HYDAC ELECTRONIC

Protokoll-Beschreibung CANopen

HDA 4000/7000

(Originalanleitung)

Mat Nr..: 669821 / Stand: 2014/01/30

Inhaltsverzeichnis

Vorwort

Für Sie, den Benutzer unseres Produktes, haben wir in dieser Dokumentation die wichtigsten Hinweise zum Bedienen und Warten zusammengestellt.

Sie dient Ihnen dazu, das Produkt kennen zu lernen und seine bestimmungsgemäßen Einsatzmöglichkeiten optimal zu nutzen.

Diese Dokumentation muss ständig am Einsatzort verfügbar sein.
Bitte beachten Sie, dass die in dieser Dokumentation gemachten Angaben der Softwaretechnik zu dem Zeitpunkt der Literaturerstellung entsprechen.

Entdecken Sie beim Lesen dieser Dokumentation Fehler oder haben weitere Anregungen und Hinweise, so wenden Sie sich bitte an:

HYDAC ELECTRONIC GMBH
Technische Dokumentation
Hauptstraße 27
66128 Saarbrücken
-Deutschland-
Tel: +49(0)6897 / 509-01
Fax: +49(0)6897 / 509-1726
Email: electronic@hydac.com

Die Redaktion freut sich über Ihre Mitarbeit.

„Aus der Praxis für die Praxis“

1 Einleitung

Die Druckmessumformer HDA 4000/7000 CANopen (PT) entsprechen dem CANopen Standard gemäß folgenden Profilen und Standards:

- CiA Draft Standard 301
  Application Layer and Communication Profile
  Version 4.02, Date: 13 February 2002
- CiA Draft Standard 404
  Device Profile Measuring Devices and Closed-Loop Controllers
  Version 1.2, Date: 15. May 2002
- CiA Draft Standard Proposal 305
  Layer setting services (LSS) and protocols
  Version: 2.2 , 26 August 2008
- CiA Application Note 801
  CANopen Automatic bit-rate detection -Recommended practice and application hints
  Version 1.0 01 January 2005

Dieses Handbuch beschreibt die von HDA 4000/7000 CANopen (PT) unterstützten Funktionen. Dabei werden Grundkenntnisse von CAN und CANopen vorausgesetzt. Die genaue Funktionsweise ist in den *oben genannten Dokumenten* beschrieben. Da diese in Englisch abgefasst sind, werden die in diesem Handbuch beschriebenen Dinge, zur besseren Zuordnung, mit dem in der Spezifikation genannten englischen Begriff bezeichnet und *kursiv* dargestellt.

2 Funktionen der HDA 4000/7000 CANopen (PT)

- Erfassung des aktuellen Druckwertes und / oder Temperaturwertes mit:
  Druck:
  - 1kHz Sample-Rate
  - 0,2% Genauigkeit
  - 12-Bit Auflösung

  Temperatur:
  - 1kHz Sample-Rate
  - 1°C Genauigkeit
- Umrechnung des Druck-/Temperaturwertes in einen beliebig skalierbaren, linearen Prozesswert.
- Senden des aktuellen Prozesswertes als PDO bei folgenden Ereignissen:
  - Synchron über empfangene SYNC-Objekte
  - Asynchron zyklisch im Bereich von 1 Millisekunde bis 1 Minute
  - Bei Messwertänderungen über eine einstellbare Differenz
  - Beim Überschreiten einer einstellbaren Grenze
  - Beim Unterschreiten einer einstellbaren Grenze

3 Übertragungsraten

Der HDA 4000/7000 CANopen (PT) unterstützt folgende Übertragungsraten (Baudraten):

- 1 Mbit/s
- 800 kbit/s
- 500 kbit/s
- 250 kbit/s
- 125 kbit/s
- 50 kbit/s
- 20 kbit/s
- 10 kbit/s

Das Timing entspricht der DS301, *Bit rates and timing*.

Die verwendete Übertragungsrate ist in einem nicht-flüchtigen Speicher hinterlegt. Sie ist im Auslieferzustand auf 250 kbit/s eingestellt und kann über den CAN-Bus geändert werden (Siehe *Object Dictionary Index 2002.*
Zusätzlich ist eine automatische Bitratensuche möglich.

4 CAN-Frames

Der HDA 4000/7000 CANopen (PT) unterstützt die in der Spezifikation geforderten 11-bit Standard – Frames mit 11-bit Identifier. Extented Frames mit 29-bit Identifier werden nicht unterstützt, aber toleriert. Das bedeutet, dass Extented Frames nicht erkannt werden, aber auch nicht zu Fehlern führen.

5 Node-ID

Zum Betrieb des HDA 4000/7000 CANopen (PT) in einem CANopen-Netzwerk ist es notwendig, dass eine innerhalb des Netzes einmalige *Node-ID* eingestellt wird.

Die eingestellte *Node-ID* ist wie die Übertragungsrate in einem nicht-flüchtigen Speicher hinterlegt und kann ebenfalls über den CAN-Bus (siehe *Object Dictionary Index 2001*) eingestellt werden. Im Auslieferzustand ist die Adresse 1 eingestellt.

6 Übertragungsdienste

6.1 Service Data Object (SDO)

Bei CANopen werden alle Daten eines Gerätes (Einstellparameter und Messdaten) in einem *Object Dictionary* unter einem definierten *Index* abgelegt. Manche Einträge des *Object Dictionary* werden mit einem *Subindex* noch weiter untergliedert. Mit den *SDOs* können nun andere Netzteilnehmer das *Object Dictionary* des HDA 4000/7000 CANopen (PT) auslesen oder beschreiben.
Der HDA 4000/7000 CANopen (PT) übernimmt dabei die Rolle eines *Servers*, das Gerät, dass die Daten auslesen oder beschreiben will, die eines *Clients*.

Zum Transfer von Daten muss der HDA 4000/7000 CANopen (PT) ein *Receive-SDO* besitzen, mit dem er Daten empfängt und ein *Transmit-SDO*, mit dem er die Daten sendet. Ablauf des Datentransfers:

Auslesen des Object Dictionary :

1. Ein Gerät (*Client*) sendet das *Receive-SDO* des HDA 4000/7000 CANopen (PT) (*Server*).
   In diesem *SDO* befindet eine Kennung, dass das *Object Dictionary* gelesen werden soll, sowie der gewünschte *Index* und *Subindex*.
2. Der HDA 4000/7000 CANopen (PT) (*Server*) sendet sein *Transmit-SDO*. In diesem befindet sich ebenfalls der *Index* und der *Subindex*, sowie die gelesenen Daten.

Beschreiben des Object Dictionary :

1. Ein Gerät (Client) sendet das Receive-SDO des HDA 4000/7000 CANopen (PT) (Server).
   In diesem SDO befindet eine Kennung, dass das Object Dictionary beschrieben werden soll, sowie der gewünschte Index, Subindex und die einzuschreibenden Daten.
2. Der HDA 4000/7000 CANopen (PT) (Server) sendet sein Transmit-SDO. In diesem befindet sich ebenfalls der Index und der Subindex, sowie eine Kennung, dass das Object Dictionary beschrieben wurde.

Sollte dabei ein Fehler auftreten, z.B. dass der angegebene *Index* nicht existiert, oder dass versucht wurde einen *read only* Eintrag zu beschreiben, oder dass die Daten nicht innerhalb des Gültigkeitsbereiches lagen, so enthält das Transmit-SDO eine entsprechende *Abort SDO Transfe*r - Kennung und einen entsprechenden *Abort Code* (siehe [1])

Das Objektverzeichnis des HDA 4000/7000 CANopen (PT) ist so aufgebaut, dass jeder Eintrag maximal 4 Byte Daten belegt, so dass alle Daten *expedited* übertragen werden können. Das heißt, dass alle Daten in ein einziges *SDO* gepackt werden, was zu besonders effektiven Übertragungen führt.

Die jeweilige *COB-ID* des *SDO* entspricht dem in der DS301 festgelegten *Pre defined Connection Set* und ist nicht änderbar.

COB-IDs für Service Data Objects

SDO	***COB-ID***
Receive – SDO	600h+Node-ID
Transmit – SDO	580h+Node-ID

6.2 Process Data Object (PDO)

Die Datenübertragung mittels *SDOs* ist zwar sehr flexibel, hat aber für die Übertragung von Messwerten oder Stellgrößen einige Nachteile: Es kann nur ein Datum gelesen werden, die Daten müssen erst mit einem *SDO* angefordert werden und dadurch dass der jeweilige *Index* und *Subindex* mit übertragen wird, steigt der so genannte Overhead weiter.

Aus diesem Grund definiert CANopen so genannte *Process Data Objects*. Diese enthalten nur die notwendigen Nutzdaten. Es gibt zwei Arten von *PDOs*:

1. *Transmit-PDOs*
   Hiermit kann ein Messgerät seine Messwerte senden.
2. *Receive-PDOs*
   Hiermit können einem Stellglied oder einer Regelungseinheit die Stellgrößen übertragen werden.

Welche Daten sich nun in einem *PDO* befinden wird durch das so genannte *PDO-Mapping* festgelegt. Dieses *PDO-Mapping* ist im *Object Dictionary* hinterlegt (siehe *Object Dictionary, Index 1A00*).

Mit welcher ID und bei welchem Ereignis ein *PDO* übertragen wird, ist in dem *PDO-Transmission Type* festgelegt. Diese Einstellungen befinden sich ebenfalls im *Object Dictionary* hinterlegt (siehe *Object Dictionary, Index 1800*).

Ereignisse, die zum Senden eines *PDOs* führen:

1. Empfang eines *SYNC* Objektes (Synchroner Transfer).
2. Ablauf einer einstellbaren Zykluszeit im Bereich von 1 Millisekunde bis 1 Minute (Zyklischer Transfer).
3. Der Messwert hat sich um einen einstellbaren Betrag gegenüber dem letzten PDO geändert.
4. Der Messwert hat eine einstellbare Grenze überschritten.
5. Der Messwert hat eine einstellbare Grenze unterschritten.

Der HDA 4000/7000 CANopen (PT) implementiert ein *Transmit-PDO*, welches den aktuellen Druckwert und den Status des Signaleinganges überträgt.

Die DS404 sieht als Standardeinstellung die Übertragung des aktuellen Messwertes als 32-bit Wert, sowie den Status als 8-bit Wert vor. Es ist aber möglich, diese Einstellung über das *PDO-Mapping* dahingehend zu ändern, dass der Messwert als 16-bit-Wert übertragen wird. Die Übertragung des Status kann ebenfalls durch ein geändertes *PDO-Mapping* unterbunden werden.

6.3 Synchronisation Object (SYNC)

SYNC Objekte dienen zur Realisierung eines synchronen Datentransfers. Ein *SYNC* Objekt ist im Prinzip eine CAN Nachricht mit einem definierten Identifier, ohne Daten. CANopen unterscheidet zwischen *SYNC Producer* und *SYNC Consumers. SYNC Producer* sind Geräte am Bus, die in einstellbaren Zeitabständen ein *SYNC* senden. *SYNC Consumers* sind Geräte, die auf den Empfang eines *SYNC* reagieren. In einem CANopen Netzwerk können mehrere *SYNC* Objekte existieren. Unterschieden werden die einzelnen *SYNC* Objekte anhand der *SYNC-ID*, welche dem verwendeten CAN Identifier entspricht. Die verwendete *SYNC-ID* ist im *Object Dictionary* hinterlegt.

Der HDA 4000/7000 CANopen (PT) bietet die Funktionalität eines *SYNC Consumers*. Bei entsprechender Einstellung des *PDO Transmission Type* wird beim Empfang eines *SYNC* ein *PDO* gesendet. Die *SYNC-ID* ist auf 80h voreingestellt und kann im *Object Dictionary* geändert werden (siehe *Object Dictionary, Index 1005*). Unter *PDO Transmission Type* kann die Anzahl der empfangenen *SYNC* Objekte, die zum Senden eines *PDO*, führt eingestellt werden.

6.4 Emergency Object (EMCY)

EMCY Objekte werden beim Auftreten eines Fehlers gesendet. *EMCY* Objekte enthalten einen *Emergency Error Code,* den Inhalt des *Error register* sowie ein *Manufacturer specific Error Field*. Ist ein gemeldeter Fehler beseitigt oder verschwunden, so wird dies ebenfalls durch ein spezielles *EMCY* Objekt gemeldet.

Eine Emergency-Nachricht wird gesendet, wenn ein Fehler auftritt oder dieser Fehler wieder verschwindet. Die Nachricht ist folgendermaßen aufgebaut:

EmergencyErrorCode (hex)	**ManufacturerSpecificErrorField (hex)**	**Beschreibung**
FF00	Bitmaske	Gerätespezifischer Fehler
8100	Fehlerliste	Kommunikationsfehler
0000	0	Kein Fehler vorhanden

Gerätespezifischer Fehler

Bit-Nummer	**Beschreibung**
0	Fehler beim Laden der Anwendereinstellungen
1	Fehler beim Speichern der Anwendereinstellungen
2	Fehler beim Laden der Werkseinstellungen
3	Fehler beim Lesen des Druckwertes
4	Fehler beim Lesen des Temperaturwertes
5	Fehler beim Laden der Kalibrierdaten

Kommunikationsfehler

Fehlercode (hex)	Beschreibung
0x01	Stuff Error
0x02	Form Error
0x03	Ack Error
0x04	Bit1 Error
0x05	Bit0 Error
0x10	Bus Off
0x11	Crc Error

Beispiel

Fehler beim Lesen des Druckwertes

Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
FF	**00**	**00**	**00**	**00**	**00**	**08**	**00**

Das *EMCY* Objekt hat die voreingestellte ID 80h+*Node-ID* und kann im *Object Dictionary* geändert werden (siehe *Object Dictionary, Index 1014*).

6.5 Heartbeat

Mit dem *Heartbeat Protocol* kann eine Überwachung der einzelnen Teilnehmer durchgeführt werden. CANopen unterscheidet zwischen folgenden Funktionen:

1. *Heartbeat Producer*
   sendet in zyklischen Abständen ein *Heartbeat* Objekt.
2. *Heartbeat Consumer*
   überwacht das Senden von bestimmten *Heartbeat* Objekten.

Die Zykluszeit ist im *Object Dictionary* in Millisekunden einstellbar. Eine Zeitangabe von 0 bedeutet „*Heartbeat* nicht aktiv“.

Mit dem *Heartbeat* Objekt wird immer der Status des *Heartbeat Producers* als Byte mit übertragen.

Bedeutung des Heartbeat- Objektinhaltes

Wert	Status	Anmerkung
0	*BOOTUP*	Das Gerät hat gebootet.
4	*STOPPED*	Das Gerät ist gestoppt.
5	*OPERATIONAL*	Das Gerät arbeitet normal.
127	*PRE-OPERATIONAL*	Das Gerät sendet keine *PDOs*, kann aber *SDOs* bearbeiten.

Der HDA 4000/7000 CANopen (PT) kann als *Heartbeat Producer* arbeiten. Die ID des *Heartbeat* ist 700h + *Node-ID*. Die Zeit ist mit 0 (nicht aktiv) voreingestellt und kann geändert werden (siehe *Object Dictionary, Index 1017*).

6.6 Network Management Services (NMT)

NMT Objekte dienen dazu Geräte zu starten, zu stoppen oder zurückzusetzen. CANopen unterscheidet zwischen folgenden Funktionalitäten:

1. *NMT Master*
   steuert andere Knoten.
2. *NMT Slave*
   wird von einem *Master* gesteuert.

In einem CANopen Netzwerk existiert nur ein *NMT* Objekt mit dem Identifier 0. Es werden immer 2 Bytes übertragen. Das erste Byte enthält den *Command Specifier*, der den Befehl repräsentiert, das zweite Byte enthält die *Node-ID* des Knotens, der diesen Befehl ausführen soll. Ein Wert von 0 bedeutet, dass dieser Befehl für alle Knoten gilt. Folgende Befehle sind möglich:

NMT Befehle

1. *Start Remote Node*
   Der Knoten wechselt in den Zustand *Operational*.
2. *Stop Remote Node*
   Der Knoten wechselt in den Zustand *Stopped*.
3. *Enter Pre-Operational*
   Der Knoten wechselt in den Zustand *Pre-Operational*.
4. *Reset Node*
   Der "Device Profile Specific"-OD-Bereich wird zurückgesetzt, die Baudrate wird gegebenenfalls neu initialisiert und wechselt dann in den Zustand Reset Communication.
5. *Reset Communicatiuon*
   Die Kommunikationseinheit des Knotens wird zurückgesetzt und danach wechselt der Knoten in den Zustand *Pre-Operational.*

Der HDA 4000/7000 CANopen (PT) arbeitet als *NMT Slave* und unterstützt alle *NMT* Dienste.

6.7 Boot Up Protocol

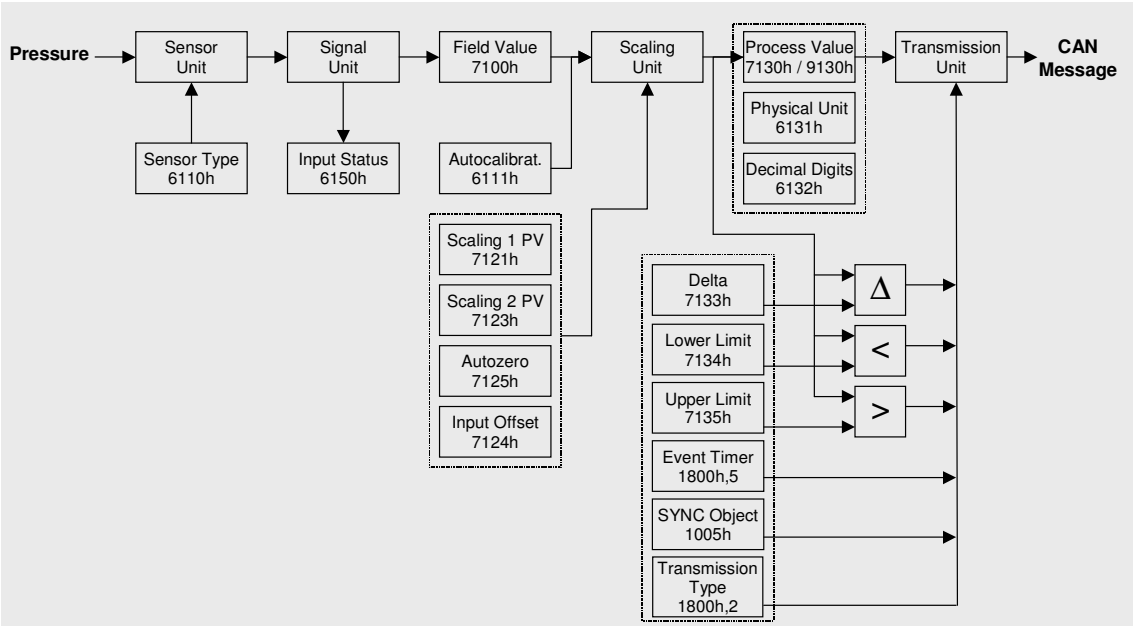
Wenn ein *NMT Slave* nach der Initialisierung in den Zustand *Pre-operational* wechselt, sendet er jeweils ein *Boot Up* Objekt. Dies ist im Prinzip nichts anders als ein *Heartbeat* Objekt mit dem Status 0.

Beim HDA 4000/7000 CANopen (PT) ist diese Funktion realisiert.

7 Datenfluss im HDA 4000/7000 CANopen (PT)

Nachfolgendes Bild zeigt den Datenfluss innerhalb des
HDA 4000/7000 CANopen (PT), sowie die jeweiligen Indizes des *Object Dictionary*.

Datenfluss im HDA 4000/7000 CANopen (PT)



7.1 Sensor Unit

Die Sensoreinheit erfasst den Druck und/oder die Temperatur und wandelt ihn/sie in ein elektrisches Signal um. Der Typ der Sensoreinheit (*Pressure Transducer)* ist in *Sensor Type* hinterlegt.

7.2 Signal Unit

Dieses elektrische Signal wird in der Signaleinheit aufbereitet, in ein digitales Signal umgewandelt und steht als *FieldValue* zur Verfügung. Der Status der Signalaufbereitung ist in *InputStatus* hinterlegt.

7.3 Scaling Unit

Der *FieldValue* wir danach in der *Scaling Unit* in einen *ProcessValue* umgerechnet.

Die Angaben *PhysicalUnit* und *DecimalDigits* beschreiben wie der *ProcessValue* zu interpretieren ist. Steht in *PhysicalUnit* z.B. die Einheit bar und in *DecimalDigits* der Wert 1, so entspricht ein *ProcessValue* von 127 der Messgröße 12,7 bar.

Die Umrechnung des *FieldValue* in den *ProcessValue* erfolgt mit den beiden Skalierwerten *Scaling1PV* und *Scaling2PV*. Der erste Wert gibt an, welcher Prozesswert der unteren Messbereichsgrenze entspricht (z.B. 0 bar). Der zweite Wert gibt an, welcher Prozesswert der oberen Messbereichsgrenze entspricht (z.B. 400 bar).

Mit *InputOffset* kann noch ein zusätzlicher Versatz hinzugerechnet werden. Mit *AutoZero* wird *InputOffset* so gesetzt, dass der aktuelle Prozesswert den Wert 0 (bzw. den minimalen Prozesswert) hat.

Mit *Autocalibration* kann ein Offsetfehler (nur Druck!) der Signalkonditionierung ausgeglichen werden. Dazu ist es notwendig, dass an dem HDA 4000/7000 CANopen (PT) kein Druck ansteht und der Offsetfehler nicht größer als ±3% ist.

7.4 Transmission Unit

Tritt eines der folgenden Ereignisse ein, so wird in Abhängigkeit des eingestellten *TransmissionType* der Wert des PDO gesendet.

1. Der aktuelle *ProcessValue* hat sich um mehr als *Delta* gegenüber dem letzten übertragenen *Process Value* geändert.
2. Die Grenze *LowerLimit* wurde unterschritten.
3. Die Grenze Upper*Limit* wurde überschritten.
4. Der *EventTimer* ist abgelaufen (zyklische Übertragung).
5. Ein oder mehrere *Sync* Objekte wurden empfangen (synchrone Übertragung).

8 Das Object Dictionary

8.1 Aufbau des Object Dictionary

Im *Object Dictionary* sind, wie bereits mehrfach erwähnt, alle Daten hinterlegt. In den folgenden Kapiteln sind die vom HDA 4000/7000 CANopen (PT) unterstützten Einträge aufgeführt. Die Angabe des Index erfolgt spezifikationsgemäß immer in hexadezimaler Notation, ohne dass die hexadezimale Darstellung extra angezeigt wird. Bei jedem Eintrag ist auch die Zugriffsart angegeben. CANopen unterscheidet dabei folgende Zugriffsarten:

Zugriffsarten (*Acces Type*) für das *Object Dictionary*

1. const
   Kann nur gelesen werden und liefert immer den gleichen Wert.
2. read only
   Kann nur gelesen werden, der Wert kann sich aber während des Betriebes ändern.
3. write only
   Der Eintrag kann nur geschrieben werden.
4. read write
   Der Eintrag kann geschrieben und gelesen werden.

CANopen unterscheidet innerhalb des Data Dictionary folgende Bereiche:

Bereiche des Object Dictionary:

5. Index 0 .. 1FFF: *Communication profile specific entries*
   Einstellungen, die für alle CANopen-Geräte gelten. Diese Einträge sind in der DS301 festgelegt.
6. Index 6000 .. 9FFF: *Device profile specific entries*
   Gerätespezifische Daten, die in einer *Draft Standard* festgelegt sind. Der HDA 7000 CAN hat das Geräteprofil DS404 realisiert.
7. Index 2000 .. 5FFF: *Manufacturer specific entries*
   Herstellerspezifische zusätzliche Daten, die in keiner Spezifikation festgelegt sind.

8.2 Struktur des gerätespezifischen Teils nach DS404

Der HDA 4000/7000 CANopen (PT) implementiert die DS404. Diese beschreibt das Verhalten und die Funktionalität von Mess-, Steuerungs- und Regelungsgeräten.

Zunächst einige Anmerkungen zum Layout des *Object Dictionarys*:

Mehrkanaligkeit
Die DS404 ist auf mehrkanalige Geräte ausgerichtet. Das heißt, dass bei allen Einträgen im *Subindex 0* die Anzahl der Kanäle steht und unter dem jeweiligen *Subindex* dann der Wert.
Der HDA 4000/7000 CANopen (PT) realisiert einen Kanal oder zwei Kanäle. Deshalb enthält der *Subindex 0* überall den Wert 1 oder 2 und die eigentlichen Messwerte oder Einstellungen sind unter *Subindex 1* oder *Subindex 2* hinterlegt.

Datentypen
Die DS404 ist so aufgebaut, dass es für alle Datentypen eine eigene Sektion im *Object Dictionary* gibt:

Index	Datentyp
6000 bis 6FFFh	Floating Point oder speziell codierte Daten (Status, etc.)
7000 bis 7FFFh	16 bit Integer
8000 bis 8FFFh	32 bit Integer
9000 bis 9FFFh	24 bit Integer

Funktionsblöcke
Die DS404 unterteilt ein Gerät in verschiedene Funktionsblöcke: Analogeingangsblock, Analogausgangsblock, Digitaleingangsblock, Digitalausgangsblock, Reglerblock und Alarmblock.
Der HDA 4000/7000 CANopen (PT) hat den Analogeingangsteil *(Analogue Input Function Block*) realisiert. Diese Einträge liegen im Bereich X100h bis X1FFh.

Reihenfolge der Einträge
Auf Grund dieser Aufteilung bezieht sich die Reihenfolge der Einträge nur auf den Wert der letzten 3 Stellen des Index. Die erste Stelle spezifiziert den Datentyp und wird nicht als Ordnungskriterium berücksichtigt.

9 Einträge im Object Dictionary

Im Folgenden sind die vom HDA 4000/7000 CANopen (PT) realisierten Funktionalitäten aufgezeigt. Eine detaillierte Beschreibung der Einträge kann in [1] und [2] nachgelesen werden.

9.1 Communication Profile Specific Entries (DS301)

9.1.1 Index 1000: DeviceType (read only)

Enthält die Nummer des verwendeten Geräteprofiles, hier die Nummer 404, sowie die profilspezifische Erweiterung, hier eine 2, für die Unterstützung der Analogen Eingangseinheit.

9.1.2 Index 1001: ErrorRegister (read only)

Enthält den aktuellen Fehlerzustand (siehe *EMCY,* sowie [1]).

9.1.3 Index 1002: ManufacturerStatusRegister (read only)

Enthält verschiedene Errorflags.
In dem 32bit-Wert haben die Bits folgende Bedeutung:

Bit0: 1= Fehler beim Lesen der Setup-Daten
Bit1: 1= Fehler beim Schreiben der Setup-Daten
Bit2: 1= Fehler beim Lesen der Werkseinstellungen
Bit3: 1= Interner Fehler bei der A/D-Wandlung
Bit4: 1= Interner Fehler in der A/D-Ankoppelung
Bit5: 1= Interner Bus-Fehler
Bit6: 1= Fehler bei der internen Programmüberwachung

Die Bits 0 bis 6 werden gelöscht, wenn der Fehler behoben ist. Um festzustellen, ob ein Fehler aufgetreten war, werden diese 7 Bits 0x00 bis 0x06 auf die Bits 0x10 bis 0x16 gespiegelt. Die gespiegelten Bits bleiben gesetzt, wenn der Fehler einmal aufgetreten war.

9.1.4 Index 1005: SyncMessageIdentifier (read write)

Hier kann die *COB-ID* für das *SYNC*-Objekt eingestellt werden.

9.1.5 Index 1008: ManufacturerDeviceName (const)

Liefert den Gerätenamen als Zeichenkette. („HDA4“) bzw. („HDA7“).

9.1.6 Index 1009: ManufacturerHardwareVersion (const)

Liefert die Hardwareversion als Zeichenkette (z.B. „HV02“).

9.1.7 Index 100A: ManufacturerSoftwareVersion (const)

Liefert die Softwareversion als Zeichenkette (z.B. „0101“). Die beiden ersten Zeichen bezeichnen die Version, die letzten beiden Zeichen den Revisionsstand.

9.1.8 Index 1010: StoreParameters

Durch Einschreiben der Zeichenkette „save“ werden die aktuellen Einstellungen in den nicht flüchtigen Speicher übertragen.
Der HDA 4000/7000 CANopen (PT) speichert Einstellungen nicht automatisch wenn sie geändert werden, sondern nur auf Anforderung.

ACHTUNG: Geänderte Einstellungen müssen mit *StoreParameters* explizit gesichert werden, sonst gehen sie beim Abschalten des Gerätes oder bei den *NMT*-Befehlen *Reset Node* und *Reset Communication* verloren.

CANopen bietet die Möglichkeit mit Hilfe verschiedener *Subindexes* verschiedene Parameterbereiche zu sichern. Unterstützt wird hier Subindex 1, 2 und 3.

Nähere Informationen sind [1] zu entnehmen.

Verwendete Subindizes:

0: LargestSubindexSupported (read only)
1: StoreAllParameters (read write)
2: StoreCommunicationParamters (read write) (index von 0x1000 to 0x1FFF)
3: StoreApplicationParamteres (read write) (index von 0x6000 to 0x9FFF)

9.1.9 Index 1011: RestoreDefaultParameters

Durch Einschreiben der Zeichenkette „load“ werden die werksseitigen Voreinstellungen in den nicht flüchtigen Speicher übertragen.
Der HDA 4000/7000 CANopen (PT) arbeitet allerdings bis zum Abschalten oder bis zur Ausführung der Befehle *Reset Node* und *Reset Communication* noch mit den aktuellen Einstellungen weiter.

CANopen bietet die Möglichkeit mit Hilfe verschiedener *Subindexes* verschiedene Parameterbereiche zu restaurieren.
Unterstützt wird hier Subindex 1, 2 und 3.

Nähere Informationen sind [1] zu entnehmen.

Achtung: Die Einstellungen der Baudrate und der NodeID bleiben bei *RestoreDefaultParameter* erhalten.

Verwendete Subindizes:

0: LargestSubindexSupported (read only)
1: RestoreAllDefaultParameters (read write)
2: RestoreCommunicationParamters (read write) (index von 0x1000 to 0x1FFF)
3: RestoreApplicationParamteres (read write) (index von 0x6000 to 0x9FFF)

9.1.10 Index 1014: CobIdEmergencyMessage (read write)

Hier kann die *COB-ID* für das *EMCY*-Objekt eingestellt werden (siehe *EMCY*).

9.1.11 Index 1015: InhibitTime (read write)

Die Inhibit Time gibt den Zeitraum an, den der Knoten bei Auftreten eines kritischen Fehlers wartet, bis er das Emergency-Objekt verschickt.

9.1.12 Index 1017: ProducerHeartbeatTime (read write)

Hier kann die *Heartbeat* - Zeit in Millisekunden eingestellt werden. Der Wert 0 bedeutet, dass diese Funktion nicht aktiv ist (siehe *Heartbeat*).

9.1.13 Index 1018 IdentityObject

Das Identity Objekt identifiziert den HDA 4000/7000 CANopen (PT). Die Identifikation besteht aus vier 32bit-Zahlen. Die Kombination dieser 4 Zahlen ergibt eine weltweit eindeutige Identifikation eines Gerätes.

Verwendete Subindizes:

0: LargestSubIndexSupported (read only)
1: VendorID (read only)
Eindeutiger Herstellercode (0xda für HYDAC ELECTRONIC GmbH)
2: ProductCode (read only)
Produktcode der Hydac Electronic
3: RevisionNumber (read only)
Revisionsnummer des Gerätes.
4: SerialNumber (read only)
Seriennummer des Gerätes

Product Codes:

0x001	HDA 7000
0x003	HDA 7000 P+T
0x004	HDA 4000
0x005	HDA 4000 P+T

9.1.14 Index 1800: TransmitPDOParameters

Diese Einträge legen die PDO-Übertragung fest. Im Einzelnen sind dies:

Parameter zur PDO-Übertragung

1. *COB-ID*
   Legt den Identifier für das PDO fest. Das höchstwertigste Bit (Bit31) des Eintrages gehört nicht mehr zur ID und hat die Bedeutung „*disable PDO*“. Ist dieses Bit gesetzt, so ist die Übertragung des PDO gesperrt.
2. *Transmission Type*
   Legt den Übertragungstyp fest.
   Werte zwischen 0 und 240 bedeuten eine synchrone Übertragung. Die Zahl steht für die Anzahl der SYNC Objekte, die empfangen werden müssen, bis das PDO gesendet wird.
   Der Wert 254 bedeutet eine herstellerspezifische Übertragung und der Wert 255 eine geräteprofilspezifische Übertragung. Bei 254 und 255 wird das PDO zyklisch gesendet, sofern eine Zeit (*Event Time*) ungleich 0 eingestellt ist. Bei 255 findet zusätzlich noch eine Übertragung statt, wenn der Messwert um mehr als ein eingestellter Betrag von dem Wert der letzten Übertragung abweicht, oder falls der Wert eingestellte Grenzen unter- bzw. überschreitet (siehe *Object Dictionary, Index* 7133, 7134 und 7135).
3. *Event Time*
   Legt die Zykluszeit für asynchrone Übertragungen bei Transmission Type 254 und 255 in Millisekunden fest. Der Wert 0 bedeutet keine zeitgesteuerte Übertragung.

Verwendete Subindizes:

0: LargestSubindexSupported (read only)
1: COBIDUsedByPDO (read write)
2: TransmissionType (read write)
5: EventTimer(read write)

9.1.15 Index 1A00: TransmitPDOMapping

Mit diesem Eintrag wird festgelegt, welche Daten mit dem PDO übertragen werden. Subindex 0 gibt die Anzahl der Daten im PDO an. Unter Subindex 1 ist der Index und Subindex sowie die Anzahl der Bits des ersten Datums hinterlegt, entsprechendes gilt für den Subindex 2.

Im Auslieferzustand hat der HDA 4000/7000 CANopen (PT) folgende Einträge:

Subindex	Inhalt	Bedeutung
0	2	Zwei Werte werden im *PDO* übertragen.
1	0x91300120	Der erste Wert im *PDO* ist der Wert von *Index 9130*, *Subindex* 01 mit einer Breite von 20h (=32 bit)
2	0x61500108	Der zweite Wert im PDO ist der Wert von *Index 6150*, *Subindex* 01 mit einer Breite von 8 bit

Verwendete Subindizes:

0: NumberOfMappedApplicationObjectsInPDO (read write)
Es sind die Werte 0, 1, 2, 3 und 4 zulässig. Entsprechend wird kein PDO übertragen, oder ein PDO mit ein oder zwei Werten.

1: 1stObjectToBeMapped (read write)
Beim HDA 4000/7000 CANopen (PT) sind folgende Werte zulässig:
0x71300110 = Aktueller Prozesswert1 mit 16bit-Breite, oder
0x91300120 = Aktueller Prozesswert1 mit 32bit-Breite, oder
0x61300120 = Aktueller Prozesswert1 in Float32-Darstellung, oder
0x61500108 = Status des 1. analogen Eingangskanals.
Falls ein zweiter Sensor vorhanden ist auch:
0x71300210 = Aktueller Prozesswert2 mit 16bit-Breite, oder
0x91300220 = Aktueller Prozesswert2 mit 32bit-Breite, oder
0x61300220 = Aktueller Prozesswert2 in Float32-Darstellung, oder
0x61500208 = Status des 2. analogen Eingangskanals.

2: 2ndObjectToBeMapped (read only)
wie unter 1:

Hinweis: Der HDA 4000/7000 CANopen (PT) führt nur eine 16bit-Auswertung durch. Die Darstellung des Wertes als 32bit-Wert erfolgt nur, um das *Default-PDO-Mapping* der DS404 zu erfüllen.

9.1.16 Index 1F80: NMT-Startup (read / write)

Wird Bit 2 gesetzt, so wird automatisch bei Erreichen des „Pre-Operational“ Status in den „Operational“ Status gewechselt. Erlaubte Werte sind: 0x8 und 0xC

9.2 Device Profile Specific Entries (DS404)

9.2.1 Index 7100: FieldValue

Dieser Eintrag enthält den unskalierten Wert, den die Signaleinheit liefert (siehe „Datenfluss im HDA 4000/7000 CANopen (PT)“, Seite 114).

12Bit-Wert der Messgröße 0=0% des Messbereichs; 4096 = 100% des Messbereichs

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: FieldValue1 (read only)
Falls ein zweiter Sensor vorhanden ist auch:
2: FieldValue2 (read only)

9.2.2 Index 6110: SensorType

Dieser Eintrag enthält den Sensortyp.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: SensorType1 (read only)
Falls ein zweiter Sensor vorhanden ist auch:
2: SensorType2 (read only)
Werte: 0x5A für Druckmessumformer
0x64 für Temperatur

9.2.3 Index 6111: AutoCalibration

Durch Einschreiben der Zeichenkette „cali“ wird die Skalierungseinheit auf 0 kalibriert (siehe „Datenfluss im HDA 4000/7000 CANopen (PT), Seite 14). Dazu ist es notwendig, dass am HDA 4000/7000 CANopen (PT) kein Druck bzw. Temperatur ansteht. Liegt der Wert außerhalb des zulässigen Toleranzbandes von +/- 3% FS (Full Scale = Bezogen auf den gesamten Messbereich), wird der Transfer mit dem Code 06090030h (Siehe [1], SDO Abort Codes) abgebrochen.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: AutoCalibration1 (write only)

9.2.4 Index 6112: OperatingMode

Mit diesem Eintrag kann der Eingangskanal in einen speziellen Betriebsmodus versetzt oder abgeschaltet werden. Beim HDA 4000/7000 CANopen (PT) sind die Kanäle nicht abschaltbar.
Deshalb ist der Eintrag nur lesbar und liefert immer den Wert 1=*Normal Operation*.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: OperatingMode1 (read only)
Falls ein zweiter Sensor vorhanden ist auch:
2: OperatingMode2 (read only)

9.2.5 Index 6114: ADCSampleRate

Mit diesem Eintrag wird die Samplerate, also die Abtastrate, festgelegt. Die Angabe erfolgt nach Norm als Vielfaches von Mikrosekunden, der
HDA 4000/7000 CANopen (PT) rundet dabei immer auf volle Millisekunden ab. Alle Angaben unter 1000 bewirken eine Samplerate von 1 Millisekunde.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: ADCSampleRate1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: ADCSampleRate2 (read write)

9.2.6 Index 7120: InputScaling1FV

Dieser Eintrag dient zur Skalierung des Prozesswertes und enthält den Wert von *FieldValue* bei der unteren Messbereichsgrenze. Normalerweise entspricht dies bei InputScaling1FV1 einem Druck von 0 bar (siehe „Datenfluss im
HDA 4000/7000 CANopen (PT)“, Seite 14 und [1], *Analogue Input Function Block*).

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: InputScaling1FV1 (read only)
Falls ein zweiter Sensor vorhanden ist auch:
2: InputScaling1FV2 (read only)

9.2.7 Index 6121/7121/9121: InputScaling1PV

Dieser Eintrag dient zur Skalierung des Prozesswertes und enthält den Wert von *ProcessValue* bei der unteren Messbereichsgrenze. Index 7121 hat eine Datenbreite von 16bit und Index 9121 von 32bit, Index 6121 enthält eine 32bit-breite Float-Zahl. Normalerweise entspricht dies bei InputScaling1PV1 einem Druck von 0 bar. Mit der Änderung dieses Eintrages können beliebige Messbereiche realisiert werden. Soll z.B. der Messbereich 50..700 kg betragen, so muss der Wert 50 eingeschrieben werden (siehe „Datenfluss im HDA 4000/7000 CANopen (PT)“, Seite 14 und [1], *Analogue Input Function Block*). Der Wert ist vorzeichenbehaftet!

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: InputScaling1PV1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: InputScaling1PV2 (read write)

9.2.8 Index 7122: InputScaling2FV

Dieser Eintrag dient zur Skalierung des Prozesswertes und enthält den Wert von *FieldValue* bei der oberen Messbereichsgrenze (siehe „Datenfluss im HDA 4000/7000 CANopen (PT)“, Seite 14 und [1], *Analogue Input Function Block*).

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: InputScaling2FV1 (read only)
Falls ein zweiter Sensor vorhanden ist auch:
2: InputScaling2FV2 (read only)

9.2.9 Index 6123/7123/9123: InputScaling2PV

Dieser Eintrag dient zur Skalierung des Prozesswertes und enthält den Wert von *ProcessValue* bei der oberen Messbereichsgrenze. Index 7123 hat eine Datenbreite von 16bit und Index 9123 von 32bit, Index 6123 enthält eine 32bit-breite Float-Zahl. Mit der Änderung dieses Eintrages können beliebige Messbereiche realisiert werden. Soll z.B. der Messbereich 50..700 kg betragen, so muss der Wert 700 eingeschrieben werden (siehe „Datenfluss im HDA 4000/7000 CANopen (PT)“, Seite 14 und [1], *Analogue Input Function Block*).

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: InputScaling2PV1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: InputScaling2PV2 (read write)

9.2.10 Index 6124/7124/9124: InputOffset

Dieser Eintrag definiert einen zusätzlichen Offset für den Prozesswert (siehe „Datenfluss im HDA 4000/7000 CANopen (PT)“, Seite 14 und [1], *Analogue Input Function Block*).
Index 7124 hat eine Datenbreite von 16bit und Index 9124 von 32bit, Index 6124 enthält eine 32bit-breite Float-Zahl.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: InputOffset1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: InputOffset2 (read write)

9.2.11 Index 6125/7125: InputAutoZero

Durch Einschreiben der Zeichenkette „zero“ wird der *InputOffset* so gesetzt, dass der aktuelle Prozesswert 0 liefert (siehe „Datenfluss im HDA 4000/7000 CANopen (PT)“, Seite 14 und [1], *Analogue Input Function Block*).
Index 7125 hat eine Datenbreite von 16bit und Index 6125 enthält eine 32bit-breite Float-Zahl.

Bei einem Lesezugriff auf Index 1 wird ein Fehler (abort = ReadError) ausgelöst.
Bei einem Schreibzugriff auf Index 1 wird überprüft, ob "zero" im Eingangspuffer steht. Falls ja wird der MeasuredOffset gesetzt, falls nein wird ein Fehler (abort=CanNotTransfer) zurückgegeben.

Verwendete Subindizes:
0: NumberOfEntries (read only)
1: InputAutoZero1 (write only)
Falls ein zweiter Sensor vorhanden ist auch:
2: InputAutoZero2 (write only)

9.2.12 Index 6130/7130/9130: ProcessValue

Diese Einträge enthalten den aktuellen Prozesswert. Index 7130 hat eine Datenbreite von 16bit und Index 9130 von 32bit, Index 6130 enthält eine 32bit-breite Float-Zahl. Da der HDA 4000/7000 CANopen (PT) grundsätzlich mit einer Datenbreite von 16bit rechnet, liefern alle Integer-Einträge den gleichen Wert. Diese wurden realisiert, um ein spezifikationsgerechtes *PDO-Mapping* zu ermöglichen. Dieses sieht nämlich als Voreinstellung den 32 bit-Wert vor. Zur Reduzierung des Datenverkehrs im Netz kann der 16bit-Wert jedoch *gemapped* werden (siehe „Datenfluss im HDA 4000/7000 CANopen (PT)“ auf Seite 14 und „TransmitPDOMapping“ auf Seite 21).

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: ProcessValue1 (read only)
Falls ein zweiter Sensor vorhanden ist auch:
2: ProcessValue2 (read only)

9.2.13 Index 6131: PhysicalUnitProcessValue

Dieser Eintrag enthält die Maßeinheit des Prozesswertes entsprechend der Empfehlung DR303.
Die eingestellte Maßeinheit hat keinerlei Einfluss auf die Signalverarbeitung. Zur Interpretation der Daten, siehe [3]

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: PhysicalUnitProcessValue1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: PhysicalUnitProcessValue2 (read write)
Werte: “°C” im Auslieferungszustand
“bar” im Auslieferungszustand

9.2.14 Index 6132: ProcessValueDecimalDigits

Dieser Eintrag enthält die Anzahl der Nachkommastellen für die Interpretation des Prozesswertes.

Beispiel:
Ist *ProcessValueDecimalDigits1* auf 2 eingestellt und *PhysicalUnitProcessValue1* auf „bar“, so entspricht ein *ProcessValue1* von 4711, dem Messwert 47,11 bar.
Wenn *ProcessValueDecimalDigits2* auf 1 eingestellt und *PhysicalUnitProcessValue2* auf „°C“, so entspricht ein *ProcessValue2* von 365, dem Messwert 36,5 °C.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: ProcessValueDecimalDigits1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: ProcessValueDecimalDigits2 (read write)

9.2.15 Index 6133/7133/9133: InterruptDeltaPV

Dieser Eintrag dient zur Steuerung der *PDO* Übertragung. Index 7133 hat eine Datenbreite von 16bit und Index 9133 von 32bit, Index 6133 enthält eine 32bit-breite Float-Zahl. Wenn der aktuelle *ProcessValue* um mehr als *InterruptDeltaPV* von dem zuletzt übertragenen abweicht, wird er übertragen. Dadurch werden ständige Übertragungen mit annähernd gleichem Inhalt vermieden. Damit dieser Mechanismus aktiviert ist, muss *TransmissionType* auf *ProfileSpecific* eingestellt sein (siehe *TransmissionType,* Seite 21).

Ist *InterruptDeltaPV* auf 0 gesetzt, so ist dieser Mechanismus ebenfalls deaktiviert.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: InterruptDeltaPV1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: InterruptDeltaPV2 (read write)

9.2.16 Index 6134/7134/9134: InterruptLowerLimit

Dieser Eintrag dient zur Steuerung der *PDO* Übertragung. Index 7134 hat eine Datenbreite von 16bit und Index 9134 von 32bit, Index 6134 enthält eine 32bit-breite Float-Zahl. Wenn der aktuelle *ProcessValue* den Grenzwert *InterruptLowerLimit* unterschreitet, wird er übertragen. Um Dauerübertragungen bei Messwertschwankungen um den Grenzwert herum zu vermeiden, muss der aktuelle Wert um mindestens 1% des eingestellten Messbereiches wieder über den Grenzwert steigen, damit bei einem erneuten Unterschreiten der Wert gesendet wird.

Damit dieser Mechanismus aktiviert ist, muss *TransmissionType* auf *ProfileSpecific* eingestellt sein (siehe *TransmissionType,* Seite 21).

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: InterruptLowerLimit1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: InterruptLowerLimit2 (read write)

9.2.17 Index 6135/7135/9135: InterruptUpperLimit

Dieser Eintrag dient zur Steuerung der *PDO* Übertragung. Index 7135 hat eine Datenbreite von 16bit und Index 9135 von 32bit, Index 6135 enthält eine 32bit-breite Float-Zahl. Wenn der aktuelle *ProcessValue* den Grenzwert *InterruptUpperLimit* überschreitet, wird er übertragen. Um Dauerübertragungen bei Messwertschwankungen um den Grenzwert herum zu vermeiden, muss der aktuelle Wert um mindestens 1% des eingestellten Messbereiches wieder unter den Grenzwert fallen, damit bei einem erneuten Überschreiten der Wert gesendet wird.

Damit dieser Mechanismus aktiviert ist, muss *TransmissionType* auf *ProfileSpecific* eingestellt sein (siehe *TransmissionType,* Seite 21).

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: InterruptUpperLimit1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: InterruptUpperLimit2 (read write)

9.2.18 Index 6136/7136/9136: TriggerHysteresis

Dieser Eintrag dient zur Festlegung der Trigger-Hysterese. Index 7136 hat eine Datenbreite von 16bit und Index 9136 von 32bit, Index 6136 enthält eine 32bit-breite Float-Zahl. Erst nach Über- oder Unterschreiten des in der Hysterese angegebenen Wertes (bei der unteren / oberen Triggergrenze) wird der Trigger ausgelöst. (So kann vermieden werden, dass der Trigger permanent auslöst, wenn die Werte um die eingestellte obere oder untere Triggergrenze schwanken.)

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: TriggerHysteresis1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: TriggerHysteresis2 (read write)

9.2.19 Index 6150: AnlogInputStatus

Dieser Eintrag zeigt den Status der Signaleinheit an. Das Status-Byte hat dabei folgende Bedeutung:

Bit 0: 1=Defekt am Signaleingang
Bit 1: 1=Positive Überlast ist vorhanden
Bit 2: 1=Negative Überlast ist vorhanden

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: AnlogInputStatus1 (read only)
Falls ein zweiter Sensor vorhanden ist auch:
2: AnlogInputStatus2 (read only)

9.2.20 Index 61A0: FilterType

Dieser Eintrag legt den Filtertyp fest. 1 bedeutet *Moving average*, 2 bedeutet *Repeating average*. Bei allen anderen Werten erfolgt keine Filterung. Das genaue Verfahren ist in [2] beschrieben.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: FilterType1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: FilterType2 (read write)

9.2.21 Index 61A1: FilterConstant

Dieser Eintrag legt die Filterkonstante fest. Eine Filterung findet nur statt, wenn die Konstante größer als 1 ist. Das genaue Verfahren ist in [2] beschrieben.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: FilterConstant1 (read write)
Falls ein zweiter Sensor vorhanden ist auch:
2: FilterConstant2 (read write)

9.3 Manufacturer Specific Entries

9.3.1 Index 2001: NodeID (read write)

Unter diesem Index ist die *Node-ID* hinterlegt. Zum Setzen der *NODE-ID* muss zur Sicherheit die Zeichenkette „set“ an die Adresse angehängt werden.

NodeID	´s´=0x73	´e´=0x65	´t´=0x74

Z.B. um die *Node-ID* auf 5 zu stellen, müssen folgende Datenbytes im SDO übertragen werden: 0x05, 0x73, 0x65, 0x74. Durch die Low-High-Darstellung bei CANopen muss deshalb folgender 32-bit-wert übertragen werden: 0x74657302.

Damit die neue *Node-ID* wirksam wird, muss erst der Befehl *StoreParameters* übertragen werden und danach der Knoten neu gestartet werden.

9.3.2 Index 2002: Baudrate (read write)

Unter diesem Index ist die Baudrate hinterlegt. Zum Setzen der Baudrate muss zur Sicherheit die Zeichenkette „set“ angehängt werden.

Die Zuordnung des Eintrages zu der Baudrate entspricht der DS305, *Layer Setting Services and Protocols*.

Eintrag	**0**	**1**	**2**	**3**	**4**	**6**	**7**	**8**	**9**
Baudrate	1 Mbit/s	800 kbit/s	500 kbit/s	250 kbit/s	125 kbit/s	50 kbit/s	20 kbit/s	10 kbit/s	auto

Baudrate	´s´=0x73	´e´=0x65	´t´=0x74

Z.B. um die Baudrate auf 125kbit/s zu stellen, müssen folgende Datenbytes im SDO übertragen werden: 0x04, 0x73, 0x65, 0x74. Durch die Low-High-Darstellung bei CANopen muss deshalb folgender 32-bit-wert übertragen werden: 0x74657304.

Damit die neue Baudrate wirksam wird, muss erst der Befehl *StoreParameters* übertragen werden und danach der Knoten neu gestartet werden.

Ist die Baudrate auf 9 gestellt, sucht sich der HDA 7000 CAN die aktuelle Baudrate anhand mitgehörter Bus-Frames. Nähere Informationen sind [5] zu entnehmen.

9.3.3 Index 2121: OriginalInputScaling1PV

Dieser Eintrag enthält die original eingestellte untere Messbereichsgrenze. Im Auslieferzustand ist der Wert mit Index 7121 *InputScaling1PV* identisch. *InputScaling1PV* kann aber im Betrieb geändert werden, um andere Prozessgrößen zu realisieren. Wenn der Befehl *RestoreDefaultParameters* empfangen wird, wird *OriginalInputScaling1PV* auf *InputScaling1PV* kopiert.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: OriginalInputScaling1PV1 (read only)
Falls ein zweiter Sensor vorhanden ist auch:
2: OriginalInputScaling1PV2 (read only)

9.3.4 Index 2123: OriginalInputScaling2FV

Dieser Eintrag enthält die original eingestellte obere *FieldValue*. Im Auslieferzustand ist der Wert mit Index 7122 *InputScaling2FV* identisch. *InputScaling2FV* kann aber im Betrieb geändert werden, um andere Prozessgrößen zu realisieren. Wenn der Befehl *RestoreDefaultParameters* empfangen wird, wird *OriginalInputScaling2FV* auf *InputScaling2FV* kopiert.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: OriginalInputScaling2FV1 (read only)
Falls ein zweiter Sensor vorhanden ist auch:
2: OriginalInputScaling2FV2 (read only)

9.3.5 Index 2131: OriginalPhysicalUnitProcessValue

Dieser Eintrag enthält die original eingestellte Einheit. Im Auslieferzustand ist der Wert mit Index 6131 *PhysicalUnitProcessValue* identisch. *PhysicalUnitProcessValue* kann aber im Betrieb geändert werden, um andere Prozessgrößen zu realisieren. Wenn der Befehl *RestoreDefaultParameters* empfangen wird, wird *OriginalPhysicalUnitProcessValue* auf *PhysicalUnitProcessValue* kopiert.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: OriginalPhysicalUnitProcessValue1 (read only)
Falls ein zweiter Sensor vorhanden ist auch:
2: OriginalPhysicalUnitProcessValue2 (read only)

9.3.6 Index 2132: OriginalDecimalDigitsProcessValue

Dieser Eintrag enthält die original eingestellte Anzahl an Nachkommastellen. Im Auslieferzustand ist der Wert mit Index 6131 *DecimalDigitsProcessValue* identisch. *DecimalDigitsProcessValue* kann aber im Betrieb geändert werden, um andere Prozessgrößen zu realisieren. Wenn der Befehl *RestoreDefaultParameters* empfangen wird, wird *OriginalDecimalDigitsProcessValue* auf *DecimalDigitsProcessValue* kopiert.

Verwendete Subindizes:

0: NumberOfEntries (read only)
1: OriginalDecimalDigitsProcessValue1 (read only)
Falls ein zweiter Sensor vorhanden ist auch:
2: OriginalDecimalDigitsProcessValue2 (read only)

10 Layer setting services (LSS) und Protokolle

Die LSS-Dienste und Protokolle, dokumentiert in CiA DS-305 V2.2, siehe [4], unterstützen das Abfragen und Konfigurieren verschiedener Parameter des Data Link Layers und des Application Layers eines LSS-Slaves durch ein LSS-Master über das CAN Netzwerk.

Unterstützt werden folgende Parameter:

- Node-ID
- Baudrate
- LSS-Adresse, gemäß dem Identity Objekt 1018h

Der Zugriff auf den LSS-Slave erfolgt dabei über seine LSS-Adresse, bestehend aus:

- Vendor-ID
- Produkt-Code
- Revisions-Nummer und
- Serien-Nummer

Das Mess-System unterstützt folgende Dienste:

Switch mode services

- Switch mode selective
  - einen bestimmten LSS-Slave ansprechen
- Switch mode global
  - alle LSS-Slaves ansprechen

Configuration services

- Configure Node-ID
  - Node-ID konfigurieren
- Configure bit timing parameters
  - Baudrate konfigurieren
- Activate bit timing parameters
  - Baudrate aktivieren
- Store configured parameters
  - konfigurierte Parameter speichern

Inquiry services

- Inquire LSS-address
  - LSS-Adresse anfragen
- Inquire Node-ID
  - Node-ID anfragen

Identification services

- LSS identify remote slave
  - Identifizierung von LSS-Slaves innerhalb eines bestimmten Bereichs
- LSS identify slave
  - Rückmeldung der LSS-Slaves auf das vorherige Kommando
- LSS identify non-configured remote slave
  - Identifizierung von nicht-konfigurierten LSS-Slaves, Node-ID = FFh
- LSS identify non-configured slave
  - Rückmeldung der LSS-Slaves auf das vorherige Kommando

10.1 Finite state automaton, FSA

Der FSA entspricht einer Zustandsmaschine und definiert das Verhalten eines LSS-Slaves. Gesteuert wird die Zustandsmaschine durch LSS COBs erzeugt durch einen LSS-Master, oder NMT COBs erzeugt durch einen NMT-Master, oder lokale NMT-Zustandsübergänge.

Die LSS-Modes unterstützen folgende Zustände:
(0) Initial: Pseudo-Zustand, zeigt die Aktivierung des FSAs an
(1) LSS waiting: Unterstützung aller Dienste wie unten angegeben
(2) LSS configuration: Unterstützung aller Dienste wie unten angegeben
(3) Final: Pseudo-Zustand, zeigt die Deaktivierung des FSAs an

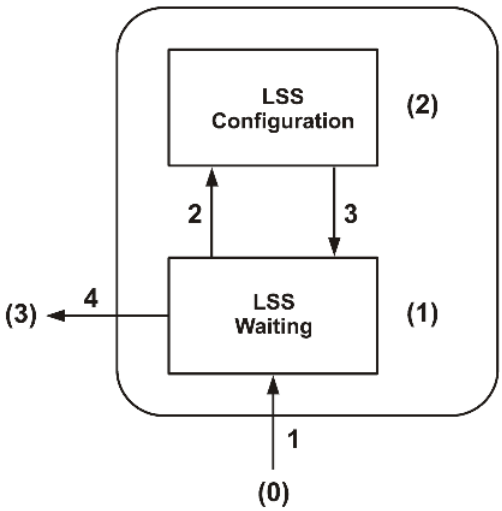


Abbildung 1: LSS-Modes

Zustandsverhalten der unterstützten Dienste

Dienste	Waiting	Configuration
Switch mode global	Ja	Ja
Switch mode selective	Ja	Nein
Activate bit timing parameters	Nein	Ja
Configure bit timing parameters	Nein	Ja
Configure Node-Id	Nein	Ja
Store configuration	Nein	Ja
Inquire LSS-address	Nein	Ja
LSS identify remote slave	Ja	Ja
LSS identify slave	Ja	Ja
LSS identify non-configured remote slave	Ja	Ja
LSS identify non-configured slave	Ja	Ja

LSS FSA Zustandsübergänge

Übergang	Ereignisse	Aktionen
1	Automatischer Übergang nach der Initialisierung beim Eintritt entweder in den NMT PRE OPERATIONAL Zustand oder NMT STOPPED Zustand, oder NMT RESET COMMUNICATION Zustand mit Node-ID = FFh.	keine
2	LSS 'switch state global' Kommando mit Parameter 'configuration_switch' oder 'switch state selective' Kommando	keine
3	LSS 'switch state global' Kommando mit Parameter 'waiting_switch'	keine
4	Automatischer Übergang, wenn eine ungültige Node-ID geändert wurde und die neue Node-ID erfolgreich im nichtflüchtigen Speicher abgelegt werden konnte UND der Zustand LSS waiting angefordert wurde.	keine

Sobald das LSS FSA weitere Zustandsübergänge im NMT FSA von NMT PRE OPERATIONAL auf NMT STOPPED und umgekehrt erfährt, führt dies nicht zum Wiedereintritt in den LSS FSA.

10.2 Übertragung von LSS-Diensten

Über die LSS-Dienste fordert der LSS-Master die einzelnen Dienste an, welche dann durch den LSS-Slave ausgeführt werden. Die Kommunikation zwischen LSS-Master und LSS-Slave wird über die implementierten LSS-Protokolle vorgenommen.
Ähnlich wie bei der SDO-Übertragung, werden auch hier zwei COB-IDs für das Senden und Empfangen benutzt:

COB-ID	Bedeutung
0x7E4	LSS-Slave → LSS-Master
0x7E5	LSS-Master → LSS-Slave

Tabelle 1: COB-IDs für LSS Services

10.2.1 LSS-Nachrichtenformat

Der maximal 8 Byte lange Datenbereich einer CAN-Nachricht wird von einem LSS-Dienst wie folgt belegt:

CS	Daten						
Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7

Tabelle 2: LSS-Nachricht

Byte 0 enthält die ***Command-Specifier*** (CS), danach folgen 7 Byte für die Daten.

10.3 Switch mode Protokolle

10.3.1 Switch mode global Protokoll

Das angegebene Protokoll hat den *Switch mode global service* implementiert und steuert das Zustandsverhalten des LSS-Slaves. Über den LSS-Master können alle LSS-Slaves im Netzwerk in den *Waiting Mode* oder *Configuration Mode* gebracht werden.

LSS-Master --> LSS-Slave

0	1	2	3	4	5	6	7
COB-ID	**CS**	**Mode**	Reserved by CiA				
0x7E5	04	0 = Waiting Mode 1 = Configuration Mode					

10.3.2 Switch mode selective Protokoll

Das angegebene Protokoll hat den *Switch mode selective service* implementiert und steuert das Zustandsverhalten des LSS-Slaves. Über den LSS-Master kann nur der LSS-Slave im Netzwerk in den *Configuration Mode* gebracht werden, dessen LSS- Adressattribute der LSS-Adresse entsprechen.

LSS-Master --> LSS-Slave

0	1	2	3	4	5	6	7
COB-ID	**CS**	**Vendor-Id**			Reserved by CiA		
0x7E5	64	LSB		MSB			

0	1	2	3	4	5	6	7
COB-ID	**CS**	**Product-Code**			Reserved by CiA		
0x7E5	65	LSB		MSB			

0	1	2	3	4	5	6	7
COB-ID	**CS**	**Revision-Number**			Reserved by CiA		
0x7E5	66	LSB		MSB			

0	1	2	3	4	5	6	7
COB-ID	**CS**	**Serial-Number**			Reserved by CiA		
0x7E5	67	LSB		MSB			

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB-ID	**CS**	Reserved by CiA					
0x7E4	68						

10.4 Configuration Protokolle

10.4.1 Configure Node-Id Protokoll

Das angegebene Protokoll hat den *Configure NMT-Address service* implementiert. Über den LSS-Master kann die Node-ID eines einzelnen LSS-Slaves im Netzwerk konfiguriert werden. Hierbei darf sich nur ein LSS-Slave im *Configuration Mode* befinden. Zur Speicherung der neuen Node-ID muss das *Store configuration protocol* an den LSS-Slave übertragen werden. Um die neue Node-ID zu aktivieren, muss der NMT-Dienst *Reset Communication* (0x82) aufgerufen werden.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Node-ID**	Reserved by CiA					
0x7E5	17	1…127 und 255						

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Error Code**	**Spec. Error**	Reserved by CiA				
0x7E4	17							

Error Code
0: Ausführung erfolgreich
1…254: Reserved
255: applikationsspezifischer Fehler aufgetreten

Specific Error
Wenn Error Code = 255 --> applikationsspezifischer Fehler aufgetreten,
sonst reserviert durch die CiA

10.4.2 Configure bit timing parameters Protokoll

Das angegebene Protokoll hat den *Configure bit timing parameters service* implementiert. Über den LSS-Master kann die Baudrate eines einzelnen LSS-Slaves im Netzwerk konfiguriert werden. Zur Speicherung der neuen Baudrate muss das *Store configuration protocol* an den LSS-Slave übertragen werden.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Table Selector**	**Table Index**	Reserved by CiA				
0x7E5	19	0	0…7					

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Error Code**	**Spec. Error**	Reserved by CiA				
0x7E4	19							

Table Selector
0: Standard CiA Baudraten-Tabelle

Table Index
0: 1 Mbit/s
1: 800 kbit/s
2: 500 kbit/s
3: 250 kbit/s
4: 125 kbit/s
5: 50 kbit/s
6: 20 kbit/s
7: 10 kbit/s

Error Code
0: Ausführung erfolgreich
1: selektierte Baudrate nicht unterstützt
2…254: Reserved
255: applikationsspezifischer Fehler aufgetreten

Specific Error
Wenn Error Code = 255 --> applikationsspezifischer Fehler aufgetreten,
sonst reserviert durch die CiA

10.4.3 Activate bit timing parameters Protokoll

Das angegebene Protokoll hat den *Activate bit timing parameters service* implementiert und aktiviert die über *Configure bit timing parameters protocol* festgelegte Baudrate bei allen LSS-Slaves im Netzwerk, die sich im *Configuration Mode* befinden.

LSS-Master --> LSS-Slave

0	1	2	3	4	5	6	7
COB-ID	**CS**	**Switch Delay [ms]**		Reserved by CiA			
0x7E5	21	LSB	MSB				

Switch Delay
Der Parameter *Switch Delay* definiert die Länge zweier Verzögerungsperioden (D1, D2) mit gleicher Länge. Damit wird das Betreiben des Busses mit unterschiedlichen Baudratenparametern verhindert. Nach Ablauf der Zeit D1 und einer individuellen Verarbeitungsdauer wird die Umschaltung intern im LSS-Slave vorgenommen. Nach Ablauf der Zeit D2 meldet sich der LSS-Slave wieder mit CAN-Nachrichten und der neu eingestellten Baudrate.

Es gilt:
Switch Delay > längste vorkommende Verarbeitungsdauer eines LSS-Slaves

10.4.4 Store configuration Protokoll

Das angegebene Protokoll hat den *Store configured parameters service* implementiert. Über den LSS-Master können die konfigurierten Parameter eines einzelnen LSS-Slaves im Netzwerk in den nichtflüchtigen Speicher abgelegt werden. Hierbei darf sich nur ein LSS-Slave im *Configuration Mode* befinden.

LSS-Master --> LSS-Slave

0	1	2	3	4	5	6	7
COB-ID	**CS**	Reserved by CiA					
0x7E5	23						

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB-ID	**CS**	**Error Code**	**Spec. Error**	Reserved by CiA			
0x7E4	23						

Error Code
0: Ausführung erfolgreich
1: *Store configuration* nicht unterstützt
2…254: Reserved
255: applikationsspezifischer Fehler aufgetreten

Specific Error
Wenn Error Code = 255 --> applikationsspezifischer Fehler aufgetreten,
sonst reserviert durch die CiA

10.5 Inquire LSS-Address Protokolle

10.5.1 Inquire Identity Vendor-ID Protokoll

Das angegebene Protokoll hat den *Inquire LSS-Address service* implementiert. Über den LSS-Master kann die Vendor-ID eines einzelnen LSS-Slaves im Netzwerk ausgelesen werden. Hierbei darf sich nur ein LSS-Slave im *Configuration Mode* befinden.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB-ID	**CS**	Reserved by CiA						
0x7E5	90							

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Vendor-Id** (= Index 1018h:01)				Reserved by CiA		
0x7E4	90	LSB			MSB			

10.5.2 Inquire Identity Product-Code Protokoll

Das angegebene Protokoll hat den *Inquire LSS-Address service* implementiert. Über den LSS-Master kann der Hersteller-Gerätename eines einzelnen LSS-Slaves im Netzwerk ausgelesen werden. Hierbei darf sich nur ein LSS-Slave im *Configuration Mode* befinden.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB-ID	**CS**	Reserved by CiA						
0x7E5	91							

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Product-Code** (= Index 1018h:02)				Reserved by CiA		
0x7E4	91	LSB			MSB			

10.5.3 Inquire Identity Revision-Number Protokoll

Das angegebene Protokoll hat den *Inquire LSS-Address service* implementiert. Über den LSS-Master kann die Revisionsnummer eines einzelnen LSS-Slaves im Netzwerk ausgelesen werden. Hierbei darf sich nur ein LSS-Slave im *Configuration Mode* befinden.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB-ID	**CS**	Reserved by CiA						
0x7E5	92							

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Revision-Number** (= Index 1018h:03)				Reserved by CiA		
0x7E4	92	LSB			MSB			

10.5.4 Inquire Identity Serial-Number Protokoll

Das angegebene Protokoll hat den *Inquire LSS-Address service* implementiert. Über den LSS-Master kann die Seriennummer eines einzelnen LSS-Slaves im Netzwerk ausgelesen werden. Hierbei darf sich nur ein LSS-Slave im *Configuration Mode* befinden.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB-ID	**CS**	Reserved by CiA						
0x7E5	93							

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Serial-Number** (= Index 1018h:04)				Reserved by CiA		
0x7E5	93	LSB			MSB			

10.6 Inquire Node-ID Protokoll

Das angegebene Protokoll hat den *Inquire Node-ID service* implementiert. Über den LSS-Master kann die Node-ID eines einzelnen LSS-Slaves im Netzwerk ausgelesen werden. Hierbei darf sich nur ein LSS-Slave im *Configuration Mode* befinden.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB-ID	**CS**	Reserved by CiA						
0x7E5	94							

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Node-ID**	Reserved by CiA					
0x7E4	94	1…127 und 255						

Node-ID
Entspricht der Node-ID des selektierten Gerätes. Wenn die Node-ID eben gerade erst über den *Configure Node-ID service* geändert wurde, wird die ursprüngliche Node-ID zurückgemeldet. Erst nach Ausführung des NMT-Dienstes *Reset Communication* (0x82) wird die aktuelle Node-ID zurückgemeldet.

10.7 Identification Protokolle

10.7.1 LSS identify remote slave Protokoll

Das angegebene Protokoll hat den *LSS identify remote slaves service* implementiert. Über den LSS-Master können LSS-Slaves im Netzwerk in einem bestimmten Bereich identifiziert werden. Alle LSS-Slaves, die dem angegebenen Vendor-ID, Product-Code, Revision-No. – Bereich und Serial-No. – Bereich entsprechen, antworten mit dem *LSS identify slave protocol*.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Vendor-Id**				Reserved by CiA		
0x7E5	70	LSB			MSB			

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Product-Code**				Reserved by CiA		
0x7E5	71	LSB			MSB			

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Revision-Number-Low**				Reserved by CiA		
0x7E5	72	LSB			MSB			

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Revision-Number-High**				Reserved by CiA		
0x7E5	73	LSB			MSB			

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Serial-Number-Low**				Reserved by CiA		
0x7E5	74	LSB			MSB			

	0	1	2	3	4	5	6	7
COB-ID	**CS**	**Serial-Number-High**				Reserved by CiA		
0x7E5	75	LSB			MSB			

10.7.2 LSS identify slave Protokoll

Das angegebene Protokoll hat den *LSS identify slave service* implementiert. Alle LSS-Slaves, die den im *LSS identify remote slaves protocol* angegebenen LSS-Adress-Attributen entsprechen, antworten mit diesem Protokoll.

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB-ID	**CS**	Reserved by CiA						
0x7E4	79							

10.7.3 LSS identify non-configured remote slave Protokoll

Das angegebene Protokoll hat den *LSS identify non-configured remote slave service* implementiert. Über den LSS-Master werden alle nichtkonfigurierten LSS-Slaves (Node-ID = FFh) im Netzwerk identifiziert. Die betreffenden LSS-Slaves antworten mit *dem LSS identify non-configured remote slave protocol*.

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB-ID	**CS**	Reserved by CiA					
0x7E4	76						

10.7.4 LSS identify non-configured slave Protokoll

Das angegebene Protokoll hat den LSS *identify non-configured slave service* implementiert. Alle LSS-Slaves, die eine ungültige Node-ID (FFh) besitzen, antworten nach Ausführung *des LSS identify non-configured slave protocol* mit diesem Protokoll.

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB-ID	**CS**	Reserved by CiA					
0x7E4	80						

10.7.5 Fastscan Protokoll

Das angegebene Protokoll hat den LSS *fastscan service* implementiert. Der LSS-Slaves, die über alle 4 LSS Sub-Einträge identifiziert wurden wechselt danach in den *Configuration Mode.*

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB-ID	**CS**	**ID Nummer**			**Bit Check**	**LSS Sub**	**LSS Next**
0x7E4	81	LSB		MSB			

11 Anschluss

Der Anschluss kann mit Hilfe der beigelegten gerätespezifischen Steckerbelegung durchgeführt werden, siehe Bedienanleitung HDA 7000 CANopen (Mat.nr. 669827, die Bedienanleitung ist Teil des Lieferumfangs HDA 7000 CANopen).

11.1 Einschalten der Versorgungsspannung

Nachdem der Anschluss und alle Einstellungen vorgenommen worden sind, kann die Versorgungsspannung eingeschaltet werden.

Nach dem Einschalten der Versorgungsspannung und Beendigung der Initialisierung geht das Mess-System in den Vor-Betriebszustand (PRE-OPERATIONAL). Dieser Zustand wird durch die Boot-Up-Meldung „**COB-ID 0x700+Node ID**“ bestätigt. Falls das Mess-System einen internen Fehler erkennt, wird eine Emergency-Meldung mit dem Fehlercode übertragen (siehe Emergency Object (EMCY) Seite11).

Im PRE-OPERATIONAL-Zustand ist zunächst nur eine Parametrierung über Service-Daten-Objekte möglich. Es ist aber möglich, PDOs unter Nutzung von SDOs zu konfigurieren. Ist das Mess-System in den Zustand OPERATIONAL überführt worden, ist auch eine Übertragung von PDOs möglich (siehe Network Management Services (NMT), Seite13).

11.2 Einstellen der Node-ID und Baudrate mittels LSS-Dienste

11.2.1 Konfiguration der Node-ID, Ablauf

Annahme:

- LSS-Adresse unbekannt
- der LSS-Slave ist der einzige Teilnehmer in Netzwerk
- es soll die Node-ID 12 dez. eingestellt werden

Vorgehensweise:

- LSS-Slave mit dem Commannd Specifier 04 *Switch mode global protocol*, Mode = 1 in den *Configuration Mode* bringen. (siehe Switch mode global Protokoll, Seite 35)
- Commannd Specifier 17 *Configure NMT-Address protocol*, Node-ID = 12 ausführen. (siehe Configure Node-Id Protokoll, Seite 36) --> Rückmeldung abwarten und erfolgreiche Ausführung überprüfen, --> Error Code = 0.

- Commannd Specifier 23 *Store configuration protocol* ausführen.
  --> Rückmeldung abwarten und erfolgreiche Ausführung überprüfen,
  --> Error Code = 0.
  (siehe Store configuration Protokoll, Seite 38)
- Versorgungsspannung des LSS-Slaves aus-, danach wieder einschalten. Die neue Konfiguration ist jetzt aktiv.

11.2.2 Konfiguration der Baudrate, Ablauf

Annahme:

- LSS-Adresse unbekannt
- der LSS-Slave ist der einzige Teilnehmer in Netzwerk
- es soll die Baudrate 125 kbit/s eingestellt werden

Vorgehensweise:

- NMT- Commannd Specifier *Stop Remote Node* (0x02) aufrufen, um den LSS-Slave in den *Stopped state* zu bringen. Der LSS-Slave sollte keine CAN-Nachrichten mehr senden --> Heartbeat abgeschaltet
  (siehe Network Management Services (NMT), Seite 13)
- LSS-Slave mit dem Commannd Specifier 04 *Switch mode global protocol*, Mode = 1 in den *Configuration Mode* bringen.
  (siehe Switch mode global Protokoll, Seite 35)
- Commannd Specifier 19 *Configure bit timing parameters protocol* ausführen, Table Selector = 0, Table Index = 4
  --> Rückmeldung abwarten und erfolgreiche Ausführung überprüfen,
  --> Error Code = 0.
  (siehe Configure bit timing parameters Protokoll, Seite 37)
- Commannd Specifier 21 *Activate bit timing parameters protocol* aufrufen, damit die neue Baudrate aktiv wird.
  (siehe Activate bit timing parameters Protokoll, Seite 38)
- Commannd Specifier 23 *Store configuration protocol* ausführen.
  --> Rückmeldung abwarten und erfolgreiche Ausführung überprüfen,
  --> Error Code = 0.
  (siehe Store configuration Protokoll, Seite 38)
- Versorgungsspannung des LSS-Slaves aus-, danach wieder einschalten. Die neue Konfiguration ist jetzt aktiv.

12 Inbetriebnahme

12.1 CAN – Schnittstelle

Die CAN-Bus-Schnittstelle ist durch die internationale Norm ISO/DIS 11898 definiert und spezifiziert die zwei untersten Schichten des CAN Referenz-Models.

Die Konvertierung der Mess-System-Information in das CAN-Protokoll (CAN 2.0A) geschieht über einen CAN-Kontroller. Die Funktion des CAN-Kontrollers wird durch einen Watchdog überwacht.

Das CANopen Kommunikationsprofil (CiA Standard DS 301, siehe [1]) basiert auf dem CAN Application Layer (CAL) und beschreibt, wie die Dienste von Geräten benutzt werden. Das CANopen Profil erlaubt die Definition von Geräteprofilen für eine dezentralisierte E/A.

Das Mess-System mit CANopen Protokoll unterstützt das Geräteprofil für Messgeräte (CiA Draft Standard 404, Version 1.2, siehe [2]).

Die Kommunikations-Funktionalität und Objekte, welche im Messgeräteprofil benutzt werden, werden in einer EDS-Datei (Electronic Data Sheet) beschrieben. Wird ein CANopen Konfigurations-Hilfsprogramm benutzt (z.B. CANSETTER), kann der Benutzer die Objekte (SDO´s) des Mess-Systems auslesen und die Funktionalität programmieren.

12.2 EDS-Datei

Die EDS-Datei (elektronisches Datenblatt) enthält alle Informationen über die Mess-System-spezifischen Parameter sowie Betriebsarten des Mess-Systems. Die EDS-Datei wird durch das CANopen-Netzwerkkonfigurationswerkzeug eingebunden, um das Mess-System ordnungsgemäß zu konfigurieren bzw. in Betrieb nehmen zu können.

Die EDS-Datei die zum Gerät passt, kann am Gerätenamen und der dem Gerät entsprechenden Major-Revisionsnummer des Gerätes im Dateinamen erkannt werden

Die Datei inklusive der vorliegenden CANopen-Schnittstellenbeschreibung befindet sich zum Download auf unserer Homepage unter http://www.hydac.de/de-de/service/download-software-auf-anfrage/software/software-download/electronic.html.

13 Weiterführende Literatur:

[1] DS301, Version 4.02
Application Layer and Communication Profile
[2] DS404, Version 1.2
Device Profile Measuring Devices and Closed-Loop Controllers
[3] DR303-2, Version 1.0
Representation of SI Units and Prefixes
[4] CiA Draft Standard Proposal 305
Layer setting services (LSS) and protocols
Version: 2.2 , 26 August 2008
[5] CiA Application Note 801
CANopen Automatic bit-rate detection -Recommended practice and application hints
Version 1.0 01 January 2005

HYDAC ELECTRONIC GMBH
Hauptstr. 27
D-66128 Saarbrücken
Germany

Web: www.hydac.com
E-Mail: electronic@hydac.com
Tel.: +49 (0)6897 509-01
Fax.: +49 (0)6897 509-1726

HYDAC Service
Für Fragen zu Reparaturen steht Ihnen der HYDAC Service zur Verfügung.

HYDAC SERVICE GMBH
Hauptstr. 27
D-66128 Saarbrücken
Germany

Tel.: +49 (0)6897 509-1936
Fax.: +49 (0)6897 509-1933

Anmerkung

Die Angaben in dieser Bedienungsanleitung beziehen sich auf die beschriebenen Betriebsbedingungen und Einsatzfälle. Bei abweichenden Einsatzfällen und/oder Betriebsbedingungen wenden Sie sich bitte an die entsprechende Fachabteilung.

Bei technischen Fragen, Hinweisen oder Störungen nehmen Sie bitte Kontakt mit Ihrer HYDAC-Vertretung auf.

Technische Änderungen sind vorbehalten.

HYDAC ELECTRONIC

Protocol Description CANopen

HDA 4000/7000

(Translation of the original instruction)

Part no : 669821 / Status: 2014/01/30

Table of Contents

Preface

This manual provides you, as user of our product, with key information on the operation and maintenance of the equipment.

It will acquaint you with the product and assist you in obtaining maximum benefit in the applications for which it is designed.

Always keep the manual with the device for immediate reference.
Note that the data on the software technology provided in this manual refers to that available at the time of publication.

If you discover errors while reading the documentation or have additional suggestions or notes, contact us at:

HYDAC ELECTRONIC GMBH
Technical Documentation
Hauptstraße 27
66128 Saarbrücken
-Germany-
Phone:+49(0)6897 / 509-01
Fax.: +49 (0)6897 509-1726
Email: electronic@hydac.com

The editorial team looks forward to hearing from you.

„Putting experience into practice“

1 Introduction

The pressure transducers HDA 4000/7000 CANopen (PT) meet the CANopen standards according to the following profiles and standards:

- CiA Draft Standard 301
  Application Layer and Communication Profile
  Version 4.02, Date: 13 February 2002
- CiA Draft Standard 404
  Device Profile Measuring Devices and Closed-Loop Controllers
  Version 1.2, Date: 15. May 2002
- CiA Draft Standard Proposal 305
  Layer setting services (LSS) and protocols
  Version: 2.2 , 26 August 2008
- CiA Application Note 801
  CANopen Automatic bit-rate detection -Recommended practice and application hints
  Version 1.0 01 January 2005

This manual describes the functions supported by the HDA 4000/7000 CANopen (PT). A basic knowledge of CAN and CANopen is assumed. The exact function is described in the a.m. *Draft Standards*. Since both specifications are issued in English, the features described in this manual are identified using the English description from the specification and are shown in *italics*, for clarity.

2 Functions of HDA 4000/7000 CANopen (PT)

- Measuring the actual pressure value and/or temperature value using:
  Pressure:
  - 1kHz sample rate
  - 0.2% accuracy
  - Resolution 12 bit

  Temperature:
  - 1kHz sample rate
  - Accuracy 1 °C
- Conversion of the pressure/ temperature value into a user-scaleable linear process value.
- Sending the actual process value as a PDO for the following events:
  - Synchronously in relation to received SYNC objects
  - Asynchronously, cyclically, in the range 1 millisecond to 1 minute
  - When the measured value changes in relation to an adjustable differential
  - When an adjustable limit is exceeded
  - When the value falls below an adjustable limit

3 Transmission rates

The HDA 4000/7000 CANopen (PT) supports the following transmission rates (Baud rates):

- 1 Mbit/s
- 800 kbit/s
- 500 kbit/s
- 250 kbit/s
- 125 kbit/s
- 50 kbit/s
- 20 kbit/s
- 10 kbit/s

The timing complies with DS301, *Bit rates and timing*.

The transmission rate used is stored in a non-volatile memory. When supplied, it is set to 250 kbit/s and can be changed either via the CAN-Bus (see Object Dictionary Index 2002).
An additional automatic bitrate search function is possible.

4 CAN Frames

The HDA 4000/7000 CANopen (PT) supports the 11-bit base frames with 11-bit identifier required in the specification. Extended frames with 29-bit identifier are not supported but are tolerated. This means that extended frames are not recognised but neither do they cause errors.

5 Node ID

To operate the HDA 4000/7000 CANopen (PT) in a CANopen network a unique *Node ID* must be set within the network.

The set *Node ID* is stored in a non-volatile memory, like the transmission rate, and can also be adjusted via the CAN bus (see *Object Dictionary Index 2001*).When supplied the address 1 is set.

6 Transmission Services

6.1 Service Data Object (SDO)

With CANopen, all the device's data (setting parameters and measured data) is filed in an *Object Dictionary* under a specified *Index*. Some entries of the *Object Dictionary* are sub-divided still further using a *Subindex*. Using the *SDOs,* other network nodes can read from or write to the *Object Dictionary* of the HDA 4000/7000 CANopen (PT).
The HDA 4000/7000 CANopen (PT) takes on the role of a *Server*, and the device which wants to read or write the data, takes on the role of a *Client*.

To transmit data the HDA 4000/7000 CANopen (PT) must have a *Receive-SDO* with which it receives data and a *Transmit-SDO* with which it sends data. Sequence of the transmission of data:

Reading from Object Dictionary:

1. A device (Client) sends the Receive-SDO of the HDA 4000/7000 CANopen (PT) (Server).
   In this *SDO* there is an identification to say that the *Object Dictionary* is to be read, as well as the *Index* and *Subindex*.
2. The HDA 4000/7000 CANopen (PT) (*Server*) sends its *Transmit-SDO*. Here also are the *Index* and the *Subindex*, and the read data.

Writing to the Object Dictionary:

1. A device (Client) sends the Receive-SDO of the HDA 4000/7000 CANopen (PT) (Server).
   In this SDO there is an identification, to say that the Object Dictionary is to be written to, as well as the required index, subindex and the data to be entered.
2. The HDA 4000/7000 CANopen (PT) (Server) sends its Transmit-SDO. In this there are also the index and the subindex, as well as an identification to say the Object Dictionary has been written to.

If an error should occur, e.g. the specified *Index* does not exist, or an attempt is made to write to a *read only* entry, or the data is not within the valid range, then the Transmit-SDO receives an appropriate *Abort SDO Transfe*r identification and a corresponding *Abort Code* (see [1])

The Object Directory of the HDA 4000/7000 CANopen (PT) is configured such that each entry has a maximum of 4 bytes of data, so that all data can be transmitted as *expedited*. In other words, all data can be packed in a single *SDO*, resulting in particularly effective transmissions.

The particular *COB ID* of the *SDO* corresponds to the *Pre defined Connection Set* defined in the DS301 *a*nd cannot be altered.

COB IDs for Service Data Objects

SDO	*COB ID*
Receive – SDO	600h+Node ID
Transmit – SDO	580h+Node ID

6.2 Process Data Object (PDO)

Data transmission using *SDOs* is indeed very flexible, but has some disadvantages when transmitting measured values or actuating variables: only one piece of data can be read, the data must first be requested with an *SDO* and because the relevant *Index* and *Subindex* is also transmitted, what is known as the overhead increases further.

For this reason CANopen defines what are known as *Process Data Objects*. These contain only the necessary useful data. There are two types of *PDO*:

1. *Transmit PDOs*
   With these, the measuring instrument can send its measured values.
2. *Receive PDOs*
   With these, the actuating variables can be transmitted to an actuator or a controller.

What is known as the *PDO Mapping* stipulates which data is now in a *PDO*. This *PDO Mapping* is stored in the *Object Dictionary* (see *Object Dictionary, Index 1A00*).

The *PDO Transmission Type* stipulates with which ID and for which event a *PDO* is transmitted. These settings are also stored in the *Object Dictionary* (see *Object Dictionary, Index 1800*).

Events which result in a PDO being sent:

1. Receipt of a *SYNC* object (synchronous transmission).
2. Expiry of an adjustable cycle time in the range 1 milliseconds to 1 minute (cyclical transmission).
3. The measured value has changed by an adjustable amount compared to the last PDO.
4. The measured value has exceeded an adjustable limit.
5. The measured value has fallen below an adjustable limit.

The HDA 4000/7000 CANopen (PT) implements a *Transmit PDO*, which transmits the actual pressure value and the status of the signal input.

The DS404 stipulates the transmission of the actual measured value as a 32-bit value, and the status as an 8-bit value, as the default setting. However, it is possible to change this setting via *PDO Mapping* so that the measured value is transmitted as a 16-bit value. The transmission of the status can also be disabled by changing the *PDO Mapping*.

6.3 Synchronisation Object (SYNC)

SYNC objects are used to implement a synchronous data transmission. A *SYNC* object is in principle a CAN message with a defined identifier, without data. CANopen differentiates between *SYNC Producers* and *SYNC Consumers. SYNC Producers* are devices on the bus which send a *SYNC* at adjustable time intervals. *SYNC Consumers* are devices which react to receiving a *SYNC*. In a CANopen network several *SYNC* objects can exist. The individual *SYNC* objects are differentiated by means of the *SYNC ID* which corresponds to the CAN identifier used. The *SYNC ID* used is stored in the *Object Dictionary*.

The HDA 4000/7000 CANopen (PT) provides the functionality of a *SYNC Consumer*. When the *PDO Transmission Type* is set appropriately, a *PDO* is sent on receipt of a *SYNC*. The *SYNC ID* is pre-set to 80h and can be changed in the *Object Dictionary* (see *Object Dictionary, Index 1005*). In PDO Transmission Type the number of received SYNC objects which result in a PDO being sent, can be set.

6.4 Emergency Object (EMCY)

EMCY objects are sent when an error occurs. *EMCY* objects contain an *Emergency Error Code,* the contents of an *Error register* as well as a *Manufacturer specific Error Field*. If a notified error is eliminated or disappears, this is also notified by a special *EMCY* object.

An emergency message will be sent, if an error occurs or if this error disappears The message is structured as follows:

EmergencyErrorCode (hex)	**ManufacturerSpecificErrorField (hex)**	**Description**
FF00	Bit mask	device-specific error
8100	Error code	Communication error
0000	0	There are no errors

Device-specific error

Bit-Number	**Description**
0	Error while loading the user settings
1	Error while saving the user settings
2	Error while loading the factory setting
3	Error while reading the pressure value
4	Error while reading the temperature value
5	Error while loading the calibration data

Communication error

Error code (hex)	Description
0x01	Stuff Error
0x02	Form Error
0x03	Ack Error
0x04	Bit1 Error
0x05	Bit0 Error
0x10	Bus Off
0x11	Crc Error

Example

Error while reading the pressure value

Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
FF	**00**	**00**	**00**	**00**	**00**	**08**	**00**

The *EMCY* object has the pre-set ID 80h+*Node ID* and can be changed in the *Object Dictionary* (see *Object Dictionary, Index 1014*).

6.5 Heartbeat

Using the *Heartbeat Protocol* the individual nodes can be monitored. CANopen differentiates between the following functions:

1. *Heartbeat Producer*
   sends a *Heartbeat o*bject in cyclical intervals.
2. *Heartbeat Consumer*
   monitors the sending of certain *Heartbeat* objects.

The cycle time can be adjusted in the *Object Dictionary* in milliseconds. If a time interval of 0 is specified, this indicates "*Heartbeat* not active".

With the *Heartbeat* object the status of the *Heartbeat Producers* is also transmitted as bytes.

Meaning of the Heartbeat object contents

Value	Status	NOTE
0	*BOOTUP*	The device has booted up.
4	*STOPPED*	The device has stopped.
5	*OPERATIONAL*	The device is working normally.
127	*PRE-OPERATIONAL*	The device is not sending any *PDOs*, but can modify *SDOs*.

The HDA 4000/7000 CANopen (PT) can operate as a *Heartbeat Producer*. The ID of the Heartbeat is 700h + Node ID. The time has been pre-set to 0 (not active) and can be changed (see Object Dictionary, Index 1017).

6.6 Network Management Services (NMT)

NMT objects are used to start, stop or reset devices. CANopen differentiates between the following functionalities:

1. *NMT Master*
   controls other nodes.
2. *NMT Slave*
   is controlled by a *Master*.

In a CANopen network only one *NMT* object exists with the Identifier 0. Two bytes are always transmitted. The first byte contains the *Command Specifier*, which represents the command, the second byte contains the *Node ID* of the node which carries out this command. A value of 0 indicates that this command is valid for all nodes. The following commands are possible:

NMT commands

1. *Start Remote Node*
   The node changes to the *Operational* condition.
2. *Stop Remote Node*
   The node changes to the *Stopped* condition.
3. *Enter Pre-Operational*
   The node changes to the *Pre-Operational* condition.
4. *Reset Node*
   The "Device Profile Specific" OD range is reset, the baud rate is, if necessary, re-initialised and then changes to the Reset Communication condition.
5. *Reset Communicatiuon*
   The communication unit of the node is reset and then the node changes to the *Pre-Operational* condition.

The HDA 4000/7000 CANopen (PT) operates as a *NMT Slave* and supports all the *NMT* services.

6.7 Boot Up Protocol

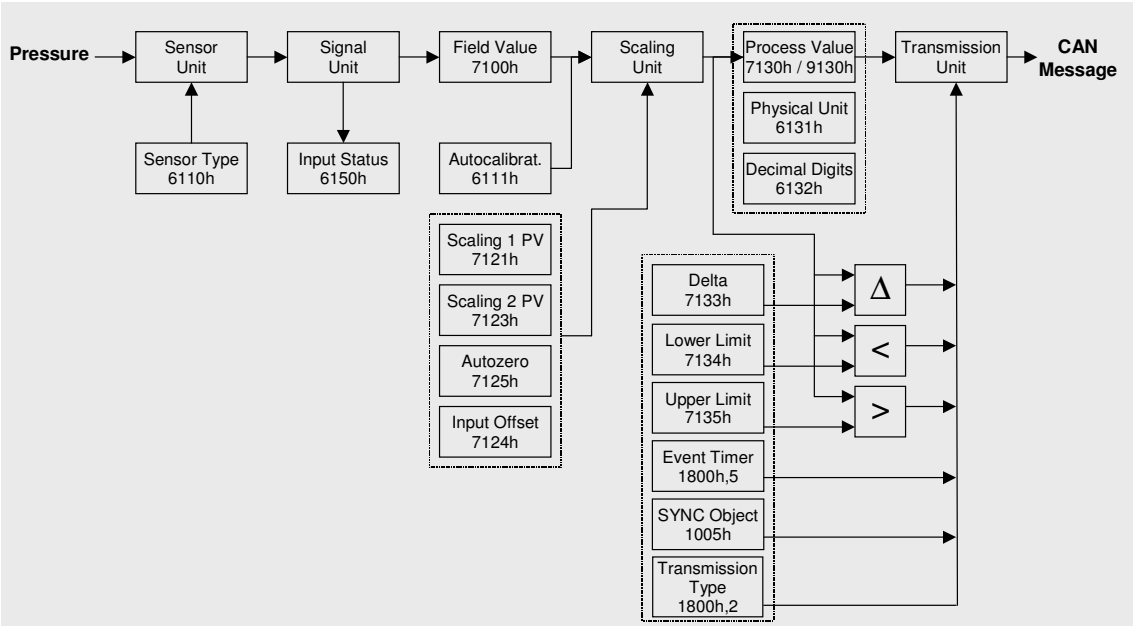
If a *NMT Slave* changes to the *Pre-operational* condition after initialisation, it sends a *Boot Up o*bject in each case. In principle this is no different to a *Heartbeat* object with the status 0.

On the HDA 4000/7000 CANopen (PT) this function is implemented.

7 Data flow in the HDA 4000/7000 CANopen (PT)

The following diagram shows the flow of data within the HDA 4000/7000 CANopen (PT), and the relevant indices of the *Object Dictionary*.

Data flow in the HDA 4000/7000 CANopen (PT)



7.1 Sensor Unit

The sensor unit measures the pressure and/or temperature and converts it into an electrical signal. The type of sensor (*Pressure Transducer)* is stored in *Sensor Type*.

7.2 Signal Unit

The electrical signal is processed in the signal unit, converted to a digital signal and is available as a *FieldValue*. The status of the signal process is stored in *InputStatus*.

7.3 Scaling Unit

The *FieldValue* is then converted in the *Scaling Unit* into a *ProcessValue*.

The indications *PhysicalUnit* and *DecimalDigits* describe how the *ProcessValue* is to be interpreted.If in *PhysicalUnit* for example, the unit is bar and in *DecimalDigits* the value is 1, then a *ProcessValue* of 127 corresponds to the measured value of 12.7 bar.

The conversion of the *FieldValue* into the *ProcessValue* is carried out using the two scaling values *Scaling1PV a*nd *Scaling2PV*. The first value indicates which process value corresponds to the lower range limit (e.g. 0 bar). The second value indicates which process value corresponds to the upper range limit (e.g. 400 bar).

With *InputOffset* an additional offset can be calculated. With *AutoZero, InputOffset* is set so that the actual process value has the value 0 (or the minimum process value).

With *Autocalibration* an offset error (pressure only!) in the signal conditioning can be cleared. The HDA 4000/7000 CANopen (PT) must be depressurized for this purpose and the offset must not be more than ±3%.

7.4 Transmission Unit

If one of the following events occurs, then depending on the preset *TransmissionType,* the value of the PDO is sent.

1. The actual *ProcessValue* has changed by more than *Delta* compared to the last transmitted *Process Value*.
2. The value has fallen below the *LowerLimit*.
3. The Upper*Limit* has been exceeded.
4. The *EventTimer* has expired (cyclical transmission).
5. One or more *Sync* objects have been received (synchronous transmission).

8 The Object Dictionary

8.1 Set-up of the Object Dictionary

As already mentioned, all the data is stored in the *Object Dictionary*. The entries supported by the HDA 4000/7000 CANopen (PT) are listed in the following chapters. The index is always shown in hexadecimal notation, according to the specification, without the hexadecimal representation being shown specially. For each entry, the corresponding type of access is shown. CANopen differentiates the following access types:

Access Types for the Object Dictionary

1. const
   Read-only, and always delivers the same value.
2. read only
   Read-only, however, the value can be changed during operation.
3. write only
   The entry is write-only.
4. read write
   The entry can be read from and written to.

CANopen differentiates between the following areas of the data dictionary:

Areas of the Object Dictionary:

5. Index 0 .. 1FFF: *Communication profile specific entries*
   Settings which apply to all CANopen devices. These entries are defined in DS301.
6. Index 6000 .. 9FFF: *Device profile specific entries*
   Device-specific data which are defined in a *Draft Standard*. The HDA 7000 CAN has implemented the device profile DS404.
7. Index 2000 .. 5FFF: *Manufacturer specific entries*
   Manufacturer-specific, additional data which is not defined in any specification.

8.2 Structure of the device-specific part according to DS404

The HDA 4000/7000 CANopen (PT) implements DS404. This describes the performance and the functionality of measuring devices and closed-loop controllers.

First, some notes on the layout of the *Object Dictionary*:

Multi-channel capability
DS404 is aimed at multi-channel instruments. This means that the number of channels is given for all entries in *Subindex 0* and then the value in the appropriate *Subindex*. The HDA 4000/7000 CANopen (PT) puts one or two channels into effect. Therefore the Subindex 0 contains the value 1 or 2 throughout and the actual measured values or settings are stored in Subindex 1 or Subindex 2.

Data types
DS404 is set up in such a way that each data type has its own section in the *Object Dictionary*:

Index	Data type
6000 to 6FFFh	Floating Point or specially encoded data (status, etc.)
7000 to 7FFFh	16 bit integer
8000 to 8FFFh	32 bit integer
9000 to 9FFFh	24 bit integer

Function blocks
DS404 subdivides a device into different function blocks: analogue input block, analogue output block, digital input block, digital output block, controller block and alarm block.
The HDA 4000/7000 CANopen (PT) has implemented the analogue input part *(Analogue Input Function Block*). These entries are in the area X100h to X1FFh.

Order of entries
As a result of this segmenting, the order of the entries relates only to the value of the last 3 digits of the index. The first digit specifies the data type and is ignored as far as classification is concerned.

9 Entries in the Object Dictionary

Listed below are the functionalities implemented by the HDA 4000/7000 CANopen (PT). A detailed description of the entries can be found in profiles [1] and [2].

9.1 Communication Profile Specific Entries (DS301)

9.1.1 Index 1000: DeviceType (read only)

Contains the number of the device profile being used, in this case the number 404, as well as the profile-specific extension, in this case a 2, for support of the analogue input device.

9.1.2 Index 1001: ErrorRegister (read only)

Contains the actual error condition (see *EMCY,* and [1]).

9.1.3 Index 1002: ManufacturerStatusRegister (read only)

Contains different error flags.
In the 32bit value the bits have the following meaning:

Bit0: 1= error when reading the Setup Data
Bit1: 1= error when writing to the Setup Data
Bit2: 1= error when reading the factory settings
Bit3: 1= internal error during the A/D conversion
Bit4: 1= internal error in the A/D coupling
Bit5: 1= internal bus error
Bit6: 1= error during the internal program monitoring

Bits 0 to 6 are deleted when the error is eliminated. To find out if an error had occurred, these 7 bits 0x00 to 0x06 are mirrored in bits 0x10 to 0x16. The mirrored bits are not deleted if the error occurred once.

9.1.4 Index 1005: SyncMessageIdentifier (read write)

Use this to adjust the *COB ID* for the *SYNC o*bject.

9.1.5 Index 1008: ManufacturerDeviceName (const)

Delivers the device name as a character string. („HDA4“) or („HDA7“).

9.1.6 Index 1009: ManufacturerHardwareVersion (const)

Delivers the hardware version as a character string (e.g. "HV02").

9.1.7 Index 100A: ManufacturerSoftwareVersion (const)

Delivers the software version as a character string (e.g. "0101"). The first two characters indicate the version, the last two the revision status.

9.1.8 Index 1010: StoreParameters

By entering the character string "save" the current settings are transferred to the non-volatile memory.
The HDA 4000/7000 CANopen (PT) does not automatically store settings if they are changed, but only when requested.

WARNING: Any changed settings must be saved explicitly using *StoreParameters*, otherwise they will be lost when the instrument is switched off or when the *NMT* commands *Reset Node* and *Reset Communication* are carried out.

CANopen has the option of restoring different parameter areas with the help of various *Subindexes*. The subindexes 1,2 and 3 are supported.

For further information, please see profile [1].

Subindices used:

0: LargestSubindexSupported (read only)
1: StoreAllParameters (read write)
2: StoreCommunicationParamters (read write) (index of 0x1000 to 0x1FFF)
3: StoreApplicationParamteres (read write) (index of 0x6000 to 0x9FFF)

9.1.9 Index 1011: RestoreDefaultParameters

By entering the character string "load" the factory settings are transferred into the non-volatile memory.
However the HDA 4000/7000 CANopen (PT) goes on working with the actual settings until it is switched off or until the commands *Reset Node* and *Reset Communication* are carried out.

CANopen has the option of restoring different parameter areas with the help of various *Subindexes*.
The subindexes 1,2 and 3 are supported.

For further information, please see profile [1].

Warning: The Baud rate and Node ID settings remain unchanged with RestoreDefaultParameter.

Subindices used:

0: LargestSubindexSupported (read only)
1: RestoreAllDefaultParameters (read write)
2: RestoreCommunicationParamters (read write) (index of 0x1000 to 0x1FFF)
3: RestoreApplicationParamteres (read write) (index of 0x6000 to 0x9FFF)

9.1.10 Index 1014: CobIdEmergencyMessage (read write)

Use this to adjust the *COB ID* for the *EMCY* object (see *EMCY*).

9.1.11 Index 1015: InhibitTime (read write)

The Inhibit Time indicates waiting period of the Node from the time when a critical failure occurrs until sending the Emergency Object.

9.1.12 Index 1017: ProducerHeartbeatTime (read write)

Use this to adjust the *Heartbeat* time in milliseconds. The value 0 indicates that this function is not active (see *Heartbeat*).

9.1.13 Index 1018 IdentityObject

The Identity Object identifies the HDA 4000/7000 CANopen (PT). The identification consists of four 32bit numbers. Combining these 4 numbers, you get a worldwide uniquely assigned identification for a device.

Subindices used:

0: LargestSubIndexSupported (read only)
1: VendorID (read only)
Unique manufacturer code (0xda for HYDAC ELECTRONIC GmbH)
2: ProductCode (read only)
Hydac Electronic product code
3: RevisionNumber (read only)
Revision number of the device
4: SerialNumber (read only)
Serial number of the device

Product Code

0x001	HDA 7000
0x003	HDA 7000 P+T
0x004	HDA 4000
0x005	HDA 4000 P+T

9.1.14 Index 1800: TransmitPDOParameters

These entries determine the PDO transmission. In detail these are:

Parameters for PDO transmission

1. *COB ID*
   Determines the identifier for the PDO. The highest value bit (Bit31) of the entry no long belongs to the ID and has the meaning "*disable PDO*". If this bit is set, then transmission of the PDO is disabled.
2. *Transmission Type*
   Determines the transmission type.
   Values between 0 and 240 mean synchronous transmission. The figure represents the number of SYNC objects which have to be received before the PDO is sent. The value 254 indicates a manufacturer-specific transmission and the value 255 a device-profile-specific transmission. With 254 and 255 the PDO is sent cyclically, providing a time (*Event Time*) other than 0 is set. With 255 another transmission takes place if the measured value differs from the value of the last transfer by more than a set amount, or if the value exceeds or falls below the pre-set limits (see *Object Dictionary, Index* 7133, 7134 and 7135).
3. *Event Time*
   Determines the cycle time for asynchronous transmissions for Transmission Types 254 and 255 in milliseconds. The value 0 denotes no time-controlled transmission.

Subindices used:

0: LargestSubindexSupported (read only)
1: COBIDUsedByPDO (read write)
2: TransmissionType (read write)
5: EventTimer(read write)

9.1.15 Index 1A00: TransmitPDOMapping

Use this entry to determine which data is transferred with the PDO. Subindex 0 indicates the amount of data in the PDO. The index and subindex and the number of bits of the first piece of data is stored in Subindex 1, similarly for Subindex 2.

When supplied, the HDA 4000/7000 CANopen (PT) has the following entries:

Sub Index	Contents	Meaning
0	2	Two values are transferred in the *PDO.*
1	0x91300120	The first value in the *PDO* is the value of *Index 9130*, *Subindex* 01 with a width of 20h (=32 bit)
2	0x61500108	The second value in the PDO is the value of *Index 6150*, *Subindex* 01 with a width of 8 bit

Subindices used:

0: NumberOfMappedApplicationObjectsInPDO (read write)
Values 0, 1, 2, 3 and 4 are permitted. This means: no PDO is transmitted, or one PDO with one or two values is transmitted.

1: 1stObjectToBeMapped (read write)
In HDA 4000/7000 CANopen (PT) the following values are permitted:
0x71300110 = actual process value1 with 16bit width, or
0x91300120 = actual process value1 with 32bit width, or
0x61300120 = actual process value1 in Float32 version, or
0x61500108 = status of 1st analogue input channel.
In case a second sensor is available, also:
0x71300210 = actual process value2 with 16bit width, or
0x91300220 = actual process value2 with 32bit-Breite, or
0x61300220 = actual process value2 in Float32 version, or
0x61500208 = status of 2nd analogue input channel.

2: 2ndObjectToBeMapped (read only)
as in 1:

Note: The HDA 4000/7000 CANopen (PT) carries out only a 16bit evaluation. Showing the value as a 32bit value is just to comply with the *Default-PDO-Mapping* of DS404.

9.1.16 Index 1F80: NMT-Startup (read / write)

If bit 2 is set, status is changed automatically to OPERATIONAL state immediately after reaching PRE-OPERATIONAL mode. Permitted values are: 0x8 and 0xC.

9.2 Device Profile Specific Entries (DS404)

9.2.1 Index 7100: FieldValue

This entry contains an unscaled value which delivers the signal unit (see "Data flow in the HDA 4000/7000 CANopen (PT)", Page 114).

12Bit value of the measured value 0=0% within the measuring range; 4096 = 100% of the measuring range

Subindices used:

0: NumberOfEntries (read only)
1: FieldValue1 (read only)

In case a second sensor is available, also:
2: FieldValue2 (read only)

9.2.2 Index 6110: SensorType

This entry contains the sensor type.

Subindices used:

0: NumberOfEntries (read only)
1: SensorType1 (read only)
In case a second sensor is available, also:
2: SensorType2 (read only)
Values: 0x5A for pressure transducers
0x64 for temperature

9.2.3 Index 6111: AutoCalibration

By inputting the character string "cali", the scaling unit is calibrated to 0 (see "Data flow in the HDA 4000/7000 CANopen (PT)", Page 14). For this for the HDA 4000/7000 CANopen (PT) must be depressurised and temperature must be within the tolerance. If the value is outside the permitted tolerance band of +/- 3% FS (Full Scale = relative to the complete measuring range), the transmission will be aborted with the Code 06090030h (see [1], SDO Abort Codes).

Subindices used:

0: NumberOfEntries (read only)
1: AutoCalibration1 (write only)

9.2.4 Index 6112: OperatingMode

With this entry, the input channel can be set to a particular operating mode or switched off. In HDA 4000/7000 CANopen (PT) the channels cannot be switched off. This is why the entry can only be read and always provides value 1 = Normal Operation

Subindices used:

0: NumberOfEntries (read only)
1: OperatingMode1 (read only)
In case a second sensor is available, also:
2: OperatingMode2 (read only)

9.2.5 Index 6114: ADCSampleRate

Use this entry to stipulate the sample rate. The figure is a multiple of milliseconds, according to the standard, the HDA 4000/7000 CANopen (PT) always rounds down to whole milliseconds. All figures under 1000 produce a sample rate of 1 millisecond.

Subindices used:

0: NumberOfEntries (read only)
1: ADCSampleRate (read write)
In case a second sensor is available, also:
2: ADCSampleRate2 (read write)

9.2.6 Index 7120: InputScaling1FV

This entry is used for scaling the process value and contains the value of the *FieldValue* for the lower measuring range limit. Normally, at InputScaling 1FV1, it corresponds to a pressure of 0 bar (see "Data flow in the HDA 4000/7000 CANopen (PT)", Page 14 and [1], *Analogue Input Function Block*).

Subindices used:

0: NumberOfEntries (read only)
1: InputScaling1FV1 (read only)
In case a second sensor is available, also:
2: InputScaling1FV2 (read only)

9.2.7 Index 6121/7121/9121: InputScaling1PV

This entry is used to scale the process value and contains the value of *ProcessValue* for the lower measuring range limit. Index 7121 has a data width of 16bit and Index 9121 of 32bit, Index 6121 contains a 32bit with Float Number. Normally, this corresponds with a pressure of 0bar (at Input Scaling 1PV1). Altering this entry, any measuring ranges can be realised. If, for example, the measuring range is to be 50..700 kg, then the value 50 must be input (see "Data flow in the HDA 4000/7000 CANopen (PT)", Page 14 and [1], *Analogue Input Function* Block). The value is signed.

Subindices used:

0: NumberOfEntries (read only)
1: InputScaling1PV1 (read write)
In case a second sensor is available, also:
2: InputScaling1PV2 (read write)

9.2.8 Index 7122: InputScaling2FV

This entry is used for scaling the process value and contains the value of *FieldValue* for the upper measuring range limit (see "Data flow in the HDA 4000/7000 CANopen (PT)", Page 14 and [1], *Analogue Input Function Block*).

Subindices used:

0: NumberOfEntries (read only)
1: InputScaling2FV1 (read only)
In case a second sensor is available, also:
2: InputScaling2FV2 (read only)

9.2.9 Index 6123/7123/9123: InputScaling2PV

This entry is used for scaling the process value and contains the value of *ProcessValue* for the upper measuring range limit. Index 7123 has a data width of 16bit and Index 9123 of 32bit, Index 6123 contains a 32bit with Float Time. By changing this entry, any measuring range can be implemented. If, for example, the measuring range is to be 50..700 kg, then the value 50 must be input (see "Data flow in the HDA 4000/7000 CANopen (PT)", Page 14 and [1], *Analogue Input Function Block*).

Subindices used:

0: NumberOfEntries (read only)
1: InputScaling2PV1 (read write)
In case a second sensor is available, also:
2: InputScaling2PV2 (read write)

9.2.10 Index 6124/7124/9124: InputOffset

This entry defines an additional offset for the process value (see "Data flow in the HDA 4000/7000 CANopen (PT)", Page 14 and [1], *Analogue Input Function Block*).
Index 7124 has a data width of 16bit and index 9124, 32bit. Index 6124 contains a 32 bit wide Float Number.

Subindices used:

0: NumberOfEntries (read only)
1: InputOffset1 (read write)
In case a second sensor is available, also:
2: InputOffset2 (read write)

9.2.11 Index 6125/7125: InputAutoZero

By inputting the character string "zero", the *InputOffset* is set in such a way that the actual process value delivers 0 (see "Data flow in the HDA 4000/7000 CANopen (PT)", Page 14 and [1], *Analogue Input Function Block*).
Index 7125 has a data width of 16bit and Index 6125 contains a 32 bit wide Float Number.

With a read access to Index 1 an error is triggered (abort = ReadError).
With a write access to Index 1, it it checket if "zero" stands in the input buffer. If this is the case, the MeasuredOffset is set, if it is not, an error is reported (abort = CanNotTransfer).

Subindices used:
0: NumberOfEntries (read only)
1: InputAutoZero1 (write only)
In case a second sensor is available, also:
2: InputAutoZero2 (write only)

9.2.12 Index 6130/7130/9130: ProcessValue

These entries contain the actual process value. Index 7130 has a data width of 16bit and Index 9130 of 32bit, Index 6130 contains a 32bit with Float Number. Since HDA 4000/7000 CANopen (PT) generally calculates with a data width of 16bit, all integer entries provide the same value. These are implemented to enable specification-compliant *PDO-Mapping*. This stipulates the 32 bit value as the default. To reduce the data traffic in the network, the 16bit value can however be mapped (see "Data flow in the HDA 4000/7000 CANopen (PT)", on Page 14 and "TransmitPDOMapping" on Page 21).

Subindices used:

0: NumberOfEntries (read only)
1: ProcessValue1 (read only)
In case a second sensor is available, also:
2: ProcessValue2 (read only)

9.2.13 Index 6131: PhysicalUnitProcessValue

This entry contains the unit of measurement of the process value according to the recommendation DR303.
The preset unit of measurement has no effect on signal processing. To interpret the data, see [3]

Subindices used:

0: NumberOfEntries (read only)
1: PhysicalUnitProcessValue1 (read write)
In case a second sensor is available, also:
2: PhysicalUnitProcessValue2 (read write)
Values: "°C" when delivered
"bar" when delivered

9.2.14 Index 6132: ProcessValueDecimalDigits

This entry contains the number of decimal digits for interpreting the process value.

Example:
If the *ProcessValueDecimalDigits1* is set to 2 and *PhysicalUnitProcessValue1* is set to "bar", then a *ProcessValue1* of 4711 corresponds to the measured value 47.11 bar.
If the *ProcessValueDecimalDigits2* is set to 1 and *PhysicalUnitProcessValue2 is set to"℃"* then a *ProcessValue2* of 365 corresponds to the measured value 36,5℃.

Subindices used:

0: NumberOfEntries (read only)
1: ProcessValueDecimalDigits1 (read write)
In case a second sensor is available, also:
2: ProcessValueDecimalDigits2 (read write)

9.2.15 Index 6133/7133/9133: InterruptDeltaPV

This entry is used to control the *PDO* transmission. Index 7133 has a data width of 16bit and index 9133 of 32bit. Index 6133 contains a 32 bit wide Float Number. If the actual *ProcessValue* deviates from the last one transmitted by more than the *InterruptDeltaPV*, it is transmitted. This prevents constant transmissions where the contents are very similar. In order to activate this mechanism, *TransmissionType* must be set to *ProfileSpecific* (see *TransmissionType,* Page 21).

If *InterruptDeltaPV* is set to 0, then this mechanism is also deactivated.

Subindices used:

0: NumberOfEntries (read only)
1: InterruptDeltaPV1 (read write)
In case a second sensor is available, also:
2: InterruptDeltaPV2 (read write)

9.2.16 Index 6134/7134/9134: InterruptLowerLimit

This entry is used to control the *PDO* transmission. Index 7134 has a data width of 16bit and Index 9134 of 32bit, Index 6134 contains a 32bit with Float Number. If the actual ProcessValue falls below the limit InterruptlowerLimit it is transmitted. In order to avoid constant transmissions because the measured value is fluctuating around the limit value, the actual value, the actual value must rise above the limit again by at least 1% of the preset measuring range, before another transmission is made for a fall in value.

In order for this mechanism to be activated, the *TransmissionType* must be set to *ProfileSpecific* (See *TransmissionType,* Page 21).

Subindices used:

0: NumberOfEntries (read only)
1: InterruptLowerLimit1 (read write)
In case a second sensor is available, also:
2: InterruptLowerLimit2 (read write)

9.2.17 Index 6135/7135/9135: InterruptUpperLimit

This entry is used to control the *PDO* transmission. Index 7135 has a data width of 16bit and Index 9135 of 32bit, Index 6135 contains a 32bit with Float Number. If the actual ProcessValue exceeds the limit InterruptlowerLimit it is transmitted. In order to avoid constant transmissions because the measured value is fluctuating around the limit value, the actual value must exceed the limit again by at least 1% of the preset measuring range, before another transmission is made for this event.

In order for this mechanism to be activated, the *TransmissionType* must be set to *ProfileSpecific* (See *TransmissionType,* Page 21).

Subindices used:

0: NumberOfEntries (read only)
1: InterruptUpperLimit1 (read write)
In case a second sensor is available, also:
2: InterruptUpperLimit2 (read write)

9.2.18 Index 6136/7136/9136: TriggerHysteresis

This entry is used to preset the trigger hysteresis. Index 7136 has a data width of 16bit and Index 9136 of 32bit, Index 6136 contains a 32bit with Float Number. Only after falling below or exceeding the preset value in the hysteresis (at the lower / upper trigger limit) the trigger is actuated. (Thus, it can be avoided that the trigger is permanently actuated when the values fluctuate around the preset upper or lower trigger limit.)

Subindices used:

0: NumberOfEntries (read only)
1: TriggerHysteresis1 (read write)
In case a second sensor is available, also:
2: TriggerHysteresis2 (read write)

9.2.19 Index 6150: AnlogInputStatus

This entry indicates the status of the signal unit. The status byte has the following meaning:

Bit 0: 1=fault on the signal input
Bit 1: 1=positive overload present
Bit 2: 1=negative overload present

Subindices used:

0: NumberOfEntries (read only)
1: AnlogInputStatus1 (read only)
In case a second sensor is available, also:
2: AnlogInputStatus2 (read only)

9.2.20 Index 61A0: FilterType

This entry determines the filter type. 1 indicates *Moving average*, 2 indicates *Repeating average*. For all other values, there is no filtration. The exact procedure is described in profile [2].

Subindices used:

0: NumberOfEntries (read only)
1: FilterType1 (read write)
In case a second sensor is available, also:
2: FilterType2 (read write)

9.2.21 Index 61A1: FilterConstant

This entry determines the filter constant. Filtering only takes place, if the constant is more than 1. The exact proceeding is described in [2].

Subindices used:

0: NumberOfEntries (read only)
1: FilterConstant1 (read write)
In case a second sensor is available, also:
2: FilterConstant2 (read write)

9.3 Manufacturer Specific Entries

9.3.1 Index 2001: NodeID (read write)

The *Node ID* is stored in this index. To set the *NODE ID* the character string "set" must be added to the address for safety.

NodeID	´s´=0x73	´e´=0x65	´t´=0x74

For example, to set the *Node-ID* to 5, the following data bytes must be transmitted in the SDO: 0x05, 0x73, 0x65, 0x74. Because of the low-high representation, with CANopen therefore, the following 32-bit value must be transmitted: 0x74657302.

In order for the new *Node ID* to be effective, the command *StoreParameters* must first be transmitted and the node must be restarted afterwards.

9.3.2 Index 2002: Baud rate (read write)

The Baud rate is stored in this index. To set the Baud rate, the character string "set" must be added for safety.

The allocation of the entry to the Baud rate complies with DS305, *Layer Setting Services and Protocols*.

Entry	**0**	**1**	**2**	**3**	**4**	**6**	**7**	**8**	**9**
Baud rate	1 Mbit/s	800 kbit/s	500 kbit/s	250 kbit/s	125 kbit/s	50 kbit/s	20 kbit/s	10 kbit/s	auto

Baud rate	´s´=0x73	´e´=0x65	´t´=0x74

For example, to set the Baud rate to 125kbit/s, the following data bytes must be transmitted in the SDO: 0x04, 0x73, 0x65, 0x74. Because of the low-high representation, with CANopen therefore, the following 32-bit value must be transmitted: 0x74657304.

In order for the new *Baud rate* to be effective, the command *StoreParameters* must first be transmitted and the node must be restarted afterwards.

If the baud rate is set to 9, the HDA 7000 CAN will search the actual baud rate with the help of listen bus frames. For further information, please see profile [5].

9.3.3 Index 2121: OriginalInputScaling1PV

This entry contains the original lower measuring range limit. When supplied the value is identical to Index 7121 *InputScaling1PV*. *InputScaling1PV* can however be changed during operation, to implement other process variables. If the command *RestoreDefaultParameters* is received, the *OriginalInputScaling1PV* is copied to *InputScaling1PV*.

Subindices used:

0: NumberOfEntries (read only)
1: OriginalInputScaling1PV1 (read only)
In case a second sensor is available, also:
2: OriginalInputScaling1PV2 (read only)

9.3.4 Index 2123: OriginalInputScaling2FV

This entry contains the original upper *FieldValue*. When supplied the value is identical to Index 7122 *InputScaling2FV*. *InputScaling2FV* can however be changed during operation to implement other process variables. If the command *RestoreDefaultParameters* is received, *OriginalInputScaling2FV* is copied to *InputScaling2FV*.

Subindices used:

0: NumberOfEntries (read only)
1: OriginalInputScaling2FV1 (read only)
In case a second sensor is available, also:
2: OriginalInputScaling2FV2 (read only)

9.3.5 Index 2131: OriginalPhysicalUnitProcessValue

This entry contains the originally set unit. When supplied, the value is identical to Index 6131 *PhysicalUnitProcessValue*. *PhysicalUnitProcessValue* can however be changed during operation to implement other process variables. If the command *RestoreDefaultParameters* is received, *OriginalPhysicalUnitProcessValue* is copied to *PhysicalUnitProcessValue*.

Subindices used:

0: NumberOfEntries (read only)
1: OriginalPhysicalUnitProcessValue1 (read only)
In case a second sensor is available, also:
2: OriginalPhysicalUnitProcessValue2 (read only)

9.3.6 Index 2132: OriginalDecimalDigitsProcessValue

This entry contains the original number of decimal places. When supplied the value is identical to Index 6131 *DecimalDigitsProcessValue*. *DecimalDigitsProcessValue* can however be changed during operation to implement other process variables. If the command *RestoreDefaultParameters* is received, *OriginalDecimalDigitsProcessValue* is copied to *DecimalDigitsProcessValue*.

Subindices used:

0: NumberOfEntries (read only)
1: OriginalDecimalDigitsProcessValue1 (read only)
In case a second sensor is available, also:
2: OriginalDecimalDigitsProcessValue2 (read only)

10 Layer setting services (LSS) and protocols

The LSS services and protocols, documented in CIA DS-305 V2.2 (see [4]), are used to inquire or to change the settings of certain parameters of the Data Link Layers and the Application Layers of a LSS slave by a LSS master via the CAN network.

Following parameters are supported:

- Node ID
- Baud rate
- LSS address, under the terms of identity object 1018h

Access to the LSS slave thereby is made by its LSS address, consisting of:

- Vendor ID
- **Product Code**
- Revision number
- Serial number

The measuring system supports the following services:

Switch mode services

- Switch mode selective
  - To address a specific of LSS slave
- Switch mode global
  - To address the total of LSS slaves

Configuration services

- Configure Node ID
  - Configure Node ID
- Configure bit timing parameters
  - Configure baud rate
- Activate bit timing parameters
  - Activate baud rate
- Store configured parameters
  - Store configured parameters

Inquiry services

- Inquire LSS-address
  - Inquire LSS address
- Inquire Node ID
  - Inquire Node ID

Identification services

- LSS identify remote slave
  - Identification of LSS slaves within an certain array
- LSS identify slave
  - Response of all LSS slaves to the previous command
- LSS identify non-configured remote slave
  - Identification of non-configured LSS slaves, Node ID = FFh
- LSS identify non-configured slave
  - Response of all LSS slaves to the previous command

10.1 Finite state automaton, FSA

The FSA is equivalent to a state machine and defines the behaviour of an LSS slave. This state behaviour is controlled by LSS COBs produced by the LSS master or NMT COBs generated by a NMT master or local NMT transitions.

The LSS Modes support the following states:
Initial: Pseudo-State, shows the activation of the FSAs
(1) LSS waiting: Support for all services as indicated below
(2) LSS configuration: Support for all services as indicated below
(3) Final: Pseudo-State, shows the deactivation of the FSAs

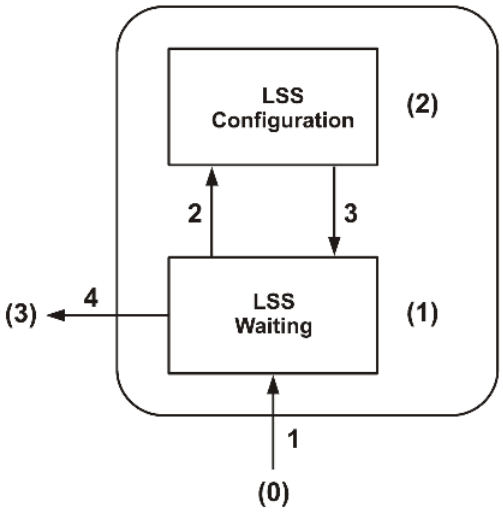


Figure 1: LSS Modes

State behavior of the supported services

Services	Waiting	Configuration
Switch mode global	Yes	Yes
Switch mode selective	Yes	No
Activate bit timing parameters	No	Yes
Configure bit timing parameters	No	Yes
Configure Node ID	No	Yes
Store configuration	No	Yes
Inquire LSS address	No	Yes
LSS identify remote slave	Yes	Yes
LSS identify slave	Yes	Yes
LSS identify non-configured remote slave	Yes	Yes
LSS identify non-configured slave	Yes	Yes

LSS FSA state transitions

Transition	Events	Actions
1	Automatic transition after initialisation on entering either the state NMT PRE OPERATIONAL mode or NMT STOPPED state, or NMT RESET COMMUNICATION mode with Node ID = FFh.	None
2	LSS 'switch state global' command with parameters 'configuration_switch' or 'switch state selective' command	None
3	LSS 'switch state global' command with parameter 'waiting_switch'	None
4	Automatic transition when an invalid Node ID has been changed and the new Node ID could be successfully stored in non-volatile memory AND the state of LSS waiting has been requested.	None

Once the LSS FSA further state transitions of the NMT FSA on NMT PRE into OPERATIONAL STOPPED and experienced vice versa, the result is not the re-entry into the LSS FSA.

10.2 Transmission of LSS services

By means of LSS services, the LSS master requests services to be performed by the LSS slave. Communication between LSS master and LSS slave is carried out by means of implemented LSS protocols.
Similar as in the case of SDO transmission, here, two COB IDs for sending and receiving are used as well:

COB ID	Meaning
0x7E4	LSS-Slave → LSS-Master
0x7E5	LSS-Master → LSS-Slave

Table 1: COB IDs for LSS services

10.2.1 LSS message format

The data field of a CAN message being max. 8 byte long is used by a LSS service as follows:

CS	Data						
Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7

Table 2: LSS message

Byte 0 contains the ***Command-Specifier*** (CS), afterwards 7 bytes of data are following.

10.3 Switch mode protocols

10.3.1 Switch mode global Protokoll

The given protocol has implemented the *Switch mode global service* and controls the state behavior of the LSS slave. By means of the LSS master all LSS slaves in the network can be switched to *Waiting Mode* or *Configuration Mode*.

LSS-Master --> LSS-Slave

0	1	2	3	4	5	6	7
COB ID	**CS**	**Mode**	Reserved by CiA				
0x7E5	04	0 = Waiting Mode 1 = Configuration Mode					

10.3.2 Switch mode selective protocol

The given protocol has implemented the *Switch mode selective service* and controls the state behavior of the LSS slave. By means of the LSS master only this LSS slave in the network can be switched to *Configuration Mode*, whose LSS address attributes correspond with the LSS address.

LSS-Master --> LSS-Slave

0	1	2	3	4	5	6	7
COB ID	**CS**	**Vendor ID**			Reserved by CiA		
0x7E5	64	LSB		MSB			

0	1	2	3	4	5	6	7
COB ID	**CS**	**Product Code**			Reserved by CiA		
0x7E5	65	LSB		MSB			

0	1	2	3	4	5	6	7
COB ID	**CS**	**Revision number**			Reserved by CiA		
0x7E5	66	LSB		MSB			

0	1	2	3	4	5	6	7
COB ID	**CS**	**Serial number**			Reserved by CiA		
0x7E5	67	LSB		MSB			

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB ID	**CS**	Reserved by CiA					
0x7E4	68						

10.4 Configuration protocols

10.4.1 Configure Node ID Protokoll

The given protocol has implemented the *Inquire LSS address service*. By means of the LSS master the Node ID of a single LSS slave in the network can be configured. Only one device is to be switched to *Configuration Mode*. For storage of the new Node ID the *Store configuration protocol* must be transmitted to the LSS slave. To activate the new Node ID the NMT service Reset Communication (0x82) must be addressed.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Node ID**	Reserved by CiA					
0x7E5	17	1…127 and 255						

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Error Code**	**Spec. Error**	Reserved by CiA				
0x7E4	17							

Error Code
0: Protocol successfully completed
1…254: Reserved
255: application specific error occurred

Specific Error
if Error Code = 255 --> application specific error occurred,
otherwise reserved by CiA

10.4.2 Configure bit timing parameters protocol

The given protocol has implemented the *Configure bit timing parameters service*. By means of the LSS master the Baud rate of a single LSS slave or of all LSS slaves in the network can be configured. For storage of the new Baud rate the *Store configuration protocol* must be transmitted to the LSS slave.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Table Selector**	**Table Index**	Reserved by CiA				
0x7E5	19	0	0…7					

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Error Code**	**Spec. Error**	Reserved by CiA				
0x7E4	19							

Table Selector
0: Standard CiA Baud rate table

Table Index
0: 1 Mbit/s
1: 800 kbit/s
2: 500 kbit/s
3: 250 kbit/s
4: 125 kbit/s
5: 50 kbit/s
6: 20 kbit/s
7: 10 kbit/s

Error Code
0: Protocol successfully completed
1: selected Baud rate not supported
2…254: Reserved
255: application specific error occurred

Specific Error
if Error Code = 255 --> application specific error occurred,
otherwise reserved by CiA

10.4.3 Activate bit timing parameters protocol

The protocol activates the Baud rate configured via the *Configure bit timing parameters protocol* in all LSS slaves in the network being in *Configuration Mode*.

LSS-Master --> LSS-Slave

0	1	2	3	4	5	6	7
COB ID	**CS**	**Switch Delay [ms]**		Reserved by CiA			
0x7E5	21	LSB	MSB				

Switch Delay
The parameter Switch Delay defines the length of two delay periods (D1, D2) with equal length. These are necessary to avoid operating the bus with differing Baud rate parameters. After the time D1 and an individual processing duration, the switching internally in the LSS slave is performed. After the time D2 the LSS slave responds with CAN messages and the newly configured Baud rate.

The following applies:
Switch Delay > longest occurring processing duration of a LSS slave

10.4.4 Store configuration protocol

The given protocol has implemented the *Store configuration service*. By means of the LSS master the configured parameters of a single LSS slave in the network can be stored into the non-volatile memory. Only one device can be switched to Configuration Mode.

LSS-Master --> LSS-Slave

0	1	2	3	4	5	6	7
COB ID	**CS**	Reserved by CiA					
0x7E5	23						

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB-ID	**CS**	**Error Code**	**Spec. Error**	Reserved by CiA			
0x7E4	23						

Error Code
0: Protocol successfully completed
1: *Store configuration* not supported
2…254: Reserved
255: application specific error occurred

Specific Error
if Error Code = 255 --> application specific error occurred,
otherwise reserved by CiA

10.5 Inquire LSS address protocols

10.5.1 Inquire Identity Vendor ID protocol

The given protocol has implemented the *Inquire LSS address service*. By means of the LSS master the Vendor ID of a single LSS slave in the network can be configured. Only one device can be switched to *Configuration Mode*.

LSS-Master --> LSS-Slave

0	1	2	3	4	5	6	7
COB ID	**CS**	Reserved by CiA					
0x7E5	90						

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB ID	**CS**	**Vendor Id** (= Index 1018h:01)			Reserved by CiA		
0x7E4	90	LSB		MSB			

10.5.2 Inquire Identity Product Code Protocol

The given protocol has implemented the *Inquire LSS address service*. By means of the LSS master the product name of a single LSS slave in the network can be read out. Only one device can be switched to Configuration Mode.

LSS-Master --> LSS-Slave

0	1	2	3	4	5	6	7
COB ID	**CS**	Reserved by CiA					
0x7E5	91						

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB ID	**CS**	**Product Code** (= Index 1018h:02)			Reserved by CiA		
0x7E4	91	LSB		MSB			

10.5.3 Inquire Identity Revision Number Protocol

The given protocol has implemented the *Inquire LSS address service*. By means of the LSS master the Identity Revision Number of a single LSS slave in the network can be read out. Only one LSS slave can be switched to Configuration Mode.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB ID	**CS**	Reserved by CiA						
0x7E5	92							

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Revision Number** (= Index 1018h:03)				Reserved by CiA		
0x7E4	92	LSB			MSB			

10.5.4 Inquire Identity Serial Number protocol

The given protocol has implemented the *Inquire LSS address service*. Via the LSS master the Serial Number of a single LSS slave in the network can be read out. Only one LSS slave can be switched to Configuration Mode.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB ID	**CS**	Reserved by CiA						
0x7E5	93							

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Serial Number** (= Index 1018h:04)				Reserved by CiA		
0x7E5	93	LSB			MSB			

10.6 Inquire Node ID protocol

The given protocol has implemented the *Inquire Node ID service.*
Via the LSS master the Serial Number of a single LSS slave in the network can be read out. Only one LSS slave can be switched to Configuration Mode.

LSS-Master --> LSS-Slave

0	1	2	3	4	5	6	7
COB ID	**CS**	Reserved by CiA					
0x7E5	94						

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB ID	**CS**	**Node ID**	Reserved by CiA				
0x7E4	94	1…127 and 255					

Node ID
Corresponds to the Node ID of the selected device. If the Node ID recently has been changed by *the Configure Node ID service*, the original Node ID is returned. Only after implementation of the NMT Service Reset Communication (0x82), the current Node ID is returned.

10.7 Identification Protocols

10.7.1 LSS identify remote slave protocol

The given protocol has implemented the *LSS identify remote slaves service*. By means of the LSS master LSS slaves in the network can be identified within a certain range. All LSS slaves with matching Vendor ID, Product Code, Revision No. and Serial No. range, respond by the *LSS identify slave protocol*.

LSS-Master --> LSS-Slave

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Vendor ID**				Reserved by CiA		
0x7E5	70	LSB			MSB			

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Product Code**				Reserved by CiA		
0x7E5	71	LSB			MSB			

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Revision Number Low**				Reserved by CiA		
0x7E5	72	LSB			MSB			

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Revision Number High**				Reserved by CiA		
0x7E5	73	LSB			MSB			

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Serial Number Low**				Reserved by CiA		
0x7E5	74	LSB			MSB			

	0	1	2	3	4	5	6	7
COB ID	**CS**	**Serial Number High**				Reserved by CiA		
0x7E5	75	LSB			MSB			

10.7.2 LSS identify slave protocol

The given protocol has implemented the *LSS identify slave service*. All LSS slaves with matching LSS attributes given in the *LSS identify remote slaves protocol*, respond by this protocol.

LSS-Slave --> LSS-Master

	0	1	2	3	4	5	6	7
COB ID	**CS**	Reserved by CiA						
0x7E4	79							

10.7.3 LSS identify non-configured remote slave protocol

The specified protocol has implemented the LSS identify non-configured remote slave service. By means of the LSS-Master all non-configured LSS Slaves (Node ID = identified FFh) in the network are identified. *The related LSS Slaves respond by the LSS identify non-configured remote slave protocol.*

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB ID	**CS**	Reserved by CiA					
0x7E4	76						

10.7.4 LSS identify non-configured slave Protocol

The specified protocol has implemented the LSS identify non-configured slave service. All LSS slaves having an invalid Node ID (FFh) respond by this protocol after having completed the LSS identify non-configured slave protocol.

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB ID	**CS**	Reserved by CiA					
0x7E4	80						

10.7.5 Fastscan protocol

The specified protocol has implemented the LSS fastscan service. LSS slaves change into the Configuration Mode after having been identified via all 4 LSS sub entries.

LSS-Slave --> LSS-Master

0	1	2	3	4	5	6	7
COB ID	**CS**	**ID Number**			**Bit Check**	**LSS Sub**	**LSS Next**
0x7E4	81	LSB		MSB			

11 Connection

The connection can be carried out by means of the enclosed device specific pin assignment (see user manual HDA 7000 CANopen (part no. 669827). The user manual is included in the delivery of HDA 7000 CANopen).

11.1 Switching on the supply voltage

After connection and all settings have been carried out, the supply voltage can be switched on.

After POWER ON and finishing the initialization, the measuring system enters the PRE-OPERATIONAL state. This status is acknowledged by the Boot-Up message "**COB ID 0x700+Node ID**". If the measuring system detects an internal error, an emergency message with the error code are transmitted (see chapter "Emergency Object (EMCY)", page 11).

In PRE-OPERATIONAL state, at first, only a parameter setting via Service Data Objects is possible. But it is possible to configure PDOs with the help of SDOs. If the measuring system has entered the OPERATIONAL state, a transmission of PDOs is possible as well (see Network Management Services (NMT), page13).

11.2 Setting the Node ID and Baud rate by means of LSS services

11.2.1 Configuration of the Node ID, sequence

Assumption:

- LSS address unknown
- only one LSS slave should be in the network
- the Node ID 12 dec. shall be set

Procedure:

- Perform service 04 *Switch mode global protocol*,
  Mode = 1, to switch the LSS slave to *Configuration Mode*
  (see Switch mode global Protokoll, page35)..
- Command Specifier 17 *Configure NMT Address Protocol*,
  Node ID = 12 (see Configure Node ID Protokoll, page 36.
  --> Wait for feedback and check successful execution,
  --> Error Code = 0.

- Command Specifier 23 *Store Configuration Protocol*
  --> Wait for feedback and check successful execution,
  --> Error Code = 0.(see Store configuration protocol, page 38.

- Switch off the supply voltage of the LSS slave OFF and then ON again. Now the new configuration is activated.

11.2.2 Configuration of the Baud rate, sequence

Assumption:

- LSS address unknown
- only one LSS slave should be in the network
- the Baud rate 125 kbit/s shall be set

Procedure:

- Perform NMT service *Stop Remote Node* (0x02), to switch the LSS slave to *Stopped state*. The LSS slave shouldn't send any CAN-messages
  --> Heartbeat switched off (see Network Management Services (NMT), page 13).
- Perform service 04 *Switch mode global protocol*,
  Mode = 1, to switch the LSS slave to *Configuration Mode*
  (see Switch mode global Protokoll, page 35).
- Perform service 19 *Configure bit timing parameters protocol*, Table Selector = 0, Table Index = 4
  --> Wait for feedback and check successful execution,
  --> Error Code = 0
  (see Configure bit timing parameters protocol, page 37).
- Perform Command Specifier 21 *Activate bit timing parameters protocol*, to activate the new Baud rate (seeActivate bit timing parameters protocol, page38).
- Perform Command Specifier 23 *Store Configuration Protocol*
  --> Wait for feedback and check successful execution,
  --> Error Code = 0
  (see Store configuration protocol, page 38).
- Switch off the supply voltage of the LSS slave OFF and then ON again. Now the new configuration is activated.

12 Commissioning

12.1 CAN interface

The CAN bus interface is defined by the international standard ISO/DIS 11898 and specifies the two lowest layers of the ISO/DIS CAN Reference Model.

The conversion of the measuring system information to the CAN message format (CAN 2.0A) is done by a CAN-controller. The function of the CAN-controller is controlled by a watchdog.

The CANopen Communication Profile (CIA standard DS 301, see [1]) is a subset of CAN Application Layer (CAL) and describes how the services are used by devices. The CANopen Profile allows the definition of device profiles for decentralized I/O.

The measuring system with CANopen protocol support the Device Profile for measuring devices (CIA Draft Standard 404, Version 1.2, see [2]).

The communication functionality and objects, used in the measuring device profile, are described in an EDS-File (Electronic Data Sheet). When using a CANopen Configuration Tool (e.g.: CANSETTER), the user can read out the objects of the measuring system (SDOs) and program the functionality.

12.2 EDS file

The EDS (electronic datasheet) contains all information on the measuring system-specific parameters and the measuring system's operating modes. The EDS file is integrated using the CANopen network configuration tool to correctly configure or operate the measuring system.

The EDS file matching the device can be identified by the device name and by the device related major revision number in its file name.

The file including the its CANopen interface description can be found and downloaded from our homepage. Address: http://www.hydac.de/de-de/service/download-software-auf-anfrage/software/software-download/electronic.html.

13 Further literature:

[1] DS301, Version 4.02
Application Layer and Communication Profile
[2] DS404, Version 1.2
Device Profile Measuring Devices and Closed-Loop Controllers
[3] DR303-2, Version 1.0
Representation of SI Units and Prefixes
CiA Draft Standard Proposal 305
Layer setting services (LSS) and protocols
Version: 2.2 , 26 August 2008
CiA Application Note 801
CANopen Automatic bit-rate detection -Recommended practice and application hints
Version 1.0 01 January 2005

HYDAC ELECTRONIC GMBH
Hauptstr. 27
D-66128 Saarbrücken
Germany

Web: www.hydac.com
Email: electronic@hydac.com
Phone:+49 (0)6897 509-01
Fax: +49-(0)6897-509-1726

HYDAC Service
If you have any questions concerning repairwork, please don't hesitate to contact HYDAC Service:

HYDAC SERVICE GMBH
Hauptstr. 27
D-66128 Saarbrücken
Germany

Phone:+49 (0)6897 509-1936
Fax.: +49 (0) 6897 509-1933

NOTE

The information and particulars provided in this manual apply to the operating conditions and applications described herein. For applications or operating conditions not described, please contact the relevant technical department.

If you have any questions, suggestions, or encounter any problems of a technical nature, please contact your Hydac representative.

Subject to technical modifications.