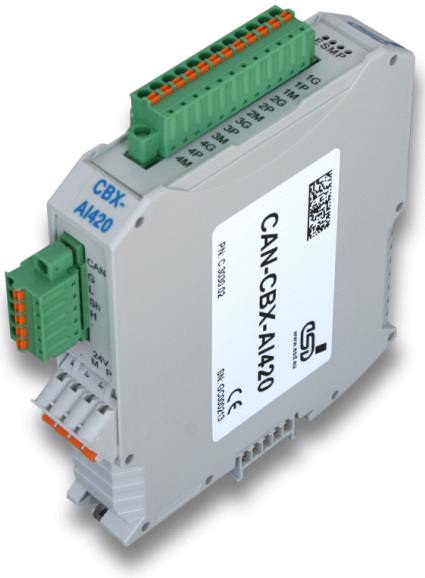
CAN-CBX-AI420

4 A/D-Wandler-Eingänge, 20 Bit



Handbuch

zu Artikel C.3030.02

esd electronic system design gmbh
Vahrenwalder Str. 207 • 30165 Hannover • Germany
http://www.esd.eu
Phone: +49 (0) 511 3 72 98-0 • Fax: +49 (0) 511 3 72 98-68

Hinweis

Der Inhalt dieses Handbuches wurde mit größter Sorgfalt erarbeitet und geprüft. **esd** übernimmt jedoch keine Verantwortung für Schäden, die aus Fehlern in der Dokumentation resultieren könnten. Insbesondere Beschreibungen und technische Daten sind keine zugesicherten Eigenschaften im rechtlichen Sinne.

esd hat das Recht, Änderungen am beschriebenen Produkt oder an der Dokumentation ohne vorherige Ankündigung vorzunehmen, wenn sie aus Gründen der Zuverlässigkeit oder Qualitätssicherung vorgenommen werden oder dem technischen Fortschritt dienen.

esd electronic system design gmbh
Vahrenwalder Str. 207
30165 Hannover

Tel.: 0511/372 98-0
FAX : 0511/372 98-68
E-Mail: info@esd.eu
Internet: www.esd.eu

Dokument-Datei:	I:\Texte\Doku\MANUALS\CAN\CBX\AI420\Deutsch\CAN-CBX-AI420_Handbuch_de_14.wpd
Datum des Ausdrucks:	2014-10-10

Platinenversion:	ab CAN-CBX-AI420 Rev. 1.0
Firmwareversion:	ab Rev. 1.8

Änderungen in den Kapiteln
Die hier aufgeführten Änderungen im Dokument betreffen sowohl Änderungen in der Hardware als auch reine Änderungen in der Beschreibung der Sachverhalte.

Version	Kapitel	Änderungen gegenüber Vorversion
1.4	-	Hinweise zu Sicherheit, Konformität etc. und Darstellungskonventionen eingefügt
	2.	Technische Daten überarbeitet
	3.1	Bild neu, Hinweis auf Leiteranschluss/Leiterquerschnitt eingefügt
	3.2	Beschreibung der LEDs aktualisiert
	3.3	Abbildung erneuert, Beschreibung der Kodierschalter und Baudraten überarbeitet
	4.	Ehemaliges Kapitel “Beschreibung der Baugruppen” verschoben in Unterkapitel des Kapitels “Steckerbelegungen”. Steckerbelegungen überarbeitet, Hinweise zu Leiteranschluss/Leiterquerschnitt neu
	5.	Kapitel verschoben und aktualisiert
	6.	Kapitel verschoben und aktualisiert
	7.	Kapitel “Software” umbenannt in “CANopen-Firmware”, Allgemeiner Teil (Kapitel 7.1 - 7.4) aktualisiert und umstrukturiert
	7.6	Parameter-Beschreibung eingefügt
	7.8	Kapitel zur Erklärung der Abkürzungen und Begriffe eingefügt
	7.9.1	Produktspezifische Eigenschaften CAN-CBX-AI420 in Tabelle übernommen, neue Objekte 1006_h, 1019_h, 1020_h, 1029_h, $1F80_h$, $1F91_h$
	7.9.2-7.9.21	Kapitel überarbeitet, Beschreibung der produktspezifischen Eigenschaften nur noch in Tabelle in Übersicht der 1000er-Objekte (Kapitel 7.9.1),
	7.9.22	Kapitel “NMT Startup ($1F80_h$)” neu
	7.9.23	Kapitel “Self Starting Nodes Timing Parameters ($1F91_h$)” neu
	7.10	Übersicht der Objekte
	7.10.5	Beschreibung Objekt 6421_h überarbeitet
	7.10.7, 7.10.8	Beschreibung der Objekte 6424_h, 6425_h eingefügt,
	7.10.9	Beschreibung Objekt 6426_h überarbeitet
	7.11.2, 7.11.4	Sample-Rate geändert in Sample-Time, Kapitel 7.11.2 überarbeitet
	7.11.8, 7.11.9	Beschreibung der Objekte 2404_h, 2405_h
	8.	Kapitel überarbeitet
	9.	EU-Konformitätserklärung aktualisiert
	10.	Kapitel “Bestellhinweise” verschoben und ergänzt

Weitere technische Änderungen vorbehalten.

Sicherheitshinweise

- Bitte beachten Sie im Umgang mit dem CAN-CBX-Modul die folgenden Sicherheitshinweise und lesen Sie dieses Handbuch aufmerksam durch, um Schäden am CAN-CBX-Modul und Verletzungen zu vermeiden.
- Das CAN-CBX-Modul darf nicht geöffnet werden.
- Das Gerät muss vor der Inbetriebnahme fest montiert sein.
- Lassen Sie keine Flüssigkeiten in das CAN-CBX-Modul eindringen, da sonst elektrische Schläge oder Kurzschlüsse die Folge sein können.
- Schützen Sie das CAN-CBX-Modul vor Feuchtigkeit und Dämpfen.
- Schützen Sie das CAN-CBX-Modul vor Stößen und Vibrationen.
- Das CAN-CBX-Modul wird möglicherweise während des normalen Betriebs warm. Achten Sie stets auf ausreichende Luftzufuhr, damit die Wärme abgeführt werden kann.
- Betreiben Sie das CAN-CBX-Modul nicht in unmittelbarer Nähe von Wärmequellen und setzen Sie es keiner unnötigen Wärmestrahlung aus. Die zulässige Umgebungstemperatur ist in den technischen Daten festgelegt.
- Verwenden Sie keine beschädigten Leitungen für den Anschluss des CAN-CBX-Moduls und beachten Sie die Verdrahtungshinweise zum CAN-Bus am Ende dieses Handbuchs.
- Bei Beschädigungen am Gerät, die die Sicherheit betreffen könnten, müssen unverzüglich geeignete Maßnahmen getroffen werden, die eine Gefährdung von Menschen und Haus- und Nutztieren sowie Gütern verhindern.
- Mit der Einrichtung verbundene Stromkreise müssen gegen gefährliche Spannungen ausreichend geschützt sein (SELV nach EN 60950-1).
- Das CAN-CBX-Modul darf nur an Versorgungsstromkreisen betrieben werden, die berührungssicher sind. Ein Netzteil, welches eine Schutzkleinspannung (SELV oder PELV) nach EN 60950-1 zur Verfügung stellt, erfüllt diese Bedingung.

Qualifiziertes Personal

Diese Dokumentation wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuer- und Automatisierungstechnik. Die Installation und Inbetriebnahme des Produkts darf nur von qualifiziertem Personal vorgenommen werden, das berechtigt ist, Geräte, Systeme und Stromkreise gemäß den Standards der Sicherheitstechnik in Betrieb zu nehmen.

Konformität

Das CAN-CBX-AI420 ist ein industrielles Produkt und erfüllt die in der Konformitätserklärung am Ende dieses Handbuchs angegebenen EG-Richtlinien und Normen zur EMV für industrielle Umgebungen.

Warnung: In Wohnbereich, Geschäfts- und Gewerbebereichen sowie Kleinbetrieben kann das CAN-CBX-AI420 Funkstörungen verursachen. In diesem Fall ist es erforderlich, dass der Anwender angemessene Maßnahmen ergreift.

Hinweis: Die zugesicherten EMV-Eigenschaften werden eingehalten, wenn:
- für die analogen Eingänge Leitungen mit maximal 3 m Länge eingesetzt werden.
- die Tragschiene an den Funktionserde-Kontakt (FE) angeschlossen ist!
  Bitte achten Sie darauf, den Widerstand des Kabels so gering wie möglich zu halten.

Bestimmungsgemäßer Gebrauch

Die bestimmungsgemäße Verwendung ist der Einsatz als CANopen Modul mit analogen Eingängen. Der Hersteller haftet nicht für Schäden, die durch unsachgemäßen Gebrauch, nicht bestimmungsgemäße Verwendung oder in Folge von Nichtbeachtung der Sicherheitshinweise und Warnungen verursacht werden.
Jeder Eingriff in das CAN-CBX-Modul durch nicht von esd autorisierte Personen führt zum Verlust aller Garantieansprüche.

- Das CAN-CBX-Modul ist nur für die Anwendung innerhalb von Gebäuden vorgesehen.
- Das CAN-CBX-Modul darf nicht in explosionsgefährdeten Bereichen und Zonen für Gase und Stäube sowie in explosivstoffgefährdeten Bereichen eingesetzt werden.
- Der Einsatz zu medizinischen Zwecken ist nicht zulässig.

Wartungshinweis

Innerhalb und außerhalb des CAN-CBX-Modul befinden sich keine vom Anwender zu wartenden Komponenten. Jeder Eingriff in das Gerät durch nicht von esd autorisierte Personen führt zum Verlust aller Garantieansprüche.

Umwelthinweis

Auf Dauer unbrauchbar gewordene Geräte sind in geeigneter Weise zu entsorgen oder dem Hersteller zur Entsorgung zu übergeben. Bitte leisten auch Sie Ihren Beitrag zum Schutz unserer Umwelt.

Inhalt

Darstellungskonventionen

In diesem Handbuch werden die folgenden Darstellungen zur Unterscheidung und Hervorhebung der aufgelisteten Programmbestandteile verwendet.

Darstellung von	**Beispiel**
Datei- und Pfadname	`/dev/null or <stdio.h>`
Funktionsnamen	***open()***
Konstanten	`NULL`
Datentypen	`uint32_t`
Variablennamen	*Count*

Das Dokument enthält die folgenden Textboxen um wichtige Informationen und Warn- und Gefahrenhinweise hervorzuheben.

Achtung:
Warn- und Gefahrenhinweise, die sie zu ihrer persönlichen Sicherheit sowie zur Vermeidung von Sachschäden beachten müssen.

Hinweis:
Hinweise enthalten wichtige oder nützliche Informationen.

Zahlendarstellung
Alle Zahlenangaben in diesem Dokument sind Dezimalzahlen, sofern nicht anders angegeben. Hexadezimalzahlen sind mit einem angehängten $_h$ dargestellt. Zum Beispiel, wird 42 hexadezimal als $2A_h$ dargestellt.

1. Übersicht

1.1 Beschreibung des Moduls

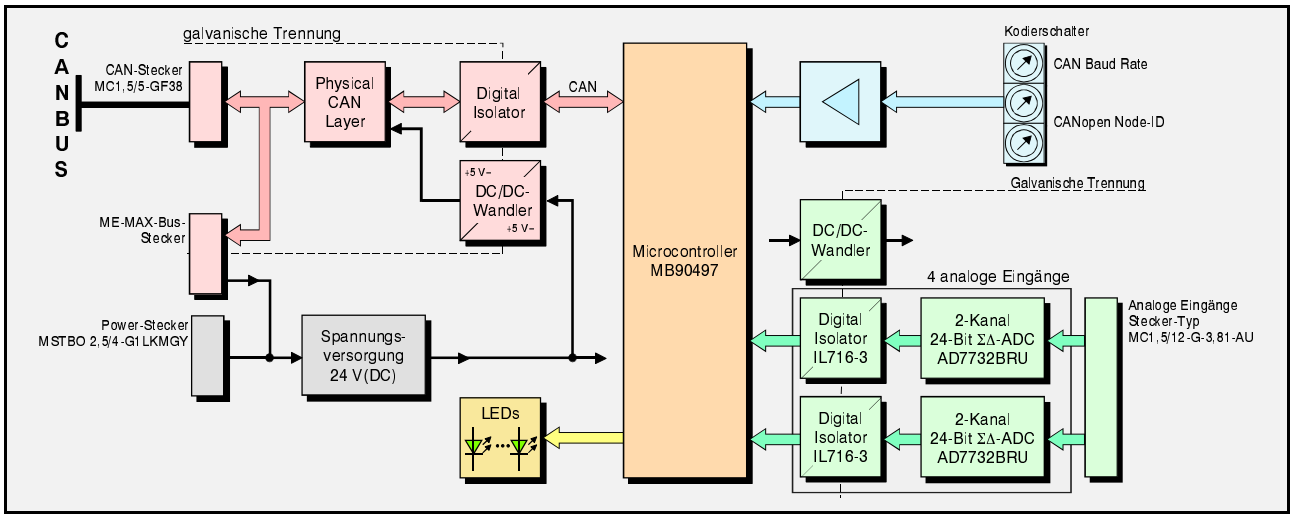


Abb. 1: Blockschaltbild des CAN-CBX-AI420-Moduls

Das CAN-CBX-AI420-Modul verfügt über einen MB90F497 Microcontroller, der die CAN-Daten in seinem lokalen SRAM zwischen speichert. Die Firmware wird im Flash gehalten. Zur Speicherung von Parametern dient ein serielles EEPROM.

Die Wandlung der vier differentiellen analogen Eingänge erfolgt über zwei $\Sigma\Delta$-Wandler. Die Wandler haben eine Auflösung von bis zu 24 Bit. Die in der Praxis erreichte Auflösung hängt wesentlich von der gewählten Sample-Time und der äußeren Beschaltung ab. Der Eingangsspannungsbereich der analogen Eingänge beträgt ±10 V.
Der Anschluss der Eingänge erfolgt über einen 12-poligen Schraub/Steckverbinder. Zum Schutz der anderen Baugruppen sind die analogen Eingänge über digitale Isolatoren galvanisch getrennt.

Die Versorgungsspannung und der CAN-Bus-Anschluss werden entweder über den in die Hutschiene integrierten InRailBus-Verbinder oder über separate Stecker zugeführt.

Die zu ISO 11898 kompatible CAN-Schnittstelle gestattet eine maximale Datenübertragungsrate von 1 MBit/s. Das CAN-Interface ist durch einen digitalen Datenkoppler und DC/DC-Wandler galvanisch getrennt.

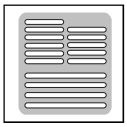
Die CANopen Knotennummer und die CAN-Bitrate können über drei außen liegende Drehschalter eingestellt werden.

2. Technische Daten

2.1 Allgemeine technische Daten

Versorgungsspannung	Nennspannung: 24 V/DC Eingangsspannungsbereich: 24 V ±20% Stromaufnahme (24 V, 20 ˚C): ca. 56 mA
Steckverbinder	24V (4-pol. Leiterplattensteckverbinder mit Federkraftanschluss-Kontakten X100) - 24V-Spannungsversorgung X101 (5-pol. CAN-CBX-TBUS-Verbinder, Phoenix Contact) - CAN-Schnittstelle und Spannungsversorgung über InRailBus 1G - 4 M (12-pol. Leiterplattensteckverbinder mit Federkraftanschluss-Kontakten, X500) - analoge Eingänge CAN (5-pol. Leiterplattensteckverbinder mit Federkraftanschluss-Kontakten, X400) - CAN-Schnittstelle Nur für Test und Programmierzwecke: X200 (6-pol. Leiterplattensteckverbinder) der Stecker befindet sich innerhalb des Gehäuses
Temperaturbereich	0 ˚C ... +70 ˚C Umgebungstemperatur
Luftfeuchtigkeit	max. 90%, nicht kondensierend
IP-Schutzklasse	IP20
Verschmutzungsgrad	maximal zulässig nach DIN EN 61131-2: Verschmutzungsgrad 2
Gehäuse	Kunststoffgehäuse für Tragschienenmontage NS35/7,5 DIN EN 60715
Abmessungen	Breite: 22,5 mm, Höhe: 99 mm, Tiefe: 114,5 mm (ohne Stecker)
Gewicht	140 g

Tabelle 1: Allgemeine Daten des Moduls



2.2 CPU-Baugruppe

CPU	16 Bit µC MB90F497
RAM	2 kByte integriert
Flash	64 kByte integriert
EEPROM	min. 256 Byte

Tabelle 2: Microcontroller

2.3 CAN-Schnittstelle

Anzahl	1
CAN-Controller	MB90F497, gemäß ISO 11898-1 (CANopen-Software unterstützt nur 11-Bit CAN-Identifier)
Galvanische Trennung des CAN-Interfaces gegenüber den anderen Baugruppen	über magnetischen Datenkoppler (ADUM120BR) und DC/DC-Wandler
Physical CAN Layer	Physical Layer gemäß ISO 11898-2, Übertragungsrate programmierbar von 10 kBit/s bis 1 MBit/s
Busabschluss	Abschlusswiderstand ist bei Bedarf extern zu setzen
Anschluss	5-pol. Leiterplattensteckverbinder mit Federkraftanschluss-Kontakten oder über CAN-CBX-TBUS-Verbinder, Phoenix Contact (InRailBus)

Tabelle 3: Daten der CAN-Schnittstelle

2.4 Analoge Eingänge

Anzahl	4 differenzielle ΣΔ-Wandler-Eingänge
Wandler-Typ	AD7732BRU
Auflösung	Auflösung des Wandlers bis zu 24 Bit
Eingangsspannungsbereich	±10 V
Eingangsimpedanz	min.: 100 kΩ, typ.: 124 kΩ
Wandlungszeit	programmierbar
Galvanische Trennung gegenüber den anderen Baugruppen	über magnetischen Datenkoppler
Schutzschaltungen	Überspannungsfestigkeit der Wandler-Eingänge: - bis zu ±16,5 V ohne Auswirkung auf den benachbarten Kanal - absolutes Maximum ±50 V

Tabelle 4: Daten der analogen Eingänge

2.5 Software-Unterstützung

Die Firmware des Moduls unterstützt CANopen® nach den CiA® CANopen Spezifikationen CiA 301 [1] und CiA DS-401 [2].

Die EDS-Datei des CAN-CBX-AI420 kann von der esd-Webseite unter www.esd.eu heruntergeladen werden.

3. Hardware-Installation

3.1 Anschlussplan

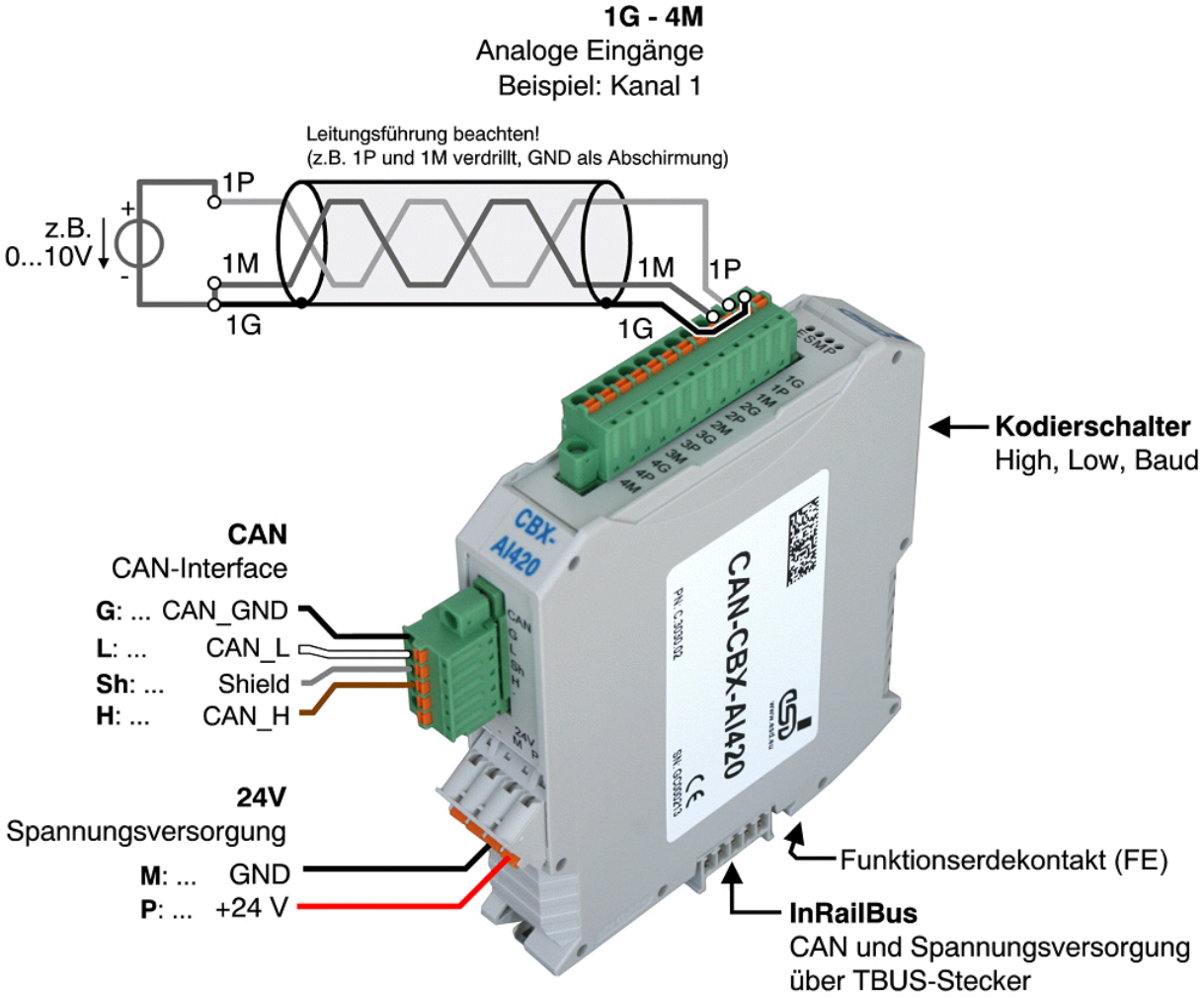


Abb. 2: Anschlüsse des CAN-CBX-AI420-Moduls

Hinweis:
Hinweise zum Leiteranschluss/Leiterquerschnitt entnehmen Sie bitte Seite 30.
Die Signalbelegung der Steckverbinder ist in tabellarischer Form ab Seite 24 abgedruckt.

3.2 LED-Anzeigen

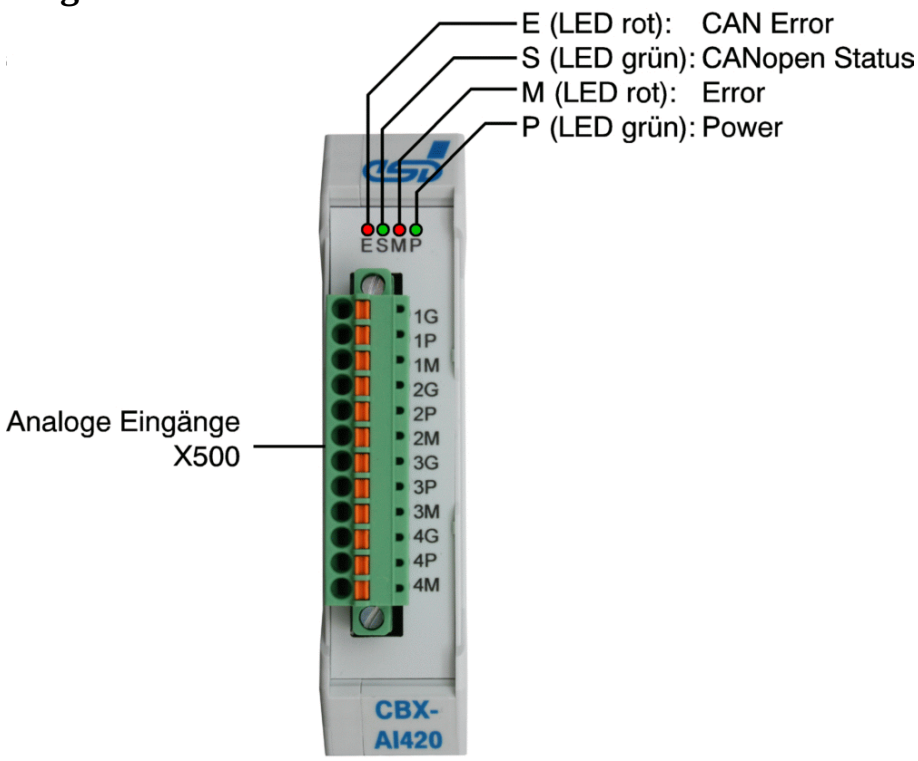


Abb. 3: Position der LEDs in der Frontplatte

Das CAN-CBX-AI420-Modul verfügt über vier Status-LEDs. Die Bezeichnung der Blinkzustände der einzelnen LEDs ist in Anlehnung an die von der CiA empfohlenen Bezeichnungen gemäß [3] gewählt. Die Bedeutung der Blinkzustände ist in den folgenden Kapiteln beschrieben.

3.2.1 Blinkzustände

Es gibt prinzipiell 8 Blinkzustände, die die LEDs annehmen können:

Blinkzustand	Anzeige
an	LED an
aus	LED aus
Blinken	LED blinkt mit 2,5 Hz
Flackern	LED flackert mit 10 Hz
1 Blitz	LED 200 ms an, 1400 ms aus
2 Blitze	LED 200 ms an, 200 ms aus, 200 ms an, 1000 ms aus
3 Blitze	LED 2x (200 ms an, 200 ms aus) + 1x (200 ms an, 1000 ms aus)
4 Blitze	LED 3x (200 ms an, 200 ms aus) + 1x (200 ms an, 1000 ms aus)

Tabelle 5: Anzeigefunktion der LEDs

Hinweis:
Rote und grüne LEDs werden entsprechend der CANopen Spezifikation [3] grundsätzlich gegenphasig angesteuert.
Bei bestimmten Blinkzuständen kann es bei gleichzeitiger Betrachtung aller LEDs zu einer Fehlinterpretation von Blinkzuständen nebeneinanderliegender LEDs kommen. Es wird daher ggf. empfohlen die Blinkzustände nebeneinanderliegender LEDs einzeln zu betrachten, indem die anderen LEDs verdeckt werden.

3.2.2 Bedeutung der CAN-Error-LED

LED-Kennung			Anzeigefunktion	
Auf-druck	Name	Farbe	Blinkzustand	Bedeutung
E	CAN-Error	rot	aus	kein Fehler
			1 Blitz	CAN-Controller ist in *Error Active*
			an	CAN-Controller ist *Bus Off* (oder beim Einschalten ist Kodierschalterstellung ID-Node > $7F_h$; siehe Besondere Blinkzustände Seite 16)
			2 Blitze	Heartbeat- oder Nodeguard-Fehler aufgetreten. Die LED geht automatisch wieder aus, wenn wieder Nodeguard/Heartbeat-Messages empfangen werden.

Tabelle 6: Anzeigefunktion der roten CANopen Error-LED

3.2.3 Bedeutung der CANopen-Status-LED

LED-Kennung			Anzeigefunktion	
Auf-druck	Name	Farbe	Blinkzustand	Bedeutung
S	CANopen-Status	grün	Blinken	*Preoperational*
			an	*Operational*
			1 Blitz	*Stopped*
			3 Blitze	Modul ist im Bootloader-Mode, die Power LED ist dabei aus. (oder Kodierschalterstellung ID-Node > $7F_h$ beim Einschalten; siehe Seite 16)

Tabelle 7: Anzeigefunktion der CANopen Status-LED

3.2.4 Bedeutung der Error-LED

LED-Kennung			Anzeigefunktion	
Auf-druck	Name	Farbe	Blinkzustand	Bedeutung
M	Error	rot	aus	kein Fehler
			an	CAN Overrun Error Die Sample-Rate ist so hoch eingestellt, dass die Firmware nicht in der Lage ist, alle Daten auf dem CAN-Bus zu senden.
			2 Blitze	Internal Software Error z.B.: - gespeicherte Daten hatten eine ungültige Checksumme und Default-Werte wurden geladen - interner Watchdog hat ausgelöst - Blinkzustand bleibt, bis das Modul einen Reset durchführt oder ein Fehler an den Ausgängen auftritt.
			Blinken	Fehler der A/D-Eingangsspannung Eingangsspannung an mindestens einem Kanal ≥ 10V

Tabelle 8: Anzeigefunktion der Error-LED

3.2.5 Bedeutung der Power-LED

LED-Kennung			Anzeigefunktion	
Auf-druck	Name	Farbe	Blinkzustand	Bedeutung
P	Power	grün	aus	keine Versorgungsspannung oder Modul ist im Bootloader-Modus (ID = 0), dieser Zustand wird durch die CANopen-Status-LED angezeigt (3 Blitze)
			an	Versorgungsspannung liegt an

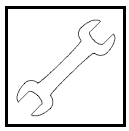
Tabelle 9: Anzeigefunktion der Power-LED

3.2.6 Besondere Blinkzustände

Dieser Zustand wird von der CANopen-Status-LED und der CAN-Error-LED gemeinsam angezeigt:

LED-Anzeige	Bedeutung
CANopen-Status-LED: 3 Blitze CAN-Error-LED: an	Die Kodierschalter für die Node-ID stehen auf einer ungültigen ID. Die Firmware-Anwendung wird angehalten.

Tabelle 10: Besondere Blinkzustände



3.2.7 Zuordnung LED-Bezeichnung zu Bezeichnung im Schaltplan

Bezeichnung auf CAN-CBX-AI420	Bezeichnung im Schaltplan *[1)]
E	LED200A
S	LED200B
M	LED200C
P	LED200D

*[1)] Der Schaltplan ist nicht Bestandteil dieses Handbuchs.

3.3 Kodierschalter

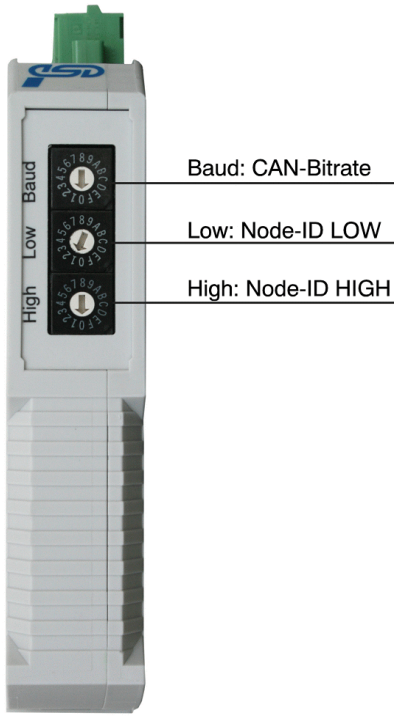


Abb. 4: Position der Kodierschalter

Achtung!
Die Auswertung der Kodierschalter durch die Firmware erfolgt beim Einschalten des Moduls. Änderungen der Einstellungen müssen daher **vor dem Einschalten** durchgeführt werden, da Änderungen während des Betriebes keine Auswirkungen haben.

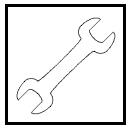
Auch nach einem Reset (z.B. NMT-Reset) werden die Einstellungen erneut eingelesen.

3.3.1 Einstellung der Node-ID über die Kodierschalter

Der Adressbereich des CAN-CBX-Moduls ist *dezimal* von 1 bis 127 bzw. *hexadezimal* von 01_h bis $7F_h$ einstellbar.

Der Kodierschalter **HIGH** dient zur Einstellung der 3 höherwertigen Bits (höherwertiges Nibble), der Kodierschalter **LOW** zur Einstellung der 4 niederwertigen Bits.

Hinweis:
Die folgenden Einstellungen sollten vermieden werden:
Einstellungen der Kodierschalter höher als $7F_h$ führen zu Fehlermeldungen, die rote CAN-Error-LED leuchtet dauerhaft.
Bei Einstellung der Kodierschalter auf 00_h befindet sich das Modul im Bootlader-Modus.



3.3.2 Einstellung der CAN-Bitrate über den Kodierschalter

Die CAN-Bitrate wird mit dem Kodierschalter **Baud** eingestellt.
Über den Kodierschalter können Werte von 0_h bis F_h eingestellt werden. Die Einstellung der Bitrate erfolgt entsprechend der folgenden Tabelle:

Kodierschalter-Stellung [Hex]	Bitrate [kBit/s]
0	1000
1	$666,\overline{6}$
2	500
3	$333,\overline{3}$
4	250
5	166
6	125
7	100
8	$66,\overline{6}$
9	50
A	$33,\overline{3}$
B	20
C	12,5
D	10
E	800
F	$83,\overline{3}$ *2

*2) Ab Firmware-Version 2.02 implementiert

Tabelle 11: Index der Bitrate

3.3.3 Zuordnung Kodierschalterbezeichnung zum Schaltplan

Bezeichnung auf CAN-CBX-AI420	Bezeichnung im Schaltplan *1)
Baud	SW301
Low	SW300
High	SW302

*1) Der Schaltplan ist nicht Bestandteil dieses Handbuchs.

3.4 Einbau des Moduls bei Verwendung des InRailBus-Verbinders

Sollen der CAN-Bus und die Versorgungsspannung über den InRailBus angeschlossen werden, gehen Sie bitte wie folgt vor:

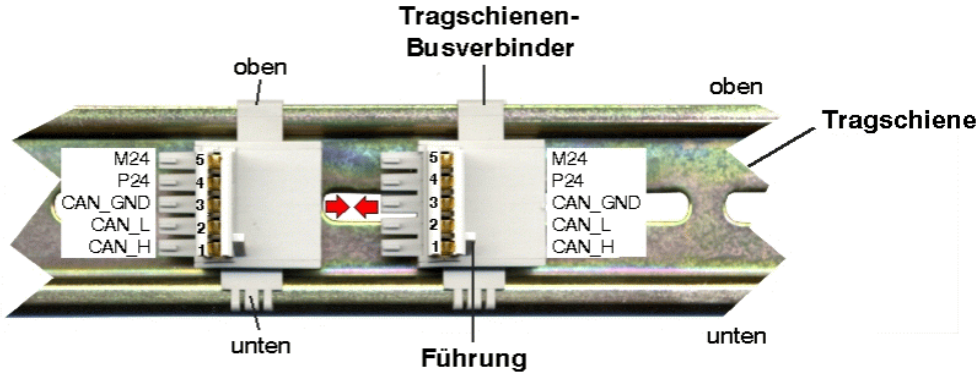


Abb. 5: Tragschiene mit Busverbinder

1. Der Tragschienen-Busverbinder des InRailBus wird an die Tragschiene angelegt und durch leichtes Andrücken auf der Tragschiene aufgerastet. Durch Zusammenstecken der einzelnen Busverbinder werden die Kommunikations- und Leistungssignale untereinander kontaktiert. Die Busverbinder können beliebig vor oder nach dem Aufstecken des CAN-CBX-Moduls verbunden werden.

2. Halten Sie das Modul leicht schräg nach hinten gekippt und setzen Sie das CBX-Modul auf den Busverbinder, so dass der obere Teil der Tragschiene dabei in die Einkerbung greift.

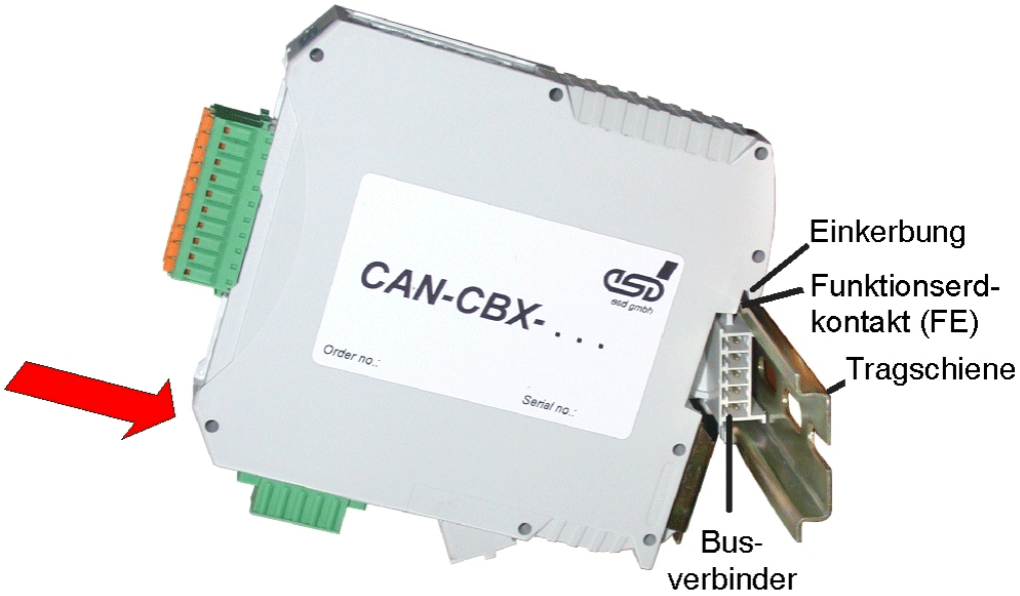


Abb. 6 : Einsetzen des CAN-CBX-Moduls

3. Schwenken Sie nun das CBX-Modul auf die Tragschiene auf, indem sie das Modul entsprechend

der Pfeilrichtung in Abbildung 6 nach unten an die Tragschiene heran drücken. Dabei wird das Gehäuse durch die Führungsschiene des Busverbinders mechanisch geführt.

4. Beim Aufschwenken des CAN-CBX-Moduls rastet der untere Fußriegel auf dem unteren Teil der Tragschiene ein. Das Modul sitzt nun fest auf der Tragschiene und ist über den Busverbinder mit dem InRailBus verbunden.
   Verbinden Sie ggf. noch die Busverbinder untereinander und schließen Sie die +24 V Versorgungsspannung und das CAN-Interface an den InRailBus an.

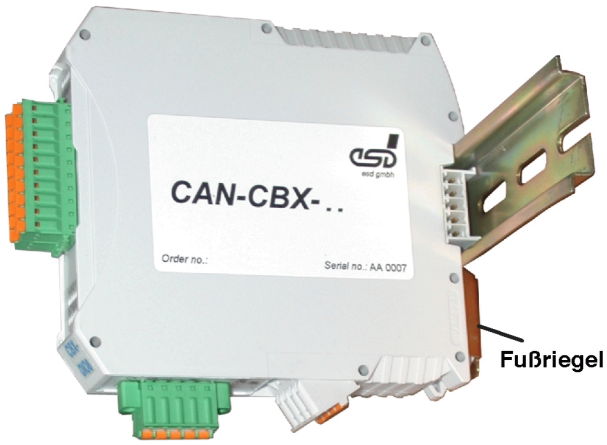


Abb. 7: Eingebautes CAN-CBX-Modul

3.4.1 Anschluss über den CBX-InRailBus

Um die Versorgungsspannung und die CAN-Signale über den InRailBus anschließen zu können, benötigen sie einen Anschlussstecker. Dieser Stecker ist nicht im Lieferumfang des CAN-CBX-Moduls enthalten. Er muss gesondert bestellt werden (Bestell-Nr. C.3000.02, siehe auch Bestellhinweise).

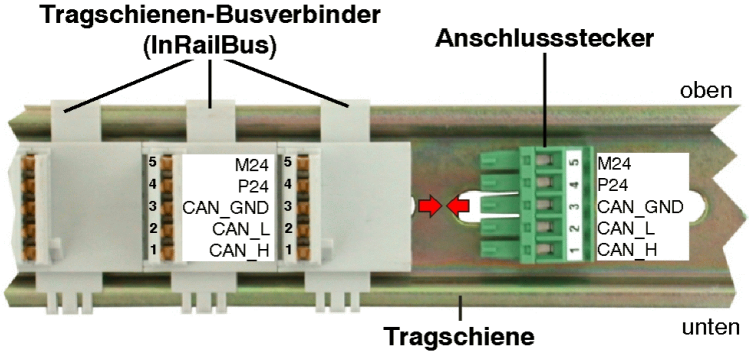


Abb. 8: Tragschiene mit InRailBus-Verbindern und Anschlussstecker

Stecken Sie den Anschlussstecker, wie in Abb. 8 beschrieben, von rechts in die Buchsenseite des äußeren Tragschienen-Busverbinders des InRailBus. Schließen sie nun das CAN-Interface und die Versorgungsspannung über den Anschlussstecker an.

3.4.2 Anschluss der Versorgungsspannung

Der Anschluss der Versorgungsspannung erfolgt über den +24V-Stecker oder über den InRailBus-Stecker.

Achtung!
Bitte beachten Sie die Sicherheitshinweise und Anforderungen bezüglich der Versorgungsstromkreise (siehe Seite 4)!

Achtung!
Die Anschlüsse der 24 V Spannungsversorgung sind intern verbunden und dürfen nicht von zwei unabhängigen Stromquellen gleichzeitig versorgt werden!

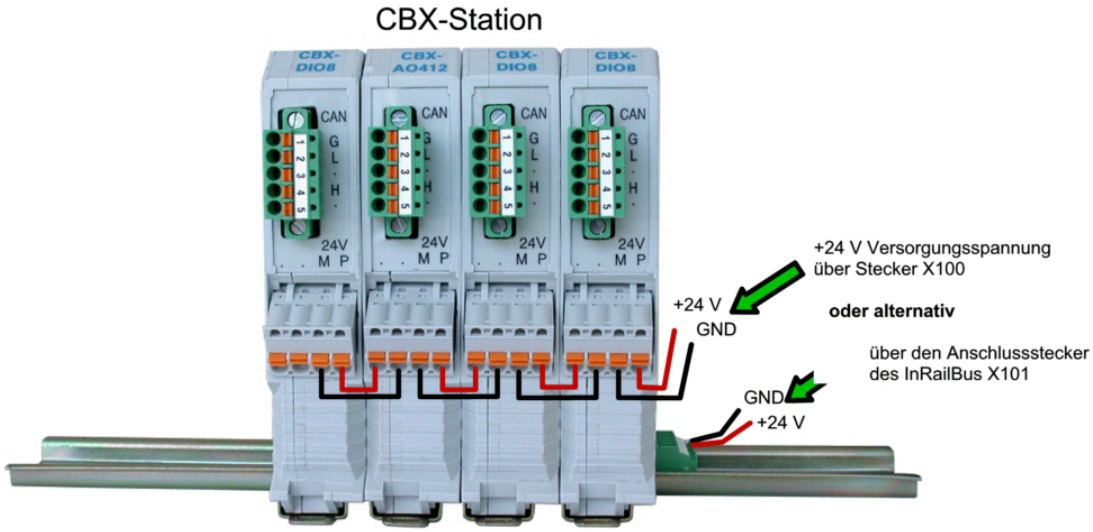


Abb. 9: CAN-CBX-Station mit Anschluss der Versorgungsspannung

Erdung der Tragschiene

Hinweis:
Die Tragschiene muss an den Funktionserde-Kontakt (FE) angeschlossen werden!
Bitte achten Sie darauf, den Widerstand des Kabels so gering wie möglich zu halten.

Der Funktionserde-Kontakt ist ein Strompfad mit geringem Widerstand zwischen Stromkreis und Erde, der nicht als Schutzmaßnahme gedacht ist, sondern zur Erhöhung der Stabilität dient. Er bietet keinen Berührungsschutz für Personen.

3.4.3 Anschluss von CAN

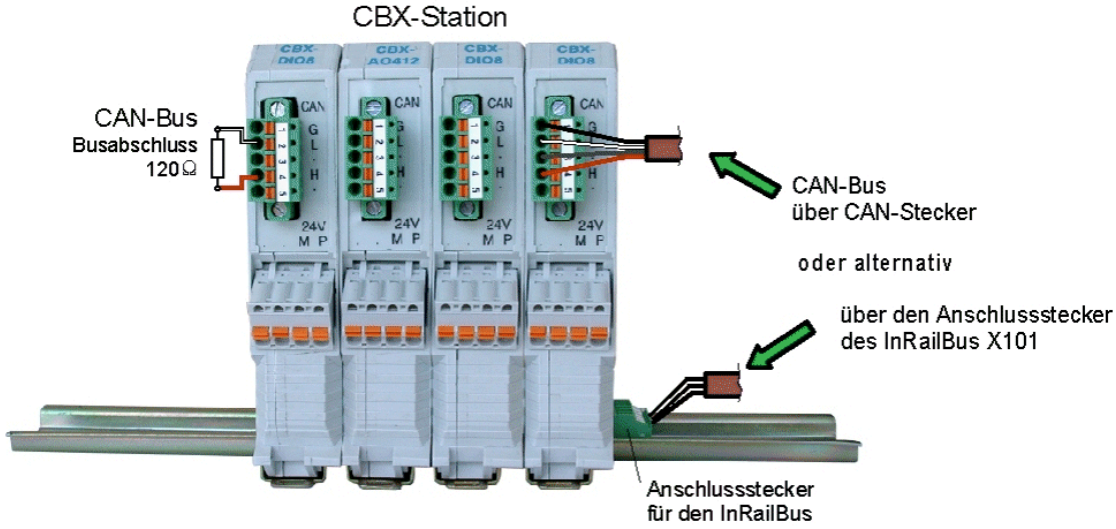


Abb. 10: Anschluss von CAN an die CBX-Station

Generell besteht die Möglichkeit die CAN-Signale über den InRailBus oder über den CAN-Stecker eines äußeren CAN-CBX-Moduls der CBX-Station einzuspeisen. Die Signale werden dann über den InRailBus an die folgenden CAN-CBX-Module der CAN-CBX-Station weitergeleitet. Die CAN-Signale dürfen über den CAN-Stecker des CAN-CBX-Moduls, das am anderen Ende der CBX-Station montiert ist, weitergeführt werden. Über die CAN-Stecker der mittleren CAN-CBX-Module der CBX-Station dürfen die CAN-Signale jedoch nicht weitergeführt werden, da dies zu unzulässigen Verzweigungen führt.

Bitte beachten Sie, dass an das CAN-CBX-Modul, das sich am Ende des InRailBus befindet ein Bus-Abschlusswiderstand angeschlossen werden muss, wenn der CAN-Bus dort endet (siehe Abb. 10).

3.5 Ausbau des CAN-CBX-Moduls vom InRailBus

Ist das CAN-CBX-Modul über den InRailBus verbunden, gehen Sie beim Ausbau wie folgt vor:

Lösen Sie das Modul von der Tragschiene indem Sie den Fußriegel (siehe Abb. 7) nach unten ziehen (z.B. mit einem Schraubendreher). Dabei löst sich das Modul unten von der Tragschiene und kann abgezogen werden.

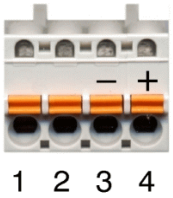
Hinweis:
Es ist möglich, einzelne Gehäuse aus dem Verbund zu lösen, ohne die InRailBus-Verbindung zu unterbrechen, da beim Ziehen einzelner Module aus dem Verbund die Signalkette nicht unterbrochen wird.

4. Steckerbelegungen

4.1 Spannungsversorgung 24V (X100)

Gerätebuchse: Phoenix Contact MSTBO 2,5/4-G1L-KMGY
Leitungsstecker: Phoenix Contact FKCT 2,5/4-ST, 5.0 mm Raster,
Federkraftanschluss-Kontakte,
Phoenix Contact Bestell-Nr.: 19 21 90 0 (im Lieferumfang enthalten)
Zu Leiteranschluss und Leiterquerschnitt siehe Seite 30.

Pin-Zuordnung:



Pin-Belegung:

Gehäusebezeichnung	24V			
	·	·	M	P
Stecker-Aufdruck	(frei)	(frei)	-	+

Pin-Nr.	1	2	3	4
Signal	P24 (+ 24 V)	M24 (GND)	M24 (GND)	P24 (+ 24 V)

Siehe auch Anschlussplan Seite 13.

Hinweis:
Die Pins 1 und 4 sind intern miteinander verbunden.
Die Pins 2 und 3 sind intern miteinander verbunden.

Signalbeschreibung:

P24... Versorgungsspannung +24 V
M24... Bezugspotenzial

4.2 CAN

4.2.1 CAN-Interface-Schaltung

Das CAN Physical Layer ist gemäß ISO 11898-2 ausgelegt. Die differentiellen CAN-Bus-Signale sind von den anderen Potenzialen über einen digitalen Isolator und DC/DC-Wandler galvanisch getrennt.

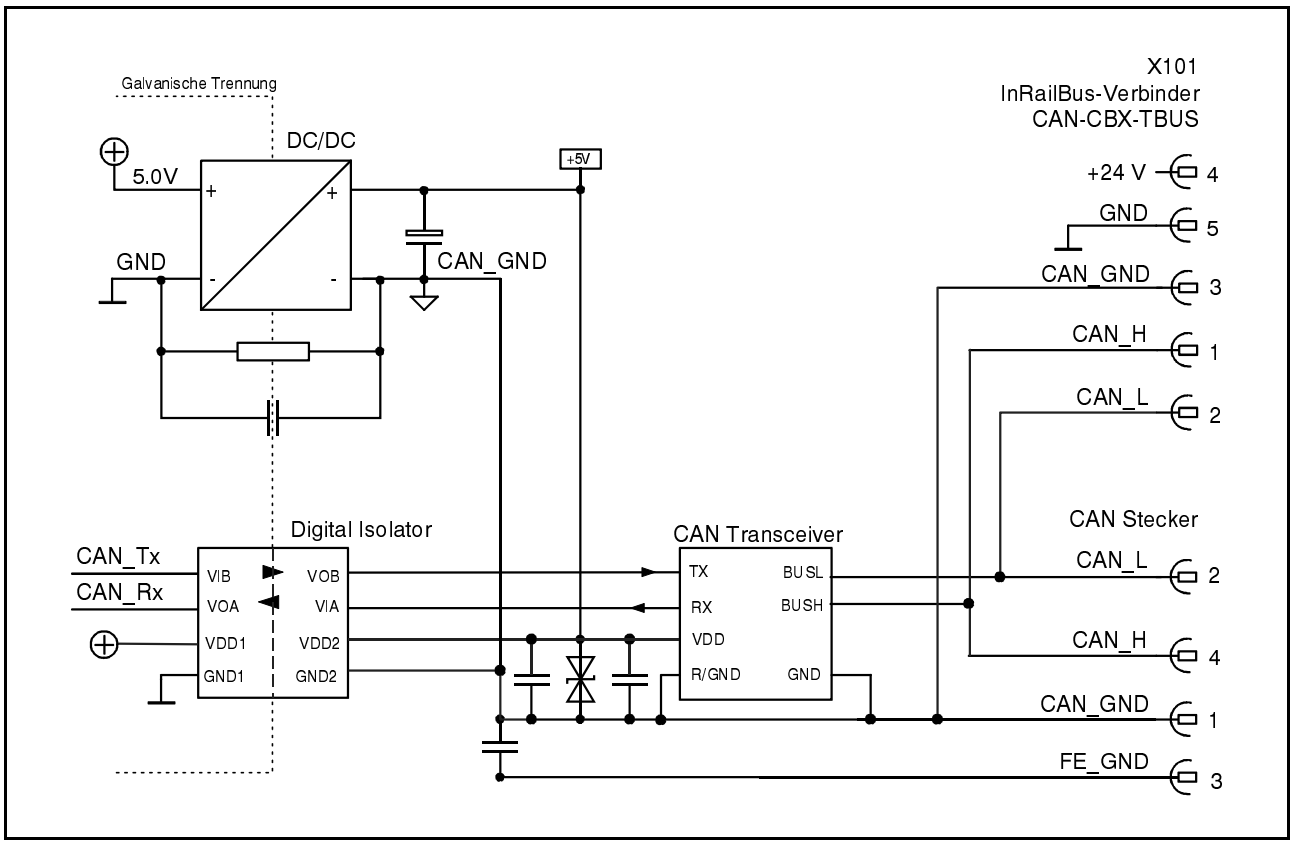
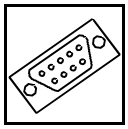


Abb. 11: Schaltung des CAN-Interface

Die CAN Schnittstelle kann entweder über den CAN-Stecker oder optional über den InRailBus angeschlossen werden. Benutzen Sie für den Anschluss über den InRailBus den Tragschienenbusverbinder des CBX-InRailBus (CAN-CBX-TBUS), siehe Bestellhinweise (Seite 99).

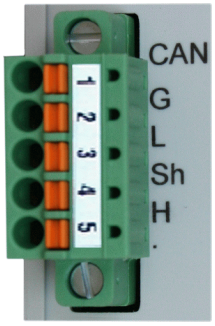


Steckerbelegung

4.2.2 CAN-Stecker (X400)

Gerätebuchse: Phoenix Contact MC 1,5/5-GF-3,81
Leitungsstecker: Phoenix Contact FK-MCP 1,5/5-STF-3,81, Federkraftanschluss-Kontakte
Phoenix Contact Bestell-Nr.: 1851261 (im Lieferumfang enthalten)
Zu Leiteranschluss und Leiterquerschnitt siehe Seite 30.

Pin-Zuordnung:
(Ausschnitt Leitungsstecker mit Gehäuseaufdruck)



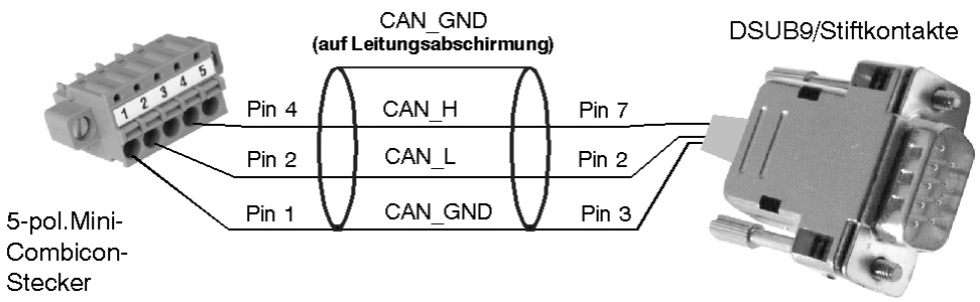
Pin-Belegung:

Gehäuse-aufdruck	Signal	Pin
G	CAN_GND	1
L	CAN_L	2
Sh	Shield	3
H	CAN_H	4
•	-	5

Signalbeschreibung:

CAN_L, CAN_H ... CAN-Signale
CAN_GND ... Bezugspotenzial des lokalen CAN-Physical Layers
Shield ... Anschluss für Leitungsabschirmung (bei Hutschienenmontage direkter Kontakt zu Hutschienenpotenzial)
- ... nicht angeschlossen

Empfehlung eines Adapterkabels 5-pol-COMBICON (hier Leitungsstecker FK-MCP1,5/5-STF-3,81 mit Federkraftanschluss-Kontakten) auf 9-pol-DSUB:

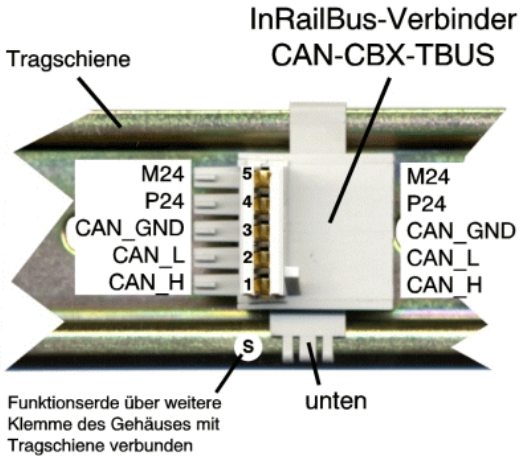


Die Belegung des 9-poligen DSUB-Steckers erfolgt nach CiA DS 102.

4.2.3 CAN und Versorgungsspannung über InRailBus X101

Steckertyp: Tragschienensteckverbinder CAN-CBX-TBUS
(Phoenix Contact ME 22,5 TBUS 1,5/5-ST-3,81 KMGY)

Steckeraufsicht:

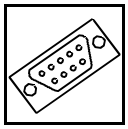


Pin-Belegung:

Pin	Signal
5	M24 (GND)
4	P24 (+24 V)
3	CAN_GND
2	CAN_L
1	CAN_H
S	FE (PE_GND)

Signalbeschreibung:

CAN_L,
CAN_H ... CAN-Signale
CAN_GND ... Bezugspotenzial des lokalen CAN-Physical Layers
P24... Versorgungsspannung +24 V
M24... Bezugspotenzial
FE... direkt am Gehäuse des Moduls: Funktionserde (mit Hutschienenpotenzial verbunden)



Steckerbelegung

4.3 Analoge Eingänge

4.3.1 Schaltung Analoger Eingänge

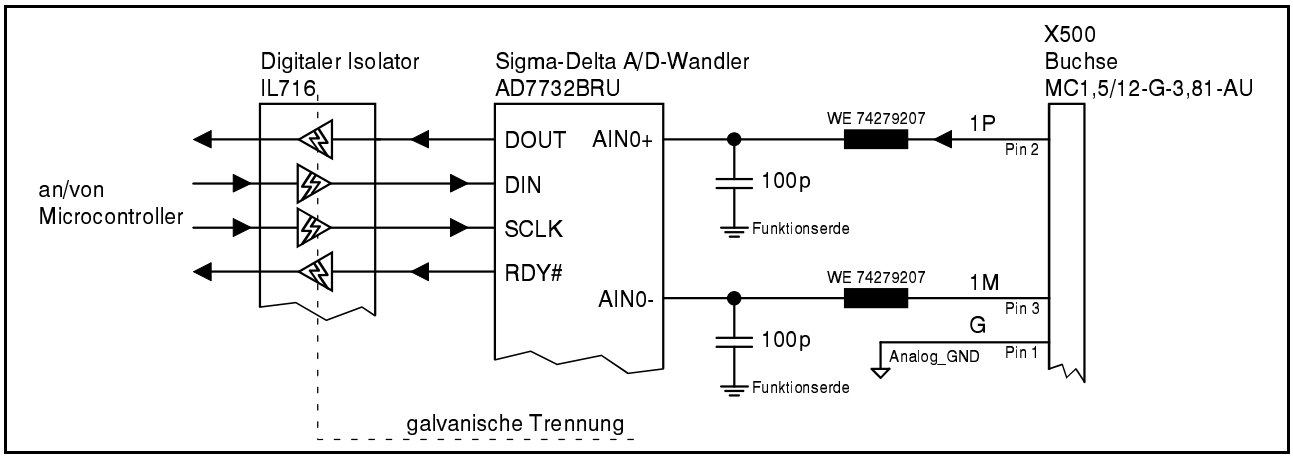


Abb. 12: Schaltung der analogen Eingänge (Beispiel: Kanal 1)

Achtung:
Durch die interne Schaltung des ΣΔ-Wandlers (siehe Bild unten) und die Verdrahtung des Gebersignals kommt es zu einem Gain-Fehler bei der Messung. Dieser wird um so geringer, je niederohmiger der Anschluss des Gebers gewählt wird.
Außerdem sollte der Widerstand der beiden Messleitungen des differentiellen Analogeingangs identisch sein.

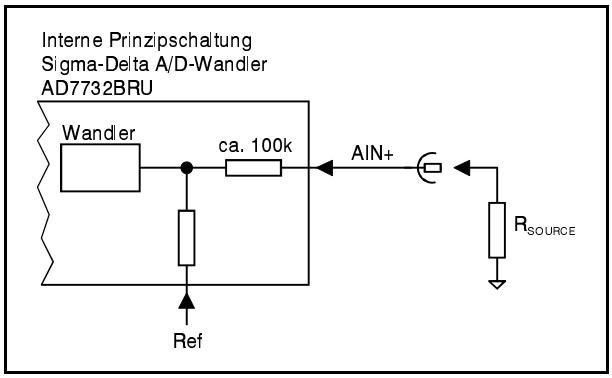
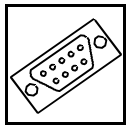


Abb. 13: Interne Prinzipschaltung des ΣΔ-Wandlers

Mit steigendem Widerstand R_{SOURCE} (siehe Abbildung 13) steigt der Gain-Fehler.

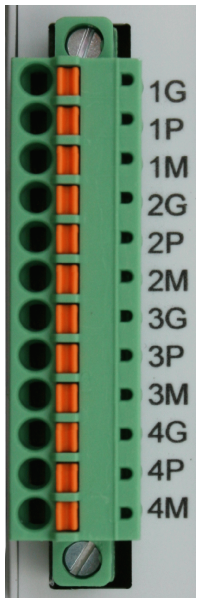


4.3.2 Analoge Eingänge X500

Gerätebuchse: Phoenix Contact MC 1,5/12-GF-3,81
Leitungsstecker: Phoenix Contact MC 1,5/12-STF-3,81 , Federkraftanschluss-Kontakte
Phoenix Contact Bestell-Nr.: 1851290 (im Lieferumfang enthalten)
Zu Leiteranschluss und Leiterquerschnitt siehe Seite 30.

Pin-Zuordnung:

(Ansicht Leitungsstecker mit Gehäuseaufdruck)



Pin Belegung

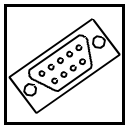
Pin:	Signal:
1	1G
2	1P
3	1M
4	2G
5	2P
6	2M
7	3G
8	3P
9	3M
10	4G
11	4P
12	4M

Signalbeschreibung:

*x*P... Plus-Pin des differenziellen analogen Eingangs (*x* = 1, 2, 3, 4)
*x*M... Minus-Pin des differenziellen analogen Eingangs (*x* = 1, 2, 3, 4)
*x*G ... Bezugspotenzial Analog-Ground (*x* = 1, 2, 3, 4)

Hinweis:
Die *x*G-Pins sind galvanisch miteinander verbunden.
Die *x*P- und *x*M-Eingänge können einem beliebigen *x*G-Kontakt zugeordnet werden.
Um geringe Leiterschleifen zu erreichen wird empfohlen einen nahegelegenen GND-Pin zu verwenden.

Hinweis:
Die zugesicherten EMV-Eigenschaften werden eingehalten, wenn für die analogen Eingänge Leitungen mit maximal 3 m Länge eingesetzt werden.

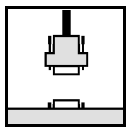


Steckerbelegung

4.4 Leiteranschluss/Leiterquerschnitt

Die folgende Tabelle enthält einen Auszug aus den technischen Daten der verwendeten Stecker.

Schnittstelle	**Spannungs-versorgung 24 V** [7]	**Analoge Ausgänge, CAN** [8]
Steckertyp Leitungsstecker (Artikelfamilie)	**FKCT 2,5/..-ST KMGY**	**FK-MCP 1,5/..-STF-3,81**
Anschlussart	Zugfeder-anschluss	Federkraft-anschluss
Abisolierlänge	10 mm	9 mm
Leiterquerschnitt starr min	0,2 mm²	0,14 mm²
Leiterquerschnitt starr max	2,5 mm²	1,5 mm²
Leiterquerschnitt flexibel min	0,2 mm²	0,14 mm²
Leiterquerschnitt flexibel max	2,5 mm²	1,5 mm²
Leiterquerschnitt flexibel mit Aderendhülse ohne Kunststoffhülse min	0,25 mm²	0,25 mm²
Leiterquerschnitt flexibel mit Aderendhülse ohne Kunststoffhülse max	2,5 mm²	1,5 mm²
Leiterquerschnitt flexibel mit Aderendhülse mit Kunststoffhülse min	0,25 mm²	0,25 mm²
Leiterquerschnitt flexibel mit Aderendhülse mit Kunststoffhülse max	2,5 mm²	0,5 mm²
Leiterquerschnitt AWG/kcmil min	24	26
Leiterquerschnitt AWG/kcmil max	12	16
2 Leiter gleichen Querschnitts starr min	unzulässig	unzulässig
2 Leiter gleichen Querschnitts starr max	unzulässig	unzulässig
2 Leiter gleichen Querschnitts flexibel min	unzulässig	unzulässig
2 Leiter gleichen Querschnitts flexibel max	unzulässig	unzulässig
2 Leiter gleichen Querschnitts flexibel mit Aderendhülse ohne Kunststoffhülse min	unzulässig	unzulässig
2 Leiter gleichen Querschnitts flexibel mit Aderendhülse ohne Kunststoffhülse max	unzulässig	unzulässig
2 Leiter gleichen Querschnitts flexibel mit Zwillingsaderendhülsen (TWIN-AEH) mit Kunststoffhülse min	0,5 mm²	unzulässig
2 Leiter gleichen Querschnitts flexibel mit Zwillingsaderendhülsen (TWIN-AEH) mit Kunststoffhülse max	1,0 mm²	unzulässig
AWG nach UL/CUL min	26	28
AWG nach UL/CUL max	12	16



5. Korrekte Verdrahtung galvanisch getrennter CAN-Netze

Generell sind bei der CAN-Verdrahtung sämtliche gültigen Richtlinien (DIN, VDE) bzgl. EMV-gerechtem Aufbau, Leitungsführung, Leiterquerschnitte, zu verwendende Materialien und Mindest-abstände zu beachten.

5.1 Leicht störbehaftete Industrieumgebung (zweiadrig verdrillte Leitung)

5.1.1 Grundregeln

Hinweis:
esd garantiert die EU-Konformität des Produkts, wenn für die CAN-Verdrahtung mindestens Kabel mit einfach abgeschirmten **zweiadrig** verdrillten Leitungen verwendet werden, die die Anforderungen der Norm ISO 11898-2 erfüllen. Einfach abgeschirmte vieradrig verdrillte Leitungen, wie in Kapitel 5.2 beschrieben, stellen die EU-Konformität ebenfalls sicher.

Die folgenden Grundregeln für die CAN-Bus Verdrahtung mit einfach abgeschirmten zweiadrig verdrillten Leitungen sollten unbedingt beachtet werden:

1	Es ist ein geeigneter Leitungstyp (Wellenwiderstand ca. 120 Ω ±10%) mit ausreichendem Ader-querschnitt (0.22 mm²) zu verwenden. Der Spannungsabfall auf der Leitung ist zu berücksichtigen!
2	Für den Einsatz in leicht störbehafteter Industrieumgebung ist mindestens ein zweiadriges CAN Kabel zu verwenden. Verbinden von ● zwei verdrillten Adern mit den CAN Signalleitungen (CAN_H, CAN_L) und ● der Kabel-Abschirmung mit dem Bezugspotenzial (CAN_GND)!
3	Das Bezugspotenzial CAN_GND muss an genau **einem** Punkt mit Funktionserde (FE) verbunden werden.
4	Ein CAN-Netz darf sich nicht verzweigen (Ausnahme: kurze Stichleitungen) und muss an beiden Enden mit dem Wellenwiderstand der Leitung (in der Regel 120 Ω ±10%) abgeschlossen werden (zwischen den Signalen CAN_L und CAN_H und **nicht** an GND)!
5	Stichleitungen sind so kurz wie möglich zu halten (l < 0,3 m)!
6	Die Baudrate muss an die Leitungslänge angepasst werden.
7	Die CAN-Leitungen sollten nicht in unmittelbarer Nähe von Störquellen verlegt werden. Lässt sich dies nicht vermeiden, so sind doppelt abgeschirmte Leitungen vorzuziehen.

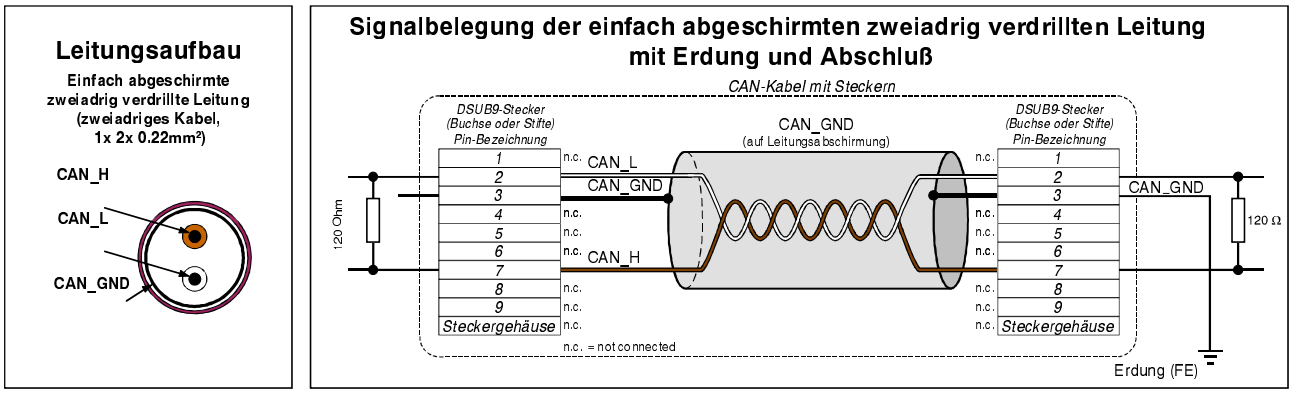


Abb. 14: CAN Verdrahtung für den Einsatz in leicht störbehafteter Industrieumgebung

5.1.2 Verkabelung

- Bei Geräten, die pro CAN-Netz nur einen CAN-Stecker besitzen, T-Stück und Stichleitung (kürzer als 0,3 m) verwenden (als Zubehör lieferbar).

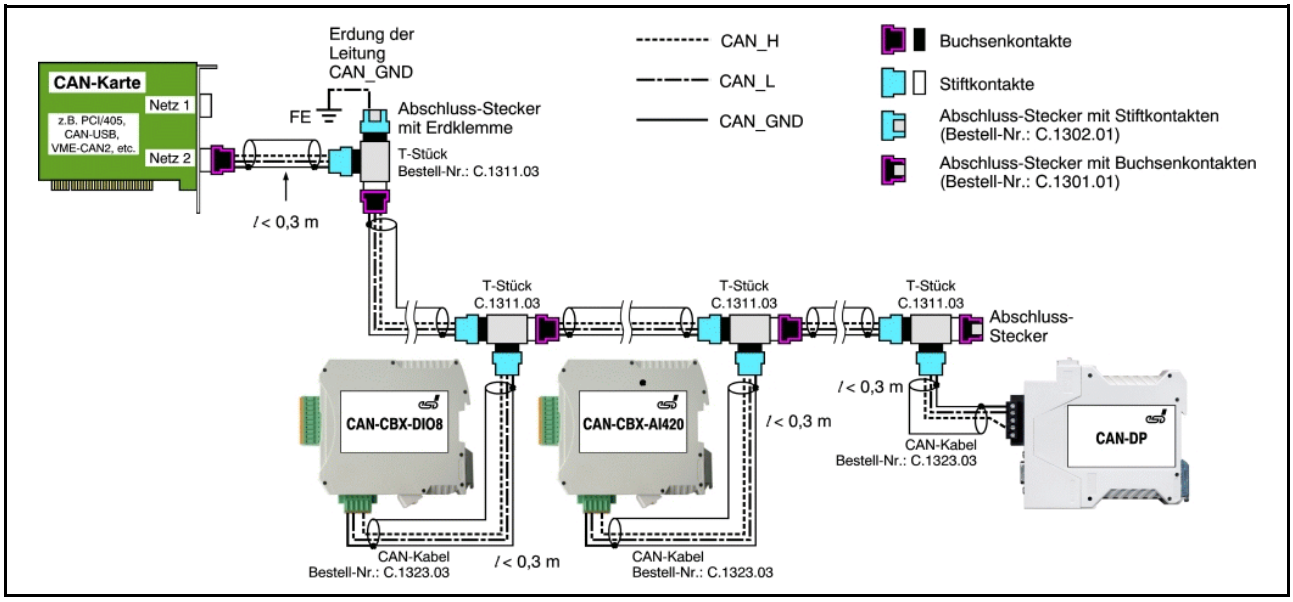


Abb. 15: Beispiel für korrekte Verdrahtung einfach abgeschirmter zweiadrig verdrillter Leitung

5.1.3 Abschlusswiderstand

- Verfügt das verwendete CAN-Interface nicht über einen integrierten CAN-Busabschluss und befindet es sich an einem Ende des CAN-Busses, so ist ein externer Abschlussstecker zu verwenden!

- 9-polige DSUB-Abschlussstecker mit Stift- oder Buchsenkontakten und Erdungsklemme sind als Zubehör erhältlich.

5.2 Stark störbehaftete Industrieumgebung (vieradrig verdrillte Leitung)

5.2.1 Grundregeln

Die folgenden Grundregeln für die CAN-Bus Verdrahtung mit einfach abgeschirmten vieradrig verdrillten Leitungen sollten unbedingt beachtet werden:

1	Es ist ein geeigneter Leitungstyp (Wellenwiderstand ca. 120 Ω ±10%) mit ausreichendem Aderquerschnitt (0.22 mm²) zu verwenden. Der Spannungsabfall auf der Leitung ist zu berücksichtigen!
2	Für den Einsatz in stark störbehafteter Industrieumgebung ist ein vieradriges CAN-Kabel zu verwenden. Verbinden von ● zwei verdrillten Adern mit den CAN Signalleitungen (CAN_H, CAN_L) und ● die anderen beiden verdrillten Adern mit dem Bezugspotenzial (CAN_GND) und ● die Leitungsabschirmung an Funktionserde (FE) an mindestens einem Punkt!
3	Das Bezugspotenzial CAN_GND muss an genau **einem** Punkt mit Funktionserde (FE) verbunden werden.
4	Ein CAN-Netz darf sich nicht verzweigen (Ausnahme: kurze Stichleitungen) und muss an beiden Enden mit dem Wellenwiderstand der Leitung (in der Regel 120 Ω ±10%) abgeschlossen werden (zwischen den Signalen CAN_L und CAN_H und **nicht** an GND)!
5	Stichleitungen sind so kurz wie möglich zu halten (l < 0,3 m)!
6	Die Baudrate muss an die Leitungslänge angepasst werden.
7	Die CAN-Leitungen sollten nicht in unmittelbarer Nähe von Störquellen verlegt werden. Lässt sich dies nicht vermeiden, so sind doppelt abgeschirmte Leitungen vorzuziehen.

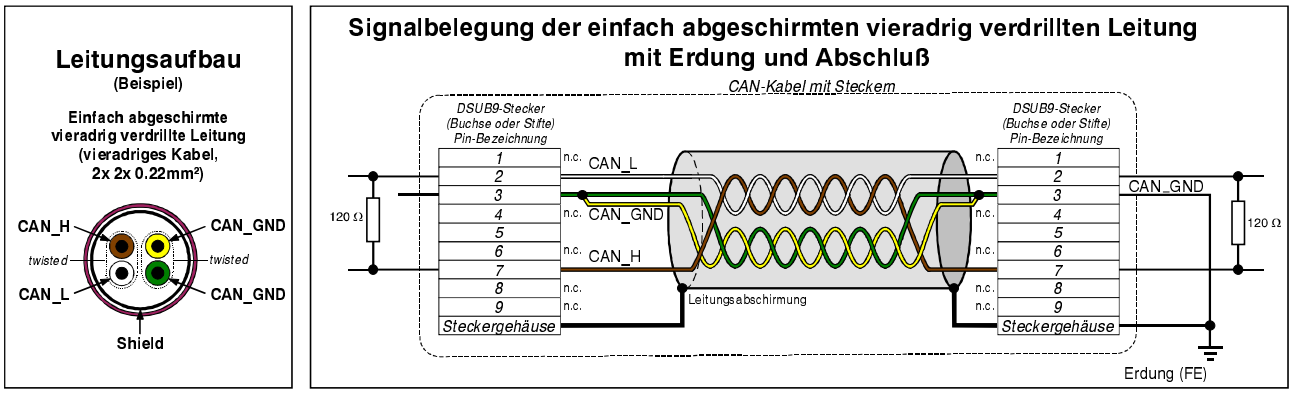


Abb. 16: CAN-Verdrahtung für stark störbehaftete Industrieumgebung

5.2.2 Verkabelung

Achtung:
Werden einfach abgeschirmte, vieradrig verdrillte Leitungen verwendet, ist für den CAN-Bus Steckverbinder ein T-Verbindungsstecker zu verwenden, der den Anschluss zweier CAN-Kabel an einen Stecker gestattet, und bei dem die Kabel-Abschirmung (Shield) durchgeführt wird, z.B. DSUB9-Stecker von ERNI (ERBIC CAN BUS MAX, Bestell-Nr.:154039).

Die Verwendung des esd-T-Connectors (Bestell-Nr: C.1311.03) wird für einfach abgeschirmte vieradrig verdrillte Leitungen nicht empfohlen, da das Schirm-Potenzial des leitenden DSUB-Gehäuses nicht durch diesen T-Connector-Typ durchgeführt wird.

Darüber hinaus ist die gemischte Verwendung von zwei- und vieradrig verdrillten Leitungen zu vermeiden!

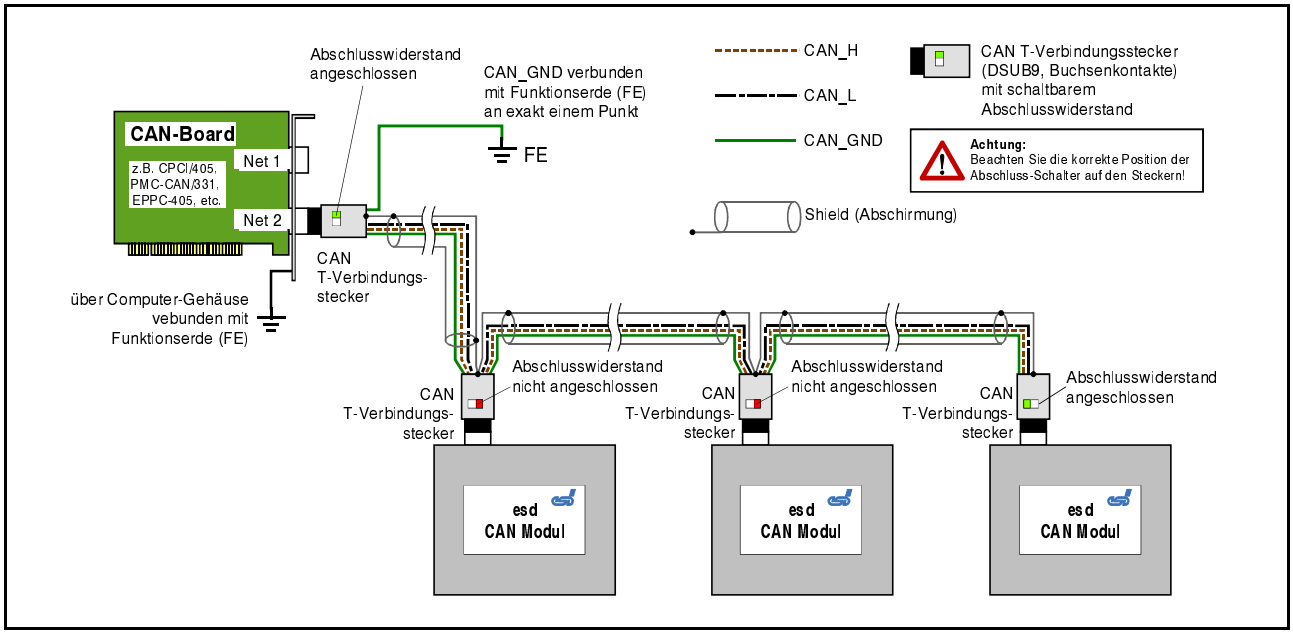
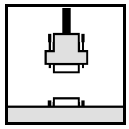


Abb. 17: Beispiel für korrekte Verdrahtung einfach abgeschirmter, vieradrig verdrillter Leitungen

5.2.3 Abschlusswiderstand

- Verfügt das verwendete CAN-Interface nicht über einen integrierten CAN-Busabschluss und befindet es sich an einem Ende des CAN-Busses, so ist ein externer Abschlussstecker zu verwenden!

- 9-polige DSUB-Abschlussstecker mit integriertem, umschaltbarem Abschlusswiderstand können z.B. von ERNI (ERBIC CAN BUS MAX, Buchsenkontakte, Bestell-Nr.:154039) bezogen werden.



5.3 Erdung

- Bei CAN-Modulen mit galvanischer Trennung muss CAN_GND zwischen den CAN-Modulen verbunden werden!

- CAN_GND muss an **exakt einem** Punkt im Netz mit dem Erdpotenzial (FE) verbunden werden!

- Jedes CAN-Modul mit galvanischer Verbíndung zum Erdpotential wirkt wie eine Erdung.
  Aus diesem Grund darf nicht mehr als ein CAN-Modul mit galvanischer Verbindung zum Erdpotential angeschlossen werden!

- Die Erdung kann z.B. an einem Abschlussstecker vorgenommen werden.

5.4 Bus-Länge

- Optokoppler verzögern die CAN-Signale. esd-Module erreichen typischerweise eine Leitungslänge von 37 m bei 1 MBit/s. Voraussetzung hierfür ist ein abgeschlossenes Netz ohne Impedanzstörungen, wie z.B. längere Stichleitungen >> 0.3 m.

Bit-Rate [kBit/s]	typische Werte der erreichbaren Leitungslänge **mit esd-Interface** l_{max} [m]	**CiA-Empfehlungen** (07/95) für erreichbare Leitungslängen l_{min} [m]
1000	37	25
800	59	50
$666,\overline{6}$	80	-
500	130	100
$333,\overline{3}$	180	-
250	270	250
166	420	-
125	570	500
100	710	650
$83,\overline{3}$	850	-
$66,\overline{6}$	1000	-
50	1400	1000
$33,\overline{3}$	2000	-
20	3600	2500
12,5	5400	-
10	7300	5000

Tabelle 12: Erreichbare Leitungslängen in Abhängigkeit von der Bitrate (mit esd-CAN-Interfaces)

Hinweis:
Bitte beachten Sie die Empfehlungen gemäß ISO 11898 für die Auswahl der Leitungsquerschnitte in Abhängigkeit von der Kabellänge.

5.5 Beispiele für CAN-Kabel

5.5.1 Kabel für leicht störbehaftete Industrieumgebung (zweiadrig)

Hersteller	Leitungstyp
U.I. LAPP GmbH Schulze-Delitzsch-Straße 25 70565 Stuttgart Germany www.lappkabel.de	z.B. UNITRONIC ®-BUS CAN UL/CSA (1x 2x 0.22) (UL/CSA approved) Bestell-Nr.: 2170260 UNITRONIC ®-BUS-FD P CAN UL/CSA (1x 2x 0.25) (UL/CSA approved) Bestell-Nr.: 2170272
ConCab GmbH Äußerer Eichwald 74535 Mainhardt Germany www.concab.de	z.B. BUS-PVC-C (1x 2x 0,22 mm²) Bestell-Nr.: 93 022 016 (UL appr.) BUS-Schleppflex-PUR-C (1x 2x 0,25 mm²) Bestell-Nr.: 94 025 016 (UL appr.)

5.5.2 Kabel für stark störbehaftete Industrieumgebung (vieradrig)

Hersteller	Leitungstyp
U.I. LAPP GmbH Schulze-Delitzsch-Straße 25 70565 Stuttgart Germany www.lappkabel.de	z.B. UNITRONIC ®-BUS CAN UL/CSA (2x 2x 0.22) (UL/CSA approved) Bestell-Nr.: 2170261 UNITRONIC ®-BUS-FD P CAN UL/CSA (2x 2x 0.25) (UL/CSA approved) Bestell-Nr.: 2170273
ConCab GmbH Äußerer Eichwald 74535 Mainhardt Germany www.concab.de	z.B. BUS-PVC-C (2x 2x 0,22 mm²) Bestell-Nr.: 93 022 026 (UL appr.) BUS-Schleppflex-PUR-C (2x 2x 0,25 mm²) Bestell-Nr.: 94 025 026 (UL appr.)

Hinweis:
Fertig konfektionierte Leitungen können bei **esd** bezogen werden.

6. CAN-Bus Troubleshooting Guide

Der CAN-Bus Troubleshooting Guide ist eine Anleitung zum Auffinden und Beseitigen der häufigsten Hardware-Fehlerursachen in der CAN-Bus-Verdrahtung.

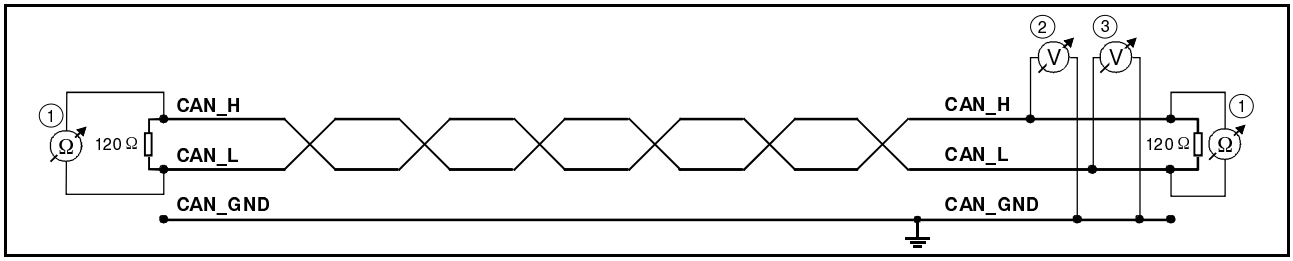


Abb. 18: Vereinfachtes Schaltbild eines CAN-Netzwerks

6.1 Bus-Abschluss

Der Bus-Abschluss wird verwendet, um den Widerstand eines Knotens an den Widerstand der verwendeten Busleitung anzupassen. Ist die Impedanz falsch angepasst, wird das gesendete Signal nicht ganz von der Last aufgenommen und zum Teil in die Übertragungsleitung zurück reflektiert. Sind die Quellen-, Übertragungsleitungs- und Last-Impedanz gleich groß, so werden die Reflexionen beseitigt. Dieser Test misst den Gesamtwiderstand der beiden CAN-Datenleitungen und des angeschlossenen Abschlusswiderstandes.

Zum Testen, verfahren Sie bitte wie folgt:

1. Schalten Sie die Versorgungsspannungen aller angeschlossenen CAN-Knoten aus.
2. Messen Sie den DC-Widerstand zwischen CAN_H und CAN_L an den Enden des Netzwerks (1) (siehe obere Abbildung) und in der Mitte (sofern das Netzwerk-Kabel aus mehr als einem Leitungsabschnitt besteht). .

Der gemessene Wert sollte zwischen 50 Ω und 70 Ω liegen. Der gemessene Wert sollte an jedem Punkt des Netzwerks etwa gleich groß sein.

Liegt der ermittelte Wert unter 50 Ω, stellen Sie bitte sicher, dass:
- kein **Kurzschluss** zwischen den CAN_H- und CAN_L-Leitungen besteht
- **nicht mehr als zwei** Abschlusswiderstände angeschlossen sind
- die Transceiver der einzelnen Knoten nicht defekt sind.

Liegt der ermittelte Wert über 70 Ω, stellen Sie bitte sicher, dass:
- alle CAN_H- und CAN_L- Leitungen korrekt angeschlossen sind
- zwei Abschlusswiderstände von **je 120 Ω** an Ihr CAN-Netzwerk angeschlossen sind (einer an jedem Ende).

6.2 Erdung

CAN_GND des CAN-Netzwerks darf nur an einer einzigen Stelle mit dem Funktionserde-Potenzial (FE) verbunden sein. Dieser Test zeigt an, ob die Abschirmung an mehreren Stellen geerdet ist. Zum Testen verfahren Sie bitte wie folgt:

1. Trennen Sie die CAN_GND von dem Erdpotenzial (FE).
2. Messen Sie den DC-Widerstand zwischen CAN_GND und Erdpotenzial (siehe nebenstehende Abbildung).
3. Verbinden Sie CAN_GND mit dem Erdpotenzial.

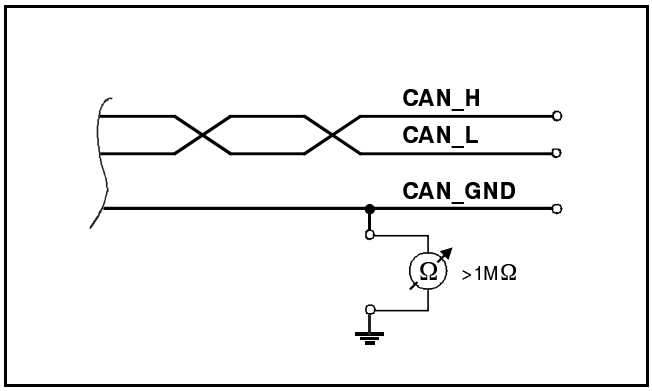


Abb. 19: Vereinfachtes Schaltbild Erdungsmessung

Der Widerstand sollte größer als ein 1 MΩ sein. Ist er kleiner, suchen Sie bitte nach zusätzlichen Erdungen der CAN_GND-Leitung.

6.3 Kurzschluss in der CAN-Verdrahtung

Ein CAN Bus kann möglicherweise auch dann noch Daten übertragen, wenn CAN_GND und CAN_L kurzgeschlossen sind, dadurch wird aber die Fehlerrate stark ansteigen. Stellen Sie sicher, dass zwischen CAN_GND und CAN_L kein Kurzschluss besteht!

6.4 CAN_H/CAN_L-Spannungen

Jeder Knoten verfügt über einen CAN-Transceiver, der differentielle Signale auf den Datenleitungen generiert. Ruht die Netzwerk-Kommunikation, betragen die CAN_H- und CAN_L-Spannungen etwa 2.5 V. Defekte Transceiver können diese Ruhespannungen verändern und die Netzwerk-Kommunikation unterbrechen.

Um auf defekte Transceiver zu testen, verfahren Sie bitte wie folgt:
1. Schalten Sie alle Versorgungsspannungen an.
2. Beenden sie jegliche Netzwerk-Kommunikation.
3. Messen Sie die DC-Spannung zwischen CAN_H und GND ② (siehe Abbildung auf vorhergehender Seite).
4. Messen Sie die DC-Spannung zwischen CAN_L und GND ③ (siehe Abbildung auf vorhergehender Seite).

Die gemessene Spannung sollte zwischen 2.0 V und 4.0 V liegen.
Ist die Spannung kleiner als 2.0 V oder größer als 4.0 V, ist es möglich, dass ein oder mehrere Knoten

defekte Transceiver haben. Bei einer Spannung die unter 2.0 V liegt, überprüfen Sie bitte den Anschluss der CAN_H- und CAN_L-Leitungen. Bei einer Spannung, die oberhalb von 4.0 V liegt, überprüfen Sie bitte auf überhöhte Spannung.

Um einen Knoten mit einem defekten Transceiver zu finden, überprüfen Sie bitte den Widerstand des CAN-Transceivers (siehe folgendes Kapitel).

6.5 CAN-Transceiver-Widerstands-Test

CAN-Transceiver verfügen über einen Schaltkreis, der CAN_H und CAN_L kontrolliert. Die Erfahrung zeigt, dass elektrische Beschädigung an einem oder beiden der Schaltkreise den Leckstrom in diesen Schaltkreisen erhöhen kann.

Um den Leckstrom durch die CAN-Schaltungen zu messen, benutzen Sie bitte ein Widerstands-messgerät und:

1. Schalten Sie den Koten ④ **aus** und trennen Sie ihn vom Netzwerk (siehe untere Abbildung).

2. Messen Sie den DC-Widerstand zwischen CAN_H und CAN_GND ⑤ (siehe untere Abbildung).

3. Messen Sie den DC-Widerstand zwischen CAN_L und CAN_GND ⑥ (siehe untere Abbildung).

Der gemessene Widerstand sollte für jedes Signal etwa 500 kΩ betragen. Liegt der Widerstand deutlich niedriger, ist der CAN-Transceiver möglicherweise defekt.
Ein weiterer Hinweis auf einen fehlerhaften CAN-Transceiver ist eine sehr hohe Abweichung der beiden gemessenen Eingangswiderstände (>> 200%).

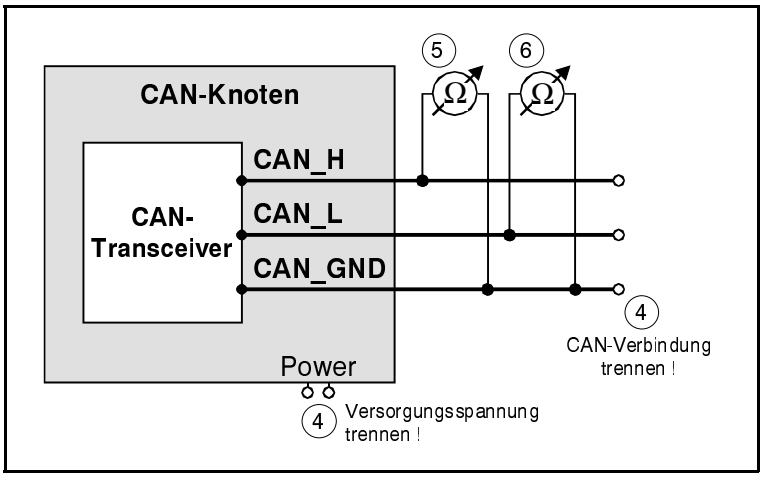


Abb. 20: Messung des Eingangswiderstandes des CAN-Transceivers

7. CANopen-Firmware

Dieses Kapitel enthält neben allgemeinen Grundlagen zum Thema CANopen die wichtigsten Informationen über die implementierten Funktionen.

Eine komplette CANopen-Beschreibung läge außerhalb des Rahmens dieses Handbuchs.
Weitergehende Informationen sind daher der CANopen - Dokumentation [1] zu entnehmen.

7.1 Begriffsdefinition

COB ...	Communication Object	
Emergency-Id...	Emergency Data Object	Notfalldaten
NMT...	Network Management (Master)	Netzwerkmanagement
SDO...	Service Data Object	Service-Daten (Parameter)
Sync...	Sync(frame) Telegramm	Synchronisations-Identifier

PDOs (Process Data Objects)
Die PDOs dienen zur Übertragung der Prozessdaten.
Im ‘Transmit’-PDO (TPDO) sendet das CAN-CBX-Modul Daten auf dem CANopen-Netz.
Im ‘Receive’-PDO (RPDO) empfängt das CAN-CBX-Modul Daten aus dem CANopen-Netz.

SDOs (Service Data Objects)
Die SDOs dienen zur Übertragung von modulinternen Konfigurations- und Parameterdaten. Im Gegensatz zu den PDOs werden die SDO-Nachrichten bestätigt. Einer Schreib- oder Leseanforderung auf ein Daten-Objekt folgt immer ein Bestätigungstelegramm mit einem Fehlerindex.

7.2 NMT-Boot-up

Das CAN-CBX-Modul kann mit dem in [1] beschriebenen ‘Minimal - Boot-up’ initialisiert werden.

Im einfachsten Fall reicht nach dem Einschalten ein Telegramm zum Umschalten aus dem Zustand *Pre-Operational* in den Zustand *Operational*. Dazu ist an den CAN-Identifier ‘0000_h’ z. B. das 2-Byte-Telegramm ‘01_h’, ‘00_h’ zu senden (= Start Remote Node all Devices).

7.3 Das CANopen-Objektverzeichnis

Das Objektverzeichnis ist eine (sortierte) Gruppierung von Objekten, auf die über das CAN-Netzwerk zugegriffen werden kann. Jedes Objekt in diesem Verzeichnis wird über einen 16-Bit-Index adressiert. In den Objektverzeichnissen wird der Index in hexadezimaler Form angegeben.

Der Index kann 16-Bit-Parameter nach der CANopen-Spezifikation [1] oder ein herstellerspezifischer Code sein. Anhand der höherwertigen Bits des Index wird festgelegt, zu welcher Objektklasse der Parameter gehört.

Zum Objektverzeichnis gehören unter anderem:

Index	Objekt
0001_h ... $009F_h$	Definition von Datentypen
1000_h ... $1FFF_h$	Communication Profile Area
2000_h ... $5FFF_h$	Manufacturer Specific Profile Area
6000_h ... $9FFF_h$	Standardised Device Profile Area
$A000_h$ ... $FFFF_h$	reserviert

7.4 Communication Parameter

Die Communication Parameter der PDOs (gemäß [1]) werden als SDOs Service Data Objects) auf der ID '**600_h + NodeID**' übertragen (Request). Der Empfänger quittiert die Parameter auf der ID '**580_h + NodeID**' (Response).

Die **Node-ID** (Modul-Nr.) wird über die Kodierschalter Low und High konfiguriert. Die möglichen Einstellungen sind in Kapitel " Einstellung der Node-ID über die Kodierschalter" auf Seite 19 beschrieben.

7.4.1 Zugriff auf das Objektverzeichnis über SDOs

Die SDOs (Service Data Objects) dienen zum Zugriff auf das Objektverzeichnis eines Gerätes.
Ein SDO stellt daher einen 'Kanal' zum Zugriff auf die Parameter des Gerätes dar. Der Zugriff über diesen Kanal ist im Zustand *operational* und *pre-operational* möglich.

Dieses Kapitel beschreibt nicht alle möglichen sondern nur einige wichtige Arten des Zugriffs im CAN-CBX-Modul.
Definitionen der Zugriffsmodi können [1] entnommen werden.

Ein SDO ist wie folgt aufgebaut:

Identifier	Befehls-code	Index		Sub-Index	LSB	Datenfeld	MSB
		(low)	(high)				

Beispiel:

600_h+ NodeID	23_h (write)	00_h	14_h (Index=1400_h) (Receive-PDO-Comm-Para)	01_h (COB-def.)	$7F_h$	04_h	00_h	00_h COB Node ID = $047F_h$

Beschreibung der SDOs:

Identifier
Die Parameter werden auf der ID '600_h + NodeID' übertragen (Request).
Der Empfänger quittiert die Parameter auf der ID '580_h + NodeID' (Response).

Befehlscode
Der gesendete Befehlscode setzt sich unter anderem aus dem Command Specifier und der Länge zusammen. Häufig benötigte Kombinationen sind z.B.:

$40_h = 64_{dez}$: Read Request, d.h. ein Parameter soll gelesen werden
$23_h = 35_{dez}$: Write Request mit 32 Bit Daten, d.h. ein Parameter soll gesetzt werden

Das CAN-CBX-Modul antworten auf jedes empfangene Telegramm mit einem Antworttelegramm. Das

Antworttelegramm kann folgende Befehlscodes enthalten:

$43_h = 67_{dez}$: Read Response mit 32 Bit Daten, dieses Telegramm enthält den gewünschten Parameter
$60_h = 96_{dez}$: Write Response, d.h. ein Parameter wurde erfolgreich gesetzt
$80_h = 128_{dez}$: Error Response, d.h. das CAN-CBX-Modul meldet einen Fehler.

Häufig verwendete Befehlscodes
Die folgende Tabelle gibt eine Übersicht über häufig verwendete Befehlscodes. Die Kommando-Frames müssen immer 8 Daten-Bytes beinhalten. Hinweise zur Syntax und weitere Befehlscodes sind in [1] nachzulesen.

Kommando	für Anzahl Datenbytes	Befehlscode
Write Request (Initiate Domain Download)	1 2 3 4	$2F_h$ $2B_h$ 27_h 23_h
Write Response (Initiate Domain Download)	-	60_h
Read Request (Initiate Domain Upload)	-	40_h
Read Response (Initiate Domain Upload)	1 2 3 4	$4F_h$ $4B_h$ 47_h 43_h
Error Response (Abort Domain Transfer)	-	80_h

Index, Sub-Index
Der Index und der Sub-Index werden in den Kapiteln “Device Profile Area” und “Manufacturer Specific Profile Area” dieses Handbuches beschrieben.

Datenfeld
Das maximal 4 Byte lange Datenfeld ist grundsätzlich nach der Regel ‘niederwertiges Byte zuerst, höherwertiges Byte zuletzt’ aufgebaut. Dabei steht das niederwertige Byte immer in ‘Data 1’, bei 16-Bit-Werten steht das höchstwertige Byte (Bits 8...15) in ‘Data 2’, und bei 32-Bit-Werten steht das MSB (Bits 24...31) in ‘Data 4’.

Fehlercodes des SDO-Transfers

Die folgenden SDO Abort Codes (Fehlercodes) können zur Anwendung kommen (gemäß [1]):

SDO Abort Codes	Erläuterung
05040001_h	falscher Command Specifier
06010002_h	Schreibzugriff ist hier falsch
06020000_h	falscher Index
06040041_h	Objekt kann nicht auf PDO gemapped werden
06060000_h	kein Zugriff wegen Hardware Fehler
06070010_h	falsche Anzahl Daten-Bytes
06070012_h	Länge des Service-Parameters ist zu groß
06070013_h	Länge des Service-Parameters ist zu klein
06090011_h	falscher Sub-Index
06090030_h	gesendeter Parameter außerhalb des zulässigen Wertebereiches
08000000_h	nicht definierte Fehlerursache
08000020_h	Daten können nicht übertragen oder gespeichert werden
08000022_h	wegen des aktuellen Device States können Daten nicht übertragen oder gespeichert werden
08000024_h	Zugriff auf Flash fehlgeschlagen

7.5 Übersicht der verwendeten CANopen-Identifier

Funktion	Identifier	Bemerkungen
Netzwerkmanagement	0	NMT
SYNC	80_h	Sync an alle, (konfigurierbar über Objekt 1005_h)
Emergency Message	80_h + *NodeID*	konfigurierbar über Objekt 1014_h
TPDO2	280_h + *NodeID*	PDO2 vom CAN-CBX-AI420 (Tx) (Objekt 1801_h)
TPDO3	380_h + *NodeID*	PDO3 vom CAN-CBX-AI420 (Tx) (Objekt 1802_h)
Client-SDO	580_h + *Node-ID*	SDO vom CAN-CBX-AI420 (Tx)
Server-SDO	600_h + *Node-ID*	SDO zum CAN-CBX-AI420 (Rx)
Node Guarding	700_h + *NodeID*	konfigurierbar über Objekt $100E_h$

NodeID: Eingestellte CANopen-Adresse [1_h...$7F_h$]

7.5.1 Einstellung der COB-ID

Die COB-IDs die einstellbar sind (außer der von SYNC), werden zunächst von der Einstellung der Node-ID über die Kodierschalter (siehe Seite 18) abgeleitet. Wurden die COB-IDs über SDO beschrieben, gilt diese Einstellung, auch wenn der Schalter danach auf eine andere Node-ID eingestellt wird.
Um die Default-COB-ID wieder vom Schalter zu übernehmen, müssen die *Comm defaults* oder alle Defaults geladen werden (Objekt 1011_h).

7.6 Default-PDO-Belegung

Die PDOs (Process Data Objects) dienen zur Übertragung der Prozessdaten. Das PDO-Mapping ist veränderbar. Die folgenden Tabellen zeigen das Default-Mapping bei Auslieferung des Moduls:

PDO	CAN-Identifier	Länge	Übertragungsrichtung	Default-Belegung
TPDO1	n.a.	n.a.	n.a.	TPDO1 wird nicht verwendet
TPDO2	280_h + Node-ID	8 Byte	vom CAN-CBX-AI420 (Sende-PDO)	A/D-Werte Kanal 1 und 2 als 32 Bit-Werte
TPDO3	380_h + Node-ID	8 Byte	vom CAN-CBX-AI420 (Sende-PDO)	A/D-Werte Kanal 3 und 4 als 32 Bit-Werte

TPDO2 (CAN-CBX-AI420 ->)

CAN-Identifier: 280_h + Node-ID

Byte	0	1	2	3	4	5	6	7
Parameter	*Read_Analog_Input_32_1*				*Read_Analog_Input_32_2*			

TPDO3 (CAN-CBX-AI420 ->)

CAN-Identifier: 380_h + Node-ID

Byte	0	1	2	3	4	5	6	7
Parameter	*Read_Analog_Input_32_3*				*Read_Analog_Input_32_4*			

Parameter-Beschreibung:

Bezeichner	Beschreibung	Datentyp	siehe Seite
Read_Analog_Input_32_x	Lesen des analogen Ausgangs x (x = 1-4), siehe Objekt 6402_h	integer 16	81

7.7 Lesen der Analogwerte

7.7.1 Meldung der analogen Eingänge

Die Übertragungsarten für die analogen Eingänge können unterschieden werden in:

- *azyklisch, synchron*: Die Sendung erfolgt nach dem Empfang einer SYNC-Message (PDO - Übertragungstyp 0), wenn sich die Daten geändert haben.
- *zyklisch, synchron*: Die Sendung erfolgt, nachdem jeweils eine bestimmte Anzahl SYNC-Messages empfangen wurden (PDO-Übertragungstyp 1...240).
- *synchron, remote request*: Der Zustand der Eingänge wird mit jeder SYNC-Message gespeichert und nach dem Empfang eines RTR-Frames gesendet (PDO-Übertragungstyp 252).
- *asynchron, remote request*: Nach dem Empfang eines RTR-Frames wird der letzte ermittelte Zustand der Eingänge gesendet (PDO-Übertragungstyp 253).
- *ereignisgesteuert, asynchron*: Die Sendung erfolgt, wenn sich der Zustand bestimmter analoger Eingänge ändert (PDO-Übertragungstyp 254, 255).

7.7.2 Unterstützte Übertragungsarten nach DS-301

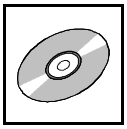
Transmission Type	PDO-Transmission					unterstützt von CAN-CBX-AI420
	cyclic	acyclic	synchro-nous	asynchro-nous	RTR	
0		X	X			x
1...240	X		X			x
241...251	reserviert					-
252			X		X	x
253				X	X	x
254				X	X	x
255				X	X	x

7.8 Communication Profile Area

7.8.1 Verwendete Bezeichnungen und Abkürzungen

Folgende Bezeichnungen werden in den Tabellen zur Beschreibung der Kommunikationsparameter verwendet:

PDO-Mappable	Für diesen Sub-Index des PDOs ist PDO-Mapping möglich
Save to EEPROM	Der Wert dieses Parameters wird im lokalen EEPROM gespeichert, wenn das Kommando ‘save’ aufgerufen wird (siehe Seite 61)
Datentyp	Datentyp (z.B. unsigned 8, unsigned 32)
Zugriff	Zugelassene Zugriffsarten auf diesen Parameter ro... read only Dieser Parameter kann nur gelesen werden. Schreibzugriffe führen zu einer Fehlermeldung. const.... constant Dieser Parameter kann vom Anwender nicht verändert werden. Er ist Lesbar. Schreibzugriffe führen zu einer Fehlermeldung. rw... read / write Dieser Parameter kann gelesen oder gesetzt werden.
Wertebereich	Wertebereich des Parameters
Default-Wert	Grundeinstellung des Parameters
Name	Name und Kurzbeschreibung des Parameters

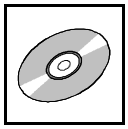


7.9 Implementierte CANopen-Objekte

Eine detaillierte Beschreibung der Objekte ist in der CiA DS-301 nachzulesen.

7.9.1 Übersicht der Communication Profile Objekte mit Produktspezifischen Werten

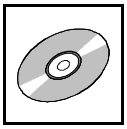
Index	Sub-Index	Beschreibung	Datentyp	Zugriff	Default-Wert
1000_h	-	Device Type	unsigned 32	ro	00040191_h
1001_h	-	Error Register	unsigned 8	ro	Unterstützte Fehler Bits: 0: generic 2: voltage 4: communication error
1003_h	A_h	Pre-Defined-Error-Field	unsigned 32	rw	00_h
1005_h	-	COB-ID-Sync	unsigned 32	rw	80_h
1006_h	-	Communication Cycle Period	unsigned 32	rw	00_h
1008_h	-	Manufacturer Device Name	visible string	ro	"CAN-CBX-AI420"
1009_h	-	Manufacturer Hardware Version	visible string	ro	x.yy (Versionsabhängig)
$100A_h$	-	Manufacturer Software Version	visible string	ro	x.yy (Versionsabhängig)
$100C_h$	-	Guard Time	unsigned 16	rw	0000_h
$100D_h$	-	Life Time Factor	unsigned 8	rw	00_h
$100E_h$	-	Node Guarding Identifier	unsigned 32	rw	Node-ID + 700_h
1010_h	4	Store Parameter	unsigned 32	rw	Parameter, die gespeichert bzw. geladen werden können: 1005_h ... 1029_h, 6421_h ... 6426_h, 2310_h, 2311_h, 2401_h ... 2405_h
1011_h	4	Restore Parameter	unsigned 32	rw	
1014_h	-	COB-ID Emergency Object	unsigned 32	rw	80_h + Node-ID
1015_h	-	Inhibit Time EMCY	unsigned 16	rw	00_h
1016_h	1	Consumer Heartbeat Time	unsigned 32	rw	00_h
1017_h	-	Producer Heartbeat Time	unsigned 16	rw	00_h
1018_h	4	Identity Object	unsigned 32	ro	Vendor Id: 00000017_h Prod. Code: 23030002_h
1019_h	-	Synchronous Counter Overflow	unsigned 8	rw	00_h
1020_h	0-2	Verify Configuration	unsigned 32	ro	n.a.
1029_h	3	Error Behaviour	unsigned 8	ro	00_h



Implementierte CANopen Objekte

Index	Sub-Index	Beschreibung	Datentyp	Zugriff
1801_h	5	2. Transmit PDO-Parameter	PDO CommPar (20_h)	rw
1802_h	5	3. Transmit PDO-Parameter	PDO CommPar (20_h)	rw
$1A01_h$	2	2. Transmit PDO-Mapping	PDO Mappping (21_h)	rw
$1A02_h$	2	3. Transmit PDO-Mapping	PDO Mappping (21_h)	rw

Index	Sub-Index (max.)	Beschreibung	Datentyp	Zugriff	Produktspezifische Eigenschaften
$1F80_h$	-	NMT startup	unsigned 32	rw	default: 2 (autostart disabled)
$1F91_h$	1	Self starting nodes timing parameters	unsigned 16	rw	default: 64_h (= 100 ms)



7.9.2 Device Type (1000_h)

INDEX	**1000_h**
Name	*device type*
Datentyp	unsigned 32
Zugriff	ro
Default-Wert	siehe Kapitel 7.9.1, Seite 49

Beispiel: Auslesen des Device Type

Der CANopen Master sendet unter dem Identifier '603_h' (600_h + Node-ID) den Read Request an das CAN-CBX-Modul mit der Modul-Nr. 3 (Node-ID=3_h):

ID	RTR	LEN	DATA							
			1	2	3	4	5	6	7	8
603_h	0_h	8_h	40_h	00_h	10_h	00_h	00_h	00_h	00_h	00_h
			Read Request	Index=1000_h		Sub Index				

Das CAN-CBX-Modul Nr. 3 antwortet dem Master anhand der Read Response unter dem Identifier '583_h' (580_h + Node-ID) mit dem Wert des Device Type:

ID	RTR	LEN	DATA							
			1	2	3	4	5	6	7	8
583_h	0_h	8_h	43_h	00_h	10_h	00_h	**94_h**	**01_h**	**02_h**	**00_h**
			Read Response	Index=1000_h		Sub Index	Beispiel hier: Device Profile DS 404		Beispiel hier: Input	

ausgegebener Wert des Device Types in diesem Beispiel: 0002.0194_h.
Der Wert des Device Types für dieses CBX-Modul ist in Kapitel 7.9.1 auf Seite 49 abgedruckt.

Das Datenfeld ist grundsätzlich nach der Regel 'niederwertiges Byte zuerst, höherwertiges Byte zuletzt' aufgebaut (siehe Seite 43, Datenfeld).

7.9.3 Error Register (1001_h)

Das CAN-CBX-Modul nutzt das Error-Register, um Fehlermeldungen anzuzeigen.

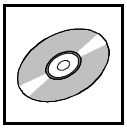
INDEX	**1001_h**
Name	*error register*
Datentyp	unsigned 8
Zugriff	ro
Default-Wert	0

Bits des Error-Registers:

Bit	Bedeutung
0	*generic*
1	*current*
2	*voltage*
3	*temperature*
4	*communication error* (overrun, error state)
5	*device profile*
6	reserved
7	*manufacturer*

Eine Liste der von diesem Modul unterstützten Fehler-Bits ist in Kapitel 7.9.1 auf Seite 49 abgedruckt.

Die nicht unterstützten Bits werden immer als ‘0’ zurückgegeben. Liegt ein Fehler vor, so ist das entsprechende Fehler-Bit auf ‘1’ gesetzt.



7.9.4 Pre-defined Error Field (1003_h)

INDEX	**1003_h**
Name	*pre-defined error field*
Datentyp	unsigned 32
Zugriff	ro
Default-Wert	No

Im *pre-defined error field* wird eine Liste der zuletzt aufgetretenen Fehler gespeichert. Der Sub-Index 0 enthält die aktuelle Anzahl der in der Liste gespeicherten Fehler. Unter Sub-Index 1 wird der zuletzt aufgetretene Fehler abgelegt. Tritt ein neuer Fehler auf, so wird der vorhergehende Fehler auf Sub-Index 2 gespeichert und der neue Fehler unter Sub-Index 1 usw.. So entsteht eine Liste mit der Fehler-Historie.

Der Fehlerspeicher ist wie ein Ringspeicher aufgebaut. Ist er gefüllt, wird der älteste Eintrag gelöscht um Platz für den aktuellen Eintrag zu schaffen.

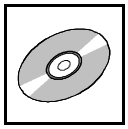
Dieses Modul unterstützt maximal 10 Fehlereinträge. Beim Eintreten des 11. Fehlers wird der älteste Fehlereintrag gelöscht. Zum Löschen der gesamten Fehler-Liste ist der Sub-Index ‘0’ auf ‘0’ zu setzen. Dies ist der einzig zulässige Schreibzugriff auf das Objekt.

Mit jedem neuen Eintrag in die Liste sendet das Modul ein **Emergency-Frame**, um den Fehler mitzuteilen.

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
1003_h	0	*no_of_errors_in_list*	0, 1...A_h	-	unsigned 8	rw
	1	*error-code n*	0...$FFFFFFFF_h$	-	unsigned 32	ro
	2	*error-code (n-1)*	0...$FFFFFFFF_h$	-	unsigned 32	ro
	:	:	:	:	:	ro
	A_h	*error-code (n-9)*	0...$FFFFFFFF_h$	-	unsigned 32	ro

Bedeutung der Variablen:

no_of_errors_in_list - enthält die aktuelle Anzahl der in der Liste eingetragenen Fehler-Codes
n = Nummer des zuletzt aufgetretenen Fehlers
- zum Löschen der Fehlerliste ist diese Variable auf ‘0’ zu setzen
- ist *no_of_errors_in_list* ≠ 0, so wird das Error-Register (Object 1001_h) gesetzt



Implementierte CANopen Objekte

error-code x Der 32-Bit lange Fehler-Code setzt sich aus dem in [1] aufgeführten CANopen-Emergency-Error-Code und den von esd definierten Fehler-Codes (Manufacturer-Specific Error Field) zusammen.

Bit:	31 16	15 0
Inhalt:	*manufacturer-specific error field*	*emergency-error-code*

manufacturer-specific error field: immer '00', es sei denn
emergency-error-code = 2300_h (siehe unten)

emergency-error-code: Es werden die folgenden Fehler-Codes unterstützt:

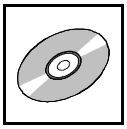
- 8110_h - CAN Overrun Error
  - Sample-Rate ist so hoch eingestellt, dass die Firmware nicht in der Lage ist, alle Daten auf dem CAN-Bus zu senden
- 8120_h - CAN in Error Passive Mode
- 8130_h - Lifeguard Error / Heartbeat Error
- 8140_h - Recovered from "Bus Off"
- 8240_h - Unexpected SYNC data lenght
- 6000_h - Software-Fehler:
  -EEPROM Checksum Fehler (keine Übertragung dieser Message als Emergency)
- 6110_h - Internal Software Error
  z.B.:
  - gespeicherte Daten hatten ungültige Checksumme und Default-Werte wurden geladen
  - interner Watchdog hat ausgelöst
- $FF10_h$ - Data lost (A/D-Daten Overflow)
- 5000_h - Hardware Fehler (z.B. A/D-Wandler defekt)
- 5030_h - Sensor-Fehler

Emergency Message

Die Daten des vom CAN-CBX Modul gesendeten Emergency-Frames sind wie folgt aufgebaut:

Byte:	0	1	2	3	4	5	6	7
Inhalt:	*emergency-error-code* (siehe oben)		*error-register* 1001_h	*no_of_errors_in_list* $1003,00_h$	-			

Eine Emergency Message wird beim ersten Auftreten eines Fehlers gesendet. Tritt der selbe Fehler ein weiteres Mal auf, wird keine weitere Emergency Message gesendet.
Sobald eine Fehlermeldung zurückgenommen wird, wird wiederum eine Emergency Message gesendet.



7.9.5 COB-ID of SYNC-Message (1005_h)

INDEX	**1005_h**
Name	*COB-ID SYNC message*
Datentyp	unsigned 32
Zugriff	rw
Default-Wert	siehe Kapitel 7.9.1, Seite 49

Struktur des Parameters:

Bit-No.	Wert	Bedeutung
31 (MSB)	-	do not care
30	0/1	0: keine SYNC Erzeugung 1: Modul erzeugt SYNC
29	0	immer 0, da 11-Bit-ID
28...11	0	immer 0, da keine 29-Bit-ID unterstützt werden
10...0 (LSB)	x	Bit 0...10 der SYNC-COB-ID

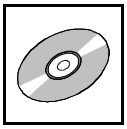
Der Identifier kann Werte zwischen 0...$7FF_h$ annehmen.

7.9.6 Communication Cycle Period (1006_h)

INDEX	**1006_h**
Name	*Communication Cylcle Period*
Datentyp	unsigned 32
Zugriffsart	rw
Default-Wert	0 µs

Wertebereich des Parameters:

Wert	Bedeutung
0	Übertragung von SYNC-Nachrichten ausgesetzt
1...$FFFFFFFF_h$	Zyklus-Zeit in Mikrosekunden (Beispiel: 4E20h entspricht einer Zykluszeit von 20000µs = 20 ms)



7.9.7 Manufacturer Device Name (1008_h)

INDEX	**1008_h**
Name	*manufacturer device name*
Datentyp	visible string
Default-Wert	siehe Kapitel 7.9.1, Seite 49

Eine ausführliche Beschreibung des SDO Uploads ist [1] zu entnehmen.

7.9.8 Manufacturer Hardware Version (1009_h)

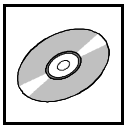
INDEX	**1009_h**
Name	*manufacturer hardware version*
Datentyp	visible string
Default-Wert	string: z.B. ‘1.0’ (Versionsabhängig)

Das Lesen der Hardware-Version erfolgt ähnlich wie das Lesen des Manufacturer Device Names über das SDO Upload Protokoll. Eine ausführliche Beschreibung des Uploads ist [1] zu entnehmen.

7.9.9 Manufacturer Software Version ($100A_h$)

INDEX	**$100A_h$**
Name	*manufacturer software version*
Datentyp	visible string
Default-Wert	string: z.B.: ‘1.2’ ’ (Versionsabhängig)

Das Lesen der Hardware-Version erfolgt ähnlich wie das Lesen des Manufacturer Device Names über das SDO Upload Protokoll. Eine ausführliche Beschreibung des Uploads ist [1] zu entnehmen.



7.9.10 Guard Time ($100C_h$) und Life Time Factor ($100D_h$)

Das CAN-CBX-Modul unterstützt das Node-Guarding oder alternativ die Heartbeat-Funktion (siehe Seite 69).

Hinweis:
Nach den CiA-Empfehlungen sollte bevorzugt die Heartbeat-Funktion verwendet werden. Das Node-Guarding sollte nur noch für bestehende Systeme eingesetzt werden!

Guard Time und Life Time Factor werden zusammen ausgewertet. Die Multiplikation beider Werte ergibt die Life Time. Die Guard Time wird in Millisekunden angegeben.

INDEX	**$100C_h$**
Name	*guard time*
Datentyp	unsigned 16
Zugriff	rw
Default-Wert	0 [ms]
Minimaler Wert	0
Maximaler Wert	$FFFF_h$ (65,535 s)

INDEX	**$100D_h$**
Name	*life time factor*
Datentyp	unsigned 8
Zugriff	rw
Default-Wert	0
Minimaler Wert	0
Maximaler Wert	FF_h

7.9.11 Node Guarding Identifier ($100E_h$)

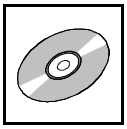
Das Modul unterstützt nur 11-Bit-Identifier.

INDEX	**$100E_h$**
Name	*node guarding identifier*
Datentyp	unsigned 32
Zugriff	rw
Default-Wert	700_h + Node-ID

Struktur des Parameters *node guarding identifier* :

Bit-No.	Bedeutung
31 (MSB) 30	reserved
29...11	immer 0, da keine 29-Bit-ID unterstützt werden
10...0 (LSB)	Bit 0...10 des Node Guarding Identifiers

Der Identifier kann Werte zwischen 1...$7FF_h$ annehmen.



7.9.12 Store Parameters (1010_h)

INDEX	**1010_h**
Name	*store parameters*
Datentyp	unsigned 32

Mit diesem Kommando werden die Parameter im “nicht-flüchtigen” Speicher, hier im EEPROM, gespeichert. Dazu werden die unten aufgeführten Parameter Gruppen unterschieden.
Die eingestellten Parameter sind sofort nach der Übergabe aktiv. Das ‘nichtflüchtige’ Speichern der Parameter erfolgt jedoch nicht automatisch. Es muss über einen Schreibzugriff auf das Objekt 1010_h veranlasst werden und sollte nur erfolgen, wenn sich das Modul im Zustand *pre-operational* befindet. Beim Schreiben des Index muss die unten angegebene Byte-Folge gesendet werden.

Das Lesen des Index gibt Informationen über die implementierten Speicher-Funktionen zurück (siehe [1]).

Index	Sub-Index	Beschreibung	Wertebereich	Datentyp	Zugriff
1010_h	0	*number_of_entries*	4	unsigned 8	ro
	1	*save_all_parameters* (Objekte 1000_h ... $9FFF_h$)	no default, write: $65\ 76\ 61\ 73_h$ (= ASCII: ‘e’ ‘v’ ‘a’ ‘s’)	unsigned 32	rw
	2	*save_communication_parameter* (Objekte 1000_h ... $1FFF_h$)		unsigned 32	rw
	3	*save_application_parameter* (Objekte 6000_h ... $9FFF_h$)		unsigned 32	rw
	4	*save_manufacturer_parameter* (Objekte 2000_h ... $5FFF_h$)		unsigned 32	rw

Belegung der Variablen

save all parameters

speichert die Parameter aller Objekte (sofern vorhanden), deren Zugriffsrecht Read/Write (rw) ist.

save_communication_parameter

speichert alle Communication Parameter der Objekte (1000_h ... $1FFF_h$, sofern vorhanden), deren Zugriffsrecht Read/Write (rw) ist (hier z.B. 1005_h ... 1029_h).

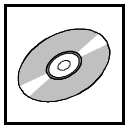
save_application_parameter

speichert alle Application Parameter der Objekte (Objekte 6000_h ... $9FFF_h$, sofern vorhanden), deren Zugriffsrecht Read/Write (rw) ist (hier z.B. $6xxx_h$).

save_manufacturer_parameter

speichert alle Manufacturer Parameter der Objekte (Objekte 2000_h ... $5FFF_h$, sofern vorhanden), deren Zugriffsrecht Read/Write (rw) ist (hier z.B. $2xxx_h$).

Der Speicher-Modus wird über den Inhalt dieses Objektes angezeigt:



Implementierte CANopen Objekte

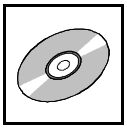
Bit 1 des Objekts 1010_h, Sub-Index 1 ist nicht gesetzt, d.h. das CAN-CBX-Modul speichert die Konfiguration nicht ‘automatisch’, sondern muss durch Schreiben der Zeichenkette ‘save’ (73_h 61_h 76_h 65_h, Reihenfolge aus CAN-Telegramm) in das Objekt 1010_h, Sub-Index 1-4 extra dazu veranlasst werden.

Beim Lesen eines Sub-Index liefert das CANopen Device Informationen über die Speicher-Funktionalität im folgenden Format:

Bit:	31 2	1	0
Inhalt:	reserved	auto	cmd
	0	0	1
	MSB		LSB

Bit	Wert	Beschreibung
auto	0 1	das CAN-CBX-Modul speichert die Parameter **nicht** autonom das CAN-CBX-Modul speichert die Parameter autonom
cmd	0 1	das CAN-CBX-Modul speichert die Parameter **nicht** auf Befehl das CAN-CBX-Modul speichert die Parameter auf Befehl

Autonomes Speichern bedeutet, dass die Parameter “nicht-flüchtig” ohne weitere Abfrage gespeichert werden.



7.9.13 Restore Default Parameters (1011_h)

INDEX	**1011_h**
Name	*restore default parameters*
Datentyp	unsigned 32

Mit diesem Kommando werden die bei der Auslieferung aktiven Default-Parameter wieder geladen. Dazu werden die unten aufgeführten Parameter Gruppen unterschieden.
Alle individuellen Einstellungen, die im EEPROM gespeichert worden sind, gehen verloren, das betrifft auch Kalibrierungswerte.
Die Default-Parameter sind erst nach einem Reset aktiv. Das Aktivieren der Parameter muss über einen Schreibzugriff veranlasst werden.
Beim Schreiben des Index muss die unten angegebene Byte-Folge gesendet werden. Das Lesen des Index gibt Informationen über die implementierten Restore-Funktionen zurück (siehe [1])

Index	Sub-Index	Beschreibung	Wertebereich	Datentyp	Zugriff
1011_h	0	*number_of_entries*	4	unsigned 8	ro
	1	*restore_all_default_parameters* (1000_h ... $9FFF_h$)	no default, write: $64\ 61\ 6F\ 6C_h$ (= ASCII: ‘d’ ‘a’ ‘o’ ‘l’)	unsigned 32	rw
	2	*restore_communication_default_parameter* (1000_h ... $1FFF_h$)		unsigned 32	rw
	3	*restore_application_default_parameter* (6000_h ... $9FFF_h$)		unsigned 32	rw
	4	*restore_manufacturer_default_parameter* (2000_h ... $5FFF_h$)		unsigned 32	rw

Belegung der Variablen

restore_all_default_parameters
lädt die Default-Werte aller Parameter aller Objekte (sofern vorhanden).

restore_communication_default_parameter
lädt die Default-Werte aller Communication Parameter der Objekte (1000_h ... $1FFF_h$, sofern vorhanden, z.B. 1005_h ... 1029_h).

restore_application_default_parameter
lädt die Default-Werte der Application Parameter der Objekte (6000_h ... $9FFF_h$, sofern vorhanden, z.B. $6xxx_h$).

restore_manufacturer_default_parameter
lädt die Default-Werte aller Manufacturer Parameter der Objekte (2000_h ... $5FFF_h$, sofern vorhanden, hier z.B. $2xxx_h$).

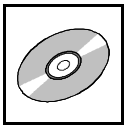
Bit 0 des Objekts 1011_h, Sub-Index 1 ist gesetzt, d.h. das CAN-CBX-Modul lädt die Standardwerte

durch Schreiben der Zeichenkette ‘load’ (64_h 61_h $6F_h$ $6C_h$, Reihenfolge aus CAN-Telegramm) in das Objekt 1011_h, Sub-Index 1-4.

Beim Lesen eines Sub-Index liefert das CANopen Device Informationen über die Restore-Funktionalität im folgenden Format:

Bit:	31 1	0
Inhalt:	reserved	cmd
	0	1
	MSB	LSB

Bit	Wert	Beschreibung
cmd	0	die Default-Parameter des CAN-CBX-Moduls werden **nicht** geladen
	1	die Default-Parameter des CAN-CBX-Moduls werden geladen



7.9.14 COB_ID Emergency Message (1014_h)

INDEX	**1014_h**
Name	*COB-ID emergency object*
Datentyp	unsigned 32
Default-Wert	80_h + Node-ID

Dieses Objekt bestimmt die COB-ID der Emergency Message (EMCY).

Die Struktur des Objektes ist in der folgenden Tabelle beschrieben:

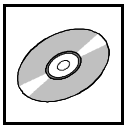
Bit-No.	Wert	Bedeutung
31 (MSB)	0/1	0: EMCY existiert / ist gültig 1: kein EMCY / EMCY ist nicht gültig
30	0	reserviert (immer 0)
29	0	immer 0, da 11-Bit ID
28...11	0	immer 0, da keine 29-Bit-ID unterstützt werden
10...0 (LSB)	x	Bit 0...10 des COB-ID

Der Identifier kann Werte zwischen 0...$7FF_h$ annehmen.

7.9.15 Inhibit Time EMCY (1015_h)

INDEX	**1015_h**
Name	*inhibit_time_emergency*
Datentyp	unsigned 16
Zugriffsart	rw
Wertebereich	0...$FFFF_h$
Default-Wert	0

Mit diesem Objekt kann die *Inhibit Time* für die Emergency-Nachricht festgelegt werden. Die Zeit wird als ein Vielfaches von 100 µs angegeben.



7.9.16 Consumer Heartbeat Time (1016_h)

INDEX	**1016_h**
Name	*consumer heartbeat time*
Datentyp	unsigned 32
Default-Wert	No

Zur gegenseitigen Überwachung der CANopen-Module (insbesondere zum Erkennen von Verbindungsausfällen) kann die Heartbeat-Funktion genutzt werden. Im Gegensatz zum Node Guarding/Life Guarding kommt die Heartbeat-Funktion ohne RTR-Frames aus.

Funktionsweise:
Ein Modul, der sog. Heartbeat-Producer sendet auf dem Node-Guarding-Identifier (siehe Object $100E_h$) zyklisch eine Heartbeat-Nachricht auf dem CAN-Bus. Ein oder mehrere Heartbeat-Consumer empfangen die Nachricht. Die Nachricht muss innerhalb der auf dem Heartbeat-Consumer gespeicherten Heartbeat-Time empfangen werden, sonst wird auf dem Heartbeat-Consumer-Modul ein Heartbeat-Event ausgelöst. Auf dem CAN-CBX-Modul löst der Heartbeat-Event einen Heartbeat-Error aus.
Jedes Modul kann Heartbeat-Producer und Heartbeat-Consumer sein. Das CAN-CBX-Modul unterstützt in einem CAN-Netz maximal einen Heartbeat-Producer.

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
1016_h	0	*number_of_entries*	1	1	unsigned 8	ro
	1	*consumer-heartbeat_time*	$0...007FFFFF_h$	0	unsigned 32	rw

Bedeutung der Variablen *consumer-heartbeat_time_x*:

	consumer-heartbeat_time_x		
Bit	31 24	23 16	15 0
Belegung	reserved (immer '0')	*Node-ID* (unsigned 8)	*heartbeat_time* (unsigned 16)

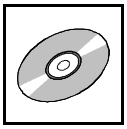
Node-ID Node-Id des zu überwachenden Heartbeat-Producers.

heartbeat_time Zeit in [ms], innerhalb der sich der zu überwachende Heartbeat-Producer auf dem Node-Guarding-ID melden muss, damit kein Heartbeat-Event ausgelöst wird. Diese Consumer-Heartbeat-Time muss immer größer sein als die Producer-Heartbeat-Time.

Beispiel:

consumer-heartbeat_time = 0031 $03E8_h$

=> *Node-ID* = 31_h = 49_d
=> *heartbeat time* = $3E8_h$ = 1000_d => 1 s



7.9.17 Producer Heartbeat Time (1017_h)

INDEX	**1017_h**
Name	*producer heartbeat time*
Datentyp	unsigned 16
Default-Wert	0 ms

Hier wird die Zeit eingetragen, mit der das CAN-CBX-Modul zyklisch einen Heartbeat-Frame auf dem Node-Guarding-ID sendet.

Wird für die Producer-Heartbeat-Time ein Wert größer Null eingesetzt, so ist sie aktiv und unterbindet das Node-/ Life-Guarding (siehe Seite 59).
Wird die Producer-Heartbeat-Time auf ‘0’ gesetzt, so wird das Senden des Heartbeats durch dieses Modul beendet.

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
1017_h	0	*producer-heartbeat_time*	0...$FFFF_h$	0 ms	unsigned 16	rw

producer-heartbeat_time Zykluszeit in [ms] des Heartbeat-Producers zum Senden des Heartbeats auf dem Node-Guarding-ID (siehe Object $100E_h$).
Die Consumer-Heartbeat-Time der überwachenden Module muss immer größer sein als die Producer-Heartbeat-Time dieses Heartbeat-sendenden Moduls.

7.9.18 Identity Object (1018_h)

INDEX	**1018_h**
Name	*identity object*
Datentyp	unsigned 32
Default-Wert	No

Dieses Objekt enthält allgemeine Informationen zum CAN-Modul.

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
1018_h	0	*no_of_entries*	4	4	unsigned 8	ro
	1	*vendor_id*	0...$FFFFFFFF_h$	0000 0017_h	unsigned 32	ro
	2	*product_code*	0...$FFFFFFFF_h$	siehe Kapitel 7.9.1 ab Seite 49	unsigned 32	ro
	3	*revision_number*	0...$FFFFFFFF_h$	0	unsigned 32	ro
	4	*serial_number*	0...$FFFFFFFF_h$	-	unsigned 32	ro

Bedeutung der Variablen:

vendor_id Diese Variable enthält die esd-Vendor-ID. Hier ist immer der Wert 00000017_h eingetragen.

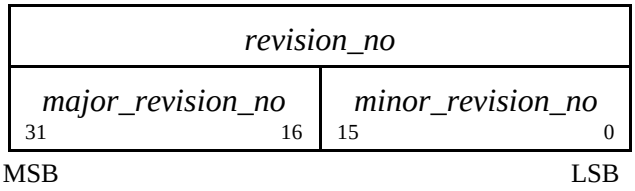
product_code Hier ist die esd-Artikelnummer des Produkts abgelegt.
Die Nibble des Langwortes haben folgende Bedeutung:

product_code = `abcd efgh`$_h$

`a`: 1... Artikelnummer beginnt mit Buchstaben “K”
2....Artikelnummer beginnt mit Buchstaben “C”
`bcde`: 4-stellige Hex-Zahl, die als Dezimalzahl vor dem Dezimalpunkt interpretiert wird.
`f`: wird zur Zeit nicht ausgewertet
`gh`: 2-stellige Hex-Zahl, die als Dezimalzahl nach dem Dezimalpunkt interpretiert wird.

Beispiel: ‘2303 2002_h’ entspricht der Artikelnummer ‘C.3032.02’.

revision_number Hier ist die Software-Version abgelegt. In den oberen zwei Bytes sind gemäß [1] die Revisionsnummern der wesentlichen (major) Änderungen aufgeführt und in den unteren zwei Bytes die Revisionsnummern einfacher Korrekturen oder Änderungen (minor).



serial_number Hier wird die Seriennummer-Kennung der Hardware gelesen. Die ersten beiden Zeichen der Seriennummer sind Buchstaben, die das Fertigungslos kennzeichnen. Die folgenden Zeichen geben die eigentliche Seriennummer wieder.

In den beiden höherwertigen Bytes von *serial_no* sind die Buchstaben des Fertigungsloses kodiert. Sie enthalten jeweils den ASCII-Code des Buchstaben mit dem hochwertigsten Bit auf '1' gesetzt, um Buchstaben und Zahlen unterscheiden zu können:
(ASCII-Code) + 80_h = gelesenes_Byte

In den beiden niederwertigen Bytes ist die Nummer des Moduls als BCD-Wert enthalten.

Beispiel:
Wird der Wert 'C1C2 0105_h' gelesen, so entspricht dies der Hardware-Seriennummer-Kennung 'AB 0105'. Dieser Wert muss der aufgeklebten Seriennummer des Moduls entsprechen.

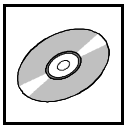
7.9.19 Synchronous Counter Overflow Value (1019_h)

INDEX	**1019_h**
Name	*Synchron_Counter_Overflow*
Datentyp	unsigned 8
Default-Wert	0

Dieses Objekt bestimmt den höchsten Wert, den der SYNC-Counter erreichen kann.

Der Wertebereich des Objektes ist in der folgenden Tabelle beschrieben:

Wert	Bedeutung
0	Die SYNC-Nachricht wird als CAN-Nachricht mit der Länge ‘0’ gesendet.
1	reserviert
2...240	Die SYNC-Nachricht wird als CAN-Nachricht mit der Länge ‘1’ gesendet. Das erste Byte der SYNC-Nachricht enthält den aktuellen Wert des SYNC-Counters.
241...255	reserviert



7.9.20 Verify Configuration (1020_h)

INDEX	**1020_h**
Name	*verify configuration*
Datentyp	unsigned 32
Default-Wert	No

In diesem Objekt können Datum und Tageszeit der letzten Konfiguration abgelegt werden, um zu einem späteren Zeitpunkt prüfen zu können, ob die gespeicherte Konfiguration der erwarteten entspricht.

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
1020_h	0	*no_of_entries*	2	2	unsigned 8	ro
	1	*configuration_date*	0...$FFFFFFFF_h$	0	unsigned 32	rw
	2	*configuration_time*	0...$FFFFFFFF_h$	0	unsigned 32	rw

Bedeutung der Variablen:

configuration_date Datum der letzten Konfiguration des Moduls, angegeben in Tagen seit dem 01.01.1984.

configuration_time Zeit in ms seit Mitternacht am Tag der letzten Konfiguration.

7.9.21 Error Behaviour Object (1029_h)

INDEX	**1029_h**
Name	*error behaviour object*
Datentyp	unsigned 8
Default-Wert	No

Tritt ein Error-Event ein (z.B. Heartbeat-Error), so wechselt das Modul in den Zustand, der in der Variablen *communication_error* oder *output_error* definiert ist.

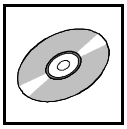
Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
1029_h	0	*no_of_error_classes*	1	1	unsigned 8	ro
	1	*communication_error*	0...2	0	unsigned 8	rw

Bedeutung der Variablen:

Variable	Bedeutung
no_of_error_classes	Anzahl der Fehler-Klassen (hier immer ‘1’)
communication_error	Heartbeat/Lifeguard-Fehler und *Bus off*

Das Modul wechselt beim Auftreten eines Fehlers in den jeweils angegebenen Modus.

Wert der Variablen	Modus, in den das Modul im Fehlerfall wechselt
0	pre-operational (nur wenn der aktuelle Zustand = operational)
1	no state change
2	stopped



7.9.22 NMT Startup ($1F80_h$)

INDEX	**$1F80_h$**
Name	*NMT startup*
Datentyp	unsigned 32
Default-Wert	2

Um in Umgebungen, in denen kein NMT-Master verfügbar ist, CANopen-Nodes starten zu können, ist der NMT Startup implementiert worden.

Über den NMT-Startup kann der Autostart eines CANopen-Nodes eingeschaltet oder ausgeschaltet werden. Weitere Funktionen des Parameters *NMT startup* werden zur Zeit nicht unterstützt.

Der Wertebereich des Objektes ist in der folgenden Tabelle beschrieben:

Wert	Bedeutung
$0000\ 0002_h$	Autostart disabled (default)
$0000\ 0008_h$	Autostart enabled
alle anderen Werte	reserviert

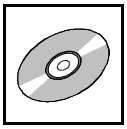
7.9.23 Self Starting Nodes Timing Parameters ($1F91_h$)

INDEX	**$1F91_h$**
Name	*Self starting nodes timing parameters*
Datentyp	unsigned 16

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
$1F91_h$	0	*number_of_entries*	1	1	unsigned 8	ro
	1	*NMT master detection timeout*	0...$FFFF_h$	64_h	unsigned 16	rw

Sub-Index 1 dieses Objektes enthält die Verzögerungszeit in [ms] zwischen dem Übergang von “preoperational” > “operational”. In der Default-Einstellung sind dies 100 ms.

Die Sub-Indizes 2 und 3 dieses Objektes werden nicht unterstützt.



7.9.24 Objekt Transmit PDO Communication Parameter (1801_h, 1802_h)

Mit diesem Objekten werden die Eigenschaften der Sende-PDOs definiert.

INDEX	**1801_h 1802_h**
Name	*transmit PDO parameter*
Data Type	PDOCommPar

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
1801_h	0	*number_of_entries*	0...FF_h	5	unsigned 8	ro
	1	*COB-ID used by PDO*	1...800007FF_h	280_h +Node-ID	unsigned 32	rw
	2	*transmission type*	0...FF_h	FF_h	unsigned 8	rw
	3	*inhibit time*	0...$FFFF_h$	0	unsigned 16	rw
	4	*reserved*	0..FF_h	0	unsigned 8	const
	5	*event timer*	0...$FFFF_h$	0	unsigned 16	rw
1802_h	0	*number_of_entries*	0...FF_h	5	unsigned 8	ro
	1	*COB-ID used by PDO*	1...800007FF_h	380_h +Node-ID	unsigned 32	rw
	2	*transmission type*	0...FF_h	FF_h	unsigned 8	rw
	3	*inhibit time*	0...$FFFF_h$	0	unsigned 16	rw
	4	*reserved*	0..FF_h	0	unsigned 8	const
	5	*event timer*	0...$FFFF_h$	0	unsigned 16	rw

Es werden die *transmission types* 0, 1...240 und 252...255 unterstützt.

7.9.25 Transmit PDO Mapping Parameter ($1A01_h$, $1A02_h$)

Mit dem Objekten wird die Zuordnung der Sendedaten zu den Tx-PDOs definiert.

INDEX	**$1A01_h$, $1A02_h$**
Name	*transmit PDO mapping*
Data Type	PDO Mapping

Die folgende Tabelle zeigt die Belegung der Transmit PDO Mapping Parameter:

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
$1A01_h$	0	*number of entries*	0...FF_h	2_h	unsigned 8	rw
	1	*Read_Analog_Input_32_1*	0...$FFFFFFFF_h$	6402 0120_h	unsigned 32	rw
	2	*Read_Analog_Input_32_2*	0...$FFFFFFFF_h$	6402 0220_h	unsigned 32	rw
$1A02_h$	0	*number of entries*	0...FF_h	2_h	unsigned 8	rw
	1	*Read_Analog_Input_32_3*	0...$FFFFFFFF_h$	6402 0320_h	unsigned 32	rw
	2	*Read_Analog_Input_32_4*	0...$FFFFFFFF_h$	6402 0420_h	unsigned 32	rw

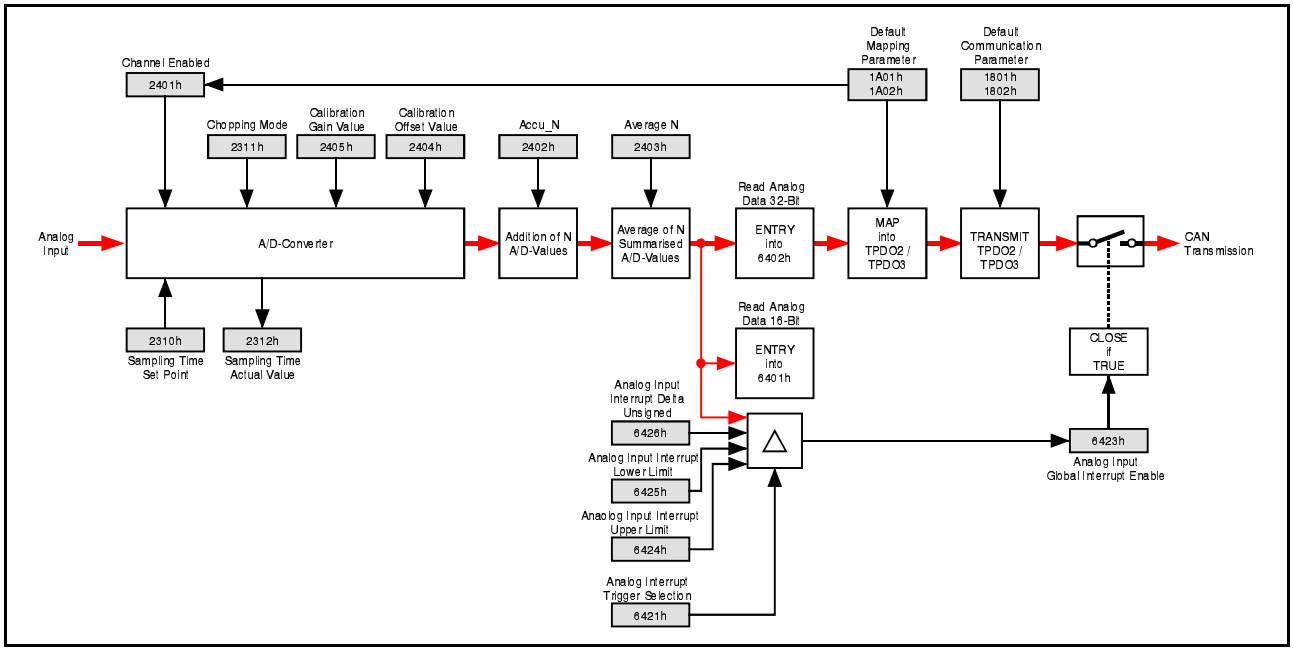
Hinweis:
Die lokale Firmware lässt ein beliebiges TxPDO-Mapping zu, d.h. auch Kombinationen von 16-Bit- und 32-Bit-A/D-Werten innerhalb eines Frames sind möglich.

7.10 Device Profile Area

7.10.1 Übersicht der implementierten Objekte (6401_h ...6426_h)

Index	Name	Datentyp
6401_h	*Read Analog Inputs 16-Bit*	integer16
6402_h	*Read Analog Inputs 32-Bit*	integer 32
6421_h	*Analog Interrupt Trigger Selection*	unsigned 8
6423_h	*Analog Input Global Interrupt Enable*	boolean
6424_h	*Analog Input Interrupt Upper Limit*	integer 32
6425_h	*Analog Input Interrupt Lower Linit*	integer 32
6426_h	*Analog Input Interrupt Delta Unsigned*	unsigned 32

7.10.2 Zusammenhang der implementierten Objekte für die analogen Eingänge



Beispiel hier: Modul befindet sich im Zustand ‘Operational’

7.10.3 Read Input 16-Bit (6401_h)

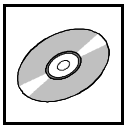
Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
6401_h	0	*Anzahl der Einträge*	4	4	unsigned 8	ro
	1	*Read_Analog_Input_16_1*	8000_h...$7FFF_h$	-	integer 16	ro
	2	*Read_Analog_Input_16_2*	8000_h...$7FFF_h$	-	integer 16	ro
	3	*Read_Analog_Input_16_3*	8000_h...$7FFF_h$	-	integer 16	ro
	4	*Read_Analog_Input_16_4*	8000_h...$7FFF_h$	-	integer 16	ro

Belegung der Variablen *Read_Analog_Input_16_x* (x = 1...4):
Die Endziffer des Variablen-Namens gibt die Nummer des jeweiligen analogen Eingangskanals an. Es werden die höherwertigen 16 Bits der A/D-Wandler in Zweierkomplement-Darstellung verwendet. Die Werte der Variablen sind die über Offset, Gain, Mittelwertbildung und ggf. Addition korrigierten Messwerte.

Wert der Variablen *Read_Analog_Input_16_x*																	gemessene Spannung
Binär (Bit 15...0)																Hexadezimal	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8000_h	-10,00 V
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	$FFFF_h$	-0,3052 mV
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000_h	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0001_h	+0,3052 mV
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	$7FFF_h$	+10,0 V - (1 $LSB_{VARIABLE}$) = 9,99969 V

Auflösung des Messwertes:
1 LSB bezogen auf die Variable *Read_Analog_Input_16_x* (x = 1...4):
1 $LSB_{VARIABLE}$ => 10 V / 8000_h => 0,3052 mV

Die Ermittlung des korrigierten Messwertes mit Offset, Gain, Mittelwertbildung und ggf. Addition wird auf Seite 82 für Read Input 32-Bit dargestellt.



7.10.4 Read Input 32-Bit (6402_h)

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
6402_h	0	*Anzahl der Einträge*	4	4	unsigned 8	ro
	1	*Read_Analog_Input_32_1*	80000000_h ... $7FFFFFFF_h$	-	integer 32	ro
	2	*Read_Analog_Input_32_2*	80000000_h ... $7FFFFFFF_h$	-	integer 32	ro
	3	*Read_Analog_Input_32_3*	80000000_h ... $7FFFFFFF_h$	-	integer 32	ro
	4	*Read_Analog_Input_32_4*	80000000_h ... $7FFFFFFF_h$	-	integer 32	ro

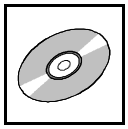
Belegung der Variablen *Read_Analog_Input_32_x* (x = 1...4):
Die Endziffer des Variablen-Namens gibt die Nummer des jeweiligen analogen Eingangskanals an. Die Daten werden linksbündig in Zweierkomplement-Darstellung übergeben. Die Werte der Variablen sind die über Offset, Gain, Mittelwertbildung und ggf. Addition korrigierten Messwerte.

Ermittlung des Spannungswertes:

Wert der Variablen *Read_Analog_Input_32_x*																								Spannung
Binär (Bit 31...0)																								Hexa-dezimal	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	80000000_h	-10,0 V
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	$FFFFFFFF_h$	-4,6566 nV
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	00000000_h	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	00000001_h	+4,6566 nV
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	$7FFFFFFF_h$	+10,0 V -(1 $LSB_{VARIABLE}$)

Auflösung des Messwertes:
1 LSB bezogen auf die Variable *Read_Analog_Input_32_x* (x = 1...4):
1 $LSB_{VARIABLE}$ => 10,0 V / $8000\ 0000_h$ => 4,6566 nV

Hinweis: 24-Bit-Auflösung des A/D-Wandlers:
1 $LSB_{AD\text{-}CONVERTER}$ => 10 V / $80\ 0000_h$ => 1,19209 µV



Device Profile Area

Berechnung des Messwertes:
Die Zusammenhänge der Objekte sind in der Abbildung auf Seite 79 veranschaulicht. Im Folgenden sind die einzelnen Schritte zur Berechnung des in den Objekten 6401_h und 6402_h zurückgegebenen A/D-Wertes aufgeführt.

Firmware-interne Variablen (vom Anwender nicht lesbar):

AD-Wert . . . . . .unkorrigierter Messwert des A/D-Wandlers

Ana_In_32_og_x . . . . .mit *Offset* und *Gain* korrigierter A/D-Wert

Ana_In_32_accu_x . . .durch Addition von *n*A/D-Werten gebildeter A/D-Wert

x . . . . . .Nummer des A/D-Kanals (1, 2, 3, 4)

Über CANopen-Objekte zugängliche Variablen:

Gain_x . . . . . Gain-Faktor (Objekt 2405_h, siehe Seite 97), default: *Gain* = 1

Offset_x . . . . . Offset-Wert (Objekt 2404_h, siehe Seite 96), default: *Offset* = 0

n . . . . . Anzahl der Additionen (Objekt 2402_h, siehe Seite 94), default *n* = 1 (keine Additionen)

m . . . . . Anzahl der gemittelten Werte (Objekt 2403_h, siehe Seite 95), default: *m* = 32 (gleitender Mittelwert aus 32 A/D-Werten)

Read_Analog_Input_32_x . . A/D-Wandler-Wert, der durch die gleitende Mittelwertbildung von *m* A/D-Werten (die bereits mit *Gain*, *Offset* und Addition bearbeitet wurden) gebildet wurde (Objekt 6401_h, 6402_h)

Berechnungsschritte:

1. Korrektur mit Gain-Faktor und Offset-Wert:

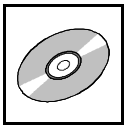
$$Ana\_In\_32\_og\_x = Offset\_x + (AD\text{-}Wert \cdot Gain\_x)$$

2. Addition der Messwerte:

$$Ana\_In\_32\_accu\_x = Ana\_In\_32\_og\_x_{(1)} + Ana\_In\_32\_og\_x_{(2)} + \ldots + Ana\_In\_32\_og\_x_{(n)}$$

3. Gleitende Mittelwertbildung:

$$Read\_Analog\_Input\_32\_x = \frac{Ana\_In\_32\_accu\_x_{(1)} + Ana\_In\_32\_accu\_x_{(2)} + \ldots + Ana\_In\_32\_accu\_x_{(m)}}{n \cdot m}$$



7.10.5 Analog Input Interrupt Trigger (6421_h)

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
6421_h	0	*Zahl der analogen Eingänge*	4	4	unsigned 8	ro
	1	*Analog_Input_IRQ_Trigger_1*	0 - 7	7	unsigned 8	rw
	2	*Analog_Input_IRQ_Trigger_2*	0 - 7	7	unsigned 8	rw
	3	*Analog_Input_IRQ_Trigger_3*	0 - 7	7	unsigned 8	rw
	4	*Analog_Input_IRQ_Trigger_4*	0 - 7	7	unsigned 8	rw

Mit diesem Objekt wird festgelegt, wann ein Interrupt ausgelöst wird.

Belegung der Variablen *Analog_Input_IRQ_Trigger_x* (x = 1...4):

Bit:	7	6	5	4	3	2	1	0
Bedeutung:	reserviert für zukünftige Anwendungen			nicht unterstützt		**Wert hat sich um mehr als Delta (6426_h) geändert**	**Lower Limit (6425_h) unter-schritten**	**Upper Limit (6424_h) über-schritten**

Beispiel:

Analog_Input_IRQ_Trigger_x	**Bedeutung**
00_h	kein Interrupt Trigger
03_h	Interrupt Upper Limit (Objekt 6424_h) und Interrupt Lower Limit (Objekt 6425_h) aktiviert.
04_h	Interrupt Trigger, wenn sich A/D-Wert um mehr als den über Objekt 6426_h eingestellten Wert (Delta) geändert hat

7.10.6 Global Interrupt Enable (6423_h)

Index	Sub-Index	Beschreibung	Wertebereich [Boolean]	Default [Boolean]	Datentyp	Zugriff
6423_h	0	*Analog_Input_Global_Interrupt_Enable*	true, false	false	boolean	rw

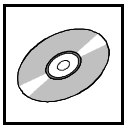
Mit dem Objekt *Analog Input Global Interrupt Enable* wird die Interrupt-Funktion des Moduls freigegeben oder unterbunden. Die Werte der Objekte 6421_h und 6426_h bleiben davon unberührt.

Der Default-Wert ist so eingestellt, das bei einer Änderung des analogen Eingangswertes kein Interrupt ausgelöst wird.

Wertebereich:

Analog_Input_Global_Interrupt_Enable	Wert	Bedeutung
true	1	Global Interrupt enabled
false	0	Global Interrupt disabled (Default-Einstellung)

Hinweis:
Es ist zwingend erforderlich *Analog_Input_Global_Interrupt_Enable* auf ‘true’ zu setzten, damit ein Event-gesteuertes Senden der TPDO gestartet werden kann.



7.10.7 Interrupt Upper Limit (6424_h)

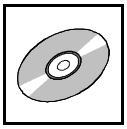
Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
6424_h	0	*Anzahl_der_Einträge*	4	4	unsigned 8	ro
	1	*Analog_Input_Interrupt_Upper_Limit_1*	80000000_h ... $7FFFFFFF_h$	0	integer 32	rw
	2	*Analog_Input_Interrupt_Upper_Limit_2*	80000000_h ... $7FFFFFFF_h$	0	integer 32	rw
	3	*Analog_Input_Interrupt_Upper_Limit_3*	80000000_h ... $7FFFFFFF_h$	0	integer 32	rw
	4	*Analog_Input_Interrupt_Upper_Limit_4*	80000000_h ... $7FFFFFFF_h$	0	integer 32	rw

Mit dem Objekt *Analog_Input_Interrupt_Upper_Limit* wird ein oberer Grenzwert für die Interrupt-Funktion festgelegt.

7.10.8 Interrupt Lower Limit (6425_h)

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
6425_h	0	*Anzahl_der_Einträge*	4	4	unsigned 8	ro
	1	*Analog_Input_Interrupt_Lower_Limit_1*	80000000_h ... $7FFFFFFF_h$	0	integer 32	rw
	2	*Analog_Input_Interrupt_Lower_Limit_2*	80000000_h ... $7FFFFFFF_h$	0	integer 32	rw
	3	*Analog_Input_Interrupt_Lower_Limit_3*	80000000_h ... $7FFFFFFF_h$	0	integer 32	rw
	4	*Analog_Input_Interrupt_Lower_Limit_4*	80000000_h ... $7FFFFFFF_h$	0	integer 32	rw

Mit dem Objekt *Analog_Input_Interrupt_Lower_Limit* wird ein unterer Grenzwert für Interrupt-Funktion festgelegt.



7.10.9 Analog Input Interrupt Delta (6426_h)

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
6426_h	0	*Anzahl_Analog_Inputs*	4	4	unsigned 8	ro
	1	*Analog_Input_IRQ_Delta_1*	0...FFFFFFFF$_h$	0	unsigned 32	rw
	2	*Analog_Input_IRQ_Delta_2*	0...FFFFFFFF$_h$	0	unsigned 32	rw
	3	*Analog_Input_IRQ_Delta_3*	0...FFFFFFFF$_h$	0	unsigned 32	rw
	4	*Analog_Input_IRQ_Delta_4*	0...FFFFFFFF$_h$	0	unsigned 32	rw

Mit diesem Objekt wird die Abweichung des Analogwertes (Delta) eingestellt, die für das Auslösen eines Interrupts ausgewertet werden kann. Die Art der Auswertung wird über das Objekt 6421_h ausgewählt. Die Interrupt-Freigabe erfolgt über das Objekt 6423_h.

Es werden Spannungsänderungen sowohl in positiver als auch in negativer Richtung ausgewertet. Dabei gibt der Wert *Analog_Input_IRQ_Delta_x* immer den Absolutwert der Differenz aus aktuellem AD-Wert abzüglich des letzten vorhergehenden AD-Werts.
Die Differenz der AD-Werte wird in Objekt 6426_h immer als positiver Betrag eingegeben.

Sind die Objekte 6421_h und 6423_h entsprechend konfiguriert, erfolgt das Auslösen des Interrupts sobald die Differenz der AD-Werte größer oder gleich dem Wert *Analog_Input_Interrupt_IRQ_Delta_x* ist. Default-mäßig ist für diesen Wert ‘0' vorgegeben. Somit wird der Interrupt immer ausgelöst, da die absolute Differenz immer ≥ 0 ist.

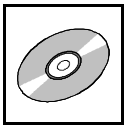
Wertebereich:

Wert der Variablen *Analog_Input_IRQ_Delta_x*	Spannungsänderung	Bemerkung
0001 0000$_h$	312,5 µV	minimaler Wert
:	:	-
FFFF FFFF$_h$	+20,48 V	maximaler Wert
0066 0000$_h$	31,875 mV	Beispiel
3400 0000$_h$	4,16 V	Beispiel

7.11 Manufacturer Specific Profile Area

7.11.1 Übersicht der Manufacturer Specifc Objects (2310_h ... 2405_h)

Index	Name	Data Type
2310_h	*Sampling Rate Set Point*	unsigned 16
2311_h	*Chopping Mode*	unsigned 8
2312_h	*Sampling Rate Actual Value*	unsigned 16
2401_h	*Channel Enabled*	unsigned 8
2402_h	*Accu_N*	unsigned 8
2403_h	*Average_N*	unsigned 8
2404_h	*Calibration Offset Value*	integer 32
2405_h	*Calibration Gain Value*	integer 16



7.11.2 Sample Time Set Point (2310_h)

Index	Sub-index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
2310_h	0	*Anzahl der Einträge*	1	1	unsigned 8	ro
	1	*Sample_Time_Set_Point*	82_h...$14FC_h$	4994	unsigned 16	rw

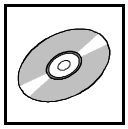
Über dieses Objekt wird der Sollwert der Sample-Time übergeben. Die Sample-Time enthält die Wandlungszeit des A/D-Wandlers AD7732. Der Sollwert wird für die Einstellung der Sample-Time aller vier möglichen A/D-Kanäle ausgewertet. Je größer die Sampling-Time gewählt wird, desto besser wird die Auflösung der Messwerte.

Belegung der Variablen *Sample_Time_Set_Point*:
Der Sollwert der Sample-Time wird in Schritten zu 0,5 µs eingetragen:

Beispiele:

Beispielwerte der Variablen *Sample_Time_Set_Point _x*		Sollwert der Sample-Time	Output Data Rate	Effective Resolution (bei ±10 V)[9]
Hexadezimal	Dezimal			
0082_h	130	65 µs	15 384,6 Hz	≈17 Bits
0083_h	131	65,5 µs	15 267 Hz	≈17 Bits
$00A4_h$	164	82 µs	12166 Hz	17,3 Bits
$019E_h$	414	207 µs	4 826 Hz	19.0 Bits
0316_h	789	395 µs	2 534 Hz	19,5 Bits
$03E5_h$	998	499 µs	2005 Hz	19,7 Bits
$07CE_h$	1998	999 µs	1001 Hz	20.3 Bits
1382_h	4994	2497 µs (Default)	400 Hz	21 Bits
$14FC_h$	5372	2686 µs	372 Hz	21 Bit

Die obige Tabelle zeigt Beispiele der technisch sinnvollen Grenzwerte mit Angabe der Effective Resolution (bei ±10 V)[9].



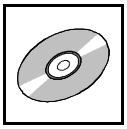
Manufacturer Specific Profile Area

Grenzwerte der Sample Time:

Chopping Mode	Minimalwert		Maximalwert	
	Sample Time	Eintrag in Objekt 2310_h [DEZ]	Sample-Time pro Kanal	Eintrag in Objekt 2310_h [DEZ]
inaktiv	65 µs	130	1357 µs	2714
aktiv	82 µs	164	2686 µs	5372

Hinweis:
Durch Mittelwertbildung wird die Anzahl der ausgegebenen Analogwerte herabgesetzt (siehe Objekte 2402_h, 2403_h).

Hinweis:
Bitte beachten Sie mögliche Einschränkungen durch die begrenzte Bandbreite ihres CAN-Netzwerkes.



7.11.3 Chopping Mode (2311_h)

Index	Sub-Index	Beschreibung	Wertebereich [boolean]	Default	Datentyp	Zugriff
2311_h	0	*Anzahl der Einträge*	4	4	unsigned 8	ro
	1	*Chop_Mode_1*	0, 1	1	boolean	rw
	2	*Chop_Mode_2*	0, 1	1	boolean	rw
	3	*Chop_Mode_3*	0, 1	1	boolean	rw
	4	*Chop_Mode_4*	0, 1	1	boolean	rw

Wird für die A/D-Wandler der Chopping Mode aktiviert, so reduziert sich der Offset des Messwerts. Der Chopping Mode reduziert aber auch die Wandlungsrate der A/D-Wandler.

Details zum Chopping Mode können im Datenblatt[9] des A/D-Wandlers AD7732(z.B. unter www.analog.com erhältlich) nachgelesen werden.
Es wird empfohlen mit aktiviertem Chopping Mode (*Chop_Mode_x*: "1") zu arbeiten.

Wertebereich:

Chop_Mode_x	Wert	Bedeutung
false	0	kein Chopping Mode
true	1	Chopping Mode aktiviert

7.11.4 Sample Time Actual Value (2312_h)

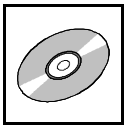
Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
2312_h	0	*Anzahl der Einträge*	2	2	unsigned 8	ro
	1	*Sample_Time_Actual _1*	0...$FFFF_h$	-	unsigned 16	ro
	2	*Sample_Time_Actual _2*	0...$FFFF_h$	-	unsigned 16	ro
	3	*Sample_Time_Actual _3*	0...$FFFF_h$	-	unsigned 16	ro
	4	*Sample_Time_Actual _4*	0...$FFFF_h$	-	unsigned 16	ro

Über dieses Objekt kann der Istwert der eingestellten Sample-Time gelesen werden.
(siehe auch Hinweise zu Sample-Time-Einstellung auf Seite 89)

Belegung der Variablen *Sample_Time_Actual_x* (x = 1...4):

Für die Default-Einstellung der Objekte gilt: der gelesene Wert multipliziert mit 0,5 ergibt die Sample-Time in [µs]:

Wert der Variablen *Sample_Time_Actual _x*	Istwert der Sample-Time
0	0 µs
1	0,5 µs
:	:
$FFFF_h$	32767,5 µs



7.11.5 Channel Enabled (2401_h)

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
2401_h	0	*Anzahl der Einträge*	4	4	unsigned 8	ro
	1	*Channel_Enabled_1*	0, 1	-	boolean	ro
	2	*Channel_Enabled_2*	0, 1	-	boolean	ro
	3	*Channel_Enabled_3*	0, 1	-	boolean	ro
	4	*Channel_Enabled_4*	0, 1	-	boolean	ro

Modul im Zustand Operational:
Ob ein A/D-Kanal gewandelt wird, macht die lokale Firmware davon abhängig, ob der Kanal im PDO-Mapping einem TPDO zugeordnet worden ist. Kanäle, die nicht gemappt sind, werden auch nicht gewandelt. Die Daten der Objekte 6401_h und 6402_h werden nicht aktualisiert, wenn keine neuen A/D-Wandlungen erfolgen.

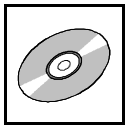
Module im Zustand Pre-Operational:
Die Objekte können per SDO-Zugriff gelesen werden.

Werden nicht alle Kanäle gewandelt, so erhöht dies die mögliche Sampling-Rate der verbliebenen Kanäle.

Mit dem Objekt *Channel Enabled* kann ermittelt werden, welche A/D-Kanäle gewandelt werden und welche nicht.

Wertebereich:

Channel_Enabled_x	Wert	Bedeutung
false	0	A/D-Wandler-Kanal ist nicht “gemapped”
true	1	A/D-Wandler-Kanal ist “gemapped”



Manufacturer Specific Profile Area

7.11.6 Accu N (2402_h)

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
2402_h	0	*Anzahl der Einträge*	4	4	unsigned 8	ro
	1	*Accu_Count_1*	0...8	0	unsigned 8	rw
	2	*Accu_Count_2*	0...8	0	unsigned 8	rw
	3	*Accu_Count_3*	0...8	0	unsigned 8	rw
	4	*Accu_Count_4*	0...8	0	unsigned 8	rw

Mit diesem Objekt wird definiert, wie viele Analogwerte aufaddiert werden sollen.

Durch entsprechende Vorverarbeitung des Ausgangswertes des A/D-Wandlers ist das Ergebnis dieser Addition immer ein 16 Bit-Wert in Zweier-Komplement-Darstellung.

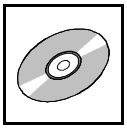
Die Anzahl der Additionen errechnet sich wie folgt:

n ... Anzahl der Additionen

$$n = 2^{(Accu\_Count\_x)}$$

Es können also bis zu 256 Werte aufaddiert werden.

Vorteil: Filter mit Decimation, Verbesserung der Auflösung, Verringerung der Datenrate



7.11.7 Average N (2403_h)

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
2403_h	0	*Anzahl der Einträge*	4	4	unsigned 8	ro
	1	*Average_Count_1*	0...5	5	unsigned 8	rw
	2	*Average_Count_2*	0...5	5	unsigned 8	rw
	3	*Average_Count_3*	0...5	5	unsigned 8	rw
	4	*Average_Count_4*	0...5	5	unsigned 8	rw

Mit diesem Objekt wird definiert, aus wie vielen zwischengespeicherten Analogwerten der A/D-Wandler den gleitenden Mittelwert bildet. Bei Lesezugriffen auf die A/D-Werte wird dann der Mittelwert der letzten $2^{(Average\_Count\_x)}$ Samples gelesen.
Da die A/D-Werte in einem Ringpuffer zwischengespeichert werden, steht nach jeder Wandlung ein neuer Mittelwert zur Verfügung.

Die Anzahl der gemittelten Werte errechnet sich wie folgt:

m ... Anzahl gemittelter Werte

$$m = 2^{(Average\_Count\_x)}$$

Es kann also über die letzten 1, 2, 4, 8, 16 oder 32 (default) Werte gemittelt werden.

Hinweis:
Bei Eingangssignalen mit einer Frequenz << F_{sample} erhält man durch das Filter eine Verbesserung der Auflösung von

Average_Count /2

Bits.

Als weiteren Effekt erhält man eine Notchfiltercharakteristik mit Nullstellen bei

$F_{sample}/2^k$ mit k = 1 ... *Average_Count_x*.

Beispiel:
Bei einer Sample-Frequenz von 200 Hz und *Average_Count* = 2 erhält man also eine Unterdrückung der Frequenzen von 100 Hz und 50 Hz.

Vorteil: nach jeder Wandlung steht ein neuer Mittelwert zur Verfügung
Nachteil: höherer interner Rechenaufwand als bei der Addition der Messwerte (siehe Objekt 2402_h)

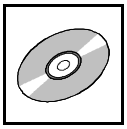
7.11.8 Calibration Offset Value (2404_h)

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
2404_h	0	*Anzahl der Einträge*	4	4	unsigned 8	ro
	1	*Calibration_Offset_1*	80000000_h ... $7FFFFFFF_h$	$Offset_{Factory\_}1$	integer 32	rw
	2	*Calibration_Offset_2*	80000000_h ... $7FFFFFFF_h$	$Offset_{Factory\_}2$	integer 32	rw
	3	*Calibration_Offset_3*	80000000_h ... $7FFFFFFF_h$	$Offset_{Factory\_}3$	integer 32	rw
	4	*Calibration_Offset_4*	80000000_h ... $7FFFFFFF_h$	$Offset_{Factory\_}4$	integer 32	rw

Die Default-Werte $Offset_{Factory\_}x$ (x = 1 ...4) ergeben sich bei der Werkskalibrierung des CAN-CBX-AI420. Über das Objekt 1011_h (*load_manufacurer_parameter*) können diese Modul-spezifischen Default-Werte neu geladen werden.

Wertebereich:

Wert der Variablen *Calibration_Offset_x*	Spannungs-Offset
$8000\ 0000_h$	-10,0 V
:	:
0	0
:	:
$7FFF\ FFFF_h$	+10,0 V - (1 $LSB_{AD\text{-}CONVERTER}$) => 9,99999 V



7.11.9 Calibration Gain Value (2405_h)

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
2405_h	0	*Anzahl der Einträge*	4	4	unsigned 8	ro
	1	*Calibration_Gain_1*	8000_h...$7FFF_h$	$Gain_{Factory\_}1$	integer 16	rw
	2	*Calibration_Gain_2*	8000_h...$7FFF_h$	$Gain_{Factory\_}2$	integer 16	rw
	3	*Calibration_Gain_3*	8000_h...$7FFF_h$	$Gain_{Factory\_}3$	integer 16	rw
	4	*Calibration_Gain_4*	8000_h...$7FFF_h$	$Gain_{Factory\_}4$	integer 16	rw

Mit diesem Objekt kann das Gain der A/D-Wandler-Kanäle korrigiert werden. Der Gain-Wert, mit dem der gemessene A/D-Wandler-Wert multipliziert wird, wird wie folgt gesetzt:

$$Gain\_x = 1 + \frac{Calibration\_Gain\_x}{2^{18}}$$

mit x= 1, 2, 3, 4

Damit ergibt sich ein Wertebereich für den Gain-Faktor von 0,875 ... 1,125

Die Default-Werte $Gain_{Factory\_}x$ (x = 1 ...4) ergeben sich bei der Werkskalibrierung des CAN-CBX-AI420. Über das Objekt 1011_h (*load_manufacurer_parameter*) können diese Modul-spezifischen Default-Werte neu geladen werden.

7.12 Firmware-Verwaltung über DS-302-Objekte ($1F50_h$...$1F52_h$)

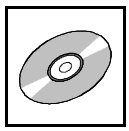
Die im folgenden beschriebenen Objekte werden benötigt, wenn die Firmware des Gerätes ausgetauscht werden soll.

Achtung:
Das Firmware-Update sollte nur durch erfahrene Anwender durchgeführt werden!
Unsachgemäße Durchführung des Updates kann zum ungewollten Löschen des Speichers und zum Verlust der Firmware führen. Das Modul kann dann nicht weiter betrieben werden!

Hinweis:
Für ein Firmware-Update bietet esd das Programm CANfirmdown. Bitte wenden Sie sich dazu an unseren Support.

Im normalen DS-301-Modus kann auf das Objekt $1F50_h$ nicht zugegriffen werden.
Die Objekte $1F51_h$ und $1F52_h$ stehen auch im normalen DS-301-Modus zur Verfügung.
Weitere Informationen zu den Objekten und zum Firmware-Update entnehmen Sie bitte [5]

Index	Sub-Index	Beschreibung	Datentyp	Zugriff
$1F50_h$	0	Boot-Loader: Firmware Download	domain	rw
$1F51_h$	1	Boot-Loader: FLASH-Kommando	unsigned 8	rw
$1F52_h$	0,1,2	Boot-Loader: Firmware-Datum	unsigned 32	ro



7.12.1 Download-Steuerung über Objekt ($1F51_h$)

INDEX	**$1F51_h$**
Name	Program Control
Datentyp	unsigned 8
Zugriff	rw
Wertebereich	$0...FE_h$
Default-Wert	0

Hinweis:
Der Wertebereich dieses Objektes in der Implementierung für die CAN-CBX-AI420 weicht vom Wertebereich, der in [5] angegeben ist, ab.
Weitere Informationen zum Objekt $1F51_h$ und zum Firmware-Update entnehmen Sie bitte [5]

7.12.2 Verify Application Software ($1F52_h$)

Index	Sub-Index	Beschreibung	Wertebereich	Default	Datentyp	Zugriff
$1F52_h$	0	*Anzahl der Einträge*	2	2	unsigned 8	ro
	1	*Application_Software_Date*	$0...FFFF\ FFFF_h$	-	unsigned 32	ro
	2	*Application_Software_Time*	$0...0526\ 5C00_h$	-	unsigned 32	ro

Beschreibung der Variablen:

Application_Software_Date — Datum der Erstellung der verwendeten Firmware, angegeben in Anzahl der Tage seit dem 1. Januar 1984

Application_Software_Time — Zeit der Erstellung der verwendeten Firmware, angegeben in Anzahl der Millisekunden ab Mitternacht.

8. Referenzen

[1] CiA 301
CANopen® Application layer and communication profile V 4.2.0 (12.2011)
CAN in Automation (CiA) e. V., Nürnberg, Deutschland

[2] CiA Draft Standard Proposal 401 V3.0 (10.2006)
CANopen Device profile for generic IO modules

[3] CiA Draft Recommendation 303 V1.3 (08.2006)
CANopen Additional specification, Part 3: Indicator specification

[4] CAN Application Layer for Industrial Applications CiA/DS202-2 February 1996
CMS Protocol Specification

[5] CiA Draft Standard Proposal 302 V4.1 (04.2010)
Additional application layer functions, Part 3: Configuration and program download

[6] Linear Technology, Datenblatt: LTC 2600/LTC2610/LTC2620 Octal 16-/14-/12-Bit Rail-to-rail DACs in Lead SSOP, USA, 2600fa, LT/TP1103 1K RevA

[7] Phoenix Contact GmbH & Co. KG, Blomberg.
Technische Daten entnommen aus COMBICON Online-Katalog,
https://www.phoenixcontact.com/online/portal/de
Leiterplattensteckverbinder - FKCT-2,5/4-ST KMGY - 1921900, Abgerufen am 09.10.2013

[8] Phoenix Contact GmbH & Co. KG, Blomberg.,
Technische Daten entnommen aus COMBICON Online-Katalog,
https://www.phoenixcontact.com/online/portal/de
Leiterplattensteckverbinder - FK-MCP 1,5/...-STF-3,81, Abgerufen am 09.10.2013

[9] Analog Devices, Inc. Datenblatt: AD7732 Data Sheet Rev A, 06/2011, USA
http://www.analog.com

9. EU-Konformitätserklärung

EU-KONFORMITÄTSERKLÄRUNG
EU DECLARATION OF CONFORMITY

Adresse / *Address*
esd electronic system design gmbh
Vahrenwalder Str. 207
30165 Hannover
Germany

esd erklärt, dass das Produkt *esd declares, that the product*	Typ, Modell, Artikel-Nr. *Type, Model, Article No.*
CAN-CBX-AI420	**C.3030.02**
die Anforderungen der Normen *fulfills the requirements of the standards*	**EN 61000-6-2:2005,** **EN 61000-6-4:2007+A1:2011**
gemäß folgendem Prüfbericht erfüllt. *according to test certificate.*	**H-K00-0302-09**
Das Produkt entspricht damit der EU-Richtlinie „EMV" *Therefore the product corresponds to the EU Directive 'EMC'*	**2014/30/EU**
Das Produkt entspricht der EU-Richtlinie „RoHS" *The product corresponds to the EU Directive 'RoHS'*	**2011/65/EU**

Diese Erklärung verliert ihre Gültigkeit, wenn das Produkt nicht den Herstellerunterlagen entsprechend eingesetzt und betrieben wird, oder das Produkt abweichend modifiziert wird.
This declaration loses its validity if the product is not used or run according to the manufacturer's documentation or if non-compliant modifications are made.

Name / *Name*	T. Ramm
Funktion / *Title*	CE-Koordinator / *CE Coordinator*
Datum / *Date*	Hannover, 2014-08-11

Thorsten Ramm

Rechtsgültige Unterschrift / *authorized signature*

I:\Texte\Doku\MANUALS\CAN\CBX\AI420\CE-Konformität\CAN-CBX-AI420_EU_Konformitaetserklaerung_2014-08-11.odt

10. Bestellhinweise

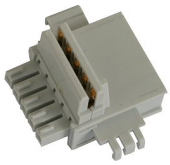
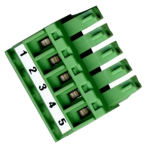
Typ	Eigenschaften	Bestell-Nr.
CAN-CBX-AI420	CAN-CBX-AI420 4 analoge Eingänge, 20 Bit, incl. 1x CAN-CBX-TBUS (C.3000.01)	C.3030.02
Zubehör		
CAN-CBX-TBUS 	Tragschienen-Busverbinder des CBX-InRailBus für CAN-CBX-Module, (ein Busverbinder ist im Lieferumfang des CAN-CBX-Moduls enthalten)	C.3000.01
CAN-CBX-TBUS-Connector 	Anschlussstecker des CBX-InRailBus zum Anschluss der +24V Versorgungsspannung und des CAN-Interface, Buchsenkontakte	C.3000.02
CAN-CBX-TBUS-Connection adapter 	Anschlussstecker des CBX-InRailBus zum Anschluss der +24V Versorgungsspannung und des CAN-Interface, Stiftkontakte	C.3000.03

Tabelle 13: Bestellhinweise

PDF-Handbücher
Handbücher sind in Englisch und üblicherweise auch in Deutsch erhältlich. Die Verfügbarkeit der Handbücher entnehmen Sie bitte der unteren Tabelle.
Die Handbücher im PDF-Format können Sie kostenlos von unserer Webseite www.esd.eu herunterladen.

Handbücher		**Bestell-Nr.**
CAN-CBX-AI420-MD	Anwenderhandbuch in Deutsch	C.3030.20
CAN-CBX-AI420-EN	Anwenderhandbuch in Englisch	C.3030.21

Tabelle 14: Verfügbare Handbücher

Gedruckte Handbücher
Benötigen Sie zusätzlich einen Ausdruck des Handbuches, kontaktieren Sie bitte unser Sales-Team sales@esd.eu für ein Angebot. Gedruckte Handbücher können gegen eine Gebühr bestellt werden.

EDS Datei
Die EDS-Datei des CAN-CBX-AI420 kann von der esd-Webseite unter www.esd.eu heruntergeladen werden.