AAM36-58 CANOPEN INSTALLATION MANUAL

CANopen®

Eltra®
focused on motion

Liability to modification without notice
Eltra reserves the right to make changes without further notice to any products or data herein to improve reliability, function, or design.

Warranty disclaimer
Information furnished by Eltra is believed to be accurate and reliable.
However, Eltra does not assume any liability arising out of the application or use of this information, nor the application or use of any product or circuit described herein, neither does it convey any license under its patent rights nor the rights of others.
The specified product features and technical data shall in no case not constitute a guarantee declaration.

Document information
03/2020 - first revision, english version
05/2020 - added italian translation and connection table

Eltra SPA Unipersonale
Via Guido Salvagnini, 17
36040 Sarego (VI)
Italy
Phone: +39 0444 436489
Fax: +49 0444 835335
info@eltra.it
www.eltra.it

TABLE OF CONTENTS

CANopen®

DEVICE DESCRIPTION

Product description

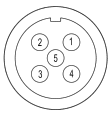
Eltra AA / AAM 36-58 encoders are available in different mechanical versions. Key features are size and shape. The standard sizes are 36mm and 58mm flange diameter. Different types of shapes according to shafts and flanges are available.
The shaft or the hollow blind shaft will be connected to the rotating part of which the angular position or rotation you want to measure. The encoder itself is mounted by several mechanical flanges or torque supports.
A stub cable or M12 sized connector provides the electrical connection to the CAN-network.
A bicolour status LED at the top indicates the different states of the encoder during use and helps with configuration and troubleshooting.

Connections

CONNECTIONS

Function	Cable	5 pin M12
+ V DC	brown	2
0 V	white	3
CAN_H	green	4
CAN_L	yellow	5
CAN_GND (shield)	shield	1
⏚	encoder housing	encoder housing

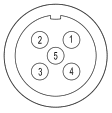
M12 connector(5 pin)
M12 A coded
solder side view FV



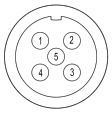
CONNECTIONS - WITH GALVANIC ISOLATION

Function	5 pin female M12	5 pin male M12
+ V DC	2	2
0 V	3	3
CAN_H	4	4
CAN_L	5	5
CAN_GND (shield)	1	1
⏚	encoder housing	encoder housing

M12 connector (5 pin)
M12 A coded - female
solder side view FV



M12 connector (5 pin)
M12 A coded - male
solder side view MV



LED status indicator and signal codes

Definition of LED indication types:

- Red LED indications = "Physical Layer" information
- Green LED indications = "NMT-Status" information
- No colour = LED off

Led colour	Led status	Meaning
●	ON	OPERATIONAL state
●	Blinking (200ms)	PRE-OPERATIONAL state
●	Single flash (1s)	STOPPED state
●	ON	NOT READY / BUS OFF
●	Single flash (1s)	Warning, occurence in error frames
●	Double flash (200ms each)	Error, a Node-guard event or a Heartbeat event (heartbeat consumer) has occurred
●	Triple flash (200ms each)	Encoder is bus-passive
● ●	Flickering	Baudrate-Auto-Detection in progress or LSS config mode started

Predefined Connection Settings

By default all Eltra Canopen encoders are set on Node-ID=127h and Baudrate=Auto-Detection.

Services	COB-ID
NMT	000h
SYNC	080h
EMCY	080h + Node-ID
PDO1(tx)	180h + Node-ID
PDO2(tx)	280h + Node-ID
PDO3(tx)	380h + Node-ID
SDO(rx)	600h + Node-ID
SDO(rx)	580h + Node-ID

QUICK START

CAN network integration

The default Node-ID (Object 2101h sub-Index: 00h) is 7Fh=127d.
For operating in a CAN-Network, the encoder's baudrate has to be set. The common ways to set the baudrate is via LSS (CiA DSP-305) or a SDO command.
The encoder has the capability to detect the baudrate of the network automatically (object 2100h sub-Index: 00h value: 09h - Baudrate-auto-detection).
So usually the baudrate setup is not necessary. To detect the valid baudrate the encoder stays passive and scans the communication at the bus.
When the baudrate is detected, the encoder is set to this rate, sends its boot-up message and switches into pre-operational mode.
To prevent possible collisions in case double assigned Node-ID it is recommended to use a 1:1 connection with a bus master for configuration (e.g. a laptop computer with suitable hard- and software). Set the master on the intended baudrate and use SDO or LSS services to configure the encoder.

SDO command (set the Node-ID)

After connecting the encoder with the CAN bus respectively the master (e.g. a laptop computer with suitable hardware and software) the LED starts "flickering red and green" (see LED indications).
First send one or more SYNC messages, which the encoder can use to detect the baudrate:

CAN-ID	DLC	Command	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6
080h	8	00h	00h	00h	00h	00h	00h	00h	00h

The encoder will detect and lock on the used baudrate. It will send its boot-up message and the LED starts to blink green.
To set the encoders Node-ID the object 2101h, Sub-Index 00h has to be accessed (only possible in the PRE-OPERATIONAL state).
Send a write-SDO-command with the intended Node-ID (in hex):

CAN-ID	DLC	Command	Object L	Object H	Sub-Index	Byte3	Byte1	Byte2	Byte3
600h+ID	8	2Fh	01h	21h	00h	Node-ID	00h	00h	00h

An example for a Node-ID might be:

Node-ID (dec)	Node-ID (hex)
1	01h
2	02h
...	...
6	06h
...	...
127	7Fh

The change of the Node-ID via SDO will be effective after a reset of the encoder (hard reset or NMT reset). The new Node-ID is stored into the EEPROM immediately and without a further command. The setting of the Node-ID via LSS is described in the proper chapter.

Encoder start-up (basic operations)

Before connecting the encoder to the bus of application please carefully check the mechanical and electrical installation documents.
When the encoder is completely integrated into the application it can be switched into OPERATIONAL mode by the "Start-All-Nodes-Command".
The encoder is now operational (LED shows green ON) and starts sending its data via the several process data objects (PDO).
The encoders default configuration plans that the PDO1 is triggered once the position value changes.
The position value (object 6004h) is mapped in PDO1 and transmitted as an Unsigned32.
By default PDO2 transmits the same value but synchronously on the reception of a SYNC message.
Heartbeat is switched off and will not be transmitted by default. The encoder is now configured and ready for basic applications.

Communication objects

The standard communication objects comply with the CiA specification 301 v4.02 and have the object addresses in the range 1000h ... 1FFFh.

Object n.	Name	Sub index	Description	Data type	Access type	Map	Default value
1000h	Device type	0h	(MSB) Encoder type (LSB) device profile no.	Unsigned32	Constant	no	MT: 0002 0196h ST: 0001 0196h
1001h	Error register	0h	Indication of internal failures and part of an emergency object	Unsigned8	Read only	yes	00h
1002h	Manufacturer status register	0h	General status register for manufacturer specific purpose	Unsigned32	Read only	yes	Dynamic
1003h	Predefined error field	00h	Stores occurring errors indicated by EMCY; volatile;	Unsigned8	Read/write	no	Dynamic
		01h	Standard error field 1	Unsigned32	Read only		
		02h	Standard error field 2	Unsigned32	Read only		
		03h	Standard error field 3	Unsigned32	Read only		
		04h	Standard error field 4	Unsigned32	Read only		
		05h	Standard error field 5	Unsigned32	Read only		
1005h	COB-ID SYNC-Message	00h	COB-ID of the SYNC message	Unsigned32	Read/write	no	0000 0080h
1008h	Manufacturer device name	00h	Manufacturer device name	String256	Constant	no	ST: AA 36-58-CNP MT: AAM36-58-CNP
1009h	Manufacturer hardware version	00h	Contains the hardware revision assigned by the manufacturer	String16	Constant	Constant	“n.nn” format
100Ah	Manufacturer software version	00h	Contains the software revision assigned by the manufacturer	String72	Constant	no	“n.nn” format
100Ch	Guard time	00h	Defines the guard time in milliseconds; 0h= node guard protocol disabled	Unsigned16	Read/write	no	0000h
100Dh	Life time factor	00h	Contains the life time factor for the node guard protocol	Unsigned8	Read/write	no	00h
1010h	Store parameters	00h		Unsigned8	Constant	no	04h
		01h	Save all parameters	Unsigned32	Read/write		0000 0001h
		02h	Save communication parameters	Unsigned32	Read/write		0000 0001h
		03h	Save application parameters	Unsigned32	Read/write		0000 0001h
		04h	Save manufacturer parameters	Unsigned32	Read/write		0000 0001h
1011h	Restore default parameters		Restores factory settings	Unsigned8	Constant	no	04h
			Restore all parameters	Unsigned32	Read/write		0000 0001h
			Restore communication parameters	Unsigned32	Read/write		0000 0001h
			Restore application parameters	Unsigned32	Read/write		0000 0001h
			Restore manufacturer parameters	Unsigned32	Read/write		0000 0001h
1014h	COB-ID Emergency Object	00h	Defines the COB-ID of the emergency object (EMCY)	Unsigned32	Read/write	no	0000 0080h + Node-ID
1015h	Inhibit time EMCY	00h	Defines the minimum pause (in 100 ms steps) between single EMCYs	Unsigned16	Read/write	no	0000h
1016h	Consumer heartbeat time	00h	Defines the time frame within the heartbeat consumer awaits an incoming heartbeat otherwise triggering an EMCY	Unsigned8	Constant	no	01h
		01h	Heartbeat-Consumer cycle time	Unsigned32	Read/write		0000 0000h
1017h	Producer heartbeat time	00h	Defines the heartbeat cycle time in steps of 1 ms. 0h = heartbeat disabled	Unsigned16	Read/write	no	0000h

Object n.	Name	Sub index	Description	Data type	Access type	Map	Default value
1018h	Identity Object	00h		Unsigned8	Constant	no	04h
		01h	Vendor-ID	Unsigned32	Constant		0000 046C h
		02h	Product Code	Unsigned32	Constant		ST: 057D 2E90 h MT: 0584 5A80 h
		03h	Revision Number	Unsigned32			Current revision
		04h	Serial Number	Unsigned32			Current serial
1020h	Verify configuration	00h	The time of the last configuration can be logged here. If the configu-ration was changed after setting this value, the object is set to zero autonomously	Unsigned8	Constant	no	02h
		01h	Configuration date	Unsigned32	Read/write		0000 0000h
		02h	Configuration date	Unsigned32	Read/write		0000 0000h
1029h	Error behaviour	00h	Changing the encoders behaviour in case of a node-guarding or heartbeat event, etc.	Unsigned8	Constant	no	02h
		01h	Communication error	Unsigned8	Read/write		00h
		02h	Encoder error	Unsigned8	Read/write		00h
1800h	Transmit PDO1 com. parameter	00h	Defines the com. parameters of the TPDO	Unsigned8	Constant	no	05h
		01h	COB-ID or PDO	Unsigned32	Read/write		180h + Node-ID
		02h	Transmission type	Unsigned8	Read/write		FEh
		05h	Event-Timer	Unsigned16	Read/write		0000h
1801h	Transmit PDO2 com. parameter	00h	Defines the com. parameters of the TPDO	Unsigned8	Constant	no	05h
		01h	COB-ID for PDO	Unsigned32	Read/write		280h + Node-ID
		02h	Transmission type	Unsigned8	Read/write		01h
		05h	Event-Timer	Unsigned16	Read/write		0000h
1802h	Transmit PDO3 com. parameter	00h	Defines the com. parameters of the TPDO	Unsigned8	Constant	no	05h
		01h	COB-ID for PDO	Unsigned32	Read/write		380h + Node-ID
		02h	Transmission type	Unsigned8	Read/write		01h
		05h	Event-timer	Unsigned16	Read/write		0000h
1803h	Transmit PDO4 com. parameter	00h	Defines the com. parameters of the TPDO	Unsigned8	Constant	no	05h
		01h	COB-ID for PDO	Unsigned32	Read/write		480h + Node-ID
		02h	Transmission Type	Unsigned8	Read/write		01h
		05h	Event-Timer	Unsigned16	Read/write		0000h
1A00h	TPDO1 mapping parameter	00h	Defines the PDO-mapping of the TPDO	Unsigned8	Read/write	no	01h
		01h	Mapped application object 1	Unsigned32	Read/write		6004 0020h
	variable, depends on sub-index 00h	02h - 08h	Mapped application object 2 - 8	Unsigned32	Read/write		
1A01h	TPDO2 mapping parameter	00h	Defines the PDO-mapping of the TPDO	Unsigned8	Read/write	no	01h
		01h	Mapped application object 1	Unsigned32	Read/write		6004 0020h
	variable, depends on sub-index 00h	02h - 08h	Mapped application object 2 - 8	Unsigned32	Read/write		

Object n.	Name	Sub index	Description	Data type	Access type	Map	Default value
1A02h	TPDO3 mapping parameter	00h	Defines the PDO-mapping of the TPDO	Unsigned8	Read/write	no	no
		01h	Mapped application object 1	Unsigned32	Read/write		
	variable, depends on sub-index 00h	02h - 08h	Mapped application object 2 - 8	Unsigned32	Read/write		
1A03h	TPDO4 mapping parameter	00h	Defines the PDO-mapping of the TPDO	Unsigned8	Read/write	no	no
	variable, depends on sub-index 00h	01h - 08h	Mapped application object 1 - 8	Unsigned32	Read/write		
1F80h	NMT-Start-up-behaviour	00h	Defines the start-up behaviour of encoder	Unsigned32	Read/write	no	no

Device specific objects

The device specific objects comply with the CiA encoder profile specification 406 v3.2 and have the object addresses in the range 6000h ... 9FFFh.

Object n.	Name	Sub index	Description	Data type	Access type	Map	Default value
6000h	Operating Parameters	00h	Changing / indicating the operating parameters	Unsigned16	Read/write	no	0004h
6001h	Measuring units per revolution	00h	Changing / indicating the single-turn resolution	Unsigned32	Read/write	no	0000 4000h
6002h	Total measuring range	00h	Changing / indicating the total measuring range	Unsigned32	Read/write	no	i*
6003h	Preset value	00h	Setting / indicating the preset value to adapt the position value to the application	Unsigned32	Read/write	no	0000 0000h
6004h	Position value	00h	Current position value	Unsigned32	Read only	yes	Dynamic
6008h	High precision position value	00h	Current position value, when measuring range > 32 bit	Unsigned64	Read only	yes	Dynamic
6009h	High precision Preset Value	00h	Setting/indicating the High-precision-preset-value. Access via segmented or block transfer	Unsigned64	Read/write	no	0000 0000 0000 0000h
6030h	Speed Value	00h	Rotation speed in units (bit) per second	Unsigned8	Read only	yes	01h
		01h	Speed value	Signed16	Read only		Dynamic
6040h	Acceleration Value	00h	Acceleration value in units(bit) per second^2	Unsigned8	Read only	yes	01h
		01h	Acceleration value	Signed16	Read only		Dynamic
6050h	Jerk Value	00h	Jerk value in units (bit) per second^3	Unsigned8	Read only	yes	01h
		01h	Jerk value	Signed16	Read only		Dynamic
6200h	Cyclic-Timer	00h	Changing / indicating the transmission period of asynchro-nous TPDOs	Unsigned16	Read/write	no	0001h
6300h	CAM state register	00h	Status bits of the cams of the corresponding cam channel	Unsigned8	Read only	yes	01h
		01h	Cam state channel1 0b=inactive / 1h=active	Unsigned8	Read only		00h
6301h	CAM enable register	00h	Changing / indicating the cam enable bits of the corresponding cam channel	Unsigned8	Read only	no	01h
		01h	Cam enable channel1 0b=inactive / 1b=active	Unsigned8	Read/write		00h
6302h	CAM polarity register	00h	Changing / Indicating the inversion of the corresponding cam in (6300h)	Unsigned8	Constant	no	01h
		01h	Cam polarity channel1 0b=cam state not inverted / 1b=cam state inverted	Unsigned8	Read/write		00h
6310h	CAM1 low limit	00h	Indicating the lower switching point of the 1st cam	Unsigned8	Constant	no	01h
		01h	Changing lower switching point CAM1	Signed32	Read/write		0000 0000h
6311h	CAM2 low limit	00h	Indicating the lower switching point of the 2nd cam	Unsigned8	Constant	no	01h
		01h	Changing lower switching point CAM2	Signed32	Read/write		0000 0000h

Object n.	Name	Sub index	Description	Data type	Access type	Map	Default value
6312h	CAM3 low limit	00h	Indicating the lower switching point of the 3rd cam	Unsigned8	Constant	no	01h
		01h	Changing lower switching point CAM3	Signed32	Read/write		0000 0000h
6313h	CAM4 low limit	00h	Indicating the lower switching point of the 4th cam	Unsigned8	Constant	no	01h
		01h	Changing lower switching point CAM4	Signed32	Read/write		0000 0000h
6314h	CAM5 low limit	00h	Indicating the lower switching point of the 5th cam	Unsigned8	Constant	no	01h
		01h	Changing lower switching point CAM5	Signed32	Read/write		0000 0000h
6315h	CAM6 low limit	00h	Indicating the lower switching point of the 6th cam	Unsigned8	Constant	no	01h
		01h	Changing lower switching point CAM6	Signed32	Read/write		0000 0000h
6316h	CAM7 low limit	00h	Indicating the lower switching point of the 7th cam	Unsigned8	Constant	no	01h
		01h	Changing lower switching point CAM7	Signed32	Read/write		0000 0000h
6317h	CAM8 low limit	00h	Indicating the lower switching point of the 8th cam	Unsigned8	Constant	no	01h
		01h	Changing lower switching point CAM8	Signed32	Read/write		0000 0000h
6320h	CAM1 high limit	00h	Indicating the upper switching point of the 1st cam	Unsigned8	Constant	no	01h
		01h	Changing upper switching point CAM1	Signed32	Read/write		0000 0000h
6321h	CAM2 high limit	00h	Indicating the upper switching point of the 2nd cam	Unsigned8	Constant	no	01h
		01h	Changing upper switching point CAM2	Signed32	Read/write		0000 0000h
6322h	CAM3 high limit	00h	Indicating the upper switching point of the 3rd cam	Unsigned8	Constant	no	01h
		01h	Changing upper switching point CAM3	Signed32	Read/write		0000 0000h
6323h	CAM4 high limit	00h	Indicating the upper switching point of the 4th cam	Unsigned8	Constant	no	01h
		01h	Changing upper switching point CAM4	Signed32	Read/write		0000 0000h
6324h	CAM5 high limit	00h	Indicating upper switching point CAM5	Unsigned8	Constant	no	01h
		01h	Changing upper switching point CAM5	Signed32	Read/write		0000 0000h
6325h	CAM6 high limit	00h	Changing / Indicating the upper switching point of the 6th cam	Unsigned8	Constant	no	01h
		01h	Changing upper switching point CAM6	Signed32	Read/write		0000 0000h
6326h	CAM7 high limit	00h	Changing / Indicating the upper switching point of the 7th cam	Unsigned8	Constant	no	01h
		01h	Changing upper switching point CAM7	Signed32	Read/write		0000 0000h
6327h	CAM8 high limit	00h	Changing / Indicating the upper switching point of the 8th cam	Unsigned8	Constant	no	01h
		01h	Changing upper switching point CAM8	Signed32	Read/write		0000 0000h

Object n.	Name	Sub index	Description	Data type	Access type	Map	Default value
6330h	CAM1 hysteresis	00h	Indicating the hysteresis for the 1st cam	Unsigned8	Constant	no	01h
		01h	Changing the hysteresis for the 1st cam	Unsigned32	Read/write		0000 0000h
6331h	CAM2 hysteresis	00h	Indicating the hysteresis for the 2nd cam	Unsigned8	Constant	no	01h
		01h	Changing the hysteresis for the 2nd cam	Unsigned32	Read/write		0000 0000h
6332h	CAM3 hysteresis	00h	Indicating the hysteresis for the 3rd cam	Unsigned8	Constant	no	01h
		01h	Changing the hysteresis for the 3rd cam	Unsigned32	Read/write		0000 0000h
6333h	CAM4 hysteresis	00h	Indicating the hysteresis for the 4th cam	Unsigned8	Constant	no	01h
		01h	Changing the hysteresis for the 4th cam	Unsigned32	Read/write		0000 0000h
6334h	CAM5 hysteresis	00h	Indicating the hysteresis for the 5th cam	Unsigned8	Constant	no	01h
		01h	Changing the hysteresis for the 5th cam	Unsigned32	Read/write		0000 0000h
6335h	CAM6 hysteresis	00h	Indicating the hysteresis for the 6th cam	Unsigned8	Constant	no	01h
		01h	Changing the hysteresis for the 6th cam	Unsigned32	Read/write		0000 0000h
6336h	CAM7 hysteresis	00h	Indicating the hysteresis for the 7th cam	Unsigned8	Constant	no	01h
		01h	Changing the hysteresis for the 7th cam	Unsigned32	Read/write		0000 0000h
6337h	CAM8 hysteresis	00h	Indicating the hysteresis for the 8th cam	Unsigned8	Constant	no	01h
		01h	Changing the hysteresis for the 8th cam	Unsigned32	Read/write		0000 0000h
6400h	Area state register	00h	Indicating if the current position is in or outside the work area	Unsigned8	Constant	yes	01h
		01h	Status of the area state register: 00h = within area 03h = outside work area 05h = outside work area	Unsigned8	Read only		Dynamic
6401h	Work area low limit	00h	Number of sub-indices	Unsigned8	Constant	no	01h
		01h	Changing / Indicating the work area low limit	Signed32	Read/write		0000 0000h
6402h	Work area high limit	00h	Number of sub-indices	Unsigned8	Constant	no	01h
		01h	Changing / Indicating the work area high limit	Signed32	Read/write		0000 4000h
6500h	Operating status	00h	Indicates the operating state of the device	Unsigned16	Read only	no	Dynamic
6501h	Measuring units per revolution	00h	Indication of the singleturn resolution	Unsigned32	Constant	no	0000 4000h
6502h	Number of distinguishable revolutions	00h	Indication of the multiturn resolution	Unsigned16	Constant	no	ST: 0001h MT: FFFFh
6503h	Alarms	00h	Alarm set by malfunction	Unsigned16	Read only	yes	Dynamic
6504h	Supported alarms	00h	Information about supported alarms	Unsigned16	Constant	no	0001h

Object n.	Name	Sub index	Description	Data type	Access type	Map	Default value
6505h	Warnings	00h	Warning set on deviation of certain parameters	Unsigned16	Read only	yes	Dynamic
6506h	Supported warnings	00h	Information about supported warnings	Unsigned16	Constant	no	7001h
6507h	Profile and software version	00h	Revision of the implemented encoder profile and software	Unsigned32	Constant	no	0105 0302h
6508h	Operating time	00h	not supported	Unsigned32	Constant	no	FFFF FFFFh
6509h	Offset value	00h	Offset value, calculated from the preset value (6003h)	Signed32	Read only	no	0000 0000h
650Ah	Module identification	00h	Manufacturer specific offset	Unsigned8	Constant	no	03h
		01h	Manufacturer offset value	Signed32	Constant		00h
		02h	Manufacturer min.-position	Signed32	Constant		-
		03h	Manufacturer max.-position	Signed32	Constant		-
650Bh	Serial number	00h	Serial number of the encoder, hard wired with object 1018h-04h	Unsigned8	Constant	no	01h
		01h	Serial number	Unsigned32	Constant		i*
6510h	Number of high-precision revolutions	00h	Indicates the maximum possible high-precision multiturn resolution	Unsigned40	Constant	no	0080 0000 0000h

Manufacturer specific objects

The objects in the addresses in the range 2000h ... 5FFFh are manufacturer specific and not defined by the CiA.

Object n.	Name	Sub index	Description	Data type	Access type	Map	Default value
2100h	Baudrate	00h	Setting the baudrate	Unsigned8	Read/write	no	09h
2101h	Node-ID	00h	Setting the node-ID	Unsigned8	Read/write	no	7Fh
2103h	BUS-Off Auto-Reset	00h	Defines the time in BUS OFF, before automatically resetting. 0h = no automatic reset, 01h-FFh = time in seconds	Unsigned8	Read/write	no	00h
2105h	Integration value	00h	Number of values for filtering speed, acceleration and jerk	Unsigned8	Read/write	no	02h
		01h	Integration-Position value filter	Unsigned8	Read/write		01h
		02h	Integration-Speed value filter	Unsigned16	Read/write		03E8h
2106h	Speed scaling	00h	Speed value scaling	Unsigned8	Constant	no	02h
		01h	Multiplier	Unsigned16	Read/write		0001h
		02h	Divisor	Unsigned16	Read/write		0001h
2107h	Frequency-Limit	00h	Limit for Speed value	Unsigned16	Read/write	no	FFFFh
2120h	Customer EEPROM area	00h	Object to store any customer data	Unsigned8	Constant	no	08h
		01h	Customer data 1	Unsigned32	Read/write		FFFF FFFFh
		02h	Customer data 2	Unsigned32	Read/write		FFFF FFFFh
		03h	Customer data 3	Unsigned32	Read/write		FFFF FFFFh
		04h	Customer data 4	Unsigned32	Read/write		FFFF FFFFh
		05h	Customer data 5	Unsigned32	Read/write		FFFF FFFFh
		06h	Customer data 6	Unsigned32	Read/write		FFFF FFFFh
		07h	Customer data 7	Unsigned32	Read/write		FFFF FFFFh
		08h	Customer data 8	Unsigned32	Read/write		FFFF FFFFh
2500h	Temperature object	00h	Monitoring the internal operating temperature	Unsigned8	Constant	yes	05h
		01h	Current temperature value	Signed16	Read only		Dynamic
		02h	Upper Limit	Signed16	Read/write		+100 (°C)
		03h	Lower Limit	Signed16	Read/write		-40 (°C)
		04h	Maximum value occurred	Signed16	Read only		Dynamic
		05h	Minimum value occurred	Signed16	Read only		Dynamic
2502h	Error history	00h	Non-volatile error history	Unsigned32	Read only	no	Dynamic
		01h	Standard Error field 1	Unsigned32	Read only		Dynamic
		02h	Standard Error field 2	Unsigned32	Read only		Dynamic
		03h	Standard Error field 3	Unsigned32	Read only		Dynamic
		04h	Standard Error field 4	Unsigned32	Read only		Dynamic
		05h	Standard Error field 5	Unsigned32	Read only		Dynamic
2503h	Alarms history	00h	Logging of alarms occurred, number of alarms	Unsigned8	Constant	no	Dynamic
		01h	Alarm 1	Unsigned16	Read only		Dynamic
		02h	Alarm 2	Unsigned16	Read only		Dynamic
		03h	Alarm 3	Unsigned16	Read only		Dynamic
		04h	Alarm 4	Unsigned16	Read only		Dynamic
		05h	Alarm 5	Unsigned16	Read only		Dynamic
2504h	Warnings history	00h	Logging of warnings occurred, number of warnings	Unsigned8	Read/write	no	Dynamic
		01h	Warning 1	Unsigned16	Read only		Dynamic
		02h	Warning 2	Unsigned16	Read only		Dynamic
		03h	Warning 3	Unsigned16	Read only		Dynamic
		04h	Warning 4	Unsigned16	Read only		Dynamic
		05h	Warning 5	Unsigned16	Read only		Dynamic

OBJECT DESCRIPTION

Network management (NMT) commands

To switch between the encoders states (STOPPED, PRE-OPERATIONAL, OPERATIONAL) or to trigger a soft reset, there are different NMT commands. The messages are 3 bytes each and will not be acknowledged. The CAN-ID of the NMT is always ZERO and therefor has the highest priority.

CAN-ID	DLC	Command	Node-ID
0	02h	Byte0	Byte1

Structure of NMT-command

Command:

The command determines the intended reaction of the addressed node.

Command	Value
Start node	01h
Stop node	02h
Pre-operational	80h
Reset node	81h
Reset communication	82h

Commands for NMT-command

Node-ID:

The command determines the intended reaction of the addressed node.

Command	Value
All nodes	00d
Valid Node-IDs	01 ... 127d
Invalid Node-IDs	128 ... 255d

Node-ID values for NMT-commands

Heartbeat protocol

By default the heartbeat protocol is disabled.
The encoder can either send a heartbeat (producer heartbeat) or monitor the heartbeat of other nodes (consumer heartbeat).

Producer heartbeat (Encoder sends its heartbeat)
The producer heartbeat can be enabled by setting the producer heartbeat time in milliseconds respectively can be disabled by setting the producer heartbeat time to 00h. This is done by object 1017h, sub-index 0 (00h=OFF, time in milliseconds = 0 ... 9999h).

Consumer Heartbeat (Encoder monitors an external heartbeat)
The object 1016h, sub-Index=01h, defines the consumer heartbeat time. The encoder uses this time to monitor another heartbeat producer. If the monitored heartbeat does not occur within this time (e.g. device broken), the encoder sends an EMCY message with error code 8130h (Life guard or heartbeat error).
The object also defines the node-ID to be monitored.

Reserved (00h)	Node-Id	Heartbeat Producer time
Bit 31-24	Bit 23-16	Bit 15-0

Monitor external heartbeat

A time value of 0 or a node value 0 or higher than 127 disables the function.
Example for monitoring the node 127d =7Fh with a heartbeat consumer time of 10000 milliseconds (=2710h). The encoder is assumed to be node 1:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Time L	Time H	Producer Node-ID
601h	8	23h	16h	10h	01h	10h	27h	7Fh

Example configuration of a consumer heartbeat

Emergency messages (EMCY)

An emergency is sent when a failure occurs either on the bus or within the device. The error is coded within an EMCY message.

Object 1014h defines the COB-ID of the emergency message. The default value is 80h + device node-ID (1 ... 127).
Basic CAN Frames or Extended CAN Frames can be used (Bit 29 = 1).

General structure of an emergency message is:

CAN-ID	DLC	Byte0	Byte1	Byte2	Byte3	Byte4
80+ID	8	Error code L	Error code H	Error Register	Info1	Info2

Basic structure of EMCY

Error code (H,L)	Description
0000h	No error
4200h	Temperature out of tolerance
5000h	Hardware failure (EEPROM)
8110h	CAN communication error (overrun)
8120h	CAN communication error (passive state)
8130h	Heartbeat / Life guarding error
8140h	Bus Off recovery

Emergency error code list

Error register:

Interpretation of object 1001h (bit interpretation, default = 00000000).

Bit	7	6	5	4	3	2	1	0
Info	Constant	Constant	Constant	Communication	Temperature	Constant	Constant	EEPROM error

Error register

List info field:

The info field depends on the error codes.

Error Code	Field	Bit	Hex-value	Error description
4200h	Infofield 1 (Byte3)	6	40h	Temperature read error
		5	20h	Low limit exceeded
		4	10h	High limit exceeded

Error Code	Field	Bit	Hex-value	Error description
5000h	Infofield 2 (Byte4)	0	01h	EEPROM error in init-phase
		3	08h	EEPROM write timeout

Error Code	Field	Bit	Hex-value	Error description
8120h + 8100h	Infofield 1 (Byte3) Low nibble	0	1h	Active, no error
		1+2	6h	Bus warning
		0+1+2	7h	Bus passive
8120h + 8100h	Infofield 1 (Byte3) High nibble	0	1h	Bit
		1	2h	Stuffing error
		0+1	3h	Form
		2	4h	CRC
		0+2	5h	Ack

Infofield list

The low nibble describes the CAN state, the high nibble gives further information about the error. The transmission of EMCY messages can be disabled by setting bit 31 (MSB) in object 1014h-00h. By changing 1015h a minimum pause between two EMCYs can be defined (in steps of 100µs).

Error Objects

Manufacturer status register

Interpretation of object 1002h (assignment bit - meaning, standard = 00h).

Bit	7	6	5	4	3	2	1	0
Info	Constant	Constant	Constant	Constant	Constant	EEPROM*	MT*	ST*(1)

Bit	15	14	13	12	11	10	9	8
Info	ST*(8)	ST*(7)	ST*(6)	ST*(5)	ST*(4)	ST*(3)	ST*(2)	ST*(1)

Bit	23	22	21	20	19	18	17	16
Info	ST*(15)	ST*(14)	ST*(13)	ST*(12)	ST*(11)	ST*(10)	ST*(9)	ST*(8)

Bit	31	30	29	28	27	26	25	24
Info	MT*(9)	MT*(8)	MT*(7)	MT*(6)	MT*(5)	MT*(4)	MT*(3)	MT*(2)

Manufacturer status register

*= Errortype (number) I for detailed definitions please contact our technical support.

Alarms

Interpretation of object 6503h
(assignment bit - meaning, standard = 0000000000000000).

Bit	15 ... 1	0
Info	Constant	Position error

Alarms - Object 6503h

Warnings

Interpretation of object 6505h
(assignment bit - meaning, standard = 0000000000000000).

Bit	15	14	13	12	11 ... 1	0
Info	Constant	Temp. read filed	Undertemp.	Overtemp.	Constant	Frequency limit

Warnings - Object 6505h

Electronic cam switch (CAM)

The encoder provide the possibility to configure an electronic cam switch with 8 cams in one single channel. Every cam is defined by its low and high limit, the hysteresis and the polarity.

CAM-state register

The cam state register (object 6300h) represents the state of the 8 cam switches, one bit per cam. For example the cam state register has the value of 89h:

Position	7 (MSB)	6	5	4	3	2	1	0 (LSB)
Type	CAM 8	CAM 7	CAM 6	CAM 5	CAM 4	CAM 3	CAM 2	CAM 1
Value	1	0	0	0	1	0	0	1
Logic	High	Low	Low	Low	High	Low	Low	High

CAM-state register - Value 89h

That means that the cams 1, 4 and 8 are high and the rest are low. If e.g. the cam 4 toggles to low due to the change of the position value, the cam state register would become 81h:

Position	7 (MSB)	6	5	4	3	2	1	0 (LSB)
Type	CAM 8	CAM 7	CAM 6	CAM 5	CAM 4	CAM 3	CAM 2	CAM 1
Value	1	0	0	0	0	0	0	1
Logic	High	Low	Low	Low	Low	Low	Low	High

CAM-state register - Value 81h

The cams are independent to each other so the cam state register can take on 256 combinations to control a machine.

CAM-enable register

Each cam can separately be enabled or disabled by the object 6301h sub-Index 01h. The cams are represented by the bits of the object, 1 = ON, 0 = OFF. For example CAM 2, CAM 4 and CAM 7 shall be enabled, so the configuration is:

Position	7 (MSB)	6	5	4	3	2	1	0 (LSB)
Type	CAM 8	CAM 7	CAM 6	CAM 5	CAM 4	CAM 3	CAM 2	CAM 1
Value	0	1	0	0	1	0	1	0
Logic	High	Low	Low	Low	Low	Low	Low	High

CAM-enable register - Value 4Ah

That means writing 4h to object 6301h sub-index 01h. The cams 2, 4 and 7 are now enabled and can switch depending on their configured limits and the position value.

CAM-polarity register

The cam-polarity-register object 6302h sub-index 01h alters the polarity of the corresponding cam states in cam state register.
By default all cams are high (=1b) when the position value is within the limits of the cam.
E.g. if the cam polarity register is set to 13h (=00010011b) the cams 1, 2 and 6 are inverted (Bit = 0b (Low), while position value inside limits).

Position	7 (MSB)	6	5	4	3	2	1	0 (LSB)
Type	CAM 8	CAM 7	CAM 6	CAM 5	CAM 4	CAM 3	CAM 2	CAM 1
Value	0	0	0	1	0	0	1	1
Logic	High	Low	Low	Low	Low	Low	Low	High

Example CAM-polarity register

CAM-low limit

The object CAM-low limit sets the lower switching position for a cam. Each cam has its own CAM-low-limit object. (See object dictionary 6310h...6317h).
Within the low-limit objects the sub index represents a cam channel. The encoder provides one channel with 8 cams.
Please note that the cam-high-limit always has to be lower than the corresponding low-limit. Therefore the high-limit must be usually configured before the corresponding low-limit.

CAM-high limit
The CAM-High-Limit defines the upper switching position for a cam, similar to the CAM-low-limit. Therefore each cam has its own high-limit-object (see object dictionary 6320h .. 6327h).

CAM-hysteresis
The CAM-Hysteresis defines the width of the cam hysteresis for each single cam (see object dictionary 6320h...6327h).

Device profile
Object 1000h provides the number of the implemented device profile and the device type:

0001 0196h –singleturn encoder DS-406 device profile
0002 0196h –multiturn encoder DS-406 device profile

SYNC
1005h is the selected COB-ID on which the encoder awaits the SYNC message. BasicCAN frames and ExtendedCAN frames (Bit 29 = 1b) are supported. The encoder is a SYNC consumer, not a producer.

Encoder designation
Object 1008h delivers the encoder designation. Only sub index 0 is supported. The value of this object depends on the variant of the firmware.

AA 36-58-CNP - singleturn CANopen
AAM 36-58-CNP - multiturn CANopen

Error behaviour
On a CAN communication error an OPERATIONAL encoder switches into PREOPERATIONAL. The behaviour on CAN bus errors can be changed by object 1029h sub-index 01h and the behaviour on encoder errors can be changed by sub-index 02h.
The following values are valid on sub-index 01h and 02h.

Value	Description
00h	Default behaviour, go PRE-OPERATIONAL
01h	Do not change current NMT state
02h	Go STOPPED

Selection of encoder reaction on errors

NMT start-up behaviour
Index 1F80h determines the encoders NMT-start-up behaviour (only sub-index 0 is supported). By sending a “start all nodes” the encoder takes the task of a basic NMTmaster. The configuration has to be saved into the EEPROM. There are 3 options:

Value	Description
00h	Default behaviour, go PRE-OPERATIONAL
01h	Send NMT-command “Start All Nodes”
02h	Go OPERATIONAL

Selection of start-up behaviuour

Bus-Off Auto-Reset
Index 2103h configures the encoder behaviour when it enters Bus-off state. The default value “0” means that the encoder will remain bus-off until reset. By changing this value the time can be configured in seconds after which the encoder will automatically switch to CAN-Error Active.
This feature has to be used with caution, because it can have a critical impact on the whole bus system.

Customer Data
The object 2120h provides the possibility to store up to 8 data objects (4 byte per object) into the internal EEPROM. Each data is accessed by a sub-index (1...8). The data is stored autonomous, a “save”’ command is not necessary.

Temperature
The 2500h provides the current internal temperature of the encoder, as well as the possibility to set temperature limits for the device. Sub-indices 0 to 5 are supported. The temperature value is updated every minute. The unit is °C. Crossing the temperature limits will set the error register (object 1001h-00h) to 1000b (=08h) and trigger a non-recurring EMCY message. The warning object (6505h) will also be effected.
By default the limits are set to the maximum values allowed, but can be tightened.

Verify Configuration
You can write the time of the last valid configuration into object 1020h.
This object is also readable. Any change in the configuration will automatically reset this object to zero. Then the new time of configuration can be set.

Please note that all changes in parameters, unless otherwise specified, have to be saved into the EEPROM, e.g. by using the “Store All Parameters” command (see ”Saving parameters into EEPROM”). Otherwise the encoder will return to the last configuration saved after a reset.

SETTING-UP THE ENCODER

Configuration via LSS

The Layer Setting Services Protocol is specified in the Draft Standard Proposal 305. The LSS allows to configure the encoder even when the Node-ID is not assigned correctly (e.g. the default Node-ID doesn't match the application before configuration). The encoder provides the following LSS services:

- Switch state global
- Switch state selective
- Configure baudrate service
- Configure node-ID service
- Store configuration service
- Identification and inquire services (Node-ID, Vendor-ID, Product code, Revision number, Serial number)

A LSS message has the following form:

CAN-ID	DLC	Command	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6

For the CAN-ID applies:

- LSS-Master -> LSS-Slave: 2021(7E5h)
- LSS-Slave) -> LSS-Master: 2020(7E4h)

To use LSS the encoder has to be STOPPED or PRE-OPERATIONAL. Then the encoder can be set into LSS mode by two ways:

- Switch Mode Global
- Switch Mode Selective

LSS configuration by "Switch Mode Global"

Connect the LSS master with the encoder. If possible start encoder before master. The baudrate used by the master will be detected by the encoder. Use the NMT command to switch the encoder into "STOPPED" mode. Send the following message:

7E5h	04h	01h	00h	00h	00h	00h	00h	00h

Command to set encoder Stopped Mode

The encoder is now in configuration mode and now the baudrate and Node-ID of the encoder can be configured via LSS (see proper section).

LSS configuration by "Switch Mode Selective"

Connect the LSS master with the encoder. If possible start encoder before master. The baudrate used by the master will be detected by the encoder. Use the NMT command to switch the encoder into "STOPPED" mode.
With the switch mode selective a certain device can be selected by sending four identification messages:

LSS-command	Information	Description
40h	Vendor-ID	0000 046C h
41h	Product code	ST: 057D 2E90 h MT: 0584 5A80 h
42h	Revision number	Revision of the encoder
43h	Serial number	Serial number of the encoder

LSS-Selective-Identification-Commands

After the last of the four identification messages was send, the encoder will respond with:

LSS-command	Information	Description
44h	Mode	Mode = 1 -> configuration mode Mode = 0 -> operation mode

Answer of encoder to LSS-Selective-Identification-Commands

The encoder is now in configuration mode. Now the encoder baudrate and Node-ID can be set using LSS (see proper chapter).
As soon as the encoder has entered the LSS configuration mode (selective or global) baudrate and Node-ID can be changed by LSS. After changing the settings have to be stored and the configuration mode has to be deactivated. (see below "End LSS configuration mode").

End LSS configuration mode

When the configuration is completed the changed parameters must be stored and the encoder has to be switched into PRE-OPERATIONAL state by using the following message sequence and a final reset (e.g. a power reset):
Step 1 - store parameters:

7E5h	17h	00h	00h	00h	00h	00h	00h	00h

End LSS configuration mode – Step 1 - store parameters

Step 2 - leave configuration mode:

7E5h	04h	00h	00h	00h	00h	00h	00h	00h

End LSS configuration mode – Step 2 - leave configuration mode

Step 3 - reset (e.g. NMT "reset node" or power reset)

Baudrate settings

To set the baudrate send the following command:

CAN-ID	Command	Sub-index	Baudate	Byte2	Byte3	Byte4	Byte5	Byte6
7E5h	13h	00h	Baudrate	00h	00h	00h	00h	00h

Set baudrate

The following baud rates can be selected:

Value	Baudrate
0	1 Mbit/s
1	800 kbit/s
2	500 kbit/s
3	250 kbit/s
4	125 kbit/s
5	100 kbit/s
6	50 kbit/s
7	20 kbit/s
8	10 kbit/s
9	Auto

Baudrate coding

Check the LSS slaves answer to the command above:

CAN-ID	Command	Error code	Specific error	Byte2	Byte3	Byte4	Byte5	Byte6
7E5h	13h	00h	00h	00h	00h	00h	00h	00h

Answer of LSS-slave

Error Code:

- 00h = OK
- 01h = Baudrate not supported

Specific Error:

- 00h = OK
- FFh = Application specific error

Possibly the communication with the encoder fails after the configuration because the configuration tool and the encoder might operate on different baud rates, so you have to change the baudrate configuration of your tool.
Before changing the baudrate please check the baudrate of the application. Please be sure that the configuration tool supports that baudrate and please make a note of the selected baudrate (e.g. in this manual or on the encoder label).

Node-ID setting

Use the following command to change the encoder's Node-ID:

CAN-ID	Command	Node-ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6
7E5h	11h	Node-ID	00h	00h	00h	00h	00h	00h

set Node-ID

Valid Node-IDs can be:

Encoder numer (d)	Node-ID (h)
1	01
2	02
...	...
127	7F

Valid Node-IDs

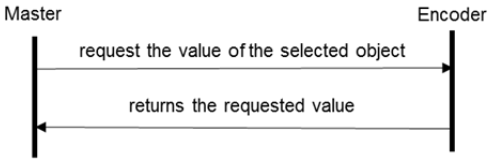
Please remind to leave the LSS configuration mode after configuration (see above).

Configuration via SDO

If not specified otherwise, all the following configurations have to be saved into the EEPROM ("Saving parameters into EEPROM").

SDO access on objects

SDO communication can be used to read or write on objects:



Read object

The structure of a SDO message is:
Client (master) to server (encoder):

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	40h	04h	60h	00h	00h	00h	00h	00h

Example SDO master to encoder

The payload of the SDO is 4 bytes of data (d1d2d3d4):

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
560h+ID	8	43h	04h	60h	00h	d4	d3	d2	d1

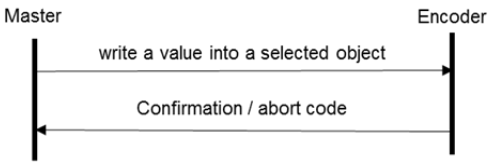
Example SDO answer

Following table shows the overview of the command values:

Command	Type	Description
22h	Write command	Parameter to encoder
23h	Write command	4 Byte Parameter to encoder
27h	Write command	3 Byte Parameter to encoder
2Bh	Write command	2 Byte Parameter to encoder
2Fh	Write command	1 Byte Parameter to encoder
60h	Acknowledge	Parameter received
40h	Read command	Parameter from Encoder
42h	Response	Parameter to SDO master
43h	Response	4 Byte Parameter to SDO master
47h	Response	3 Byte Parameter to SDO master
4Bh	Response	2 Byte Parameter to SDO master
4Fh	Response	1 Byte Parameter to SDO master
80h	Abort code	Failure / Failure code
41h	Response	SDO segmented transfer started (see CiA 301)

Command definitions

Writing an object:



Write object

The following example shows the structure of a SDO telegram:
Master sends 1 byte of data (d1) to the encoder:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Fh	00h	21h	00h	d1	00h	00h	00h

Example SDO send by master

The encoder acknowledges without data bytes:

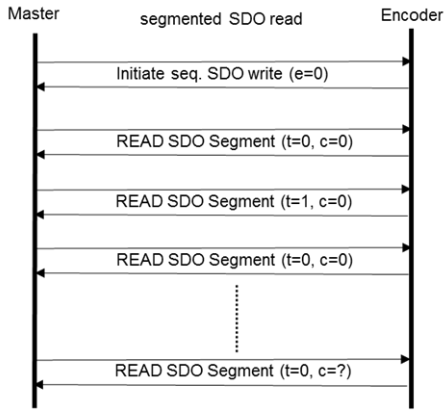
CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
560h+ID	8	2Fh	00h	21h	00h	00h	00h	00h	00h

Example SDO answer

SDO access on objects larger than 4 bytes

As seen in other paragraph the payload of a single (expedited) SDO is 4 byte. For larger data there is the possibility of a “normal” segmented SDO transfer or block transfer for up to 127 segments of 4 byte. For example to read the “high precision preset position value (Object 6008h)” or perform a “high precision preset (Object 6009h)”, the segmented SDO transfer must to be used.

Segmented read access on an object:



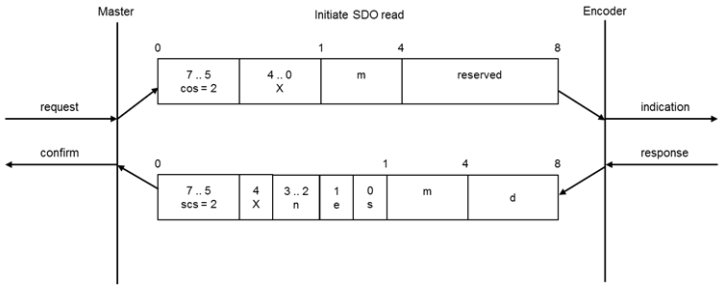
Segmented SDO read access

In the following example the 8 byte “High Precision Position Value” (object 6008h) is read:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	40h 0100 0000b ccs=2, e=0, s=0	08h	06h	00h	d1	00h	00h	00h

SDO read request on object 6008h

Initializing segmented read access:



ccs	client command specifier	2 = initiate read (upload) request
scs	server command specifier	2 = initiate read (upload) response
n	indicates that bytes [8-n,7] don’t contain segmented data	only valid if e=1 and s=1, otherwise 0
e	transfer type	0 = segmented transfer 1 = expedited transfer
s	size indicator	0 = data set size not indicated 1 = data set size indicated
m	multiplexor	index/sub index of data to be transferred
d	data	e=0, s=0 -> d is reserved. e=0, s=1 -> d = number of bytes to be read. e=1, s=1 -> d = data of length 4-n to be read. e=1, s=0 -> d = unspecified number of bytes to be read
X	not used	always 0
	reserved	reserved for further use, always 0

Abbreviations used in the figure

The encoders confirms a segmented SDO transfer of 8 bytes data:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
560h+ID	8	41h 0100 0000b scs=2, e=0, s=1	08h	06h	00h	08h	00h	00h	00h

Confirm SDO read access of object 6008h

Read SDO segment:

Read SDO segment

ccs	client command specifier	3 = read (upload) segment request
scs	server command specifier	0 = read (upload) segment response
t	toggle bit	must alternate for each subsequent segment with t=0 for the first segment. Equal for each pair of request and response
c	more segments indicator	0 = more segments to be read (uploaded) 1 = no more segments to be read (uploaded)
seg-data	segment data	at most 7 byte of segment data
n	number of bytes that don't contain segment data	bytes [8-n;7] don't contain segment n = 0 if no segment size indicated
X	not used	always 0
	reserved	reserved for further use, always 0

Abbreviations used in the figure

Then the first segment is requested:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	60h 0110 0000b ccs=3, t=0	00h	00h	00h	08h	00h	00h	00h

Read of first segment

The encoder answers with the first data segment:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
580h+ID	8	00h 0000 0000b scs=0, t=0, n=0, c=0	data	data	data	data	data	data	data

Answer with first segment

Then the next segment is requested:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	70h 0111 0000b ccs=3, t=1	00h	00h	00h	08h	00h	00h	00h

Request next segment

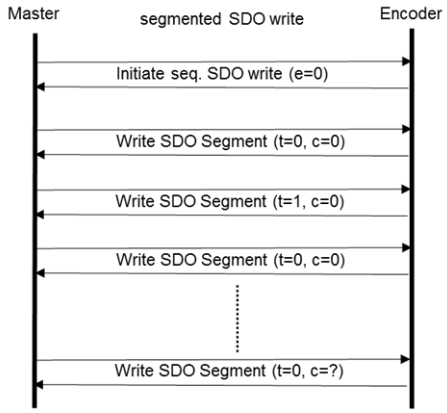
The encoder answers with the next data segment:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
580h+ID	8	1Dh 0001 1101b scs=0, t=1, n=6, c=1	data	x	x	x	x	x	x

Answer with the next segment

Within this segment the encoder indicates that this was the last data segment and that only the first data byte contained valid data. The 7 data bytes of the first segment and the single valid data byte of the data bytes represent the 8 byte "High Precision Position value" (object 6008h).

Segmented write access on an object



Segmented SDO write access

The next example shows how to use segmented SDO to write an 8 byte value into the "High precision preset value" (object 6009h). This preset value will set the corresponding "High Precision Position value" (6008h) to the designated value:
SDO write request for 8 bytes of data on object 6009h:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	21h 0010 0001b ccs=1, e=0, s=1	09h	06h	00h	08h	00h	00h	00h

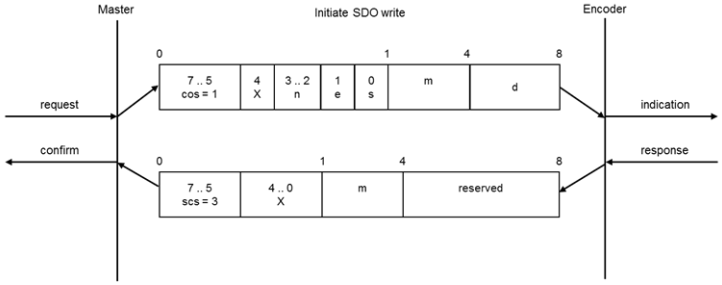
SDO write access of object 6009h

The encoder confirms the segmented SDO transfer and requests the first segment:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
580h+ID	8	60h 0110 0000b scs=3	09h	06h	00h	08h	00h	00h	00h

Acknowledgement of write access of object 6009h

Initializing segmented write access:



Initiate SDO write

ccs	client command specifier	1 = initiate write (download) request
scs	server command specifier	3 = initiate write (download) response
n	indicates that bytes [8-n,7] don't contain segmented data	only valid if e=1 and s=1, otherwise 0
e	transfer type	0 = segmented transfer 1 = expedited transfer
s	size indicator	0 = data set size not indicated 1 = data set size indicated
m	multiplexor	index/sub index of data to be transferred
d	data	e=0, s=0 -> d is reserved. e=0, s=1 -> d = number of bytes to be read. e=1, s=1 -> d = data of length 4-n to be read. e=1, s=0 -> d = unspecified number of bytes to be read
X	not used	always 0
	reserved	reserved for further use, always 0

Abbreviations used in the figure

Then the first data segment is send:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	00h 0000 0000b ccs=0, t=0, n=0, c=0	data	data	data	data	data	data	data

Send first segment

The encoder confirms and requests the next segment:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
580h+ID	8	20h 0010 0000b scs=1, t=0	data	data	data	data	data	data	data

Acknowledgement send by the encoder

Write SDO segment:

Write SDO segment

ccs	client command specifier	0 = write (download) segment request
scs	server command specifier	0 = write (download) segment response
t	toggle bit	must alternate for each subsequent segment with t=0 for the first segment. Equal for each pair of request and response
c	more segments indicator	0 = more segments to be read (uploaded) 1 = no more segments to be read (uploaded)
seg-data	segment data	at most 7 byte of segment data
n	number of bytes that don't contain segment data	bytes [8-n;7] don't contain segment n = 0 if no segment size indicated
X	not used	always 0
	reserved	reserved for further use, always 0

Abbreviations used in the figure

Now the next data segment can be send:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	1Dh 0001 1101b ccs=0, t=1, n=6, c=1	data	x	x	x	x	x	x

Send next segment

Within this segment it is indicated that this was the last data segment and that only the first data byte contained valid data.

The encoder confirms:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
580h+ID	8	20h 0010 0000b scs=1, t=1	data	x	x	x	x	x	x

Acknowledgement send by the encoder

The 7 data bytes of the first segment and the single valid data byte of the data bytes represent the 8 byte "High Precision Position Preset Value" (object 6009h).

Baudrate selection

The encoders provide an automatic baudrate detection. It is also possible to use a fixed baudrate which can be set by either LSS (as described above) or SDO. The configuration of the encoder is only possible in Pre-Operational mode.
To alter the baudrate the object 2100h into Sub-Index 00h has to be changed. This can be achieved with a simple SDO write command with the target baudrate as data.

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Fh	00h	21h	00h	Baud	00h	00h	00h

SDO command - set baudrate

The following values represent the valid baud rates:

Value	Baudrate
0	1 Mbit/s
1	800 kbit/s
2	500 kbit/s
3	250 kbit/s
4	125 kbit/s
5	100 kbit/s
6	50 kbit/s
7	20 kbit/s
8	10 kbit/s
9	Auto

Baudrate coding

The new baudrate will become effective after a reset of the encoder (hard reset or NMT reset). Writing on object 2100h is not protected and the change will be immediately stored in the internal EEPROM. It is not necessary to perform a "save parameters".

Node-ID selection

It is possible to change the Node-ID of the encoder by SDO. To set the Node-ID the object 2101h, sub-Index 00h, has to be changed (only possible in Pre-Operational state) with a simple SDO write command:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Fh	01h	21h	00h	Node	00h	00h	00h

Node_ID selection

Valid Node-IDs can be:

Encoder numer (d)	Node-ID (h)
1	01
2	02
...	...
127	7F

Valid Node-IDs

The new Node-ID will become effective after an encoder reset (hard reset or NMT reset). Writing on object 2101h is not protected and the change will be immediately stored in the internal EEPROM. It is not necessary to perform a “save parameters”.
Changing the Node-ID automatically adjusts the PDO and EMCY COB-IDs. After the first manual storage, they are set to their current value and will be no longer automatically adjusted. Performing the “Restore Defaults” command will re-enable automatic adjustment.

Basic NMT commands

This subsection describes several basic NMT commands. Basic information are available in the CANopen protocl information section.
To set the encoder into **Operational state**, the “Start remote node” command is used:

CAN-ID	DLC	Command byte	Node-ID
0	02h	01h	0 ... 127d

NMT command - Start remote node

To change the encoder into **Stopped state**, the “Stop remote node” command is used:

CAN-ID	DLC	Command byte	Node-ID
0	02h	02h	0 ... 127d

NMT command - Stop remote node

To switch the encoder into **Pre-Operational state**, the “Enter Pre-Operational State” command is used:

CAN-ID	DLC	Command byte	Node-ID
0	02h	80h	0 ... 127d

NMT command - Enter Pre-operational state

A **reset of communication** with a change into Pre-Operational after re-initialisation will be achieved by:

CAN-ID	DLC	Command byte	Node-ID
0	02h	82h	0 ... 127d

NMT command - Reset node communication

To perform a **soft reset** of the encoder, the "Reset Remote Node" is used. After the reset the encoder will send his boot-up message and enter Pre-Operational by default:

CAN-ID	DLC	Command byte	Node-ID
0	02h	81h	0 ... 127d

NMT command - Reset remote node

Heartbeat settings

To configure and start the producer heartbeat (e.g. heartbeat every 5000 milliseconds; 5000d=1388h) use SDO on object 1017h:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Bh	17h	10h	00h	88h	13h	00h	00h

Example of heartbeat setting

This is the structure of a heartbeat message:

CAN-ID	DLC	Data/Remote	Byte0
600h+ID	1	d	NMT-Status

Structure of a heartbeat message

NMT-state;

NMT-Status	Code
Boot-up	00h
Stopped	04h
Pre-Operational	7Fh
Operational	05h

Heartbeat NMT-state coding

PDO Configuration

PDO parameters

Four PDOs can be parameterised. The configuration of the PDO payload is called "PDO mapping". The default configuration is:

Object	PDO	Default configuration	Mapped process data
1800h	PDO1	Asynchronous / on change of position value	Position-value
1801h	PDO2	Syncronous / on every SYNC	Position-value
1802h	PDO3	Syncronous / on every SYNC	High precision-value
1804h	PDO4	Disabled	-

Default PDO configuration

There are five different types of transmission for every PDO:

Sub-Index 2	Sub-Index 5	Description
01h-F0h	not necessary	PDO synchronous / on a SYNC
FFh	0000h	PDO disabled
FEh	0001h-FFFFh	PDO asynchronous / triggered by event timer AND change in position value
FEh	0000h	PDO asynchronous / triggered by change of position value
FFh	0001h-FFFFh	PDO asynchronous / triggered by event timer

Selectable PDO transmission types

Please remember that parameters can be changed in Pre-Operational only and have to be saved into EEPROM.
To completely disable a PDO, the MSB of the PDO-COBID object must be changed:

PDO	Object	COB-ID object PDO enabled	COB-ID object PDO disabled
1	1800h	4000 0181h	C000 0181h
2	1801h	4000 0281h	C000 0281h
1	1802h	4000 0381h	C000 0381h
1	1803h	4000 0481h	C000 0481h

PDO deactivation

For example PDO1 shall be disabled by this SDO write command:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	00h	18h	01h	81h	01h	00h	C0h

Example of PDO1 deactivation

Advanced parameterisation of the PDO COB-ID (objects 1800h-01h, objects 1801h-01h, objects 1802h-01h, objects 1803h-01h) is possible. As long as no "Save communication objects" or "Save all parameters" has been performed, a change of the Node-ID will automatically effect the COB IDs.
After a save command, the PDO COB-IDs have to be changed manually or perform a "Restore all parameters".

Synchronous PDO

A PDO can be configured for synchronous transmission, e.g. to respond on a SYNC message. The sub index 2 of the transmission type parameter determines after which number of SYNCs received the PDO will be transmitted. For example PDO1 is configured 01h in 1800h-02h:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Fh	00h	18h	02h	01h	00h	00h	00h

Parametrization of PDO1 Sub-Index 2

Transmission type for PDO1 is now synchronous. In Operational state the PDO1 will be sent as a response on every SYNC message received.

Asynchronous PDO

Cyclic (triggered by internal event timer):

PDOs can be configured for asynchronous cyclic transmission. Therefore the transmission type in object 1800h-02h (respectively 1801h-02h, 1802h-02h, 1803h-02h) has to be set to FFh. Sub index 5 of the same object is the cycle time in milliseconds.
Example: PDO1 transmitting asynchronously cyclic:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Fh	00h	18h	02h	01h	00h	00h	00h

Parametrization of PDO1 Sub-Index 5

Example: PDO1 with a cycle time of 30 milliseconds (1Eh):

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Bh	00h	18h	05h	1Eh	00h	00h	00h

Parametrization of PDO1 Sub-Index 5 to 30ms

PDO1 is now in asynchronous mode and will be sent every 30 milliseconds when the encoder is in Operational state.

Triggered by a manufacturer specific event (change of position value):

To use this transmission type, sub-index 2 has to be FEh and the event timer in sub-index 5 has to be disabled (00h), for example:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Fh	00h	18h	02h	FEh	00h	00h	00h

Parametrization of PDO1 Sub-Index 2

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Bh	00h	18h	05h	01h	00h	00h	00h

Parametrization of PDO1 Sub-Index 5

After resetting, the encoder's PDO1 will be in asynchronous mode and be sent out if the measuring value changes.
The use of this setting may cause heavy bus load. Therefore the recommendation is to use of synchronous or timer triggered transmission.

Variable PDO-mapping

Variable PDO-mapping means that the PDO payload can be configured by the user. This mapping must match between encoder and receiver. The maximum payload for a PDO is 8 bytes. The mapping is also limited by the size of the objects to be mapped. E.g. the "position value" (4 bytes), the "speed value" (2 bytes) and the "acceleration" value (2 bytes) can be mapped into the same object. Due to the fixed size of a CAN frame this produces less bus load than transmitting the three objects by 3 individual PDOs. This table shows a possible PDO mapping:

Object	Sub-index	Datatype	Size	Description
6004h	00h	Unsigned32	4 byte	Position value
6030h	01h	Integer16	2 byte	Speed value
6040h	01h	Integer16	2 byte	Acceleration value

Example of mapping table

The data 1, 2 and 3 are spread over the PDOs 8 payload bytes. The current payload is 4byte + 2 byte + 2byte = 8 byte. So the PDO is used with 100% efficiency.
The resulting PDO has this structure:

PDO1:

CAN-ID	DLC	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
180h+ID	8	1d	1c	1b	1a	2b	2a	3b	3a

Structure of PDO1

1a, 1b, 1c, 1d = 4 bytes of information 1; 2a, 2b = 2 bytes of information 2; 3a, 3b = 2 bytes of information 3.

To use the PDO mapping the mapping parameters for the transmit PDO have to be configured (see object dictionary).

- Step 1: Delete current mapping
- Step 2: Re-mapping the PDO
- Step 3: Activating the new mapping

For example to change the PDO1 mapping you have to access the PDO1 mapping parameter object 1A00h.

Step 1: delete current mapping

First the sub-index 0 of the Mapping parameter object has to be set to zero:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Fh	00h	1Ah	00h	00h	00h	00h	00h

Mapping parameter

The encoder is now ready for remapping.

Step 2: remapping the PDO

Mapping of the position value: (No.:1 (Size 32 bit = 20h) into object 1A00h sub-index 1 for PDO1):

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	00h	1Ah	01h	20h	00h	04h	60h

Mapping position value

The SDO command contains the object to be mapped and its size (object 6004h, sub-index 0, Size 20h = 4 Byte).

Mapping of speed value (No.:2 (Size 16 bit = 10h) into object 1A00h sub-index 2 for PDO1):

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	00h	1Ah	03h	10h	01h	30h	60h

Mapping speed value

The SDO command contains the object to be mapped and its size: (Object 6030h, sub-index 1, Size 10h = 2 Byte).

Mapping of acceleration value (No.:3 (Size 16 bit = 10h) into object 1A00h sub-index 3 for PDO1):

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	00h	1Ah	03h	10h	01h	40h	60h

Mapping acceleration value

The SDO command contains the object to be mapped and its size: (Object 6040h, sub-index 1, Size 10h = 2 Byte).

Step 3: activate the new mapping

To activate the new mapping, the new number of mapped objects must be written into sub-index 0 of the mapping parameter object. In our example three objects are mapped, therefor sub-index 0 has to be set to 03h.:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Fh	00h	1Ah	00h	03h	00h	00h	00h

Mapping parameter - activate new mapping

The re-mapping of PDO1 is now completed and valid, but it should be saved into the EEPROM.

Changing resolution and direction

To change resolution and direction of the encoder the scaling option has to be activated; while activating the scaling the counting direction (clockwise (CW) or counter-clockwise (CCW) can be changed in one step (default setting is CW)).
The counting direction is referred to the movement of the axis when looking onto the flange side of the encoder.

The object for this configuration is 6000h sub-index 00h. Here is the list of possible settings:

Code Byte0	Scaling	Direction
00h	OFF	Clockwise (CW)
01h	OFF	Counter-clockwise (CCW)
04h (default)	ON	Clockwise (CW)
05h	ON	Counter-clockwise (CCW)

Counting directions and scaling parameters

This is an example how to set the “Operating parameters” object 6000h to “scaling ON” and “CCW”:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	2Bh	00h	60h	00h	05h	00h	00h	00h

Example of setting operating parameters

The encoder responds with a standard SDO acknowledge. Now scaling is active and the two scaling parameters “measuring range per revolution” and “total measuring range” are applied. Singleturn resolution and total measuring range can now be changed.

- the measuring range per revolution or singleturn resolution is the number of units (bit) per revolution.
- the total measuring range is the singleturn resolution multiplied with the number of countable revolutions (multiturn resolution).

Example: Singleturn resolution: 4096 codes per revolution = 12 bit = 10 00h
Total measuring range: 536 870 912 codes = 29 bit = 20 00 00 00h
=> Max Multiturn resolution: 29 Bit - 12 Bit = 17 Bit = 131072 revolutions (02 00 00h)

The singleturn resolution is editable in object 6001h:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	01h	60h	00h	00h	10h	00h	00h

Change of singleturn-resolution by SDO

00 00 10 00h represents the designated singleturn resolution. The encoder responds with a SDO acknowledge.

The total measuring range can be changed similarly by object 6002h. In the example a 29 bit total measuring range is selected, with a 12 bit singleturn resolution 17 bit rotations are counted before returning to zero:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	02h	60h	00h	00h	00h	00h	20h

Change of total measuring range by SDO

20 00 00 00h is the designated total measuring range.

Singleturn resolution and total measuring range do not have to match the bit grid. Every value between 1 and the maximum is valid.
The total measuring range cannot be less than the singleturn resolution. The result of an invalid setting will be an abort code.

Position preset

With object 6003h the encoder position can be shifted to a preset value. E.g. the zero position of your application can be set without time-consuming mechanical alignment. Just mount the encoder and set the preset object 6003h to the designated position value (p1-p4):

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	03h	60h	00h	p1	p2	p3	p4

Set position preset

to set the zero position: p1, p2, p3, p4 = 00h, 00h, 00h, 00h.
The use of PDOs to check the current position value is not needed, simply perform a SDO read access on the position value object 6004h:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	40h	04h	60h	00h	00h	00h	00h	00h

Check current position

The encoder will respond with the current position value.

Position value filtering

The encoder provides an internal filtering for the position value.
Sub-index 1 of object 2105h is the filter parameter for the internal “ IIF”-filter (infinite impulse response filter).
01h for the filter parameter deactivates the filter, the maximum value is 04h. A filtered position value is more stable at the cost of less dynamic.

Change speed-integration and speed scaling

The encoder uses an “integration time” to calculate the speed value. This time interval can be adjusted by object 2105h, sub-Index 2.
The unit for this time is milliseconds. The default value of 1000 milliseconds is suitable for most applications.
The change of the integration time will result in a more or less dynamic behaviour of the speed value, similar (but independent) to the filtering of the position value.
The speed scaling can be edited by object 2106h. The Sub-Indices 1 (= numerator) and 2 (= denominator) form a scaling factor (here: “z”) for the speed scaling. Default value is “1”. The speed value is always given in Increments per second.

Object 2106h is a signed16 value with the limits of ±32767 representing ±120 rotations per second.
For example the speed shall be scaled to a maximum of ±2500 rpm:

z = scaling factor	z= k/n	(1)
n = max rotations per second	z= 120/2500	(2)
k = calculation factor = 120	z= 6/125	(3)

So object 2106h-01h must be set to 6d = 06h and 2106h-01h set to 125d=7Dh, so the limits of ±32767 are scaled to ±2500 U/min.
Applying this scaling, the limits ±32767 corresponds with ±2500 rpm.

Frequency limit

If the speed value exceeds the frequency limit 2107h a warning flag is set (no EMCY).
The valid area is 1 to 65535 representing the maximum allowed rotation speed (e.g. 2520 rpm = 42 rotations per second = 002Ah as frequency limit).

CAM configuration

This section gives an example how to configure the cam-channel, please note that the configuration must be done in Pre-Operational state.

e.g. CAM1 0°...180°, CAM2 180° ... 360°, CAM3 0° ... 60°, this means for single cams:

CAM	Angular area	Lower CAM-limit	Upper CAM-limit	Hysteresis
1	0° ... 180°	0	2048	0
2	180° ... 360°	2049	4095	0
3	0° ... 60°	0	682	0

Example of CAM configuration

To enable the individual cams the CAM-enable-register (object 6301h-01h) is used. For example the setting 00000111b = 07h enables the first three cams.

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	5	2Fh	01h	63h	01h	07h	00h	00h	00h

Enable first three cams

Now the cam high-limit 1, 2, and 3 can be set as in the table above:

CAM 1 = 2048d = 0800h

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	20h	63h	01h	00h	08h	00h	00h

CAM high-limit 1

CAM 2 = 4095d = 0FFFh

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	21h	63h	01h	FFh	0Fh	00h	00h

CAM high-limit 2

CAM 3 = 682d = 02AAh

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	23h	63h	01h	AAh	02h	00h	00h

CAM high-limit 3

The setting of the CAM low-limit 1, 2 and 3 is similar:

CAM 1 = 0d = 00h

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	10h	63h	01h	00h	00h	00h	00h

CAM low-limit 1

CAM 2 = 2049d = 0801h

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	11h	63h	01h	01h	08h	00h	00h

CAM low-limit 2

CAM 3 = 0d = 00h

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	12h	63h	01h	00h	00h	00h	00h

CAM low-limit 3

In this example the CAM-Hysteresis shall be 0, so there is no change necessary, with the CAM-Polarity-Register the polarity of the cams can be inverted.
Using object 6300h Sub-Index 1 the CAM-state-register can be read.
The CAM-state-register is also PDO mappable. For more details see “CAM-state-register” paragraph.
To save the configuration into the EEPROM, see "Saving parameters into EEPROM” paragraph.

Storage of parameters

Saving parameters into EEPROM

Non-volatile storage of parameters using object 1010h:

Sub-index	Access mode	Description
0	Constant, read only	Number of objects
1	Write only	Save all parameters
2	Write only	Save communication objects
3	Write only	Save application objects
4	Write only	Save manufacturer objects

Saving parameters

To trigger the storage operation the "ASCII" value for "save" (in hex: 65766173h) has to be written into the dedicated sub-index.
E.g. “Save all Parameters”:

CAN-ID	DLC	Command	Object L	Object H	Sub-index	Byte0	Byte1	Byte2	Byte3
600h+ID	8	23h	10h	10h	01h	73h	61h	76h	65h

Save all parameters

Restoring default parameters from EEPROM

Restoring default settings by using object 1011h:

Sub-index	Access mode	Description
0	Constant	Number of objects
1	Write only	Restore all parameters
2	Write only	Restore communication objects
3	Write only	Restore application objects
4	Write only	Restore manufacturer objects

Restoring parameters

To restore the default settings the "ASCII" value "load" (in hex: 64616F6Ch) has to be written to the dedicated sub-index of the object.
Please take note that the baudrate and node-ID settings, as well as the customer data object, will not be restored.

GENERAL CAN INFORMATIONS

CAN physical and transport layer

CAN is a field bus. It operates with the CSMA/CA (Carrier Sense Multiple Access / Collision Avoidance) method. It means that collisions during bus access are avoided by a so called bitwise arbitration. The bits are coded NRZ-L (Non Return to Zero - Low).

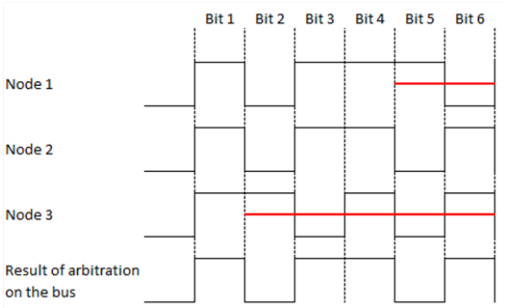
A cyclic redundancy check (CRC) and other safety mechanisms provide a secure transmission. For synchronisation a mechanism called "bit stuffing" is used. CAN is a multi-master system, i.e. several equal bus nodes can be connected without a bus master supervising the communication. In principle a CAN bus can be realized with copper wire or in fibre optic cable.

The common CAN implementation with copper wire operates with differential signals, transmitted via two wires: CAN_{HIGH}, CAN_{LOW}. Therefore CAN has a good common mode rejection ratio.

Data is transmitted with bits that can either be dominant or recessive. The dominant (0) always overwrites the recessive (1).
The topology of a CAN network is a line, which can be extended by stubs. The maximum length of a stub is limited to 0,5m.

The network always has to be terminated on both ends with 120 Ohm each (between CAN_{HIGH} and CAN_{LOW}). Other locations or values are not allowed.

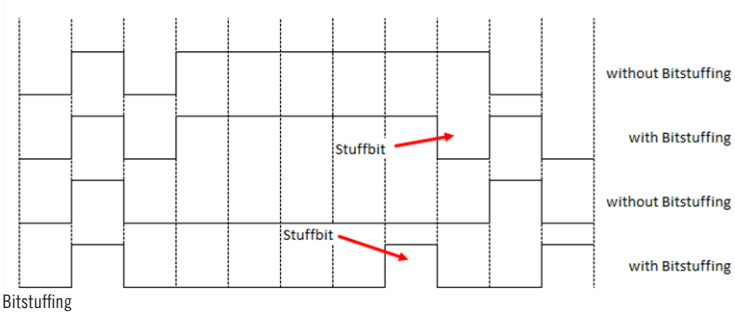
The arbitration mentioned before is used to control the bus access of the nodes by prioritization of the CAN-Identifier of the different messages.
Every node monitors the bus. If more than on node wants access on the bus, the node with the highest priority of the messages ID succeeds and the other nodes retreat until there is "silence" on the bus (see below example). Technically the first dominate bit of the ID send overwrites the corresponding recessive bit of the other IDs. In case that more than one node uses the same CAN-ID an error occurs only at a collision within the rest of the frame.
On principle a CAN-ID should only be used by a single node.



Example of arbitration

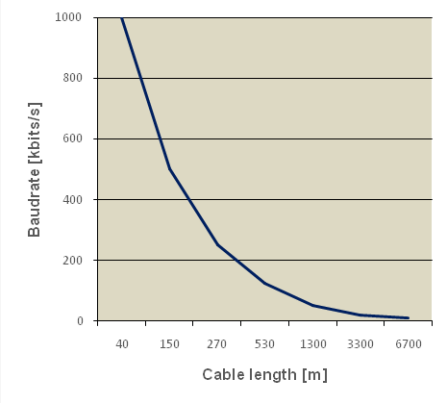
Due to the arbitration there is a ranking of the messages. The message with the lowest ID has the highest priority and therefore it has almost instant access on the bus. The exception is that an ongoing transmission will not be interrupted. So time critical messages should be assigned to the high priority CAN-IDs, but even then there is no determination in the time of transmission (non-deterministic transmission).

For the arbitration all nodes have to be synchronised. Due to the lack of a separate clock signal, the transmission of many identical bits in line would lead to the loss of synchronisation. The so called bit-stuffing is used to prevent this case. After five equal bits a complementary bit will be inserted into the transmission (the application will not notice). So the nodes can keep up resynchronising on the bit edges (see below figure).



Bitstuffing

A CAN network can operate with baud rates up to 1 Mbit/s.
Due to the necessary synchronisation of the nodes, the maximum delay caused by the length of the cable has to be limited.
The limitation corresponds with the baudrate. There is a common recommendation of the maximum cable length at several baud rates:

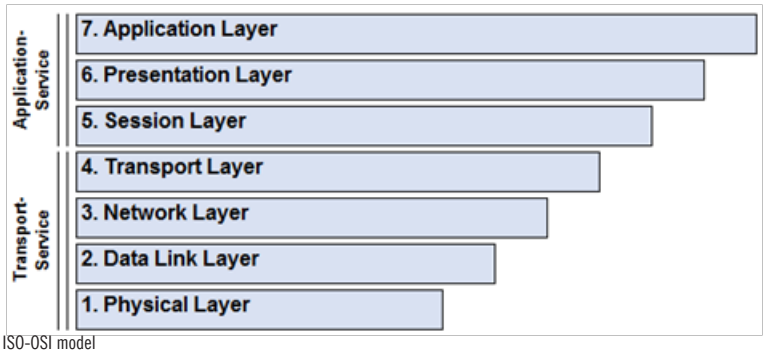


CAN baud rates and recommended cable length limits

Baud rates (kbit /s)	Cable lenght (m)
10	6700
20	3300
50	1300
125	530
250	270
500	130
1000	< 40

CANopen

CANopen is a specified higher protocol (layer 7 protocol)



ISO-OSI model

With CANopen it is possible to transfer larger amounts of data, emergency telegrams and process data.
CANopen describes how the communication is performed. That means that parameters to configure a device are transmitted in a defined form (profile).
A CANopen profile defines objects representing the different functions of a device. These objects form a table called object dictionary.
The communication profile defines the basic services and parameters of a CANopen device (e.g. service data objects SDOs, process data objects PDOs, used CAN-IDs, etc.).
The device profile defines the specific functions of a device family (e.g. encoders, i/o devices, ...).
For encoders the device profile is the encoder profile CiA 406.

Specifications and profiles

Overview

The CANopen specifications were defined by the CiA in Draft Standards. Concerning the Eltra encoders the following specifications are from special importance:

Specification	Description
CiA 301	Application Layer and Communication Profile
CiA 303	Cabling/pin assignment, Representation of units, Indicator specification
CiA 305	Configuration of baudrate and Node-ID via LSS
CiA 306	Electronic Data Sheet (EDS)
CiA 406	Device / Application profile

Draft standards

Mechanisms of communication

There are several different CANopen communication services:

SDO Service Data Object

It is used to access to the object dictionary. There is one single SDO-channel.
Two identifiers are assigned to the SDO channel, one for each direction of transmission.
For SDO the 8 byte CAN frame is divided into 1 byte command, a multiplexor of 2 byte index and 1 byte sub index of the object dictionary, and 4 byte of payload. For bigger payloads either segmented or block transfer is used.
A SDO transmission will always be acknowledged by the receiver. In case of a failure an “abort message” is send.
The internal delay time of the Eltra encoders is 1 millisecond maximum.

PDO Process Data Object

It is used for transmission of process data. The Eltra encoders provide up to four PDOs.
A PDO uses the full length of the data area of a CAN frame (8 bytes) for the process data without additional overhead.
PDOs will not be acknowledged and are suitable for time critical applications.
By using the full 8 bytes for data, there is no additional information about transmitted objects. Therefore the PDO producer and the PDO consumer have to define the PDO-mapping.
PDOS can be sent on different ways:

- On request: a node sends a RTR frame to ID of the designated PDO and the encoder returns the PDO. (The CiA strongly recommends not to use RTR frames. Therefore RTR is not supported by Eltra encoders)
- Synchronously: on the reception of a SYNC message the node send its PDOs.
- Asynchronously: the sending of the PDOs is triggered by an internal event (e.g. the internal event timer).

Object dictionary

The object dictionary lists all data types, objects and functions of the communication and the device profile. There are also manufacturer specific objects listed. The objects are addressed by 16-bit indices (lines) and 8-bit sub-indices (columns).

Index(hex)	Object description
0000	Reserved
0001 001F	Static data types
0020 003F	Complex data types
0040 005F	Manufacturer specific data types
0060 007F	Profile specific static data types
0080 009F	Profile specific complex data types
00A0 0FFF	Reserved
1000 1FFF	Communication profile objects
2000 5FFF	Manufacturer specific objects
6000 9FFF	Objects from the "Standard device profiles"
A000 AFFF	Network variables
B000 FFFF	Reserved / system variables

Structure of the object dictionary

Network management (NMT)

A CANopen network always needs a network management master. The NMT master controls the NMT states of all connected nodes.
A node can be switched into three different states:

- Pre-Operational
- Operational
- Stopped

After a CANopen node is switched on and the communication and the internal application is initialised, the node switches into pre-operational state. From this state the NMT-Master can switch the node into the other states. To show that a node is ready after boot up, it sends a “boot-up message”. These messages uses the CAN-ID of the Emergency service (EMCY) and the message is permanently associated with the Node-ID.

Description of the NMT-states

In PRE-OPERATIONAL state SDO communication is enable while PDO communication is disabled.

Object	Communication enabled
SDO	yes
PDO	no
NMT	yes
SYNC	no
EMCY	yes
Heartbeat	yes

Available communication – Pre-Operational

In OPERATIONAL state device can send and receive PDOs.

Object	Communication enabled
SDO	yes
PDO	yes
NMT	yes
SYNC	yes
EMCY	yes
Heartbeat	yes

Available communication – Operational

In STOPPED state the communication is almost completely disabled, the device only reacts on NMT commands (e.g. start node).

Object	Communication enabled
SDO	no
PDO	no
NMT	yes
SYNC	no
EMCY	no
Heartbeat	yes

Available communication – Stopped

Heartbeat and Node-Guarding

There are two possible ways to supervise the operational availability of a CAN node during operation, Heartbeat or Node-Guarding
The heartbeat protocol is independent from the master and it is the recommended mechanism (Eltra suggests to use it). The device sends autonomous and cyclic a "life" message.
When using the Node Guarding protocol, the NMT master sends RTR frames to the slaves, which have to answer within a defined time.
If the answer is missing, this is detected by the master. This protocol leads to a high dependence on the master, that is the reason why it is not recommended.
A variation of the Heartbeat is the Bootup-Message. This type is sent out once the encoder is started and includes no information (Data is 00h). Only by interpreting the COB-ID of the message, the senders Node-ID is obvious (COB-ID = 700h + Node-ID).

Emergency messages

Failures of a CAN node are announced by emergency messages (EMCY message). The EMCY message contains an error code identifying the problem.
A node also can be configured to send no EMCYs messages.

Via Guido Salvagnini, 17
36040 - Sarego - Vicenza - ITALY
T: +39 0444 436489
F: +39 0444 835335
eltra@eltra.it | www.eltra.it