Your Global Automation Partner

TURCK

CANopen Encoders
RM-101/102/109/121

Manual

1 Summary

These operating instructions do not contain information about the installation of the RM-101/102/109. You will find these in separate installation instructions.

Firmware and device file (EDS) versions
The firmware and EDS file versions at the date of release of this documentation are the following:

- Firmware Version **V1.06** with integrated bootloader
- Hardware Release **V1.01**
- EDS file V2.2 **RM101_102.eds**

2 Technical details and encoder characteristics

Supply voltage and power consumption
10…30 VDC
max. 30 mA (no load)

Hardware characteristics

Singleturn technology	Magnetic 2-axes Hall sensor
Singleturn resolution	16384 steps / revolution (14 bits)
Singleturn accuracy	+/-1° (over the whole temperature range)
Internal cycle time	1 ms
Multiturn technology	Magnetic revolution counter
Multiturn resolution	Maximum 2^{24} bits revolutions

Watchdog controlled Device
CANopen Interface Transceiver according ISO 11898
Function display and diagnostics via 2 LEDs

Supported standards and protocols
CAL-based Communication Profile for Industrial System
CiA Draft Standard 301 Communication Profile
CiA Draft Standard 305 Layer Setting Services
CiA Draft Standard 406 Device Profile for Encoders
CiA Draft Standard 302-3 Framework for CANopen Managers (Bootloader)

3 Supported standards and protocols

The CANopen encoders of the RM-101/102/109 series support the latest CANopen communication profile according to DS 301 V4.2 . In addition, device-specific profiles such as the encoder profile DS 406 V4.0 are available.
The additionally integrated LSS Services DS 305 allow assigning the node number and configuring the CAN bit rate directly via the CAN Bus. The LSS functionalities allow quick and easy detection and addressing of new, unconfigured devices.

The integrated bootloader according to DS 302-3 offers the flexibility for a later programming of devices in CAN/CANopen networks, as CANopen offers, with the SDO transfer, standardized mechanisms for transmitting large data volumes. The bootloader itself works independently of the application, as a minimal CANopen slave node according to CiA-301.

Bootloader characteristics

- NMT Error - Heart Beat Message by the bootloader is provided
- Emergency Services can be used with restrictions
- CANopen Layer Setting Services are supported
- CANopen bootloader is a SDO server
- PDO Service is not supported

Software update can be performed with a CANopen Master or a configuration tool via the user area of the Code-FLASH memory.

Operating modes

The following operating modes can be selected: Polled Mode, Cyclic Mode, Sync Mode. 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 a Flash memory, which had previously been saved. The following may be combined as PDO (PDO Mapping): position, speed, acceleration, temperature, as well as the status of the two limit switches.

Error status

A 2-color LED on the backside indicates the operating and error status of the CAN bus, as well as the condition of internal diagnostics. CANopen encoders are available both in solid shaft and hollow shaft versions; their protection level reaching IP 67 allows using them even in tough industrial environments.

CANopen Communication Profile DS 301 V4.02

CANopen represents a unified user interface and thus allows 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 profiles that have been standardized. The CAN in Automation (CiA) manufacturers and users group is in charge of the creation 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 reading and writing of data

CANopen uses four communication objects (COB) with different properties

- Process Data objects (PDO) for real-time data
- Service Data objects (SDO) for parameters and program transmission
- Network Management (NMT, Life-Guarding, Heartbeat)
- Predefined objects (for synchronization, 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 communication profile section, a section covering the device profile as well as a section specific to the manufacturer.

The Encoder Device Profile DS 406
This profile describes a vendor-independent mandatory definition of the interface with regard to encoders. The profile determines 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 three Object classes:

• **Encoder Class C1**
Describes all the basic functions that the encoder must include

• **Encoder 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.

• **Encoder Class C2**
Supports all Class C2 functions, thus including all mandatory C1 and C2 functions. High-resolution encoders are adapted in Class C3. Heartbeat and LSS functionality are also supported.

LSS services DS 305 V2.0
CiA DSP 305 CANopen Layer Setting Service and Protocol (LSS) were created to enable the following parameters to be read and changed through the network:
- Node address
- Baud rate
- LSS address

These features increase the "plug–and-play" compatibility of the device, while the configuration possibility has been significantly simplified. The LSS Master is responsible for configuring these parameters on one or more slaves on the network.

Data transmission
CANopen data is 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 highly dynamic exchange of real-time data (e.g. encoder position, speed, comparative position status) with a maximum length of 8 bytes. This data is transmitted with 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 bytes of position and 2 bytes 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).

4 Quick-Start Guide - First start-up

- Carry out the electrical installation (power supply, bus connection)
- Switch on the device
- If the display flashes red (1sec)
  Check the CAN connection (CAN_H,CAN_L, active CAN node must be available, termination, baud rate (default 250 kbits/s)
- Set the bus parameters via the LSS services or directly via the bus parameters

Setting the required baud rate	Object 2100h baud rate 250Kbit/s
Setting the node address	Object 2101h node address 0x3F
Setting the termination	Object 2102h termination 0x1
Saving the bus parameters	Object 2105h Save all Bus Parameters

- If setting is correct, the green LED flashes (Pre-operational Mode) and a bootup message with content 00 is sent
- Setting the asynchronous transmit parameters TPDO 1 Event timer for asynchronous operation
  Object 1800h, Subindex 05h (e.g. 10ms) or Object 6200h
- Saving the device parameters with Object 1010h,01h
- Device on/off cycle or Reset Node command (81h 00h)
- If setting is correct, the green LED flashes (Pre-operational Mode)
- NMT command Start-Operational (01h 00h)
- If setting is correct, the green LED must be on constantly (Operational Mode) and the position is output on the first TPDO (e.g. 0x1BF) with a 10 ms cycle

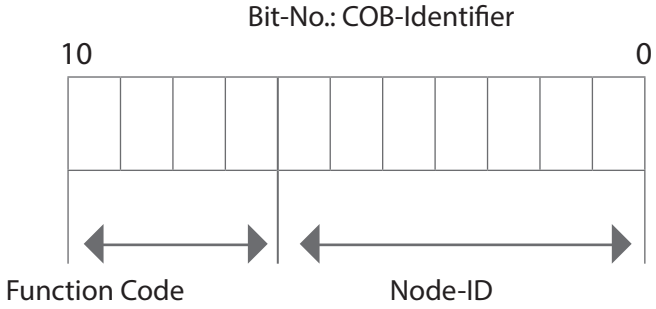
Setting further optional device parameters

- Setting of a possibly required scaling Object 6000h
- Setting the direction of rotation Object 6000h
  Example: 4= scaling is activated, 5 = scaling on and direction of rotation CCW
- Setting the measuring range Object 6001h,6002h
  Example: MUR = 360,TMR = 3600
  The encoder has now a resolution of 360 steps and will restart from zero after 10 revolutions
- Setting the zero point or the preset value Object 6003h e.g. 0
- Setting a working range Object 6401h,6402h
- Setting the times required for speed calculation Object 6031h
- Saving the device parameters with Object 1010h,01h

5 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 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.**



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

Broadcast objects of the generic pre-defined connection set

COB	Function	Resulting CAN-ID
NMT	0000_b	0 (000_h)
SYNC	0001_b	128 (080_h)
TIME STAMP	0010_b	256 (100_h)

Peer-to-Peer objects of the generic pre-defined connection set

Object	Function code (binary)	Resulting CAN-ID
EMERGENCY	0001_b	129 (81_h) - 255 (FF_h)
PDO1 (tx)	0011_b	385 (181_h) - 511 ($1FF_h$)
PDO1 (rx)	0100_b	513 (201_h) - 639 ($27F_h$)
PDO2 (tx)	0101_b	641 (281_h) - 767 ($2FF_h$)
PDO2 (rx)	0110_b	769 (301_h) - 895 ($37F_h$)
PDO3 (tx)	0111_b	897 (381_h) - 1023 ($3FF_h$)
PDO3 (rx)	1000_b	1025 (401_h) - 1151 ($47F_h$)
PDO4 (tx)	1001_b	1153 (481_h) - 1279 ($4FF_h$)
PDO4 (rx)	1010_b	1281 (501_h) - 1407 ($57F_h$)
SDO (tx)	1011_b	1409 (581_h) - 1535 ($5FF_h$)
SDO (rx)	1100_b	1537 (601_h) - 1663 ($67F_h$)
NMT Error Control	1110_b	1793 (701_h) - 1919 ($77F_h$)

Restricted, Reserved Object
Restricted CAN-IDs

CAN-ID	Used by COB
0 (000_h)	NMT
1 (001_h)	reserved
257 (101_h) - 384 (180_h)	reserved
1409 (581_h) - 1535 ($5FF_h$)	default SDO (tx)
1537 (601_h) - 1663 ($67F_h$)	default SDO (rx)
1760 ($6E0_h$)	reserved
1793 (701_h) - 1919 ($77F_h$)	NMT Error Control
2020 (780_h) - 2047 ($7FF_h$)	reserved

Process Data Transmission

The 3 PDO services PDO1 (tx) ,PDO2 (tx) … PDO4 (tx) are available for process data transmission. 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)
- The answer to a RTR request is not supported

6 PDO mapping

PDO mapping is the representation of the application objects (real-time data) from the object dictionary in the process data objects. The CANopen device profiles provide for every device type a default mapping that is suitable for most applications. So the default mapping for encoders simply represents the outputs according to their physical sequence in the transmit process data objects.
The current mapping can be read via corresponding entries in the object dictionary, the so-called mapping tables. The number of mapped objects, which are listed in the following, appears at the first position of the mapping table (Subindex 0). The tables can be found in the object dictionary under Index 0x1A00h for the TxPDOs.

PDO transmission type: Transmission Type (parametrizing)
The PDO transmission type defines how the transmission of the PDOs is triggered:

Transmission type	Cyclic	Acyclic	Synchronous	Asynchronous	RTR
0		X	X		
1-240	X		X		
241-251	reserved				
252			X		X
253				X	X
254				X	
255				event-driven	

Acyclic Synchronous
The PDOs of transmission type 0 work synchronously, but not cyclically. A device whose TxPDO is configured on transmission type 0, determines its input data when receiving the SYNC (synchronous process map) and then sends it, if the data corresponds to an event (e.g. an input change). So transmission type 0 combines the transmission trigger „event-driven" and the transmission (and possibly sampling) or processing moment „receiving SYNC".

Cyclic Synchronous
For transmission type 1-240, the PDI is emitted cyclically: after every „nth" SYNC (n=1...240). Since the transmission types cannot only be combined in the network, but also in a device, it is possible to define a fast cycle for positions (n=1), while the temperature data is transmitted in a slower cycle(e.g. n=10). The cycle time (SYNC rate) can be monitored (Object 0x1006), the device then reacts in case of SYNC failure according to the definition of the device profile and switches its outputs in error state.

Only RTR
Transmission types 252 and 253 apply for process data objects that are transmitted exclusively upon request by a remote frame. 253 is asynchronous, data is determined here permanently and transmitted upon request. This transmission type is generally not recommended, as the data retrieval is only supported incompletely by some CAN controllers. Moreover, as the CAN controllers answer automatically to remote frames (without prior request for up-to-date input data), the current condition of the polled data can possibly be questionable. For this reason, transmission types 252 and 253 are not supported.

Asynchronous
The transmission types 254 + 255 are asynchronous or also event-driven. For transmission type 254, the event is manufacturerspecific,
for 255 it is defined in the device profile. In the simplest case, the event is the change of an input value - so every value change is transmitted. The asynchronous transmission type can be coupled with the Event Timer and thus also provides input data when no event occurred currently. For TT 255, it must be noted that the inhibit time must be set > 100, otherwise a CAN Overrun error may occur, as the position on the last digit is constantly changing.

Variable mapping
Mostly, the default assignment of the process data objects (Default Mapping) is already sufficient to meet the requirements. It is possible, for specific application cases, to modify this assignment: So the encoders support the variable mapping, in which the application objects (output data) can be assigned freely to the PDOs. To that purpose, the mapping tables must be configured: As from CANopen Version 4, only the following procedure is allowed, and must be strictly observed:

Procedure for changing the mapping parameters
1 First erase the PDO (0x1800h, Subindex 1, set bit 31 to „1")
2 Set subindex 0 in the mapping parameter (0x1A00h) to „0"
3 Modify the mapping entries (0x1A00h, SI 1..8)
4 Set subindex 0 in the mapping parameter to a valid value. The device then checks the consistency of the entries
5 Create the PDO by entering the identifier (0x1800h Subindex 1 set bit 31 to 0)

Standard setting for the **Mapping of the transmit PDOs1-3:**

Mapping	TPDO1 1800h	TPDO2 1801h	TPDO3 1802h
Mapping object	1A00h	1A01h	1A02h
Entry	0x60040020	0x60040020	0x60300120
Process	Position	Position	Speed
Object	6004h	6004h	6030h
Subindex	00	00	01
Length	20h (32 Bit)	20h (32 Bit)	20h (32 Bit)
	Asynchronous	Synchronous	Asynchronous

Transmit TPDO 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	XX	XX	Position MSB

Default setting: Transmission Type = 254, Event Timer 0ms

Transmit TPDO2 (1801h) Position synchronous

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

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

Default setting: Transmission Type = 0x01, Syn-Mode , Sync at every pulse, Event Timer 0ms

Transmit TPDO3 (1802h) Speed asynchronous

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

Message	Byte 0	Byte 1	Byte2	Byte 3
3BF	Speed LSB	XX	XX	Speed MSB

The speed value is signed (Signed INT) .

Default setting: Transmission Type = 254, Event Timer 0ms

Service Data Transmission

SDO-COB-ID
The parameters stated in the object dictionary are read and written via service data objects (SDO). These SDOs are Multiplexed Domains, that is to say data structures of any size, provided with a multiplexer (address). The multiplexer comprises a 16-bit index and a 8-bit subindex that address the corresponding entries in the object dictionary.

The following identifiers are available as a standard for the SDO services: SDO (tx) (Encoder -› Master): 580h (1408) + node number
SDO (tx) (Encoder Master):
580h (1408) + node number
SDO (rx) (Master Encoder):
600h (1536) + node number

Restricted CAN-IDs

CAN-ID	used by COB
0 (000_h)	NMT
1 (001_h) - 127 ($07F_h$)	reserved
257 (101_h) - 384 (180_h)	reserved
1409 (581_h) - 1535 ($5FF_h$)	default SDO (tx)
1537 (601_h) - 1663 ($67F_h$)	default SDO (rx)
1760 ($6E0_h$) - 1791 ($6FF_h$)	reserved
1793 (701_h) - 1919 (77_h)	NMT Error Control
2020 (780_h) - 2047 ($7FF_h$)	reserved

The command byte describes the type of the SDO message

Command (Expedited Protocol)	Type	Function
22h	SDO(rx), Initiate Download Request	Send parameters to encoder (max. data length 4 byte)
23h	SDO(rx), Initiate Download Request	Send parameters to encoder (data length 4 byte)
2Bh	SDO(rx), Initiate Download Request	Send parameters to encoder (data length 2 byte)
2Fh	SDO(rx), Initiate Download Request	Send parameters to encoder (data length 1 byte)
60h	SDO(tx), Initiate Download Request	Acknowledgement of receipt by Master
40h	SDO(rx), Initiate Download Request	Request of parameters from encoder
43h	SDO(tx), Initiate Download Request	Parameters to Master, data length = 4 byte (unsigned 32)
4Bh	SDO(tx), Initiate Download Request	Parameters to Master, data length = 2 byte (unsigned 16)
4Fh	SDO(tx), Initiate Download Request	Parameters to Master, data length = 1 byte (unsigned 8)
80h	SDO(tx), Abort Domain Transfer	Encoder sends an error code to Master

Error Messages
An error message (Command 80h) replaces the normal acknowledgement (response) in case of an error.
The error message also includes communication protocol errors (e.g. wrong command byte) as object dictionary access errors (e.g. wrong index, write attempt on a read-only object, wrong data length, etc.).
The error codes are described in the CANopen profile (DS 301) and in the device profile (DSP 406).

Emergency error code classes

Error code	Description
$00XX_h$	Error reset or no error
$10XX_h$	Generic error
$20XX_h$	Current
$21XX_h$	Current, CANopen device input side
$22XX_h$	Current inside the CANopen device
$23XX_h$	Current, CANopen device output side
$30XX_h$	Voltage
$31XX_h$	Mains voltage
$32XX_h$	Voltage inside the CANopen device
$33XX_h$	Output voltage
$40XX_h$	Temperature
$41XX_h$	Ambient temperature
$42XX_h$	CANopen device temperature
$50XX_h$	CANopen device hardware
$60XX_h$	CANopen device software

7 Implemented LSS services DS 305

LSS Hardware Restrictions (LSS Address)
All LSS slaves must have a valid entry in the object dictionary for the Identity Object [1018h] in order to be able to carry out a selective configuration of the node. This Object is made of the following subindices:

- 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, a revision number and a serial number are set by the manufacturer. The LSS address must be unique in the network.

LSS Operating Restrictions
In order to ensure a perfect LSS functionality, all devices in the network must support the LSS services. There can only be one LSS master. All nodes must be started up wit the same baud rate.
LSS communication can only take place in "Stop" mode or in "Pre-Operational" mode.

8. Bootloader functionality DS302

An increasing number of CANopen devices need today bootloaders in order to replace the firmware in the field. Due to the large flash memory needs, the use of a complete CANopen stack is often not suitable for bootloaders. This is why Turck's CANopen bootloader has been developed with the focus on low memory requirements.
It supports only the CANopen services (SDO, NMT-Slave, Heartbeat Producer) and objects that are most essential for a bootloader. So, Turck's bootloader needs less flash resources and is nevertheless CANopen-compatible. Expansions such as LSS are available as options.

Bootloader highlights

- CANopen compatible
- Reduced resource requirements
- Optional LSS support
- Own, configurable node address (Standard 02)
- Configurable baud rate
- Producer Heartbeat
- Program Download
- Memory Range Erase
- Emergency Messages for errors while flashing
- Own EDS file

9 Electrical installation supply voltage and CANbus

This chapter gives information about the electrical installation, configuration and start-up of the CANopen encoder RM-101/102/109.

Electical installation
Switch the system off!
Please note that the entire system should be powered down during the electrical installation.

The electrical installation requires connectors or connecting cables (see the data sheet of the RM-101/102/109).

Bus connection

Terminal assignment

Interface	Type of connection	Cable (isolate unused wires individually before initial start-up)					
2	C**M	Signal:	+V	0 V	CAN_GND	CAN_H	CAN_L
		Cable color:	BN	WH	GY	GN	YE

Interface	Type of connection	M12 connector, 5-pin					
2	H1*51	Signal:	+V	0 V	CAN_GND	CAN_H	CAN_L
		Pin:	2	3	1	4	5

Connect the shielding and the housing of the encoder
Respect the maximum cable lengths in case of stub lines and for the total length of the CANbus.
If possible, mount all cables with traction relief.
Check the maximum supply voltage on the device.

Function and status LEDs

The device is provided with two LEDs to display its status and error messages
green LED = BUS Status
red LED = ERR Display
orange = visual combination of both LEDs

Display	LED	Meaning	Cause of error	Addition
Bus off	●	No connection to master [2]	Data line interrupted Wrong baud rate Swapped data line	Watch out for combination with ERR LED. If ERR-LED is also turned off, check power supply [3]
Bus Flashing about 250 ms	●	Connection to master Pre-operational status		SDO communication
Bus Flashing about 1 sec.	●	Connection to master Stopped status		SDO communication not possible. Only NMT commands
Bus on	●	Connection to master Operational status		PDO transfer is active
ERR off	● ●	Device runs error-free		Watch out for combination with BUS LED
ERR Flashing	●	Connection to master interrupted red LED (1000 ms)	Combination with bus-status	BUS LED flashes green or is on, depending on Object 1029h error behavior
ERR on	●	BUS OFF status	Bus short circuit or wrong baud rate	

The individual LED messages can also appear in combination
[2] Master can be PLC or 2nd communication partner
[3] Operating voltage

LED combinations during operation

Display	LED	Meaning	Cause of error	Addition
ERR Flashing		Red LED flashing Red LED flashes briefly, duration 3 sec.	Temperature overflow. Sensor monitoring. Single step error. Sensor LED current monitoring	Device on CAN bus. Connection to master OK + additional cause of error

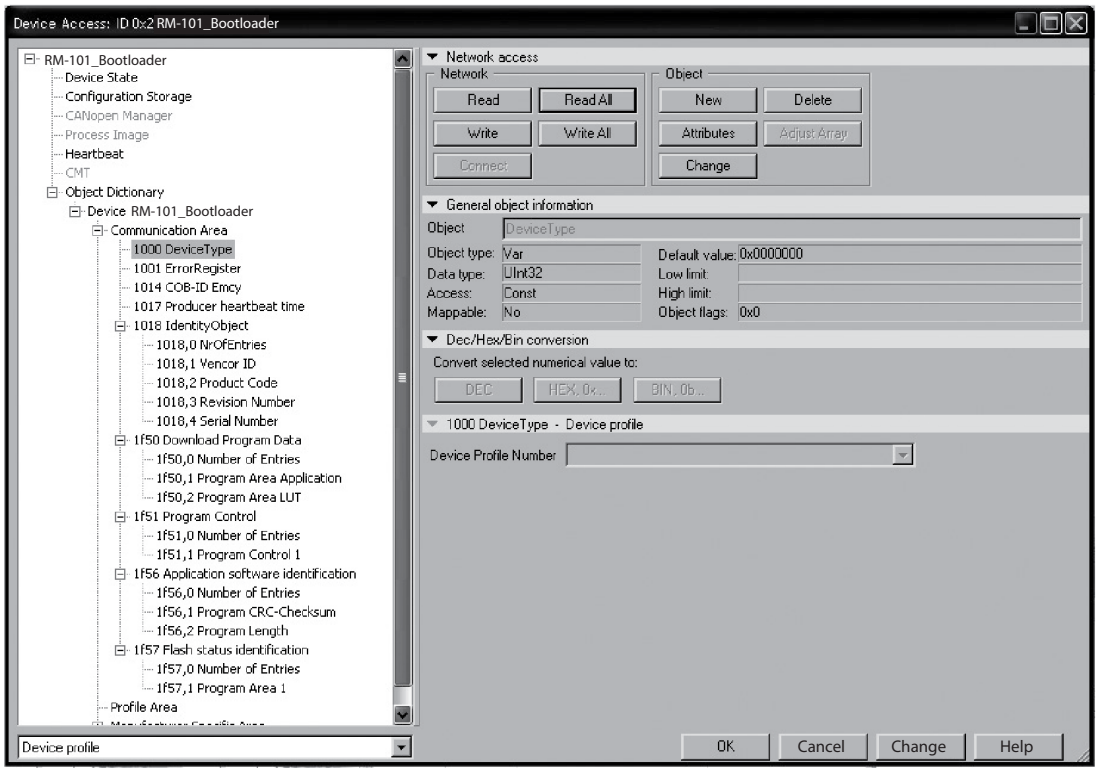
Error display upon switching on

Display	LED	Meaning	Cause of error	Addition
ERR + BUS Flashing		Alternating quick flashing of green and red LED	Data connection to sensor is faulty. Sensor is defective	Device needs to be sent to manufacturer for service
ERR + BUS Flashing		Alternating quick flashing of green and red LED (300 ms)	Watchdog error	Device needs to be sent to manufacturer for service
ERR Flashing		Connection to master interrupted		No CAN Bus available
Bus +Err flashing fast 50 ms		LSS Layer Service active. Global mode activated	Encoder waiting for configuration	LSS mode

Bootloader status upon switching on

Display	LED	Meaning	Cause of error	Addition
Both flashing approx. 250ms		Bootloader	Device is ready for firmware download	Device boots with addr. 02
Green flashing Red briefly on		Bootloader	Erase Program Area	Erasing process in progress
Green permanent Red permanent		Bootloader	Program is being flashed in the memory	Flashing operation in progress

10 Bootloader – general functionality and further start-up

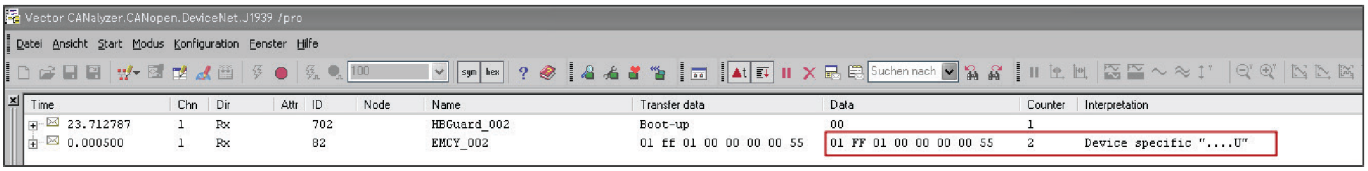


CANopen Manager tool CANsetter®

A configuration tool (here ProCANopen® of vector informatik) allows configuring and activating the bootloader. The most important objects are:

- 1F50h Download Program Data
- 1F51h Program Control
- 1F56h Application software identification

Behavior of devices without flashed firmware
After switching on the encoder, it sends a Boot-up message with address 0x02. If a subsequent Emergency Message with byte no. 8 as Code 0x55 is transmitted, an application is missing.



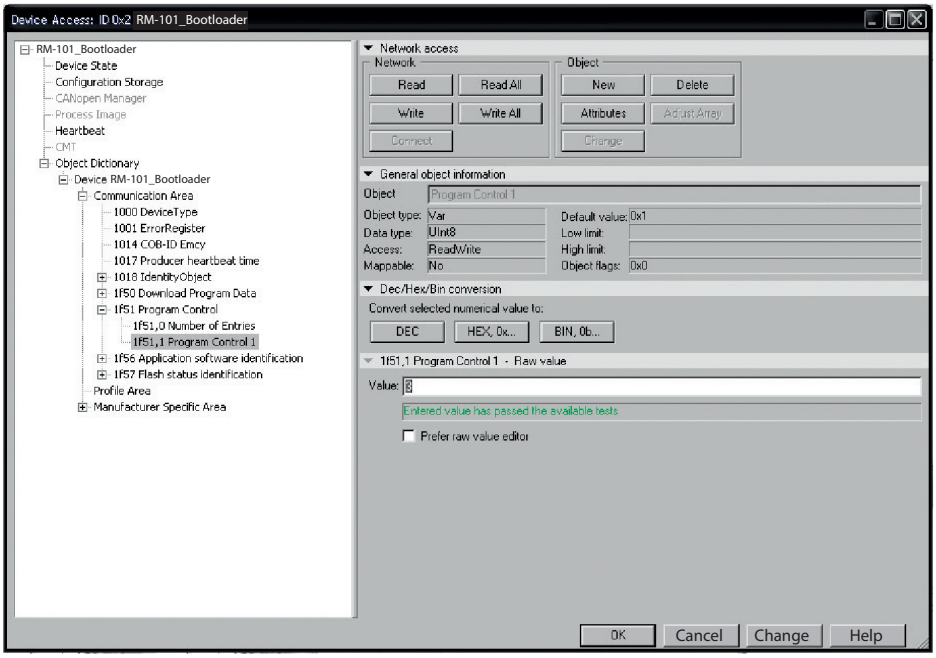
From this moment, the bootloader is ready for operation and can carry out SDO commands.

Status information of both LED´s

Display	LED	Meaning	Cause of error	Addition
Both flashing alternately about 250ms		Bootloader	Device is ready for firmware download	Device boots with addr. 02

11 Bootloader – Flashing new firmware Programming procedure

Erase program code area (Erase Customer Area)
Object 0x1F51 Program Control
Subindex 01 Program control 1
Entry : 03
Erase program area 0x3000 … 0x16FFF
Function: Erase



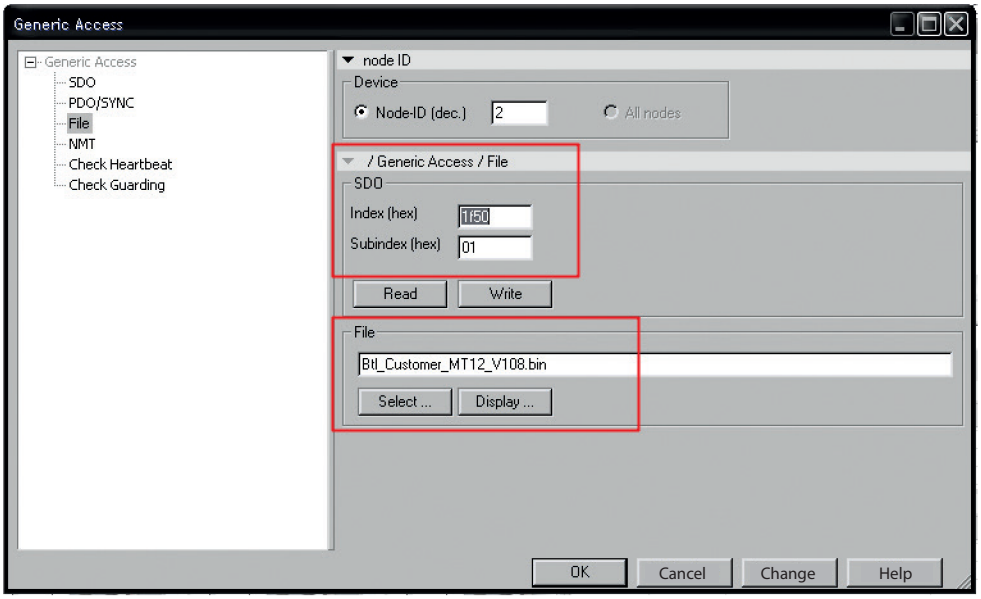
The erasing operation of the flash memory requires about 150 ms.

Display	LED	Meaning	Addition
Green flashing Red briefly, permanently		Bootloader, Erase Program	Erasing operation in progress

Prior to downloading the program itself, the area with this function must be erased.

Selection generic access -> Load data
Object 0x1F50 Program Download
Subindex 01 Program control 1
Entry : Programm download
Function : Write

Example: Btl_Customer_MT12_V106.bin



Status information of both LED´s

The writing process requires about 35 sec.

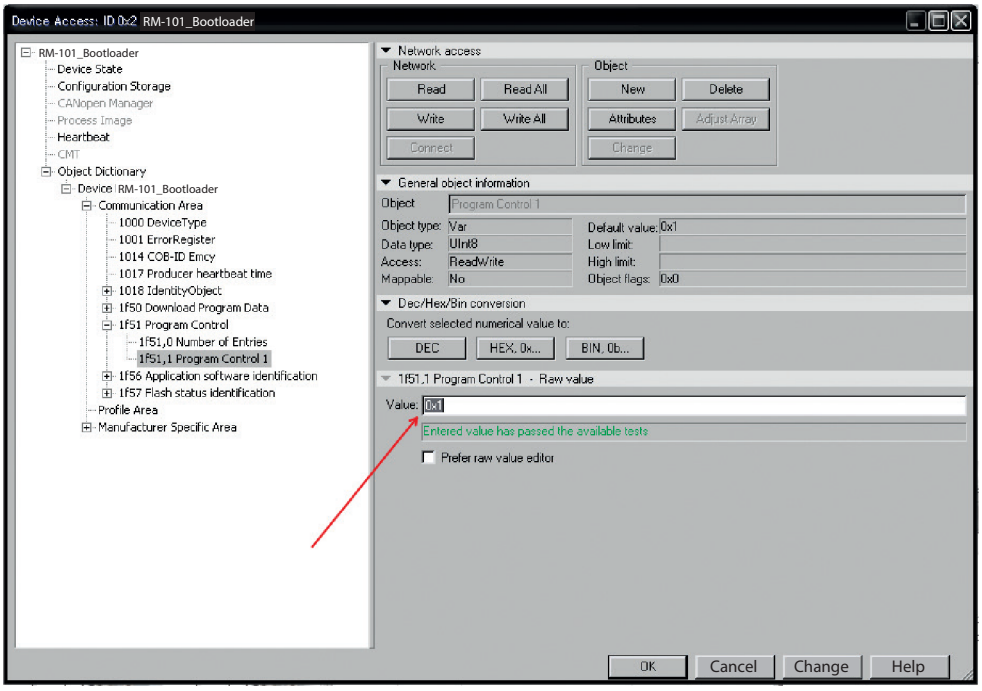
Display	LED	Meaning	Addition
Green permanent Red permanent		Bootloader flashing process	Program is being flashed in the memory Flashing process in progress

Extract of a program transmission

Transfer data	Data	Counter	Interpretation
	30 00 00 00 00 00 00 00	9725	Download Segment Rsp Tl
17 fb 6c e7 61 fa d1	00 17 FB 6C E7 61 FA D1	9726	Download Segment Rq. T0 C0 ".ûlçaúÑ"
	20 00 00 00 00 00 00 00	9726	Download Segment Rsp T0
df 15 8f a8 ee 4f a1	10 DF 15 8F A8 EE 4F A1	9727	Download Segment Rq. Tl C0 "ß.□¨îO¡"
	30 00 00 00 00 00 00 00	9727	Download Segment Rsp Tl
ee dd 05 d5 a8 ee df	00 EE DD 05 D5 A8 EE DF	9728	Download Segment Rq. T0 C0
	20 00 00 00 00 00 00 00	9728	Download Segment Rsp T0
1e 51 02 fc a1 5d 01	10 1E 51 02 FC A1 5D 01	9729	Download Segment Rq. Tl C0 ".Q.ü¡]."
	30 00 00 00 00 00 00 00	9729	Download Segment Rsp Tl
ef 16 8f d8 e7 4f a1	00 EF 16 8F D8 E7 4F A1	9730	Download Segment Rq. T0 C0 "ï.□Øç0¡"
	20 00 00 00 00 00 00 00	9730	Download Segment Rsp T0

Selection generic access -> Start program

Object 0x1F51 Program Control
Subindex 01 Program control 1
Entry : 0x01
Function: 1 Program Start



Status information of both LED's
Program start of the actual application

The encoder boots with the previously saved node address and the set baud rate of the CAN application (e.g. 0x3F).
The bootup message signals error-free behavior.

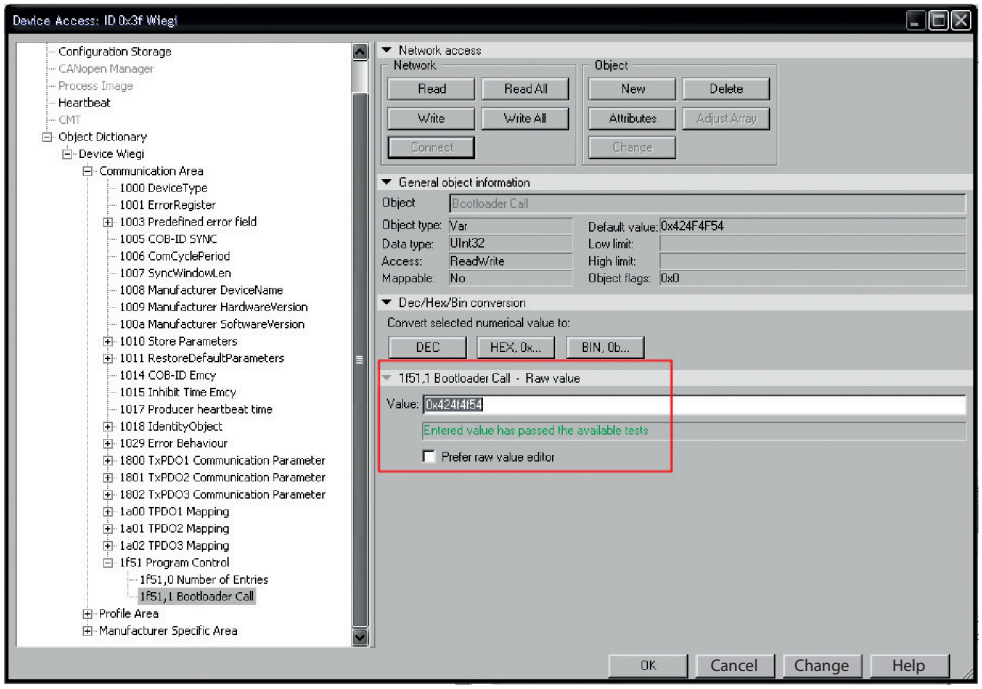
Green flashing Rot off		CANopen application	Pre-Operational	Bootup message

=> If the firmware baud rate is different from that of the bootloader, the red LED flashes.

Red flashing Green off		CANopen application	Pre-Operational	Bootup message

12 Calling the bootloader via CANopen Object

CANsetter -> Select encoder Object table
Object 0x1F51 Program Start
Switching over to the bootloader -> Entry in Object 1F51,1h
Entry: 0x424F4F54 [BOOT]



The device re-starts in bootloader mode and is ready for programming

Both flashing alternately about 250ms		Bootloader	Device is ready for firmware download	Device boots with addr. 02

The bootloader has again the control over the functionality of the encoder.

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

Exactly two conditions must be met for devices that are to be connected to a CANopen network - all devices must have the same baud rate and the node address must be unique within the network.

The condition for a use under LSS is that there is a 1:1 CAN connection with the device. A special dialogue mode then allows modifying the baud rate and the node address. COB-ID 0x7E5 is sent from the master to the slave, the slave answers with COBID 0x7E4.

LSS messages are always 8-byte long. Unused bytes are reserved and should be filled with 0. The LSS service can also modify the node address of a LSS slave. To that purpose, the LSS master sets the LSS slave back in the configuration mode. It then transmits its new node address to the LSS slave. The LSS slave answers to this to tell the LSS master whether this node address lies within the supported range.

After switching back to the operation mode of the LSS slave, the latter performs a software reset, in order to be able to configure the communication objects with its new node number. Standard CiA DSP-305 describes further LSS services.

Switch Mode Global

To complete the LSS service, the device is switched back from the LSS configuration mode to the Preoperational mode with the command „Switch Mode Global“:

Identifier	DLC	Data							
		0	1	2	3	4	5	6	7
0x7E5	8	0x04	mod	reserved					

Switch Mode Global Service

mod: new LSS mode
0 = switch on operation mode
1 = switch on configuration mode

The device automatically performs a new boot-up (Reset node), after which all new settings become valid.

Configure Bit-Timing

Identifier	DLC	Data							
		0	1	2	3	4	5	6	7
0x7E5	8	0x13	tab	ind	reserved				

Configure Bit Timing Service
tab: indicates the baud rate table to be used
0 = baud rate table defined in compliance with CiA DSP-305 1 … 127 = reserved
128 … 255 = can be defined by the user
Ind: Index in the baud rate table, in which the new baud rate for the CANopen device is stored.

Standardized baud rates according to CiA DS305:
Please note: The table has a reversed order as the standard baud rate table for Object 2101h.

Configure Node-ID
Assignment of a new node address

Identifier	DLC	Data							
		0	1	2	3	4	5	6	7
0x7E5	8	0x11	nid	reserved					

Configure Node-ID Service
nid: new node address for the LSS slave (values from 1 to 127 allowed)

Answer to Configure Node ID

Identifier	DLC	Data							
		0	1	2	3	4	5	6	7
0x7E4	8	0x11	err	spec	reserved				

Answer to Configure Node ID Service

err: Error code
0 = completed successfully
1 = node address invalid (only values from 1 to 127 allowed)
2 ... 254 = reserved
255 = special error code in spec
spec: manufacturer-specific error code (when err =255)

14 Default settings of the CANopen Objects at delivery

Communication parameters DS 406

Index (hex)	Name	Access	Standard value* *Subject to technical modifications
1000h	Device Type	RO	0x000B0196
1001h	Error Register	RO	0
1003h	Pre-defined Error Field	RO	0
1005h	COB-ID Sync	RW	0x80
1008h	Device Name	Const	RM-101/102
1009h	Hardware Version	Const	G202515
100Ah	Software Version	Const	BTL_RM-101/102_V106
1010h	Store Parameters	RW	„save“
1011h	Restore Parameters	RW	„load“
1014h	COB-ID Emergency	RW	0xBF
1017h	Producer heartbeat time	RW	0
1018h	Identity	RO	0x9c Vendor ID
1029h	Error Behavior	RW	
01h	Communication Error	RW	0 = Pre-Op Modus
02h	Device Profile Specific RW	1 = no change	
03h	Manufacturer-specific RW	1 = no change	
1800h	PD01 Communication Parameter	RW	
01h	COB-ID		180h + node number
02h	Transmission Type		254 (asynchronous)
03h	Inhibit Time		0 (step 100 µs)
05h	Event timer		0 (step 1 µs)
06h	Sync. Start Value		1
1801h	TPD02 Communication Parameter	RW	
01h	COB-ID		280h + node number
02h	Transmission Type		1 (synchron)
03h	Inhibit Time		0 (step 100 µs)
05h	Event timer		0 (step 1 µs)
06h	Sync. Start Value		1
1802h	TPD03 Communication Parameter	RW	
01h	COB-ID		380h + node number
02h	Transmission Type		254 (asynch)
03h	Inhibit Time		0 (step 100 µs)
05h	Event timer		0 (step 1 µs)
06h	Sync. Start Value		1
1A00h			
01h	1. Mapped Object		0x60040020
1A01h			
01h	1. Mapped Object		0x60040020
1A02h			
01h	1. Mapped Object		0x60300120

Device parameters DS 406

Index (hex)	Name	Access	Standard value
6000h	Operating Parameter Scalierung off /CW	RW	0
6001h	Measuring Units per Revolution	RW	2^{14} bits (0x4000)
6002h	Total Measuring Range	RW	2^{32} bits (0x10000000)
6003h	Preset value	RW	0
6004h	Position 32-Bit Value (Scaled)	RO	
600Bh	Position 43-Bit HighRes Raw value	RO	
600Ch	Position 32-Bit Raw Value	RO	
6030h	Speed Value	RO	
6040h	Acceleration value	RO	
6031H	Speed Parameter	RW	
	Speed Source Selector	RO	2
	Speed Integration Time	RW	100
	Speed Calc Multiplier	RW	1
	Speed Calc Divisor	RW	1
6041H	Acceleration Parameter	RW	
	Acceleration Source Selector	RO	2
	Acceleration Integration Time	RW	100
	Acceleration Calc Multiplier	RW	1
	Acceleration Calc Divisor	RW	1
6200h	Cyclic Timer (see TPDO1 Comm.Par)	RW	0
6400h	Working Area State	RO	0
6401h	Work area low limit 0,1h	RW	0
6402h	Work area high limit 0,1h	RW	Max.Resolution - 1
6500h	Operating Status	RO	0
6501h	Singleturn Resolution	RO	2^{14} bits (0x4000)
6502h	Number Revolutions Multiturn	RO	2^{29} bits 536870912
6503h	Alarms	RO	0
6504h	Supported Alarms	RO	0x8001
6505h	Warnings	RO	0
6506h	Supported Warnings	RO	0xC005
6507h	Profile and Version	RO	0x01060400
6508h	Operating Time	RO	
6509h	Offset Value	RO	
650Ah	Module Identification	RO	
650Bh	Serial Number	RO	
650Dh	Absolute Accuracy	RO	0x0A (10 Bit)
650EH	Device Capability	RO	2

Manufacturer area 2000h

Index (hex)	Name	Access	Standard value
2100h	Baudrate	RW	0xFF (250Kbit/s)
2101h	Node address	RW	0xFF (0x3F)
2102h	Termination	RW	1
2103h	Firmware Check sum	RO	
2105h	Save All Bus Parameters	RW	0x65766173
2150h	Temperature Sensor in 1/10 °C	RO	248 (example)
2162h	CRC-Value Raw Position	RO	

15 The DS301 communication profile

Object dictionary (OV)
All communication objects and all user objects are summarized in the Object dictionary (OD).
In the CANopen device model, the OD is the link between the application and the CANopen communication unit.
Every entry in the object dictionary represents an object and is identified by a 16-bit index. An index can include up to 256 subindexes.
This allows differentiating up to 65536 x 254 elements independently of the „11-bit identifiers". (Subindexes 0 and 255 cannot be used freely.)
The assignment of communication and device profile objects to a respective index is precisely defined in profiles; this defines, with the Object dictionary, a unique interface between the application and the communication towards the outside.
So, for example, every CANopen node in the network knows that the heartbeat interval can be found on Index 1017h, and every node or every configuration program has a read or write access to it.

Object dictionary areas (OD)

Index area	Use
0000	not used
0001-009F	Data types (special case)
00A0-0FFF	reserved
1000-1FFF	Communication profile
2000-5FFF	Manufacturer-specific area
6000-9FFF	Up to 8 standardized device profiles
A000-AFFF	Process images of IEC61131 devices
B000-BFFF	Process images of CANopen gateways according to CiA 302-7
C000-FFFF	reserved

Service Data Objects (SDO)
Offer a service for accessing the Object dictionary. Every CANopen device needs at least one SDO server that receives and processes SDO requests from other devices. With the default setting, messages intended for the SDO server of a device use the node number of the recipient + 0x600 as a COB-ID or as the „identifier" for the CAN message. The answer of the SDO server uses the node number of transmitter + 0x580 as the „identifier".
These relatively high, and therefore low-priority IDs are used to transmit entries in the OD. There is a protocol for this SDO transfer, which requires 4 bytes to encode the transmission direction, the index and the subindex. So, only 4 bytes of the 8 bytes of a CAN data field remain for the data content. For objects whose data content exceeds 4 bytes, there are two further protocols for fragmented SDO transfer.

Process data objects (PDO)
Unlike the low-priority and protocol data-overloaded SDO transfer, the process data objects (PDO) offer a faster possibility for transporting process data. The „identifiers" used for the PDO transfer are, for the default settings, in the COB range from 0x181 to 0x57F and have higher priority than the SDO messages.
They contain only usage data, therefore offering 8 bytes. The content of the usage data is defined by PDO mapping entries. These are objects in the OD that, as an assignment table, define which data will be transmitted via a PDO. This data is contained in other objects of the OD.
The values of several objects can be transmitted in a PDO, and the recipients of the PDO can use only parts of this data, according to their PDO mapping entries. When receiving a PDO, the data is written in other objects of the OD, for example in a digital output object, according to the mapping entries. PDOs can be transmitted cyclically, event-driven, upon request or in a synchronized way.

Network Management Objects (NMT)
They are used to manage the network. There are, among others, messages that cause a status change in a device or broadcast global error messages.
The Sync object sends or receives the high-priority SYNC massage used for synchronizing the nodes in the network and ensures, with the time-stamp object, a unified time in the network. There are many other objects available in the communication profile and device profiles.

16 Object dictionary (OD)

Object 1000h Device Type
This Object shows information about the device and the device profile.

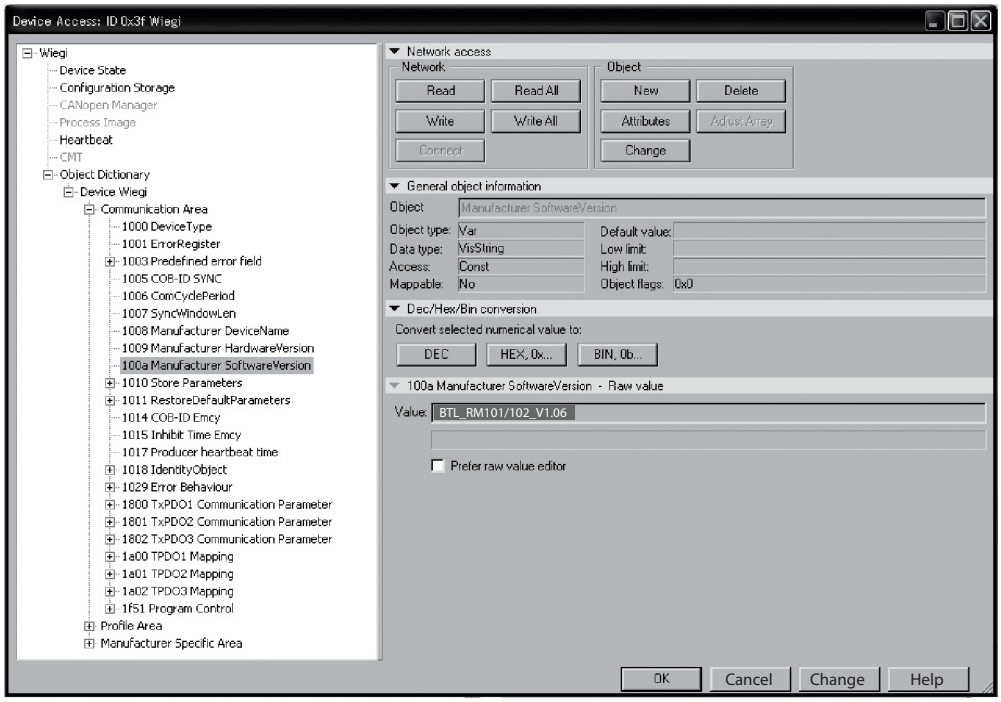
Bit 0-15 indicates the profile version of the device.
Bit 16-23 specifies the encoder type.

31 30	29 27	26 24	23 16	15 0
r(eserved)	SRDO	PDO	Encoder type	Device profile number
Additional information				196_h
MSB				LSB

The Turck encoder uses the type 0Bh

Field	Bit	Value	Description
Encoder type	16 to 23	00_h	Reserved
		01_h	Single-turn absolute rotary encoder
		02_h	Multi-turn absolute rotary encoder
		03_h	Single-turn absolute rotary encoder with electronic turn-count
		04_h	Incremental rotary encoder
		05_h	Incremental rotary encoder with electronic counting
		06_h	Incremental linear encoder
		07_h	Incremental linear encoder with electronic counting
		08_h	Absolute linear encoder
		09_h	Absolute linear encoder with cyclinc coding
		$0A_h$	Multi-sensor encoder interface
		$0B_h$	Multi-turn absolute rotary HR encoder with electronic turn-count
		$0C_h$ to FF_h	Reserved
PDO	24 to 26	000_b	Default PDO mapping
		001_b to 110_b	Reserved
		111_b	Manufacturer-specific PDO mapping
SRDO	27 to 29 000b SRDO mapping not supported		
		001_b to 101_b	Reserved
		110_b	Default SRDO mapping
		111_b	Manufacturer-specific SRDO mapping
r(eserved)	30 to 31	00_b	Reserved for compatibility reasons

Object 100Ah Manufacturer Software Version
Information about the currently implemented software e.g. V1.06



Object 1010h Store CANopen Parameters
Command „save“ under subindex 1h (save all Parameters) stores the parameters in the non-volatile memory (FLASH MEMORY).
This sub-item stores all communication objects, application objects and manufacturer-specific objects.
This operation requires an average time of about 30-40 ms.

In order to avoid any unintentional storage, the command is only carried out when the string „save“ is entered as code word in this subindex.
Parameter „save“
Command bytes: 23 10 10 01 73 61 76 65
Answer: 60 10 10 01 00 00 00 00 when saving was successful

Object 1011h: Load CANopen factory default values
The command „load“ under subindex 1h resets all parameters to their standard values. In order to avoid any unintentional loading of the standard values, the command is only carried out when the string „load“ is entered as code word in this subindex.

Parameter „load“
Command bytes: 23 10 11 01 6C 6F 61 64
Answer: 60 10 10 01 00 00 00 00

Object 1017h: Producer Heartbeat Object
The Producer heartbeat time defines the heartbeat cycle. If this function is not required, input a 0 time. This function is activated with a time starting from 1 ms to max. 65535ms.

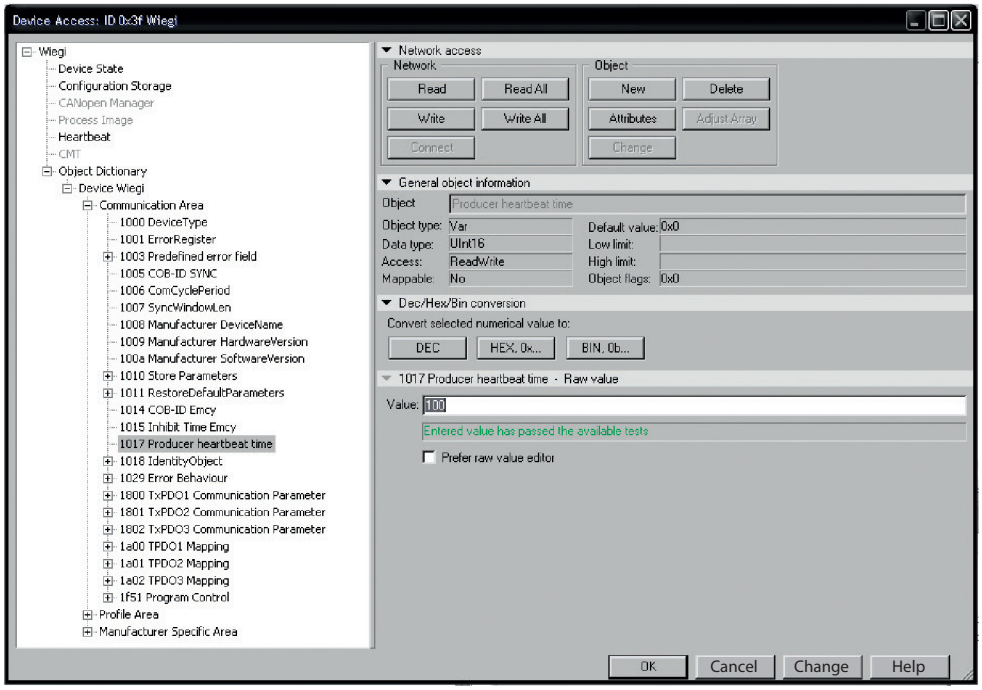
Object Description

Index	**1017h**
Name	Producer Heartbeat Time
Object Code	VAR
Data Type	UNSIGNED 16
Category	Conditional: Mandatory if guarding not supported

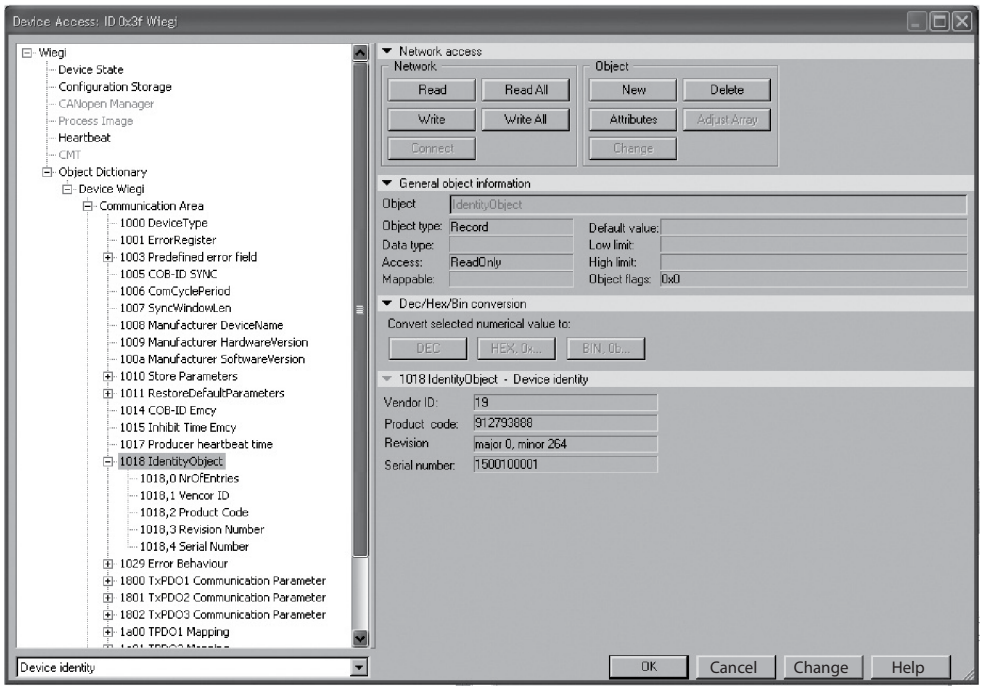
Entry Description

Access	rw
PDO Mapping	No
Value Range	UNSIGNED 16
Default Value	0

A „Heartbeat producer“ transmits the message cyclically according to the set time. The content of the data byte corresponds to the CAN node status. (Pre-op, Operational, Stopped).
Heartbeat is used to monitor the node. If this is deactivated, guarding is used for monitoring. Here, in this example, a value of 100ms is set for the Heartbeat.



Object 1018h: Identity Object
Information about the manufacturer and the device:



1018 RECORD Device – Identification read only

Sub-Index 0h : Number of subindexes“
returns the value 4
Sub-Index 1h: „read“ only
returns the Vendor-ID (000000009ch) Turck
Subindex 2h: returns the Product Code
(e. g. 0x36682121 CANopen encoder)
Sub-Index 3h: „read” only
returns the software revision number (e. g. 108)
Sub-Index 4h: „read” only
returns the 10-digit serial number of the encoder

17 Emergency messages

Emergency Objects arise in case of 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 as long as the error is active. Once the error is eliminated, a new Emergency Object with the content 0 (Error Reset or No Error) is generated and transmitted over the bus.
„Emergency"-type messages are used to signal the failures of a device. The Emergency telegram transmits a code that identifies the error univocally (defined in the communication profile CiA 301as well as in the respective device profiles CiA 40x).

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

Emergency Object Data

Example of a message for overtemperature:

Transfer Data	00	42	09	80	56	20	50	2E

[Errcode]	4200	Temperature threshold value of the encoder exceeded
[Error Register]	09	Error register
[ManufacturerSpecific1]	80	Error register
[ManufacturerSpecific2]	56	Momentary temperature
[ManufacturerSpecific3]	20	Current threshold value lower range
[ManufacturerSpecific4]	50	Current threshold value higher range
[ManufacturerSpecific5]	2E	Versions register

The behavior in case of an error is described in Object 1029h Error Behavior.

Implemented error codes
The table shows an extract of the available error codes. The Emergency message is sent autonomously by every CANopen device. The current version of the CANopen communication profile also allows switching off the transmission of an Emergency message.

Error Code	Error register	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Remarks
FF00	01	13	05	45	00	00	Sensor Sync. Failure
		81					Error Mask Register
		45					Error Register
4200	01	20	81	A8			System Temperature Error
		81					Actual Temp Register
		A2					Temperature Upper Reg
8110	11	00					CAN Overrun Error
8120	11	00					CAN Passive Error Mode
8130	01	00					LifeGuard or Heardbeat Error
FF01	01	00	XX	XX	XX	XX	Bootloader no Firmware
5000	01	E1					Write Error in Dataflash
5051	01	03					Error Multiturn

Object 1029h Error Behavior
If a serious error is detected, 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 error
- 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 Specific
- internal error

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

Value range 8 bits

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

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

18 CANopen Communication Profile DS 301 Supported Objects

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
1010	VAR	RW	Store parameters (Device Profile)	O	Unsigned32
1011	VAR	RW	Store parameters (Device Profile)	O	Unsigned32
1014	VAR	RO	COB_ID Emcy	O	Unsigned32
1015	VAR	RW	Inhibit Time Emcy	O	Unsigned32
1017	VAR	RW	Producer Heartbeat time	O	Unsigned32
1018	RECORD	RO	Identity Object	O	Unsigned32
1029	ARRAY	RW	Error Behavior	O	Unsigned32
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
1F51	ARRAY	RW	Program Control	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

19 CANopen encoder device profile DS 406 Supported 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	Unsigned32
600B	VAR	RO	Position HighRes Raw Value	M	Unsigned64
600C	VAR	RO	Position Raw Value	M	Unsigned32
6030	ARRAY	RO	Speed Value	O	Unsigned16
6031	ARRAY	RW	Speed Calculation Parameter	M	Unsigned16
6040	ARRAY	RO	Acceleration Value	O	Signed16
6041	ARRAY	RW	Accel Calculation Parameter	M	Unsigned16
6200	VAR	RW	Cyclic Timer	M	Unsigned16
6400	ARRAY	RO	Working Area state	O	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	Unsigned16
6504	VAR	RO	Supported alarms	M	Unsigned16
6505	VAR	RO	Warnings	M	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
650D	VAR	RO	Absolute Accuracy	M	Unsigned8
650E	VAR	RO	Device Capability	M	Unsigned8

20 Manufacturer-specific Objects DS 406 Area 2000h

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
2150	VAR	RO	Temperature Sensor in 1/10°C	O	Unsigned16
2162	VAR	RO	Raw positon data CRC16	O	Unsigned16
			unsupported Objects		

21 Speed output configuration

The rotational speed of the encoder shaft is determined as the value difference between two physical (unscaled) position values with a fixed time interval of 5ms … 2000ms -> Object 6031h

To adapt the speed measurement to the concerned application, the operator can use 3 parameterizable objects in the manufacturer-specific area. In case of high speeds, the integration time of the measurement can be reduced to achieve accordingly high dynamics. The number of mean values especially affects the dynamics of the measurements; it must be determined according to the application.

Speed measurement accuracy
The accuracy of the measurement depends mainly on the following parameters:

- actual speed
- temporal speed change (internal dynamics)

Object 6031h: Speed Integration Time
The speed is calculated according to the following formula:

$$\text{Speed} = \frac{\text{Position change}}{\text{Integration time}} \times \text{Calculation Multiplier} / \text{Calculation Divisor}$$

6031 Speed Parameter		Ranges	Default
Subindex 0	Number of channels	4	4
Subindex 1	Speed Source Selector	2	2
Subindex 2	Integration Time Value	5 … 2000	100
Subindex 3	Calculation Multiplier	1	1
Subindex 4	Calculation Divisor	1	1

A variable integration time of 5...2000ms can be input under subindex 6031h,Sub2. A parameter is available under Object 6031h, Sub3 Speed Calculation Multiplier as a multiplier for a unit factor, or a divisor can be programmed under Object 6031h, Sub4. The Speed Source Selector 6031h,Sub1 is set to 2 (600Ch Raw Position) as a standard and cannot be modified.

Object 6031,sub3 /4 only allows influencing the speed output with the unit [unit/sec], the output in RPM is not parameterizable. Speed output occurs either in RPM or as a number of steps per second and is set in Object 6000h Bit 12.

22 Objects of Encoder Profile DS 406

Object 6000h Operating Parameters

Code sequence: 0 = increasing for clockwise rotation (cw)
1 = increasing for counter-clockwise rotation (cw)
Scaling: 0 = disable
1 = enable observe Object 6001, 6002
Speed Format: 0 = revolutions/min
1 = units/second
Startup Mode: 0 = Boot-up after Pre-operational
1 = Boot-up after operational

Default setting: 0x0

Bit	Function	Bit = 0	Bit = 1	C1	C2
0	Code sequence	CW	CCW	m*	m*
1	Commissioning Diagnostic Control	Disabled	Enabled	o	o
2	Activate scaling	Disabled	Enabled	o	m
3	Measuring direction	Forward	Reward	o**	o**
4..11	Reserved for further use				
12	Universal Scaling Function	Disabled	Enabled	o	o
13	Speed Format	RPM	Units/sec	o	o
14	Start-up automatic in OP-Mode	Disabled	Enabled	o	o
15	Event Mode Position	Disabled	Enabled	o	o

*m = function has to be supported **o = optional

Object 6001h: Measuring units per revolution (MUR) (resolution)
This parameter adjusts the desired resolution per revolution. The encoder internally calculates the corresponding scaling factor. The calculated scaling factor MURF (which multiplies the physical position value), is calculated by using the following formula:
MURF = Measuring units per revolution (6001h) / phys. resolution singleturn (6501h)

Byte 0	Byte 1	Byte 2	Byte 3
2^7... 2^0	2^{15}...2^8	2^{23}...2^{16}	2^{31}...2^{34}

value range: **1....max. physical resolution** (16384) 2^{14} bits
Only valid when scaling (6000h bit2) is activated.

Object 6002h: Total Measuring Range (TMR)
This parameter adjusts the total measurement range of singleturn and multiturn. The maximum physical resolution affected with a factor. The factor is always < 1. After the scaled overall position of measurement units, the encoder resets to zero (with limitations).

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
2^7... 2^0	2^{15}...2^8	2^{23}...2^{16}	2^{31}...2^{34}

Value range: 1....max. physical resolution (4294967296) 2^{32} bits

Note:
If TMR/MUR is altered, the ratio between TMR and MUR is also checked. An attempt to set a value for TMR that would result in an invalid ratio will trigger an error message and the new value will be dismissed. The old value will remain in the encoder. Only valid when scaling (6000h bit2) is activated.

Object 6003h: Preset value
The position value of the encoder is set to the entered preset value.
This will align the zero position of the encoder with the machine zero point.

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
2^7... 2^0	2^{15}...2^8	2^{23}...2^{16}	2^{31}...2^{34}

Value range: 1....max. physical resolution (4294967295 (2^{32}) - 1 bits
Default setting: 0

Note:
When entering the preset value, the system checks automatically whether the point lies within the activated scaling or total measurement range. If this is not the case, the entry is rejected.

Object 6004h: Position value unscaled or scaled
The encoder outputs the current position value (possibly calculated with a scaling factor).

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
2^7... 2^0	2^{15}...2^8	2^{23}...2^{16}	2^{31}...2^{34}

Value range: 1....max. physical resolution (4294967296) (2^{32}) bits
The TMR/MUR ratio is active when scaling is activated, otherwise, the sensor outputs its 32-bit raw position.

Object 600Bh: Position value High-Resolution Raw Data
In addition to Object 6004h, the position data can be output as High-Resolution raw data. The data is transmitted as a 64-bit value. This value is used internally as a basis for calculation. The encoder outputs the current original position value with max. 43 bits directly from the sensor.

Data content:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
2^7…2^0	2^{15}…2^8	2^{23}…2^{16}	2^{31}…2^{34}	2^{39}…2^{32}	2^{47}…2^{40}	2^{55}…2^{48}	2^{63}…2^{56}

Value range: 0....maximum physical resolution 8796093022208 (2^{43}) bits for MT

Object 600Ch: RAW position value data
In addition, the position data can be output as raw data. The data is transmitted as a 32-bit value, a CRC for the portion data can also be added to the mapping*. Object 2162h may be used to that purpose.
The encoder outputs the current original position value directly from the sensor, without scaling.

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
2^7…2^0	2^{15}…2^8	2^{23}…2^{16}	2^{31}…2^{34}

Value range: 0....maximum physical resolution 4294967295 (2^{32})-132 bits for MT

Object 6030h: Speed Value
The encoder outputs the current calculated speed (possibly with a scaling factor) as a signed 32-bit value. The speed depends on the settings of Object 6031h. These values affect the calculation and the result.

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
2^7…2^0	2^{15}…2^8	2^{23}…2^{16}	2^{31}…2^{34}

Value range: 0....+/- maximum speed 12,000 RPM (signed value)

Note:
For values > 6,000 RPM a warning message is generated and the warning bit „Speed exceeded bit 0“ is set in Object Warnings 6505h. Parameters also affecting this object are mentioned in Object 6031h.

Object 6040h: Acceleration Value
The encoder outputs the current calculated acceleration (signed) as a signed 32-bit value. The acceleration is calculated from the speed changes and thus also depends indirectly on the settings of Object 6031h.
Irrespective of this, the settings of Object 6041h become decisive. All settings of these values affect the calculation and the result.

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
$2^7 \ldots 2^0$	$2^{15} \ldots 2^8$	$2^{23} \ldots 2^{16}$	$2^{31} \ldots 2^{34}$

Value range: 0.... +/- maximum acceleration
Negative values mean a negative acceleration (the speed decreases)

Object 6041h: Acceleration Parameter

6041 Accel Parameter		Range	Default
Subindex 0	Number of channels	4	4
Subindex 1	Speed Source Selector	2	2
Subindex 2	Integration Time Value	5 … 2000	100
Subindex 3	Calculation Multiplier	1	1
Subindex 4	Calculation Divisor	1	1

An average acceleration a is the change in time of speed v and can thus be described formally from the derivative of speed versus time t, this calculation results in an average acceleration from the difference between speeds Δv at 2 different moments Δt (t2-t1).

Note:
$a = \Delta v / \Delta t$ or $a = v2 - v1 / t2 - t1$

Object 6200h: Cycle timer
Defines the cycle time with which the current position is output using 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 Subindex (05h) in the first Transmit PDO.

Data content:

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

Value range: 0 ... FFFFh (65535) gives a cycle time in milliseconds

Object 6500h: Display Operating Status
This Object displays the status of the programmed settings of Object 6000h.

Data content:

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

Data content: see Object 6000h

Object 6502h: Number of Multiturn revolutions
This Object allows reading the current number of revolutions. The value depends on the encoder type and any value from 4096 (12 bits) up to (32 bits) could occur. This value only affects the number of revolutions. It does not affect the resolution.

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
$2^7 \ldots 2^0$	$2^{15} \ldots 2^8$	$2^{23} \ldots 2^{16}$	$2^{31} \ldots 2^{34}$

Value range: 1 …4294967296
Default setting corresponds to 262144 for MT

Object 6503h: Alarms
In addition to the errors that are signaled 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
$2^7 \ldots 2^0$	$2^{15} \ldots 2^8$

Bit no.	**Description**	**Value = 0**	**Value = 1**
Bit 0	Position error	Position value valid	Position error
…			
Bit 15	Sensor error	No error	Error

In both cases, if an alarm occurs, an emergency message (ID=80h+node number) is sent together with the error code.

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

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 is set to 1 for as long as the tolerance is exceeded or the warning applies.

Data content:

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

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

Bit no.	Description	Value = 0	Value = 1
Bit 0	Overspeed	none	Exceeded > 6000 rpm
Bit 1	not used		
Bit 2	Watchdog Status	System OK	Reset carried out
…			
Bit 14	Temperature error	Temperature OK	Overtemperature
Bit 15	Internal memory error	ok	Error*

* When Bit 15 is active, an emergency message (ID=80h+node number) is sent together with Error code 5200h (Device Hardware).

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 2 values
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.

Name	Bit	Value	Definition
out of range	0	0b	Position between minimum and maximum value (refer to module identification, object 650Ah)
		1b	Position (refers to minimum and maximum value in module identification, object 650 Ah) is reached or exceeded
range overflow	1	0b	No range oveflow
		1b	Position is higher than the position value set in work area high limit (object 6402h)
range underflow	2	0b	No range underflow
		1b	Position is lower than the position value set in work area low limit (object 6401h)

Object 6401h: Working Area Low Limit 2 values
Object 6402h: Working Area High Limit 2 values
These two parameters configure the working area. The state inside and outside this area can be signaled 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
$2^{7}…2^{0}$	$2^{15}…2^{8}$	$2^{23}…2^{16}$	$2^{31}…2^{34}$

Value range: 1....maximum physical resolution 4294967295 (2^{32}) bits - 1
2 values are available both for the upper and for the lower area.

Object 6500h: Operating Status
This Object reflects the current state of the encoder.

Data content:

Byte 0	Byte 1
$2^{7}…2^{0}$	$2^{15}…2^{8}$

Range of values: see Object 6505h

Object 2100h: Baud rate
This Object allows modifying the baud rate via software. As a standard, the value is set to FFh, so the setting of the LSS shows a reconfigured node. If the value is set between 1..9 and the parameter is saved using the Object 2105h Save All Bus Parameters, then, on the next Power ON or reset node, the device will boot up with the modified baud rate, and the currently set baud rate is displayed.

Data content:

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

Value range 1 ...8 (see table Baud rate at start-up)
Default Setting: 0x05h 250Kbit/s

Note:
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 remain unchanged.

Object 2101h: Node address
This Object allows modifying the node address via software. As a standard, the value is set to 0xFFh, so the setting of the LSS shows a reconfigured node. If the value is set between 1..127 and the parameter is saved using the Object 2105h Save All Bus Parameters, then, on the next Power ON or reset node, the device will boot up with the modified node address, and the currently set address is displayed.

Data content:

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

Value range 1 ...127 or 1..7Fh
Default Setting: 0x3Fh Address 63

Note:
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 remain unchanged.

Object 2102h: CAN bus termination OFF / ON
This Object allows setting the bus termination via software. As a standard, the value is set to 10, which means that the encoder is terminated.

Data content:

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

Value range 0..1
*for devices with cable outlet and CAN connection = 1

Object 2103h: Firmware Flash version
This object is used to display the current firmware version as a 16-bit hexadecimal value. This value allows checking whether the device is up-to-date.

Data content:

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

Value range up to FFFFh
Example: C47Ah current firmware

Object 2105h: Save All Bus Parameters
This parameter stores the desired bus parameters (Object 2100h, 2101h, 2102h) permanently in the Flash memory. This object is an additional protection against unintentional baud rate or node address modifications.
Only the targeted storage using parameter „save“ (hexadecimal 0x65766173) saves the bus parameters baud rate, node address and termination permanently.

Data content:

Byte 0	Byte 1	Byte 2	Byte 3
$2^7 \ldots 2^0$	$2^{15} \ldots 2^8$	$2^{23} \ldots 2^{16}$	$2^{31} \ldots 2^{34}$

Value range: „save“ in hexadecimal 0x65766173
Command bytes : 23 10 10 01 73 61 76 65
Answer: 60 10 10 01 00 00 00 00 when saving was successful

Object 2150h : Actual temperature Position-Sensor *
This Object is used to display the current temperature inside the sensor as a signed 16-bit hexadecimal value. This value allows determining the momentary device temperature.

Data content:

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

Value range up to 00…FFFFh
Example: 0x103 corresponds approximately 25.9°C

* The temperature value can be mapped as a 16-bit value in the process data, where it will be updated every 6 minutes. Accuracy is ± 2°C, the measurement takes place within the encoder electronics

Object 2162h : RAW-Data CRC16
The current Object 2160h Position raw data generates a standard CRC16.
CRC-CCITT (CRC-16) x16 + x12 + x5 + 1 (Polynomial 0x1021)
The implementation performs a polynomial division when the start value used is 1021h. This corresponds to a polynomial division if the first n bits of the data flow are complemented. A start value different from 0000... should be preferred, since otherwise missing bits within leading zeros would not be detected in the data flow (just like in a usual division, the leading zeros are not taken into consideration for a polynomial division).

Start value (seed value) used here = 0x1021
The polynomial is displayed as a 16-bit hexadecimal value.
This value is used to check the momentary raw position data of the device.

Data content:

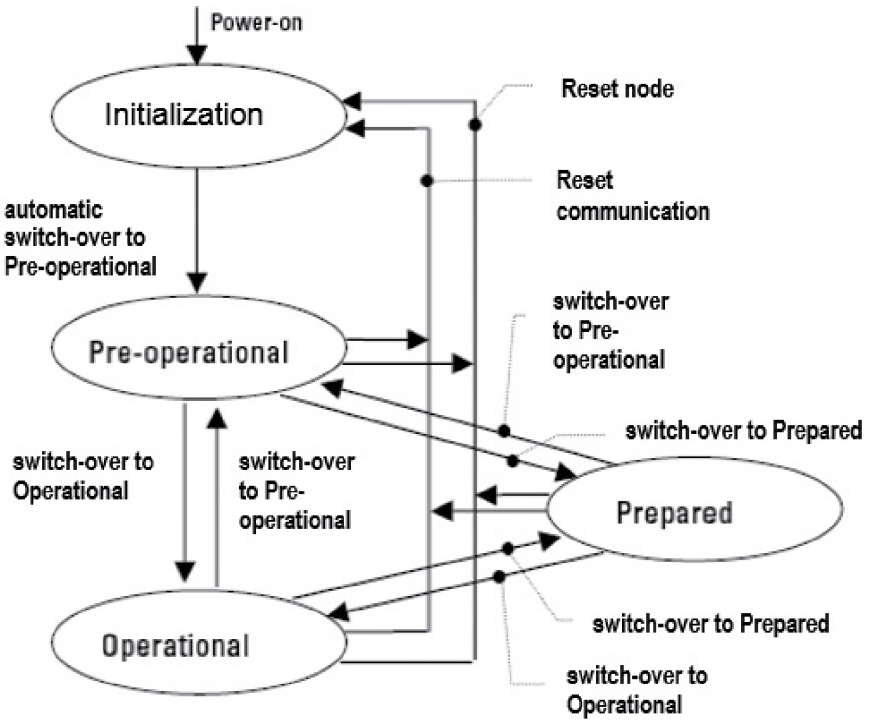
Byte 0	Byte 1
$2^7 \ldots 2^0$	$2^{15} \ldots 2^8$

Value range up to FFFFh
Example: 4FA6h current CRC16 for the raw position data

Objects not mentioned
All Objects not mentioned here serve as additional information and can be found in Encoder profile DS 406.

23 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):



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 through 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.

24 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.
A NMT Object is structured as follows:

Byte 0	Byte 1
2^7... 2^0	2^{15}...2^8

COB ID = 0
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.

Command Byte (Hex)	**Description**
01h	Start_Remote_Node: Change to Operational
02h	Stop_Remote_Node: Change to Non-Operational
80h	Enter_Pre-Operational_State: Change to Pre-Operational
81h	Reset_Node: Reset Node[1]
82h	Reset_Communication: Reset Communication[2]

[1] All the parameters in the whole Object Dictionary will have their values set to Power-On values.
[2] Only the parameters in the section Communication Profile of the Object Dictionary will have their values set to Power-On values.

25 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
MT	Multiturn encoder
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
ST	Singleturn encoder
SYNC	Synchronization telegram. Stations on the Bus reply to the SYNC command by transmitting their process value

26 Glossary

Baud rate
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 include 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 customized 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 their 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:

- File description (name, version, creation date, among others)
- General device information (manufacturer's name and code)
- Device name and type, version, LMT address
- Supported baud rates and Boot-up ability
- Description of the supported Objects and their attributes

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.

27 Troubleshooting

Cause	CANbus	Error source	LED
No communication No LED	No communication	Power supply available? GNC and VCC reversed Power supply interruption	Both dark
Red LED flashing	No arbitration possible Error frames	CANbus interrupted Wrong baud rate Termination CAN_High and CAN_Low bus lines reversed	Red flashing 1 sec.
Device does not answer to request Green flashing	Bootup message present	Wrong node address RTR request is not supported	Green 250 ms
Device does not answer to Start-Operational Green flashing	Bootup message present SDO transfer present	Wrong node address Use global command Start OP 01 00	Green 250 ms
Device does not answer to Reset-Node Green flashing	Bootup message present SDO transfer present	Wrong node address Use global command Reset node 81 00	Green 250 ms
Device does not answer to Enter Pre-OP Green constantly on	Operational mode PDO transfer	Wrong node address Use global command Enter Pre-OP 80 00	Green constantly on
Device switches autonomously from OP to Pre-OP mode	Operational mode PDO transfer Fall back in Pre-OP	Error Behavior Object setting Occurrence of an error Page 35	Green 250 ms

Turck Inc. sells its products through Authorized Distributors. These distributors provide our customers with technical support, service and local stock. Turck distributors are located nationwide – including all major metropolitan marketing areas.
For Application Assistance or for the location of your nearest Turck distributor, call:
1-800-544-7769

Specifications in this manual are subject to change without notice. Turck also reserves the right to make modifications and makes no guarantee of the accuracy of the information contained herein.
Literature and Media questions or concerns?
Contact Marketing Turck USA – tusa.marketing@turck.com

TURCK

28 subsidiaries and over
60 representations worldwide!

Printed in USA

MA1033 B 10/18

www.turck.com