4000 ist.

2056h Limit Switch Tolerance Band

Funktion

Gibt an, wie weit positive oder negative Endschalter überfahren werden dürfen, bis die Steuerung einen Fehler auslöst.

Dieses Toleranzband ist beispielsweise erforderlich, um Referenzfahrten - bei denen Endschalter betätigt werden können - fehlerfrei abschließen zu können.

Objektbeschreibung

Index	2056_h
Objektname	Limit Switch Tolerance Band
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben

PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	$000001F4_h$
Firmware Version	FIR-v1426
Änderungshistorie	

2057h Clock Direction Multiplier

Funktion

Mit diesem Wert wird der Takt-Zählwert im Takt-/Richtungsmodus multipliziert, bevor er weiterverarbeitet wird.

Objektbeschreibung

Index	2057_h
Objektname	Clock Direction Multiplier
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000080_h
Firmware Version	FIR-v1426
Änderungshistorie	

2058h Clock Direction Divider

Funktion

Durch diesen Wert wird der Takt-Zählwert im Takt-/Richtungsmodus dividiert, bevor er weiterverarbeitet wird.

Objektbeschreibung

Index	2058_h
Objektname	Clock Direction Divider
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h
Firmware Version	FIR-v1426
Änderungshistorie	

2059h Encoder Configuration

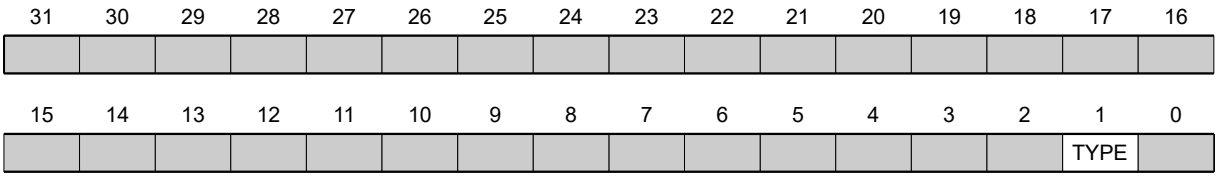
Funktion

Mit diesem Objekt kann die Versorgungsspannung und der Typ des Encoders umgeschaltet werden.

Objektbeschreibung

Index	2059_h
Objektname	Encoder Configuration
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Tuning
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1614: Eintrag "Speicherbar" geändert von "ja, Kategorie: Applikation" auf "ja, Kategorie: Tuning".

Beschreibung



TYPE

Legt den Typ des Encoders fest. Das Bit muss den Wert "0" bei einem differentiellen Encoder haben. Für einen single-ended Encoder muss das Bit auf "1" gesetzt werden.

205Ah Encoder Boot Value

Funktion

Tipp

Dieses Objekt hat nur bei Verwendung eines Absolut-Encoders eine Funktion. Wird kein Absolut-Encoder verwendet, ist der Wert immer 0.

Aus diesem Objekt kann die initiale Encoderposition beim Einschalten der Steuerung (in **benutzerdefinierten Einheiten**) ausgelesen werden.

Objektbeschreibung

Index	$205A_h$
Objektname	Encoder Boot Value

Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1446
Änderungshistorie	Firmware Version FIR-v1512: Tabellen-Eintrag "Zugriff" bei Subindex 00 geändert von "lesen/schreiben" auf "nur lesen".

205Bh Clock Direction Or Clockwise/Counter Clockwise Mode

Funktion

Mit diesem Objekt lässt sich der Takt-Richtungs-Modus (Wert = "0") auf den Rechts-/Linkslauf-Modus (Wert = "1") umschalten.

Objektbeschreibung

Index	$205B_h$
Objektname	Clock Direction Or Clockwise/Counter Clockwise Mode
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1504
Änderungshistorie	

2060h Compensate Polepair Count

Funktion

Ermöglicht, motorunabhängig Fahrsätze zu beauftragen.

Objektbeschreibung

Index	2060_h
Objektname	Compensate Polepair Count
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	00000001_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Wird dieser Eintrag auf 1 gesetzt, wird die Polpaarzahl automatisch bei allen Geschwindigkeits-, Beschleunigungs- und Jerk-Parametern eingerechnet.

Ist der Wert 0, geht die **Polpaarzahl**, wie bei herkömmlichen Schrittmotorsteuerungen, in die Vorgabewerte mit ein und muss bei einem Motorwechsel berücksichtigt werden.

2061h Velocity Numerator

Funktion

Beinhaltet den Zähler, der zum Umrechnen von benutzerdefinierten Geschwindigkeitswerten in die internen Umdrehungen/Sekunde verwendet wird. Siehe Kapitel **Benutzerdefinierte Einheiten**.

Objektbeschreibung

Index	2061_h
Objektname	Velocity Numerator
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h
Firmware Version	FIR-v1426
Änderungshistorie	

2062h Velocity Denominator

Funktion

Beinhaltet den Nenner, der zum Umrechnen von benutzerdefinierten Geschwindigkeitswerten in die internen Umdrehungen/Sekunde verwendet wird. Siehe Kapitel **Benutzerdefinierte Einheiten**.

Objektbeschreibung

Index	2062_h
Objektname	Velocity Denominator
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	0000003C$_h$
Firmware Version	FIR-v1426
Änderungshistorie	

2063h Acceleration Numerator

Funktion

Beinhaltet den Zähler, der zum Umrechnen von benutzerdefinierten Beschleunigungswerten in die internen Umdrehungen/Sekunde^2 verwendet wird. Siehe Kapitel **Benutzerdefinierte Einheiten**.

Objektbeschreibung

Index	2063$_h$
Objektname	Acceleration Numerator
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001$_h$
Firmware Version	FIR-v1426
Änderungshistorie	

2064h Acceleration Denominator

Funktion

Beinhaltet den Nenner, der zum Umrechnen von benutzerdefinierten Beschleunigungswerten in die internen Umdrehungen/Sekunde^2 verwendet wird. Siehe Kapitel **Benutzerdefinierte Einheiten**.

Objektbeschreibung

Index	2064$_h$
Objektname	Acceleration Denominator
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000003C$_h$
Firmware Version	FIR-v1426
Änderungshistorie	

2065h Jerk Numerator

Funktion

Beinhaltet den Zähler, der zum Umrechnen von benutzerdefinierten Ruckwerten in die internen Umdrehungen/Sekunde ^{3} verwendet wird. Siehe Kapitel **Benutzerdefinierte Einheiten**.

Objektbeschreibung

Index	2065_h
Objektname	Jerk Numerator
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h
Firmware Version	FIR-v1426
Änderungshistorie	

2066h Jerk Denominator

Funktion

Beinhaltet den Nenner, der zum Umrechnen von benutzerdefinierten Ruckwerten in die internen Umdrehungen/Sekunde ^{3} verwendet wird. Siehe Kapitel **Benutzerdefinierte Einheiten**.

Objektbeschreibung

Index	2066_h
Objektname	Jerk Denominator
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$0000003C_h$
Firmware Version	FIR-v1426
Änderungshistorie	

2084h Bootup Delay

Funktion

Definiert den Zeitraum zwischen Anlegen der Versorgungsspannung an die Steuerung und der Funktionsbereitschaft der Steuerung in Millisekunden.

Objektbeschreibung

Index	2084_h
Objektname	Bootup Delay
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	

2101h Fieldbus Module Availability

Funktion

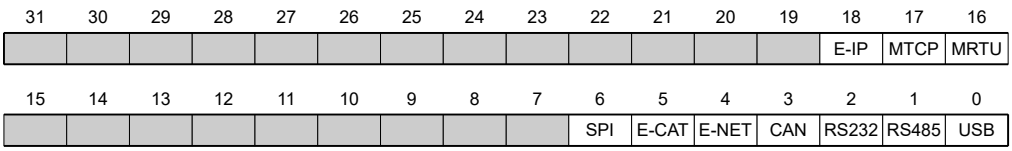
Zeigt die verfügbaren Feldbusse an.

Objektbeschreibung

Index	2101_h
Objektname	Fieldbus Module Availability
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000040_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1626: Eintrag "Object Name" geändert von "Fieldbus Module" auf "Fieldbus Module Availability".

Beschreibung

Die Bits 0 bis 15 zeigen die physikalische Schnittstelle an, die Bits 16 bis 31 das benutzte Protokoll (falls notwendig).



USB

Wert = "1": Der Feldbus USB ist verfügbar.

RS-485
Wert = "1": Eine RS-485 Schnittstelle ist verfügbar.

RS-232
Wert = "1": Eine RS-232 Schnittstelle ist verfügbar.

CAN
Wert = "1": Der Feldbus CANopen ist verfügbar.

E-NET
Wert = "1": Eine Ethernet Schnittstelle ist verfügbar.

E-CAT
Wert = "1": Eine EtherCAT Schnittstelle ist verfügbar.

SPI
Wert = "1": Eine SPI Schnittstelle ist verfügbar.

MRTU
Wert = "1": Das benutzte Protokoll ist Modbus RTU.

MTCP
Wert = "1": Das benutzte Protokoll ist Modbus TCP

E-IP
Wert = "1": Das benutzte Protokoll ist EtherNet/IP

2102h Fieldbus Module Control

Funktion

Mit diesem Objekt können bestimmte Feldbusse (physikalischen Schnittstellen und Protokolle) aktiviert/deaktiviert werden.

Objektbeschreibung

Index	2102_h
Objektname	Fieldbus Module Control
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Kommunikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000040_h
Firmware Version	FIR-v1540
Änderungshistorie	Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "ja, Kategorie: Applikation" auf "ja, Kategorie: Kommunikation".

Beschreibung

Im Objekt **2103**$_h$:1$_h$ werden alle physikalischen Schnittstellen/Protokolle angezeigt, welche aktiviert/deaktiviert werden können. Diese können in diesem Objekt (2102$_h$) geschaltet werden. Der gegenwärtige Status der aktivierten Feldbusse steht im Objekt **2103**$_h$:2$_h$.

Dabei gilt die folgende Verteilung der Bits:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
													E-IP	MTCP	MRTU

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									SPI	E-CAT	E-NET	CAN	RS232	RS485	USB

USB
USB Schnittstelle

RS-485
RS-485 Schnittstelle

RS-232
RS-232 Schnittstelle

CAN
CANopen Schnittstelle

E-NET
EtherNET Schnittstelle

E-CAT
EtherCAT Schnittstelle

SPI
SPI Schnittstelle

MRTU
Modbus RTU Protokoll

MTCP
Modbus TCP Protokoll

E-IP
EtherNet/IP Protokoll

2103h Fieldbus Module Status

Funktion

Zeigt die aktiven Feldbusse an.

Objektbeschreibung

Index	2103$_h$
Objektname	Fieldbus Module Status
Object Code	ARRAY
Datentyp	UNSIGNED32

Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1540
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Fieldbus Module Disable Mask
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

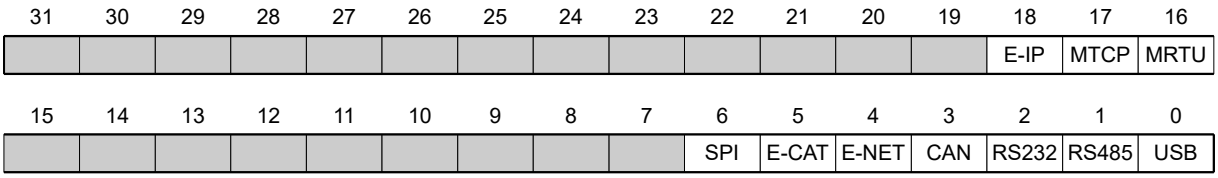
Subindex	02_h
Name	Fieldbus Module Enabled
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000040_h

Beschreibung

Subindex 1 (Fieldbus Module Disable Mask): Im diesem Subindex werden alle physikalischen Schnittstellen und Protokolle angezeigt, welche aktiviert oder deaktiviert werden können. Ein Wert "1" bedeutet, dass dieser Feldbus deaktivierbar ist.

Subindex 2 (Fieldbus Module Enabled): Dieser Subindex zeigt alle zur Zeit aktivierten physikalischen Schnittstellen und Protokolle an. Der Wert "1" bedeutet, dass der Feldbus aktiv ist.

Für Subindex 1 und 2 gilt folgende Verteilung der Bits:



USB
USB Schnittstelle

RS-485
RS-485 Schnittstelle

RS-232
RS-232 Schnittstelle

CAN
CANopen Schnittstelle

E-NET
EtherNET Schnittstelle

E-CAT
EtherCAT Schnittstelle

SPI
SPI Schnittstelle

MRTU
Modbus RTU Protokoll

MTCP
Modbus TCP Protokoll

E-IP
EtherNet/IP Protokoll

2300h NanoJ Control

Funktion

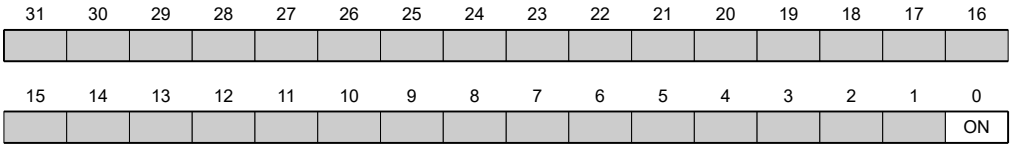
Steuert die Ausführung eines NanoJ-Programms.

Objektbeschreibung

Index	2300_h
Objektname	NanoJ Control
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1436: Eintrag "Object Name" geändert von "VMM Control" auf "NanoJ Control".

Beschreibung



ON

Schaltet das NanoJ-Programm ein (Wert = "1") oder aus (Wert = "0").

Bei einer steigenden Flanke in Bit 0 wird das Programm zuvor neu geladen und der Variablenbereich zurückgesetzt.

Hinweis

Das Starten des NanoJ Programms kann bis zu 200ms dauern.

2301h NanoJ Status

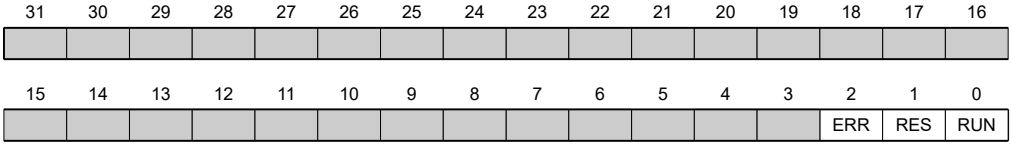
Funktion

Zeigt den Betriebszustand des Benutzerprogramms an.

Objektbeschreibung

Index	2301_h
Objektname	NanoJ Status
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1436: Eintrag "Object Name" geändert von "VMM Status" auf "NanoJ Status".

Beschreibung



RUN

Wert = "0": Programm ist angehalten, Wert = "1": NanoJ-Programm läuft .

RES

Reserviert.

ERR

Programm wurde mit Fehler beendet. Fehlerursache kann aus dem Objekt **2302**$_h$ ausgelesen werden.

2302h NanoJ Error Code

Funktion

Zeigt an, welcher Fehler bei der Ausführung des Benutzerprogramms aufgetreten ist.

Objektbeschreibung

Index	2302$_h$
Objektname	NanoJ Error Code
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000$_h$
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1436: Eintrag "Object Name" geändert von "VMM Error Code" auf "NanoJ Error Code".

Beschreibung

Fehlercodes bei Programmausführung:

Nummer	Beschreibung
0000$_h$	Kein Fehler
0001$_h$	Firmware unterstützt verwendete Funktion (noch) nicht
0002$_h$	Nicht oder falsch initialisierter Pointer
0003$_h$	Unerlaubter Zugriff auf System-Resource
0004$_h$	Hardfault (interner Fehler)
0005$_h$	Code wird zu lange ohne yield() oder sleep() ausgeführt

Nummer	Beschreibung
0006_h	Unerlaubter Zugriff auf System-Resource
0007_h	Zu viele Variablen auf dem Stack
0100_h	Ungültige NanoJ Programmdatei

Fehler bei dem Zugriff auf ein Objekt:

Nummer	Beschreibung
$10xxxxyy_h$	Ungültiges Mapping in der NanoJ-Programmdatei: Der Wert in "xxxx" benennt den Index, der Wert in "yy" den Subindex des Objekts, das gemappt werden soll aber nicht gemappt werden kann.
1000_h	Zugriff auf ein nicht existierendes Objekt im Objektverzeichnis
1001_h	Schreibzugriff auf schreibgeschützten Eintrag im OD
1002_h	Interner Dateisystemfehler

Dateisystem Fehlercodes beim Laden des Benutzerprogramms:

Nummer	Beschreibung
10002_h	Interner Dateisystemfehler
10003_h	Speichermedium nicht bereit
10004_h	Datei nicht gefunden
10005_h	Ordner nicht gefunden
10006_h	Ungültiger Dateiname/Ordnername
10008_h	Zugriff auf Datei nicht möglich
10009_h	Datei/Verzeichnis Objekt ist ungültig
$1000A_h$	Speicherrmedium ist schreibgeschützt
$1000B_h$	Laufwerksnummer ist ungültig
$1000C_h$	Arbeitsbereich des Laufwerks ist ungültig
$1000D_h$	Kein gültiges Dateisystem auf dem Laufwerk
$1000E_h$	Erstellung des Dateisystems ist fehlgeschlagen
$1000F_h$	Zugriff innerhalb der geforderten Zeit nicht möglich
10010_h	Zugriff wurde zurückgewiesen

230Fh Uptime Seconds

Funktion

Dieses Objekt enthält die Betriesbstunden seit dem letzen Start der Steuerung in Sekunden.

Hinweis

Dieses Objekt wird nicht gespeichert, die Zählung beginnt nach dem Einschalten wieder mit "0".

Objektbeschreibung

Index	$230F_h$

Objektname	Uptime Seconds
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1436
Änderungshistorie	

2310h NanoJ Input Data Selection

Funktion

Beschreibt die Object Dictionary-Einträge, die in das Input PDO-Mapping des NanoJ-Programms kopiert werden.

Objektbeschreibung

Index	2310_h
Objektname	NanoJ Input Data Selection
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1650-B472161
Änderungshistorie	Firmware Version FIR-v1436: Eintrag "Object Name" geändert von "VMM Input Data Selection" auf "NanoJ Input Data Selection". Firmware Version FIR-v1650-B472161: Eintrag "Speicherbar" geändert von "ja, Kategorie: Applikation" auf "nein". Firmware Version FIR-v1650-B472161: Tabellen-Eintrag "Zugriff" bei Subindex 00 geändert von "lesen/schreiben" auf "nur lesen". Firmware Version FIR-v1650-B472161: Tabellen-Eintrag "Zugriff" bei Subindex 01 geändert von "lesen/schreiben" auf "nur lesen".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	

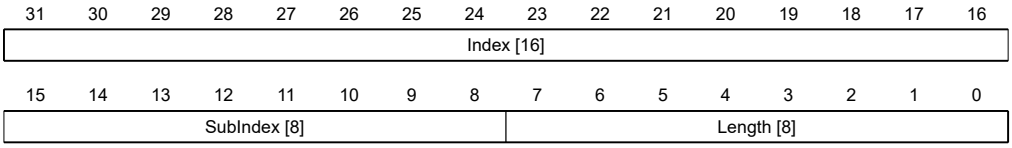
Vorgabewert	10_h

Subindex	01_h - 10_h
Name	Mapping #1 - #16
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Jeder Subindex (1-16) beschreibt jeweils ein gemapptes Objekt.

Ein Mapping-Eintrag besteht aus vier Bytes, die sich nach folgender Grafik zusammen setzen.



Index [16]

Darin ist der Index des zu mappenden Objektes enthalten

Subindex [8]

Darin ist der Subindex des zu mappenden Objektes enthalten

Length [8]

Darin ist die Länge des zu mappenden Objektes in der Einheit Bit enthalten.

2320h NanoJ Output Data Selection

Funktion

Beschreibt die Object Dictionary-Einträge, die in das Output PDO-Mapping des *NanoJ-Programms* kopiert werden, nachdem es ausgeführt worden ist.

Objektbeschreibung

Index	2320_h
Objektname	NanoJ Output Data Selection
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1650-B472161

Änderungshistorie	Firmware Version FIR-v1436: Eintrag "Object Name" geändert von "VMM Output Data Selection" auf "NanoJ Output Data Selection". Firmware Version FIR-v1650-B472161: Eintrag "Speicherbar" geändert von "ja, Kategorie: Applikation" auf "nein". Firmware Version FIR-v1650-B472161: Tabellen-Eintrag "Zugriff" bei Subindex 00 geändert von "lesen/schreiben" auf "nur lesen". Firmware Version FIR-v1650-B472161: Tabellen-Eintrag "Zugriff" bei Subindex 01 geändert von "lesen/schreiben" auf "nur lesen".

Wertebeschreibung

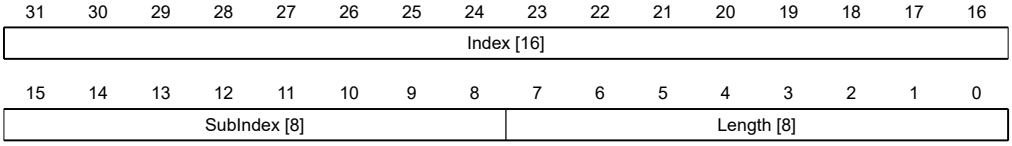
Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	10_h

Subindex	01_h - 10_h
Name	Mapping #1 - #16
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Jeder Subindex (1-16) beschreibt jeweils ein gemapptes Objekt.

Ein Mapping Eintrag besteht aus vier Byte welche sich nach nachfolgender Grafik zusammen setzen.



Index [16]

Darin ist der Index des zu mappenden Objektes enthalten

Subindex [8]

Darin ist der Subindex des zu mappenden Objektes enthalten

Length [8]

Darin ist die Länge des zu mappenden Objektes in der Einheit Bit enthalten.

2330h NanoJ In/output Data Selection

Funktion

Beschreibt die Object Dictionary-Einträge, die zunächst in das Input PDO-Mapping des NanoJ-Programms kopiert und nach dessen Ausführung wieder in das Output PDO-Mapping zurückkopiert werden.

Objektbeschreibung

Index	2330_h
Objektname	NanoJ In/output Data Selection
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1650-B472161
Änderungshistorie	Firmware Version FIR-v1436: Eintrag "Object Name" geändert von "VMM In/output Data Selection" auf "NanoJ In/output Data Selection". Firmware Version FIR-v1650-B472161: Eintrag "Speicherbar" geändert von "ja, Kategorie: Applikation" auf "nein". Firmware Version FIR-v1650-B472161: Tabellen-Eintrag "Zugriff" bei Subindex 00 geändert von "lesen/schreiben" auf "nur lesen". Firmware Version FIR-v1650-B472161: Tabellen-Eintrag "Zugriff" bei Subindex 01 geändert von "lesen/schreiben" auf "nur lesen".

Wertebeschreibung

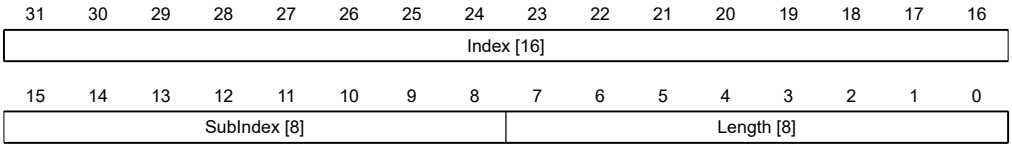
Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	10_h

Subindex	01_h - 10_h
Name	Mapping #1 - #16
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Jeder Subindex (1-16) beschreibt jeweils ein gemapptes Objekt.

Ein Mapping-Eintrag besteht aus vier Bytes, die sich nach folgender Grafik zusammen setzen.



Index [16]

Darin ist der Index des zu mappenden Objektes enthalten

Subindex [8]

Darin ist der Subindex des zu mappenden Objektes enthalten

Length [8]

Darin ist die Länge des zu mappenden Objektes in der Einheit Bit enthalten.

2400h NanoJ Inputs

Funktion

Hier befindet sich ein Array mit 32 32-Bit Integerwerten, das innerhalb der Firmware nicht verwendet wird und ausschließlich zur Kommunikation mit dem Benutzerprogramm über den Feldbus dient.

Objektbeschreibung

Index	2400_h
Objektname	NanoJ Inputs
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	nein
Firmware Version	FIR-v1426
Änderungshistorie	Die Anzahl der Einträge haben sich geändert von 2 auf 33 Firmware Version FIR-v1436: Eintrag "Object Name" geändert von "VMM Inputs" auf "NanoJ Inputs". Firmware Version FIR-v1436: Eintrag "Name" geändert von "VMM Input N#" auf "NanoJ Input N#".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	20_h

Subindex	01_h - 20_h
Name	NanoJ Input #1 - #32
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Hier können dem *NanoJ-Programm* z. B. Vorgabewerte übergeben werden.

2410h NanoJ Init Parameters

Funktion

Dieses Objekt funktioniert identisch dem Objekt **2400**$_h$ mit dem Unterschied, dass dieses Objekt gespeichert werden kann.

Objektbeschreibung

Index	2410_h
Objektname	NanoJ Init Parameters
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1450
Änderungshistorie	Firmware Version FIR-v1450: Eintrag "Data type" geändert von "INTEGER32" auf "UNSIGNED8".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	20_h

Subindex	01_h - 20_h
Name	NanoJ Init Parameter #1 - #32

Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

2500h NanoJ Outputs

Funktion

Hier befindet sich ein Array mit 32 32-Bit Integerwerten, das innerhalb der Firmware nicht verwendet wird und ausschließlich zur Kommunikation mit dem Benutzerprogramm über den Feldbus dient.

Objektbeschreibung

Index	2500_h
Objektname	NanoJ Outputs
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	nein
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1436: Eintrag "Object Name" geändert von "VMM Outputs" auf "NanoJ Outputs". Firmware Version FIR-v1436: Eintrag "Name" geändert von "VMM Output N#" auf "NanoJ Output N#".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	20_h

Subindex	01_h - 20_h
Name	NanoJ Output #1 - #32
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Hier kann das *NanoJ-Programm* Ergebnisse ablegen, die dann über den Feldbus ausgelesen werden können.

2600h NanoJ Debug Output

Funktion

Dieses Objekt enthält Debug-Ausgaben eines Benutzerprogramms.

Objektbeschreibung

Index	2600_h
Objektname	NanoJ Debug Output
Object Code	ARRAY
Datentyp	UNSIGNED8
Speicherbar	nein
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1436: Eintrag "Object Name" geändert von "VMM Debug Output" auf "NanoJ Debug Output".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	01_h - 40_h
Name	Value #1 - #64
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Beschreibung

Hier legt das NanoJ-Programm die Debug-Ausgaben ab, welche mit der Funktion `VmmDebugOutputString()`, `VmmDebugOutputInt()` und dergleichen aufgerufen wurden.

2701h Customer Storage Area

Funktion

In dieses Objekt können Daten abgelegt und gespeichert werden.

Objektbeschreibung

Index	2701_h
Objektname	Customer Storage Area
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Benutzer
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1540
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Data type" geändert von "UNSIGNED32" auf "UNSIGNED8".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	FE_h

Subindex	01_h - FE_h
Name	Storage #1 - #254
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

2800h Bootloader And Reboot Settings

Funktion

Mit diesem Objekt lässt sich ein Reboot der Firmware auslösen und das Kurzschließen der Motorwicklungen im Bootloader-Modus aus- und einschalten.

Objektbeschreibung

Index	2800_h
Objektname	Bootloader And Reboot Settings
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1540
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	03_h

Subindex	01_h
Name	Reboot Command
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Reboot Delay Time In Ms
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	Bootloader HW Config
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben

PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Die Subindizes haben folgende Funktion:

- 01_h: Wird hier der Wert $746F6F62_h$ eingetragen, wird die Firmware rebootet.
- 02_h: Zeit in Millisekunden: verzögert den Reboot der Firmware um die jeweilige Zeit.
- 03_h: mit dem Bit 0 kann das Kurzschließen der Motorwicklungen im Bootloader-Modus aus- und eingeschaltet werden:
  - Bit 0= 1 : Das Kurzschließen der Motorwicklungen im Bootloader-Modus wird ausgeschaltet.
  - Bit 0= 0 : Das Kurzschließen der Motorwicklungen im Bootloader-Modus wird eingeschaltet.

3202h Motor Drive Submode Select

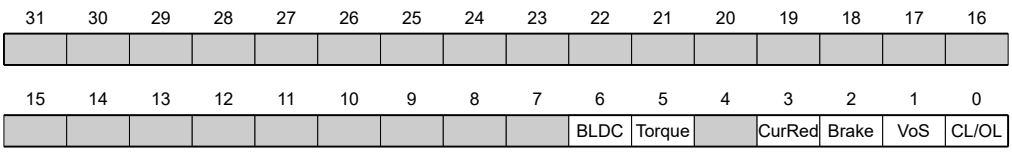
Funktion

Steuert die Reglerbetriebsart, wie z. B. die *Closed Loop*/ *Open Loop*-Umschaltung und ob der Velocity-Mode über den S-Regler simuliert wird oder mit einem echten V-Regler im *Closed Loop* arbeitet.

Objektbeschreibung

Index	3202_h
Objektname	Motor Drive Submode Select
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Bewegung
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Saveable" geändert von "ja, Kategorie: Applikation" auf "ja, Kategorie: Fahrt". Firmware Version FIR-v1540: Eintrag "Saveable" geändert von "ja, Kategorie: Fahrt" auf "ja, Kategorie: Bewegung".

Beschreibung



CL/OL

Umschaltung zwischen *Open Loop* und *Closed Loop*

- Wert = "0": *Open Loop*

- Wert = "1": *Closed Loop*

VoS

Wert = "1": V-Regler über eine S-Rampe simulieren: die Geschwindigkeitsmodi über kontinuierliche Positionsänderungen simulieren

Brake

Wert = "1": Einschalten der automatischen Bremsensteuerung.

CurRed (Current Reduction)

Wert = "1": Stromabsenkung im *Open Loop* aktiviert

Torque

nur in den Betriebsmodi **Profile Torque** und **Cyclic Synchronous Torque** aktiv

Wert = "1": M-Regler ist aktiv, andernfalls ist ein V-Regler überlagert: in den Torque-Modi wird kein V-Regler zur Geschwindigkeitsbegrenzung verwendet, das Objekt **2032**$_h$ werden also ignoriert, **3210**$_h$:3 und **3210**$_h$:4 haben keinen Einfluss auf die Regelung.

BLDC

Wert = "1": Motortyp "BLDC" (Bürstenloser Gleichstrommotor)

320Ah Motor Drive Sensor Display Open Loop

Funktion

Damit kann die Quelle für die Objekte **6044**$_h$ und **6064**$_h$ im Modus *Open Loop* geändert werden.

Objektbeschreibung

Index	320A$_h$
Objektname	Motor Drive Sensor Display Open Loop
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00$_h$
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	04$_h$

Subindex	01$_h$
Name	Commutation

Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Torque
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	Velocity
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	04_h
Name	Position
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Beschreibung

Folgende Subindizes haben eine Funktion:

- 01_h: Ungenutzt
- 02_h: Ungenutzt
- 03_h: Verändert die Quelle des Objekts **6044_h**:
  - Wert = "-1": der intern berechnete Sollwert wird in das Objekt **6044_h** eingetragen
  - Wert = "0": der Wert wird auf 0 gehalten
  - Wert = "1": der Encoder-Wert wird in das Objekt **6044_h** eingetragen
- 04_h: Verändert die Quelle des **6064_h**:
  - Wert = "-1": der intern berechnete Sollwert wird in das Objekt **6064_h** eingetragen
  - Wert = "0": der Wert wird auf 0 gehalten
  - Wert = "1": der Encoder-Wert wird in das Objekt **6064_h** eingetragen

320Bh Motor Drive Sensor Display Closed Loop

Funktion

Damit kann die Quelle für die Objekte **6044**$_h$ und **6064**$_h$ im Modus *Closed Loop* geändert werden.

Objektbeschreibung

Index	320B$_h$
Objektname	Motor Drive Sensor Display Closed Loop
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00$_h$
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	04$_h$

Subindex	01$_h$
Name	Commutation
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000$_h$

Subindex	02$_h$
Name	Torque
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000$_h$

Subindex	03$_h$
Name	Velocity
Datentyp	INTEGER32

Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	04_h
Name	Position
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Beschreibung

Folgende Subindizes haben eine Funktion:

- 01_h: Ungenutzt
- 02_h: Ungenutzt
- 03_h: Verändert die Quelle des Objekts **6044_h**:
  - Wert = "-1": der intern berechnete Sollwert wird in das Objekt **6044_h** eingetragen
  - Wert = "0": der Wert wird auf 0 gehalten
  - Wert = "1": der Encoder-Wert wird in das Objekt **6044_h** eingetragen
- 04_h: Verändert die Quelle des Objekts **6064_h**:
  - Wert = "-1": der intern berechnete Sollwert wird in das Objekt **6064_h** eingetragen
  - Wert = "0": der Wert wird auf 0 gehalten
  - Wert = "1": der Encoder-Wert wird in das Objekt **6064_h** eingetragen

3210h Motor Drive Parameter Set

Funktion

Beinhaltet die P- und I-Anteile der Strom-, Geschwindigkeits- und Positionsregler für *Open Loop* (nur Stromregler aktiviert) und *Closed Loop*.

Objektbeschreibung

Index	3210_h
Objektname	Motor Drive Parameter Set
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1626: Eintrag "Name" geändert von "S_P" auf "Position Loop, Proportional Gain (closed Loop)". Firmware Version FIR-v1626: Eintrag "Name" geändert von "S_I" auf "Position Loop, Integral Gain (closed Loop)". Firmware Version FIR-v1626: Eintrag "Name" geändert von "V_P" auf "Velocity Loop, Proportional Gain (closed Loop)".

	Firmware Version FIR-v1626: Eintrag "Name" geändert von "V_I" auf "Velocity Loop, Integral Gain (closed Loop)".
	Firmware Version FIR-v1626: Eintrag "Name" geändert von "Id_P" auf "Flux Current Loop, Proportional Gain (closed Loop)".
	Firmware Version FIR-v1626: Eintrag "Name" geändert von "Id_I" auf "Flux Current Loop, Integral Gain (closed Loop)".
	Firmware Version FIR-v1626: Eintrag "Name" geändert von "Iq_P" auf "Torque Current Loop, Proportional Gain (closed Loop)".
	Firmware Version FIR-v1626: Eintrag "Name" geändert von "Iq_I" auf "Torque Current Loop, Integral Gain (closed Loop)".
	Firmware Version FIR-v1626: Eintrag "Name" geändert von "I_P" auf "Torque Current Loop, Proportional Gain (dspDrive - Stepper Motor, Open Loop)".
	Firmware Version FIR-v1626: Eintrag "Name" geändert von "I_I" auf "Torque Current Loop, Integral Gain (dspDrive - Stepper Motor, Open Loop)".
	Firmware Version FIR-v1650-B472161: Eintrag "Name" geändert von "Torque Current Loop, Proportional Gain (dspDrive - Stepper Motor, Open Loop)" auf "Torque Current Loop, Proportional Gain (open Loop)".
	Firmware Version FIR-v1650-B472161: Eintrag "Name" geändert von "Torque Current Loop, Integral Gain (dspDrive - Stepper Motor, Open Loop)" auf "Torque Current Loop, Integral Gain (open Loop)".
	Firmware Version FIR-v1650-B472161: Eintrag "Datentyp" geändert von "INTEGER32" auf "UNSIGNED32".
	Firmware Version FIR-v1650-B472161: Eintrag "Data type" geändert von "INTEGER32" auf "UNSIGNED32".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$0A_h$

Subindex	01_h
Name	Position Loop, Proportional Gain (closed Loop)
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000800_h

Subindex	02_h
Name	Position Loop, Integral Gain (closed Loop)
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	Velocity Loop, Proportional Gain (closed Loop)
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$00002EE0_h$

Subindex	04_h
Name	Velocity Loop, Integral Gain (closed Loop)
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$0000001E_h$

Subindex	05_h
Name	Flux Current Loop, Proportional Gain (closed Loop)
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$000668A0_h$

Subindex	06_h
Name	Flux Current Loop, Integral Gain (closed Loop)
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$00002EE0_h$

Subindex	07_h
Name	Torque Current Loop, Proportional Gain (closed Loop)
Datentyp	UNSIGNED32

Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$000668A0_h$

Subindex	08_h
Name	Torque Current Loop, Integral Gain (closed Loop)
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$00002EE0_h$

Subindex	09_h
Name	Torque Current Loop, Proportional Gain (open Loop)
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$0003A980_h$

Subindex	$0A_h$
Name	Torque Current Loop, Integral Gain (open Loop)
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$0000AFC8_h$

Beschreibung

- Subindex 00_h: Anzahl der Einträge
- Subindex 01_h: Proportionalanteil des S-Reglers (Position)
- Subindex 02_h: Integralanteil des S-Reglers (Position)
- Subindex 03_h: Proportionalanteil des V-Reglers (Geschwindigkeit)
- Subindex 04_h: Integralanteil des V-Reglers (Geschwindigkeit)
- Subindex 05_h: (Closed Loop) Proportionalanteil des Stromreglers der feldbildenden Komponente
- Subindex 06_h: (Closed Loop) Integralanteil des Stromreglers der feldbildenden Komponente
- Subindex 07_h: (Closed Loop) Proportionalanteil des Stromreglers der momentbildenden Komponente
- Subindex 08_h: (Closed Loop) Integralanteil des Stromreglers der momentbildenden Komponente
- Subindex 09_h: (Open Loop) Proportionalteil des Stromreglers der feldbildenden Komponente
- Subindex $0A_h$: (Open Loop) Integralanteil des Stromreglers der feldbildenden Komponente

3212h Motor Drive Flags

Funktion

Mit diesem Objekt wird bestimmt, ob im Modus "switched on" der CiA 402 Statemachine die Ausgangsspannung für den Motor aktiv ist, oder nicht. Zudem kann die Richtung des Drehfeldes geändert werden.

Hinweis

Änderungen im Subindex 02 werden erst nach einem Neustart der Steueung aktiv. Das **Auto-Setup** muss danach erneut durchgeführt werden.

Objektbeschreibung

Index	3212_h
Objektname	Motor Drive Flags
Object Code	ARRAY
Datentyp	INTEGER8
Speicherbar	ja, Kategorie: Applikation
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1450
Änderungshistorie	Firmware Version FIR-v1512: Die Anzahl der Einträge haben sich geändert von 2 auf 3.

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	03_h

Subindex	01_h
Name	Enable Legacy Power Mode
Datentyp	INTEGER8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	02_h
Name	Override Field Inversion
Datentyp	INTEGER8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	03_h
Name	Do Not Touch Controller Settings
Datentyp	INTEGER8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Beschreibung

Für den Subindex 01_h gültige Werte:

- Wert = "0": Die Ausgangsspannung für den Motor (PWM) ist im Status "Switched On" der **CiA 402 Power State Machine** fest auf 50% eingestellt, es wird kein Haltemoment aufgebaut.
- Wert = "1": Die Ausgangsspannung für den Motor (PWM) ist im Status "Switched On" der **CiA 402 Power State Machine** über den Regler aktiv, es ist ein Haltemoment aufgebaut. Der Motor wird still gehalten.

Für den Subindex 02_h gültige Werte:

- Wert = "0": Default-Werte der Firmware benutzen
- Wert = "1": nicht Invertieren des Drehfeldes erzwingen (mathematisch positiv)
- Wert = "-1": Invertieren des Drehfeldes erzwingen (mathematisch negativ)

Für den Subindex 03_h gültige Werte:

- Wert = "0": **Auto-Setup** erkennt den Motortyp (Schrittmotor oder BLDC-Motor) und verwendet den entsprechenden vorkonfigurierten Parametersatz.
- Wert = "1": **Auto-Setup** mit den Werten für den Regler durchführen, die vor dem Auto-Setup im Objekt **3210_h** eingetragen wurden, die Werte in **3210_h** werden nicht geändert.

3220h Analog Inputs

Funktion

Zeigt die Momentanwerte der Analogeingänge in Digits an.

Durch Objekt **3221_h** kann der jeweilige Analogeingang als Strom- oder Spannungseingang konfiguriert werden.

Objektbeschreibung

Index	3220_h
Objektname	Analog Inputs
Object Code	ARRAY
Datentyp	INTEGER16

Speicherbar	nein
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Analogue Input 1
Datentyp	INTEGER16
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0000_h

Subindex	02_h
Name	Analogue Input 2
Datentyp	INTEGER16
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0000_h

Beschreibung

Formeln zum Umrechnen von [digits] in die jeweilige Einheit:

- Spannungseingang: x digits * 3,3 V / 1024 digits
- Stromeingang: x digits * 20 mA / 1024 digits

3221h Analogue Inputs Control

Funktion

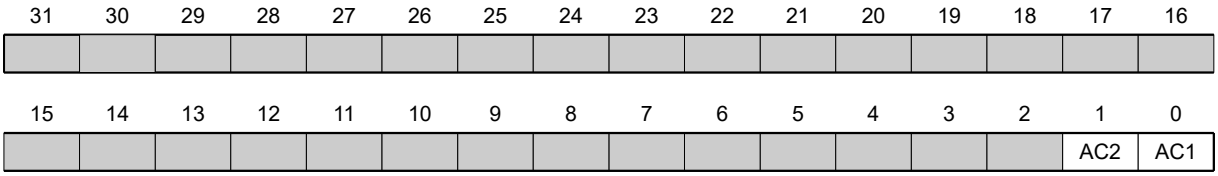
Mit diesem Objekt lässt sich ein Analog-Eingang von Spannungs- auf Strommessung umschalten.

Objektbeschreibung

Index	3221_h
Objektname	Analogue Inputs Control

Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung



Generell gilt: Wird ein Bit auf den Wert"0" gesetzt, misst der Analogeingang die Spannung, ist das Bit auf den Wert "1" gesetzt, wird der Strom gemessen.

AC1

Einstellung für Analogeingang 1

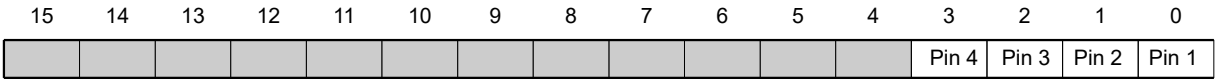
AC2

Einstellung für Analogeingang 2

3231h Flex IO Configuration

Funktion

Definiert wie die Pins (Ein- /Ausgänge 1 … 4) des Geräts belegt werden.



- Subindex 01_h *Output Mask*: Diese Bitmaske legt fest, ob der Pin als Eingang oder Ausgang verwendet wird:
  - Bit = "0:" Pin ist Eingang (Standard)
  - Bit = "1": Pin ist Ausgang
- Subindex 02_h *Pullup Mask*: Diese Bitmaske legt fest, ob der Pin ein *Pullup* oder *Pulldown* ist:
  - Bit = "0": Pin ist *Pulldown* (Standard)
  - Bit = "1": Pin ist *Pullup*

Tipp

Subindex 02_h ist für den Pin nur aktiv, wenn er über Subindex 01_h als Eingang definiert ist.

Beispiel für Subindex 01_h: Pin 2 und Pin 3 sollen Ausgänge sein, Wert ="6" (=0110_b)

Objektbeschreibung

Index	3231_h
Objektname	Flex IO Configuration
Object Code	ARRAY
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Applikation
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1650-B472161
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Output Mask
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h

Subindex	02_h
Name	Pullup Mask
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h

Beschreibung

3240h Digital Inputs Control

Funktion

Mit diesem Objekt lassen sich digitale Eingänge manipulieren wie in Kapitel **Digitale Ein- und Ausgänge** beschrieben.

Dabei gilt für alle Subindizes:

- Bit 0 bis 15 steuern die Spezialfunktionen.
- Bit 16 bis 31 steuern die Pegel der Ausgänge.

Objektbeschreibung

Index	3240_h
Objektname	Digital Inputs Control
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1426: Subindex 01_h: Eintrag "Name" geändert von "Special Function Disable" auf "Special Function Enable" Firmware Version FIR-v1512: Die Anzahl der Einträge haben sich geändert von 8 auf 9.

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	08_h

Subindex	01_h
Name	Special Function Enable
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Function Inverted
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben

PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	Force Enable
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	04_h
Name	Force Value
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	05_h
Name	Raw Value
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	06_h
Name	Input Range Select
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	07_h
Name	Differential Select
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	08_h
Name	Routing Enable
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Die Subindizes haben folgende Funktion:

- **3240**$_h$:01_h (Special Function Enable): Dieses Bit erlaubt Sonderfunktionen eines Eingangs aus- (Wert "0") oder einzuschalten (Wert "1"). Soll Eingang 1 z.B. nicht als negativer Endschalter verwendet werden, so muss die Sonderfunktion abgeschaltet werden, damit nicht fälschlicherweise auf den Signalgeber reagiert wird. Auf die Bits 16 bis 31 hat das Objekt keine Auswirkungen.
  Die Firmware wertet folgende Bits aus:
  - Bit 0: Negativer Endschalter
  - Bit 1: Positiver Endschalter
  - Bit 2: Referenzschalter

  Sollen z.B. zwei Endschalter und ein Referenzschalter verwendet werden, müssen Bits 0-2 in **3240**$_h$:01_h auf "1" gesetzt werden

- **3240**$_h$:02_h (Function Inverted): Dieses Bit wechselt von Schließer-Logik (ein logischer High-Pegel am Eingang ergibt den Wert "1" im Objekt **60FD**$_h$) auf Öffner-Logik (der logische High-Pegel am Eingang ergibt den Wert "0"). Das gilt für die Sonderfunktionen (außer den Takt- und Richtungseingängen) und für die normalen Eingänge.
  Hat das Bit den Wert "0" gilt Schließer-Logik, entsprechend bei dem Wert "1" die Öffner-Logik. Bit 0 entspricht dabei dem Eingang 1, Bit 1 dem Eingang 2 usw. .
- **3240**$_h$:03_h (Force Enable): Dieses Bit schaltet die Softwaresimulation von Eingangswerten ein, wenn es auf "1" gesetzt ist. Dann werden nicht mehr die tatsächlichen sondern die in Objekt **3240**$_h$:04_h eingestellten Werte für den jeweiligen Eingang verwendet.
- **3240**$_h$:04_h (Force Value): Dieses Bit gibt den Wert vor, der als Eingangswert eingelesen werden soll, wenn das gleiche Bit im Objekt **3240**$_h$:03_h gesetzt wurde.
- **3240**$_h$:05_h (Raw Value): Dieses Objekt beinhaltet den unmodifizierten Eingabewert.
- **60FD**$_h$ (Digital Inputs): Dieses Objekt enthält eine Zusammenfassung der Eingänge und den Spezialfunktionen.

3242h Digital Input Routing

Funktion

Dieses Objekt bestimmt die Quelle des Inputroutings, die im **60FD**$_h$ endet.

Objektbeschreibung

Index	3242_h
Objektname	Digital Input Routing
Object Code	ARRAY
Datentyp	UNSIGNED8
Speicherbar	ja, Kategorie: Applikation
Zugriff	nur lesen

PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1504
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	24_h

Subindex	01_h - 24_h
Name	Input Source #1 - #36
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00_h

Beschreibung

Der Subindex 01_h enthält die Quelle für das Bit 0 des Objekts **60FD**. Der Subindex 02_h enthält die Quelle für das Bit 1 des Objekts **60FD** und so weiter.

Die Nummer, die in eine Subindex geschrieben wird, bestimmt die Quelle für das zugehörige Bit. Die folgende Tabelle listet alle möglichen Signalquellen auf.

Nummer		
dec	**hex**	**Signalquelle**
00	00	Signal ist immer 0
01	01	Physikalischer Eingang 1
02	02	Physikalischer Eingang 2
03	03	Physikalischer Eingang 3
04	04	Physikalischer Eingang 4
05	05	Physikalischer Eingang 5
06	06	Physikalischer Eingang 6
07	07	Physikalischer Eingang 7
08	08	Physikalischer Eingang 8
09	09	Physikalischer Eingang 9
10	0A	Physikalischer Eingang 10
11	0B	Physikalischer Eingang 11
12	0C	Physikalischer Eingang 12

Nummer		
dec	**hex**	**Signalquelle**
13	0D	Physikalischer Eingang 13
14	0E	Physikalischer Eingang 14
15	0F	Physikalischer Eingang 15
16	10	Physikalischer Eingang 16
65	41	Hall Eingang "U"
66	42	Hall Eingang "V"
67	43	Hall Eingang "W"
68	44	Encoder Eingang "A"
69	45	Encoder Eingang "B"
70	46	Encoder Eingang "Index"
71	47	USB Power Signal
72	48	Status "Ethernet aktiv"
73	49	DIP-Schalter 1
74	4A	DIP-Schalter 2
75	4B	DIP-Schalter 3
76	4C	DIP-Schalter 4
77	4D	DIP-Schalter 5
78	4E	DIP-Schalter 6
79	4F	DIP-Schalter 7
80	50	DIP-Schalter 8
128	80	Signal ist immer 1
129	81	Invertierter physikalischer Eingang 1
130	82	Invertierter physikalischer Eingang 2
131	83	Invertierter physikalischer Eingang 3
132	84	Invertierter physikalischer Eingang 4
133	85	Invertierter physikalischer Eingang 5
134	86	Invertierter physikalischer Eingang 6
135	87	Invertierter physikalischer Eingang 7
136	88	Invertierter physikalischer Eingang 8
137	89	Invertierter physikalischer Eingang 9
138	8A	Invertierter physikalischer Eingang 10
139	8B	Invertierter physikalischer Eingang 11
140	8C	Invertierter physikalischer Eingang 12
141	8D	Invertierter physikalischer Eingang 13
142	8E	Invertierter physikalischer Eingang 14
143	8F	Invertierter physikalischer Eingang 15
144	90	Invertierter physikalischer Eingang 16
193	C1	Invertierter Hall Eingang "U"
194	C2	Invertierter Hall Eingang "V"
195	C3	Invertierter Hall Eingang "W"
196	C4	Invertierter Encoder Eingang "A"
197	C5	Invertierter Encoder Eingang "B"
198	C6	Invertierter Encoder Eingang "Index"
199	C7	Invertiertes USB Power Signal
200	C8	Invertierter Status "Ethernet aktiv"

Nummer		
dec	**hex**	**Signalquelle**
201	C9	Invertierter DIP-Schalter 1
202	CA	Invertierter DIP-Schalter 2
203	CB	Invertierter DIP-Schalter 3
204	CC	Invertierter DIP-Schalter 4
205	CD	Invertierter DIP-Schalter 5
206	CE	Invertierter DIP-Schalter 6
207	CF	Invertierter DIP-Schalter 7
208	D0	Invertierter DIP-Schalter 8

3250h Digital Outputs Control

Funktion

Mit diesem Objekt lassen sich die digitalen Ausgänge steuern, wie in Kapitel " **Digitale Ein- und Ausgänge**" beschrieben.

Dabei gilt für alle Subindizes:

- Bit 0 bis 15 steuern die Spezialfunktionen.
- Bit 16 bis 31 steuern die Pegel der Ausgänge.

Objektbeschreibung

Index	3250_h
Objektname	Digital Outputs Control
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1426: Subindex 01_h: Eintrag "Name" geändert von "Special Function Disable" auf "Special Function Enable" Firmware Version FIR-v1446: Eintrag "Name" geändert von "Special Function Enable" auf "No Function". Firmware Version FIR-v1512: Die Anzahl der Einträge haben sich geändert von 6 auf 9.

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	08_h

Subindex	01_h
Name	No Function
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Function Inverted
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	Force Enable
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	04_h
Name	Force Value
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	05_h
Name	Raw Value
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	06_h
Name	Reserved1
Datentyp	UNSIGNED32

Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	07_h
Name	Reserved2
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	08_h
Name	Routing Enable
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Die Subindizes haben folgende Funktion:

- 01_h: Ohne Funktion.
- 02_h: Mit diesem Subindex wird die Logik invertiert (von Öffner-Logik auf Schließer-Logik).
- 03_h: Mit diesem Subindex wird der Ausgangswert erzwungen, wenn das Bit den Wert "1" hat. Der Pegel des Ausganges wird in Subindex 4_h festgelegt.
- 04_h: Mit diesem Subindex wird der am Ausgang anzulegende Pegel definiert. Der Wert "0" liefert am digitalen Ausgang einen logischen Low-Pegel, der Wert "1" entsprechend einen logischen High-Pegel.
- 05_h: In diesem dem Subindex wird die an die Ausgänge gelegte Bitkombination abgelegt.

3252h Digital Output Routing

Funktion

Dieses Objekt weist einem Ausgang eine Signalquelle zu, die mit dem **60FE**$_h$ kontrolliert werden kann.

Objektbeschreibung

Index	3252_h
Objektname	Digital Output Routing
Object Code	ARRAY
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Applikation
Zugriff	nur lesen

PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1650-B527540
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	05_h

Subindex	01_h
Name	Output Control #1
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	1080_h

Subindex	02_h
Name	Output Control #2
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0090_h

Subindex	03_h
Name	Output Control #3
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0091_h

Subindex	04_h
Name	Output Control #4
Datentyp	UNSIGNED16

Zugriff	lesen/schreiben
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0092_h

Subindex	05_h
Name	Output Control #5
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0093_h

3320h Read Analogue Input

Funktion

Zeigt die Momentanwerte der Analogeingänge in benutzerdefinierten Einheiten an.

Objektbeschreibung

Index	3320_h
Objektname	Read Analogue Input
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	nein
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Number Of Analogue Inputs
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Analogue Input 1
Datentyp	INTEGER32
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	

Vorgabewert	00000000_h

Subindex	02_h
Name	Analogue Input 2
Datentyp	INTEGER32
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Die benutzerdefinierten Einheiten setzten sich aus Offset (**3321_h**) und Pre-scaling Wert (**3322_h**) zusammen. Sind beide Objekteinträge noch mit Default-Werten beschrieben, wird der Wert in **3320_h** in der Einheit "ADC digits" angegeben.

Formel zum Umrechnen von digits in die jeweilige Einheit:

- Spannungseingang: x digits * 3,3 V / 1024 digits
- Stromeingang: x digits * 20 mA / 1024 digits

Für die Subeinträge gilt:

- Subindex 00_h: Anzahl der Analogeingänge
- Subindex 01_h: Analogwert 1
- Subindex 02_h: Analogwert 2

3321h Analogue Input Offset

Funktion

Offset, der zum eingelesenen Analogwert (**3320_h**) addiert wird, bevor die Teilung mit dem Teiler aus dem Objekt **3322_h** vorgenommen wird.

Objektbeschreibung

Index	3321_h
Objektname	Analogue Input Offset
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Number Of Analogue Inputs
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein

Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Analogue Input 1
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Analogue Input 2
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

- Subindex 00_h: Anzahl der Offsets
- Subindex 01_h: Offset für Analogeingang 1
- Subindex 02_h: Offset für Analogeingang 2

3322h Analogue Input Pre-scaling

Funktion

Wert, mit dem der eingelesene Analogwert (**3320_h**, **3321_h**) dividiert wird, bevor er in das Objekt **3320_h** geschrieben wird.

Objektbeschreibung

Index	3322_h
Objektname	Analogue Input Pre-scaling
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Number Of Analogue Inputs

Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Analogue Input 1
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	alle Werte zulässig außer 0
Vorgabewert	00000001_h

Subindex	02_h
Name	Analogue Input 2
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	alle Werte zulässig außer 0
Vorgabewert	00000001_h

Beschreibung

Die Subindizes enthalten:

- Subindex 00_h: Anzahl der Teiler
- Subindex 01_h: Teiler für Analogeingang 1
- Subindex 02_h: Teiler für Analogeingang 2

3400h NanoSPI Comm Rx PDO Assignment

Funktion

Weist die RX-PDO Ziele des NanoSPI Comm-Busses zu. Siehe Kapitel **Abbild**.

Objektbeschreibung

Index	3400_h
Objektname	NanoSPI Comm Rx PDO Assignment
Object Code	ARRAY
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Kommunikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1650-B527540

Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Object Name" geändert von "SPI COMM RX PDO Assignment" auf "NanoSPI Comm Rx PDO Assignment". Firmware Version FIR-v1540: Eintrag "Saveable" geändert von "ja, Kategorie: Applikation" auf "ja, Kategorie: Kommunikation".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	SPI COMM PDO Mapping Index #1
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	1600_h

Subindex	02_h
Name	SPI COMM PDO Mapping Index #2
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	1601_h

Subindex	03_h
Name	SPI COMM PDO Mapping Index #3
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h

Subindex	04_h
Name	SPI COMM PDO Mapping Index #4
Datentyp	UNSIGNED16

Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h

3401h NanoSPI Comm Tx PDO Assignment

Funktion

Weist die TX PDO Ziele des NanoSPI Comm-Busses zu. Siehe Kapitel **Abbild**.

Objektbeschreibung

Index	3401_h
Objektname	NanoSPI Comm Tx PDO Assignment
Object Code	ARRAY
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Kommunikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1650-B527540
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Object Name" geändert von "SPI COMM TX PDO Assignment" auf "NanoSPI Comm Tx PDO Assignment". Firmware Version FIR-v1540: Eintrag "Saveable" geändert von "ja, Kategorie: Applikation" auf "ja, Kategorie: Kommunikation".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	SPI COMM PDO Mapping Index #1
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$1A00_h$

Subindex	02_h
Name	SPI COMM PDO Mapping Index #2
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$1A01_h$

Subindex	03_h
Name	SPI COMM PDO Mapping Index #3
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h

Subindex	04_h
Name	SPI COMM PDO Mapping Index #4
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h

3402h NanoSPI Ctrl Rx PDO Assignment

Funktion

Weist die RX PDO Ziele des NanoSPI Ctrl-Busses (SLOT-SPI) zu. Siehe Kapitel **Abbild** und **RX Mapping des Masters**.

Objektbeschreibung

Index	3402_h
Objektname	NanoSPI Ctrl Rx PDO Assignment
Object Code	ARRAY
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Kommunikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1650-B527540
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Object Name" geändert von "SPI CTRL RX PDO Assignment" auf "NanoSPI Ctrl Rx PDO Assignment".

	Firmware Version FIR-v1540: Eintrag "Saveable" geändert von "ja, Kategorie: Applikation" auf "ja, Kategorie: Kommunikation".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	SPI CTRL PDO Mapping Index #1
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	1600_h

Subindex	02_h
Name	SPI CTRL PDO Mapping Index #2
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	1601_h

Subindex	03_h
Name	SPI CTRL PDO Mapping Index #3
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h

Subindex	04_h
Name	SPI CTRL PDO Mapping Index #4
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	0000_h

3403h NanoSPI Ctrl Tx PDO Assignment

Funktion

Weist die TX PDO Ziele des NanoSPI Ctrl-Busses (SLOT-SPI) zu. Siehe Kapitel **Abbild** und **TX Mapping des Masters**.

Objektbeschreibung

Index	3403_h
Objektname	NanoSPI Ctrl Tx PDO Assignment
Object Code	ARRAY
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Kommunikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1650-B527540
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Object Name" geändert von "SPI CTRL TX PDO Assignment" auf "NanoSPI Ctrl Tx PDO Assignment". Firmware Version FIR-v1540: Eintrag "Saveable" geändert von "ja, Kategorie: Applikation" auf "ja, Kategorie: Kommunikation".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	SPI CTRL PDO Mapping Index #1
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$1A00_h$

Subindex	02_h
Name	SPI CTRL PDO Mapping Index #2
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$1A01_h$

Subindex	03_h
Name	SPI CTRL PDO Mapping Index #3
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h

Subindex	04_h
Name	SPI CTRL PDO Mapping Index #4
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h

340Fh NanoSPI Ctrl Statusword

Funktion

Statusword des SPI CTRL Busses.

Objektbeschreibung

Index	$340F_h$
Objektname	NanoSPI Ctrl Statusword
Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1540
Änderungshistorie	

3410h NanoSPI Comm Controlword

Funktion

Controlword des SPI Sub-Masters (siehe **SPI-Sub-Master**)

Objektbeschreibung

Index	3410_h
Objektname	NanoSPI Comm Controlword
Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Kommunikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Object Name" geändert von "SPI NanoSPI Comm Controlword" auf "NanoSPI Comm Controlword". Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Kommunikation".

Beschreibung

Der Sub-Master kann über das Controlword in verschiedene Zustände geschalten werden (siehe nachfolgende Tabelle). Der tatsächliche Status ist aus dem statusword **3411_h** zu entnehmen.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												M_OP	M_IN	M_FC	M_ON

M_ON (Switch Sub-Master to "ON")

- Wert = "1": Schaltet den Sub-Master ein
- Wert = "0": Schaltet den Sub-Master wieder ab, die Schnittstelle verhält sich dann wie ein Sub-Slave

M_FC (Sub-Master full control)

Wert = "1": Der Sub-Master schaltet sich in den Zustand "Init" und anschließend sofort in den Zustand "Operational". In diesem Zustand wirkt sich eine Änderung der PDO-Konfiguration nicht aus..

M_IN (Switch Sub-Master to "INIT")

Wert = "1": Schaltet den Sub-Master in den Zustand "Init"

M_OP (Switch Sub-Master to "OPERATIONAL")

Wert = "1": Schaltet den Sub-Master in den Zustand "Operational". In diesem Zustand wirkt sich eine Änderung der PDO-Konfiguration nicht aus.

3411h NanoSPI Comm Statusword

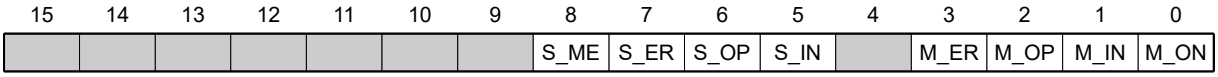
Funktion

In diesem Objekt ist das Statusword des Sub-Masters und des Sub-Slaves enthalten.

Objektbeschreibung

Index	3411_h
Objektname	NanoSPI Comm Statusword
Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Object Name" geändert von "SPI NanoSPI Comm Statusword" auf "NanoSPI Comm Statusword".

Beschreibung



M_ON (Sub-Master is "ON")
Wert = "1": Der Sub-Master ist eingeschaltet

M_IN (Sub-Master state "INIT")
Wert = "1": Der Sub-Master befindet sich im Status "Init".

M_OP (Sub-Master state "OPERATIONAL")
Wert = "1": Der Sub-Master befindet sich im Status "Operational".

M_ER (Sub-Master state "ERROR")
Wert = "1": Der Sub-Master befindet sich im Status "Error"

S_IN (Sub-Slave state "INIT")
Wert = "1": Der Sub-Slave befindet sich im Status "Init".

S_OP (Sub-Slave state "OPERATIONAL")
Wert = "1": Der Sub-Slave befindet sich im Status "Operational".

S_ER (Sub-Slave state "ERROR")
Wert = "1": Der Sub-Slave befindet sich im Status "Error".

3412h NanoSPI SDO Control

Funktion

Über das Controlword lässt sich eine in 3413h oder 3414h vorbereitete SDO-Nachricht vom Sub-Master an den Sub-Slave schicken. Siehe **SPI-Sub-Master**.

Objektbeschreibung

Index	3412_h
Objektname	NanoSPI SDO Control
Object Code	VARIABLE
Datentyp	UNSIGNED8
Speicherbar	ja, Kategorie: Kommunikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Object Name" geändert von "NanoSPI Can Master Controlword" auf "NanoSPI CAN Message Controlword". Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Kommunikation". Firmware Version FIR-v1650-B527540: Eintrag "Object Name" geändert von "NanoSPI CAN Message Controlword" auf "NanoSPI SDO Control".

Beschreibung

7	6	5	4	3	2	1	0
				ANS	MSG	RW	START

START

Wert = "1": Startet das Versenden der Nachricht

RW (Read or write)

Dieses Bit wird ignoriert, wenn Bit 2 (MSG) den Wert 1 enthält.

- Wert=0: Die SDO-Nachricht bewirkt ein Lesen aus dem Objektverzeichnis des Sub-Slave
- Wert=1: Die SDO-Nachricht schreibt den übermittelten Wert in das Objektverzeichnis des Sub-Slave

MSG (Message type)

- Wert=0: Die Daten aus dem Objekt **3413_h** werden versendet
- Wert=1: Die Daten aus dem Objekt **3414_h** werden versendet

ANS (Answer is ready)

Wert=1: Die Antwort zu der versendeten Nachricht ist angekommen (kann im **3415**$_h$ entnommen werden).

3413h NanoSPI SDO Request

Funktion

In dieses Objekt lassen sich Index, Subindex, Länge und Datenwert eintragen, welche vom Sub-Master an den Sub-Slave gesendet werden (siehe **NanoSPI-Mailbox**). Der Subindex 1 wird beim Versenden der Nachricht über **3412**$_h$ automatisch mit dem korrekten Wert beschrieben. Alternativ kann eine bereits fertig vorliegende Nachricht in das Objekt **3414**$_h$ eingetragen werden.

Objektbeschreibung

Index	3413_h
Objektname	NanoSPI SDO Request
Object Code	RECORD
Datentyp	SDO_EXPEDITED_MESSAGE
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Object Name" geändert von "NanoSPI Can Message Transmit" auf "NanoSPI CAN Message Transmit". Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Kommunikation". Firmware Version FIR-v1650-B527540: Eintrag "Object Name" geändert von "NanoSPI CAN Message Transmit" auf "NanoSPI SDO Request". Firmware Version FIR-v1650-B527540: Eintrag "Datentyp" geändert von "CAN_OPEN_MESSAGE" auf "SDO_EXPEDITED_MESSAGE". Firmware Version FIR-v1650-B527540: Tabellen-Eintrag "Zugriff" bei Subindex 00 geändert von "lesen/schreiben" auf "nur lesen". Firmware Version FIR-v1650-B527540: Eintrag "Name" geändert von "CAN Header" auf "SDO Header".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	05_h

Subindex	01_h

Name	SDO Header
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	02_h
Name	Index
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h

Subindex	03_h
Name	Subindex
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	04_h
Name	Length
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	05_h
Name	Data
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Wird der Wert aus dem Objektverzeichnis des Sub-Slaves gelesen, werden nur folgende Angaben benötigt (die Subindexe 4 und 5 bleiben unberücksichtigt):

- Index in 3413_h:2

- Subindex in 3413_h:3

Um einen Wert in das Objektverzeichnis des Slaves zu schreiben, werden folgende Angaben gebraucht:

- Index in 3413_h:2
- Subindex in 3413_h:3
- Länge des Objektes im Objektverzeichnis des Sub-Slaves in Byte in 3413_h:4
- Zu schreibender Wert in 3413_h:5

3414h NanoSPI SDO Raw Request

Funktion

In dieses Objekt können SDO-Nachrichten, welche vom Sub-Master an den Sub-Slave verschickt werden, direkt hinterlegt werden. Alternativ kann auch das Objekt **3413_h** benutzt werden.

Objektbeschreibung

Index	3414_h
Objektname	NanoSPI SDO Raw Request
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Object Name" geändert von "NanoSPI Can Message Raw" auf "NanoSPI CAN Message Raw". Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Kommunikation". Firmware Version FIR-v1650-B527540: Eintrag "Object Name" geändert von "NanoSPI CAN Message Raw" auf "NanoSPI SDO Raw Request". Firmware Version FIR-v1650-B527540: Eintrag "Name" geändert von "Can Raw Upper Bytes" auf "SDO Raw Request Upper Bytes". Firmware Version FIR-v1650-B527540: Eintrag "Name" geändert von "Can Raw Lower Bytes" auf "SDO Raw Request Lower Bytes".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	SDO Raw Request Upper Bytes

Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	SDO Raw Request Lower Bytes
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Der Subindex 1 des 3414 enthält die vier ersten Bytes einer SDO-Nachricht, der Subindex 2 entsprechend die vier letzten Bytes der SDO-Nachricht (üblicherweise die Daten eines Objekts).

Beispiel: Das Setzen des Wertes 6040_h:00 (Länge 2 Byte) auf den Wert "6" ergibt die SDO-Nachricht `2B 40 60 00 06 00 00 00`. Die ersten vier Byte werden dabei in dieses Objekt in den Subindex 1 geschrieben, die folgenden in den Subindex 2, also 3414_h:01 = $2B40600_h$ und 3414_h:02 = 000000006_h

3415h NanoSPI SDO Response

Funktion

Dieses Objekt enthält die Antwort auf eine vorher über **3414_h** gesendete Nachricht.

Objektbeschreibung

Index	3415_h
Objektname	NanoSPI SDO Response
Object Code	RECORD
Datentyp	SDO_EXPEDITED_MESSAGE
Speicherbar	nein
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Object Name" geändert von "SPI NanoSPI Can Message Receive" auf "NanoSPI CAN Message Receive". Firmware Version FIR-v1626: Tabellen-Eintrag "Zugriff" bei Subindex 00 geändert von "lesen/schreiben" auf "nur lesen". Firmware Version FIR-v1626: Tabellen-Eintrag "Zugriff" bei Subindex 01 geändert von "lesen/schreiben" auf "nur lesen". Firmware Version FIR-v1626: Tabellen-Eintrag "Zugriff" bei Subindex 02 geändert von "lesen/schreiben" auf "nur lesen". Firmware Version FIR-v1626: Tabellen-Eintrag "Zugriff" bei Subindex 03 geändert von "lesen/schreiben" auf "nur lesen".

	Firmware Version FIR-v1626: Tabellen-Eintrag "Zugriff" bei Subindex 04 geändert von "lesen/schreiben" auf "nur lesen". Firmware Version FIR-v1626: Tabellen-Eintrag "Zugriff" bei Subindex 05 geändert von "lesen/schreiben" auf "nur lesen". Firmware Version FIR-v1650-B527540: Eintrag "Object Name" geändert von "NanoSPI CAN Message Receive" auf "NanoSPI SDO Response". Firmware Version FIR-v1650-B527540: Eintrag "Datentyp" geändert von "CAN_OPEN_MESSAGE" auf "SDO_EXPEDITED_MESSAGE". Firmware Version FIR-v1650-B527540: Eintrag "Name" geändert von "CAN Header" auf "SDO Header".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	05_h

Subindex	01_h
Name	SDO Header
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	02_h
Name	Index
Datentyp	UNSIGNED16
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0000_h

Subindex	03_h
Name	Subindex
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	00_h

Subindex	04_h
Name	Length
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	05_h
Name	Data
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

3416h NanoSPI Slave Rx PDO Data

Funktion

Dieses Objekt dient dem Empfang von PDO-Daten, die der Sub-Slave sendet. Siehe **3400**$_h$

Objektbeschreibung

Index	3416_h
Objektname	NanoSPI Slave Rx PDO Data
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1540: Eintrag "Object Name" geändert von "SPI Slave Mapping PDO Received Data" auf "NanoSPI PDO Data Received From Slave". Firmware Version FIR-v1614: Die Anzahl der Einträge haben sich geändert von 11 auf 17. Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Kommunikation". Firmware Version FIR-v1650-B527540: Eintrag "Object Name" geändert von "NanoSPI PDO Data Received From Slave" auf "NanoSPI Slave Rx PDO Data".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	10_h

Subindex	01_h - 10_h
Name	Data #1 - #16
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

3417h NanoSPI Slave Tx PDO Data

Funktion

Dieses Objekt enthält Daten, die per PDO an den Sub-Slave gesendet werden sollen. Siehe **3401**$_h$.

Objektbeschreibung

Index	3417_h
Objektname	NanoSPI Slave Tx PDO Data
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Kommunikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1614: Die Anzahl der Einträge haben sich geändert von 11 auf 17. Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Kommunikation". Firmware Version FIR-v1650-B527540: Eintrag "Object Name" geändert von "NanoSPI PDO Data Transmitted To Slave" auf "NanoSPI Slave Tx PDO Data".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen

PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	10_h

Subindex	01_h - 10_h
Name	Data #1 - #16
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

3500h NanoSPI Rx PDO Mapping

Funktion

Dieses Objekt enthält die Mapping-Parameter für PDOs, die die Steuerung empfangen kann (RX-PDO). Siehe Kapitel **Abbild**.

Objektbeschreibung

Index	3500_h
Objektname	NanoSPI Rx PDO Mapping
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Kommunikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1650-B527540
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$0B_h$

Subindex	01_h
Name	Value #1

Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34160108_h

Subindex	02_h
Name	Value #2
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34160210_h

Subindex	03_h
Name	Value #3
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34160308_h

Subindex	04_h
Name	Value #4
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34160420_h

Subindex	05_h
Name	Value #5
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34160520_h

Subindex	06_h
Name	Value #6
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein

Zulässige Werte	
Vorgabewert	34160620_h

Subindex	07_h
Name	Value #7
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34160710_h

Subindex	08_h
Name	Value #8
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34160810_h

Subindex	09_h
Name	Value #9
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34160920_h

Subindex	$0A_h$
Name	Value #10
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$34160A20_h$

Subindex	$0B_h$
Name	Value #11
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$34160B10_h$

Subindex	$0C_h$
Name	Value #12
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	$0D_h$
Name	Value #13
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	$0E_h$
Name	Value #14
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

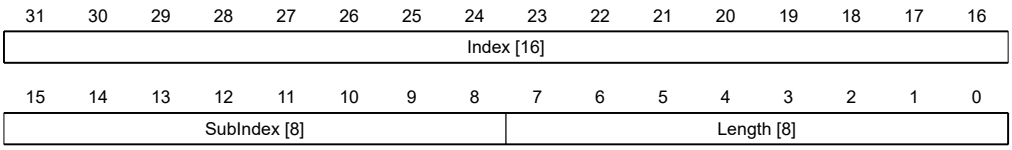
Subindex	$0F_h$
Name	Value #15
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	10_h
Name	Value #16
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Jeder Subindex beschreibt jeweils ein gemapptes Objekt.

Ein Mapping Eintrag besteht aus vier Bytes welche sich nach folgender Grafik zusammensetzen.



Index [16]

Darin ist der Index des zu mappenden Objektes enthalten.

Subindex [8]

Darin ist der Subindex des zu mappenden Objektes enthalten.

Length [8]

Darin ist die Länge des zu mappenden Objektes in der Einheit Bit enthalten.

3600h NanoSPI Tx PDO Mapping

Funktion

Dieses Objekt enthält die Mapping-Parameter für PDOs, die die Steuerung senden kann (TX-PDO). Siehe Kapitel **Abbild**.

Objektbeschreibung

Index	3600_h
Objektname	NanoSPI Tx PDO Mapping
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Kommunikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1650-B527540
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	07_h

Subindex	01_h
Name	Value #1
Datentyp	UNSIGNED32

Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34170108_h

Subindex	02_h
Name	Value #2
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34170210_h

Subindex	03_h
Name	Value #3
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34170320_h

Subindex	04_h
Name	Value #4
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34170410_h

Subindex	05_h
Name	Value #5
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34170520_h

Subindex	06_h
Name	Value #6
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	

Vorgabewert	34170610_h

Subindex	07_h
Name	Value #7
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	34170708_h

Subindex	08_h
Name	Value #8
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	09_h
Name	Value #9
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	$0A_h$
Name	Value #10
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	$0B_h$
Name	Value #11
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	$0C_h$

Name	Value #12
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	$0D_h$
Name	Value #13
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	$0E_h$
Name	Value #14
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

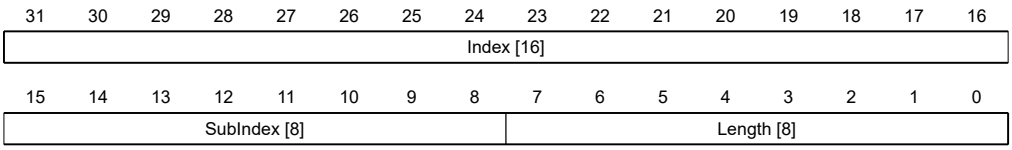
Subindex	$0F_h$
Name	Value #15
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	10_h
Name	Value #16
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Jeder Subindex beschreibt jeweils ein gemapptes Objekt.

Ein Mapping Eintrag besteht aus vier Bytes welche sich nach folgender Grafik zusammensetzen.



Index [16]

Darin ist der Index des zu mappenden Objektes enthalten.

Subindex [8]

Darin ist der Subindex des zu mappenden Objektes enthalten.

Length [8]

Darin ist die Länge des zu mappenden Objektes in der Einheit Bit enthalten.

3700h Following Error Option Code

Funktion

Das Objekt enthält die auszuführende Aktion, wenn ein Schleppfehler ausgelöst wird.

Objektbeschreibung

Index	3700_h
Objektname	Following Error Option Code
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$FFFF_h$
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Wert	Beschreibung
-32768 bis -2	Reserviert
-1	Keine Reaktion
0	Soforthalt
1	Abbremsen mit "slow down ramp" (Verzögerung (Bremsrampe) je nach Betriebsart)
2	Abbremsen mit "quick stop ramp" (Verzögerung (Bremsrampe) je nach Betriebsart)
3 bis 32767	Reserviert

4012h HW Information

Funktion

Dieses Objekt zeigt Informationen über die Hardware an.

Objektbeschreibung

Index	4012_h
Objektname	HW Information
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1540
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	01_h

Subindex	01_h
Name	EEPROM Size In Bytes
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Subindex 01: Zeigt die Größe des angeschlossenen EEPROMS in Bytes an. Der Wert "0" bedeutet, dass kein EEPROM angeschlossen ist.

4013h HW Configuration

Funktion

Mit diesem Objekt kann man bestimmte Hardware-Konfigurationen einstellen.

Objektbeschreibung

Index	4013_h
Objektname	HW Configuration
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1540
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	01_h

Subindex	01_h
Name	HW Configuration #1
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Bit 0 : reserviert

4014h Operating Conditions

Funktion

Dieses Objekt dient zum Auslesen aktueller Umgebungswerte der Steuerung.

Objektbeschreibung

Index	4014_h
Objektname	Operating Conditions
Object Code	ARRAY

Datentyp	INTEGER32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1540
Änderungshistorie	Firmware Version FIR-v1650-B472161: Tabellen-Eintrag "Zugriff" bei Subindex 01 geändert von "lesen/schreiben" auf "nur lesen". Firmware Version FIR-v1650-B472161: Tabellen-Eintrag "Zugriff" bei Subindex 02 geändert von "lesen/schreiben" auf "nur lesen". Firmware Version FIR-v1650-B472161: Eintrag "Name" geändert von "Temperature PCB [d?C]" auf "Temperature PCB [Celsius * 10]". Firmware Version FIR-v1650-B472161: Tabellen-Eintrag "Zugriff" bei Subindex 03 geändert von "lesen/schreiben" auf "nur lesen".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	03_h

Subindex	01_h
Name	Voltage UB Power [mV]
Datentyp	INTEGER32
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Voltage UB Logic [mV]
Datentyp	INTEGER32
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	03_h
Name	Temperature PCB [Celsius * 10]

Datentyp	INTEGER32
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Die Subindizes enthalten:

- 01_h: aktuelle Versorgungsspannung in [mV]
- 02_h: aktuelle Logikspannung in [mV]
- 03_h: aktuelle Temperatur in [d°C] (Zehntelgrad)

4040h Drive Serial Number

Funktion

Dieses Objekt hält die Seriennummer der Steuerung.

Objektbeschreibung

Index	4040_h
Objektname	Drive Serial Number
Object Code	VARIABLE
Datentyp	VISIBLE_STRING
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0
Firmware Version	FIR-v1450
Änderungshistorie	

4041h Device Id

Funktion

Dieses Objekt hält die ID des Geräts.

Objektbeschreibung

Index	4041_h
Objektname	Device Id
Object Code	VARIABLE
Datentyp	OCTET_STRING
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein

Zulässige Werte	
Vorgabewert	0
Firmware Version	FIR-v1540
Änderungshistorie	

Beschreibung

603Fh Error Code

Funktion

Dieses Objekt liefert den Error Code des letzten aufgetretenen Fehlers.

Er entspricht den unteren 16-Bits des Objekts **1003_h**. Für die Beschreibung der Error Codes schauen Sie unter Objekt **1003_h** nach.

Objektbeschreibung

Index	$603F_h$
Objektname	Error Code
Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Bedeutung des Fehlers siehe Objekt **1003_h** (Pre-defined Error Field).

6040h Controlword

Funktion

Dieses Objekt steuert die **CiA 402 Power State Machine**.

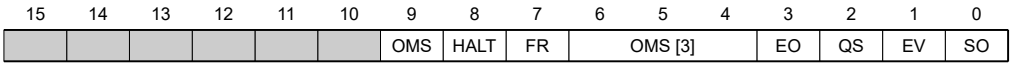
Objektbeschreibung

Index	6040_h
Objektname	Controlword
Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO

Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

Beschreibung

Teile des Objektes sind in der Funktion abhängig vom aktuell gewählten Modus.



SO (Switched On)
Wert = "1": Schaltet in den Zustand "Switched on"

EV (Enable Voltage)
Wert = "1": Schaltet in den Zustand "Enable voltage"

QS (Quick Stop)
Wert = "0": Schalten in den Zustand "Quick stop"

EO (Enable Operation)
Wert = "1": Schalten in den Zustand "Enable operation"

OMS (Operation Mode Specific)
Bedeutung abhängig vom gewählten Betriebsmodus

FR (Fault Reset)
Setzt einen Fehler zurück (falls möglich)

HALT
Wert = "1": Löst einen Halt aus, gültig in folgenden Modi:

- **Profile Position**
- **Velocity**
- **Profile Velocity**
- **Profile Torque**
- **Interpolated Position Mode**

6041h Statusword

Funktion

Dieses Objekt liefert Informationen zum Status der **CiA 402 Power State Machine**.

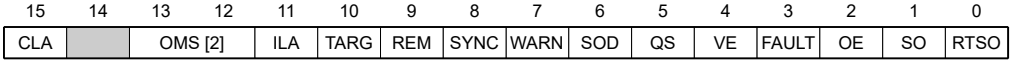
Objektbeschreibung

Index	6041_h
Objektname	Statusword
Object Code	VARIABLE
Datentyp	UNSIGNED16

Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Teile des Objektes sind in der Funktion abhängig vom aktuell gewählten Modus. Schlagen Sie im entsprechenden Unterkapitel im Kapitel **Betriebsmodi** nach.



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLA		OMS [2]		ILA	TARG	REM	SYNC	WARN	SOD	QS	VE	FAULT	OE	SO	RTSO

RTSO (Ready To Switch On)

Wert = "1": Steuerung befindet sich in dem Zustand "Ready To Switch On" (abhängig von anderen Bits, siehe nachfolgende Bitmaske)

SO (Switched On)

Wert = "1": Steuerung befindet sich in dem Zustand "Switched On" (abhängig von anderen Bits, siehe nachfolgende Bitmaske)

OE (Operation Enabled)

Wert = "1": Steuerung befindet sich in dem Zustand "Operation Enabled" (abhängig von anderen Bits, siehe nachfolgende Bitmaske)

FAULT

Fehler vorgefallen

VE (Voltage Enabled)

Spannung angelegt

QS (Quick Stop)

Wert = "0": Steuerung befindet sich in dem Zustand "Quick Stop" (abhängig von anderen Bits, siehe nachfolgende Bitmaske)

SOD (Switched On Disabled)

Wert = "1": Steuerung befindet sich in dem Zustand "Switched On Disabled" (abhängig von anderen Bits, siehe nachfolgende Bitmaske)

WARN (Warning)

Wert = "1": Warnung

SYNC (Synchronisation)

Wert = "1": Steuerung ist synchron zum Feldbus, Wert = "0": Steuerung ist nicht synchron zum Feldbus

REM (Remote)

Remote (Wert des Bits immer "1")

TARG

Zielvorgabe erreicht

ILA (Internal Limit Reached)

Limit überschritten

OMS (Operation Mode Specific)

Bedeutung abhängig vom gewählten Betriebsmodus

CLA (Closed Loop Available)

Wert = "1": Auto-Setup war erfolgreich und Encoder-Index gesehen: Closed Loop-Betrieb möglich

In der nachfolgenden Tabelle sind die Bitmasken aufgelistet, die den Zustand der Steuerung aufschlüsseln.

Statusword (6041_h)	Zustand
xxxx xxxx x0xx 0000	Not ready to switch on
xxxx xxxx x1xx 0000	Switch on disabled
xxxx xxxx x01x 0001	Ready to switch on
xxxx xxxx x01x 0011	Switched on
xxxx xxxx x01x 0111	Operation enabled
xxxx xxxx x00x 0111	Quick stop active
xxxx xxxx x0xx 1111	Fault reaction active
xxxx xxxx x0xx 1000	Fault

6042h Vl Target Velocity

Funktion

Gibt die Zielgeschwindigkeit in **benutzerdefinierten Einheiten** an.

Objektbeschreibung

Index	6042_h
Objektname	Vl Target Velocity
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$00C8_h$
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

6043h Vl Velocity Demand

Funktion

Gibt die aktuelle Zielgeschwindigkeit in Benutzereinheiten an.

Objektbeschreibung

Index	6043_h
Objektname	Vl Velocity Demand
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1426
Änderungshistorie	

6044h Vl Velocity Actual Value

Funktion

Gibt die aktuelle Istgeschwindigkeit in **benutzerdefinierten Einheiten** an.

Die Quelle dieses Objekts kann im *Open Loop*-Modus mit dem Objekt **320A**$_h$:03_h entweder auf den internen, berechneten Wert oder auf den Encoder gestellt werden.

Objektbeschreibung

Index	6044_h
Objektname	Vl Velocity Actual Value
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1426
Änderungshistorie	

6046h Vl Velocity Min Max Amount

Funktion

Mit diesem Objekt können Minimalgeschwindigkeit und Maximalgeschwindigkeit in **benutzerdefinierten Einheiten** eingestellt werden.

Objektbeschreibung

Index	6046_h
Objektname	Vl Velocity Min Max Amount
Object Code	ARRAY

Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	MinAmount
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	MaxAmount
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$00004E20_h$

Beschreibung

Subindex 1 enthält die Minimalgeschwindigkeit.

Subindex 2 enthält die Maximalgeschwindigkeit.

Wird eine Zielgeschwindigkeit (Objekt **6042_h**) vom Betrag her kleiner als die Minimalgeschwindigkeit angegeben, gilt die Minimalgeschwindigkeit und das Bit 11 (Internal Limit Reached) in **6041h Statusword$_h$** wird gesetzt.

Eine Zielgeschwindigkeit größer als die Maximalgeschwindigkeit setzt die Geschwindigkeit auf die Maximalgeschwindigkeit und das Bit 11 (Internal Limit Reached) in **6041h Statusword$_h$** wird gesetzt.

6048h Vl Velocity Acceleration

Funktion

Setzt die Beschleunigungsrampe im Velocity Mode (siehe **Velocity**).

Objektbeschreibung

Index	6048_h
Objektname	Vl Velocity Acceleration
Object Code	RECORD
Datentyp	VELOCITY_ACCELERATION_DECELERATION
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	DeltaSpeed
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$000001F4_h$

Subindex	02_h
Name	DeltaTime
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0001_h

Beschreibung

Die Beschleunigung wird als Bruch in **benutzerdefinierten Einheiten** angegeben:

Geschwindigkeitsänderung pro Zeitänderung.

Subindex 01_h: enthält die Geschwindigkeitsänderung.

Subindex 02_h: enthält die Zeitänderung.

6049h VI Velocity Deceleration

Funktion

Setzt die Verzögerung (Bremsrampe) im Velocity Mode (siehe **Velocity**).

Objektbeschreibung

Index	6049_h
Objektname	VI Velocity Deceleration
Object Code	RECORD
Datentyp	VELOCITY_ACCELERATION_DECELERATION
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	DeltaSpeed
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$000001F4_h$

Subindex	02_h
Name	DeltaTime
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0001_h

Beschreibung

Die Verzögerung wird als Bruch in **benutzerdefinierten Einheiten** angegeben:

Geschwindigkeitsänderung pro Zeitänderung.

Subindex 01_h: enthält die Geschwindigkeitsänderung.

Subindex 02_h: enthält die Zeitänderung.

604Ah Vl Velocity Quick Stop

Funktion

Dieses Objekt definiert die Verzögerung (Bremsrampe), wenn im **Velocity Mode** der Quick Stop-Zustand eingeleitet wird.

Objektbeschreibung

Index	$604A_h$
Objektname	Vl Velocity Quick Stop
Object Code	RECORD
Datentyp	VELOCITY_ACCELERATION_DECELERATION
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	DeltaSpeed
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00001388_h

Subindex	02_h
Name	DeltaTime
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben

PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0001_h

Beschreibung

Die Verzögerung wird als Bruch in **benutzerdefinierten Einheiten** angegeben:

Geschwindigkeitsänderung pro Zeitänderung.

Subindex 01_h: enthält die Geschwindigkeitsänderung.

Subindex 02_h: enthält die Zeitänderung.

604Ch Vl Dimension Factor

Funktion

Hier wird die Einheit der Geschwindigkeitsangaben für die Objekte festgelegt, welche den **Velocity Mode** betreffen.

Objektbeschreibung

Index	$604C_h$
Objektname	Vl Dimension Factor
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Vl Dimension Factor Numerator
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	02_h
Name	VI Dimension Factor Denominator
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$0000003C_h$

Beschreibung

Wird Subindex 1 auf den Wert "1" und Subindex 2 auf den Wert "1" eingestellt, erfolgt die Geschwindigkeitsangabe in Umdrehungen pro Minute.

Sonst enthält der Subindex 1 den Nenner (Multiplikator) und der Subindex 2 den Zähler (Divisor), mit dem interne Geschwindigkeitsangaben in Umdrehungen pro Sekunde verrechnet werden. Wird Subindex 1 auf den Wert "1" und Subindex 2 auf den Wert "60" eingestellt (Werkseinstellung), erfolgt die Geschwindigkeitsangabe in Umdrehungen pro Minute (1 Umdrehung pro 60 Sekunden).

605Ah Quick Stop Option Code

Funktion

Das Objekt enthält die auszuführende Aktion bei einem Übergang der **CiA 402 Power State Machine** in den Quick Stop-Zustand.

Objektbeschreibung

Index	$605A_h$
Objektname	Quick Stop Option Code
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0001_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Wert	Beschreibung
-32768 bis -1	Reserviert
0	Soforthalt
1	Abbremsen mit "slow down ramp" (Verzögerung (Bremsrampe) je nach Betriebsart) und anschließendem Zustandswechsel zu "Switch on disabled"
2	Abbremsen mit "quick stop ramp" und anschließendem Zustandswechsel zu "Switch on disabled"
3 bis 32767	Reserviert

605Bh Shutdown Option Code

Funktion

Das Objekt enthält die auszuführende Aktion bei einem Übergang der **CiA 402 Power State Machine** vom Zustand *Operation enabled* in den Zustand *Ready to switch on*.

Objektbeschreibung

Index	$605B_h$
Objektname	Shutdown Option Code
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0001_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Wert	Beschreibung
-32768 bis -1	Reserviert
0	Soforthalt
1	Abbremsen mit "slow down ramp" (Verzögerung (Bremsrampe) je nach Betriebsart) und anschließendem Zustandswechsel zu "Switch on disabled"
2 bis 32767	Reserviert

605Ch Disable Option Code

Funktion

Das Objekt enthält die auszuführende Aktion bei einem Übergang der **CiA 402 Power State Machine** vom Zustand "Operation enabled" in den Zustand "Switched on".

Objektbeschreibung

Index	$605C_h$
Objektname	Disable Option Code
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0001_h

Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Wert	Beschreibung
-32768 bis -1	Reserviert
0	Soforthalt
1	Abbremsen mit "slow down ramp" (Verzögerung (Bremsrampe) je nach Betriebsart) und anschließendem Zustandswechsel zu "Switch on disabled"
2 bis 32767	Reserviert

605Dh Halt Option Code

Funktion

Das Objekt enthält die auszuführende Aktion, wenn im Controlword **6040**$_h$ das Bit 8 (Halt) gesetzt wird.

Objektbeschreibung

Index	605D$_h$
Objektname	Halt Option Code
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0001$_h$
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Wert	Beschreibung
-32768 bis 0	Reserviert
1	Abbremsen mit "slow down ramp" (Verzögerung (Bremsrampe) je nach Betriebsart)
2	Abbremsen mit "quick stop ramp" (Verzögerung (Bremsrampe) je nach Betriebsart)
3 bis 32767	Reserviert

605Eh Fault Option Code

Funktion

Das Objekt enthält die auszuführende Aktion, wie der Motor im Fehlerfall zum Stillstand gebracht werden soll.

Objektbeschreibung

Index	$605E_h$
Objektname	Fault Option Code
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0002_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Wert	Beschreibung
-32768 bis -1	Reserviert
0	Soforthalt
1	Abbremsen mit "slow down ramp" (Verzögerung (Bremsrampe) je nach Betriebsart)
2	Abbremsen mit "quick stop ramp" (Verzögerung (Bremsrampe) je nach Betriebsart)
3 bis 32767	Reserviert

6060h Modes Of Operation

Funktion

In dieses Objekt wird der gewünschte Betriebsmodus eingetragen.

Objektbeschreibung

Index	6060_h
Objektname	Modes Of Operation
Object Code	VARIABLE
Datentyp	INTEGER8
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	

Vorgabewert	00_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

Beschreibung

Modus	Beschreibung
-2	Auto-Setup
-1	Takt-Richtungsmodus
0	No mode change/no mode assigned
1	Profile Position Mode
2	Velocity Mode
3	Profile Velocity Mode
4	Profile Torque Mode
5	Reserved
6	Homing Mode
7	Interpolated Position Mode
8	Cyclic Synchronous Position Mode
9	Cyclic Synchronous Velocity Mode
10	Cyclic Synchronous Torque Mode

6061h Modes Of Operation Display

Funktion

Zeigt den aktuellen Betriebsmodus. Siehe auch **6060h Modes Of Operation**.

Objektbeschreibung

Index	6061_h
Objektname	Modes Of Operation Display
Object Code	VARIABLE
Datentyp	INTEGER8
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00_h
Firmware Version	FIR-v1426
Änderungshistorie	

6062h Position Demand Value

Funktion

Gibt die aktuelle Sollposition in **benutzerdefinierten Einheiten** an.

Objektbeschreibung

Index	6062_h
Objektname	Position Demand Value
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	

6063h Position Actual Internal Value

Funktion

Enthält die aktuelle Drehgeberposition in Inkrementen. Im Gegensatz zu den Objekten **6062_h** und **6064_h** wird dieser Wert nach einem **Homing** nicht auf "0" gesetzt.

Hinweis

Ist die Encoderauflösung im Objekt **2052_h** = `0`, sind die Zahlenwerte dieses Objekts ungültig.

Objektbeschreibung

Index	6063_h
Objektname	Position Actual Internal Value
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	

6064h Position Actual Value

Funktion

Enthält die aktuelle Istposition in **benutzerdefinierten Einheiten**.

Die Quelle dieses Objekts kann im *Open Loop*-Modus mit dem Objekt **320A**$_h$:04$_h$ entweder auf den internen, berechneten Wert oder auf den Encoder gestellt werden.

Hinweis

Ist die Encoderauflösung im Objekt **2052**$_h$) = 0, sind die Zahlenwerte dieses Objekts ungültig.

Objektbeschreibung

Index	6064$_h$
Objektname	Position Actual Value
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000$_h$
Firmware Version	FIR-v1426
Änderungshistorie	

6065h Following Error Window

Funktion

Definiert den maximal erlaubten **Schleppfehler** in **benutzerdefinierten Einheiten** symmetrisch zur **Sollposition**.

Objektbeschreibung

Index	6065$_h$
Objektname	Following Error Window
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000100$_h$
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1504: Eintrag "Savable" geändert von "nein" auf "ja, Kategorie: Applikation".

Beschreibung

Weicht die Istposition von der Sollposition so stark ab, dass der Wert dieses Objekts überschritten wird, wird das Bit 13 im Objekt **6041**$_h$ gesetzt. Die Abweichung muss länger andauern als die Zeit in dem Objekt **6066**$_h$.

Wird der Wert des "Following Error Window" auf "FFFFFFFF"$_h$ gesetzt, wird die Schleppfehler-Überwachung abgeschaltet.

In dem Objekt **3700**$_h$ kann eine Reaktion auf den Schleppfehler gesetzt werden. Wenn eine Reaktion definiert ist, wird auch ein Fehler im Objekt **1003**$_h$ eingetragen.

6066h Following Error Time Out

Funktion

Zeit in Millisekunden, bis ein zu großer Schleppfehler zu einer Fehlermeldung führt.

Objektbeschreibung

Index	6066$_h$
Objektname	Following Error Time Out
Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0064$_h$
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1504: Eintrag "Savable" geändert von "nein" auf "ja, Kategorie: Applikation".

Beschreibung

Weicht die Istposition von der Sollposition so stark ab, dass der Wert des Objekts **6065**$_h$ überschritten wird, wird das Bit 13 im Objekt **6041**$_h$ gesetzt. Die Abweichung muss länger als die Zeit in diesem Objekt anhalten.

In dem Objekt **3700**$_h$ kann eine Reaktion auf den Schleppfehler gesetzt werden. Wenn eine Reaktion definiert ist, wird auch ein Fehler im Objekt **1003**$_h$ eingetragen.

6067h Position Window

Funktion

Gibt relativ zur Zielposition einen symmetrischen Bereich an, innerhalb dessen das Ziel als erreicht gilt in den Modi **Profile Position** und **Interpolated Position Mode**.

Objektbeschreibung

Index	6067$_h$
Objektname	Position Window

Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$0000000A_h$
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1504: Eintrag "Savable" geändert von "nein" auf "ja, Kategorie: Applikation".

Beschreibung

Ist die Abweichung der Istposition zur Zielposition kleiner als der Wert dieses Objekts, wird das Bit 10 im Objekt **6041_h** gesetzt. Die Bedingung muss länger erfüllt sein als die im Objekt **6066_h** definierte Zeit.

Wird der Wert auf "FFFFFFFF"$_h$ gesetzt, wird die Überwachung abgeschaltet.

6068h Position Window Time

Funktion

Die Istposition muss sich für diese Zeit in Millisekunden innerhalb des "Position Window" (**6067_h**) befinden, damit die Zielposition als erreicht gilt in den Modi **Profile Position** und **Interpolated Position Mode**.

Objektbeschreibung

Index	6068_h
Objektname	Position Window Time
Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0064_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1504: Eintrag "Savable" geändert von "nein" auf "ja, Kategorie: Applikation".

Beschreibung

Ist die Abweichung der Istposition zur Zielposition kleiner als der Wert des Objekts **6067_h**, wird das Bit 10 im Objekt **6041_h** gesetzt. Die Bedingung muss länger erfüllt sein als die im Objekt **6066_h** definierte Zeit.

606Bh Velocity Demand Value

Funktion

Vorgabegeschwindigkeit in **benutzerdefinierten Einheiten** für den Regler im **Profile Velocity Mode**.

Objektbeschreibung

Index	$606B_h$
Objektname	Velocity Demand Value
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Dieses Objekt enthält die Ausgabe des Rampengenerators, die gleichzeitig der Vorgabewert für den Geschwindigkeitsregler ist.

606Ch Velocity Actual Value

Funktion

Aktuelle Istgeschwindigkeit in **benutzerdefinierten Einheiten**.

Objektbeschreibung

Index	$606C_h$
Objektname	Velocity Actual Value
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	

606Dh Velocity Window

Funktion

Gibt relativ zur Zielgeschwindigkeit einen symmetrischen Bereich an, innerhalb dessen das Ziel als erreicht gilt im Modus **Profile Velocity**.

Objektbeschreibung

Index	$606D_h$
Objektname	Velocity Window
Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$001E_h$
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1614: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

Beschreibung

Ist die Abweichung der Istgeschwindigkeit zur Sollgeschwindigkeit kleiner als der Wert dieses Objekts, wird das Bit 10 im Objekt **6041_h** gesetzt. Die Bedingung muss länger erfüllt sein als die im Objekt **6066_h** definierte Zeit (siehe auch **Statusword im Modus Profile Velocity**).

606Eh Velocity Window Time

Funktion

Die Istgeschwindigkeit muss sich für diese Zeit in Millisekunden innerhalb des "Velocity Window" (**$606D_h$**) befinden, damit das Ziel als erreicht gilt.

Objektbeschreibung

Index	$606E_h$
Objektname	Velocity Window Time
Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1614: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

Beschreibung

Beschreibung

Ist die Abweichung der Istgeschwindigkeit zur Sollgeschwindigkeit kleiner als der Wert des Objekts **606D$_h$**, wird das Bit 10 im Objekt **6041$_h$** gesetzt. Die Bedingung muss länger erfüllt sein als die im Objekt **6066** definierte Zeit (siehe auch **Statusword im Modus Profile Velocity**).

6071h Target Torque

Funktion

Dieses Objekt enthält das Zieldrehmoment für den **Profile Torque** und **Cyclic Synchronous Torque** Modus in Promille des Nenndrehmoments.

Objektbeschreibung

Index	6071$_h$
Objektname	Target Torque
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0000$_h$
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

Beschreibung

Dieses Objekt wird als Tausendstel des Drehmoments gerechnet, z.B. der Wert "500" bedeutet "50%" des Nenndrehmoments, "1100" ist äquivalent zu 110%. Das Nenndrehmoment entspricht dem Nennstrom im Objekt **203B$_h$**:01.

Das Zieldrehmoment kann das Spitzendrehmoment (proportional zum Spitzenstrom in **2031$_h$**) nicht übersteigen.

6072h Max Torque

Funktion

Das Objekt beschreibt das maximale Drehmoment für den **Profile Torque** und **Cyclic Synchronous Torque** Modus in Promille des Nenndrehmoments.

Objektbeschreibung

Index	6072$_h$
Objektname	Max Torque
Object Code	VARIABLE
Datentyp	UNSIGNED16

Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Dieses Objekt wird als Tausendstel des Drehmoments gerechnet, z.B. der Wert "500" bedeutet "50%" des Nenndrehmoments, "1100" ist äquivalent zu 110%. Das Nenndrehmoment entspricht dem Nennstrom im Objekt **$203B_h$**:01.

Das Zieldrehmoment kann das Spitzendrehmoment (proportional zum Spitzenstrom in **2031_h**) nicht übersteigen.

6074h Torque Demand

Funktion

Momentaner vom Rampengenerator geforderter Drehmomentsollwert in Promille des Nominaldrehmonents für den internen Regler.

Objektbeschreibung

Index	6074_h
Objektname	Torque Demand
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Dieses Objekt wird als Tausendstel des Drehmoments gerechnet, z.B. der Wert "500" bedeutet "50%" des Nenndrehmoments, "1100" ist äquivalent zu 110%. Das Nenndrehmoment entspricht dem Nennstrom im Objekt **$203B_h$**:01.

Das Zieldrehmoment kann das Spitzendrehmoment (proportional zum Spitzenstrom in **2031_h**) nicht übersteigen.

6077h Torque Actual Value

Funktion

Dieses Objekt zeigt den aktuellen Drehmomentwert in Promille des Nenndrehmoments für den internen Regler.

Objektbeschreibung

Index	6077_h
Objektname	Torque Actual Value
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1540
Änderungshistorie	

Beschreibung

Dieses Objekt wird als Tausendstel des Drehmoments gerechnet, z.B. der Wert "500" bedeutet "50%" des Nenndrehmoments, "1100" ist äquivalent zu 110%. Das Nenndrehmoment entspricht dem Nennstrom im Objekt **$203B_h$**:01.

Das Zieldrehmoment kann das Spitzendrehmoment (proportional zum Spitzenstrom in **2031_h**) nicht übersteigen.

607Ah Target Position

Funktion

Dieses Objekt gibt die Zielposition in **benutzerdefinierten Einheiten** für den **Profile Position** und **Cyclic Synchronous Position** Modus an.

Objektbeschreibung

Index	$607A_h$
Objektname	Target Position
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$00000FA0_h$
Firmware Version	FIR-v1426

Änderungshistorie	Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

607Bh Position Range Limit

Funktion

Enthält die Minimal- und Maximalposition in **benutzerdefinierten Einheiten**.

Objektbeschreibung

Index	$607B_h$
Objektname	Position Range Limit
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Min Position Range Limit
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Max Position Range Limit
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Wird dieser Bereich über- oder unterschritten, erfolgt ein Überlauf. Um diesen Überlauf zu verhindern, können im Objekt **607D$_h$** ("Software Position Limit") Grenzwerte für die Zielposition eingestellt werden.

607Ch Home Offset

Funktion

Gibt die Differenz zwischen Null-Position der Steuerung und dem Referenzpunkt der Maschine in **benutzerdefinierten Einheiten** an.

Objektbeschreibung

Index	607C$_h$
Objektname	Home Offset
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000$_h$
Firmware Version	FIR-v1426
Änderungshistorie	

607Dh Software Position Limit

Funktion

Legt die Grenzpositionen relativ zum Referenzpunkt der Applikation in **benutzerdefinierten Einheiten** fest.

Objektbeschreibung

Index	607D$_h$
Objektname	Software Position Limit
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00$_h$
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen

PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Min Position Limit
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Subindex	02_h
Name	Max Position Limit
Datentyp	INTEGER32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Die Zielposition und die Sollposition müssen innerhalb der hier gesetzten Grenzen liegen. Der Home Offset (**607C**$_h$) wird nicht berücksichtigt.

607Eh Polarity

Funktion

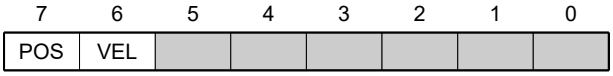
Mit diesem Objekt lässt sich die Drehrichtung umkehren.

Objektbeschreibung

Index	$607E_h$
Objektname	Polarity
Object Code	VARIABLE
Datentyp	UNSIGNED8
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Generell gilt für die Richtungsumkehr: Ist ein Bit auf den Wert "1" gesetzt, ist die Umkehrung aktiviert. Ist der Wert "0", ist die Drehrichtung wie im jeweiligen Modus beschrieben.



VEL (Velocity)

Umkehr der Drehrichtung in folgenden Modi:

- **Profile Velocity Mode**
- **Cyclic Synchronous Velocity Mode**
- **Velocity Mode**

POS (Position)

Umkehr der Drehrichtung in folgenden Modi:

- **Profile Position Mode**
- **Cyclic Synchronous Position Mode**

6081h Profile Velocity

Funktion

Gibt die maximale Fahrgeschwindigkeit in **benutzerdefinierten Einheiten** an.

Objektbeschreibung

Index	6081_h
Objektname	Profile Velocity
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$000001F4_h$
Firmware Version	FIR-v1426
Änderungshistorie	

6082h End Velocity

Funktion

Gibt die Geschwindigkeit am Ende der gefahrenen Rampe in **benutzerdefinierten Einheiten** an.

Objektbeschreibung

Index	6082_h
Objektname	End Velocity

Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	

6083h Profile Acceleration

Funktion

Gibt die maximale Beschleunigung in **benutzerdefinierten Einheiten** an.

Objektbeschreibung

Index	6083_h
Objektname	Profile Acceleration
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$000001F4_h$
Firmware Version	FIR-v1426
Änderungshistorie	

6084h Profile Deceleration

Funktion

Gibt die maximale Verzögerung (Bremsrampe) in **benutzerdefinierten Einheiten** an.

Objektbeschreibung

Index	6084_h
Objektname	Profile Deceleration
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$000001F4_h$
Firmware Version	FIR-v1426

Änderungshistorie	

6085h Quick Stop Deceleration

Funktion

Gibt die maximale Quick Stop-Verzögerung in **benutzerdefinierten Einheiten** an.

Objektbeschreibung

Index	6085_h
Objektname	Quick Stop Deceleration
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00001388_h
Firmware Version	FIR-v1426
Änderungshistorie	

6086h Motion Profile Type

Funktion

Gibt den Rampentyp für die Modi **Profile Position** und **Profile Velocity** an.

Objektbeschreibung

Index	6086_h
Objektname	Motion Profile Type
Object Code	VARIABLE
Datentyp	INTEGER16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0000_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Wert = "0": = Trapez-Rampe

Wert = "3": Rampe mit begrenztem Ruck

6087h Torque Slope

Funktion

Dieses Objekt enthält die Steigung des Drehmoments im Torque Mode.

Objektbeschreibung

Index	6087_h
Objektname	Torque Slope
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Dieses Objekt wird als Tausendstel des Drehmoments gerechnet, z.B. der Wert "500" bedeutet "50%" des Nenndrehmoments, "1100" ist äquivalent zu 110%. Das Nenndrehmoment entspricht dem Nennstrom im Objekt **$203B_h$**:01.

Das Zieldrehmoment kann das Spitzendrehmoment (proportional zum Spitzenstrom in **2031_h**) nicht übersteigen.

608Fh Position Encoder Resolution

Funktion

Virtuelle Encoder-Inkremente pro Umdrehung. Siehe Kapitel **Benutzerdefinierte Einheiten**.

Objektbeschreibung

Index	$608F_h$
Objektname	Position Encoder Resolution
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8

Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Encoder Increments
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$000007D0_h$

Subindex	02_h
Name	Motor Revolutions
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Beschreibung

Position Encoder Resolution = Encoder Increments (**608F**$_h$:01_h) / Motor Revolutions (**608F**$_h$:02_h)

6091h Gear Ratio

Funktion

Anzahl der Motorumdrehungen pro Umdrehung der Abtriebsachse.

Objektbeschreibung

Index	6091_h
Objektname	Gear Ratio
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8

Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Motor Revolutions
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	02_h
Name	Shaft Revolutions
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Beschreibung

Gear Ratio = Motor Revolutions (**6091**$_h$:01_h) / Shaft Revolutions (**6091**$_h$:02_h)

6092h Feed Constant

Funktion

Vorschub im Falle eines Linearantriebs, in **benutzerdefinierten Einheiten** pro Umdrehungen am Antrieb.

Objektbeschreibung

Index	6092_h
Objektname	Feed Constant
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported

Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Feed
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	02_h
Name	Shaft Revolutions
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000001_h

Beschreibung

Feed Constant = Feed (**6092_h**:01_h) / Shaft Revolutions (**6092_h**:02_h)

6098h Homing Method

Funktion

Dieses Objekt definiert die **Referenzfahrt-Methode** im **Homing** Mode.

Objektbeschreibung

Index	6098_h
Objektname	Homing Method
Object Code	VARIABLE
Datentyp	INTEGER8
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	23_h
Firmware Version	FIR-v1426
Änderungshistorie	

6099h Homing Speed

Funktion

Gibt die Geschwindigkeiten für den Homing Mode (**6098$_h$**) in **benutzerdefinierten Einheiten** an.

Objektbeschreibung

Index	6099$_h$
Objektname	Homing Speed
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00$_h$
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02$_h$

Subindex	01$_h$
Name	Speed During Search For Switch
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000032$_h$

Subindex	02$_h$
Name	Speed During Search For Zero
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0000000A$_h$

Beschreibung

Dieser Wert wird mit dem Zähler in Objekt **2061$_h$** und dem Nenner in Objekt **2062$_h$** verrechnet.

In Subindex 1 wird die Geschwindigkeit für die Suche nach dem Schalter angegeben.

In Subindex 2 wird die (niedrigere) Geschwindigkeit für die Suche nach der Referenzposition angegeben.

Hinweis

- Die Geschwindigkeit in Subindex 2 ist gleichzeitig die Anfangsgeschwindigkeit beim Start der Beschleunigungsrampe. Wird diese zu hoch eingestellt, verliert der Motor Schritte bzw. dreht sich überhaupt nicht. Eine zu hohe Einstellung führt dazu, dass die Indexmarkierung übersehen wird. Die Geschwindigkeit in Subindex 2 soll daher unter 1000 Schritten pro Sekunde sein.
- Die Geschwindigkeit in Subindex 1 muss größer sein als die Geschwindigkeit in Subindex 2.

609Ah Homing Acceleration

Funktion

Gibt die Beschleunigungsrampe für den Homing Mode in **benutzerdefinierten Einheiten** an.

Objektbeschreibung

Index	$609A_h$
Objektname	Homing Acceleration
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	$000001F4_h$
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Die Rampe wird nur beim Losfahren verwendet. Beim Erreichen des Schalters wird sofort auf die niedrigere Geschwindigkeit umgeschaltet und beim Erreichen der Endposition wird sofort gestoppt.

60A4h Profile Jerk

Funktion

Im Falle einer Rampe mit begrenztem Ruck kann in diesem Objekt die Größe des Rucks eingetragen werden. Ein Eintrag mit dem Wert "0" bedeutet, dass der Ruck nicht begrenzt ist.

Objektbeschreibung

Index	$60A4_h$
Objektname	Profile Jerk
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation

Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1614: Eintrag "Name" geändert von "End Acceleration Jerk" auf "Begin Deceleration Jerk". Firmware Version FIR-v1614: Eintrag "Name" geändert von "Begin Deceleration Jerk" auf "End Acceleration Jerk".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	04_h

Subindex	01_h
Name	Begin Acceleration Jerk
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$000003E8_h$

Subindex	02_h
Name	Begin Deceleration Jerk
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$000003E8_h$

Subindex	03_h
Name	End Acceleration Jerk
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$000003E8_h$

Subindex	04_h
Name	End Deceleration Jerk
Datentyp	UNSIGNED32

Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	$000003E8_h$

Beschreibung

- Subindex 01_h (*Begin Acceleration Jerk*): Anfangsruck bei Beschleunigung
- Subindex 02_h (*Begin Deceleration Jerk*): Anfangsruck bei Bremsung
- Subindex 03_h (*End Acceleration Jerk*): Abschlussruck bei Beschleunigung
- Subindex 04_h (*End Deceleration Jerk*): Abschlussruck bei Bremsung

60C1h Interpolation Data Record

Funktion

Dieses Objekt enthält die Sollposition in **benutzerdefinierten Einheiten** für den Interpolationsalgorithmus für den Betriebsmodus **Interpolated Position**.

Objektbeschreibung

Index	$60C1_h$
Objektname	Interpolation Data Record
Object Code	ARRAY
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1512
Änderungshistorie	Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	01_h

Subindex	01_h
Name	1st Set-point
Datentyp	INTEGER32

Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h

Beschreibung

Der Wert wird zum nächsten Synchronisationszeitpunkt übernommen.

60C2h Interpolation Time Period

Funktion

Dieses Objekt enthält die Interpolationszeit.

Objektbeschreibung

Index	$60C2_h$
Objektname	Interpolation Time Period
Object Code	RECORD
Datentyp	INTERPOLATION_TIME_PERIOD
Speicherbar	ja, Kategorie: Applikation
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1426
Änderungshistorie	

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	02_h

Subindex	01_h
Name	Interpolation Time Period Value
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	01_h

Subindex	02_h
Name	Interpolation Time Index
Datentyp	INTEGER8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	FD_h

Beschreibung

Die Subindizes haben folgende Funktionen:

- 01_h: Interpolationszeit.
- 02_h: Zehnerexponent der Interpolationszeit: muss den Wert -3 halten (entspricht der Zeitbasis in Millisekunden).

Es gilt dabei: Zykluszeit = Wert des **60C2**$_h$:01_h * 10 $^{\text{Wert des 60C2:02}}$ Sekunden.

60C4h Interpolation Data Configuration

Funktion

Dieses Objekt bietet die maximale Puffergröße, gibt die konfigurierte Puffer-Organisation der interpolierten Daten an und bietet Objekte zur Definition der Größe des Datensatzes und zum Löschen des Puffers. Es wird zudem verwendet, um die Position weiterer Datenpunkte zu speichern.

Objektbeschreibung

Index	$60C4_h$
Objektname	Interpolation Data Configuration
Object Code	RECORD
Datentyp	INTERPOLATION_DATA_CONFIGURATION
Speicherbar	ja, Kategorie: Applikation
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	
Firmware Version	FIR-v1512
Änderungshistorie	Firmware Version FIR-v1540: Tabellen-Eintrag "Zugriff" bei Subindex 05 geändert von "lesen/schreiben" auf "nur schreiben". Firmware Version FIR-v1540: Tabellen-Eintrag "Zugriff" bei Subindex 06 geändert von "lesen/schreiben" auf "nur schreiben". Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation". Firmware Version FIR-v1650-B472161: Tabellen-Eintrag "Zugriff" bei Subindex 01 geändert von "lesen/schreiben" auf "nur lesen".

Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	06_h

Subindex	01_h
Name	MaximumBufferSize
Datentyp	UNSIGNED32
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	02_h
Name	ActualBufferSize
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00000001_h

Subindex	03_h
Name	BufferOrganization
Datentyp	UNSIGNED8
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Subindex	04_h
Name	BufferPosition
Datentyp	UNSIGNED16
Zugriff	lesen/schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	0001_h

Subindex	05_h

Name	SizeOfDataRecord
Datentyp	UNSIGNED8
Zugriff	nur schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	04_h

Subindex	06_h
Name	BufferClear
Datentyp	UNSIGNED8
Zugriff	nur schreiben
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	00_h

Beschreibung

Der Wert des Subindex 01_h enthält die maximale mögliche Anzahl der interpolierten Datensätze.

Der Wert des Subindex 02_h enthält die momentane Anzahl der interpolierten Datensätze.

Wenn Subindex 03_h "00_h" ist, bedeutet das eine FIFO-Puffer-Organisation, wenn es "01_h" ist, gibt es eine Ring-Puffer-Organisation an.

Der Wert des Subindex 04_h ist ohne Einheit und gibt den nächsten freien Puffer-Einstiegspunkt an.

Der Wert des Subindex 05_h wird in der Einheit "Byte" angegeben. Wenn der Wert "00_h" in den Subindex 06_h geschrieben wird, löscht es die eingegangenen Daten im Puffer, deaktiviert den Zugriff und löscht alle Interpolierten Datensätze. Wenn der Wert "01_h" in den Subindex 06_h geschrieben wird, aktiviert es den Zugriff auf den Eingangs-Puffer.

60C5h Max Acceleration

Funktion

Dieses Objekt enthält die maximal zulässige Beschleunigung für den Modus **Profile Position** und **Profile Velocity**.

Objektbeschreibung

Index	$60C5_h$
Objektname	Max Acceleration
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00001388_h
Firmware Version	FIR-v1426
Änderungshistorie	

60C6h Max Deceleration

Funktion

Dieses Objekt enthält die maximal zulässige Verzögerung (Bremsrampe) für den Modus **Profile Position** und **Profile Velocity**.

Objektbeschreibung

Index	$60C6_h$
Objektname	Max Deceleration
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00001388_h
Firmware Version	FIR-v1426
Änderungshistorie	

60F2h Positioning Option Code

Funktion

Das Objekt beschreibt das Positionierverhalten im **Profile Position** Modus.

Objektbeschreibung

Index	$60F2_h$
Objektname	Positioning Option Code
Object Code	VARIABLE
Datentyp	UNSIGNED16
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	0001_h
Firmware Version	FIR-v1446
Änderungshistorie	Firmware Version FIR-v1614: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

Beschreibung

Derzeit werden nur nachfolgende Bits unterstützt:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MS	RESERVED [3]			IP OPTION [4]				RADO [2]		RRO [2]		CIO [2]		REL. OPT. [2]	

REL. OPT. (Relative Option)

Diese Bits bestimmen das Verhalten bei relativer Drehbewegung im "Profile Position" Modus, sollte Bit 6 des Kontrollwortes **6040**$_h$ = "1" gesetzt sein.

Bit 1	Bit 0	Definition
0	0	Positionsbewegungen werden relativ zu der vorherigen (intern absoluten) Zielposition ausgeführt (jeweils relativ zu 0 falls keine Zielpositon voran gegangen ist)
0	1	Positionsbewegungen werden relativ zum Vorgabewert (bzw. Ausgang) des Rampengenerators ausgeführt.
1	0	Positionsbewegungen werden relativ zur Istposition (Objekt **6064**$_h$) ausgeführt.
1	1	Reserviert

RRO (Request-Response Option)

Diese Bits bestimmen das Verhalten bei der Übergabe des Controlwords **6040**$_h$ Bit 5 ("new setpoint") - die Steuerung übernimmt in diesem Fall die Freigabe des Bits selbständig. Damit fällt die Notwendigkeit weg, das Bit anschließend extern wieder auf "0" zu setzen. Nachdem das Bit von der Steuerung aus auf den Wert "0" gesetzt wurde, wird auch das Bit 12 ("setpoint acknowledgement") im Statusword **6041**$_h$ auf den Wert "0" gesetzt.

Hinweis

Diese Optionen bringen die Steuerung dazu, das Objekt Controlword **6040**$_h$ zu modifizieren.

Bit 3	Bit 2	Definition
0	0	Die Funktionalität ist wie unter **Setzen von Fahrbefehlen** beschrieben.
0	1	Die Steuerung wird das Bit "new setpoint" frei geben, sobald die momentane Zielfahrt ihr Ziel erreicht hat.
1	0	Die Steuerung wird das Bit "new setpoint" frei geben, sobald es der Steuerung möglich ist.
1	1	Reserviert

RADO (Rotary Axis Direction Option)

Diese Bits bestimmen die Drehrichtung im "Profile Position" Modus.

Bit 7	Bit 6	Definition
0	0	Normale Positionierung ähnlich einer linearen Achse: Falls eines der "Position Range Limits" **607B**$_h$:01$_h$ und 02$_h$ erreicht oder überschritten wird, wird der Vorgabewert automatisch an das andere Ende der Limits übertragen. Nur mit dieses Bitkombination ist eine Bewegung größer als der Modulo-Wert möglich.
0	1	Positionierung nur in negativer Richtung: falls die Zielposition größer als die aktuelle Position ist fährt die Achse über das "Min Position Range Limit" aus Objekt **607D**$_h$:01$_h$ zu der Zielposition.

Bit 7	Bit 6	Definition
1	0	Positionierung nur in positiver Richtung: falls die Zielposition kleiner als die aktuelle Position ist fährt die Achse über das "Max Position Range Limit" aus Objekt **607D$_h$**:01$_h$ zu der Zielposition.
1	1	Positionierung mit dem kürzesten Weg zur Zielposition. Falls die Differenz zwischen aktueller Position und Zielposition in einem 360° System kleiner als 180° ist, fährt die Achse in positiver Richtung.

60F4h Following Error Actual Value

Funktion

Dieses Objekt enthält den aktuellen Schleppfehler in **benutzerdefinierten Einheiten**.

Objektbeschreibung

Index	60F4$_h$
Objektname	Following Error Actual Value
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000$_h$
Firmware Version	FIR-v1426
Änderungshistorie	

60FDh Digital Inputs

Funktion

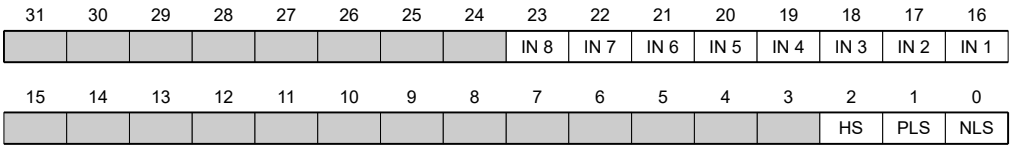
Mit diesem Objekt können die **Digitalen Eingänge** des Motors gelesen werden.

Objektbeschreibung

Index	60FD$_h$
Objektname	Digital Inputs
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	00000000$_h$
Firmware Version	FIR-v1426

Änderungshistorie	

Beschreibung



NLS (Negative Limit Switch)
negativer Endschalter

PLS (Positive Limit Switch)
positiver Endschalter

HS (Home Switch)
Referenzschalter

IN n (Input n)
Eingang n - die Anzahl der verwendeten Bits ist abhängig von der jeweiligen Steuerung.

60FEh Digital Outputs

Funktion

Mit diesem Objekt können die **Digitalausgänge** des Motors geschrieben werden.

Objektbeschreibung

Index	$60FE_h$
Objektname	Digital Outputs
Object Code	ARRAY
Datentyp	UNSIGNED32
Speicherbar	ja, Kategorie: Applikation
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

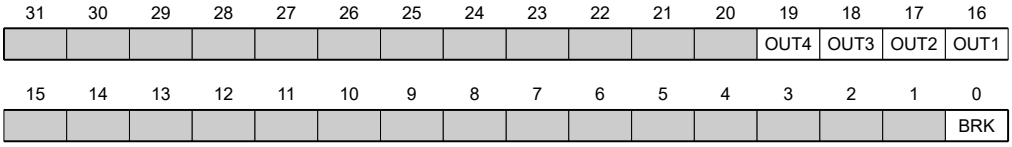
Wertebeschreibung

Subindex	00_h
Name	Highest Sub-index Supported
Datentyp	UNSIGNED8
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	01_h

Subindex	01_h
Name	Digital Outputs #1
Datentyp	UNSIGNED32
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000001_h

Beschreibung

Zum Schreiben der Ausgänge müssen noch die Einträge in Objekt **3250**$_h$, Subindex 02_h bis 05_h berücksichtigt werden.



BRK (Brake)

Bit für den Bremsenausgang (falls der Controller diese Funktion unterstützt).

OUT n (Output No n)

Bit für den jeweiligen digitalen Ausgang, die genaue Zahl der Digitalausgänge ist abhängig von der Steuerung.

60FFh Target Velocity

Funktion

In dieses Objekt wird die Zielgeschwindigkeit für den **Profile Velocity** und **Cyclic Synchronous Velocity**Mode in **benutzerdefinierten Einheiten** eingetragen.

Objektbeschreibung

Index	$60FF_h$
Objektname	Target Velocity
Object Code	VARIABLE
Datentyp	INTEGER32
Speicherbar	ja, Kategorie: Applikation
Zugriff	lesen/schreiben
PDO-Mapping	RX-PDO
Zulässige Werte	
Vorgabewert	00000000_h
Firmware Version	FIR-v1426
Änderungshistorie	Firmware Version FIR-v1626: Eintrag "Speicherbar" geändert von "nein" auf "ja, Kategorie: Applikation".

6502h Supported Drive Modes

Funktion

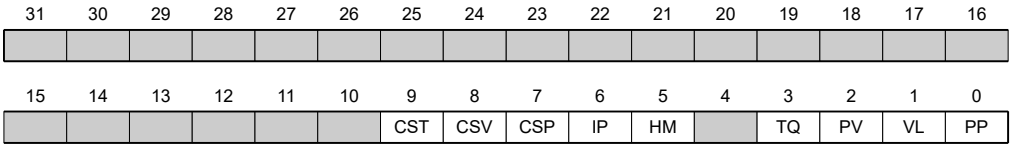
Das Objekt beschreibt die unterstützten Betriebsmodi im Objekt **6060_h**.

Objektbeschreibung

Index	6502_h
Objektname	Supported Drive Modes
Object Code	VARIABLE
Datentyp	UNSIGNED32
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	TX-PDO
Zulässige Werte	
Vorgabewert	$000003EF_h$
Firmware Version	FIR-v1426
Änderungshistorie	

Beschreibung

Ein gesetztes Bit gibt an, ob der jeweilige Modus unterstützt wird. Ist der Wert des Bits "0", wird der Modus nicht unterstützt.



PP

Profile Position Modus

VL

Velocity Modus

PV

Profile Velocity Modus

TQ

Torque Modus

HM

Homing Modus

IP

Interpolated Position Modus

CSP

Cyclic Synchronous Position Modus

CSV

Cyclic Synchronous Velocity Modus

CST

Cyclic Synchronous Torque Modus

6505h Http Drive Catalogue Address

Funktion

Dieses Objekt enthält die Web-Adresse des Herstellers als Zeichenkette.

Objektbeschreibung

Index	6505_h
Objektname	Http Drive Catalogue Address
Object Code	VARIABLE
Datentyp	VISIBLE_STRING
Speicherbar	nein
Zugriff	nur lesen
PDO-Mapping	nein
Zulässige Werte	
Vorgabewert	http://www.nanotec.de
Firmware Version	FIR-v1426
Änderungshistorie	

12 Copyrights

12.1 Einführung

In der Nanotec Software sind Komponenten aus Produkten externer Software-Hersteller integriert. In diesem Kapitel finden Sie die Copyright-Informationen zu den verwendeten externen Software-Quellen.

12.2 AES

FIPS-197 compliant AES implementation

Based on XySSL: Copyright (C) 2006-2008 Christophe Devine

Copyright (C) 2009 Paul Bakker <polarssl_maintainer at polarssl dot org>

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution; or, the application vendor's website must provide a copy of this notice.
- Neither the names of PolarSSL or XySSL nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The AES block cipher was designed by Vincent Rijmen and Joan Daemen.

http://csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf

http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

12.3 MD5

MD5C.C - RSA Data Security, Inc., MD5 message-digest algorithm

Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All rights reserved.

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

12.4 uIP

Copyright (c) 2005, Swedish Institute of Computer Science

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the Institute nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE INSTITUTE AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE INSTITUTE OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

12.5 DHCP

Copyright (c) 2005, Swedish Institute of Computer Science

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the Institute nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE INSTITUTE AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE INSTITUTE OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

12.6 CMSIS DSP Software Library

Copyright (C) 2010 ARM Limited. All rights reserved.

12.7 FatFs

FatFs - FAT file system module include file R0.08 (C)ChaN, 2010

FatFs module is a generic FAT file system module for small embedded systems.

This is a free software that opened for education, research and commercial

developments under license policy of following trems.

Copyright (C) 2010, ChaN, all right reserved.

The FatFs module is a free software and there is NO WARRANTY.

No restriction on use. You can use, modify and redistribute it for

personal, non-profit or commercial product UNDER YOUR RESPONSIBILITY.

Redistributions of source code must retain the above copyright notice.

12.8 Protothreads

Protothread class and macros for lightweight, stackless threads in C++.

This was "ported" to C++ from Adam Dunkels' protothreads C library at: http://www.sics.se/~adam/pt/

Originally ported for use by Hamilton Jet (www.hamiltonjet.co.nz) by Ben Hoyt, but stripped down for public release. See his blog entry about it for more information: http://blog.micropledge.com/2008/07/protothreads/

Original BSD-style license

Copyright (c) 2004-2005, Swedish Institute of Computer Science.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the Institute nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This software is provided by the Institute and contributors "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the Institute or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

12.9 lwIP

Copyright (c) 2001-2004 Swedish Institute of Computer Science.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This file is part of the lwIP TCP/IP stack.

Author: Adam Dunkels <adam@sics.se>