Technical Manual

Kübler

Sendix® absolut

Absolute Singleturn Encoder

Series M3658, M3678

Safety-Lock™ | High rotational speed | -40° +85° Temperature | IP High IP | High shaft load capacity | Shock/vibration resistant | Short-circuit proof | Reverse polarity protection

CANopen certified

CiA200704-301V402/20-0071

pulses for automation

© Fritz Kübler GmbH

Copyright Protection

Liability to modification without notice

As a result of ongoing efforts to improve our products, we reserve the right to make changes at any time to technical information contained in the document to hand.

Warranty Disclaimer

Fritz Kübler GmbH provides no guarantee, neither tacit nor express, in respect of the whole manual (whether this applies to the original German text or to the English translation) and assumes no liability for any damage, neither direct nor indirect, however caused.

Document information

Revised 11-2013

Screen printouts used

CANalyzer® Fa. Vector –Informatik

Fritz Kübler GmbH Schubertstr.47
78054 VS-Schwenningen / Germany
Tel. +49 (0) 7720-3903-0
Fax +49 (0) 7720-21564
E-Mail: info@kuebler.com
Internet: www.kuebler.com

Table of Contents

1 General

CANopen Singleturn Encoder Series M36X8

The CANopen encoder of Series 36X8 support the latest CANopen communication profile according **DS 301 V4.02** . In addition a device-specific encoder profile **DS 406 V3.1** is implemented.

The following operating modes can be selected: Polled Mode, Cyclic Mode, Sync Model. Moreover, scale factors, preset values, limit switch values and many other additional parameters can be programmed via the CAN-Bus. At Power ON all parameters are loaded from an EEPROM, which had previously been saved in the non-volatile memory to protect them in case of power failure. The following output values may be freely combined as **PDO** (PDO Mapping): **position, speed** as well as the status of the two **limit switches**, **alarms** and **warnings**.

The encoders are available with **a connector** or a **cable connection** - all changes to the device address and baud rate are software controlled.
One LED located on the back indicate the operating or fault status of the CAN bus, as well as the status of an internal diagnostic. CANopen encoders are available in blind hollow shaft and solid shaft versions, and are ideal for use in harsh industrial environments thanks to their IP 65 protection rating.

The CANopen Communication Profile DS 301 V4.02

CANopen represents a unified user interface and thus allows for a simplified system structure with a wide variety of devices. CANopen is optimized for the fast exchange of data in real-time systems and possesses a number of different device profile that have been standardized. The CAN in Automation (CiA) manufacturers and users group is responsible for creating and standardization of the relevant profiles.

CANopen offers

- user-friendly access to all device parameters.
- auto-configuration of the network and of the devices
- device synchronization within the network
- cyclic and event-driven process data exchange
- simultaneous read and write of data

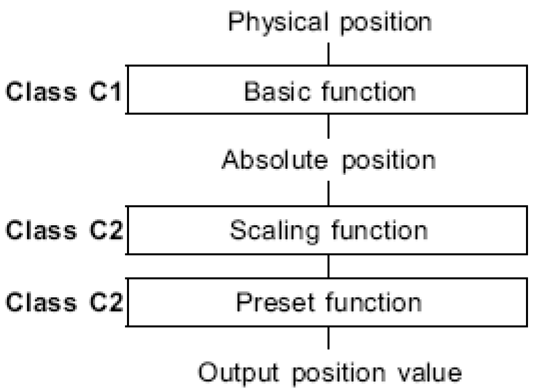
CANopen uses four communication objects (COB) with different properties

- Process Data Objects (PDO) for real-time data,
- Service Data Objects (SDO) for transmitting parameters and programs,
- Network Management (NMT, Life-Guarding, Heartbeat)
- Predefined Objects (for Synchronisation, Time-Stamp, Emergency)

All device parameters are filed in an **Object Dictionary**. This Object Dictionary contains the description, data type and structure of the parameters, as well as the address (Index).
The dictionary is divided into a communications profile section, a section covering the device profile as well as a section specific to the manufacturer.

Encoder Device Profile DS 406 V3.1

This profile describes a **vendor-independent** mandatory definition of the interface with regard to encoders. It is laid down in the profile, which CANopen functions are to be used as well as how they are to be used. This standard thus makes possible an open vendor-independent bus system.
The device profile is broken down into two Object classes:



• **Class C1** describes all the basic functions that the encoder must contain

• **Class C2** contains numerous extended functions, which must either be supported by encoders of this class (Mandatory) or which are optional. Class 2 devices thus contain all C1 and C2 mandatory functions, as well as additional optional functions dependent on the manufacturer. An address range is also defined in the profile to which the manufacturer's own special functions can be assigned.

Objectives of LSS

CiA DSP 305 CANopen *Layer Setting Service and Protocol (LSS)* services and protocols were created to enable the following parameters to be read and changed through the network:

- The CANopen Node ID
- The CAN baud rate
- The LSS address

This increases the "plug–and-play" capabilities of devices on CANopen networks as preconfiguration of the network is less restrictive. The LSS Master is responsible for configuring these parameters on one or more LSS Slaves on a CANopen network.

Data transmission

With CANopen data are transferred via two different communication types (COB=Communication Object) with different properties:

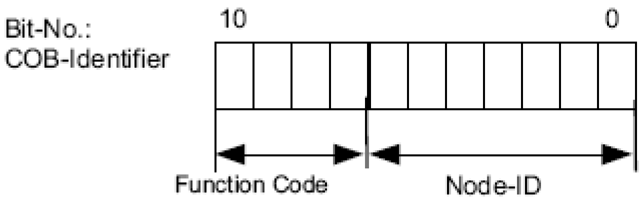
- **Process Data Objects (PDO – real-time capable)**
- **Service Data Objects (SDO)**

The Process Data Objects **(PDO)** provide high-speed exchange of real-time data (e.g. encoder position, speed, comparative position status) with a maximum length of 8 byte. These data are transmitted with a high priority (low COB-Identifier). PDOs are broadcast messages and provide their real-time data simultaneously to all desired receivers. PDOs can be mapped, i.e. 4 byte of position and 2 byte of speed can be combined in one 8 byte data word.

The Service Data Objects **(SDO)** form the communication channel for the transfer of device parameters (e.g. encoder resolution programming). As these parameters are transmitted acyclically (e.g. only once during boot-up of the network), the SDO objects have a low priority (high COB-Identifier).

Objects and Function Code in the Predefined Connection Set

For easier management of the Identifiers CANopen uses the "Predefined Master/Slave Connection Set", where all identifiers are defined with standard values in the object dictionary. These identifiers can however be changed and customized via SDO access.



The 11-bit Identifier is made up of a **4-bit function code** and a **7-bit node-ID number**.

The higher the value of the COB-Identifier, the lower is its priority!

Broadcast (network-wide) Objects

object	function code (binary)	resulting COB-ID	Communication Parameters at Index
NMT	0000	0	-
SYNC	0001	128 (80h)	1005h, 1006h, 1007h
TIME STAMP	0010	256 (100h)	1012h, 1013h

Peer-To Peer (device-to-device) Objects

object	function code (binary)	Resulting COB-IDs	Communication Parameters at Index
EMERGENCY	0001	129 (81h) – 255 (FFh)	1014h, 1015h
PDO1 (tx)	0011	385 (181h) – 511 (1FFh)	1800h
PDO1 (rx)	0100	513 (201h) – 639 (27Fh)	1400h
PDO2 (tx)	0101	641 (281h) – 767 (2FFh)	1801h
PDO2 (rx)	0110	769 (301h) – 895 (37Fh)	1401h
PDO3 (tx)	0111	897 (381h) – 1023 (3FFh)	1802h
PDO3 (rx)	1000	1025 (401h) – 1151 (47Fh)	1402h
PDO4 (tx)	1001	1153 (481h) – 1279 (4FFh)	1803h
PDO4 (rx)	1010	1281 (501h) – 1407 (57Fh)	1403h
SDO (tx)	1011	1409 (581h) – 1535 (5FFh)	1200h
SDO (rx)	1100	1537 (601h) – 1663 (67Fh)	1200h
NMT Error Control	1110	1793 (701h) – 1919 (77Fh)	1016h, 1017h

Restricted, reserved Objects

COB-ID	used by object
0 (000h)	NMT
1 (001h)	reserved
257 (101h) – 384 (180h)	reserved
1409 (581h) – 1535 (5FFh)	default SDO (tx)
1537 (601h) – 1663 (67Fh)	default SDO (rx)
1760 (6E0h)	reserved
1793 (701h) – 1919 (77Fh)	NMT Error Control
2020 (780h) – 2047 (7FFh)	reserved

2 Transmission of Process Data

Within the CANopen encoder **three PDO services PDO1 (tx) , PDO2 (tx) and PDO3(tx)** are available. A PDO transmission can be triggered by a variety of events (see Object Dictionary Index 1800h):

- **asynchronously** (event driven) by an internal cyclic device timer or by a change in the process value of the sensor data
- **synchronously** as a response to a SYNC telegram; (a SYNC command will cause all CANopen nodes to store their values synchronously, after which they are transferred in succession to the bus according to their set priority)
- **as a response** to an RTR-Telegram (per Remote Frame=recessive RTR-bit, exactly that message with the communicated ID will be requested)

Example : Default Mapping of the **PDO messages** have the following structure

Mapping	TPDO1 1800h	TPDO2 1801h	TPDO3 1802h
Mapping object	1A00h	1A01h	1A02h
Content	0x60040020	0x60040020	0x60300110
Object	6004h	6004h	6030h
Subindex	00	00	01
Length	20h(32 Bit)	20h(32 Bit)	10h(16 Bit)
	Asynchronous	Synchronous	Asynchronous

Transmit PDO 1 (1800h) Position asynchronous

Default COB-ID is 180 + Node number: e.g. 180h + 3Fh = 1BFh

Message	Byte 0	Byte 1	Byte2	Byte 3
1BF	Position LSB	Position MSB	00	00

Position values are in a range of 0 – 3FFFh oder 0 – 16383 (decimal).

Transmit PDO2 (1802h) Position synchronous (SYNC-Mode)

Default COB-ID ist 280 + Node number: e.g. 280h + 3Fh = 2BFh

Message	Byte 0	Byte 1	Byte2	Byte 3
2BF	Position LSB	Position MSB	00	00

Position values are in a range of 0 – 3FFFh oder 0 – 16383 (decimal).

Transmit PDO3 (1801h) Velocity asynchronous

Default COB-ID ist 380 + Node number: e.g. 380h + 3Fh = 3BFh

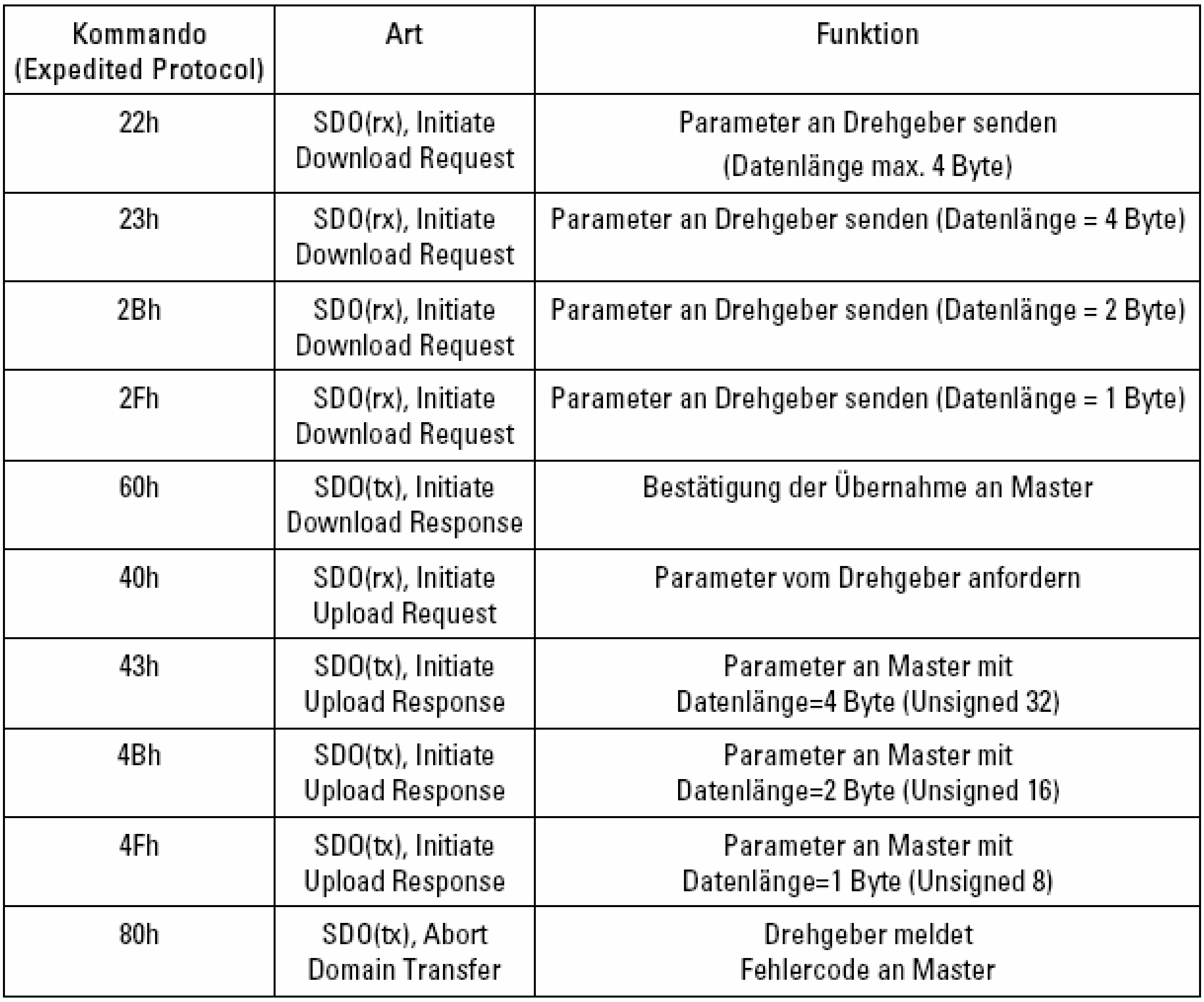
Message	Byte 0	Byte 1
3BF	Velocity LSB	Velocity MSB

The value of the velocity is signed and is shifted in a range of (plus) 0 – 1A00h resp (minus) 0 – E600h

3 Transmission of Service Data

SDO-COB-ID
The following identifiers are available as standard for the SDO services:

SDO (tx) (Encoder→Master): 580h (1408) + node number
SDO (rx) (Master→Encoder): 600h (1536) + node number
The SDO identifiers cannot be changed!

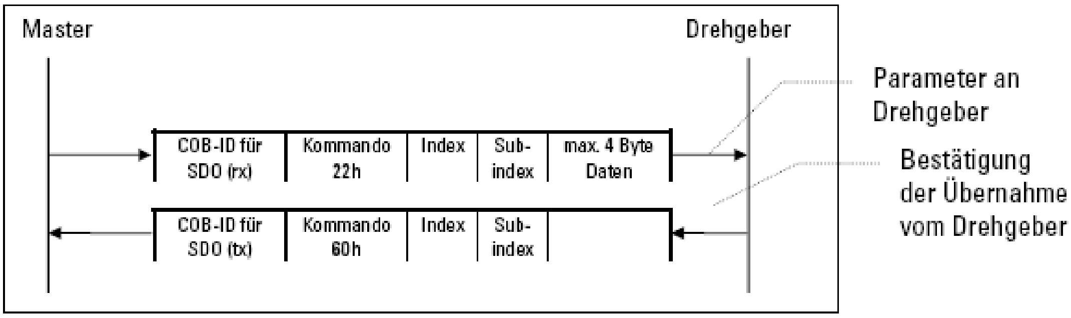


Kommando (Expedited Protocol)	Art	Funktion
22h	SDO(rx), Initiate Download Request	Parameter an Drehgeber senden (Datenlänge max. 4 Byte)
23h	SDO(rx), Initiate Download Request	Parameter an Drehgeber senden (Datenlänge = 4 Byte)
2Bh	SDO(rx), Initiate Download Request	Parameter an Drehgeber senden (Datenlänge = 2 Byte)
2Fh	SDO(rx), Initiate Download Request	Parameter an Drehgeber senden (Datenlänge = 1 Byte)
60h	SDO(tx), Initiate Download Response	Bestätigung der Übernahme an Master
40h	SDO(rx), Initiate Upload Request	Parameter vom Drehgeber anfordern
43h	SDO(tx), Initiate Upload Response	Parameter an Master mit Datenlänge=4 Byte (Unsigned 32)
4Bh	SDO(tx), Initiate Upload Response	Parameter an Master mit Datenlänge=2 Byte (Unsigned 16)
4Fh	SDO(tx), Initiate Upload Response	Parameter an Master mit Datenlänge=1 Byte (Unsigned 8)
80h	SDO(tx), Abort Domain Transfer	Drehgeber meldet Fehlercode an Master

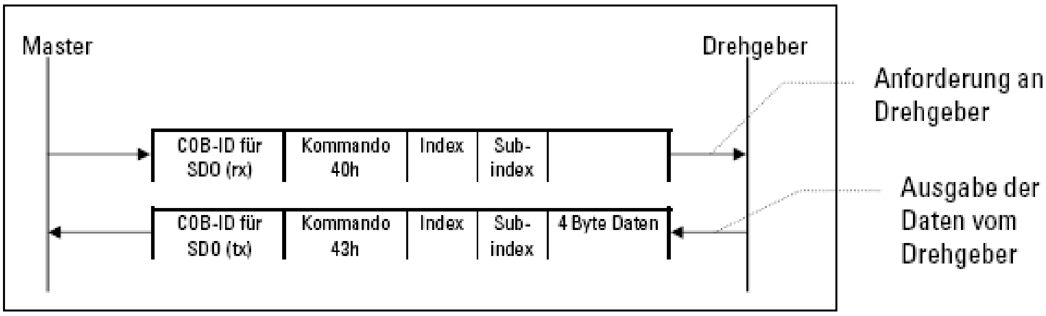
If an error occurs, then an error message (command 80h) will replace the normal confirmation (Response). The error message covers not only the communication protocol error but also the object dictionary access error (e.g. wrong index, attempted write to Read-Only Object, incorrect data length etc).

The error codes are described in the CANopen Profile (DS 301) or in the Device Profile (DSP 406).

Example: Transmission of Service Data to and from the encoder



Master überträgt Parameter an Drehgeber



Master fordert Parameter vom Drehgeber an

LSS Hardware Restrictions (LSS Address)

All LSS Slaves must support valid Object Dictionary entries for Identity object [1018h] which has 32 bits for each part of the LSS Address:

- Vendor-ID (numerical number)
- Product-Code (numerical number)
- Revision-Number (major an minor revision as numerical number)
- Serial-Number (numerical number)
- **LSS-Master CAN-ID 2021**
- **LSS-Slave CAN-ID 2020**

A Product-Code, Revision-Number and a Serial-Number are assigned by the device supplier. The LSS address which must be absolutely unique. No other LSS slave may have the same LSS address.

LSS Operating Restrictions

To function properly the following restrictions apply:

- All devices on a CANopen network must support LSS.
- There can be only one LSS Master.
- All nodes are required to start-up with the same initial baud rate.
- LSS communication can take place during any NMT state such as “stopped” or "pre-operational".

LSS Configuration and the Operation Modes

Configuration Mode

- When an LSS Slave is in this mode, it actively listens for and processes configuration commands from the LSS Master.
- Some configuration commands configure only one LSS Slave at the time (for example, to change of CANopen node ID)
- Some configuration commands configure multiple or all LSS Slave nodes (for example, to change the baud rate)

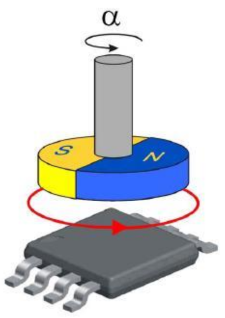
Operation Mode

- A LSS Slave in this mode ignores the configuration commands from the LSS

4 Power supply and CAN-bus connection

Power supply

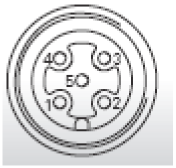
Sensor:	**Magnetic Hall -Sensor**
	14 Bit Resolution /9 Bit Accuracy
Power supply:	**10 … 30 VDC**
Current consumption:	typ. **22mA** at 24 VDC
	max. 49 mA at 10 VDC
Reverse polarity protection:	yes
CAN Transceiver:	82C251 / short cuircuit tested
Galvanic Isolation:	no
CANopen Communikation	**DS 301 V4.02**
CANopen Encoder Device Profile	**DS406 V 3.1**



Can-Cable terminal

Short name	Description	Cable color
CG	**CAN Ground**	**gray**
CL	**CAN_Low (-)**	**yellow**
CH	**CAN_High (+)**	**green**
0V	**0Volt power**	**white**
+V	**+UB power**	**brown**

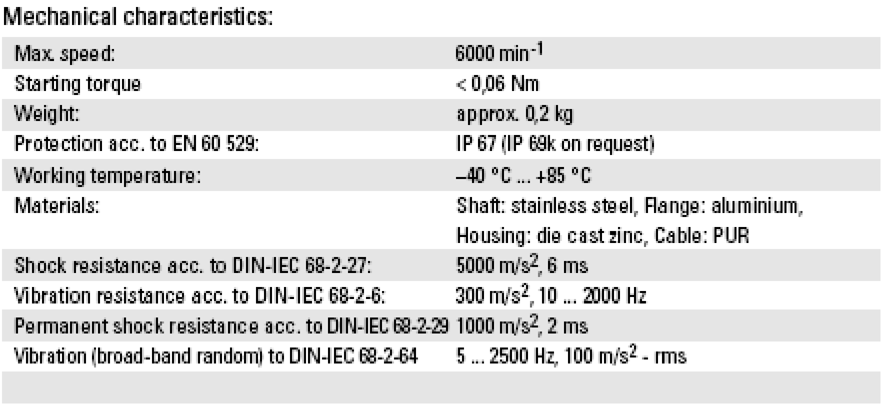
CAN - M12 Connector



Short name	Description	PIN Nr.	Color
CG	**CAN Ground**	**1**	**GY**
CL	**CAN_Low (-)**	**5**	**YE**
CH	**CAN_High (+)**	**4**	**GN**
0V	**0 Volt power**	**3**	**WH**
+V	**+UB power**	**2**	**BN**

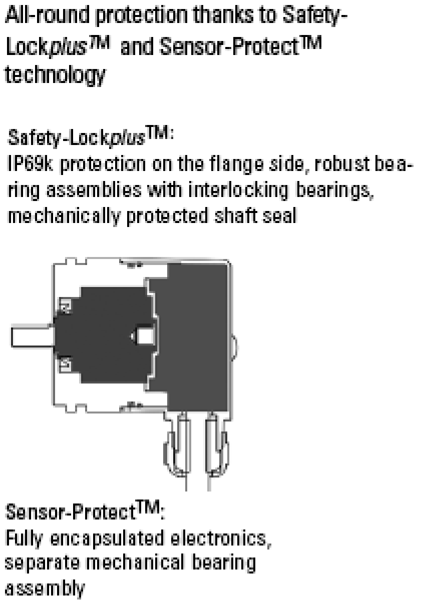
Terminal M12-Connector

Mechanical characteristics



Mechanical characteristics:

Max. speed:	6000 min^{-1}
Starting torque	< 0,06 Nm
Weight:	approx. 0,2 kg
Protection acc. to EN 60 529:	IP 67 (IP 69k on request)
Working temperature:	–40 °C ... +85 °C
Materials:	Shaft: stainless steel, Flange: aluminium, Housing: die cast zinc, Cable: PUR
Shock resistance acc. to DIN-IEC 68-2-27:	5000 m/s^2, 6 ms
Vibration resistance acc. to DIN-IEC 68-2-6:	300 m/s^2, 10 ... 2000 Hz
Permanent shock resistance acc. to DIN-IEC 68-2-29	1000 m/s^2, 2 ms
Vibration (broad-band random) to DIN-IEC 68-2-64	5 ... 2500 Hz, 100 m/s^2 - rms



All-round protection thanks to Safety-Lock*plus*™ and Sensor-Protect™ technology

Safety-Lock*plus*™:
IP69k protection on the flange side, robust bearing assemblies with interlocking bearings, mechanically protected shaft seal

Sensor-Protect™:
Fully encapsulated electronics, separate mechanical bearing assembly

It is possible to change the baud rate using the **Software** at Object 2100h or one of the appropriate **LSS-Services**. The following baud rates are available to the user:

Factory default: 250 kBit/s (value 5)

Value	Baudrate in KBit/s
0	**10**
1	**20**
2	**50**
3	**100**
4	**125**
5	**250**
6	**500**
8	**1000**

Please note the following when selecting a baud rate

The chosen cycle time (see Object 1800h,Sub-index 5 Event Timer) must be longer than the bus transfer time, to ensure that the PDOs are communicated error-free!

With a baud rate of 10 KBaud: cycle time must be at least 14 ms
With a baud rate of 20 KBaud: cycle time must be at least 10 ms
With a baud rate of 50 KBaud: cycle time must be at least 4 ms

With a cycle time=0 in Event-Mode (i.e. PDO on value change) the baud rate must be at least **125 KBaud.**

Node number

It is possible to change the node number using the **Software** at **Object 2101h** or one of the appropriate **LSS-Services.**
Factory default : 0x3F (63 decimal).

Node number 0 is reserved and must not be used by any node. The resulting node numbers lie in the range **1...7Fh** hexadecimal (1...127 decimal).

The acceptance of a new node number only becomes effective when the encoder is rebooted (Reset/Power-on) or by means of an **NMT Reset Node** command.
All previous adjustments made to the object directory are lost and are setting to the factory default.

CAN-bus Termination

The **CAN-Bus Termination** can changed in **Object 2102h** .
Factory default: 0x01 (Bus termination active)

Once the CAN bus has been looped through, it must be terminated between CAN+ and CAN- at both ends using 120 ohm bus termination resistors.

Save All Bus Parameters

The “save all bus parmeters” can changed in **Object 2105h**..
Factory default: 0x65766173 (“save”)

This object stores all bus parameters (**Object 2100h ,2101h,2102h)** permanently in an EEprom. Using the command "save" (save all Parameters) causes all the parameters to be stored. This process requires ca. 5ms. In order to prevent an inadvertent save, the instruction will only be executed if the string "save" is entered as a codeword into this Sub-Index.

The Parameter **„save“ (hexadezimal 0x65766173)** stores all bus adjustments **as Baudrate, Node number and Termination** permanently.

6 Layer Setting Services (LSS)

Exactly two conditions must be fulfilled for the interconnection of CANopen devices to a network: all devices must use the same Baudrate, and the CANopen Node-IDs must be unique. The condition for the use of the LSS is, in addition to support by the device itself, to establish a 1:1 wiring to the Node. Then the Baudrate and the Node-ID are set in dialog mode. The COB-ID **0x7E5** is used for CAN messages to the device, the device responds to COB-ID **0x7E4**. LSS messages are always a full 8 bytes long. Unused bytes are reserved and should be initialized with 0.

To make contact with a device to be configured, the "**Switch Mode Global**" command is transmitted:

0x04	**0x01**	**reserved**

This command sets the device to LSS configuration mode. Unfortunately, this very service is the only unacknowledged LSS service, to which the device will therefore not respond, even if it has carried it out. The system integrator can therefore only find out with the following command whether the device has reacted.

Next the Node-ID is requested via the "**Inquire Node-ID**" service:

0x5E	**reserved**

If successful, the device responds with:

0x5E	**Node ID**	**reserved**

If there is no response, then either the device **does not support the LSS service** or the Baudrate is not correct. If, namely, the Baudrate when supplied is not known, the above-mentioned communication sequence must be tested with all permissible CANopen Baudrates until the device is found.

The "**Configure Node-ID**" service is used to configure the new Node-ID:

0x11	**Node ID**	**reserved**

The error code is included in the device response:

0x11	**Error code**	**Error extension**	**reserved**

Error code 0 means success; error code 1 means inadmissible Node-ID; the other error codes are reserved. The error extension contains vendor-specific information but is only valid for error code 0xFF.

The Baudrate is configured with the "**Configure Bit Timing Parameters**" service:

0x13	**Bit timing**	**Table entry**	**reserved**

The standardized **CANopen Baudrates** are listed in the following table:

Baudrate table 0x00	
Table index	**Baudrate**
0	1000 kBit/s
1	800 kBit/s
2	500 kBit/s
3	250 kBit/s
4	125 kBit/s
5	reserved
6	50 kBit/s
7	20 kBit/s
8	10 kBit/s

Again the device response is:

0x13	**Error code**	**Error extension**	**reserved**

Error code 0 means success; error code 1 means inadmissible baudrate; the other error codes are reserved. The error extension contains vendor-specific information, but is only valid for error code 0xFF.

Now that the node-ID and the baudrate are configured, these settings should be saved with the "**Store Configuration"** service:

0x17	**reserved**

Whereupon the device acknowledges:

0x17	**Error code**	**Error extension**	**reserved**

Error code 0 means success; error code 1 means that the device does not support saving; error code 2 means that there is a problem with access to the storage medium; the other error codes are reserved. The error extension contains vendor-specific information, but is only valid for error code 0xFF.
Finally, the device is switched back from configuration mode to normal mode via "**Switch Mode Global":**

0x04	**0x00**	**reserved**

After being switched **physically off and on again**, the device now works with the new settings.

7 Default settings on delivery

On delivery the following software parameters have been factory set.

Description	Setting	Software
Baud rate	**250 Kbit/s**	Object 2100h = 05h
Node address	**63**	Object 2101h = 3Fh
Termination	**ON**	Object 2102h = 01h

Index (hex)	Name	Standard value
	Communication parameter	
1005h	COB-ID Sync	80h
100Ch	Guard Time	0
100Dh	Life Time Factor	0
1012h	COB-ID Time stamp	100h
1013h	High Resolution time stamp	0
1017h	Producer heartbeat time	0
1029h	Error Behaviour	0 = Comm Error
		1 = Device specific
		1 = Manufacturer Err.
1800h	**TPDO1** Communication Parameter	
01h	COB-ID	180h + Node number
02h	Transmission Type	255 (asynch)
03h	Inhibit Time	0 [steps in 100µs]
05h	Event timer	0 [steps in ms]
1801h	**TPDO3** Communication Parameter	
01h	COB-ID	280h + Node number
02h	Transmission Type	1 (synch)
03h	Inhibit Time	0 [steps in 100µs]
05h	Event timer	0 [steps in ms]
1802h	**TPDO2** Communication Parameter	
01h	COB-ID	380h + Node number
02h	Transmission Type	255 (asynch)
03h	Inhibit Time	0 [steps in 100µs]
05h	Event timer	0 [steps in ms]
1A00h	TPDO1 Mapping	
01h	1.Mapped Object	0x60040020
1A01h	TPDO3 Mapping	
01h	1.Mapped Object	0x60040020
1A02h	TPDO2 Mapping	
01h	1.Mapped Object	0x60300110

Index (hex)	Name	Standard value
	Encoder Profile	
6000h	Operating Parameter	Scaling off
6001h	Measuring Units per Revolution	16384 (14 Bit)
6002h	Total Measuring Range	16384 (14 Bit)
6003h	Preset value	0
6200h	Cyclic Timer (see TPDO1 Comm.Par)	0
6401h	Work area low limit	0
6402h	Work area high limit	16383
2100h	Baud rate	05h
2101h	Node number	3Fh
2102h	CANbus termination	1 (active)
2105h	Save All Bus Parameters	0x65766173

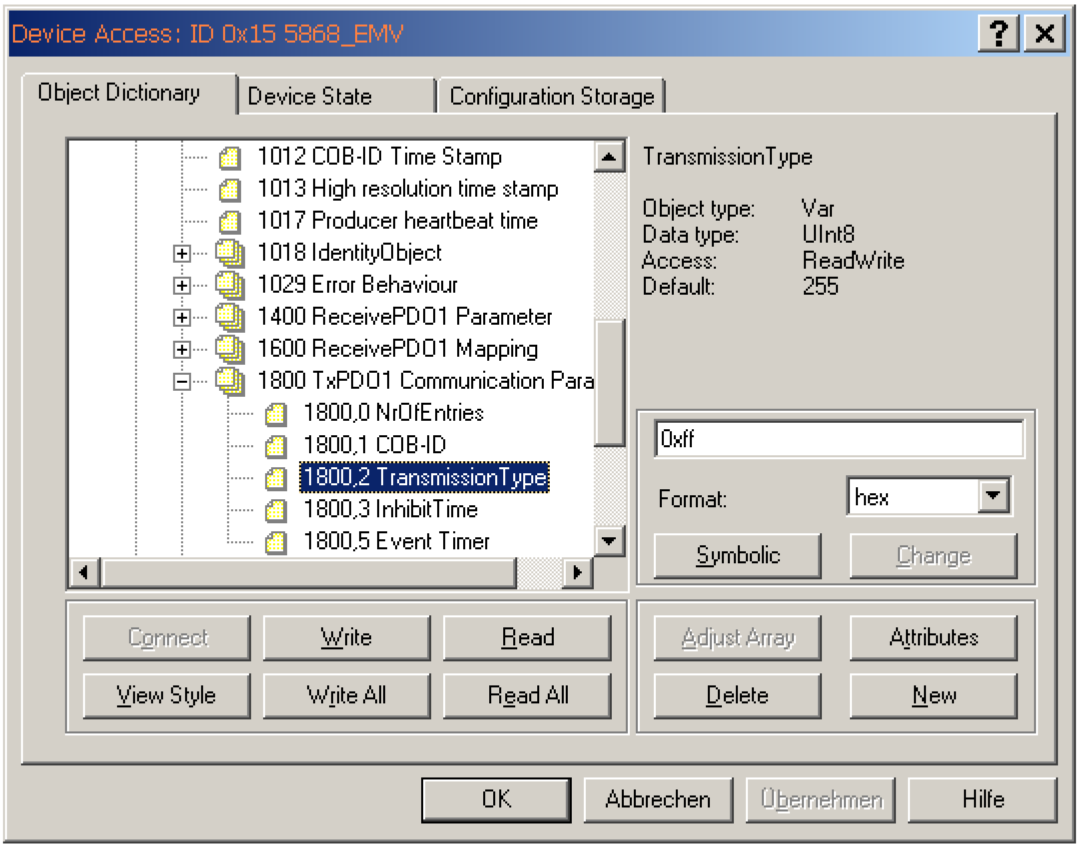
The original Standard Values (default values on delivery) can be reloaded again by means of Object **1011h** (restore parameters).

In order to ensure that parameter changes are saved in the event of power failure, then these must without fail be transferred to the EEPROM by means of Object **1010h** (store parameters).
This will cause all data already present in the EPROM to be over-written!

If errors have occurred during programming of the objects and if these parameters are then saved in the EEPROM, it will not be possible to address the encoder next time it is switched on (the encoder will send only **Emergency** messages).
This error can be cleared only by means of a general **Reset** of the encoder.

8 Communication Parameters

The COB-ID and the Transmission Type for **PDO1** are defined in the Object Dictionary Index **1800h** .



1012 COB-ID Time Stamp
1013 High resolution time stamp
1017 Producer heartbeat time
1018 IdentityObject
1029 Error Behaviour
1400 ReceivePDO1 Parameter
1600 ReceivePDO1 Mapping
1800 TxPDO1 Communication Para
1800,0 NrOfEntries
1800,1 COB-ID
1800,2 TransmissionType
1800,3 InhibitTime
1800,5 Event Timer

TransmissionType

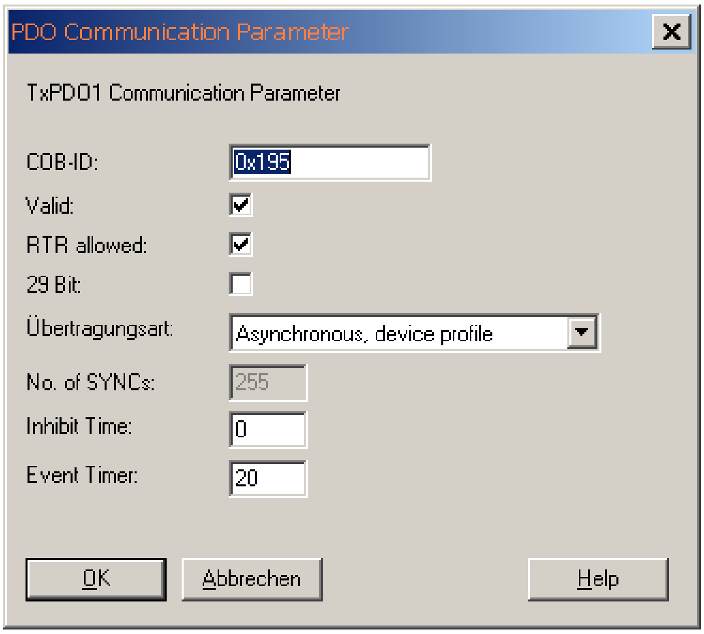
Object type:	Var
Data type:	UInt8
Access:	ReadWrite
Default:	255

0xff

Format: hex

Default-settings:

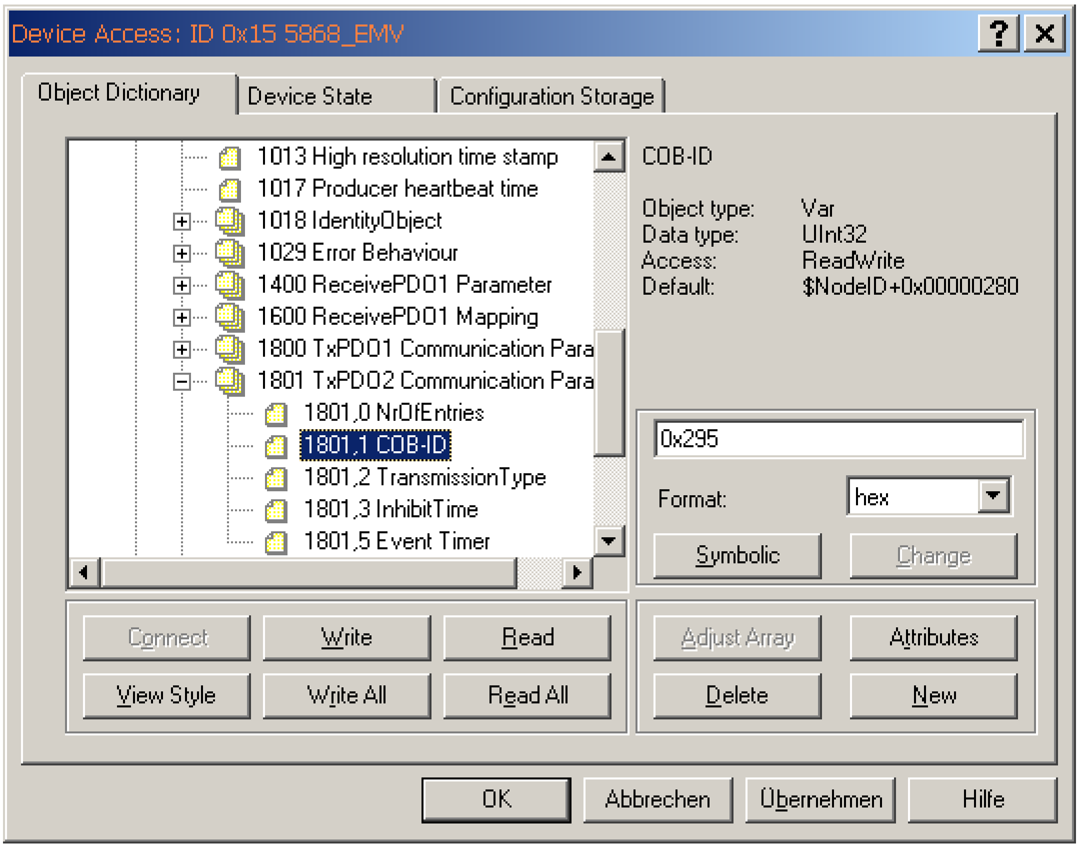
Enabling:	**PDO enabled**	**RTR allowed**
COB-ID:	**180h + node number set (here 11h)**	
Transmission type:	**255 = asynchronous acc. to device profile**	
Event Timer:	**20 ms**	



TxPDO1 Communication Parameter

COB-ID:	0x195
Valid:	☑
RTR allowed:	☑
29 Bit:	□
Übertragungsart:	Asynchronous, device profile
No. of SYNCs:	255
Inhibit Time:	0
Event Timer:	20

The COB-ID and the Transmission Type for **PDO2** are defined in the Object Dictionary Index **1800h** .



Device Access: ID 0x15 5868_EMV

Object Dictionary | Device State | Configuration Storage

- 1013 High resolution time stamp
- 1017 Producer heartbeat time
- 1018 IdentityObject
- 1029 Error Behaviour
- 1400 ReceivePDO1 Parameter
- 1600 ReceivePDO1 Mapping
- 1800 TxPDO1 Communication Para
- 1801 TxPDO2 Communication Para
  - 1801,0 NrOfEntries
  - 1801,1 COB-ID
  - 1801,2 TransmissionType
  - 1801,3 InhibitTime
  - 1801,5 Event Timer

COB-ID

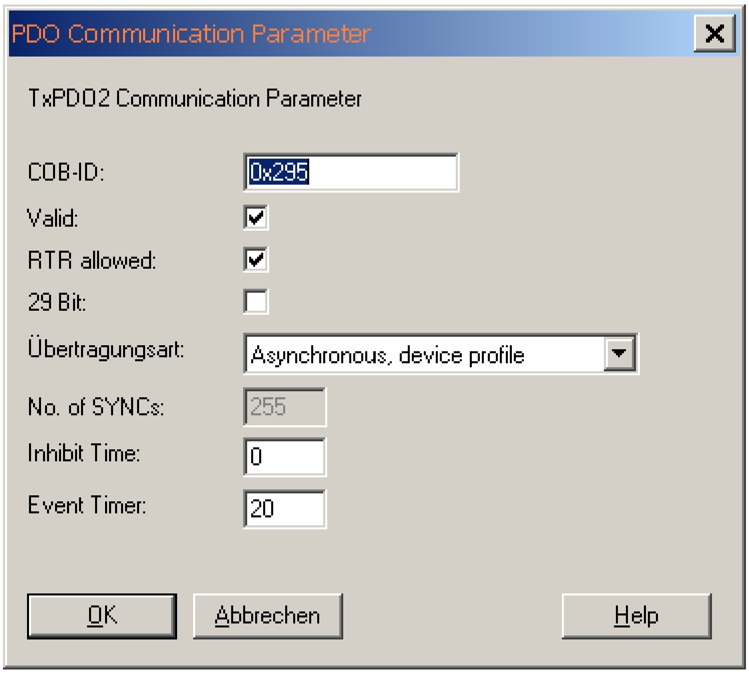
Object type: Var
Data type: UInt32
Access: ReadWrite
Default: $NodeID+0x00000280

0x295

Format: hex

Defaults:

Enabling:	**PDO enabled**	**RTR allowed**
COB-ID:	**280h + node number set (here 11h)**	
Transmission type:	**255 = asynchronous acc.to device profile**	
Event Timer:	**20 ms**	



PDO Communication Parameter

TxPDO2 Communication Parameter

COB-ID: 0x295
Valid: ☑
RTR allowed: ☑
29 Bit: ☐
Übertragungsart: Asynchronous, device profile
No. of SYNCs: 255
Inhibit Time: 0
Event Timer: 20

Definition of the Transmission type of the PDO

transmission type	PDO transmission				
	cyclic	acyclic	synchronous	asynchronous	RTR only
0		X	X		
1-240	X		X		
241-251	- reserved -				
252			X		X
253				X	X
254				X	
255				X	

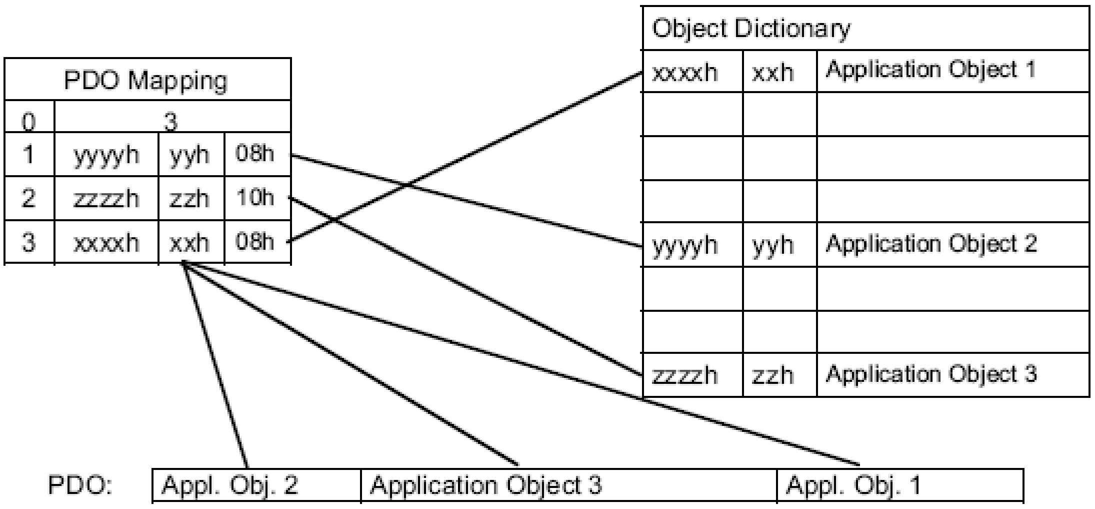
A value between 1 ...240 means that the PDO will be sent **synchronously and cyclically**. The number of the Transmission Type signifies the **quantity of SYNC** pulses that are necessary to forward the PDOs.
The Transmission Types 252 and 253 state that the PDO will only be sent when requested via an RTR.

Type 254 means that the event will be triggered depending on the application (application-specific), whereas Type 255 is dependent on the device (device-specific). Additionally for Numbers 254/255 a time-controlled **EventTimer** can be used. The values for the timer can range from **1ms ... 65535 ms**.

Variable PDO Mapping

Variable Mapping of the various objects means that the user is able to configure the content of the Transmit PDOs dependent on the application.

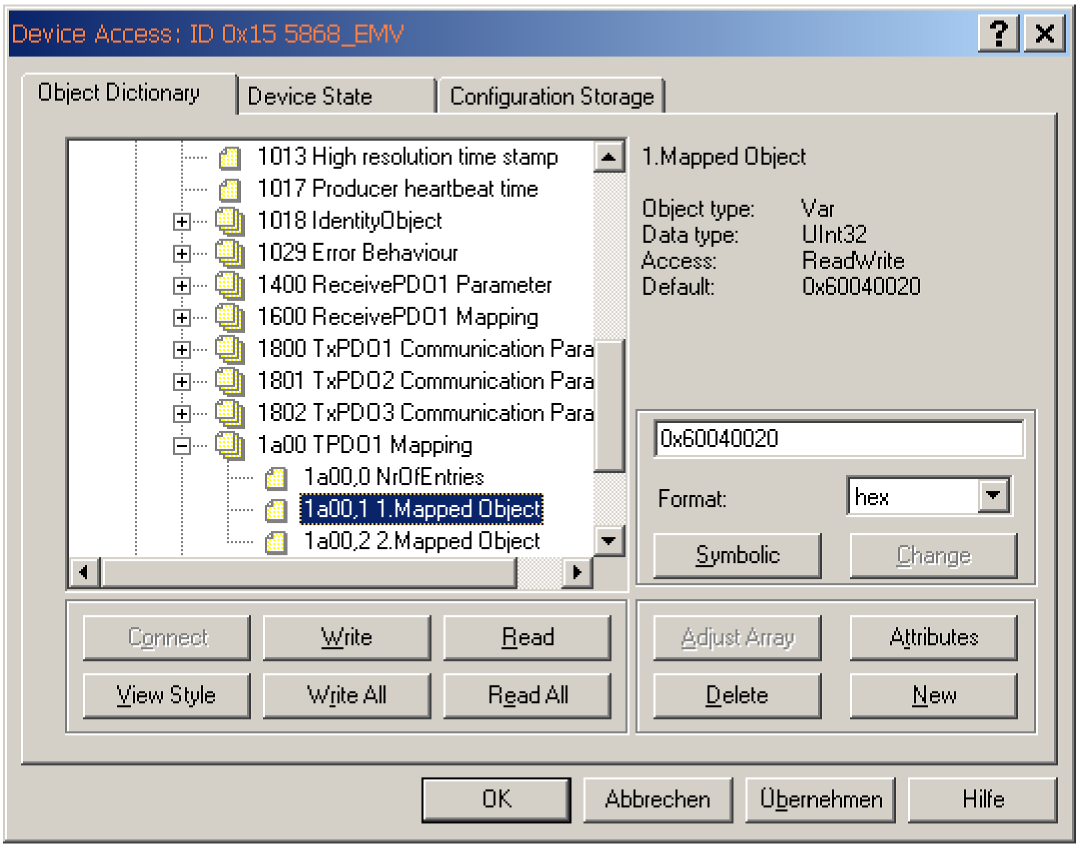
Example of an entry in the Mapping Table:
The mapped PDO consists of 3 Application Object entries of varying lengths:



Application Object 2 occupies Byte 1 (08h) in the Transmit PDO. Thereafter follows Application Object 3 with a length of 16 bit (10h = 2 bytes) and finally Application Object 1 with a length of 1 byte. In total, 32 bits are occupied in this PDO.

Structure of a Mapping entry

The Mapping Object for **PDO 1** is defined in the Object Dictionary Index 1A00h. It consists of 2 entries and can be modified by the user (variable mapping).



1013 High resolution time stamp
1017 Producer heartbeat time
1018 IdentityObject
1029 Error Behaviour
1400 ReceivePDO1 Parameter
1600 ReceivePDO1 Mapping
1800 TxPDO1 Communication Para
1801 TxPDO2 Communication Para
1802 TxPDO3 Communication Para
1a00 TPDO1 Mapping
1a00,0 NrOfEntries
1a00,1 1.Mapped Object
1a00,2 2.Mapped Object

1.Mapped Object

Object type: Var
Data type: UInt32
Access: ReadWrite
Default: 0x60040020

0x60040020

Format: hex

Byte:	MSB		LSB
	index (16 bit)	sub-index (8 bit)	object length (8 bit)

The default setting for the **Mapping of the Transmit PDO**:

Mapping	TPDO1	TPDO2	TPDO3
1.Mapping	**0x60040020**	**0x60040020**	**0x60300110**
Object	**6004h**	**6004h**	**6030h**
Subindex	**00**	**00**	**01**
Data length	**20h(32 Bit)**	**20h(32 Bit)**	**10h(16 Bit)**
	Asynchron	**Synchron**	**Asynchron**

The CANopen encoder supports **variable mapping** on all 3 Transmit PDOs.

9 Example of a variable Mapping table

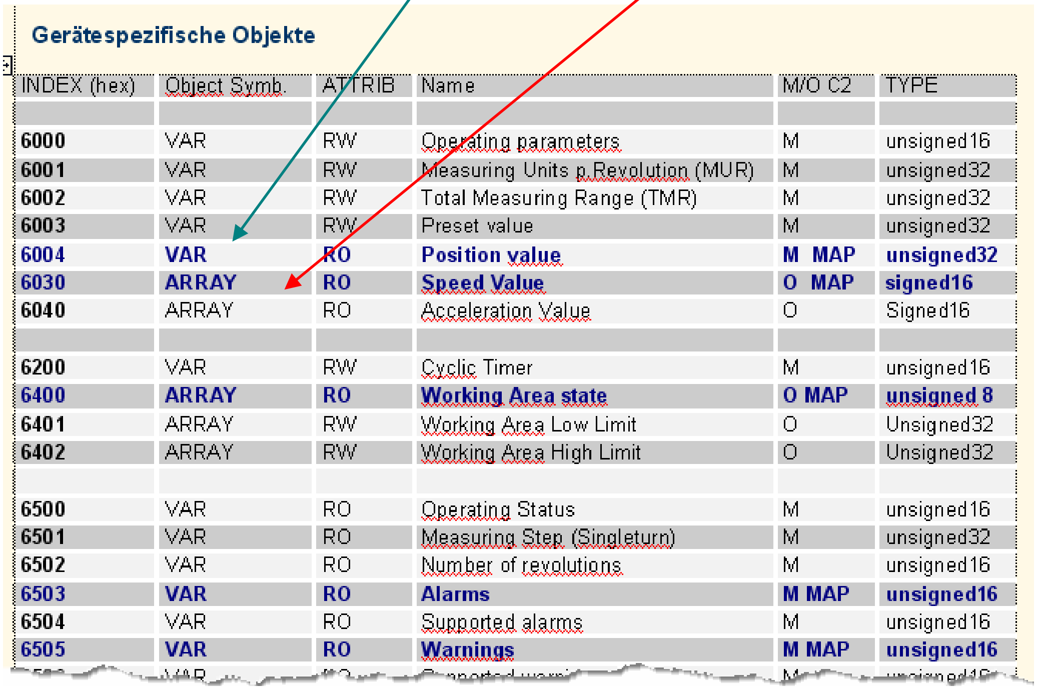
Mapping Object 1A00h

The Mapping Object **1A00h describes the 1.Transmit PDO**. It is possible to map as much as objects, until the maximum data length of **8 Bytes** is reached.
The Objects **1A01h of Transmit PDO2** and **1A02h of Transmit PDO3** following the input of **PDO1**.

Mapping table of Object 1A00h:

Mapping	TPDO1 Mapping	TPDO1 Mapping	TPDO1 Mapping
Subindex	00	01	02
Content	Nr.of Entries	1.Mapped Object	2.Mapped Object
Object	2	6004h	6030h
Subindex		00	01
Length	Byte	20h(32 Bit)	10h(16 Bit)
		Asynchronous	Asynchronous

Encoder Profile - Device specific Objetcs



Gerätespezifische Objekte

INDEX (hex)	Object Symb.	ATTRIB	Name	M/O C2	TYPE
6000	VAR	RW	Operating parameters	M	unsigned16
6001	VAR	RW	Measuring Units p.Revolution (MUR)	M	unsigned32
6002	VAR	RW	Total Measuring Range (TMR)	M	unsigned32
6003	VAR	RW	Preset value	M	unsigned32
6004	VAR	RO	Position value	M MAP	unsigned32
6030	ARRAY	RO	Speed Value	O MAP	signed16
6040	ARRAY	RO	Acceleration Value	O	Signed16
6200	VAR	RW	Cyclic Timer	M	unsigned16
6400	ARRAY	RO	Working Area state	O MAP	unsigned 8
6401	ARRAY	RW	Working Area Low Limit	O	Unsigned32
6402	ARRAY	RW	Working Area High Limit	O	Unsigned32
6500	VAR	RO	Operating Status	M	unsigned16
6501	VAR	RO	Measuring Step (Singleturn)	M	unsigned32
6502	VAR	RO	Number of revolutions	M	unsigned16
6503	VAR	RO	Alarms	M MAP	unsigned16
6504	VAR	RO	Supported alarms	M	unsigned16
6505	VAR	RO	Warnings	M MAP	unsigned16
650	VAR	RO	Supported warni	M	unsigned16

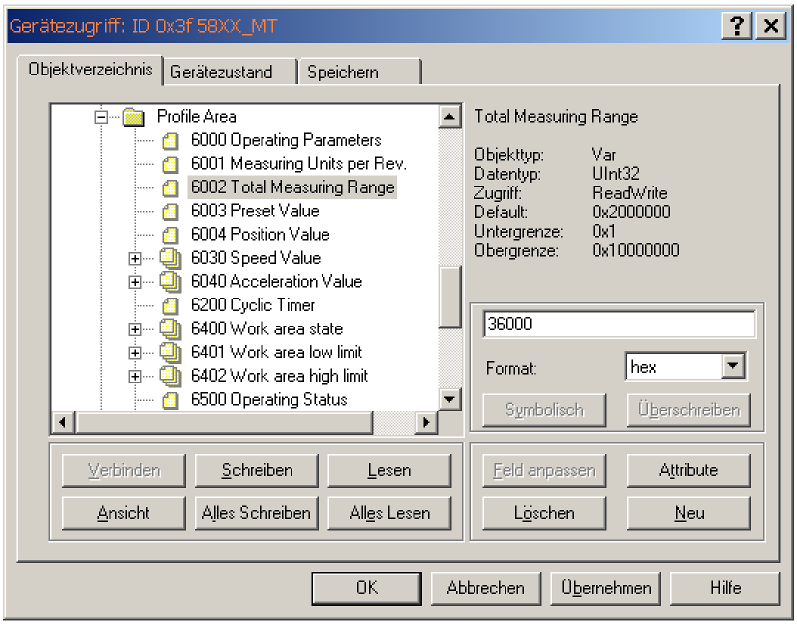
In this example two Objects are mapped to the 1.Transmit PDO at 1800h

10 Application Programming Example:

Setting up Objects

- **Total Measuring Range to 3600**
- **Measuring Units per Revolution** should be set to 3600 steps per revolution
- Position Value should be set to 0
- TPDO1 (Position) should transmit the event every 10 ms
- TPDO2 (Speed) should transmit the event every 20 ms
- **Producer Heartbeat** should be reduced to 500 ms
- Work area limits are 1000 and 3500
- The new parameters should be saved in the **EEPROM**

Total Measuring Range set to 3600



Gerätezugriff: ID 0x3f 58XX_MT

Profile Area
- 6000 Operating Parameters
- 6001 Measuring Units per Rev.
- 6002 Total Measuring Range
- 6003 Preset Value
- 6004 Position Value
- 6030 Speed Value
- 6040 Acceleration Value
- 6200 Cyclic Timer
- 6400 Work area state
- 6401 Work area low limit
- 6402 Work area high limit
- 6500 Operating Status

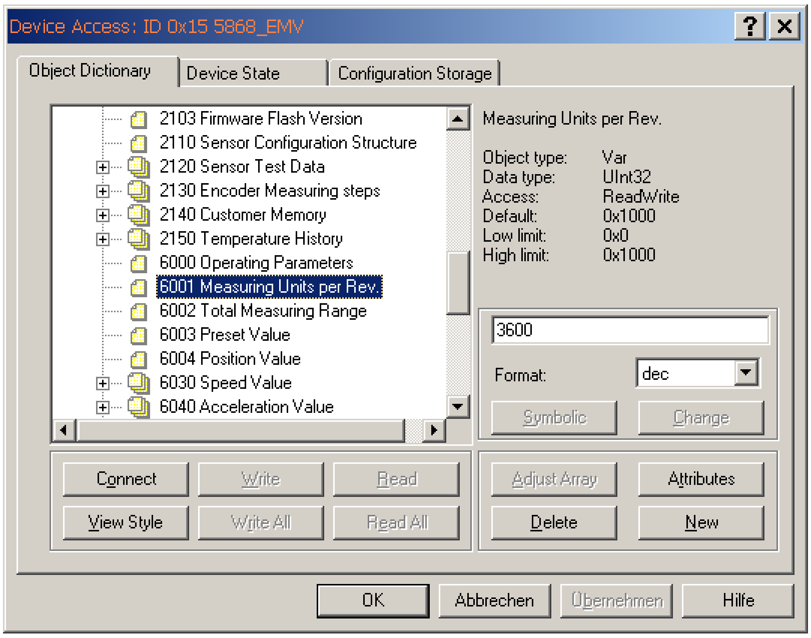
Total Measuring Range

Objekttyp:	Var
Datentyp:	UInt32
Zugriff:	ReadWrite
Default:	0x2000000
Untergrenze:	0x1
Obergrenze:	0x10000000

36000

Format: hex

Measuring Units per Revolution – limit to 3600



Device Access: ID 0x15 5868_EMV

- 2103 Firmware Flash Version
- 2110 Sensor Configuration Structure
- 2120 Sensor Test Data
- 2130 Encoder Measuring steps
- 2140 Customer Memory
- 2150 Temperature History
- 6000 Operating Parameters
- 6001 Measuring Units per Rev.
- 6002 Total Measuring Range
- 6003 Preset Value
- 6004 Position Value
- 6030 Speed Value
- 6040 Acceleration Value

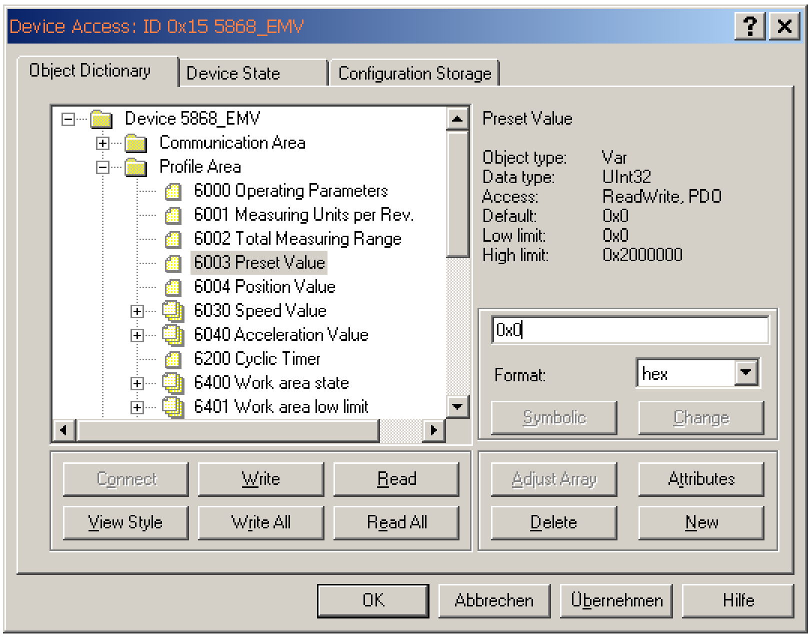
Measuring Units per Rev.

Object type:	Var
Data type:	UInt32
Access:	ReadWrite
Default:	0x1000
Low limit:	0x0
High limit:	0x1000

3600

Format: dec

Set Preset Value to 0


Device Access: ID 0x15 5868_EMV

Object Dictionary | Device State | Configuration Storage

- Device 5868_EMV
  - Communication Area
  - Profile Area
    - 6000 Operating Parameters
    - 6001 Measuring Units per Rev.
    - 6002 Total Measuring Range
    - 6003 Preset Value
    - 6004 Position Value
    - 6030 Speed Value
    - 6040 Acceleration Value
    - 6200 Cyclic Timer
    - 6400 Work area state
    - 6401 Work area low limit

Preset Value

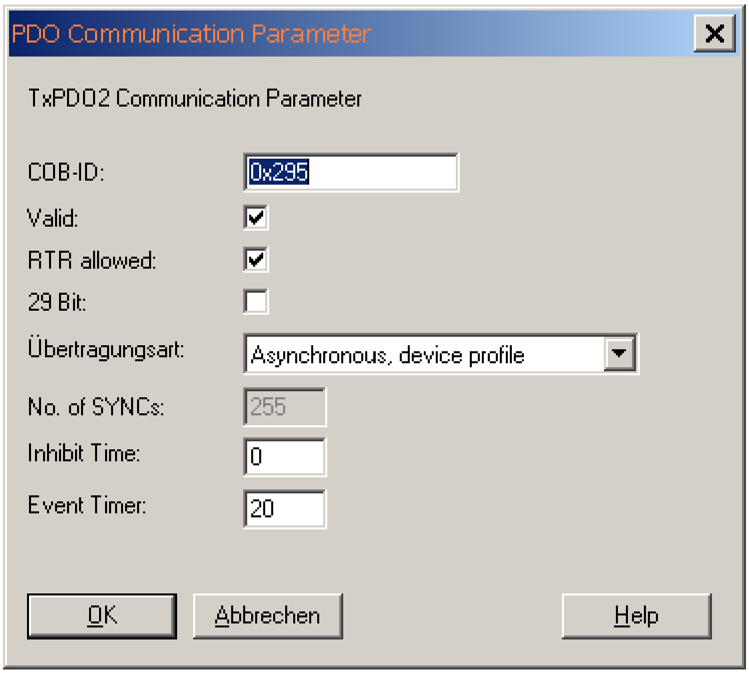
Object type:	Var
Data type:	UInt32
Access:	ReadWrite, PDO
Default:	0x0
Low limit:	0x0
High limit:	0x2000000

0x0

Format: hex

Set the values of Transmit Parameters TPDO1 and TPDO2

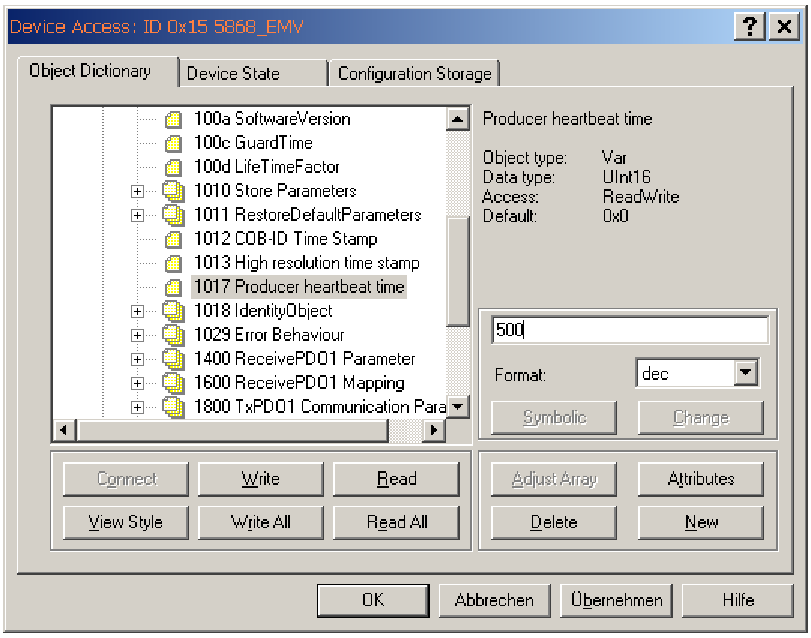
Type 254 means that the event will be triggered depending on the application, whereas Number 255 is **dependent on the device**. Additionally for Numbers 254/255 a time-controlled **EventTimer** can be used. The values for the timer can range from **1ms ... 65535 ms**.


PDO Communication Parameter

TxPDO2 Communication Parameter

COB-ID:	0x295
Valid:	☑
RTR allowed:	☑
29 Bit:	□
Übertragungsart:	Asynchronous, device profile
No. of SYNCs:	255
Inhibit Time:	0
Event Timer:	20

Producer Heartbeat - set to 500 ms



100a SoftwareVersion
100c GuardTime
100d LifeTimeFactor
1010 Store Parameters
1011 RestoreDefaultParameters
1012 COB-ID Time Stamp
1013 High resolution time stamp
1017 Producer heartbeat time
1018 IdentityObject
1029 Error Behaviour
1400 ReceivePDO1 Parameter
1600 ReceivePDO1 Mapping
1800 TxPDO1 Communication Para

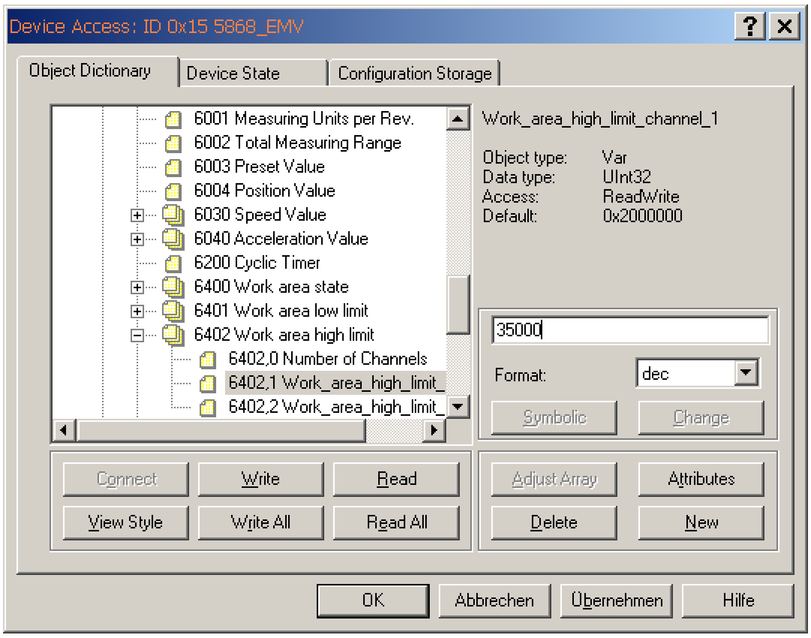
Producer heartbeat time

Object type:	Var
Data type:	UInt16
Access:	ReadWrite
Default:	0x0

500

Format: dec

Set Work Area low and high limit values



6001 Measuring Units per Rev.
6002 Total Measuring Range
6003 Preset Value
6004 Position Value
6030 Speed Value
6040 Acceleration Value
6200 Cyclic Timer
6400 Work area state
6401 Work area low limit
6402 Work area high limit
6402,0 Number of Channels
6402,1 Work_area_high_limit_
6402,2 Work_area_high_limit_

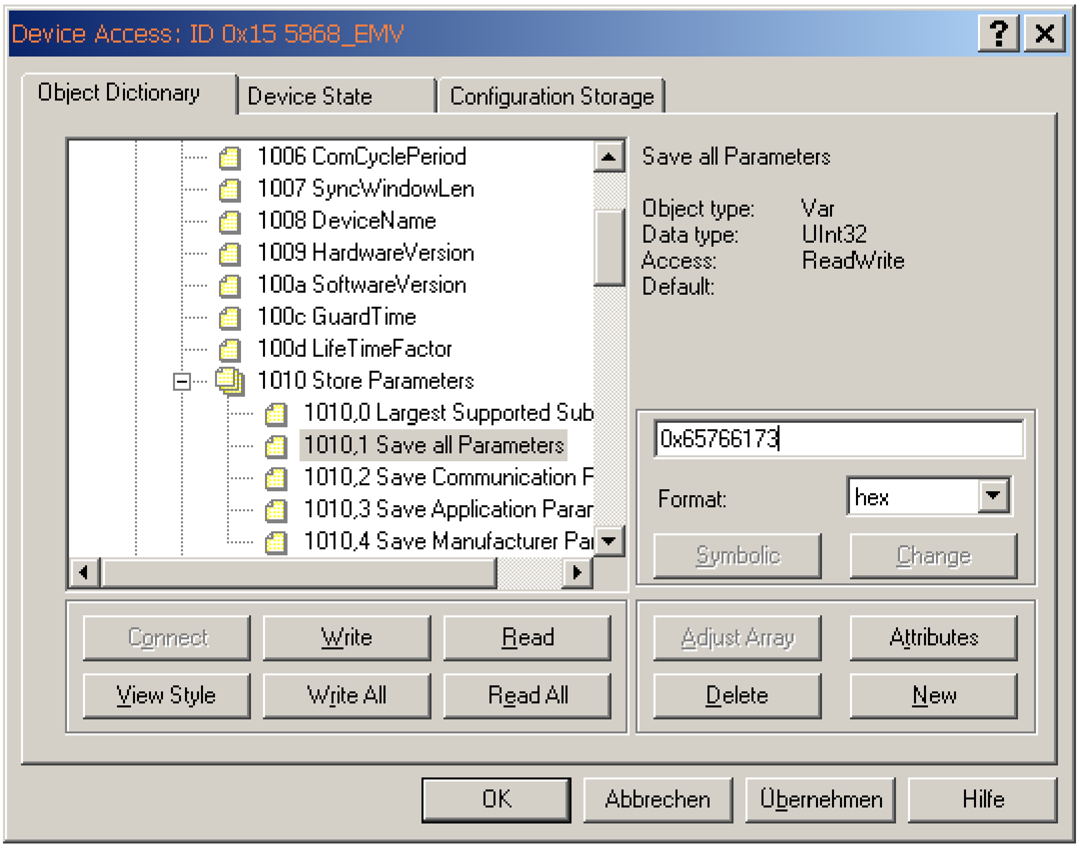
Work_area_high_limit_channel_1

Object type:	Var
Data type:	UInt32
Access:	ReadWrite
Default:	0x2000000

35000

Format: dec

Save all modified parameters in the EEPROM
Store Parameters 1010h



Save all Parameters

Object type: Var
Data type: UInt32
Access: ReadWrite
Default:

0x65766173

Format: hex

Object 1010h Store Parameters

Using the command "save" under Sub-Index 1h (save all Parameters) causes all the parameters to be stored in the non-volatile memory (EEPROM). All Communication Objects, Application Objects and Manufacturer-specific Objects are saved under this Sub-Index. This process requires ca. 14 ms.
In order to prevent an inadvertent save, the instruction will only be executed if the string "save" is entered as a codeword into this Sub-Index. A read access to the Sub-Index 1h provides information about the functionality of the memory.

Term	Content	Notes
Byte 3	**73h**	**(ASCII Code für "s")**
Byte 2	**61h**	**(ASCII Code für "a")**
Byte 1	**76h**	**(ASCII Code für "v")**
Byte 0	**65h**	**(ASCII Code für "e")**

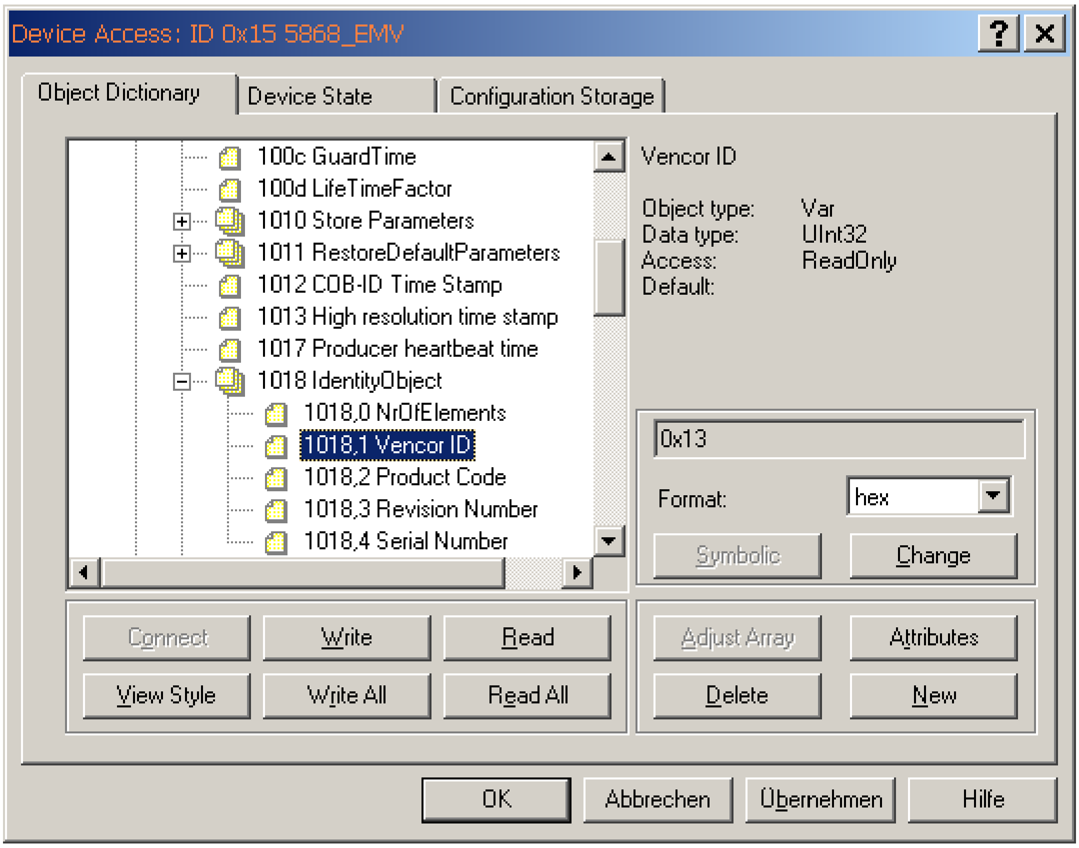
Object 1011h: Load Standard Values

Using the command "load" under Sub-Index 1h causes all parameters to be reset to their standard values. In order to prevent inadvertent loading of the standard values, the instruction will only be executed if the string "load" is entered as a codeword into this Sub-Index.

Term	Content	Notes
Byte 3	**6Ch**	**(ASCII Code für "l")**
Byte 2	**6Fh**	**(ASCII Code für "o")**
Byte 1	**61h**	**(ASCII Code für "a")**
Byte 0	**64h**	**(ASCII Code für "d")**

Communication Profile – further objects
Object 1018h: Identity Object

Information concerning the vendor and the device:


Device Access: ID 0x15 5868_EMV

Object Dictionary | Device State | Configuration Storage

- 100c GuardTime
- 100d LifeTimeFactor
- 1010 Store Parameters
- 1011 RestoreDefaultParameters
- 1012 COB-ID Time Stamp
- 1013 High resolution time stamp
- 1017 Producer heartbeat time
- 1018 IdentityObject
  - 1018,0 NrOfElements
  - 1018,1 Vencor ID
  - 1018,2 Product Code
  - 1018,3 Revision Number
  - 1018,4 Serial Number

Vencor ID

Object type: Var
Data type: UInt32
Access: ReadOnly
Default:

0x13

Format: hex

1018 RECORD Device – Identification read only

Sub-Index 0h : Number of Sub-indices“
supplies the value 4
Sub-Index 1h: "read" only
supplies the Vendor-ID (000000013h) Fritz Kübler GmbH
Sub-Index 2h: supplies the Product Code
(e.g. 0x36582001 CANopen encoder)
Sub-Index 3h: "read" only
supplies the Software revision Number
(e.g. 102)
Sub-Index 4h: "read" only
supplies the 8-digit Serial Number of the encoder

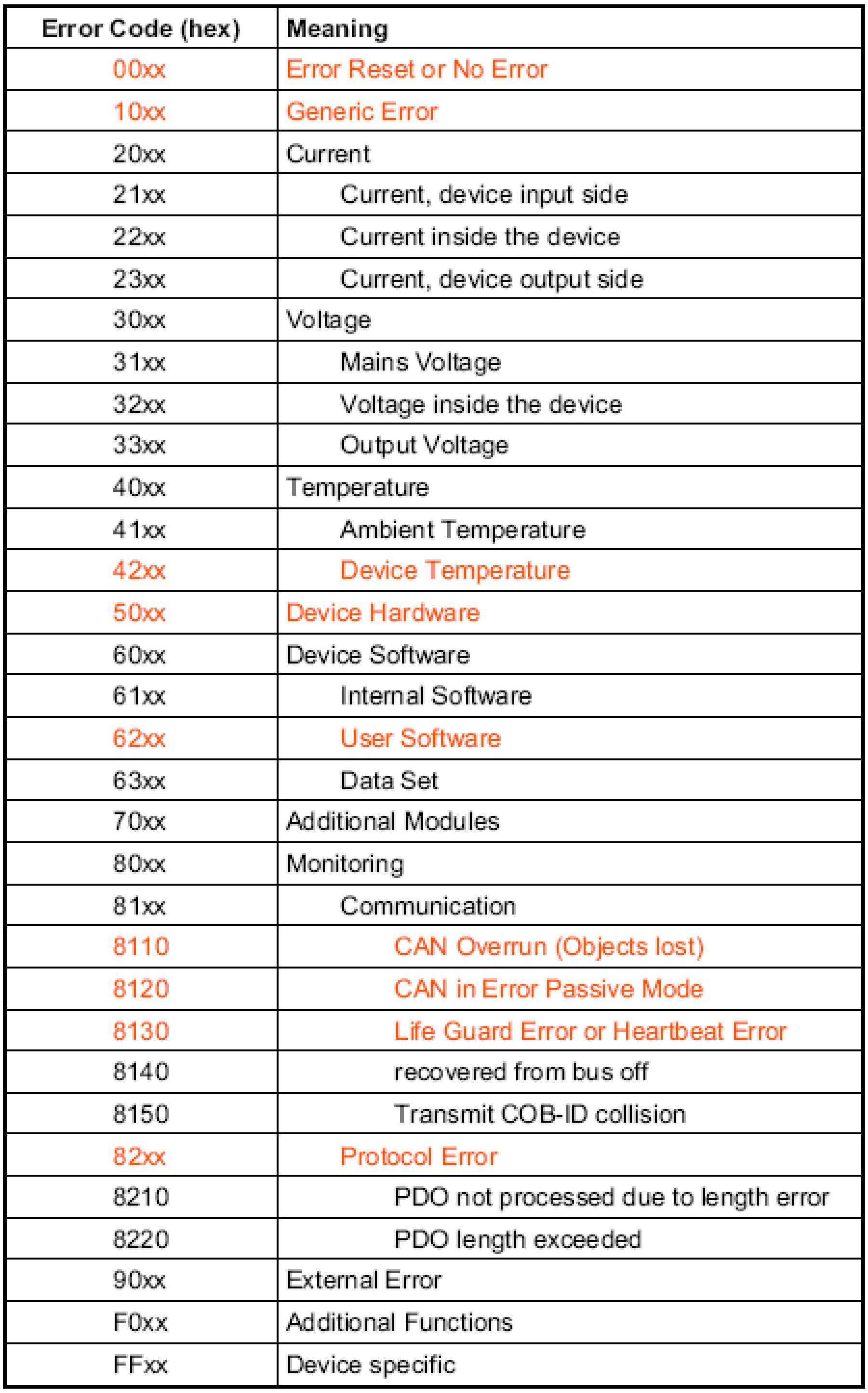
11 Emergency Objects

Emergency Objects arise with error situations within a CAN network and are triggered depending on the event and transmitted over the bus with a **high priority**.

Important: an Emergency Object is only triggered once per “Event”. No new object is generated while the error still exists. Once the error is eliminated, then a new Emergency Object with the content 0 (Error Reset or No Error) is generated and transmitted over the bus.

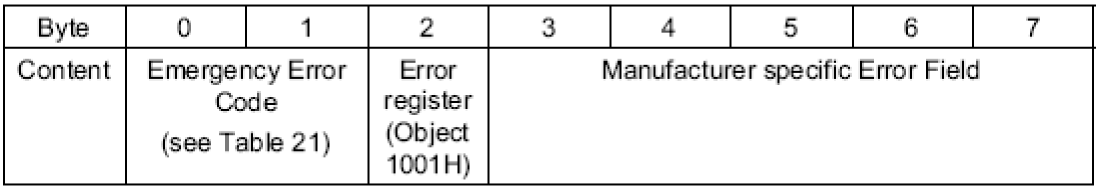
Error Codes supported

The Error Codes are highlighted in red



Error Code (hex)	Meaning
00xx	Error Reset or No Error
10xx	Generic Error
20xx	Current
21xx	Current, device input side
22xx	Current inside the device
23xx	Current, device output side
30xx	Voltage
31xx	Mains Voltage
32xx	Voltage inside the device
33xx	Output Voltage
40xx	Temperature
41xx	Ambient Temperature
42xx	Device Temperature
50xx	Device Hardware
60xx	Device Software
61xx	Internal Software
62xx	User Software
63xx	Data Set
70xx	Additional Modules
80xx	Monitoring
81xx	Communication
8110	CAN Overrun (Objects lost)
8120	CAN in Error Passive Mode
8130	Life Guard Error or Heartbeat Error
8140	recovered from bus off
8150	Transmit COB-ID collision
82xx	Protocol Error
8210	PDO not processed due to length error
8220	PDO length exceeded
90xx	External Error
F0xx	Additional Functions
FFxx	Device specific

12 Emergency Message



Byte	0	1	2	3	4	5	6	7
Content	Emergency Error Code (see Table 21)		Error register (Object 1001H)	Manufacturer specific Error Field				

Figure 34: Emergency Object Data

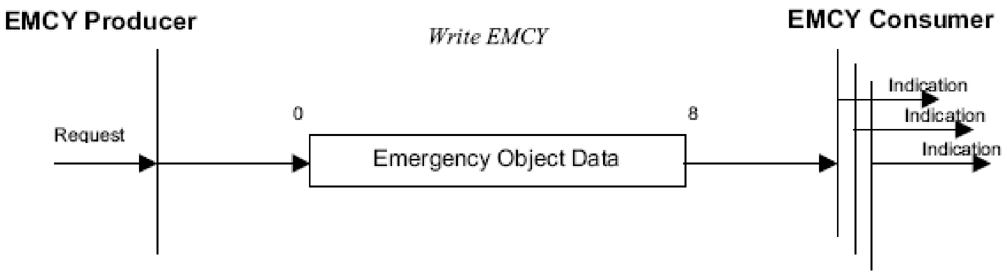
Example of an over-temperature message:

Transfer Data	**00**	**42**	**09**	**80**	**56**	**20**	**50**	**2E**

[Errcode]	**4200**	**Device Overtemperature**
[Error Register]	**09**	**Error Register**
[ManufacturerSpecific1]	**80**	Diagnosebytes
[ManufacturerSpecific2]	**56**	Diagnosebytes
[ManufacturerSpecific3]	**20**	Diagnosebytes
[ManufacturerSpecific4]	**50**	Diagnosebytes
[ManufacturerSpecific5]	**2E**	Diagnosebytes

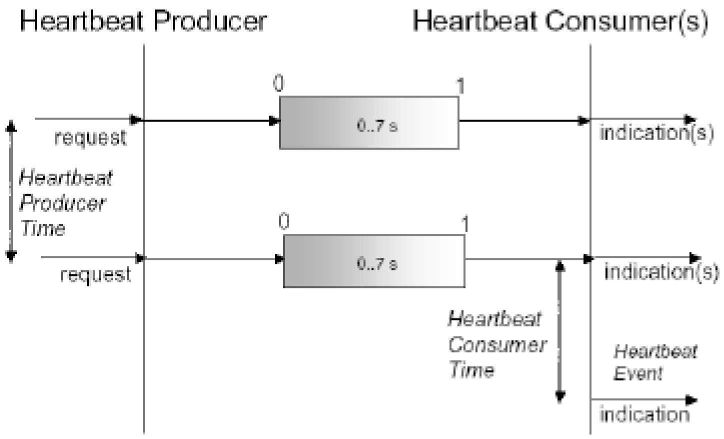
Emergency Protocol

An "unconfirmed" Service message is defined



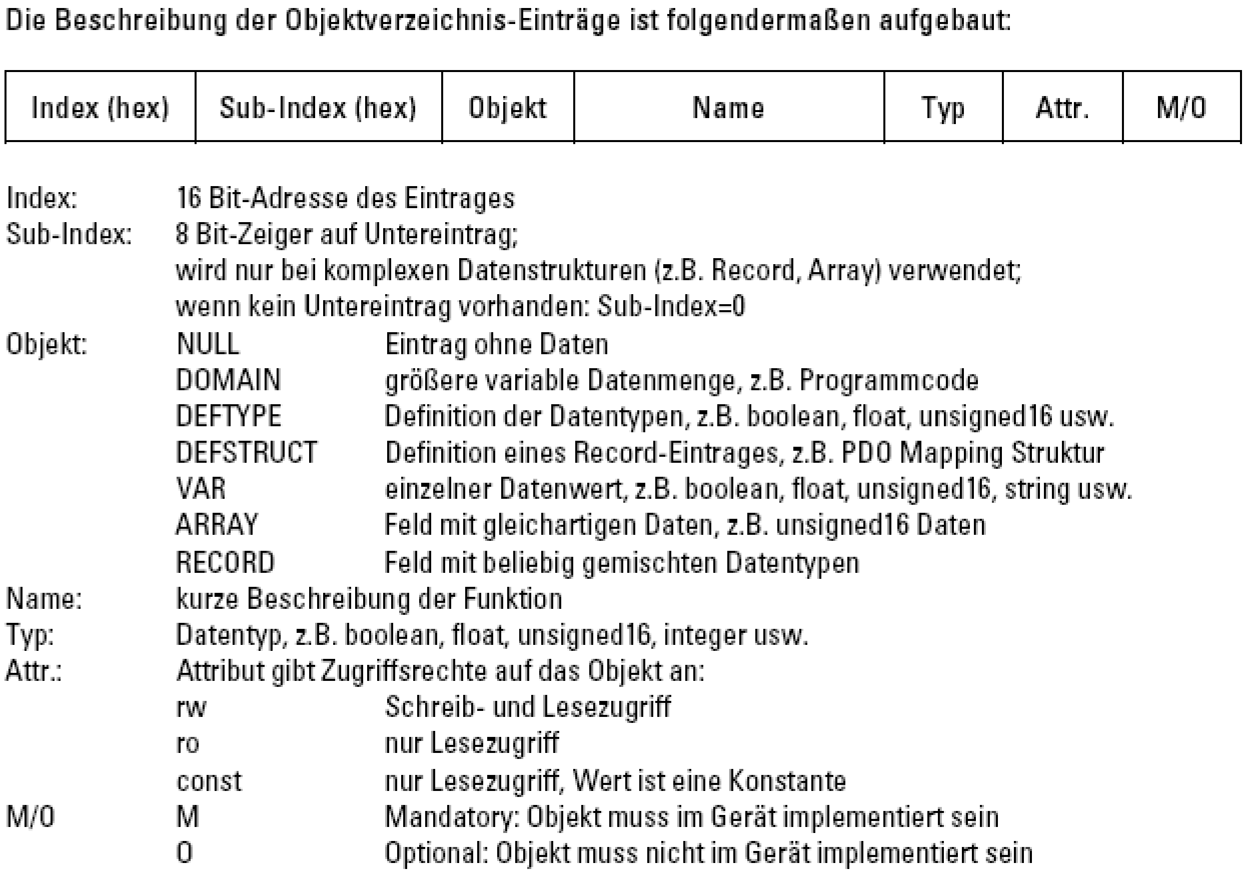
The behaviour in the case of an error is described in **Object 1029h Error Behaviour**

13 Heartbeat Protocol



Nowadays as an alternative to **Node Guarding** the modern **Heartbeat Protocol** should be used. The protocol is activated if a value > 0 is written to **Object 1017h** Producer Heartbeat Time.
A "Heartbeat–Producer" cyclically transmits this Heartbeat message. One or more "Heartbeat-Consumer(s)" can receive this Heartbeat message.
If the cyclic transmission of this Heartbeat message is missing, then a "Heartbeat Event" is generated. The behaviour in the case of an error is defined in Object 1029h Subindex 1 "Communication Error".

14 CANopen Object Dictionary



Die Beschreibung der Objektverzeichnis-Einträge ist folgendermaßen aufgebaut:

Index (hex)	Sub-Index (hex)	Objekt	Name	Typ	Attr.	M/O

Index:	16 Bit-Adresse des Eintrages	
Sub-Index:	8 Bit-Zeiger auf Untereintrag; wird nur bei komplexen Datenstrukturen (z.B. Record, Array) verwendet; wenn kein Untereintrag vorhanden: Sub-Index=0	
Objekt:	NULL	Eintrag ohne Daten
	DOMAIN	größere variable Datenmenge, z.B. Programmcode
	DEFTYPE	Definition der Datentypen, z.B. boolean, float, unsigned16 usw.
	DEFSTRUCT	Definition eines Record-Eintrages, z.B. PDO Mapping Struktur
	VAR	einzelner Datenwert, z.B. boolean, float, unsigned16, string usw.
	ARRAY	Feld mit gleichartigen Daten, z.B. unsigned16 Daten
	RECORD	Feld mit beliebig gemischten Datentypen
Name:	kurze Beschreibung der Funktion	
Typ:	Datentyp, z.B. boolean, float, unsigned16, integer usw.	
Attr.:	Attribut gibt Zugriffsrechte auf das Objekt an:	
	rw	Schreib- und Lesezugriff
	ro	nur Lesezugriff
	const	nur Lesezugriff, Wert ist eine Konstante
M/O	M	Mandatory: Objekt muss im Gerät implementiert sein
	O	Optional: Objekt muss nicht im Gerät implementiert sein

Structure of the entire Object Dictionary:

Index (hex)	Object
0000	unused
0001 - 001F	static date types
0020 - 003F	complex data types
0040 - 005F	manufacturer-specific data types
0060 - 0FFF	reserved
1000 - 1FFF	Communication Profile
2000 - 5FFF	Manufacturer-specific Profile
6000 - 9FFF	Standardized Device Profile
A000 - FFFF	reserved

15 CANopen Communication Profile DS 301 V4.02

Communication Objects

INDEX (hex)	OBJECT SYMBOL	ATTRIB	Name	M/O	TYPE
1000	VAR	CONST	Device Type	M	Unsigned32
1001	VAR	RO	Error Register	M	Unsigned8
1002	VAR	RO	Manufacturer Status	O	Unsigned32
1003	RECORD	RO	Predefined Error Field	O	Unsigned32
1004	ARRAY	RO	Number of PDO supported	O	Unsigned32
1005	VAR	RW	COB-ID Sync message	O	Unsigned32
1006	VAR	RW	Communication cycle period	O	Unsigned32
1007	VAR	RW	synchr.window length	O	Unsigned32
1008	VAR	CONST	Manufacturer Device Name	O	visible string
1009	VAR	CONST	Manufacturer Hardware Version	O	visible string
100A	VAR	CONST	Manufacturer Software Version	O	visible string
100B	VAR	RO	Node-ID	O	Unsigned32
100C	VAR	RW	Guard Time	O	Unsigned32
100D	VAR	RW	LifeTime Factor	O	Unsigned32
1010	VAR	RW	Store parameters (Device Profile)	O	Unsigned32
1011	VAR	RW	Restore parameters (Device Profile)	O	Unsigned32
1014	VAR	RO	COB_ID Emcy	O	Unsigned32
1015	VAR	RW	IInhibit Time Emcy	O	Unsigned32
1017	VAR	RW	Producer Heartbeat time	O	Unsigned16
1018	RECORD	RO	Identity Object	M	PDOComPar
1029	ARRAY	RW	Error Behaviour	O	Unsigned8
1800	**RECORD**		**1st transmit PDO Comm. Par.**	**O**	**PDOComPar**
1801	**RECORD**		**2nd transmit PDO Comm. Par.**	**O**	**PDOComPar**
1802	**RECORD**		**3rd transmit PDO Comm. Par.**	**O**	**PDOComPar**
1A00	**ARRAY**		**1st transmit PDO Mapping Par.**	**O**	**PDOMapping**
1A01	**ARRAY**		**2nd transmit PDO Mapping Par.**	**O**	**PDOMapping**
1A02	**ARRAY**		**3rd transmit PDO Mapping Par.**	**O**	**PDOMapping**

Manufacturer specific Objects

2100	VAR	RW	Baud Rate	O	Unsigned 8
2101	VAR	RW	Node number	O	Unsigned 8
2102	VAR	RW	CAN Bus Termination	O	Unsigned 8
2103	VAR	RO	Firmware Flash Version	O	Unsigned16
2105	VAR	RW	Save All Bus Parameters	O	Unsigned32
2140	Array	RW	Customer Memory	O	Unsigned32

16 CANopen Encoder Device Profile DS 406 V3.2

Device-specific Objects

INDEX (hex)	Object Symb.	ATTRIB	Name	M/O C2	TYPE
6000	VAR	RW	Operating parameters	M	unsigned16
6001	VAR	RW	Measuring Units p.Revolution (MUR)	M	unsigned32
6002	VAR	RW	Total Measuring Range (TMR)	M	unsigned32
6003	VAR	RW	Preset value	M	unsigned32
6004	**VAR**	**RO**	**Position value**	**M MAP**	**unsigned32**
6030	**ARRAY**	**RO**	**Speed Value**	**O MAP**	**signed16**
6040	ARRAY	RO	Acceleration Value	O	Signed16
6200	VAR	RW	Cyclic Timer	M	unsigned16
6400	**ARRAY**	**RO**	**Working Area state**	**O MAP**	**unsigned 8**
6401	ARRAY	RW	Working Area Low Limit	O	Unsigned32
6402	ARRAY	RW	Working Area High Limit	O	Unsigned32
6500	VAR	RO	Operating Status	M	unsigned16
6501	VAR	RO	Measuring Step (Singleturn)	M	unsigned32
6502	VAR	RO	Number of revolutions	M	unsigned16
6503	**VAR**	**RO**	**Alarms**	**M MAP**	**unsigned16**
6504	VAR	RO	Supported alarms	M	unsigned16
6505	**VAR**	**RO**	**Warnings**	**M MAP**	**unsigned16**
6506	VAR	RO	Supported warnings	M	unsigned16
6507	VAR	RO	Profile and SW version	M	unsigned32
6508	VAR	RO	Operating time	M	unsigned32
6509	VAR	RO	Offset value (calculated)	M	signed32
650A	VAR	RO	Module Identification	M	signed32
650B	VAR	RO	Serial Number	M	unsigned32

VAR	= Variable
ARRAY	= Variable Array
RW	= Read/Write
RO	= Read only
const	= Constants
Name	= Object Name
M/O	= Mandatory or Optional
MAP	**= Object mappable**

17 Objects in detail - Encoder Profile DS 306 V3.2

Object 6000h Operating Parameters

Bit 0: Code sequence:	0 = increasing when turning clockwise (cw) 1 = increasing when turning counter-clockwise (ccw) **Default: Bit = 0**
Bit 2: Scaling Function:	0 = disable, 1 = enable; Standard: Bit = 1 (s. Object 6001,6002) **Default: Bit = 0**
Bit14: Startup Mode:	0 = after Bootup Pre-Operational, 1 = **after Bootup Operational mode** **Default Bit = 0**
Bit15: Event Mode:	0 = Position output acc. to TPDO 1800h, 1 = output on each change of position **Default Bit = 0**

Bit	Function	Bit = 0	Bit =1	C1	C2
0	**Code sequence**	**CW**	**CCW**	**m***	**m***
1	Commissioning Diagnostic Control	Disabled	Enabled	o	o
2	**Enable scaling**	**Disabled**	**Enabled**	**o**	**m**
3	Measuring direction	Forward	Reverse	o**	o**
4..13	Reserved for further use				
14	**Startup automatic in OP-Mode**	**Disabled**	**Enabled**	**o**	**o**
15	**Event Mode Position**	**Disabled**	**Enabled**	**o**	**o**

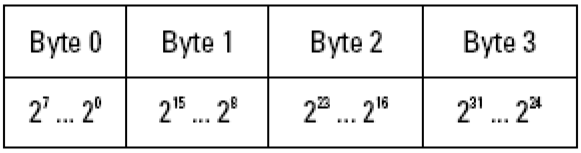
*m = Function must be supported
o = optional
orange = defaults

Object 6001h: measuring steps per revolution (Resolution)

This parameter configures the desired resolution per revolution. The encoder itself then internally calculates the appropriate scale factor. The calculated scaling factor MUR (by which the physical position value will be multiplied) is worked out according to the following formula:

MUR = Measuring steps per revolution (6001h) / **phys. resolution Singleturn (6501h)**

Data content:



Byte 0	Byte 1	Byte 2	Byte 3
$2^7 \dots 2^0$	$2^{15} \dots 2^8$	$2^{23} \dots 2^{16}$	$2^{31} \dots 2^{24}$

Range of values:	1....maximum physical resolution (16384) 14-bit
Default setting:	**16384(14-bit)**

After changing the measuring step it is necessary to set the preset value also to zero /or a value.

Object 6002h: Total number of measuring steps

This parameters configures the total number **Singleturn measuring steps**. A factor will be applied to the maximum physical resolution. The factor is always < 1 . After the stated number of measuring steps, the encoder will **reset itself to zero**.

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
$2^7 \ldots 2^0$	$2^{15} \ldots 2^8$	$2^{23} \ldots 2^{16}$	$2^{31} \ldots 2^{24}$

Range of values: 1....maximum physical resolution (16384) 14-bit
Default setting: 16384 (14-bit)

After changing the measuring step it is necessary to set the preset value also to zero /or a value.

Object 6003h: Preset Value

The position value of the encoder will be set to this preset value.
This allows, for example, for the encoder's zero position to be compared with the machine's zero position.

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
$2^7 \ldots 2^0$	$2^{15} \ldots 2^8$	$2^{23} \ldots 2^{16}$	$2^{31} \ldots 2^{24}$

Range of values: 1.... maximum physical resolution (16384) 14-bit
Default setting: 0

After transmitting the new preset value (object 6003h), it is necessary to achieve a store command at object 1010h because of the new offset values otherwise the preset is only valid during the actual power cycle.

Object 6004h: Position Value

The encoder transmits the current position value (adjusted possibly by the scaling factor)

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
$2^7 \ldots 2^0$	$2^{15} \ldots 2^8$	$2^{23} \ldots 2^{16}$	$2^{31} \ldots 2^{24}$

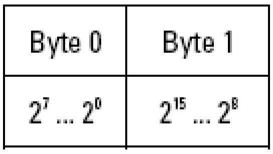
Range of values: 1.... maximum physical resolution (16384) 14-bit

Object is mappable

Object 6030h: Speed Value

The encoder outputs the current calculated speed (possibly with scaling factor) as a signed 16-bit value. The speed value is in **rpm** and the measuring time is fixed to **3ms**.

Data content:



Byte 0	Byte 1
$2^7 \dots 2^0$	$2^{15} \dots 2^8$

Range of values: 0....maximum speed 6500 RPM

With values greater than 6500 RPM a warning message will be sent and the Warning Bit "Overspeed Bit 0" in the Object Warnings 6505h will be set.

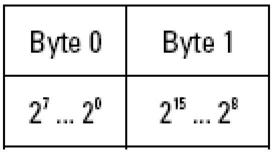
Object is mappable

Object 6200h: Cyclic Timer

Defines the cycle time, with which the current position will be output by means of PDO 1 (see Object 1800h). The timer-controlled output becomes active, as soon as a cycle time >0 is entered.

This Object is only present for reasons of compatibility with earlier profile versions. Instead of this Object, please use the Event Timer Sub index (05h) in the current Transmit PDO.

Data content:



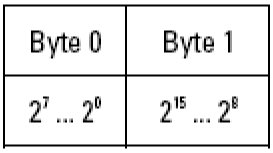
Byte 0	Byte 1
$2^7 \dots 2^0$	$2^{15} \dots 2^8$

Range of values: 0 ... FFFFh (65535) gives a cycle time in milliseconds
Standard value = 0h

Object 6500h: Display Operating Status

This Object displays the status of the programmed settings of Object 6000h.

Data content:



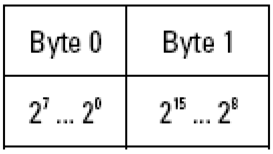
Byte 0	Byte 1
$2^7 \dots 2^0$	$2^{15} \dots 2^8$

Data content: see Object 6000h

Object 6503h: Alarms

In addition to the errors that are signalled via emergency messages, Object 6503h provides for further error messages. The corresponding error bit is set to 1 for as long as the error condition applies.

Data content:



Byte 0	Byte 1
$2^7 \dots 2^0$	$2^{15} \dots 2^8$

Bit No.	Description	Value = 0	Value = 1
Bit 0	**Position error**	Position value valid	Position error
Bit 1	**Hardware check**	No error	Error
Bit 2..15	**Not used**		

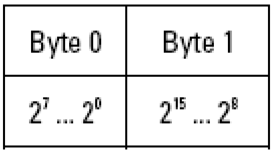
If an error occurs, then in both cases an emergency message **(ID=80h+node number)** with the error code **1000h (Generic error**) is sent.

Object is mappable

Object 6504h: Supported Alarms

This Object is used to display which alarm messages are supported by the encoder (see Object 6503h).

Data content:



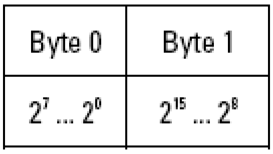
Byte 0	Byte 1
$2^7 \dots 2^0$	$2^{15} \dots 2^8$

Range of values: see Object 6503h
The alarm message is supported when the bit is set to 1
Example:
Bit 0 = 1   Position error display is supported

Object 6505h: Warnings

Warning messages show that tolerances of internal encoder parameters have been exceeded. With a warning message – unlike with an alarm message or emergency message – the measured value can still be valid. The corresponding warning bit will be set to 1 for as long as the tolerance is exceeded or the warning applies.

Data content:



Byte 0	Byte 1
$2^7 \ldots 2^0$	$2^{15} \ldots 2^8$

Bit No.	Description	Value = 0	Value = 1
Bit 0	**Overspeed**	none	exceeded
Bit 1..15	**Not used**		

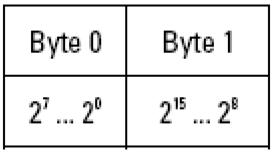
When Bit 0 is active then simultaneously an emergency message (ID=80h+node number) with the **Error code 4200h (**Device specific) is sent.

Object is mappable

Object 6506h: Supported Warnings

This Object is used to display which warning messages are supported by the encoder (see Object 6505h).

Data content:



Byte 0	Byte 1
$2^7 \ldots 2^0$	$2^{15} \ldots 2^8$

Range of values: see Object 6505h
The warning is supported when the bit is set to 1

Object 6400h: Working Area State Register

This Object contains the current state of the encoder position with respect to the programmed limits. The flags are either set or reset depending on the position of both limit values. The comparison with both limit values takes place in "real time" and can be used for real-time positioning or for limit switching.

Work_area_state							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
					smaller than LowLimit1	larger than HighLimit1	outside range1

Range of values 8-bit **Data content see Bit 0...7**

Both limit values Object 6401h and 6402h must be checked to ensure that the output signals are correctly activated !

Object is mappable

Object 6401h: Working Area Low Limit
Object 6402h: Working Area High Limit

These two parameters configure the working area. The state inside and outside this area can be signalled by means of Flag bytes (**Object 6400h Working Area State**). These area markers can also be used as software limit switches.

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
$2^7 \ldots 2^0$	$2^{15} \ldots 2^8$	$2^{23} \ldots 2^{16}$	$2^{31} \ldots 2^{24}$

Range of values: 1....maximum physical resolution (268435456) 28-bit
Default setting: 16384 (14-bit) Working Area High Limit
0 Working Area Low Limit

Object 2100h: Baud rate

This Object is used to change the baud rate via software. If the value is set between 1..9 and the parameter saved, then on the next Power ON or with a reset node, the device will boot up with the modified baud rate. After changing the baudrate it is necessary to save the parameters with **object 2105h** permanently in the EEprom.

Data content:

Byte 0
$2^7 \ldots 2^0$

Range of values 1 ...8 (see Table CANopen Baudrate)
Default setting: 05h

Object 2101h: Node address

This Object is used to change the node address via software. After changing the node address it is necessary to save the parameters with **object 2105h** permanently in the EEprom.

Data content:

Byte 0
$2^7 \dots 2^0$

Range of values 1 ...127 or 1..7Fh
Default setting: 3Fh

The **node number 0** is reserved and may not be used by any node. The resulting node numbers lie in the range **1...7Fh** hexadecimal or (1...127)

The acceptance of a new node number only becomes effective when the encoder is rebooted (Reset/Power-on) or by means of an **NMT Reset Node** command. All other settings within the object table are however retained.

Object 2102h: CAN bus termination OFF/ON

This Object can be used to set the bus termination via software. By default the value is set to 1. After changing the CAN bus Termination it is necessary to save the parameters with **object 2105h** permanently in the EEprom.

Data content:

Byte 0
$2^7 \dots 2^0$

Range of values 0..1
Default setting: 1

Please note that when software termination is selected, then the hardware settings are non-operative and vice versa.

Object 2103h: Firmware flash version

This object is used to display the current firmware version as a 16-bit hexadecimal value. This value serves to verify that the device is to the latest revision.

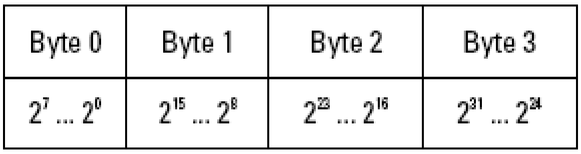
Data content:

Byte 0	Byte 1
$2^7 \dots 2^0$	$2^{15} \dots 2^8$

Range of values: to FFFFh
Example: 4FA6h current firmware

Object 2105h: Save All Bus Parameters

This object stores all bus parameters (Objekt 2100h ,2101h,2102h) permanently in an EEprom. Using the command "save" (save all Parameters) causes all the parameters to be stored. This process requires ca. 200ms. In order to prevent an inadvertent save, the instruction will only be executed if the string "save" is entered as a codeword into this Sub-Index.



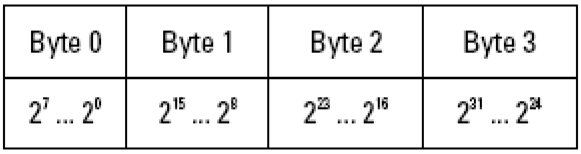
Byte 0	Byte 1	Byte 2	Byte 3
$2^7 \dots 2^0$	$2^{15} \dots 2^8$	$2^{23} \dots 2^{16}$	$2^{31} \dots 2^{24}$

Default value: **„save“ in hexadezimal 0x65766173**

Object 2140h: Customer Memory (16 Bytes)

These 4 parameters constitute a memory area for the user. **4 data words with a maximum of 4 bytes can be stored.** This area is not checked for content, which means in effect that any format can be filed.

Data content:



Byte 0	Byte 1	Byte 2	Byte 3
$2^7 \dots 2^0$	$2^{15} \dots 2^8$	$2^{23} \dots 2^{16}$	$2^{31} \dots 2^{24}$

Range of values: Numeric, alphanumeric
Default setting: 0

Object 1029h Error Behaviour

If a serious error is detected, then the device should automatically switch to **Pre-Operational** mode. The settings in this Object can be used to determine how the device is to behave when an error arises. The following error classes are covered.

1029h,Subindex 1 Communication Errors

- Bus Off state of the CAN interface
- Life guarding event has occurred
- Heartbeat monitoring has failed

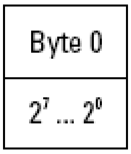
1029h,Subindex 2 Device Profile Specific

- Sensor error and Controller error
- Temperature error

1029h,Subindex 3 Manufacturer Profile Specific

- internal Controller error

The value of the Object classes is put together as follows:



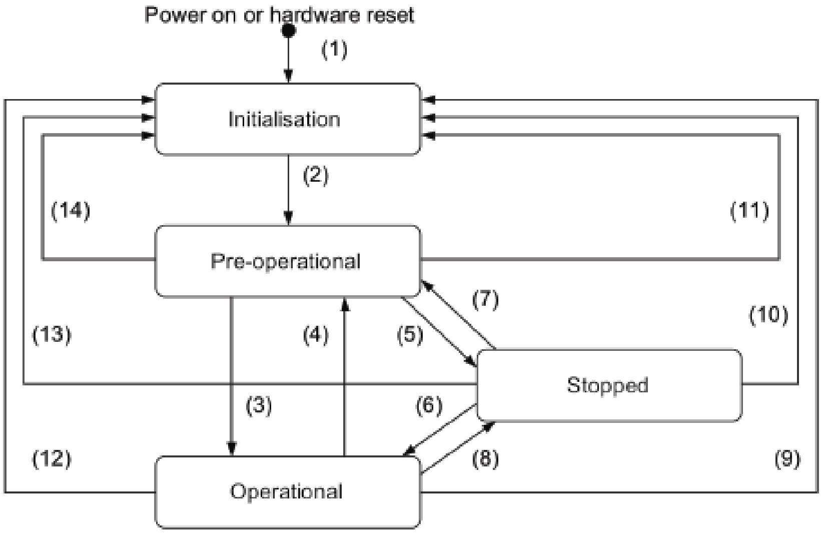
Byte 0
$2^7 \dots 2^0$

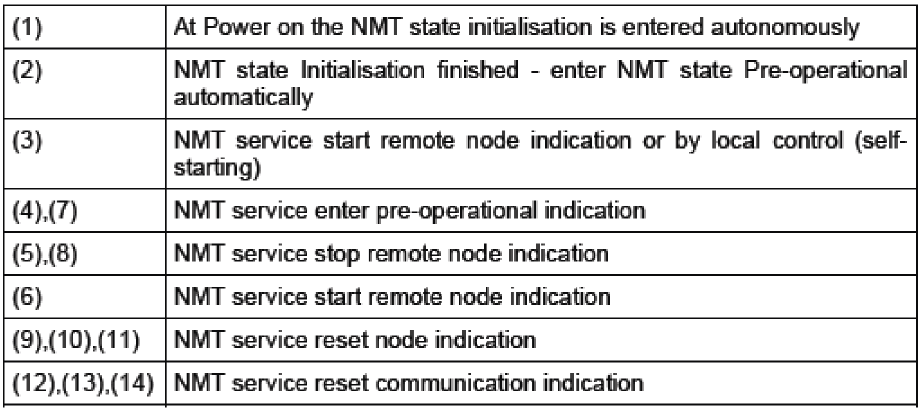
Range of values: 8-bit

- **0 Pre-Operational Mode (only if Operational Mode was active before)**
- **1 no change of mode**
- **2 Stopped Mode**
- **3 .. 127 reserved**

18 Network Management

The encoder supports the simplified Network Management as defined in the profile for "minimum capability devices" (minimum boot up).
The following function state diagram acc. to DS 301 shows the various node states and the corresponding network commands (controlled by the Network Master via NMT services):





(1)	At Power on the NMT state initialisation is entered autonomously
(2)	NMT state Initialisation finished - enter NMT state Pre-operational automatically
(3)	NMT service start remote node indication or by local control (self-starting)
(4),(7)	NMT service enter pre-operational indication
(5),(8)	NMT service stop remote node indication
(6)	NMT service start remote node indication
(9),(10),(11)	NMT service reset node indication
(12),(13),(14)	NMT service reset communication indication

Initialization: this is the initial state after the power supply is applied, following a device Reset or Power ON. The node automatically enters the Pre-operational state once it has run through the Reset and Initialization routines. The LEDs display the momentary status.

Pre-operational: The CAN node can now be addressed via SDO messages or with NMT commands under the standard identifier. Then follows the programming of the encoder or communication parameters.

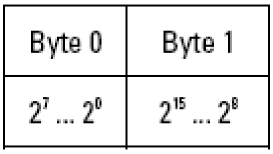
Operational: The node is active. Process values are transmitted over the PDOs. All NMT commands can be evaluated.

Prepared or **Stopped:** In this state the node is no longer active, which means that neither SDO nor PDO communications are possible. The node can be set to either the Operational or Pre-operational state by means of NMT commands.

19 NMT Commands

All NMT commands are transferred as an unconfirmed NMT Object. Because of the broadcast (network-wide) communication model, the NMT commands are recognized by each station.
An NMT Object is structured as follows:

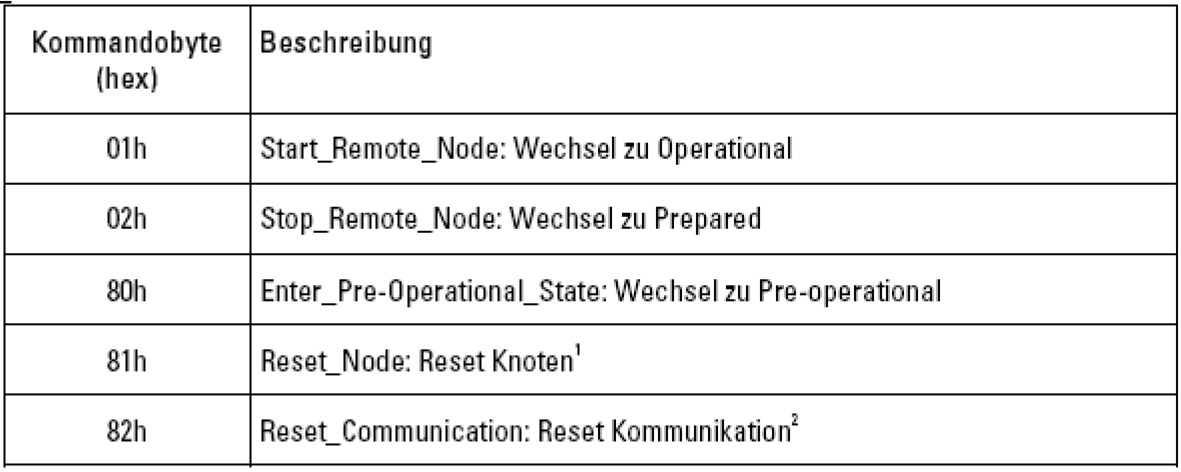
COB-ID = 0



Byte 0	Byte 1
$2^7 \dots 2^0$	$2^{15} \dots 2^8$

Byte 0 = Command byte
Byte 1 = Node number

The COB-ID of the NMT Object is always 0
The node is addressed via the node numbers. With node number 0 all nodes are addressed.



Kommandobyte (hex)	Beschreibung
01h	Start_Remote_Node: Wechsel zu Operational
02h	Stop_Remote_Node: Wechsel zu Prepared
80h	Enter_Pre-Operational_State: Wechsel zu Pre-operational
81h	Reset_Node: Reset Knoten[1]
82h	Reset_Communication: Reset Kommunikation[2]

[1] On Power ON all the parameters in the whole Object Dictionary will have their values set.

[2] On Power ON only the parameters in the section Communication Profile of the Object Dictionary will have their values set.

20 LED states

greenLED = BUS State

red LED = ERR display

Annunciator	LED	Description	Cause of error	Addendum
Bus OFF	○	No connection to the Master [2]	Data transmission line break Incorrect baud rate Inverted data line	Observe combination with ERR LED If ERR LED is also OFF, please check power supply [3]
Bus flashing ca. 250ms	🟢	Connection to Master Pre-operational state		SDO communication
Bus flashing ca. 1sec	🟢	Connection to Master Stopped state		SDO communication not possible Only NMT commands
Bus ON	🟢	Connection to Master Operational state		PDO Transfer is active
ERR OFF	○	Device working normally		Observe combination with BUS LED
ERR flashing	🔴	Connection to Master interrupted	Combination with BUS status	BUS LED green, flashing or ON - is dependent on Object 1029h Error Behaviour
ERR ON	🔴	BUS OFF State	Short circuit on the Bus or Incorrect baud rate	
ERR +Bus flashing	🟢🔴	LSS-Mode	LSS Mode Global selected	Device is waiting for a LSS-command

The individual LED annunciators can of course also occur in combinations.

[2] The Master can be a PLC or a second communication partner.
[3] Operating voltage

21 Definitions

Explanation of Symbols:

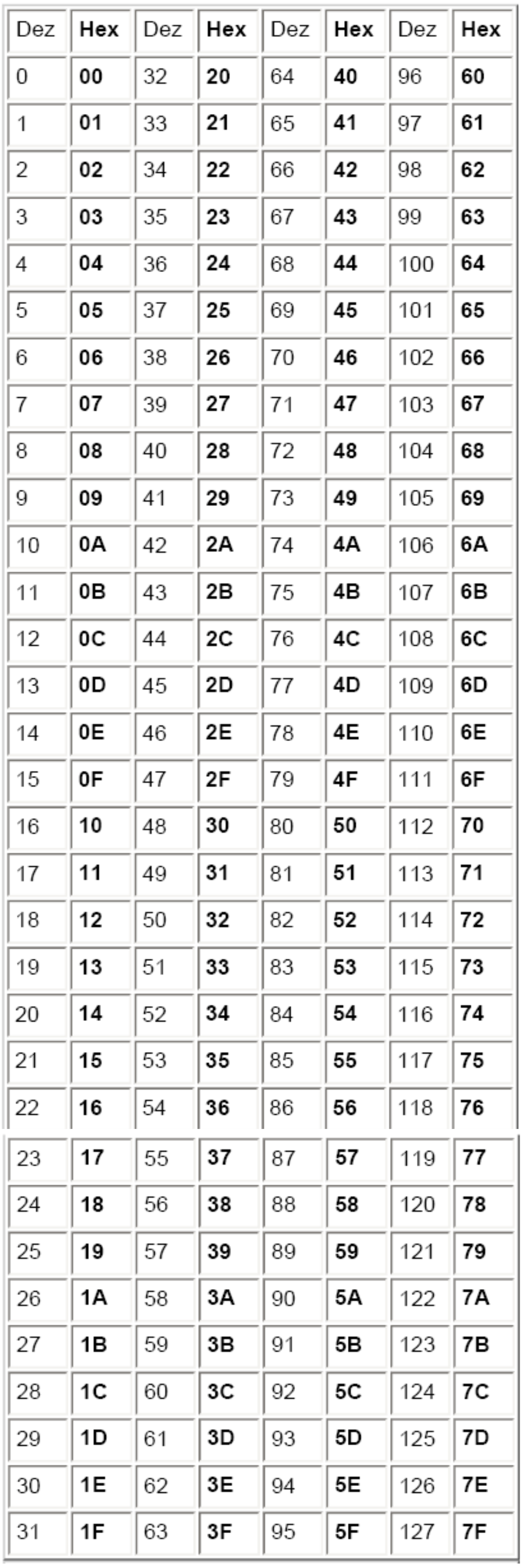
⚠	This symbol highlights those parts of the text to which particular attention must be paid. This is to ensure correct usage and to eliminate danger. This symbol provides important advice concerning the proper handling of the encoder. Non-observance of this advice can lead to malfunctions of the encoder or in the vicinity.
ℹ	This symbol refers to a special characteristic
🔌	Factory default setting of the parameter

22 Abbreviations used

CAL	CAN Application Layer. Application layer (layer 7) in the CAN Communication Model
CAN	Controller Area Network
CiA	CAN in Automation. International Association of Users and Manufacturers of CAN products
CMS	CAN Message Specification. Service element of CAL
COB	Communication Object. Transport unit in the CAN network (CAN message). Data will be sent over the network within a COB.
COB-ID	COB-Identifier. Unique identifier of a CAN message. The identifier defines the priority of the COB in the network.
DBT	Distributor. Service element of CAL, responsible for the dynamic allocation of identifiers.
DS	Draft Standard
DSP	Draft Standard Proposal;
ID	Identifier, see COB-ID
LMT	Layer Management. Service element of CAL, responsible for the configuration of the parameters in the individual layers of the communication model.
LSB	Least significant bit/byte
MSB	Most significant bit/byte
NMT	Network Management. Service element of CAL, responsible for the initialization, configuration and error handling in the network.
OSI	Open Systems Interconnection. Layer model for describing the function areas in a data communication system.
PDO	Process Data Object. Object for the exchange of process data.
RTR	Remote Transmission Request; Data request telegram.
SDO	Service Data Object. Communication Object, by means of which the Master can access the Object Dictionary of a node.
SYNC	Synchronization telegram. Stations on the Bus reply to the SYNC command by transmitting their process value.

23 Decimal-Hexadecimal Conversion Table

With numerical data, the decimal values are given as numerals with no affix (e.g. 1408), binary values are identified by the letter b (e.g. 1101b) and hexadecimal values with an h (e.g., 680h) after the numerals.



Dez	Hex	Dez	Hex	Dez	Hex	Dez	Hex
0	**00**	32	**20**	64	**40**	96	**60**
1	**01**	33	**21**	65	**41**	97	**61**
2	**02**	34	**22**	66	**42**	98	**62**
3	**03**	35	**23**	67	**43**	99	**63**
4	**04**	36	**24**	68	**44**	100	**64**
5	**05**	37	**25**	69	**45**	101	**65**
6	**06**	38	**26**	70	**46**	102	**66**
7	**07**	39	**27**	71	**47**	103	**67**
8	**08**	40	**28**	72	**48**	104	**68**
9	**09**	41	**29**	73	**49**	105	**69**
10	**0A**	42	**2A**	74	**4A**	106	**6A**
11	**0B**	43	**2B**	75	**4B**	107	**6B**
12	**0C**	44	**2C**	76	**4C**	108	**6C**
13	**0D**	45	**2D**	77	**4D**	109	**6D**
14	**0E**	46	**2E**	78	**4E**	110	**6E**
15	**0F**	47	**2F**	79	**4F**	111	**6F**
16	**10**	48	**30**	80	**50**	112	**70**
17	**11**	49	**31**	81	**51**	113	**71**
18	**12**	50	**32**	82	**52**	114	**72**
19	**13**	51	**33**	83	**53**	115	**73**
20	**14**	52	**34**	84	**54**	116	**74**
21	**15**	53	**35**	85	**55**	117	**75**
22	**16**	54	**36**	86	**56**	118	**76**
23	**17**	55	**37**	87	**57**	119	**77**
24	**18**	56	**38**	88	**58**	120	**78**
25	**19**	57	**39**	89	**59**	121	**79**
26	**1A**	58	**3A**	90	**5A**	122	**7A**
27	**1B**	59	**3B**	91	**5B**	123	**7B**
28	**1C**	60	**3C**	92	**5C**	124	**7C**
29	**1D**	61	**3D**	93	**5D**	125	**7D**
30	**1E**	62	**3E**	94	**5E**	126	**7E**
31	**1F**	63	**3F**	95	**5F**	127	**7F**

24 Glossary

Baudrate
The baud rate is the data transfer rate. It is linked to the nominal bit timing. The maximum possible baud rate is dependent on numerous factors that affect the transfer time on the bus. There is a significant connection between the maximum baud rate and the bus length and type of cable. In CANopen the various baud rates are defined between 10 Kbit/s and 1 Mbit/s.

CANopen
CANopen is a protocol based on CAN that was originally developed for industrial control systems. The specifications contain various device profiles as well as the framework for specific applications. CANopen networks are used in off-road vehicles, electronics on-board ships, medical equipment and the railways. The very flexible application layer together with the many optional features are ideal for tailor-made solutions. Furthermore, a wide variety of configuration tools are available. On this basis the user is able to define device profiles that are specific to his application. More information on CANopen can be found in the Internet at **www.can-cia.org**.

EDS file
The EDS (Electronic Data Sheet) is provided by the vendor/manufacturer of the CANopen device. It has a standardized format for describing the device. The EDS contains information concerning:

- Description of the file (name, version, date programme was generated etc.)
- General information about the device (manufacturer’s name and code)
- Device name and type, Version, LMT address
- Supported baud rates, as well as boot-up capability
- Description of the attributes of supported Objects.

Node number
Every device within a CANopen network can be identified by its node number (Node-ID). The permitted range for node numbers is from 1 to 127 and each may only occur once within a network.

Network Management
In a distributed system, various tasks arise that have to do with the configuration, initialization and control of stations on the network. This functionality is provided in CANopen by the defined service element »Network Management (NMT)«.

PDO
The Process Data Objects (PDOs) provide the actual transport means for transferring the process data (Application Objects). A PDO is transmitted by a Producer and can be received by one or more Consumers.

PDO Mapping
The size of a PDO can be up to 8 byte. It can be used to transport several Application Objects. PDO Mapping describes the definition of the structure of the Application Objects within the data field of the PDO.

SDO
The confirmed transfer of data, of any length, between two stations on the network occurs via Service Data Objects (SDOs). Data transfer takes place in the Client-Server mode.