6. Kommunikation

6.7 Terminalprogramm (Beispiel: Microsoft Windows Hyper Terminal)

Ist der Sensor mit einem PC verbunden und wird mit Spannung versorgt, kann mit ihm, unter Benutzung eines beliebigen Terminalprogramms, kommuniziert werden. Im Internet werden verschiedene Terminalprogramme als Freeware angeboten. Die einfachste Möglichkeit besteht darin, dass im Lieferumfang von Microsoft Windows enthaltene „Hyper Terminal“ zu benutzen. Standardmäßig ist dieses Programm unter **Start/Programme/Zubehör/Kommunikation** zu finden. Wenn Sie das Programm gestartet haben erscheinen nacheinander drei Fenster, in denen zunächst ein Name für die Verbindung, ein COM Port und die korrekten Kommunikationsparameter angegeben werden müssen. Die drei Fenster sind in **Abbildung 6.1** bis **Abbildung 6.3** dargestellt.

Abbildung 6.1: Microsoft Windows Hyper Terminal - Vergabe eines Namens für eine neue Verbindung.

Abbildung 6.2: Microsoft Windows Hyperterminal - Wahl der Schnittstelle zur Kommunikation. Hier COM Port 1.

Abbildung 6.3: Microsoft Windows Hyperterminal - Wahl der Schnittstellen-parameter.

6. Kommunikation

In dem nachfolgenden Eingabefenster können die entsprechenden Befehle zum Auslesen oder Konfigurieren eingegeben werden. Die Befehlsliste ist unter Kapitel 6.3 und 6.4 aufgeführt.

Beachten Sie hierbei, dass standardmäßig alle Zeichen, welche in das Terminalprogramm über die Tastatur eingegeben werden, nicht auf dem Bildschirm angezeigt werden. Dies kann im Hyper Terminal über die Option „Lokales Echo aktivieren“ geändert werden.

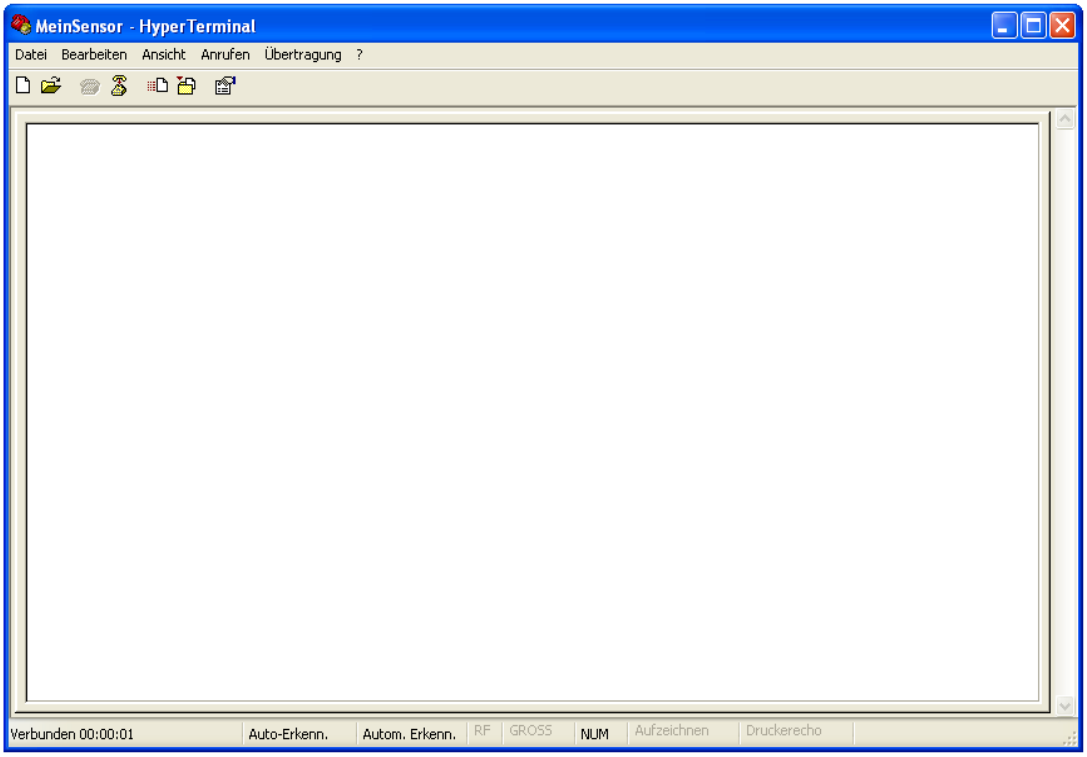

Abbildung 6.4: Windows Hyper Terminal - Eingabefenster

6.8 TCP/IP-Verbindung

Das Hyper Terminal bietet alternativ auch die Möglichkeit eine TCP/IP-Verbindung herzustellen. Sollen Sensoren über dieses Protokoll fernabgefragt werden, so ist die Wandlung des RS232-Signals mit Hilfe eines Ethernet-Gateways erforderlich. Passende Gateways können bei OLAER angefragt werden.

6.9 Software

OLAER stellt verschiedene Programme (Treiber, LabVIEW Tools und Hilfsprogramme) für den Bereich der Sensortechnik zur Verfügung.

7 CAN

7.1 CAN Kommunikation

Die CAN-Schnittstelle entspricht der „CAN 2.0B Active Specification“. Die Datenpakete entsprechen dem in **Abbildung 7.1** gezeigten Format. Die Abbildung dient nur Anschauungszwecken, die Umsetzung entspricht der CAN 2.0B Spezifikation.

Der Sensor unterstützt eine begrenzte Anzahl an Übertragungsgeschwindigkeiten auf dem CAN-Bus (vgl. **Tabelle 7.1**).

Durch CiA empfohlene und vom Sensor unterstützte Datenraten			
Datenrate	**Unterstützt**	**CiA Draft 301**	**Buslänge (nach CiA Draft Standard 301)**
1 Mbit/s	**Ja**	ja	25 m
800 kbit/s	**Nein**	ja	50 m
500 kbit/s	**Ja**	ja	100 m
250 kbit/s	**Ja**	ja	250 m
125 kbit/s	**Ja**	ja	500 m
100 kbit/s	**Nein**	nein	750 m
50 kbit/s	**Nein**	ja	1000 m
20 kbit/s	**nein**	ja	2500 m
10 kbit/s	**Nein**	ja	5000 m

Tabelle 7.1: Unterstützte Busgeschwindigkeiten bei CANopen Kommunikation und zugehörige Kabellängen

Die elektrischen Parameter der CAN-Schnittstelle sind in **Tabelle 7.2** aufgeführt.

Parameter	**Größe**	**Einheit**
Typ. Antwortzeit bei SDO-Anfragen	<10	ms
Max. Antwortzeit bei SDO-Anfragen	150	ms
Versorgungsspannung CAN-Transceiver	3,3	V
Terminierung integriert	nein	-

Tabelle 7.2: Elektrischen Parameter CAN-Schnittstelle

7. CAN

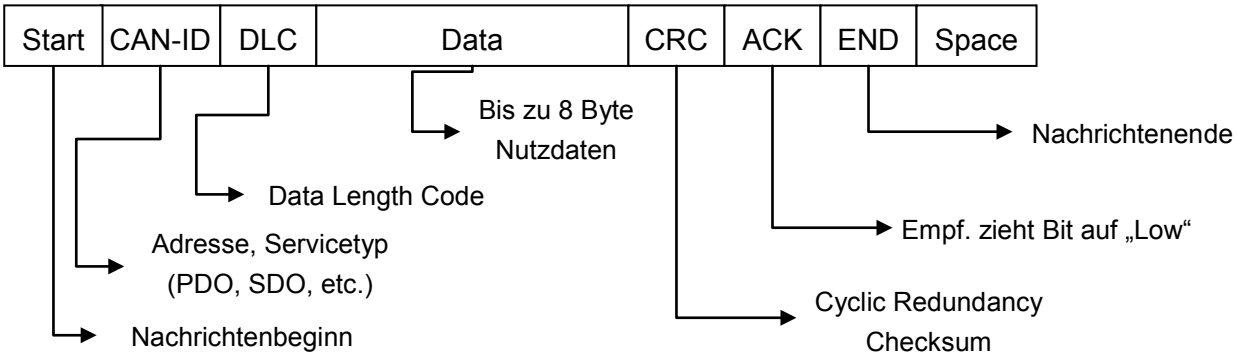

Abbildung 7.1:CAN Nachrichtenformat

7.2 CANopen

CANopen definiert das *was*, nicht das *wie* wird etwas beschrieben. Mit den implementierten Verfahren wird ein verteiltes Kontrollnetz umgesetzt das von sehr einfachen Teilnehmern bis zu sehr komplexen Steuerungen miteinander verbinden kann, ohne, dass es zu Kommunikationsproblemen zwischen den Teilnehmern kommt.

Das zentrale Konzept von CANopen ist das sogenannte **Device Object Dictionary** (OD), ein Konzept wie es ebenfalls bei anderen Feldbussystemen eingesetzt wird.

Im Nachfolgenden wird zuerst auf Object Dictionary, dann auf Communication Profile Area (CPA), und anschließend auf das *CANopen* Kommunikationsverfahren an sich eingegangen.

7.2.1 „CANopen Object Dictionary“ allgemein

Das *CANopen* Object Dictionary (OD) ist ein Objektverzeichnis in dem jedes Objekt mit einem 16-Bit Index angesprochen werden kann. Jedes Objekt kann aus mehreren Datenelementen bestehen, die über ein 8-Bit Subindex adressiert werden können.

Das prinzipielle Layout eines CANopen Objektverzeichnises ist in **Tabelle 7.3** dargestellt.

CANopen Object Dictionary	
Index (hex)	**Objekt**
0000	-
0001 - 001F	Statische Datentypen (Boolean, Integer)
0020 - 003F	Komplexe Datentypen (bestehend aus Standarddatentypen)
0040 - 005F	Komplexe Datentypen, herstellerspezifisch
0060 - 007F	Statische Datentypen (geräteprofilspezifisch)

0080 - 009F	Komplexe Datentypen (geräteprofilspezifisch)
00A0 - 0FFF	reserviert
1000 - 1FFF	Communication Profile Area (z.B. Gerätetyp, Fehlerregister, unterstützte PDOs,..)
2000 - 5FFF	Communication Profile Area (herstellerspezifisch)
6000 - 9FFF	Geräteprofilspezifische Device Profile Area (z.B. “DSP-401 Device Profile for I/O Modules”)
A000 - FFFF	reserviert

Tabelle 7.3: Allgemeine CANopen Object Dictionary Struktur

7.2.2 CANopen Communication Objects

Bei CANopen übertragene Kommunikationsobjekte sind durch Dienste und Protokolle beschrieben und sind folgendermaßen klassifiziert:

- Network Management (NMT) stellt Dienste und für Businitialisierung, Fehlerbehandlung, und Knotensteuerung
- Process Data Objects (PDOs) dienen zur Übertragung von Prozessdaten in Echtzeit
- Service Data Objects (SDOs) ermöglichen den Lese- und Schreibzugriff auf das Objektverzeichnis eines Knotens
- Special Function Object Protokol ermöglicht anwendungsspezifische Netzwerksynchronisation, Zeitstempel Übertragung und Emergency Nachrichten

Im Folgenden wird die Initialisierung des Netzes mit einem CANopen Master und einem Sensor beispielhaft beschrieben.

7. CAN

Nach Anlegen des Stromes verschickt der Sensor eine Boot-Up Nachricht innerhalb von ca. 5 Sekunden und sobald der Preoperational-Zustand erreicht ist. In diesem Zustand werden vom Sensor nur die Heartbeat-Nachrichten verschickt, falls er entsprechend konfiguriert ist (Punkt A in **Abbildung 7.2**).

Anschließend kann der Sensor über SDOs konfiguriert werden, in den meisten Fällen ist dies nicht notwendig, da die einmal eingestellten Kommunikationsparameter automatisch vom Sensor gespeichert werden (vgl. Punkt B in **Abbildung 7.2**).

Um den Sensor in den Operational Zustand zu versetzen kann entweder eine entsprechende Nachricht an alle CANopen Teilnehmer oder speziell an den Sensor verschickt werden. Im Operational Zustand verschickt der Sensor die unterstützten PDOs entsprechend seiner Konfiguration entweder in periodischen Zeitabständen oder auf Synch-Nachrichten getriggert (vgl. Punkt C in **Abbildung 7.2**).

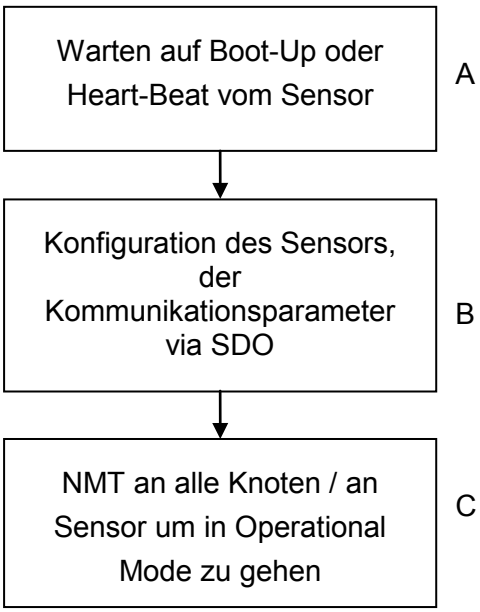

Abbildung 7.2: CANopen Bus Initialisierungsprozess

Je nach Zustand des Sensors stehen verschiedene Dienste des CANopen Protokolls zur Verfügung (vgl. **Tabelle 7.4**).

Verfügbarkeit der Dienste in Abhängigkeit des Sesnsorzustandes				
Com. Object	**Initialising**	**Pre-Operational**	**Operational**	**Stopped**
PDO			X	
SDO		X	X	
Synch		X	X	
BootUp	X			
NMT		X	X	X

Tabelle 7.4: Verfügbare CANopen Dienste in verschiedenen Sensorzuständen

7.2.3 Service Data Object (SDO)

Service Data Objects dienen dem Schreib- und Lesezugriff auf das Objektverzeichnis des Sensors. Die SDOs werden jeweils quittiert und die Übertragung findet immer nur zwischen zwei Teilnehmern statt, ein sogenanntes Client/Server-Model (vgl.: **Abbildung 7.3**).

Der Sensor kann ausschließlich als Server funktionieren, beantwortet also nur SDO-Nachrichten und schickt von sich aus keine Anfragen an andere Teilnehmer. Die SDO-Nachrichten vom Sensor an Client haben als ID die NodeID+0x580. Bei Anfragen vom Client an den Sensor (Server) wird bei der SDO-Nachricht als ID die NodeID+0x600 erwartet.

Das Standardprotokoll für SDO-Transfer, benötigt 4 Byte um die Senderichtung, Datentyp, den Index und den Subindex zu kodieren. Somit bleiben noch 4 Byte von den 8 Byte eines CAN-Datenfeldes für den Dateninhalt. Für Objekte, deren Dateninhalt größer als 4 Byte ist, gibt es zwei weitere Protokolle für sogenannten fragmentierten oder segmentierten SDO-Transfer.

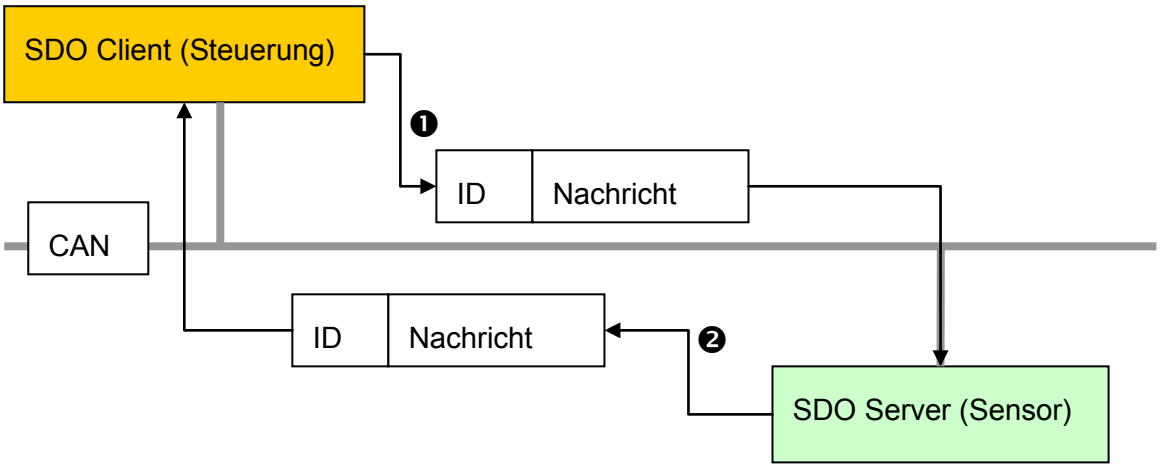

Abbildung 7.3:SDO Client/Server Beziehung

SDOs sind dazu gedacht den Sensor über Zugriff auf das Objektverzeichnis zu konfigurieren, selten benötigte Daten oder Konfigurationswerte anzufragen oder größere Datenmengen herunter zu laden. Die SDO Eigenschaften im Überblick:

- Auf alle Daten im Objektverzeichnis kann zugegriffen werden
- Bestätigte Übertragung
- Client/Server Beziehung bei der Kommunikation

Die Steuerungs- und Nutzdaten einer nicht segmentierten SDO-Standardnachricht verteilen sich auf die CAN-Nachricht wie es in **Tabelle 7.5** dargestellt ist. Die Nutzdaten einer SDO-Nachricht sind bis zu 4 Byte groß. Mit Hilfe der Steuerungsdaten einer SDO-Nachricht (Cmd, Index, Subindex) wird die Zugriffsrichtung auf das Objektverzeichnis und ggf. der übertragene Datentyp bestimmt. Für die genauen Spezifikationen des SDO Protokolls sollte der „CiA Draft Standard 301“ konsultiert werden.

CAN	CAN-ID	DLC	Nutzdaten CAN Message							
			0	1	2	3	4	5	6	7
CANopen	COB-ID	DLC	Cmd	Index		Subindex	Nutzdaten CANopen SDO Message			

SDO	11 Bit					

Tabelle 7.5: Aufbau einer SDO Nachricht

Ein Beispiel für eine SDO Abfrage der Seriennummer des Sensors aus dem Objektverzeichnis an Index 0x1018, Subindex 4, mit Datenlänge 32 Bit ist im Folgenden dargestellt. Der Client (Steuerung) schickt dazu eine Leseanfrage an den Sensor mit der ID „NodeID".(vgl. **Tabelle 7.6**)

CAN	CAN-ID	DLC	Nutzdaten CAN Message							
			0	1	2	3	4	5	6	7
CANopen	COB-ID 11 Bit	DLC	Cmd	Index		Subidx	Nutzdaten SDO			
				1	0	0	3	2	1	0
Nachricht vom Client an Sensor	0x600+ NodeID	0x08	0x40	0x18	0x10	0x04	dont care	dont care	dont care	dont care

Tabelle 7.6: SDO Downloadanfrage durch den Client an den Server

Der Sensor antwortet mit entsprechender SDO-Nachricht (vgl. **Tabelle 7.7**) in der der Datentyp, Index, Subindex und die Seriennummer des Sensors kodiert sind, hier beispielhaft die Seriennummer 200123 (0x30DBB).

CAN	CAN-ID	DLC	Nutzdaten CAN Message							
			0	1	2	3	4	5	6	7
CANopen	COB-ID 11 Bit	DLC	Cmd	Index		Subidx	Nutzdaten SDO			
				1	0	0	3	2	1	0
Nachricht vom Sensor an Client	0x580+ NodeID	0x08	0x43	0x18	0x10	0x04	0xBB	0x0D	0x30	0x00

Tabelle 7.7: SDO Downloadantwort durch den Server an den Client

Ein Beispiel für den Upload von Daten (Heartbeat-Zeit) über SDO in das Objektverzeichnis des Sensors an Index 0x1017 mit Datenlänge 16 Bit ist im Folgenden dargestellt. Der Client (Steuerung) schickt dazu eine Schreibanfrage an den Sensor mit der ID „NodeID" (vgl.**Tabelle 7.8**) um die Heartbeat-Zeit auf 1000 ms zu setzen (0x03E8).

CAN	CAN-ID	DLC	Nutzdaten CAN Message							
			0	1	2	3	4	5	6	7
CANopen	COB-ID 11 Bit	DLC	Cmd	Index		Subidx	Nutzdaten SDO			
				1	0	0	3	2	1	0
Nachricht vom Client an Sensor	0x600+ NodeID	0x08	0x2B	0x17	0x10	0x00	0xE8	0x03	0	0

Tabelle 7.8: SDO Uploadanfrage durch den Client an den Server

Der Sensor antwortet mit entsprechender SDO-Nachricht (vgl.Tabelle 7.9) in der die der bestätigt wird, dass der Zugriff erfolgreich war und der Index und Subindex kodiert sind auf die der Zugriff erfolgte.

CAN	CAN-ID	DLC	Nutzdaten CAN Message							
			0	1	2	3	4	5	6	7
CANopen	COB-ID 11 Bit	DLC	Cmd	Index		Subidx	Nutzdaten SDO			
				1	0	0	3	2	1	0
Nachricht vom	0x580+	0x08	0x60	0x17	0x10	0x00	0x00	0x00	0x00	0x00

Sensor an Client	NodeID									

Tabelle 7.9: SDO Uploadantwort durch den Server an den Client

7.2.4 Process Data Object (PDO)

PDOs sind ein oder mehrere Datensätze, die aus dem Objektverzeichnis in die bis zu 8 Bytes einer CAN-Nachricht gespiegelt sind um Daten schnell und mit möglichst wenig Zeitaufwand von einem „Producer“ zu einem oder mehreren „Consumern“ zu übertragen (vgl.: **Abbildung 7.4**). Jedes PDO hat eine einzigartige COB-ID (Communication Object Identifier), wird nur von einem einzigen Knoten verschickt, kann aber von mehreren Knoten empfangen werden und braucht nicht quittiert/bestätigt zu werden.

PDOs eignen sich ideal dazu Daten von Sensoren zur Steuerung oder von der Steuerung Daten zu Aktoren zu übertragen. PDO Attributen des Sensors im Überblick:

- Sensor unterstützt drei Sende-PDOs (TPDOs), keine Empfangs-PDOs (RPDOs). Die Level Sensoren unterstützen vier TPDOs.
- Das Mapping der Daten in PDOs ist fest und kann nicht verändert werden
- COB-IDs für TPDO1 und TPDO2 können frei gewählt werden, TPDO3 hat immer die COB-ID 0x380+NodeID (TPDO4 bei Level Sensoren hat immer die COB-ID 0x480+NodeID)
- TPDO1 und TPDO2 kann Event/Timer getriggert oder zyklisch auf SYNCH getriggert übertragen werden und ist jeweils für die beiden TPDOs individuell einstellbar, TPDO3 (und TPDO4 bei Level Sensoren) übernimmt die Einstellungen des TPDO2

Der Sensor unterstützt zwei unterschiedliche PDO Übertragungsmethoden.

1. Bei der Event- bzw. Timer-getriggerter Methode wird die Übertragung durch einen sensorinternen Timer oder Event ausgelöst
2. Bei der SYNCH-getriggerten Methode findet die Übertragung als Antwort auf eine SYNCH-Nachricht statt (CAN-Nachricht durch einen SYNCH-Producer ohne Nutzdaten). Die Antwort mit PDO erfolgt entweder bei jedem empfangenen Synch oder einstellbar alle n-Empfangene SYNCH-Nachrichten.

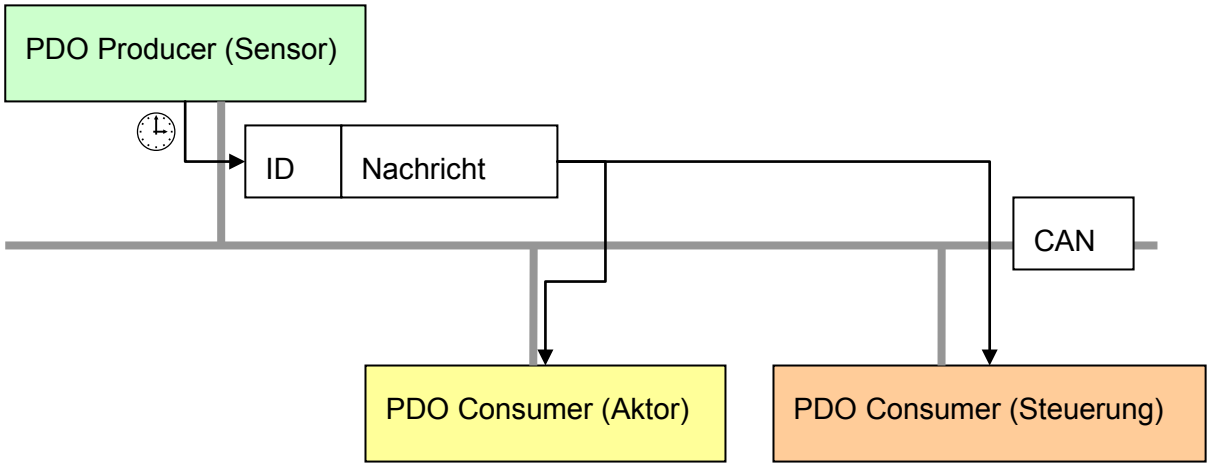

Abbildung 7.4: PDO Consumer/Producer Beziehung

7.2.5 PDO Mapping

Der Sensor unterstützt drei bis vier Transmit PDOs (TPDOs) um einen möglichst effizienten Betrieb des CAN-Busses zu ermöglichen. Der Sensor unterstützt kein dynamisches Mapping von PDOs, die Mappingparameter im OD sind also nur lesbar, aber nicht beschreibbar.

Abbildung 7.6 zeigt das Prinzip des Mappings von Objekten aus dem OD in ein TPDO, es entspricht der CiA DS-301, Kapitel 9.5.4. Welche Objekte in TPDO 1 bis 4 gemappt sind kann im OD an Index 0x1A00 bis 0x1A03 ermittelt werden. Die Struktur der PDO-Mappingeinträge ist in **Abbildung 7.5** dargestellt. Des Weiteren hat jedes TPDO eine Beschreibung der Kommunikationsparameter, also Übertragungstyp, COB-ID und gegebenenfalls Event Timer. Die Kommunikationsparameter für TPDO 1 bis 4 sind im OD an Index 0x1800 bis 0x1803 dokumentiert.

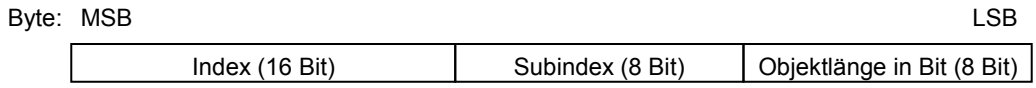

Abbildung 7.5: Grundstruktur eines PDO Mappingeintrags

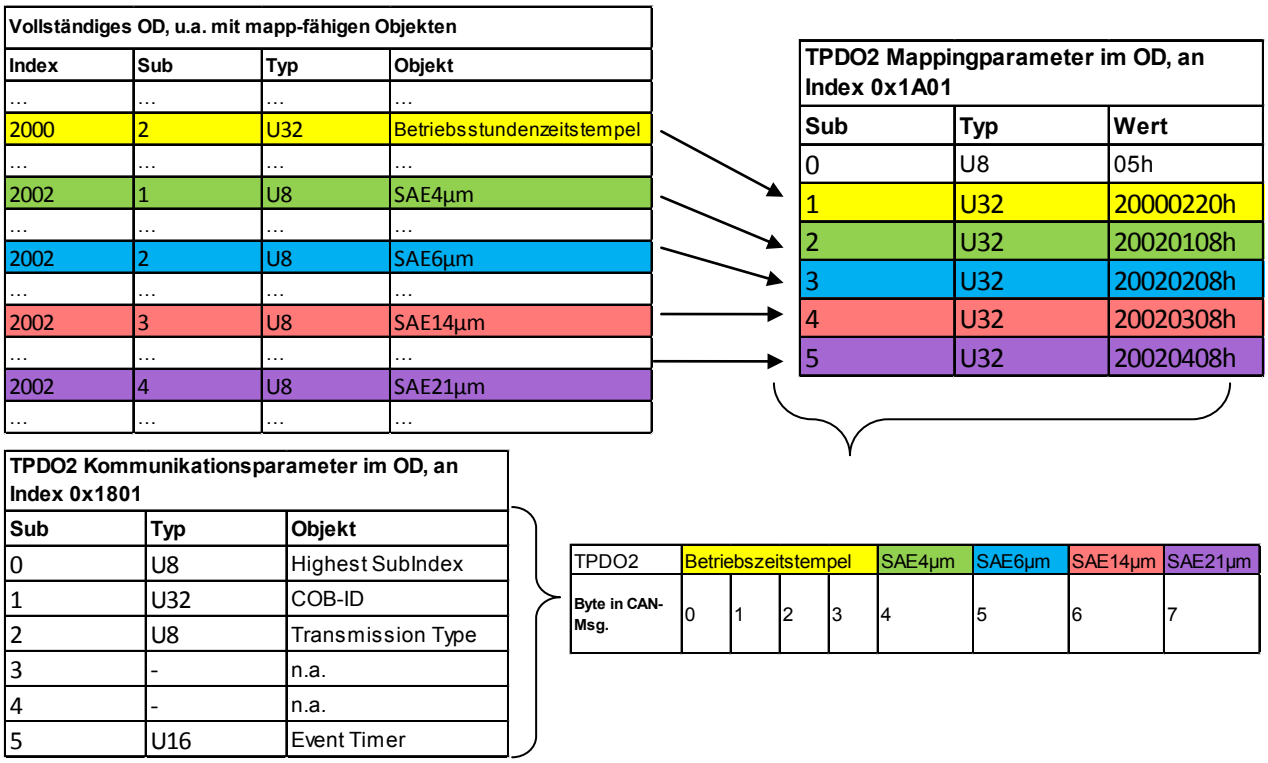

Abbildung 7.6:Prinzip des Mappings von mehreren OD-Objekten in ein TPDO

Der Sensor unterstützt bestimmte Typen des TPDO (vgl. **Tabelle 7.10**), die für die jeweiligen Kommunikationsparameter der TPDOs eingetragen werden können (vgl. Abbildung 7.6).

Durch Sensor unterstützte TPDO Typen					
Typ	**unterstützt**	**zyklisch**	**nicht zyklisch**	**synchron**	**asynchron**
0	ja		X	X	
1-240	ja	X		X	
241-253	nein				
254	ja				X
255	ja				X

Tabelle 7.10: Beschreibung der TPDO Typen

7.2.6 „CANopen Object Dictionary“ detalliert

7. CAN

Das vollständige Objektverzeichnis des Sensors ist in Tabelle 7.11 aufgeführt. Die hier möglichen Einstellungen entsprechen, bis auf wenige Ausnahmen, dem CANopen Standard wie dieser in DS-301 beschrieben ist.

Passende EDS-Dateien für die Sensoren sind verfügbar.

Communication Profile Area						
Idx	**Sldx**	**Name**	**Type**	**Attr.**	**Default**	**Notes**
1000h	0	device type	unsigned 32	ro	194h	Sensor, see DS404
1001h	0	errorregister	unsigned 8	ro	00h	mandatory, see DS301
1017h	0	producerheartbeat time	unsigned 16	rw	1388h	heartbit time in ms, range: 0..65535
1018h		identityobject	record	ro		
	0	Numberofentries	unsigned 8	ro	04h	largestsubindex
	1	Vendor ID	unsigned 32	ro	000000E6h	Argo Hytos
	2	Product Code	unsigned 32	ro	00004F4Ch	OpCom II
	3	Revision Number	unsigned 32	ro	1000	Device dependant
	4	Serial Number	unsigned 32	ro		Device dependant
1800h		Transmit PDO1 Parameter	record			
	0	Numberofentries	unsigned 8	ro	05h	largestsubindex
	1	COB-ID	unsigned 32	rw	180h+NodeID	COB-ID used by PDO, range: 181h..1FFh, can be changed while not operational
	2	transmission type	unsigned 8	rw	FFh	cyclic+synchronous, asynchronous values: 1-240, 254, 255
	5	eventtimer	unsigned 16	rw	1F4h	event timer in ms for asynchronous TPDO1
1801h		Transmit PDO2 Parameter	record			
	0	Numberofentries	unsigned 8	ro	05h	largestsubindex

	1	COB-ID	unsigned 32	rw	280h+NodeID	COB-ID used by PDO, range: 281h..2FFh, can be changed while not operational
	2	transmission type	unsigned 8	rw	FFh	cyclic+synchronous, asynchronous values: 1-240, 254, 255
	5	eventtimer	unsigned 16	rw	1F4h	event timer in ms for asynchronous TPDO2
1802h		Transmit PDO3 Parameter	record			
	0	Numberofentries	unsigned 8	ro	05h	largestsubindex
	1	COB-ID	unsigned 32	rw	380h+NodeID	COB-ID used by PDO, range: 381h..3FFh, can be changed while not operational
	2	transmission type	unsigned 8	rw	FFh	cyclic+synchronous, asynchronous values: 1-240, 254, 255
	5	eventtimer	unsigned 16	rw	1F4h	event timer in ms for asynchronous TPDO3
1A00h		TPDO1 Mapping Parameter	record			
	0	Numberofentries	unsigned 8	ro	05h	largestsubindex
	1	PDO Mapping for 1st app obj. to be mapped	unsigned 32	co	20000220h	Betriebsstundenzeitstempel der Messung, 4 Byte
	2	PDO Mapping for 2nd app obj. to be mapped	unsigned 32	co	20010108h	ISO4µm, 1 Byte im 2001h, sub 01
	3	PDO Mapping for 3rd app obj. to be mapped	unsigned 32	co	20010208h	ISO6µm, 1 Byte im 2001h, sub 02
	4	PDO Mapping for 4th app obj. to be mapped	unsigned 33	co	20010308h	ISO14µm, 1 Byte im 2001h, sub 03
	5	PDO Mapping for 5th app obj. to be mapped	unsigned 32	co	20010408h	ISO21µm, 1 Byte im 2001h, sub 04
1A01h		TPDO2 Mapping Parameter	record			
	0	Numberofentries	unsigned 8	ro	05h	largestsubindex
	1	PDO Mapping for 1st app obj. to be mapped	unsigned 32	co	20000220h	Betriebsstundenzeitstempel der Messung, 4 Byte
	2	PDO Mapping for 2nd app obj. to be mapped	unsigned 32	co	20020108h	SAE4µm, 1 Byte im 2002h, sub 01
	3	PDO Mapping for 3rd app obj. to be mapped	unsigned 32	co	20020208h	SAE6µm, 1 Byte im 2002h, sub 02
	4	PDO Mapping for 4th app obj. to be mapped	unsigned 33	co	20020308h	SAE14µm, 1 Byte im 2002h, sub 03
	5	PDO Mapping for 5th app obj. to be mapped	unsigned 32	co	20020408h	SAE21µm, 1 Byte im 2002h, sub 04

7. CAN

1A02 h		TPDO3 Mapping Parameter	record			
	0	Numberofentries	unsigned 8	ro	05h	largestsubindex
	1	PDO Mapping for 1st app obj. to be mapped	unsigned 32	co	20000120h	Betriebsstundenzähler, 4 Byte
	2	PDO Mapping for 2nd app obj. to be mapped	unsigned 32	co	20030108h	Öl-Zustandsbitsbits, 1 Byte
	3	PDO Mapping for 3rd app obj. to be mapped	unsigned 32	co	20030708h	Messbits, 1 Byte
	4	PDO Mapping for 4th app obj. to be mapped	unsigned 32	co	20030808h	Sensorstatusbits, 1 Byte
	5	PDO Mapping for 5th app obj. to be mapped	unsigned 32	co	20040008h	Temperatur 1 Byte
2000 h		Time related parameters of the sensor	record			
	0	Numberofentries	unsigned 8	ro	02h	largestsubindex
	1	Betriebsstundenzähler	unsigned 32	ro		Sensor up time in seconds
	2	Betriebsstundenzeitstempel der Messung	unsigned 32	ro		Zeitstempel der letzten Messung
2001 h		ISO Messung	record			
	0	Numberofentries	unsigned 8	ro	04h	largestsubindex
	1	ISO4µm	unsigned 8	ro		
	2	ISO6µm	unsigned 8	ro		
	3	ISO14µm	unsigned 8	ro		
	4	ISO21µm	unsigned 8	ro		
2002 h		SAE Messung	record			
	0	Numberofentries	unsigned 8	ro	04h	largestsubindex
	1	SAE4µm	unsigned 8	ro		
	2	SAE6µm	unsigned 8	ro		
	3	SAE14µm	unsigned 8	ro		
	4	SAE21µm	unsigned 8	ro		
2003 h		Condition Monitoring Bitfield	array			
	0	Numberofentries	unsigned 8	ro	08h	largestsubindex
	1	Oilspecificbits	unsigned 8	ro		0 Konzentrationsgrenze überschritten 1 Durchfluss hoch 2 Durchfluss niedrig

	2	reserved	unsigned 8	ro		
	3	reserved	unsigned 8	ro		
	4	reserved	unsigned 8	ro		
	5	reserved	unsigned 8	ro		
	6	reserved	unsigned 8	ro		
	7	Measurement info	unsigned 8	ro		0 Messung läuft 1 Messmodus auto 2 Messmodus I/O 3 Messmodus manuell 4 Alarmmodus Filter / Standard
	8	Sensor alarm	unsigned 8	ro		0 Laserstrom hoch 1 Laserstrom niedrig 2 Photospannnung hoch 3 Photospannung niedrig 4 Temperatur hoch 5 Temperatur niedrig
2004 h		Sensor Temperature	signed 8	ro		Oiltemperature in °C
2005 h		Flow index	unsigned 16	ro		Flow index (0..500)
2020 h		Commando	unsigned 8	wo		1 = Start einer Messung 2 = Stop einer Messung
2030 h		Measurement relatedsettings	record			
	0	Numberofentries	unsigned 8	ro	08h	largestsubindex
	1	Measurement Time	unsigned 32	rw		Measurement Time in s
	2	Hold Time	unsigned 32	rw		Time betweenMeasurements
	3	Operation Mode	unsigned 16	rw		0 = Time Control 1 = Digital I/O 2 = Button 3 = Automatic
	4	Historydisable	unsigned 16	rw	0	0 = Historyenabled 1 = Historydisabled
2031 h		Startup Settings	record			
	0	Numberofentries	unsigned 8	ro	01h	largestsubindex
	1	Startmode	unsigned 16	rw	0h	0 = Network with NMT Master (Init→PreOp→Start_Remote_Node→ Operational) >0 = Network without NMT Master (Init→ Operational)
2100 h		Readmem control functions	record			

	0	Numberofentries	unsigned 8	ro	04h	largestsubindex
	1	Size ofhistorymemory	unsigned 32	ro	devicedependand	size of memory in datasets
	2	Usedhistorymem	unsigned 32	ro		used datasets within memory (corresponds internaly to write pointer)
	3	Reading pointer, dataset	unsigned 32	ro		autoincrementing read pointer to a dataset for history memory reading; can be between 0 and current write pointer
	4	Clear historymemory	unsigned 16	wo		1 = clearmemory
2101h	0	Readmem Initiate segmented SDO data Upload	unsigned 16	ro		Appropriate Pointer has to be set (with 2100sub3) before start reading, Size of the record will be sent back on reading

Tabelle 7.11: Kommunikationsbezogenes Objektverzeichnis

8 Fehlerbehebung

Fehler: *Kein Kommunikation (ComPort) oder Stromausgänge < 4mA*	
Ursache	***Maßnahme***
▪ Kabel ist nicht korrekt angeschlossen	▪ Überprüfen Sie bitte zunächst den korrekten elektrischen Anschluss des Sensors bzw. des Daten- und Stromkabels. Berücksichtigen Sie bitte die vorgeschriebene Anschlussbelegung.
▪ Betriebsspannung liegt außerhalb des vorgeschriebenen Bereichs	▪ Bitte betreiben Sie den Sensor im Bereich zwischen 9 V und 33 V DC.
Fehler: *Keine serielle Kommunikation*	
Ursache	***Maßnahme***
▪ Schnittstellenkonfiguration ist fehlerhaft	▪ Überprüfen und korrigieren Sie gegebenenfalls die Einstellungen der Schnittstellen-Parameter (9600, 8,1, N, N) Testen Sie die Kommunikation mit Hilfe eines Terminal-Programms ggf. unter Verwendung eines Schnittstellenprüfers.
▪ Falscher Kommunikationsport gewählt	▪ Überprüfen und korrigieren Sie die Wahl des Kommunikationsports (z.B. COM1)
▪ Fehlerhafte Schreibweise der Sensorbefehle	▪ Überprüfen Sie die Schreibweise der Sensorbefehle. Achten Sie insbesondere auf Groß- und Kleinschreibung
▪ NumLock-Taste ist deaktiviert	▪ Aktivieren Sie die NumLock-Taste
▪ Feststelltaste ist eingerastet (Großschreibung)	▪ Deaktivieren Sie die Großschreibung
▪ Kabel falsch oder defekt	▪ Verwenden Sie möglichst OLAER Datenkabel
Fehler: *Auf allen Größen werden identische Reinheiten angezeigt*	

8. Fehlerbehebung

Ursache	***Maßnahme***
▪ Luft im Öl	▪ OPCom II druckseitig anschließen ▪ Entfernung von der Pumpe erhöhen
Fehler: *Fehlmessung der analogen Stromausgänge*	
Ursache	***Maßnahme***
▪ Es wird ein falscher Parameter ausgegeben.	▪ Korrigieren Sie die Zuordnung der Messwerte zu den Stromausgängen.
Fehler: *Laserstrom Hoch / Photospannung niedrig*	
▪ Luft im Öl	▪ OPCom II druckseitig anschließen ▪ Entfernung von der Pumpe erhöhen
▪ Zelle verschmutzt	▪ Reinigen Sie den OPCom II mit sauberem Öl oder Lösungsmittel wie z.B. Isopropanol.

Tabelle 8.1: Fehlerbeschreibung

9 Zubehör

Netzteil

Beschreibung: 24V Netzteil zum Anschluss an konfektioniertes Datenkabel SCSO 100-5030

Bestellnummer: SCSO 100-5080

Leitungsdose

Beschreibung: 8-polige, geschirmte M12-Kabeldose geeignet für Kabeldurchmesser 6…8 mm, Schutzklasse IP67, Temperaturbereich -40°C … 85°C

Bestellnummer: SCSO 100-5010

Konfektioniertes Datenkabel mit offenen Enden

Beschreibung: Geschirmtes Sensorkabel, Schutzklasse IP67, Temperaturbereich -20°C...85°C, ölfest, Seite 1 - Sensorstecker umspritzt, Seite 2 - 8 Einzellitze

Bestellnummer: SCSO 100-5020

Konfektioniertes Datenkabel für Rechneranschluss / D-Sub-Stecker 9pol

Beschreibung: Geschirmtes Sensorkabel, Schutzklasse IP67, Temperaturbereich -20°C…85°C, ölfest, Seite 1 - Sensorstecker umspritzt, Seite 2 – 9-pol. D-Sub-Buchse / Hohlstecker für Spannungsversorgung (Netzteil muss separat bestellt werden!)

Bestellnummer: SCSO 100-5030

USB/Seriell-Adapter

Beschreibung: Adapter für die Umsetzung serieller RS232-Schnittstelle auf „Universal Serial Bus“ (USB). Über USB ist es möglich mehrere Sensoren gleichzeitig anzusprechen.

Bestellnummer: SCSO 100-5040

11 Change Log

30.10.2012: Korrektur der Formel (3-2), Konformitätserklärung überarbeitet - RCK

13.11.2012: Version auf 1.04.12 hochgesetzt um mit der englischen Version synchron zu sein, und Objektverzeichnis für CANopen aktualisiert – HD

15.11.2012: Korrektur der Formel (3-2) – KN

30.11.2012: Punkt 2.2 Kalibrierung eingefügt – KN

20.03.2013:
- Kapitel ERC Berechnung hinzugefügt - KN
- Kapitel Terminalprogramm hinzugefügt - KN
- Kapitel TCP/IP Verbindung hinzugefügt - KN
- Kapitel CE Konformität ersetzt - KN
- Kapitel CAN erweitert - KN
- Kapitel Technische Daten korrigiert - KN
- Kapitel Befehlslisten getrennt und korrigiert - KN
- allgemeine Überarbeitung - KN