

CAN-CBM-AI410

4 analoge Eingänge

CANopen-
Software-Handbuch

Handbuch-Datei:	I:\texte\Doku\MANUALS\CAN\CBM\AI410\Deutsch\AI410_13S.ma9
Datum der Druckvorlagenherstellung:	21.02.2005

Beschriebene Software:	CANopen
Revision:	1.1

Änderungen in den Kapiteln

Die hier aufgeführten Änderungen im Handbuch betreffen sowohl Änderungen in der **Firmware** als auch reine Änderungen in der **Beschreibung** der Sachverhalte.

Kapitel	Änderungen gegenüber Vorversion
1.5	Kapitel 'Ermittlung der Analogwerte ' eingefügt
-	-

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.

Sämtliche Rechte an der Dokumentation liegen bei **esd**. Die Weitergabe an Dritte und Vervielfältigung jeder Art, auch auszugsweise, sind nur mit schriftlicher Genehmigung durch **esd** gestattet.

esd electronic system design gmbh

Vahrenwalder Str. 207

30165 Hannover

Tel.: 0511/372 98-0

FAX : 0511/372 98-68

E-Mail: info@esd-electronics.com

Internet: www.esd-electronics.com

1. Allgemeines	3
1.1 Begriffsdefinition	3
1.2 NMT-BOOT-up	4
1.2.1 Minimum Capability Device Boot-up	4
1.3 CANopen-Objektverzeichnis	4
1.4 Kommunikationsparameter der PDOs	5
1.4.1 Zugriff auf die Kommunikationsparameter über SDO-Telegramme	5
1.5 Ermittlung der Analogwerte	7
2. Communication Profile Area	9
2.1 Verwendete Bezeichnungen und Abkürzungen	9
2.2 Übersicht der Kommunikationsparameter	10
2.3 Beschreibung der Parameter	11
2.3.1 Device Type 1000 _h	11
2.3.2 Error Register 1001 _h	12
2.3.3 Manufacturer Status Register 1002 _h	13
2.3.4 Pre-defined Error Field 1003 _h	14
2.3.5 COB-ID of SYNC-Message 1005 _h	16
2.3.6 Manufacturer's Device Name 1008 _h	17
2.3.7 Manufacturer's Hardware Version 1009 _h	18
2.3.8 Manufacturer's Software Version 100A _h	19
2.3.9 Guard Time 100C _h und Life Time Factor 100D _h	20
2.3.10 Store Parameters 1010 _h	21
2.3.11 Restore Default Parameters 1011 _h	22
2.3.12 COB_ID Emergency Object 1014 _h	23
2.3.13 Producer Heartbeat Time 1017 _h	24
2.3.14 Identity Object 1018 _h	25
2.3.15 Verify Configuration 1020 _h	27
2.3.16 Error Behaviour Object 1029 _h	28
2.3.17 Objekt Transmit PDO Communication Parameter 1800 _h	29
2.3.18 Transmit PDO Mapping Parameter 1A00 _h	30
2.4 Unterstützte Übertragungsarten nach DS301, Tabelle 10-7	31
2.5 Node Guarding / Life Guarding	32
3. Device Profile Area	33
3.1 Übersicht	33
3.1.1 Read Analog Input 16-bit 6401 _h	34
3.1.2 Analog Input Interrupt Trigger 6421 _h	35
3.1.3 Analog Input Interrupt Source 6422 _h	36
3.1.4 Global Interrupt Enable 6423 _h	37
3.1.5 Analog Input Interrupt Delta 6426 _h	38
4. PDO-Belegung	39
5. Schnellstart	41
5.1 Konfiguration für den Einsatz im CANopen-Netzwerk	41
5.2 Tabelle der wichtigsten Identifier und Nachrichten für CANopen	42

Diese Seite ist bewusst unbedruckt.

1. Allgemeines

Dieses Kapitel enthält Grundlagen zum Thema CANopen. Eine komplette CANopen-Beschreibung läge außerhalb des Rahmens dieses Handbuchs. Weitergehende Informationen sind daher den CAL / CANopen - Dokumentationen 'CiA Draft Standard 201...207', 'CiA Draft Standard 301' und 'CiA Draft Standard Proposal 401' zu entnehmen.

1.1 Begriffsdefinition

COB ...	Communication Object	
Emergency-ID...	Emergency Data Object	Notfalldaten
n/a ...	not available	
NMT...	Network Management (Master)	Netzwerkmanagement
Rx...	receive	empfangen
SDO...	Service Data Object	Service-Daten (Parameter)
SW ...	Software	
Sync...	Sync(frame) Telegramm	Synchronisations-Identifizier
tbd ...	to be defined	Daten müssen noch festgelegt werden
Tx...	transmit	senden
Node-ID	CANopen-Modul-Nummer	

1.2 NMT-BOOT-up

1.2.1 Minimum Capability Device Boot-up

Das CAN-CBM-AI410-Modul kann mit dem im CiA-Draft Standard 301 beschriebenen ‘Minimum Capability Device’ - 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-Identifizier ‘0000_h’ z. B. das 2-Byte-Telegramm ‘01_h’, ‘00_h’ zu senden (= Start Remote Node all Devices).

1.3 CANopen-Objektverzeichnis

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

Der Index kann ein 16-Bit-Parameter nach der CANopen-Spezifikation (CiA-Draft DS301) 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 [Hex]	Objekt	Beispiel
0001 ... 009F	Definition von Datentypen	
1000 ... 1FFF	Communication Profile Area	1001 _h = Fehler-Register
2000 ... 5FFF	Manufacturer Specific Profile Area	
6000 ... 9FFF	Standardised Device Profile Area	nach Anwendungsprofil DS401
A000 ... FFFF	reserviert	

PDOs (Process Data Objects)

Die PDOs dienen zur Übertragung der Prozessdaten.

Im ‘Receive’-PDO werden Prozessdaten vom CAN-CBM-AI410-Modul empfangen.

Im ‘Transmit’-PDO sendet das CAN-CBM-AI410-Modul (welches im CANopen-Netz selbst immer SLAVE ist !) Daten auf dem CANopen-Netz.

Die asynchrone Übertragungsart der PDOs wird durch den ‘PDO transmission type’ im Kommunikationsparameter des entsprechenden Prozessdatenobjekts definiert.

Die synchrone Übertragung wird zur Zeit nicht unterstützt.

1.4 Kommunikationsparameter der PDOs

Die Kommunikationsparameter der PDOs (nach DS301) werden als SDO (Service Data Objects) auf der ID '600_h + Node-ID' übertragen (Request). Der Empfänger quittiert die Parameter auf der ID '580_h + Node-ID' (Response).

Die **Node-ID** (Modul-Nummer) wird über die Kodierschalter (SW100, SW101) eingestellt. Die möglichen Einstellungen sind im Hardware-Handbuch des CAN-CBM-AI410-Moduls ausführlich beschrieben.

1.4.1 Zugriff auf die Kommunikationsparameter über SDO-Telegramme

Die SDOs (Service-Daten-Objekte) dienen dem 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 beim CAN-CBM-AI410-Modul im Zustand *Operational* und *Preoperational* möglich.

Dieses Kapitel gibt nicht alle möglichen, sondern nur einige wichtige Zugriffsarten für das CAN-CBM-AI410-Modul wieder.

Die Definitionen für die Zugriffsarten können der CiA DS202-2 (CMS-Protocol Specification, Kap. 6: CMS Multiplexed Domain Protocols) entnommen werden.

Ein SDO ist wie folgt aufgebaut:

Identifizier	Befehls-code	Index (low)	Index (high)	Subindex	LSB	Datenfeld	MSB
--------------	--------------	-------------	--------------	----------	-----	-----------	-----

Beispiel :

Abfrage des Gerätetyps (Device type) für das Modul mit der Node-ID 19 (13_h)

Request:

600 _h + Node-ID = 613 _h	40 _h (read)	00 _h 10 _h (Index = 1000 _h) (Receive-PDO-Comm-Para)	00 _h	00 _h	00 _h	00 _h	00 _h
---	-------------------------------	--	-----------------	-----------------	-----------------	-----------------	-----------------

Response:

580 _h + Node-ID = 593 _h	43 _h (read)	00 _h 10 _h (Index = 1000 _h) (Receive-PDO-Comm-Para)	00 _h	91 _h	01 _h	02 _h	00 _h
---	-------------------------------	--	-----------------	-----------------	-----------------	-----------------	-----------------

(Device type = 0002.0191_h)

Der Wert des Device type für das Modul mit der Node-ID 19 (13_h) beträgt: 0002.0191_h

Identifizier

Die Parameter werden auf der ID '600_h + Node-ID' übertragen (Request).

Der Empfänger quittiert die Parameter auf der ID '580_h + Node-ID' (Response).

Übersicht

Befehlscode

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

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

Das CAN-CBM-AI410-Modul antwortet auf jedes empfangene Telegramm mit einem Antworttelegramm. Das Antworttelegramm kann folgende Befehlscodes enthalten:

$43_h = 67_d$: Read Response, dieses Telegramm enthält den gewünschten Parameter
 $60_h = 96_d$: Write Response, d.h. ein Parameter wurde erfolgreich gesetzt
 $80_h = 128_d$: Error Response, d.h. das CAN-CBM-AI410-Modul meldet einen Kommunikationsfehler.

Häufig verwendete Befehlscodes

Die folgende Tabelle gibt eine Übersicht über häufig verwendete Befehlscodes. Die Kommando-Frames müssen immer 8 Datenbytes enthalten. Hinweise zur Syntax und weitere Befehlscodes sind in der CiA DS202-2 (CMS-Protocol Specification, Kap. 6: CMS Multiplexed Domain Protocols) nachzulesen.

Kommando	für Anzahl Datenbytes	Befehlscode [Hex]
Write Request (Initiate Domain Download)	1	2F
	2	2B
	3	27
	4	23
Write Response (Initiate Domain Download)	-	60
Read Request (Initiate Domain Upload)	-	40
Read Response (Initiate Domain Upload)	1	4F
	2	4B
	3	47
	4	43
Error Response (Abort Domain Transfer)	-	80

Fehlercodes des SDO-Domain-Transfers

Die folgenden Fehlercodes können zur Anwendung kommen (gemäß CiA Work Draft WD-301):

Fehlercode [Hex]	CAN-CBM-AI410-Bezeichnung	Erläuterung
0x05030000	SDO_TOGGLE_BIT_NOT_ALTERNATED	Toggle Bit nicht gewechselt
0x05040001	SDO_CS_UNKNOWN	falscher Command Specifier
0x06010002	SDO_READ_ONLY	Schreibzugriff ist hier falsch
0x06020000	SDO_WRONG_INDEX	falscher Index
0x06060000	SDO_HARDWARE_ERROR	kein Zugriff durch Hardware Fehler
0x06070010	SDO_WRONG_LENGTH	falsche Anzahl Daten-Bytes
0x06070012	SDO_PARA_TO_LONG	Länge des Service-Parameters zu groß
0x06070013	SDO_PARA_TO_SHORT	Länge des Service-Parameters zu klein
0x06090011	SDO_WRONG_SUBIND	falscher Sub-Index
0x06090030	SDO_VALUE_RANGE_EXCEEDED	-
0x08000000	SDO_OTHER_ERROR	nicht definierte Fehlerursache
0x08000020	SDO_DATA_CANNOT_STORED	Daten können nicht transferiert oder gespeichert werden

Index, Sub-Index

Der Index und der Sub-Index werden in dem folgenden Kapitel 'Communication Profile Area' 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'.

1.5 Ermittlung der Analogwerte

Die A/D-Werte werden zyklisch alle 2 ms gewandelt.

Die letzten gewandelten A/D-Werte werden gesendet:

- wenn das Modul die in Objekt 1800_h unter *transmission type* festgelegte Anzahl empfangener SYNC-Messages (1...240) empfangen hat
- wenn ein Remote Request für den TxPDO empfangen wurde
- wenn ein Interrupt wegen Änderung eines A/D-Wertes aufgetreten ist (Freigabe durch Objekt 6423_h)

Diese Seite ist bewusst unbedruckt.

2. Communication Profile Area

2.1 Verwendete Bezeichnungen und Abkürzungen

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

PDO-Mapping	für diesen Sub-Index des PDOs ist PDO-Mapping möglich
Access Mode	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.
Value Range	Wertebereich des Parameters
Default-Value	Grundeinstellung des Parameters bei Auslieferung des Moduls
Name/Description	Name und Kurzbeschreibung des Parameters

2.2 Übersicht der Kommunikationsparameter

Das Format der Kommunikationsparameter kann der CiA DS301 entnommen werden. Besonderheiten der Implementierung der CAN-CBM-AI410 sind in den anschließenden Kapiteln dieses Handbuchs aufgeführt.

Das Modul unterstützt nur die in der folgenden Tabelle aufgeführten Kommunikationsparameter.

Index [Hex]	Name	Subindex	Typ	Zugriff	Default
1000	Device Type	-	Unsigned32	ro	00040191 _h
1001	Error Register	-	Unsigned8	ro	Error-Code
1002	Status Register	-	Unsigned32	ro	0
1003	Pre-Defined Error Field	0...16	Unsigned32	rw	Index 0
1005	COB ID Sync	-	Unsigned32	rw	80 _h
1008	Manufacturer's Device Name 1*)	-	Visible String 1*)	ro	CAN-CBM-AI410 3*)
1009	Manufacturer's Hardware Version	-	Visible String	ro	1.1
100A	Manufacturer's Software Version	-	Visible String 2*)	ro	1.0
100C	Guard Time	-	Unsigned16	rw	0 Sek.
100D	Life Time Factor	-	Unsigned8	rw	0
1010	Store Parameter	0, 1, 2, 3	Unsigned32	rw	-
1011	Restore Parameter	0, 1, 2, 3	Unsigned32	rw	-
1014	COB-ID Emergency message	-	Unsigned32	rw	80 _h + Node-ID
1017	Producer Heartbeat Time	-	Unsigned16	rw	0 ms
1018	Identity Object	0...4	Unsigned32	ro	00000017 _h
1020	Verify Configuration	0, 1, 2	Unsigned32	rw	0
1029	Error Behaviour	0, 1	Unsigned32	rw	0
1800	Transmit PDO Communication	0,1,2,5	PDOCommPar	--	--
1A00	Transmit PDO Mapping Parameter	0,...5	PDOMapping	ro	--

ro - Read Only, rw - Read/Write

1*) Achtung: Länge > 4 Bytes, d.h. Upload aufwendiger als bei anderen Parametern (siehe S. 17)

2*) abhängig vom Revisionsstand der Software, 3*) abhängig von der Default-Konfiguration

2.3 Beschreibung der Parameter

2.3.1 Device Type 1000_h

INDEX	1000_h
Name	<i>device type</i>
Data Type	unsigned 32
Default Value	No

Der Wert des *device type* beträgt: 0004.0191_h (Analoger Output: 0004_h
Device Profile Number: 0191_h)

Beispiel : Auslesen des Device Type

Der CANopen Master sendet unter dem Identifier '603_h' (600_h + Node-ID) den Read Request an das CAN-CBM-AI410-Modul mit der Modul-Nummer 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-CBM-AI410-Modul mit der Node-ID 3 antwortet dem Master anhand der Read Response unter dem Identifier '583_h' (580 + 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	91 _h	01 _h	04 _h	00 _h
			Read Response	Index=1000 _h		Sub Index	dig. Profile Nr.191		Analog Input	

ausgebener Wert des Device Type: 0004.0191_h

Das Datenfeld ist grundsätzlich nach der Regel 'niederwertiges Byte zuerst, höherwertiges Byte zuletzt' aufgebaut (siehe Kapitel 1.4.1, Datenfeld).

2.3.2 Error Register 1001_h

Das CAN-CBM-AI410-Modul nutzt das Error-Register, um Fehlermeldungen anzuzeigen.

INDEX	1001_h
Name	<i>error register</i>
Data Type	unsigned 8
Default Value	No

Folgende Bits des Error-Registers werden zur Zeit unterstützt:

Bit	7	6	5	4	3	2	1	0
Bedeutung	-	-	-	-	-	-	-	<i>generic</i>

Die nicht unterstützten Bits (-) werden immer als '0' zurückgegeben.

<i>generic</i>	Fehler
0	kein Fehler
1	generic Error

Folgende Meldungen sind möglich:

00 _h	keine Fehler
01 _h	generic Error

2.3.3 Manufacturer Status Register 1002_h

INDEX	1002_h
Name	<i>manufacturer status register</i>
Data Type	unsigned 32
Default Value	No

Die Bits des Status-Registers sind wie folgt belegt:

Register-Bit (Assembler)	Beschreibung	Pegeluordnung	
D25...D31	nicht belegt	0	(immer '0')
D24	I ² C-Error	0	kein I ² C-Error
		1	I ² C-Error
D3...D23	nicht belegt	0	(immer '0')
D2	Watchdog-Reset	0	kein Reset
		1	Watchdog-Reset wurde durchgeführt
D1	Nodeguard-Enabled	0	Nodeguarding ausgeschaltet
		1	Nodeguarding eingeschaltet
D0	Heartbeat-Enabled	0	Heartbeat-Funktion ausgeschaltet
		1	Heartbeat-Funktion eingeschaltet

2.3.4 Pre-defined Error Field 1003_h

INDEX	1003_h
Name	<i>pre-defined error field</i>
Data Type	unsigned 32
Default Value	No

Im *pre-defined error field* wird eine Liste der zuletzt aufgetretenen Fehler gespeichert. Der Subindex 0 enthält die aktuelle Anzahl der in der Liste gespeicherten Fehler. Unter Subindex 1 wird der zuletzt aufgetretene Fehler abgelegt. Tritt ein neuer Fehler auf, so wird der vorhergehende Fehler auf Subindex 2 gespeichert und der neue Fehler unter Subindex 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 16 Fehlereinträge. Beim Eintreten des 17. Fehlers wird der älteste Fehlereintrag gelöscht. Zum Löschen der gesamten Fehler-Liste ist der Subindex '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 [Hex]	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Default [Dez]	Datentyp	R/W
1003	0	<i>no_of_errors_in_list</i>	0, 1...10	-	unsigned 8	rw
	1	<i>error-code n</i>	0...FFFFFFFF	-	unsigned 32	ro
	2	<i>error-code (n-1)</i>	0...FFFFFFFF	-	unsigned 32	ro
	:	:	:	:	:	ro
	16	<i>error-code (n-15)</i>	0...FFFFFFFF	-	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

error-code x Der 32-Bit lange Fehler-Code setzt sich aus dem im DS-301, Tabelle 21 aufgeführten CANopen-Emergency-Error-Code und den von esd definierten Fehler-Codes (Manufacturer-Specific Error Field) zusammen.

Bit:	31 16	15 0
Inhalt:	<i>manufacturer-specific error field</i>		<i>emergency-error-code</i>	

manufacturer-specific error field: bei CAN-CBM-AI410 immer '00'

emergency-error-code: Es werden nur die folgenden Fehler-Codes unterstützt:
 0000_h - Emergency-Error-Reset or no Error
 1000_h - Emergency Generic Error
 (Fehler des I²C-EEPROMs)

Emergency-Frame

Die Daten des von der CAN-CBM-AI410 gesendeten Emergency-Frames sind wie folgt aufgebaut:

Byte:	0	1	2	3	4	5	6	7
Inhalt:	<i>emergency-error-code</i> (siehe oben)		<i>error-register</i> 1001 _h	<i>no_of_errors_in_list</i> 1003,01 _h	<i>manufacturer status register</i> 1002 _h			

2.3.5 COB-ID of SYNC-Message 1005_h

INDEX	1005_h
Name	<i>COB-ID SYNC message</i>
Data Type	unsigned 32
Default Value	80 _h

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.

2.3.6 Manufacturer's Device Name 1008_h

INDEX	1008_h
Name	<i>manufacturer's device name</i>
Data Type	visible string
Default Value	string: 'CAN-CBM-AI410'

Der Datenaustausch beim Upload des Strings muss, wie in der folgenden Tabelle dargestellt, ablaufen (alle Zahlenwerte hexadezimal). In den übergebenen Kommandos müssen immer 8 Datenbytes übertragen werden.

	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Bemerkung
Client:	40	08	10	00	00	00	00	00	upload request index 1008 _h , sub-index=0
CBM-AI410:	41	08	10	00	0C	00	00	00	upload ackn., len = 12
Client:	60	00	00	00	00	00	00	00	segment upload request
CBM-AI410:	00	43	41	4E	2D	43	42	4D	'CAN-CBM', len = 7
Client:	70	00	00	00	00	00	00	00	-
CBM-AI410:	15	2D	41	49	34	31	30	00	'-AI410', len = 5, ended

Eine ausführliche Beschreibung des Domain Uploads ist der CiA DS202-2 (CMS-Protocol Specification) zu entnehmen.

2.3.7 Manufacturer's Hardware Version 1009_h

INDEX	1009_h
Name	<i>manufacturer's hardware version</i>
Data Type	visible string
Default Value	string: z.B. '1.1'

Das Lesen der Hardware-Version erfolgt ähnlich wie das Lesen des Manufacturer Device Names über das Domain Upload Protokoll. Eine ausführliche Beschreibung des Uploads ist der CiA DS202-2 (CMS-Protocol Specification) zu entnehmen.

2.3.8 Manufacturer's Software Version 100A_h

INDEX	100A_h
Name	<i>manufacturer's software version</i>
Data Type	visible string
Default Value	string: z.B.: '1.0'

Das Lesen der Software-Version erfolgt ähnlich wie das Lesen des Manufacturer Device Names über das Domain Upload Protokoll. Eine ausführliche Beschreibung des Uploads ist der CiA DS202-2 (CMS-Protocol Specification) zu entnehmen.

2.3.9 Guard Time $100C_h$ und Life Time Factor $100D_h$

Das CAN-CBM-AI410-Modul unterstützt das Node-Guarding oder alternativ die Heartbeat-Funktion (siehe Seite 24).

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	<i>guard time</i>
Data Type	unsigned 16
Default Value	0 [ms]
Minimum Value	0
Maximum Value	$FFFF_h$ (65,535 s)

INDEX	$100D_h$
Name	<i>life time factor</i>
Data Type	unsigned 8
Default Value	0
Minimum Value	0
Maximum Value	FF_h

2.3.10 Store Parameters 1010_h

Mit diesem Kommando werden die Parameter im EEPROM gespeichert. Es wird nur das Kommando 'Save all Parameters' unterstützt.

Beim Schreiben des Index muss die unten angegebene Byte-Folge gesendet werden. Das Lesen des Index gibt Informationen über die implementierten Speicher-Funktionen zurück (näheres hierzu siehe CiA DS-301).

INDEX	1010_h
Name	<i>store parameters</i>
Data Type	unsigned 32

Index [Hex]	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Datentyp	R/W
1010	0	<i>number_of_entries</i>	3 _h	unsigned 8	ro
	1	<i>save_all_parameters</i>	no default, write: 65 76 61 73 h (= ASCII: 'e' 'v' 'a' 's')	unsigned 32	rw
	2	<i>save_communication_parameter</i>	no default, write: 65 76 61 73 h (= ASCII: 'e' 'v' 'a' 's')	unsigned 32	rw
	3	<i>save_application_parameter</i>	no default, write: 65 76 61 73 h (= ASCII: 'e' 'v' 'a' 's')	unsigned 32	rw

Parameter die abgespeichert bzw. geladen werden können:

Communication Parameter:

- 1005_h *COB-ID Sync*
- 100C_h *Guard Time*
- 100D_h *Life Time Factor*
- 1017_h *Producer Heartbeat Time*
- 1020_h Verify Configuration: *Configuration_Date, Configuration_Time*
- 1029_h Error Behaviour Object: *Communication_Error*

Application Parameter:

- 6421_h Interrupt Trigger Selection: *Analog_Input_1, Analog_Input_2, Analog_Input_3, Analog_Input_4*
- 6426_h Analog Input Interrupt Delta unsigned: *Analog_Input_1, Analog_Input_2, Analog_Input_3, Analog_Input_4*

2.3.11 Restore Default Parameters 1011_h

Communication Profile Area

Mit diesem Kommando werden die bei der Auslieferung aktiven Default-Parameter wieder aktiviert. Alle individuellen Einstellungen, die im EEPROM gespeichert worden sind, gehen verloren. Es wird nur das Kommando 'Restore all Parameters' unterstützt.

Beim Schreiben des Index muss die unten angegebene Byte-Folge gesendet werden. Das Lesen des Index gibt Informationen über die implementierten Restore-Funktionen zurück (näheres hierzu siehe CiA DS-301).

INDEX	1011_h
Name	<i>restore default parameters</i>
Data Type	unsigned 32

Index [Hex]	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Datentyp	R/W
1011	0	<i>number_of_entries</i>	3	unsigned 8	ro
	1	<i>load_all_default_parameters</i>	no default, write: 64 61 6F 6C h (= ASCII: 'd' 'a' 'o' 'l')	unsigned 32	rw
	2	<i>load_communication_parameter</i>	no default, write: 64 61 6F 6C h (= ASCII: 'd' 'a' 'o' 'l')	unsigned 32	rw
	3	<i>load_application_parameter</i>	no default, write: 64 61 6F 6C h (= ASCII: 'd' 'a' 'o' 'l')	unsigned 32	rw

Die Communication Parameter und die Application Parameter, die gespeichert, bzw. geladen werden können, sind in Objekt 1010_h (s.S. 21) aufgelistet.

2.3.12 COB_ID Emergency Object 1014_h

INDEX	1014_h
Name	<i>COB-ID emergency object</i>
Data Type	unsigned 32
Default Value	80 _h + Node-ID

Dieses Objekt bestimmt die COB-ID des Emergency Objektes (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.

2.3.13 Producer Heartbeat Time 1017_h

INDEX	1017_h
Name	<i>producer heartbeat time</i>
Data Type	unsigned 16
Default Value	0 ms

Hier wird die Zeit eingetragen, mit der das CAN-CBM-AI410-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 32).

Wird die Producer-Heartbeat-Time auf '0' gesetzt, so wird das Senden des Heartbeats durch dieses Modul beendet.

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-Identifizier (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.

Jedes Modul kann Heartbeat-Producer und Heartbeat-Consumer sein. Es kann in einem CAN-Netz mehrere Heartbeat-Producer und mehrere Heartbeat-Consumer geben. Das CAN-CBM-AI410-Modul arbeitet ausschließlich als Heartbeat-Producer.

Index [Hex]	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Default [Dez]	Datentyp	R/W
1017	0	<i>producer-heartbeat_time</i>	0..FFFF	0 ms	unsigned 16	rw

producer-heartbeat_time Zykluszeit 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.

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:

$(\text{ASCII-Code}) + 80\text{h} = \text{gelesenes_Byte}$

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

Beispiel:

Wird der Wert 'C1C2 0105h' gelesen, so entspricht dies der Hardware-Seriennummer-Kennung 'AB 0105'. Dieser Wert muß der aufgeklebten Seriennummer des Moduls entsprechen.

2.3.15 Verify Configuration 1020_h

INDEX	1020_h
Name	<i>verify configuration</i>
Data Type	unsigned 32
Default Value	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 [Hex]	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Default [Dez]	Datentyp	R/W
1020	0	<i>no_of_entries</i>	2	2	unsigned 8	ro
	1	<i>configuration_date</i>	0...FFFFFFFF	0	unsigned 32	rw
	2	<i>configuration_time</i>	0...FFFFFFFF	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.

2.3.16 Error Behaviour Object 1029_h

INDEX	1029_h
Name	<i>error behaviour object</i>
Data Type	Byte
Default Value	No

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

Index [Hex]	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Default [Dez]	Datentyp	R/W
1029	0	<i>no_of_error_classes</i>	1	1	Byte	ro
	1	<i>communication_error</i>	0, 1, 2	0	Byte	rw

Bedeutung der Variablen:

no_of_error_classes Anzahl der Fehler-Klassen (hier immer '1')

communication_error 0 - pre-operational (nur, wenn aktueller Zustand = operational)
 1 - no state change
 2 - stopped

2.3.17 Objekt Transmit PDO Communication Parameter 1800_h

Mit diesem Objekt werden die Eigenschaften eines Sende-PDO definiert.

INDEX	1800_h
Name	<i>transmit PDO parameter</i>
Data Type	PDOCommPar

Index	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Default [Hex]	Datentyp	R/W
1800	0 _h	number of entries	5	5 _h	unsigned 8	const.
	1 _h	COB-ID used by PDO	0...FFFFFFFF	180 _h + Node-ID	unsigned 32	ro
	2 _h	transmission type	0...FF	255 _d	unsigned 8	rw
	5 _h	event timer	0...FFFFFFFF	0	unsigned 16	rw

Wertebereich nach DS301

2.3.18 Transmit PDO Mapping Parameter 1A00_h

Mit dem Objekt 'Transmit PDO Mapping Parameter 1A00_h' kann die Zuordnung der Sendedaten zu den Tx-PDOs verändert werden.

INDEX	1A00_h
Name	<i>transmit PDO mapping</i>
Data Type	PDO Mapping

Die folgende Tabelle zeigt die Belegung der Transmit PDO Mapping Parameter für die Default-Konfiguration:

Index	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Default [Hex]	Datentyp	R/W
1A00	0 _h	number of entries	4	4 _h	unsigned 8	ro
	1 _h	read_analog_Input 16 Bit, subindex 1, A_In 1	0...FFFFFFFF	64010110 _h	unsigned 32	ro
	2 _h	read_analog_Input 16 Bit, subindex 2, A_In 2	0...FFFFFFFF	64010210 _h	unsigned 32	ro
	3 _h	read_analog_Input 16 Bit, subindex 3, A_In 3	0...FFFFFFFF	64010310 _h	unsigned 32	ro
	4 _h	read_analog_Input 16 Bit, subindex 4, A_In 4	0...FFFFFFFF	64010410 _h	unsigned 32	ro

Wertebereich nach DS301

2.4 Unterstützte Übertragungsarten nach DS301, Tabelle 10-7

<i>Transmission Type</i>	PDO transmission					Supported by CAN-CBM- AI410
	cyclic	acyclic	synchronous	asynchronous	RTR only	
0	-	X	X	-	-	Yes
1...240	X		X	-	-	Yes
241...251	reserved					No
252	-	-	X	-	X	Yes
253	-	-	-	X	X	Yes
254	-	-	-	X	-	Yes
255	-	-	-	X	-	Yes

2.5 Node Guarding / Life Guarding

Zur gegenseitigen Überwachung der CANopen-Module (insbesondere zum Erkennen von Verbindungsausfällen) können entweder die Heartbeat-Funktion (siehe Objekt 1017_h) oder das Node/Life Guarding genutzt werden. Beim CAN-CBM-AI410-Modul schließen sich die Überwachungsfunktionen gegenseitig aus und können nicht gleichzeitig genutzt werden.

Im Gegensatz zum Node Guarding/Life Guarding kommt die Heartbeat-Funktion ohne RTR-Frames aus.

In der CANopen-Terminologie ist das Node-Guarding die Überwachung der NMT-Slave-Module (hier CAN-CBM-AI410) durch den NMT-Master, das Life-Guarding läuft dagegen im NMT-Slave ab und überwacht den NMT-Master.

Node-Guarding:

Der NMT-Master ruft mit einer RTR-Message (CAN-Request) zyklisch ein besonderes Telegramm von seinen NMT-Slaves (hier CAN-CBM-AI410) ab. Stimmt die Antwort der Slaves nicht mit der erwarteten Antwort überein, oder erfolgt die Antwort nicht innerhalb der *guard time*, erkennt der NMT-Master einen Fehler und reagiert entsprechend. Im Antworttelegramm sind der Modulstatus und ein Toggle-Bit enthalten.

Life-Guarding:

Die NMT-Slaves überwachen, ob sie im Rahmen des Node-Guardings vom NMT-Master abgefragt werden. Bleiben diese Abfragen für die Zeitdauer der *life time* aus, so wird ein lokaler RESET ausgelöst.

Die Life Time ist das Produkt aus *guard time* (Index 100C_h) und dem *life time faktor* (Index 100D_h).

3. Device Profile Area

3.1 Übersicht

Das Modul unterstützt nur die in der folgenden Tabelle aufgeführten Objekte.

Index [Hex]	Name	Subindex [Hex]	Typ	Zugriff	Default [Hex]
6401	Read Analog Input 16-bit	0...4 _h	Integer 16	ro	No
6421	Analog Interrupt Trigger Selection	0...4 _h	Unsigned 8	ro	No
6422	Analog Input Interrupt Source	0 _h , 1 _h	Unsigned 32	ro	No
6423	Analog Input Global Interrupt Enable	0 _h	Boolean	rw	False
6426	Analog Input Interrupt Delta Unsigned	0...4 _h	Unsigned32	rw	0

ro - Read Only, rw - Read/Write

3.1.1 Read Analog Input 16-bit 6401_h

INDEX	6401_h
Name	<i>read analog input</i>
Data Type	Integer 16
Default Value	No

Das Objekt liest die Werte der Analogen Eingänge 1...4.

Index [Hex]	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Default [Hex]	Datentyp	R/W
6401	0	<i>Anzahl der Einträge</i>	4	-	Byte	ro
	1	<i>Read_Analog_Input_1</i>	0000...7FE0	-	Integer16	ro
	2	<i>Read_Analog_Input_2</i>	0000...7FE0	-	Integer16	ro
	3	<i>Read_Analog_Input_3</i>	0000...7FE0	-	Integer16	ro
	4	<i>Read_Analog_Input_4</i>	0000...7FE0	-	Integer16	ro

Byte	0								1												
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
Inhalt	Vorzeichen								Analogwert Wertebereich: 0000 _h ... 7FE0 _h								0	0	0	0	0

Die Analogwerte entsprechen jeweiligen Eingangsspannungen im Bereich von 0...10 V. Dabei steht 0000_h für 0V und die maximale Eingangsspannung von 10 V entspricht dem Wert 7FE0_h.

3.1.2 Analog Input Interrupt Trigger 6421_h

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

INDEX	6421_h
Name	<i>interrupt trigger selection</i>
Data Type	unsigned8
Default Value	see table below

Index	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Default [Hex]	Datentyp	R/W
6421	0 _h	<i>Zahl der analogen Eingänge</i>	4	4	unsigned 8	ro
	1 _h	<i>Analog_Input_1</i>	0...FF	0	unsigned 8	rw
	2 _h	<i>Analog_Input_2</i>	0...FF	0	unsigned 8	rw
	3 _h	<i>Analog_Input_3</i>	0...FF	0	unsigned 8	rw
	4 _h	<i>Analog_Input_4</i>	0...FF	0	unsigned 8	rw

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	nicht unterstützt	

3.1.3 Analog Input Interrupt Source 6422_h

INDEX	6422_h
Name	<i>analog input interrupt source</i>
Data Type	unsigned 32
Default Value	No

Dieses Objekt zeigt an, welcher Kanal einen Interrupt ausgelöst hat.

Index [Hex]	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Default [Hex]	Datentyp	R/W
6422	0	<i>no_of_interrupt_source</i>	1	1	unsigned 8	ro
	1	<i>interrupt_source</i>	0..F	-	unsigned 32	ro

Bedeutung der Variablen:

interrupt_source Diese Variable zeigt an, welcher Kanal einen Interrupt ausgelöst hat.

- 1... Interrupt ausgelöst
- 0... kein Interrupt

Die Werte für die 4 Kanäle werden in dem untersten Byte übergeben:

Bit:	7	6	5	4	3	2	1	0
Inhalt:	0	0	0	0	<i>Kanal_4</i>	<i>Kanal_3</i>	<i>Kanal_2</i>	<i>Kanal_1</i>

3.1.4 Global Interrupt Enable 6423_h

INDEX	6423_h
Name	<i>global interrupt enable</i>
Data Type	Boolean
Default Value	see table below

Mit dem Objekt Analog Input Global Interrupt Enable wird die Interrupt Funktion des Moduls freigegeben oder unterbunden. Die Werte der Objekte 6421_h, 6422_h und 6426_h bleiben davon unberührt. Der Default-Wert ist so eingestellt, das bei einer Änderung des analogen Eingangswertes kein Interrupt ausgelöst wird.

Index [Hex]	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Default [Hex]	Datentyp	R/W
6423	0	<i>Analog_Input_Global_Interrupt_Enable</i>	True, False	False	Boolean	rw

True... Global Interrupt enabled
 False... Global Interrupt disabled

3.1.5 Analog Input Interrupt Delta 6426_h

INDEX	6426_h
Name	<i>analog input interrupt delta unsigned</i>
Data Type	Unsigned32
Default Value	No

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.

Index [Hex]	Subindex [Dez]	Beschreibung	Wertebereich [Hex]	Default [Hex]	Datentyp	R/W
6426	0	<i>Anzahl_Analog_Inputs</i>	4	4	Unsigned32	ro
	1	<i>Analog_Input_1</i>	0...FFFF	0	Unsigned32	rw
	2	<i>Analog_Input_2</i>	0...FFFF	0	Unsigned32	rw
	3	<i>Analog_Input_3</i>	0...FFFF	0	Unsigned32	rw
	4	<i>Analog_Input_4</i>	0...FFFF	0	Unsigned32	rw

4. PDO-Belegung

Die PDOs (Process Data Objects) dienen zur Übertragung der Prozessdaten:

Tx-PDO	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
180 + Node-ID	Analog_Input 1		Analog_Input 2		Analog_Input 3		Analog_Input 4	

Diese Seite ist bewusst unbedruckt.

5. Schnellstart

5.1 Konfiguration für den Einsatz im CANopen-Netzwerk

Für einen schnellen Start mit der einfachsten Konfiguration sind folgende Schritte nötig:

1. CAN-Verdrahtung aufbauen (Abschlüsse nicht vergessen!)
2. Baudrate einstellen:
(Nur wenn eine andere als die Default-Einstellung gewünscht wird.)
Die Default-Bitrate beträgt 125 kBit/s. Die Bitrate kann mit den Kodierschaltern (SW100, SW101), wie im Hardware-Handbuch (Kapitel: Konfiguration über die Drehschalter) beschrieben, gesetzt werden.
3. Modul-Nummer (Node-ID) einstellen:
Sofern eine andere als die eingestellte Modul-Nummer gewünscht wird, oder die Baudrate verstellt wurde, muss die Modul-Nummer mit den Kodierschaltern (SW100, SW101), wie im Hardware-Handbuch (Kapitel Konfiguration über die Drehschalter) beschrieben, gesetzt werden.
Die Modul-Nummer darf Werte zwischen 1 und 127 ($01...7F_h$) annehmen.

Beispiel: Modul-Nummer 1 (01_h) einstellen:

- Versorgungsspannung aus
- Kodierschalter SW100 (Low) auf '1', SW 101 (High) auf '0' drehen
- Versorgungsspannung ein, Modul wacht im Zustand *Preoperational* auf

4. Modul einschalten:

Versorgungsspannung anlegen

Wurde zuvor die Modul-Nummer eingestellt, liegt die Versorgungsspannung bereits an, und das Modul befindet sich im Zustand *Preoperational* (siehe Hardware-Handbuch Kapitel: Module-Status-LEDs).

5. Modul starten mit:

CAN-Identifizier	Len	Data	
0	2 Byte	01	00

6. Status Analoge Eingänge abfragen mit:

CAN-Identifizier	RTR	Len
181	1	0

7. Modul antwortet mit:

CAN-Identifizier	Len	Data
181	8 Byte	abhängig vom Zustand der Eingänge

5.2 Tabelle der wichtigsten Identifier und Nachrichten für CANopen

CAN-Identifier [Hex]	Bezeichnung	Länge	Daten [Hex]	Bemerkungen
0	NMT	2	02 xx	CAN-CBM-AI410-Modul geht in Zustand <i>Stopped</i>
0	NMT	2	01 xx	Start (CAN-CBM-AI410-Modul geht in Zustand <i>Operational</i>)
0	NMT	2	80 xx	CAN-CBM-AI410-Modul geht in Zustand <i>Preoperational</i>
0	NMT	2	81 xx	Reset CAN-CBM-AI410-Modul
0	NMT	2	82 xx	Reset Communication (hier gleiche Funktion wie '81xx')
80 _h	NMT	0	-	Sync an alle
80 _h + Node-ID	EMCY	8	siehe S. 14	Emergency-Frame von CAN-CBM-AI410
700 _h + Node-ID	NMT	8 Bytes	Identifier	NMT error control identifier (siehe DS301)
580 _h + Node-ID	SDO	8 Bytes	Parameter	Quittierung der Kommunikationsparameter von CAN-CBM-AI410-Modul (Tx)
600 _h + Node-ID	SDO	8 Bytes	Parameter	Übertragung der Kommunikationsparameter zur CAN-CBM-AI410-Modul (Rx)
180 _h + Node-ID	Tx_PDO_1	1 Byte	Nutzdaten	(Tx/Sende-PDO)

xx = Node-ID (Modul-Nummer)

Node-ID = 1...7F_h