Smart encoders & actuators

User's guide

Ax58 CB
Ax58S CB
AxC58 CB
CC-CB / CC-CB-C

DS406 encoder profile

- AS58 singleturn encoder with resolution up to 13 bits
- AM58 multiturn encoder with resolution up to 25 bits
- Connection cap with M12 connectors or PG cable outlets
- CANopen in compliance with DS 301 and DS 406 profiles

Suitable for the following models:

- AS58, AS58S CB + CC-CB/CC-CB-C
- ASC58, ASC59, ASC60 CB + CC-CB/CC-CB-C
- AM58, AM58S CB + CC-CB/CC-CB-C
- AMC58, AMC59, AMC60 CB + CC-CB/CC-CB-C

General Contents

Preliminary information	9
1 - Safety summary	20
2 - Identification	22
3 - Mounting instructions	23
4 - Electrical connections	28
5 - Quick reference	37
6 - CANopen® interface	41
7 – Setting-up	81
8 – Default parameters list	85

General contents

User's guide 1

General contents 3

Subject index 6

Typographic and iconographic conventions 8

Preliminary information 9

Glossary of CANopen terms 10

1 - Safety summary 20

2 - Identification 22

3 - Mounting instructions 23

3.1 Solid shaft encoders 23

3.1.1. Customary installation 23

3.1.2 Installation using fixing clamps (code LKM-386) 24

3.1.3 Installation using a mounting bell (code PF4256) 24

3.2 Hollow shaft encoders 25

3.2.1 ASC58, AMC58 25

3.2.2 ASC59, AMC59 26

3.2.3 ASC60, AMC60 27

4 - Electrical connections 28

4.1 Connection cap 28

4.2 Connection cap with PGs: CC-CB 29

4.3 Connection cap with M12 connectors: CC-CB-C 30

4.4 Ground connection 31

4.5 Connection of the shield 31

4.6 Setting the baud rate: DIP A 32

4.7 Setting the node address: DIP B 33

4.8 RT Bus termination 34

4.9 Diagnostic LEDs 35

5 - Quick reference 37

5.1 Hardware resolution 37

5.1.1 Procedure to set the hardware resolution 37

5.2 Using the default settings 39

6 - CANopen® interface 41

6.1 EDS file 41

6.2 State machine 42

6.2.1 Initialization state 42

6.2.2 Pre-operational state 42

6.2.3 Operational state 43

6.2.4 Stopped state 43

6.3 Communication objects 43

6.3.1 Pre-defined connection set 44

6.4 NMT objects 44

6.5 Boot-up objects 45

6.6 PDO objects 45

PDO1 Cyclic mode: cyclic transmission 45

PDO2 and PDO3 SYNC mode: synchronous transmission 45

PDO4 Cyclic mode: cyclic transmission of the velocity 46

6.7 SDO objects....46
6.7.1 Command....47
6.8 Object dictionary....47
6.8.1 Communication Profile Area objects (DS 301)....49
1000-00 Device type....49
1001-00 Error register....49
1003 Pre-defined error field....49
1005-00 COB-ID SYNC message....49
1008-00 Manufacturer device name....49
1009-00 Manufacturer hardware version....49
100A-00 Manufacturer software version....50
100C-00 Guard time....50
100D-00 Life time factor....50
1010-01 Store parameters....50
1011-01 Restore default parameters....50
1014-00 COB-ID EMCY....51
1015-00 Inhibit time EMCY....51
1018 Identity object....52
1800 PDO1 parameters....52
1801 PDO2 parameters....53
1802 PDO3 parameters....55
1803 PDO4 parameters....56
1A00-01 PDO1 mapping parameter....58
1A01-01 PDO2 mapping parameter....58
1A02-01 PDO3 mapping parameter....59
1A03-01 PDO4 mapping parameter....59
6.8.2 Manufacturer Specific Profile Area objects....60
2104-00 Limit switch min.....60
2105-00 Limit switch max.....60
3000-00 Baud rate....60
3001-00 Node-ID....61
3005-00 Velocity format....62
3006-00 Velocity value....62
6.8.3 Device Profile Area objects (DS 406)....63
6000-00 Operating parameters....63
Code sequence....63
Scaling function....63
Limit switch min.....64
Limit switch max.....64
6001-00 Measuring units per revolution....64
6002-00 Total resolution....66
6003-00 Preset value....68
6004-00 Position value....70
6200-00 Cyclic timer....70
6500-00 Operating status....71
Code sequence....71
Scaling function....71
Limit switch min.....71
Limit switch max.....72

Current operating state....72
6501-00 Hardware counts per revolution....72
6502-00 Hardware number of turns....73
6504-00 Supported alarms....73
6506-00 Supported warnings....74
6507-00 Profile and software version....74
6508-00 Operating time....74
6509-00 Offset value....74
650A-01 Manufacturer offset value....74
650B-00 Serial number....75
6.9 SDO abort codes....76
6.10 Emergency objects....77
6.11 Node guarding protocol....78
7 – Setting-up....81
7.1 Setting the Operational, Pre-operational state....81
7.2 Reading the value of the physical resolution per revolution....81
7.3 Reading the number of physical revolutions....81
7.4 Setting the resolution per revolution....82
7.5 Setting the total resolution....82
7.6 Setting the operating parameters....82
7.7 Setting the preset value....82
7.8 Setting the Sync counter....83
7.9 Disabling the Sync mode....83
7.10 Enabling the Cyclic mode....83
8 – Default parameters list....85

Subject index

1

1000-00 Device type........49
1001-00 Error register........49
1003 Pre-defined error field........49
1005-00 COB-ID SYNC message........49
1008-00 Manufacturer device name........49
1009-00 Manufacturer hardware version........49
100A-00 Manufacturer software version........50
100C-00 Guard time........50
100D-00 Life time factor........50
1010-01 Store parameters........50
1011-01 Restore default parameters........50
1014-00 COB-ID EMCY........51
1015-00 Inhibit time EMCY........51
1018 Identity object........52
1800 PDO1 parameters........52
1801 PDO2 parameters........53
1802 PDO3 parameters........55
1803 PDO4 parameters........56
1A00-01 PDO1 mapping parameter........58
1A01-01 PDO2 mapping parameter........58
1A02-01 PDO3 mapping parameter........59
1A03-01 PDO4 mapping parameter........59

2

2104-00 Limit switch min........60
2105-00 Limit switch max........60

3

3000-00 Baud rate........60
3001-00 Node-ID........61
3005-00 Velocity format........62
3006-00 Velocity value........62

6

6000-00 Operating parameters........63
6001-00 Measuring units per revolution........64
6002-00 Total resolution........66
6003-00 Preset value........68
6004-00 Position value........70
6200-00 Cyclic timer........70
6500-00 Operating status........71
6501-00 Hardware counts per revolution........72
6502-00 Hardware number of turns........73
6504-00 Supported alarms........73
6506-00 Supported warnings........74
6507-00 Profile and software version........74
6508-00 Operating time........74
6509-00 Offset value........74
650A-01 Manufacturer offset value........74
650B-00 Serial number........75

C

COB-ID of PDO1........52
COB-ID of PDO4........56
COB-ID of the PDO2........53
COB-ID of the PDO3........55
Code sequence........63, 71
Current operating state........72

F

Flash memory error........77

I

Initialization........42

L

Last error occurred........49
Limit switch max........64, 72
Limit switch min........64, 71

N

Node guarding error........77
Number of occurred errors........49

O

Operational........43

P

Pre-operational........42
Previous errors occurred........49
Product code........52

R

Revision number........52

S

Scaling function........63, 71
Stopped........43

T

Transmission type........52, 54p., 57

V

Vendor-ID...52

Typographic and iconographic conventions

In this guide, to make it easier to understand and read the text the following typographic and iconographic conventions are used:

- parameters and objects both of the device and the interface are coloured in GREEN;
- alarms are coloured in RED;
- states are coloured in FUCSIA.

When scrolling through the text some icons can be found on the side of the page: they are expressly designed to highlight the parts of the text which are of great interest and significance for the user. Sometimes they are used to warn against dangers or potential sources of danger arising from the use of the device. You are advised to follow strictly the instructions given in this guide in order to guarantee the safety of the user and ensure the performance of the device. In this guide the following symbols are used:

⚠	This icon, followed by the word **WARNING**, is meant to highlight the parts of the text where information of great significance for the user can be found: user must pay the greatest attention to them! Instructions must be followed strictly in order to guarantee the safety of the user and a correct use of the device. Failure to heed a warning or comply with instructions could lead to personal injury and/or damage to the unit or other equipment.
ⓘ	This icon, followed by the word **NOTE**, is meant to highlight the parts of the text where important notes needful for a correct and reliable use of the device can be found. User must pay attention to them! Failure to comply with instructions could cause the equipment to be set wrongly: hence a faulty and improper working of the device could be the consequence.
ℹ	This icon is meant to highlight the parts of the text where suggestions useful for making it easier to set the device and optimize performance and reliability can be found. Sometimes this symbol is followed by the word **EXAMPLE** when instructions for setting parameters are accompanied by examples to clarify the explanation.

Preliminary information

This guide is designed to provide the most complete information the operator needs to correctly and safely install and operate the following encoders **fitted with CANopen interface**:

ASxxx12/CB-xx *	**(12 bit singleturn encoder)**
ASxxx13/CB-xx *	**(13 bit singleturn encoder)**
AMxxx12/4096CB-xx *	**(12 + 12 bit multiturn encoder)**
AMxxx13/4096CB-xx *	**(13 + 12 bit multiturn encoder)**

* + CANopen connection cap as follows (connection cap to be ordered separately):

CC-CB	CANopen interface with PG cable outlets
CC-CB-C	CANopen interface with M12 connectors

For any further information please refer to the product data sheet.

To make it easier to read the text, this guide is divided into two main sections.
In the first section (chapters 1 to 4) general information concerning the safety, the mechanical installation and the electrical connection as well as tips for setting up and running properly and efficiently the unit are provided.
In the second section (chapters 5, 6, and 7), both general and specific information is given on the CANopen interface. In this section the interface features and the objects implemented in the unit are fully described.

Glossary of CANopen terms

CANopen, like many other networking systems, has a set of unique terminology. Table below contains a few of the technical terms used in this guide to describe the CANopen interface. They are listed in alphabetical order. The Glossary is owned and copyrighted by the CAN in Automation international users' and manufacturers' group.

Application layer	The application layer is the communication entity of the OSI (Open System Interface) reference model. It provides communication services to the application program.
Application objects	Application objects are signals and parameters of the application program visible at the application layer API (application programming interface).
Application profile	Application profiles define all communication objects and application objects in all devices that the network consists of.
Asynchronous PDO	An asynchronous PDO is transmitted whenever a defined internal event occurs. This event may also be the elapsing of the PDO's event timer. If an asynchronous PDO is received the protocol software immediately updates the mapped objects in the Object Dictionary.
Boot-up message	CANopen communication service transmitted whenever a node enters the **Pre-operational** state after initialization.
Bus	Topology of a communication network, where all nodes are reached by passive links, which allows transmission in both directions.
Bus analyser	Tool, which monitors the bus and displays the transmitted bits. There are bus analysers available on the physical layer, the data link layer, and different application layers (e.g. CANopen or DeviceNet).
Bus arbitration	If at the very same moment several nodes try to access the bus, an arbitration process is necessary. At the end of this process, only one node has bus access. The bus arbitration process used in CAN protocol is CMSA/CD (Carrier Sense Multiple Access/Collision Detection) with AMP (Arbitration on Message Priority). This allows bus arbitration without destruction of messages.
Bus length	The network cable length between the both termination resistors. The bus length of CANopen networks is limited by the used transmission rate. At 1 Mbps the maximum length is 25 m. When using lower transmission rates, longer bus lines may be used: at 50 kbps a length of 1 km is possible.
Bus off state	The CAN controllers switch to bus off state when the TEC (transmit error counter) has reached 255. During bus off state, the CAN controller transmits recessive bits. When a CANopen

	device recovers from bus off state, it has to transmit the boot-up message and it is recommended to send an Emergency message with the appropriate error code.
CAN	Controller Area Network (CAN) is a serial bus system originally developed by the Robert Bosch GmbH. It is internationally standardized by ISO 11898-1. CAN has been implemented by many semiconductor manufacturers.
CAN protocol controller	The CAN protocol controller is part of a CAN module performing data en-/de-capsulation, bit-timing, CRC, bit-stuffing, error handling, failure confinement, etc.
CAN transceiver	The CAN transceiver is connected to the CAN controller and to the bus lines. It provides the line transmitter and the receiver. There are high-speed, fault-tolerant, and single-wire transceivers available as well as transceivers for power-line or fiber optic transmissions.
CANopen	Family of profiles for embedded networking in industrial machinery, medical equipment, building automation (e.g. lift control systems, electronically controlled doors, integrated room control systems), railways, maritime electronics, truck-based superstructures, off-highway and off-road vehicles, etc.
CANopen application layer	The CANopen application layer and communication profile is standardized by EN 50325-4. It defines communication services and objects. In addition, it specifies the Object Dictionary and the network management (NMT).
CANopen Manager	The CANopen manager is responsible for the management of the network. The CANopen manager device shall include the NMT (network management) Master, the SDO (service data object) manager, and the Configuration manager.
CANopen Safety	Communication protocol allowing transmission of safety-relevant data. The protocol requires just one physical CAN network. Redundancy is achieved by sending each message twice with bit-wise inverted content using two identifiers differing at least in two bits.
Certification	Official compliance test of components or devices to a specific standard. CiA officially certifies CANopen devices.
CiA DR 303	Draft recommendation for CANopen cabling and connector pin assignments, coding of prefixes and SI unit as well as LED usage.
CiA DS 102	Draft standard for high-speed transmission according to ISO 11898-2 using 9-pin D-sub connectors.
CiA DS 301	The CANopen application layer and communication profile specification covers the functionality of CANopen NMT (network management) Slave devices.
CiA DS 401	The CANopen device profile for generic I/O modules covers the definition of digital and analogue input and output devices.
CiA DS 404	The CANopen device profile for measuring devices and closed-

	loop controllers supports also multi-channel devices.
CiA DS 406	The CANopen device profile for encoders defines the communication of rotating as well as linear sensors.
CiA DSP 302	The draft standard proposal for programmable CANopen devices includes CANopen manager functions, dynamic SDO connections, standardized boot-up procedure for NMT Slaves as well as program download.
CiA DSP 304	The CANopen safety protocol specification is approved by German authorities and is compliant to SIL class 3 applications.
CiA DSP 305	The Layer Setting Services (LSS) specify how to set node-ID and transmission rate via the CANopen network.
CiA DSP 306	This draft standard proposal defines format and content of Electronic Data Sheets (EDS) to be used in configuration tools.
CiA DSP 308	The CANopen framework for maritime applications defines redundancy of networks including swapping mechanism for SDOs and PDOs.
CiA DSP 309	Set of gateway specifications for CANopen to Ethernet-based networks (e.g. Modbus TCP(IP)).
CiA DSP 402	The CANopen device profile for drives and motion controllers defines the interface to frequency inverters, servo controllers as well as stepper motors.
CiA DSP 405	The CANopen device and interface profile for IEC 61131-3 compatible controllers is based on the CiA DSP 302 specification using network variables to be mapped into PDOs, and function blocks for SDO services, etc.
CiA DSP 407	The CANopen application profile for passenger information systems developed in cooperation with the German VDV specifies interfaces for a range of devices including displays, ticket printers, passenger counting units, main onboard computer, etc.
CiA DSP 408	The CANopen device profile for hydraulic controllers and proportional valves is compliant to the bus-independent VDMA device profile.
CiA DSP 410	The CANopen device profile for inclinometer supports 16-bit as well as 32-bit sensors.
CiA DSP 412	The CANopen device profiles for medical equipment specify the interfaces for x-ray collimators, x-ray generators, stands and tables.
CiA DSP 413	The CANopen interface profiles for in-vehicle truck gateways specify gateways to ISO 11992, J1939, and other in-vehicle networks. The CANopen network is mainly used for truck- or trailer-based superstructures, e.g. as in garbage trucks, truck-mounted cranes, and concrete mixers.
CiA DSP 414	The CANopen device profile for weaving machines specifies

	the interface for feeder sub-systems.
CiA DSP 415	The CANopen application profile for asphalt pavers specifies interfaces to different devices used in road construction machinery.
CiA DSP 416	The CANopen application profile for building doors specifies interfaces for locks, sensors, and other devices used in electronically controlled building doors.
CiA DSP 417	The CANopen application profile for lift control specifies the interfaces for car controller, door controller, call controller and other controllers as well as for car units, door units, input panels, and display units, etc.
CiA DSP 418	The CANopen device profile for battery modules specifies the interface to communicate with battery chargers.
CiA DSP 419	The CANopen device profile for battery charger specifies the interface to communicate with the battery module.
CiA DSP 420	The CANopen device profile family for extruder downstream devices defines interfaces for puller, corrugator and saw devices.
CiA DSP 421	The CANopen device profile for railways specifies interfaces to sub-systems such as diesel engines, brake controllers, door controllers, etc.
CiA DSP 422	The CANopen application profile for municipal vehicles defines the communication of sub-systems used in garbage trucks.
CiA TR 308	This technical report specifies some timings for CANopen performance testing tools.
Client / Server communication	In a Client/Server communication the Client initiates the communication with the Server. It is always a point-to-point communication.
Client SDO	The Client SDO initiates the SDO communication by means of reading or writing to the Object Dictionary of the Server device.
COB ID	The COB ID is the object specifying the CAN message identifier and additional parameters such as valid/invalid and remote frame support.
Communication object (COB)	A communication object is one or more CAN messages with a specific functionality, e.g. PDO, SDO, Emergency, Time, or Error Control.
Communication profile	A communication profile defines the content of communication objects such as Emergency, Time, Sync, Heartbeat, NMT, etc. in CANopen.
Configuration Manager	The Configuration Manager (CMT) provides mechanisms for configuration of CANopen devices during boot-up.
Confirmed communication	Confirmed communication services require a bi-directional communication, meaning that the receiving node sends a confirmation that the message has been received successfully.

Conformance test plan	Definitions of test cases that have to be passed successfully in order to achieve conformance to a communication standard. The conformance test plan for CAN is standardized by ISO 16845.
Conformance test tool	A conformance test tool is the implementation of a conformance test plan.
Consumer	In CAN networks a receiver of messages is called a consumer meaning the acceptance filter is opened.
D-sub connector	Standardized connectors. Most common in use is the 9-pin D-sub connector (DIN 41652); its pin-assignment for CAN networks is specified in CiA DS 102.
Data link layer	Second layer in the OSI reference model providing basic communication services. The CAN data link layer defines data, remote, error, and overload frames.
Data type	Object attribute in CANopen defining the format, e.g. UNSIGNED8, INTEGER16, BOOLEAN, etc.
Default value	Object attribute in CANopen defining the pre-setting of not user-configured objects after power-on or application reset.
Device profile	A device profile defines the device-specific communication services including the configuration services in all details.
Draft Recommendation (DR)	This kind of recommendation is not fixed, but it is published. CiA's draft recommendations are not changed within one year.
Draft Standard (DS)	This kind of standard is not fixed, but it is published. CiA's draft standards are not changed within one year.
Draft Standard Proposal (DSP)	This kind of standard is a proposal, but it is published. CiA's draft standard proposals may be changed anytime without notification.
EDS checker	Software tool that checks the conformity of electronic data sheets. The CANopen EDS checker is available on CiA's web-site to be downloaded.
EDS generator	Software tool that generates CANopen electronic data sheets.
Electronic Data Sheet (EDS)	Electronic data sheets describe the functionality of a device in a standardized manner.
Emergency message	Pre-defined communication service in CANopen mapped into a single 8-byte data frame containing a 2-byte standardized error code, the 1-byte error register, and 5-byte manufacturer-specific information. It is used to communicate device and application failures.
EN 50325-4	CENELEC standard defining the CANopen application layer (version 4.0).
Entry category	Object attribute in CANopen defining whether this object is mandatory or optional.
Error code	CANopen specifies standardized error codes transmitted in emergency messages.

Error control message	The CANopen error control messages are mapped to a single 1-byte CAN data frame assigned with a fixed identifier that is derived from the device's Node ID. It is transmitted as boot-up message before entering **Pre-operational** state after initialization, and it is transmitted if remotely requested by the NMT Master (node guarding) or periodically by the device (heart-beat).
Event driven	Event driven messages are transmitted when a defined event occurs in the node. This may be a change of input states, elapsing of a local timer, or any other local event.
Event timer	The event timer is assigned in CANopen to one PDO. It defines the frequency of transmission.
Expedited SDO	This is a confirmed communication service of CANopen (peer-to-peer). It is made up by one SDO initiate message of the Client node and the corresponding confirmation message of the Server node. Expedited SDOs are used if not more than 4 byte of data has to be transmitted.
Flying Master	In safety-critical applications, it may be required that a missing NMT Master is substituted automatically by another stand-by NMT Master. This concept of redundancy is called Flying Master.
Form error	A corruption of one of the pre-defined recessive bits (CRC delimiter, ACK delimiter and EOF) is regarded as a form error condition that will cause the transmission of an error frame in the very next bit-time.
Function code	First four bits of the CAN identifier in the CANopen pre-defined identifier set indicating the function of the communication object (e.g. TPDO_1 or Error Control message).
Galvanic isolation	Galvanic isolation in CAN networks is performed by optocouplers or transformers placed between CAN controller and CAN transceiver chip.
Gateway	Device with at least two network interfaces transforming all seven OSI (open system interconnection) protocol layers, e.g. CANopen-to-Ethernet gateway.
Heartbeat	CANopen uses heartbeat message to indicate that a node is still alive. This message is transmitted periodically.
Heartbeat consumer time	The heartbeat consumer time defines the time when a node is regarded as no longer alive due to a missing heartbeat message.
Heartbeat producer time	The heartbeat producer time defines the transmission frequency of a heartbeat message.
Identifier	In general, the term identifier refers to a CAN message identifier. The CAN message identifier identifies the content of a data frame. The identifier of a remote frame corresponds to the identifier of the requested data frame. The identifier includes implicitly the priority for the bus arbitration.

Index	16-bit address to access the CANopen dictionary; for array and records the address is extended by an 8-bit Subindex.
Inhibit timer	Object in CANopen for PDOs and Emergency messages that forbids for the specified time (inhibit time) a transmission of this communication object.
Initialization state	NMT Slave state in CANopen that is reached automatically after power-on and communication or application reset.
Interface profile	CANopen profile that describes just the interface and not the application behaviour of device, e.g. gateway and bridge devices.
ISO 11898-1	International standard defining the CAN data link layer including LLC, MAC and PLS sub-layers.
ISO 11898-2	International standard defining the CAN high-speed MAU.
Life guarding	Method in CANopen to detect that the NMT Master does not guard the NMT Slave any more. This not recommended for new systems designs.
Line topology	Networks, where all nodes are connected directly to one bus line. CAN networks use theoretically just line topologies without any stub cable. However in practice you find tree and star topologies as well.
Master	Communication or application entity that is allowed to control a specific function. In networks this is for example the initialization of a communication service.
Multiplexed PDO (MPDO)	The MPDO is made of 8 byte including one control byte, three multiplexer bytes (containing the 24-bit Index and Subindex), and four bytes of object data.
Network length	Bus length. The network cable length between both termination resistors. The bus length of CANopen networks is limited by the used transmission rate. At 1 Mbps the maximum length is 25 m. When using lower transmission rates, longer bus lines may be used: at 50 kbps a length of 1 km is possible.
Network management	Entity responsible for the network boot-up procedure and the optional configuration of nodes. It also may include node-supervising functions such as node guarding.
Network variables	Network variables are used in programmable CANopen devices to be mapped into PDOs after programming the device.
NMT	Network management in CANopen.
NMT Master	The NMT Master device performs the network management by means of transmitting the NMT message. With this message, it controls the state machines of all connected NMT Slave devices.
NMT Slave	The NMT Slaves receive the NMT message, which contains commands for the NMT state machine implemented in CANopen devices.

NMT state machine	The NMT state machines support different states and the highest prior CAN message transmitted controls the transition to the states by the NMT Master.
Node guarding	Mechanism used in CANopen and CAL to detect bus off or disconnected devices. The NMT Master sends a remote frame to the NMT Slave that is answered by the corresponding error control message.
Node ID	Unique identifier for a device required by different CAN-based higher-layer protocols in order to assign CAN identifiers to this device, e.g. in CANopen and DeviceNet. In the pre-defined connection set of CANopen some of the CAN message identifiers are derived from the assigned Node ID.
Object Dictionary	Heart of each CANopen device containing all communication and application objects.
Operational state	In the NMT **Operational** state all CANopen communication services are available.
PDO mapping	In PDOs, there may be mapped up to 64 objects. The PDO mapping is described in the PDO mapping parameters.
Pin assignment	Definition of the use of connector pins.
Pre-defined connection set	The pre-defined connection set is a default assignment of CAN message identifiers to CANopen communication objects. Some CANopen communication objects are distributed in broadcast (NMT message, Sync message, Time message) and others are transmitted between NMT Master device and dedicated NMT Slave devices (PDO, SDO, Emergency, and Error Control). This default assignment guarantees that the CAN message identifiers are uniquely assigned in the network, if the node-ID has been assigned uniquely.
Pre-operational state	In the NMT **Pre-operational** state no CANopen PDO communication is allowed.
Process Data Object (PDO)	Communication object defined by the PDO communication parameter and PDO mapping parameter objects. It is an unconfirmed communication service without protocol overhead.
Producer	In CAN networks a transmitter of messages is called a producer.
Protocol	Formal set of conventions and rules for the exchange of information between nodes, including the specification of frame administration, frame transfer and physical layer.
Receiver	A CAN node is called receiver or consumer, if it is not transmitter and the bus is not idle.
Redundant networks	In some safety-critical applications (e.g. maritime systems), redundant networks may be required that provide swapping capability in case of detected communication failures.
Remote frame	With a remote frame another node is requested to transmit

	the corresponding data frame identified by the very same identifier. The remote frame's DLC has the value of the corresponding data frame DLC. The data field of the remote frame has a length of 0 byte.
Remote transmission request (RTR)	Bit in the arbitration field indicating if the frame is a remote frame (recessive value) or a data frame (dominant value).
Repeater	Passive component that refreshes CAN bus signals. It is used to increase the maximum number of nodes, or to achieve longer networks (>1 km), or to implement tree or meshed topologies.
Reset application	This NMT command resets all objects in CANopen devices to the default values or the permanently stored configured values.
Reset communication	This NMT command resets only the communication objects in CANopen devices to the default values or the permanently stored configured values.
RPDO	The Receive Process Data Object (RPDO) is a communication object that is received by a CANopen device.
SDO block transfer	SDO block transfer is a CANopen communication service for increasing downloading. In SDO block transfer, the confirmation is sent after the reception of a number of SDO segments.
SDO Manager	The SDO Manager handles the dynamic establishment of SDO connections. It resides on the very same node as the NMT Master.
Segmented SDO	If objects longer than 4 bytes are transmitted by means of SDO services, a segmented transfer is used. The number of segments is theoretically not limited.
Server SDO	The Server SDO receives the SDO messages from the corresponding SDO Client and responds to each SDO message or to a block of SDO messages (SDO block transfer).
Service Data Object (SDO)	SDOs provide the access to entries in the CANopen Object Dictionary. An SDO is made up of at least two CAN messages with different identifiers. SDOs are always confirmed point-to-point communication services.
SI unit	International system of units for physical values as specified in ISO 1000:1983.
Stopped state	NMT state in which only NMT messages are performed and under some conditions error control messages are transmitted.
Sub-index	8-bit sub-address to access the sub-objects of arrays and records.
Suspend transmission	CAN controllers in error passive mode have to wait additional 8 bit-times before the next data or remote frame may be transmitted.
SYNC message	Dedicated CANopen message forcing the receiving nodes to

	sample the inputs mapped into synchronous TPDOs. Receiving this message causes the node to set the outputs to values received in the previous synchronous RPDO.
Termination resistor	In CAN high-speed networks with bus topology, both ends are terminated with resistors in order to suppress reflections.
TIME message	Standardized message in CANopen containing the time as a 6-byte value given as ms after midnight and days after 1st January 1984.
TPDO	The Transmit Process Data Object (TPDO) is a communication object that is transmitted by a CANopen device.
Transmission type	CANopen object defining the scheduling of a PDO.
Value definition	Detailed description of the value range in CANopen profiles.
Value range	Object attribute in CANopen defining the allowed values that this object supports.

1 - Safety summary

Safety

- Always adhere to the professional safety and accident prevention regulations applicable to your country during device installation and operation;
- installation and maintenance operations have to be carried out by qualified personnel only, with power supply disconnected and stationary mechanical parts;
- device must be used only for the purpose appropriate to its design: use for purposes other than those for which it has been designed could result in serious personal and/or the environment damage;
- high current, voltage and moving mechanical parts can cause serious or fatal injury;
- warning! Do not use in explosive or flammable areas;
- failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of the equipment;
- Lika Electronic assumes no liability for the customer's failure to comply with these requirements.

Electrical safety

- Turn off the power supply before connecting the device;
- connect it according to the explanation in the "4 - Electrical connections" section on page 28;
- in compliance with the 2014/30/EU norm on electromagnetic compatibility, the following precautions must be taken:
  - before handling and installing, discharge electrical charge from your body and tools which may come in touch with the device;
  - power supply must be stabilized without noise, install EMC filters on device power supply if needed;
  - always use shielded cables (twisted pair cables whenever possible);
  - avoid cables runs longer than necessary;
  - avoid running the signal cable near high voltage power cables;
  - mount the device as far as possible from any capacitive or inductive noise source, shield the device from noise source if needed;
  - to guarantee a correct working of the device, avoid using strong magnets on or near by the unit;
  - minimize noise by connecting the shield and/or the connector housing and/or the frame to ground. Make sure that ground is not affected by noise. The connection point to ground can be situated both on the device side and on user's side. The best solution to minimize the interference must be carried out by the user. Provide the ground connection as close as possible to the encoder. We suggest using the ground screw provided in the cap (use a TCEI M3 x 6 cylindrical head screw with 2 tooth lock washers).

Mechanical safety

- Install the device following strictly the information in the "3 - Mounting instructions" section on page 23;
- mechanical installation has to be carried out with stationary mechanical parts;
- do not disassemble the encoder;
- do not tool the encoder or its shaft;
- delicate electronic equipment: handle with care; do not subject the device and the shaft to knocks or shocks;
- respect the environmental characteristics declared by manufacturer;
- unit with solid shaft: in order to guarantee maximum reliability over time of the mechanical parts, we recommend a flexible coupling to be installed to connect the encoder and the user's shaft; make sure the misalignment tolerances of the flexible coupling are respected;
- unit with hollow shaft: the encoder can be mounted directly on a shaft whose diameter has to respect the technical characteristics specified in the purchase order and clamped by means of the collar and, when requested, the anti-rotation pin.

2 - Identification

Device can be identified through the **order code** and the **serial number** printed on the label applied to its body. Information is listed in the delivery document too. Please always quote the order code and the serial number when reaching Lika Electronic for purchasing spare parts or needing assistance. For any information on the technical characteristics of the product refer to the technical catalogue.

Warning: encoders having order code ending with "/Sxxx" may have mechanical and electrical characteristics different from standard and be supplied with additional documentation for special connections (Technical info).

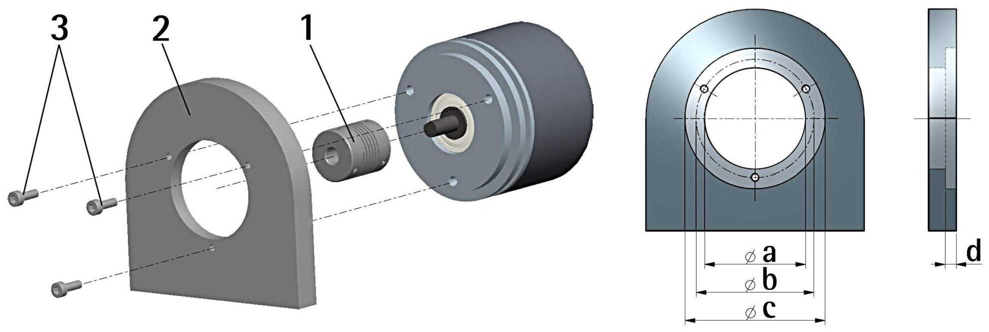
3 - Mounting instructions

WARNING
Installation and maintenance operations have to be carried out by qualified personnel only, with power supply disconnected and mechanical parts absolutely in stop.

3.1 Solid shaft encoders

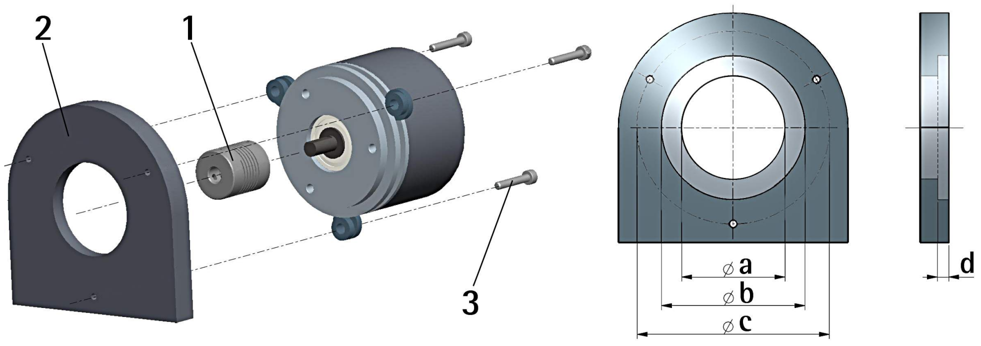
- Mount the flexible coupling **1** on the encoder shaft;
- fix the encoder to the flange **2** (or to the mounting bell) by means of screws **3**;
- secure the flange **2** to the support (or the mounting bell to the motor);
- mount the flexible coupling **1** on the motor shaft;
- make sure the misalignment tolerances of the flexible coupling **1** are respected.

3.1.1. Customary installation



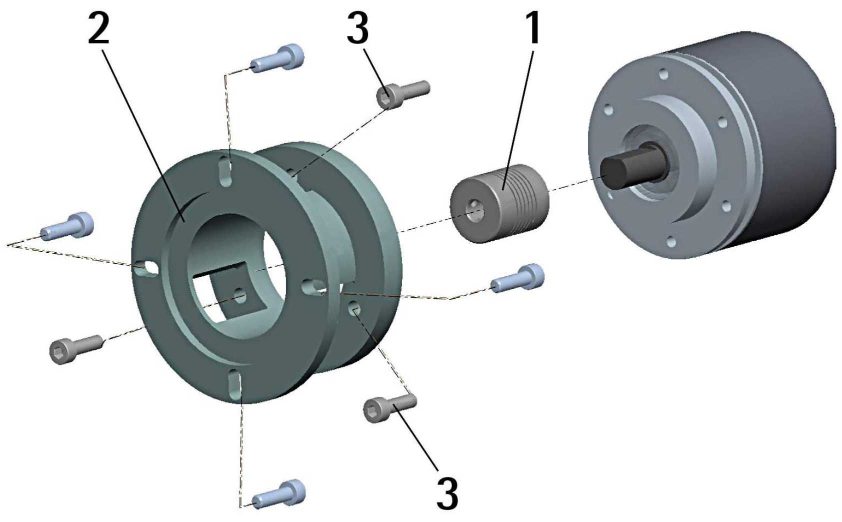
	a [mm]	b [mm]	c [mm]	d [mm]
AS58, AM58	-	42	50 F7	4
AS58S, AM58S	36 H7	48	-	-

3.1.2 Installation using fixing clamps (code LKM-386)



	a [mm]	b [mm]	c [mm]	d [mm]
AS58, AM58	-	50 F7	67	4
AS58S, AM58S	36 H7	-	67	-

3.1.3 Installation using a mounting bell (code PF4256)



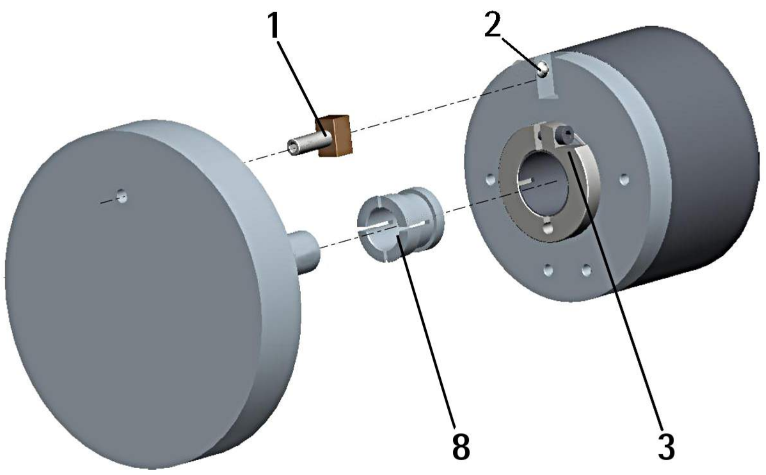
NOTE

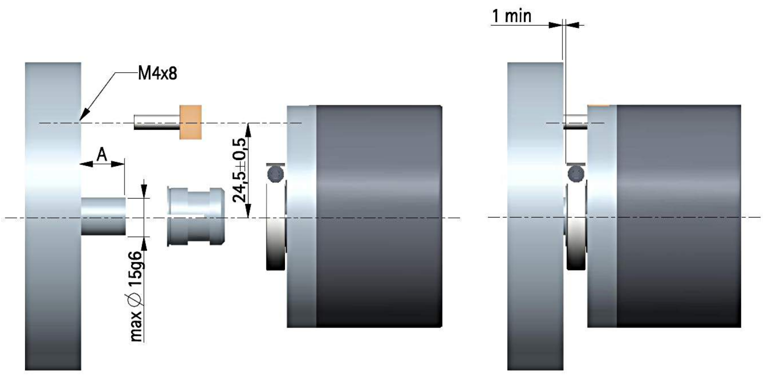
In order to guarantee reliability over time of the encoder mechanical parts, we recommend a flexible coupling to be installed between the encoder and the motor shaft. Make sure the misalignment tolerances of the flexible coupling are respected.

3.2 Hollow shaft encoders

3.2.1 ASC58, AMC58

- Fasten the anti-rotation pin **1** to the rear of the motor (secure it using a locknut);
- mount the encoder on the motor shaft using the reducing sleeve **8** (if supplied). Avoid forcing the encoder shaft;
- insert the anti-rotation pin **1** into the slot on the flange of the encoder; this secures it in place by grub screw **2**, preset at Lika;
- fix the collar **3** to the encoder shaft (apply some threadlocker to the screw **3**).

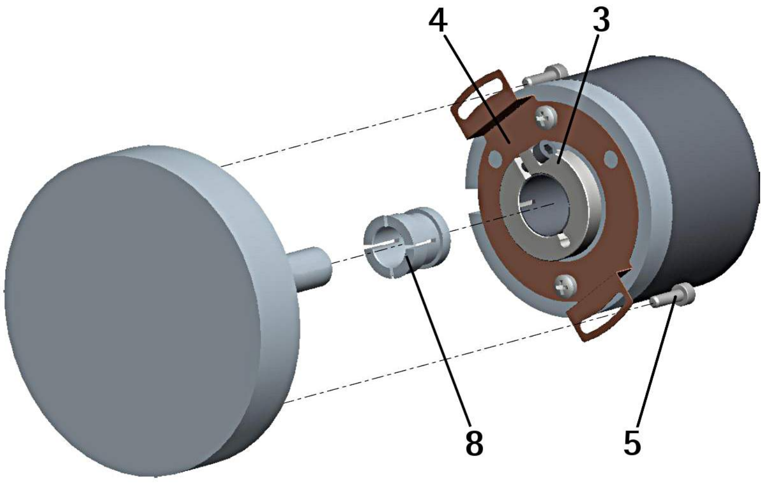


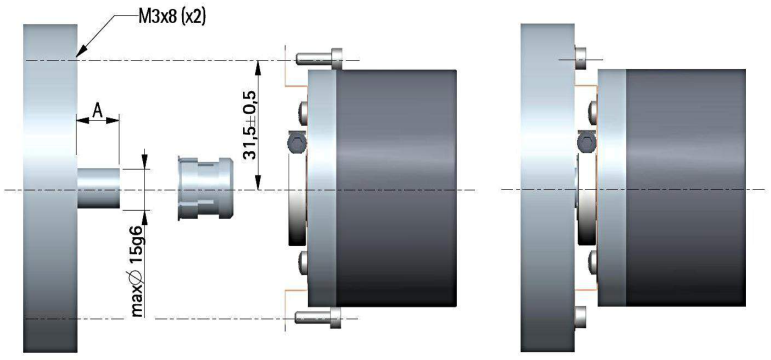


A = min. 8 mm, max. 18 mm

3.2.2 ASC59, AMC59

- Mount the encoder on the motor shaft using the reducing sleeve **8** (if supplied). Avoid forcing the encoder shaft;
- fasten the fixing plate **4** to the rear of the motor using two M3 x 8 cylindrical head screws **5**;
- fix the collar **3** to the encoder shaft (apply some threadlocker to the screw **3**).

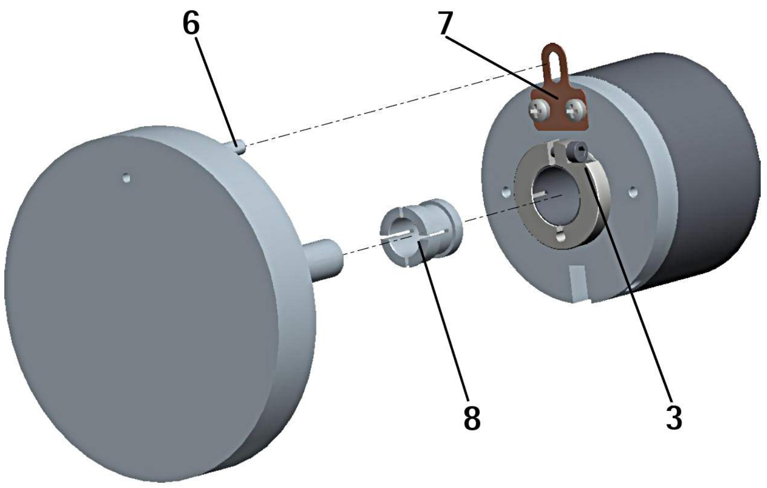


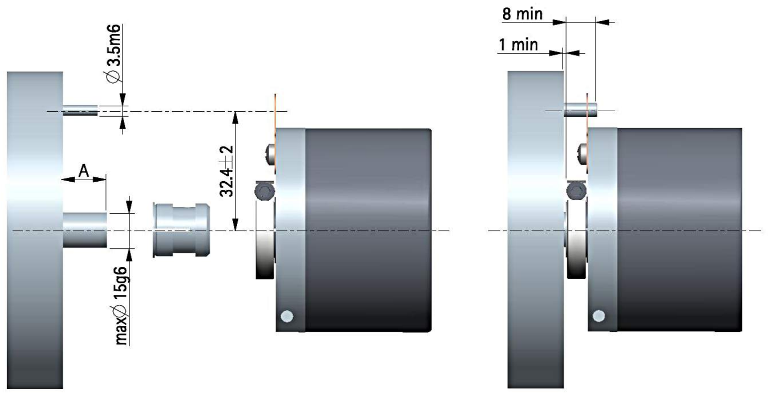


A = min. 8 mm, max. 18 mm

3.2.3 ASC60, AMC60

- Fix the tempered pin **6** to the rear of the motor;
- mount the encoder on the motor shaft using the reducing sleeve **8** (if supplied). Avoid forcing the encoder shaft;
- make sure the anti-rotation pin **6** is inserted properly into the fixing plate **7**;
- fix the collar **3** to the encoder shaft (apply some threadlocker to the screw **3**).





A = min. 8 mm, max. 18 mm

NOTE

You are strongly advised not to carry out any mechanical operations (drilling, milling, etc.) on the encoder shaft. This could cause serious damages to the internal parts and an immediate warranty loss. Please contact our technical personnel for the complete availability of "custom made" shafts.

4 - Electrical connections

WARNING
Electrical connections must be carried out by qualified personnel only, with power supply disconnected and mechanical parts compulsorily in stop.

4.1 Connection cap

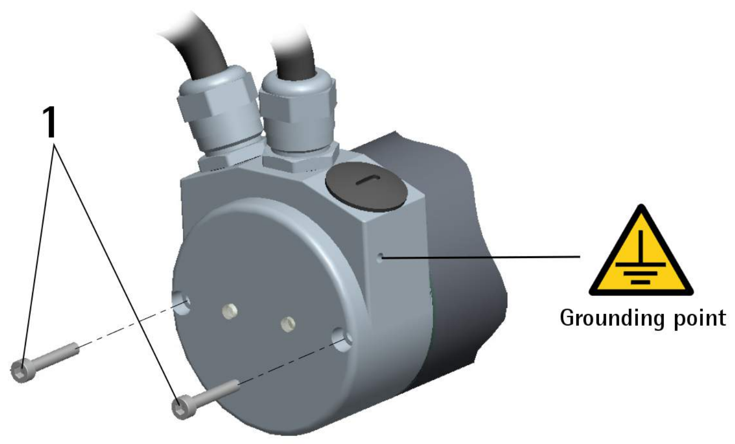
WARNING
Do not remove or mount the connection cap with power supply switched ON. Damage may be caused to internal components.

The terminal connectors for connecting the power supply and the BUS IN and BUS OUT cables (CC-CB connection cap) as well as the DIP switches meant to set the baud rate and the node ID and activate the termination resistance (both CC-CB connection cap and CC-CB-C connection cap) are located inside the encoder connection cap. Thus you must remove the connection cap to access any of them.

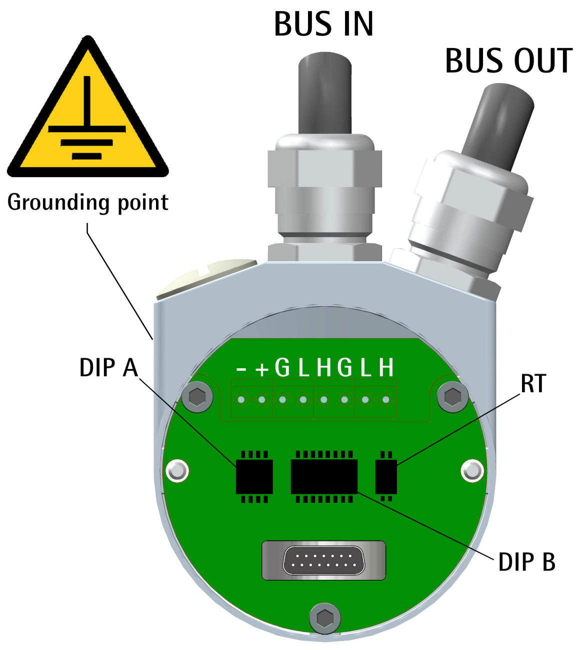
NOTE
Be careful not to damage the internal components when you perform this operation.

To remove the connection cap loosen the two screws **1**. Please be careful with the internal connector.
Always replace the connection cap at the end of the operation. Take care in re-connecting the internal connector. Tighten the screws **1** using a tightening torque of approx. 2.5 Nm.

WARNING
You are required to check that the encoder body and the connection cap are at the same potential before replacing the connection cap!



4.2 Connection cap with PGs: CC-CB



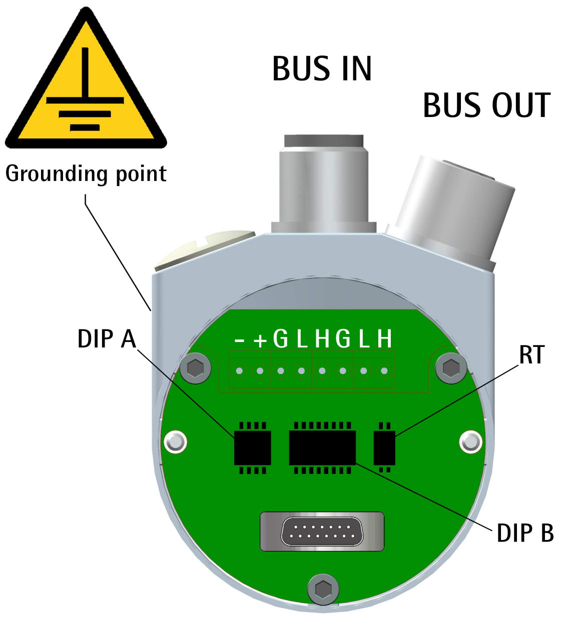
The connection cap of PG versions (CC-CB connection cap) is fitted with two PG9 cable glands for BUS IN, BUS OUT, and power supply connections. The bus cables can be connected directly to the terminal connectors located in front of each cable gland. We recommend CANbus certified cables to be used. Core diameter should not exceed Ø 1.5 mm (0.06 inches).

Terminal connector	Description
-	0Vdc power supply voltage
+	+10Vdc +30Vdc power supply voltage
G	CAN GND [1]
L	CAN Low
H	CAN High
PG	CAN Shield [2]

[1] CAN GND is the 0V reference of CAN signals, it is not connected to 0Vdc power supply voltage.

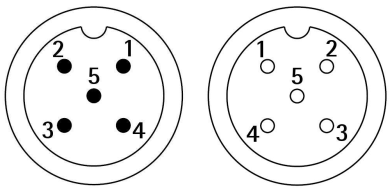
[2] Connect the cable shield to the cable gland.

4.3 Connection cap with M12 connectors: CC-CB-C



The connection cap of connector versions (CC-CB-C connection cap) is fitted with two M12 connectors with pin-out in compliance with CANopen® standard. Therefore you can use standard CAN cables commercially available.

M12 connector
A coding
(frontal side)



male
(BUS IN)

female
(BUS OUT)

M12	Description
Case	CAN Shield
1 [1]	CAN Shield
2	+10Vdc +30Vdc power supply voltage
3	0Vdc power supply voltage
4	CAN High
5	CAN Low

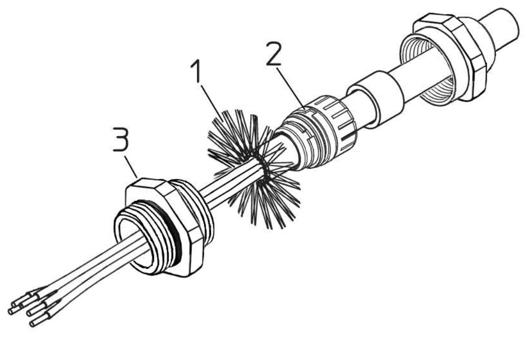
[1] CAN Shield is also connected to pin 1 to allow the connection of the shield even if the plug connector has a plastic case.

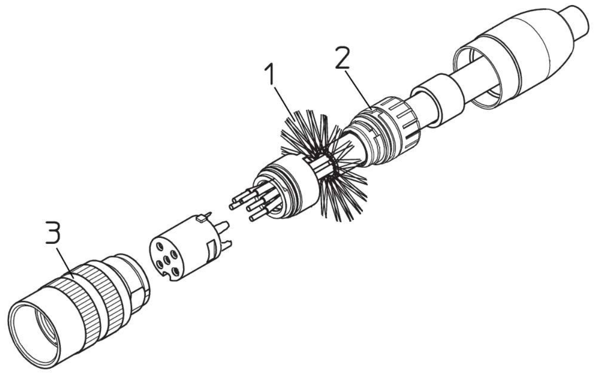
4.4 Ground connection

Minimize noise by connecting the shield and/or the connector housing and/or the frame to ground. Make sure that ground is not affected by noise. The connection point to ground can be situated both on the device side and on user's side. The best solution to minimize the interference must be carried out by the user. You are advised to provide the ground connection as close as possible to the encoder. We suggest using the ground point provided in the cap (see the Figures, use 1 TCEI M3 x 6 cylindrical head screw with 2 tooth lock washers).

4.5 Connection of the shield

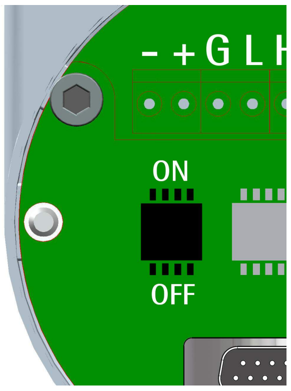
Disentangle and shorten the shielding **1** and then bend it over the part **2**; finally place the ring nut **3** of the connector. Be sure that the shielding **1** is in tight contact with the ring nut **3**.





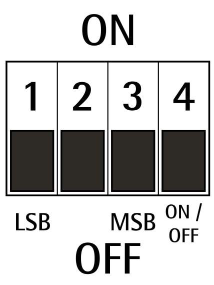
4.6 Setting the baud rate: DIP A

WARNING
Power supply must be turned off before performing this operation!



The transmission rate (baud rate) can be set either via hardware (using DIP A) or via software (by setting the **3000-00 Baud rate** object).
If the bit 4 of **DIP A** = "OFF" the bit rate is set through the **3000-00 Baud rate** object in the "Object Dictionary" and can be modified using SDO messages.
If the bit 4 of **DIP A** = "ON" the bit rate is set by DIP A.

DIP A:



Switch off the device and set the binary value of the transmission rate considering that: ON=1, OFF=0.

bit	**1** LSB	**2**	**3** MSB	**4**
	2^0	2^1	2^2	ON/OFF

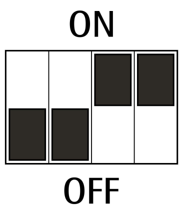
Available baud rate values:

Decimal value	**Binary value**	**Baud rate**
0	000	20 Kbit/s
1	001	50 Kbit/s
2	010	100 Kbit/s
3	011	125 Kbit/s
4	100	250 Kbit/s
5	**101**	**500 Kbit/s (default)**
6	110	800 Kbit/s
7	111	1000 Kbit/s

EXAMPLE

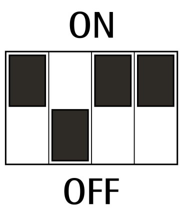
Set the baud rate to 250Kbit/s:
4_{10} = 100_2 (binary value, see table above)

bit	1	2	3	4
	2^0	2^1	2^2	2^3
	OFF	OFF	ON	ON



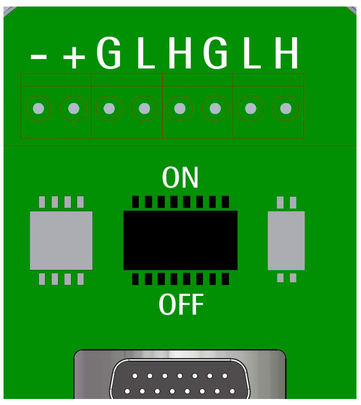
Set the baud rate to 500Kbit/s:
5_{10} = 101_2 (binary value, see table above)

bit	1	2	3	4
	2^0	2^1	2^2	2^3
	ON	OFF	ON	ON



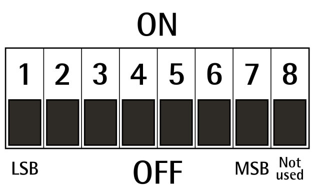
4.7 Setting the node address: DIP B

WARNING
Power supply must be turned off before performing this operation!



The node number can be set either via hardware (using DIP B) or via software (by setting the 3001-00 Node-ID object).
If all bits of **DIP B** are "OFF" (address 0) the node number is set through the 3001-00 Node-ID object of the "Object Dictionary" and can be modified using SDO messages.
If one bit at least of **DIP B** is set to "ON" the node number is set through DIP B.
Allowed node addresses range between 1 and 127. **The default value is 1.**

DIP B:

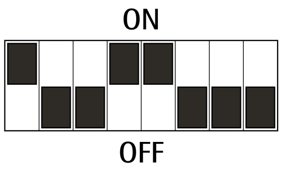


Switch off the device and set the binary value of the node number considering that: ON=1, OFF=0

bit	1 LSB	2	3	4	5	6	7 MSB	8 not used
	2^0	2^1	2^2	2^3	2^4	2^5	2^6	

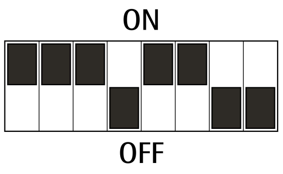
EXAMPLE
Set the node number = 25:
25_{10} = 0001 1001_2 (binary value)

bit	1	2	3	4	5	6	7	8
	2^0	2^1	2^2	2^3	2^4	2^5	2^6	
	ON	OFF	OFF	ON	ON	OFF	OFF	OFF



Set the node number = 55:
55_{10} = 0011 0111_2 (binary value)

bit	1	2	3	4	5	6	7	8
	2^0	2^1	2^2	2^3	2^4	2^5	2^6	
	ON	ON	ON	OFF	ON	ON	OFF	OFF



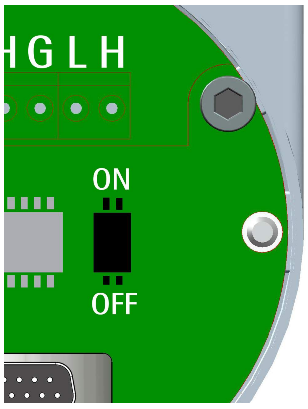
WARNING
If the baud rate and the node number are set via software, the Master device has to detect the baud rate of the Slave (scanning of baud rate) when the encoder is being installed. Once the communication has been activated, the new baud rate and node number values can be set (**3000-00 Baud rate** and **3001-00 Node-ID** objects). After having set new values, transmit a **Reset node** command and then save the parameters (**1010-01 Store parameters** object). To avoid conflict between Slaves, this operation has to be carried out when one only device is connected to the network.

4.8 RT Bus termination

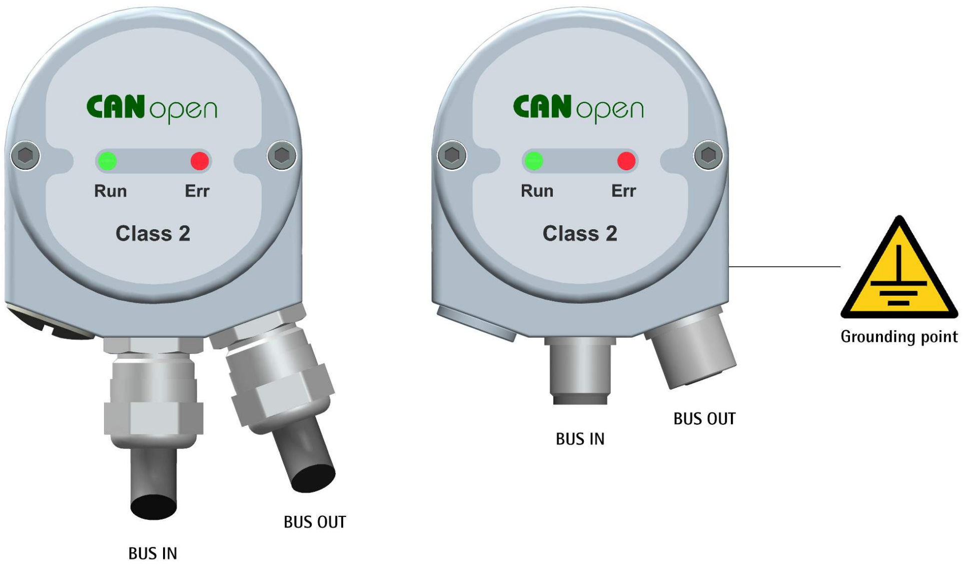
WARNING
Power supply must be turned off before performing this operation!

A bus termination resistance is provided inside the connection cap and must be activated as line termination if the encoder is at the ends of the transmission line (i.e. it is either the first or the last device in the transmission line).
Use RT Switch to activate or deactivate the bus termination.

RT	Description
1 = 2 = ON	Activated: if the encoder is either the first or the last device in the transmission line
1 = 2 = OFF	Deactivated: if the encoder is neither the first nor the last device in the transmission line



4.9 Diagnostic LEDs



Two diagnostic LEDs located in the connection cap are designed to show visually the operating or fault status of the CANopen® interface.

RUN LED (GREEN)	Description
ON	The encoder is in **Operational** state, see on page 42
Single flash	The encoder is in **Stopped** state, see on page 42
Blinking	The encoder is in **Pre-Operational** state, see on page 42

ERROR LED (RED)	**Description**
ON	Bus off, the CAN controller is switched off
Double flash	Node guarding error, see on page 78 ff
Single flash	Max. number of warning errors reached
Blinking	Generic error or **Flash memory error**, see on page 77 ff
OFF	No error

During initialization, the device carries out a hardware test to check LEDs operation. Both LEDs light up.

5 - Quick reference

5.1 Hardware resolution

WARNING
Make sure the physical resolution of the encoder matches the set resolution. Different resolutions may be set if the encoder and the connection cap have been ordered / supplied separately.

EXAMPLES

ASx58x 12/CB-xx	6501-00 Hardware counts per revolution = 4096, 6502-00 Hardware number of turns = 1.
ASx58x 13/CB-xx	6501-00 Hardware counts per revolution = 8192, 6502-00 Hardware number of turns = 1.
AMx58x 12/4096CB-xx	6501-00 Hardware counts per revolution = 4096, 6502-00 Hardware number of turns = 4096.
AMx58x 13/4096CB-xx	6501-00 Hardware counts per revolution = 8192, 6502-00 Hardware number of turns = 4096.

6501-00 Hardware counts per revolution object is described on page 72.
6502-00 Hardware number of turns object is described on page 73.

If the hardware resolution does not match the order code (see the encoder label), then it is compulsory **to set the hardware resolution** (for any information on reading the hardware resolution refer to the "5.2 Using the default settings" section on page 39).
Please note that the 6001-00 Measuring units per revolution and 6002-00 Total resolution objects refer to the scaling function, anyway the unit can run properly only if the hardware resolution is set correctly.

5.1.1 Procedure to set the hardware resolution

ID = Node Identifier.

Step 1 - Access to the configuration (object 3002h)

NOTE
To avoid unintentional access, this object is not listed in the EDS file.

Master → Encoder

COB-ID	Cmd	Index		Sub	Process data			
600+ID	23	02	30	00	41	4B	49	4C

Encoder → Master

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	02	30	00	00	00	00	00

Step 2 – Set the 6501-00 Hardware counts per revolution object

See the resolution table below in the page for B0, B1, B2, and B3 values.

Master → Encoder

COB-ID	Cmd	Index		Sub	Process data			
600+ID	23	01	65	00	B0	B1	B2	B3

Encoder → Master

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	01	65	00	00	00	00	00

Step 3 – Set the 6502-00 Hardware number of turns object

See the resolution table below in the page for B4, B5, B6, and B7 values.

Master → Encoder

COB-ID	Cmd	Index		Sub	Process data			
600+ID	23	02	65	00	B4	B5	B6	B7

Encoder → Master

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	02	65	00	00	00	00	00

Step 4 - Send a Reset node command

Master → Encoder

COB-ID	Cmd	Slave ID
000	81	ID

Step 5 - Store parameters (1010-01 Store parameters object)

Master → Encoder

COB-ID	Cmd	Index		Sub	Process data			
600+ID	23	10	10	01	73	61	76	65

Encoder → Master

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	10	10	01	00	00	00	00

Table of resolutions

Encoder type	steps/rev.				n° rev.			
	B0	B1	B2	B3	B4	B5	B6	B7
ASx58x 12/CB-xx	00	10	00	00	01	00	00	00
ASx58x 13/CB-xx	00	20	00	00	01	00	00	00
AMx58x 12/4096CB-xx	00	10	00	00	00	10	00	00
AMx58x 13/4096CB-xx	00	20	00	00	00	10	00	00

5.2 Using the default settings

Using the default settings provided by the manufacturer, you can switch on the device and read immediately its position.

Follow the instructions below to:

- read the physical resolution of the device: physical singleturn resolution **6501-00 Hardware counts per revolution**; number of physical revolutions **6502-00 Hardware number of turns**;
- set the desired cycle time: **6200-00 Cyclic timer** $\neq 0$;
- set the **Operational** mode;
- read the current position (in a cyclic mode and/or in a synchronous mode).

The default Baud rate and Node-ID are:

Baud rate = 500 Kbit/s

Node-ID = 1

Read the value of the physical resolution per revolution 6501-00 Hardware counts per revolution (physical singleturn resolution)

Master → Encoder

COB-ID	Cmd	Index		Sub	Process data			
601	40	01	65	00	-	-	-	-

Encoder → Master

COB-ID	Cmd	Index		Sub	Process data			
581	43	01	65	01	A0	A1	A2	A3

steps/rev. = ((A3<<24) | (A2<<16) | (A1<<8) | A0)

Read the number of physical revolutions 6502-00 Hardware number of turns

Master → Encoder

COB-ID	Cmd	Index		Sub	Process data			
601	40	02	65	00	-	-	-	-

Encoder → Master

COB-ID	Cmd	Index		Sub	Process data			
581	43	02	65	01	B0	B1	B2	B3

N. rev. =((B3<<24) | (B2<<16) | (B1<<8) | B0)

Set the cyclic time 6200-00 Cyclic timer (100 ms = 64h)

Master → Encoder

COB-ID	Cmd	Index		Sub	Process data			
600+ID	2B	00	62	00	64	00	-	-

Encoder → Master

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	00	62	00	00	00	-	-

Set the Operational mode

Master → Encoder

COB-ID	Cmd	Node
000	01	01

Read the position every 100 ms

Encoder → Master

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3
181	Low	...	...	High

NOTE

For further examples please refer to the "7 – Setting-up" section on page 81.

6 - CANopen® interface

Lika encoders are always Slave devices and comply with the "Device profile for encoders", Class 2.
For any omitted information concerning the CANopen protocol, refer to the "CiA Draft Standard Proposal 301. Application Layer and Communication Profile" and to the "CiA Draft Standard Proposal 406. Device profile for encoders" documents available at the address **www.can-cia.org**.

6.1 EDS file

CANopen® encoders are supplied with their own EDS file **Lika_AxxCB_DS406_Vx.eds**. EDS files are specific for each encoder model. When you need to download the file please refer to the address **www.lika.biz > ROTARY ENCODERS > ABSOLUTE ENCODERS**).
EDS file must be installed in the CANopen® Master device.
Vx is intended to indicate the file version.

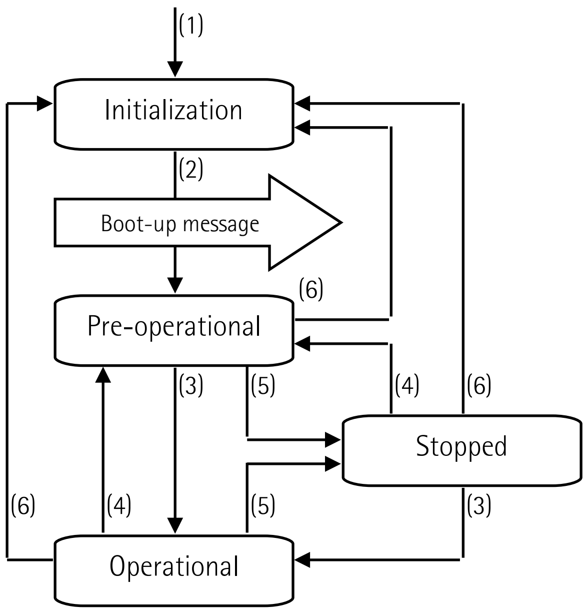
The available EDS files are:

Lika_AMxCB_DS406_Vx.eds	for multiturn encoders
Lika_ASxCB_DS406_Vx.eds	for singleturn encoders

WARNING
Please always ascertain that the EDS file conforms with the encoder model.

6.2 State machine

CANopen® devices are designed to operate using different states. The transition from one state to another is made by sending specific NMT messages (see the Figure below).



(1)	Power on
(2)	Initialization carried out, boot-up message is sent automatically
(3)	NMT message: **Start remote node**
(4)	NMT message: **Enter pre-operational**
(5)	NMT message: **Stop remote node**
(6)	NMT message: **Reset node** or **Reset communication**

6.2.1 Initialization state

This is the first state the CANopen® device enters after the power is turned on or after a hardware reset (**Reset node** command). As soon as the basic CANopen® device initialization is carried out, the device reads and loads the parameters saved on EPROM, sends a boot-up message and then switches automatically to **Pre-operational** state.

6.2.2 Pre-operational state

In this state the communication between the Master and the Slave is possible using SDO messages. They allow working parameters to be set. The Slave cannot send PDO messages. The state is signalled through the green LED (see on page 35).

To switch the Slave device to the Operational state the Master must send a **Start remote node** command using an NMT message (see on page 81).

6.2.3 Operational state

In this state the Slave device is active and all communication objects are available. The Slave device can use the parameters available in the "Object dictionary" (see on page 47) and is allowed to send process data using PDO messages. The "Object dictionary" can be accessed by using SDO messages. To switch the Slave device back to the **Pre-operational** state the Master must send an **Enter pre-operational** command using an NMT message (see on page 81).

6.2.4 Stopped state

In this state the Slave device is forced to cut off the communication with the Master (except node guarding, if active).
The communication using PDO and SDO messages is not allowed.
To switch the Slave device to either the **Pre-operational** or the **Operational** state the Master must send the specific commands **Enter pre-operational** or **Start remote node** using an NMT message (see on page 81).

NOTE
Please refer to the "7 – Setting-up" section on page 81 for an example of how the states are to be set.

6.3 Communication objects

Four different kinds of communication messages are used in a CANopen® network:

- Network management NMT protocol: NMT protocols are used by the Master to manage the nodes and the network, issue state machine change commands (i.e. to start and stop the devices), detect the remote device boot-ups and error conditions.
- Process Data Objects PDO protocol: used to process real time data (transmission of process data in real time).
- Service Data Objects SDO protocol: used to set and read values from the "Object dictionary" of a remote device.
- Special Function Objects:
  - SYNC: synchronization message used by the Master to enable the Slave devices to transmit process data (encoder position and velocity).
  - Emergency: error messages are triggered by each error event.
  - Node Guarding: used to request the state of the Slave: the NMT Master checks the NMT Slaves at regular intervals.

Relation between the device states and the communication objects:

	Initial.	Pre-oper.	Operat.	Stopped
NMT		X	X	X
PDO			X	
SDO		X	X	
SYNC			X	
EMCY		X	X	
Boot-up	X			
Node guarding		X	X	X

6.3.1 Pre-defined connection set

Master → Slave broadcast		
Type of COB (Object)	Function code (binary)	COB-ID (hex)
NMT	0000	000
SYNC	0001	080

peer-to-peer transmission		
EMERGENCY	0001	081 - 0FF
PDO 1 (tx)	0011	181 - 1FF
PDO 2 (tx)	0101	281 - 2FF
PDO 3 (tx)	0111	381 - 3FF
PDO4 (tx)	1001	481 - 4FF
SDO (tx)	1011	581 - 5FF
SDO (rx)	1100	601 - 67F
Node guard	1110	701 - 77F
Boot-up	1110	701 - 77F

The type of COB (transmitted / tx or received / rx) is viewed from the Slave device.

6.4 NMT objects

Structure of NMT objects:

COB-ID (11 bit)		2 CAN Data Bytes	
Func.code	Node ID	Command	Slave ID
0000	0	NMT Func.	Slave ID

If the Slave ID = 00h, the NMT message is issued to all nodes in the network.

NMT Function:

Command	NMT Function	State node
01 hex	**Start remote node**	Operational
02 hex	**Stop remote node**	Stopped
80 hex	**Enter pre-operational**	Pre-operational
81 hex	**Reset node**	Pre-operational
82 hex	**Reset communication**	Pre-operational

6.5 Boot-up objects

Structure of the Boot-up message:

COB-ID(hex)	1 CAN Data Byte
700+Node ID	00

6.6 PDO objects

PDO (tx) messages are always made up of 4 CAN Data Bytes and are used by the encoder to transmit the position value and/or the velocity value.

Structure of the PDO objects:

IDENTIFIER		4 CAN Data Bytes			
COB-ID(hex)		Byte 0	Byte 1	Byte 2	Byte 3
F.C.	Node-ID	Low	...	...	High
		position value (with PDO1, PDO2, PDO3)			
		velocity value (with PDO4)			

PDO1 Cyclic mode: cyclic transmission

The encoder uses the PDO1 message to transmit the position value **cyclically**, i.e. periodically and independently from the Master.
The interval between two issues is set in the **6200-00 Cyclic timer** object.
To activate (or deactivate) the cyclic mode it is necessary to set to 0 (or 1) the most significant bit of COB-ID used by PDO1 (**1800 PDO1 parameters**, sub 1 object).

PDO2 and PDO3 SYNC mode: synchronous transmission

The transmission of the position value is managed by the Master **by sending a SYNC message**.
SYNC message is a high-priority COB transmitted by the Master to request the position value of the encoder through a PDO.
If several nodes (encoders) are connected to the network, the Master receives the position values from the Slaves according to the order of the Node addresses.

The encoder can be programmed to send a reply after a set number of SYNC messages by setting a counter.
The PDO message will be transmitted after having received the set number of SYNC messages.
For PDO2 the value of the counter must be set in the **1801 PDO2 parameters**, sub 2 object.
For PDO3 refer to the **1802 PDO3 parameters**, sub 2 object.

The SYNC transmission mode can be enabled (or disabled) by setting to 0 (or 1) the most significant bit (MSB) of COB-IB used by PDO (**1801 PDO2 parameters** / **1802 PDO3 parameters**, sub1 objects).

PDO4 Cyclic mode: cyclic transmission of the velocity

The encoder uses the PDO4 message to transmit the velocity value **cyclically**, i.e. periodically and independently from the Master.
The interval between two issues is set in the **6200-00 Cyclic timer** object.
To activate (or deactivate) the cyclic mode it is necessary to set to 0 (or 1) the most significant bit of COB-ID used by PDO4 (**1803 PDO4 parameters**, sub 1 object).

NOTE
More then one transmission mode can be active at the same time.

6.7 SDO objects

SDO messages are used to set and read values from the Object dictionary of the encoder. These parameters are described in the "Object dictionary" section (see on page 47).
4 bytes at the most are used for CAN data, other 4 bytes are used for Command, Index, and Sub-index fields. SDO messages are always followed by confirmation. It follows that when the Master sends an SDO message to the Slave, the Slave always sends a reply (and a warning, should an error occur).

Structure of SDO message:

IDENTIFIER		**from 4 to 8 CAN data bytes**							
COB-ID(hex)		0	1	2	3	4	5	6	7
F.C.	Node-ID	Com	Index		Sub	Data			
		1byte	LSB	MSB	1byte	LSB	...	...	MSB

Com	command
Index	parameter index
Sub	parameter sub-index
Data	parameter value

6.7.1 Command

The command byte contains the type of telegram transmitted to the CAN network.
Three types of COB telegram are available:

- Set: it is used to send the configuration parameters to a device;
- Req: it is used by the Master to read data from a Slave device;
- Warnings: they are used by the Slave to send error messages to the Master (e.g. following a wrong SDO message: **Object does not exist in the object dictionary**, ...).

Command	COB	COB type	Data length
22h	Set	M → S request	not spec.
23h	Set	M → S request	4 bytes
2Bh	Set	M → S request	2 bytes
2Fh	Set	M → S request	1 byte
60h	Set	S → M confirmation	0 byte
40h	Req	M → S request	0 byte
42h	Req	S → M reply	not spec.
43h	Req	S → M reply	4 bytes
4Bh	Req	S → M reply	2 bytes
4Fh	Req	S → M reply	1 byte
41h	Req	S → M reply segmented SDO	
80h	Warning	S → M reply	4 bytes

6.8 Object dictionary

The most important part of a device profile is the Object Dictionary. The Object Dictionary is essentially a grouping of objects accessible via the network in an ordered, pre-defined fashion.
The user-related objects are grouped in three main areas: the Communication Profile Area, the Manufacturer Specific Profile Area and the Standardised Device Profile Area. The objects are all described in the EDS file.

The **Communication Profile Area** at indexes from 1000h to 1FFFh contains the communication specific parameters for the CANopen network. These entries are common to all devices. NMT services, PDO objects and SDO objects are described in this section. The Communication Profile Area objects comply with the "CiA Draft Standard Proposal 301 CANopen Application layer and communication profile". Refer to the "6.8.1 Communication Profile Area objects (DS 301)" section on page 49.

The **Manufacturer Specific Profile Area** at indexes from 2000h to 5FFFh is free to add manufacturer-specific functionality. Refer to the "6.8.2 Manufacturer Specific Profile Area objects" section on page 60.

The **Standardised Device Profile Area** at indexes from 6000h to 9FFFh contains all data objects common to a class of devices that can be read or written via the network. The device profiles may use entries from 6000h to 9FFFh to describe the device parameters and the device functionality. The Standardised Device Profile Area objects comply with the "CiA Draft Standard 406 CANopen Device profile for encoders". Refer to the "6.8.3 Device Profile Area objects (DS 406)" section on page 63.

In the following pages the implemented objects are listed and described as follows:

Index-subindex Object name
[data types, attribute]

- Index and subindex are expressed in hexadecimal notation.
- Attribute:
  ro = read only access
  rw = read and write access

Unsigned/Signed8 data type:

Process data bytes							
byte 4							
7	6	5	4	3	2	1	0
MSbit			...			LSbit	

Unsigned/Signed16 data type:

Process data bytes	
byte 4	byte 5
LSByte	MSByte

Unsigned/Signed32 data type:

Process data bytes			
byte 4	byte 5	byte 6	byte 7
LSByte	...	...	MSByte

Unsigned/Signed64 data type:

Process data bytes							
byte 4	byte 5	byte 6	byte 7	byte 8	byte 9	byte 10	byte 11
LSByte	...	...	...	...	...	...	MSByte

6.8.1 Communication Profile Area objects (DS 301)

1000-00 Device type

[Unsigned32, ro]
It contains information about the device type. The object describes the type of device and its functionality.

Default =	0001 0196h	=	singleturn encoder, DS 406
	0002 0196h	=	multiturn encoder, DS 406

1001-00 Error register

[Unsigned8, ro]
Should an error occur, the bit 0 in this object will be set to "1".
Default = 00h

1003 Pre-defined error field

This object is intended to show the last four errors which caused an emergency message to be triggered. For any information refer to the "6.10 Emergency objects" section on page 77.

- **00 Number of occurred errors** [Unsigned8, rw]
  (write 00h to delete the error history)
- **01 Last error occurred** [Unsigned32, ro]
- **02-04 Previous errors occurred** [Unsigned32, ro]

1005-00 COB-ID SYNC message

[Unsigned32, rw]
This object indicates the configured COB-ID of the synchronisation object (SYNC). Furthermore, it defines whether the CANopen device generates the SYNC.
Default = 0000 0080h (CANopen device generates SYNC message)

1008-00 Manufacturer device name

[String, ro]
It shows the name of the device (manufacturer).
Default = "LIKA SRL"

1009-00 Manufacturer hardware version

[String, ro]
It shows the hardware version of the device.
Default = device dependent

100A-00 Manufacturer software version

[String, ro]
It shows the software version of the device.
Default = device dependent

100C-00 Guard time

[Unsigned16, rw]
It allows to set the Guard time expressed in milliseconds (msec).
The **100C-00 Guard time** object is used in the "Node guarding protocol" controlled by the Master. For more details see the "6.11 Node guarding protocol" section on page 78.
Default = 0000h

100D-00 Life time factor

[Unsigned8, rw]
The **100D-00 Life time factor** object is used in the "Node guarding protocol" controlled by the Master. For more details see the "6.11 Node guarding protocol" section on page 78.
Default = 00h

1010-01 Store parameters

[Unsigned32, rw]
Use this object to save all parameters on non-volatile memory.
Write "**save**" (ASCII code in hexadecimal format) in the data bytes:

Master → Encoder

COB-ID	Cmd	Index		Sub	Data bytes			
600+ID	23	10	10	01	73	61	76	65
					s	a	v	e

Encoder → Master (confirmation)

COB-ID	Cmd	Index		Sub	Data bytes			
580+ID	60	10	10	01	00	00	00	00

1011-01 Restore default parameters

[Unsig32, rw]
This object allows the operator to restore all parameters to default values (default values are set at the factory by Lika Electronic engineers to allow the operator to run the device for standard operation in a safe mode).
Write "**load**" (ASCII code in hexadecimal format) in the data bytes and then issue a **Reset node** command:

Master → Encoder

COB-ID	Cmd	Index		Sub	Data bytes			
600+ID	23	11	10	01	6C	6F	61	64
					l	o	a	d

Encoder → Master (confirmation)

COB-ID	Cmd	Index		Sub	Data bytes			
580+ID	60	11	10	01	00	00	00	00

Master → Encoder (**Reset node**)

COB-ID	Cmd	Slave ID
000	81	ID

Encoder → Master (Boot-up)

COB-ID	Cmd
700+ID	00

NOTE
Save the default values after upload using the store parameters function (see the **1010-01 Store parameters** object).

1014-00 COB-ID EMCY

[Unsigned32, rw]
This object defines the COB-ID used to send emergency messages (EMCY).
If the node address is set using the internal DIP switches (i.e. at least one DIP switch for setting the node has HIGH logic level = 1), when the power is turned on, this object is always forced to the default value. Otherwise, if the node address is set via software (i.e. all DIP switches for setting the node have LOW logic level = 0) it retains the set value, unless a software procedure for setting a new address is forced at power on. For further information please refer to the "4.7 Setting the node address: DIP B" section on page 33.
Default = 0000 0080h+NodeID

1015-00 Inhibit time EMCY

[Unsigned16, rw]
Inhibit time of the emergency messages (EMCY) expressed in multiples of 100 µs. When set to 0, this function is disabled.
Default = 0000h

1018 Identity object

- **01 Vendor-ID** provided by CIA organization [Unsigned32, ro]
  Default = 0000 012Eh
- **02 Product code** [Unsigned32, ro]
  Default = 0000 0000h
- **03 Revision number** [Unsigned32, ro]
  Default = 0000 0000h

1800 PDO1 parameters

PDO1 message is used by default for cyclic transmission of the position value.
For more information refer to the "6.6 PDO objects" section on page 45.
See the **6200-00 Cyclic timer** object to set the cyclic timer.

- **01 COB-ID of PDO1** [Unsigned32, rw]

Bit number	Value	Meaning
31 (MSB)	0	PDO exists / is valid
	1	PDO does not exist / is not valid
30	0	RTR allowed on this PDO **(not implemented)**
	1	no RTR allowed on this PDO
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)
28 ... 11	0	if bit 29 = 0
	X	if bit 29 = 1: bits 28-11 of 29-bit-COB-ID
10 ... 0 (LSB)	X	bits 10-0 of COB-ID

Default = 4000 0180h+NodeID (no RTR, COB-ID)

WARNING
It is mandatory to set the bit 30 of COB-ID to 1 (value 0 is not allowed). This means that "No RTR is allowed on the PDO".
If the node address is set using the internal DIP switches (i.e. at least one DIP switch for setting the node has HIGH logic level = 1), when the power is turned on, this object is always forced to the default value. Otherwise, if the node address is set via software (i.e. all DIP switches for setting the node have LOW logic level = 0) it retains the set value, unless a software procedure for setting a new address is forced at power on.

- **02 Transmission type** [Unsigned8, rw]

Transmission type	PDO transmission	
00h (0)	Acyclic, synchronous	**not implemented**

01h ... F0h (1 ... 240)	Cyclic, synchronous	**implemented**
F1h ... FBh (241 ... 251)	**not implemented - reserved**	
FCh (252)	Synchronous, RTR only	**not implemented**
FDh (253)	Asynchronous, RTR only	**not implemented**
FEh (254)	Asynchronous, manufacturer specific	**implemented**
FFh (255)	Asynchronous, device profile specific	**not implemented**

Default = FEh (cyclic transmission, see hereafter and the **6200-00 Cyclic timer** object)

WARNING
Following an attempt to set the **Transmission Type** to 0, the value is accepted but the PDO message is not sent; following an attempt to change the **Transmission Type** to any other value that is not supported by the device, an abort message (abort code = 0609 0030h: **Invalid value for parameter**) is generated.
If the value next to the **6200-00 Cyclic timer** object ≠ 0, the PDO message is sent cyclically and the interval between two messages is the time set next to the **6200-00 Cyclic timer** object; otherwise, if the value next to the **6200-00 Cyclic timer** object = 0, the PDO message is not sent.

NOTE
Please refer to the "7 – Setting-up" section on page 81 for an example of how the **1800 PDO1 parameters** object is to be set.

1801 PDO2 parameters
PDO2 message is used by default for synchronous transmission of the position value. For more information refer to section "6.6 PDO objects" on page 45.

- **01 COB-ID of the PDO2** [Unsigned32, rw]

Bit number	Value	Meaning
31 (MSB)	0	PDO exists / is valid
	1	PDO does not exist / is not valid
30	0	RTR allowed on this PDO **(not implemented)**
	1	no RTR allowed on this PDO
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)

28 ... 11	0	if bit 29 = 0
	X	if bit 29 = 1: bits 28-11 of 29-bit-COB-ID
10 ... 0 (LSB)	X	bits 10-0 of COB-ID

Default = 4000 0280h+NodeID (no RTR, COB-ID)

WARNING
It is mandatory to set the bit 30 of COB-ID to 1 (value 0 is not allowed). This means that "No RTR is allowed on the PDO".
If the node address is set using the internal DIP switches (i.e. at least one DIP switch for setting the node has HIGH logic level = 1), when the power is turned on, this object is always forced to the default value. Otherwise, if the node address is set via software (i.e. all DIP switches for setting the node have LOW logic level = 0) it retains the set value, unless a software procedure for setting a new address is forced at power on.

- **02 Transmission type** [Unsigned8, rw]

Transmission type	PDO transmission	
00h (0)	Acyclic, synchronous	**not implemented**
01h ... F0h (1 ... 240)	Cyclic, synchronous	**implemented**
F1h ... FBh (241 ... 251)	**not implemented - reserved**	
FCh (252)	Synchronous, RTR only	**not implemented**
FDh (253)	Asynchronous, RTR only	**not implemented**
FEh (254)	Asynchronous, manufacturer specific	**implemented**
FFh (255)	Asynchronous, device profile specific	**not implemented**

Default = 01h (synchronous transmission at each SYNC command)
The position value is transmitted after the set number of SYNC commands.
The interval between the SYNC commands must be set next to this **1801 PDO2 parameters**, sub 2 object.

WARNING
Following an attempt to set the **Transmission Type** to 0, the value is accepted but the PDO message is not sent; following an attempt to change the **Transmission Type** to any other value that is not supported by the device, an abort message (abort code = 0609 0030h: **Invalid value for parameter**) is generated.
If the value next to the **6200-00 Cyclic timer** object ≠ 0, the PDO message is sent cyclically and the interval between two messages is the time set next to the **6200-00 Cyclic timer** object; otherwise, if the

value next to the **6200-00 Cyclic timer** object = 0, the PDO message is not sent.

NOTE
Please refer to the "7 – Setting-up" section on page 81 for an example of how the **1801 PDO2 parameters** object is to be set.

1802 PDO3 parameters

PDO3 message is used by default for synchronous transmission of the position value. For more information refer to the "6.6 PDO objects" section on page 45.

- **01 COB-ID of the PDO3** [Unsigned32, rw]

Bit number	Value	Meaning
31 (MSB)	0	PDO exists / is valid
	1	PDO does not exist / is not valid
30	0	RTR allowed on this PDO **(not implemented)**
	1	no RTR allowed on this PDO
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)
28 ... 11	0	if bit 29 = 0
	X	if bit 29 = 1: bits 28-11 of 29-bit-COB-ID
10 ... 0 (LSB)	X	bits 10-0 of COB-ID

Default = C000 0380h+NodeID (disable, no RTR)

WARNING
It is mandatory to set the bit 30 of COB-ID to 1 (value 0 is not allowed). This means that "No RTR is allowed on the PDO".
If the node address is set using the internal DIP switches (i.e. at least one DIP switch for setting the node has HIGH logic level = 1), when the power is turned on, this object is always forced to the default value. Otherwise, if the node address is set via software (i.e. all DIP switches for setting the node have LOW logic level = 0) it retains the set value, unless a software procedure for setting a new address is forced at power on.

- **02 Transmission type** [Unsigned8, rw]

Transmission type	PDO transmission	
00h (0)	Acyclic, synchronous	**not implemented**
01h ... F0h (1 ... 240)	Cyclic, synchronous	**implemented**

F1h ... FBh (241 ... 251)	not implemented - reserved	
FCh (252)	Synchronous, RTR only	**not implemented**
FDh (253)	Asynchronous, RTR only	**not implemented**
FEh (254)	Asynchronous, manufacturer specific	**implemented**
FFh (255)	Asynchronous, device profile specific	**not implemented**

Default = 01h (synchronous transmission at each SYNC command)
The position value is transmitted after the set number of SYNC commands.
The interval between the SYNC commands must be set next to this **1802 PDO3 parameters**, sub 2 object.

WARNING
Following an attempt to set the **Transmission Type** to 0, the value is accepted but the PDO message is not sent; following an attempt to change the **Transmission Type** to any other value that is not supported by the device, an abort message (abort code = 0609 0030h: **Invalid value for parameter**) is generated.
If the value next to the **6200-00 Cyclic timer** object ≠ 0, the PDO message is sent cyclically and the interval between two messages is the time set next to the **6200-00 Cyclic timer** object; otherwise, if the value next to the **6200-00 Cyclic timer** object = 0, the PDO message is not sent.

NOTE
Please refer to the "7 – Setting-up" section on page 81 for an example of how the **1802 PDO3 parameters** object is to be set.

1803 PDO4 parameters

PDO4 is used by default for cyclic transmission of the velocity value.
For more information refer to the "6.6 PDO objects" section on page 45.
See the **6200-00 Cyclic timer** object to set the cyclic timer.

- **01 COB-ID of PDO4** [Unsigned32, rw]

Bit number	Value	Meaning
31 (MSB)	0	PDO exists / is valid
	1	PDO does not exist / is not valid
30	0	RTR allowed on this PDO **(not implemented)**
	1	no RTR allowed on this PDO

29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)
28 ... 11	0	if bit 29 = 0
	X	if bit 29 = 1: bits 28-11 of 29-bit-COB-ID
10 ... 0 (LSB)	X	bits 10-0 of COB-ID

Default = C000 0480h+NodeID (no RTR, COB-ID)

WARNING
It is mandatory to set the bit 30 of COB-ID to 1 (value 0 is not allowed). This means that "No RTR is allowed on the PDO".
If the node address is set using the internal DIP switches (i.e. at least one DIP switch for setting the node has HIGH logic level = 1), when the power is turned on, this object is always forced to the default value. Otherwise, if the node address is set via software (i.e. all DIP switches for setting the node have LOW logic level = 0) it retains the set value, unless a software procedure for setting a new address is forced at power on.

- **02 Transmission type** [Unsigned8, rw]

Transmission type	PDO transmission	
00h (0)	Acyclic, synchronous	**not implemented**
01h ... F0h (1 ... 240)	Cyclic, synchronous	**implemented**
F1h ... FBh (241 ... 251)	**not implemented - reserved**	
FCh (252)	Synchronous, RTR only	**not implemented**
FDh (253)	Asynchronous, RTR only	**not implemented**
FEh (254)	Asynchronous, manufacturer specific	**implemented**
FFh (255)	Asynchronous, device profile specific	**not implemented**

Default = FEh (cyclic transmission, see hereafter and the **6200-00 Cyclic timer** object)

WARNING
Following an attempt to set the **Transmission Type** to 0, the value is accepted but the PDO message is not sent; following an attempt to change the **Transmission Type** to any other value that is not supported by the device, an abort message (abort code = 0609 0030h: **Invalid value for parameter**) is generated.
If the value next to the **6200-00 Cyclic timer** object ≠ 0, the PDO message is sent cyclically and the interval between two messages is the time set next to the **6200-00 Cyclic timer** object; otherwise, if the

value next to the **6200-00 Cyclic timer** object = 0, the PDO message is not sent.

NOTE
Please refer to the "7 – Setting-up" section on page 81 for an example of how the **1803 PDO4 parameters** object is to be set.

NOTE
- The transmission of PDO1, PDO2, PDO3, and PDO4 messages can be enabled (or disabled) by setting to "0" (or "1") the most significant bit (msb) used by PDO (**180xh**, sub1 object).
- The cyclic transmission or synchronous transmission can be modified by setting the **180xh** sub 2 object. If you need the position value (or the velocity value) to be transmitted every "n" SYNC commands, you must set the "n" value next to the **180xh** sub 2 object:
  01h: synchronous transmission at each SYNC command;
  02h: synchronous transmission every two SYNC commands;
  ...
  FEh: cyclic transmission:
  if **6200-00 Cyclic timer** ≠ 0 → "cyclic transmission": the cycle time is set next to the **6200-00 Cyclic timer** object;
  if **6200-00 Cyclic timer** = 0 → the PDO message is not sent.

1A00-01 PDO1 mapping parameter

[Unsig32, rw]
This object contains the mapping of the PDO the encoder uses to transmit the position value, according to the DS 406 device profile specifications.
This object describes the content of the PDO by its index, sub-index, and length.
The length contains the length of the application object expressed in bits.

31 24	23 16	15 8	7 0
Index		Sub-Index	Length
MSB			LSB

Default = 6004 0020h = **6004-00 Position value** object, the length is 32 bits.

1A01-01 PDO2 mapping parameter

[Unsig32, rw]
See the **1A00-01 PDO1 mapping parameter**, sub 1 object
Default = 6004 0020h

1A02-01 PDO3 mapping parameter

[Unsig32, rw]
See the **1A00-01 PDO1 mapping parameter**, sub 1object
Default = 6004 0020h

1A03-01 PDO4 mapping parameter

[Unsig32, rw]
This object contains the mapping of the PDO the encoder uses to transmit the velocity value, according to the manufacturer profile.
This object describes the content of the PDO by its index, sub-index and length.
The length contains the length of the application object expressed in bits.

31 24	23 16	15 8	7 0
Index		Sub-Index	Length
MSB			LSB

Default = 3006 0020h = **3006-00 Velocity value** object, the length is 32 bits.

6.8.2 Manufacturer Specific Profile Area objects

2104-00 Limit switch min.

[Unsigned32, rw]
This object is used to set the lowest software limit switch (-).
In the encoder position is greater than the value set in the object, then the bit 12 of the **6500-00 Operating status** object will be set to "0".
If the encoder position is less than the value set in this object, then the bit 12 of the **6500-00 Operating status** object will be set to "1".
To enable this function set the bit 12 **Limit switch min.** of the **6000-00 Operating parameters** object to "1".
Default = 0000 0010h

2105-00 Limit switch max.

[Unsigned32, rw]
This object is used to set the highest software limit switch (+).
If the encoder position is less than the value set in this object, then the bit 13 of the **6500-00 Operating status** object will be set to "0".
If the encoder position is greater than the value set in this object, then the bit 13 of the **6500-00 Operating status** object will be set to "1".
To enable this function set bit 13 **Limit switch max.** of the **6000-00 Operating parameters** object to "1".
Default = 003F FFF0h

3000-00 Baud rate

[Unsigned8, rw]
This object is meant to set the baud rate (transmission rate) according to the following table:

Data byte	Baud rate
00h	20 Kbit/s
01h	50 Kbit/s
02h	100 Kbit/s
03h	125 Kbit/s
04h	250 Kbit/s
05h	**500 Kbit/s (default)**
06h	800 Kbit/s
07h	1000 Kbit/s

The bit rate is set through the **3000-00 Baud rate** object only if the bit 4 in the DIP A DIP switch is set to "OFF". If the bit 4 in the DIP A DIP switch is set to "ON", the bit rate is set by DIP A. For any further information refer to the "4.6 Setting the baud rate: DIP A" section on page 32.

To change the baud rate value you have to:

- set the **3000-00 Baud rate** object;
- send a **Reset node** command (or a **Reset communication** command);

- save the parameter;
- set the Master to the new baud rate.

Default = 05h

Master → Encoder

COB-ID	Cmd	Index		Sub	Data byte
600+ID	2F	00	30	00	see table

Encoder → Master (confirmation)

COB-ID	Cmd	Index		Sub	Data byte
580+ID	60	00	30	00	00

Master → Encoder (**Reset node**)

COB-ID	Cmd	Slave ID
000	81	ID

Set the Master device to the new baud rate:

Encoder → Master (Boot-up with new baud rate)

COB-ID	Cmd
700+ID	00

NOTE
To save the new Baud rate value execute the store parameters function (see the **1010-01 Store parameters** object).
When the power is turned off, parameters not saved will be lost.

3001-00 Node-ID

[Unsigned8, rw]
This object defines the node identifier (node ID) of the device. The node addresses are allowed in the range 1 to 127. The default value is 1.

The node number is set through the **3001-00 Node-ID** object only if all bits of DIP B are set to "OFF". If one bit at least of DIP B is set to "ON" the node number is set through DIP B. For any further information refer to the "4.7 Setting the node address: DIP B" section on page 33.

To change the Node-ID value you have to:

- set the **3001-00 Node-ID** object;
- send a **Reset node** command;
- save the parameter.

Default = 01h

Master → Encoder

COB-ID	Cmd	Index		Sub	Data byte
600+ID	2F	01	30	00	new Node-ID

Encoder → Master (confirmation)

COB-ID	Cmd	Index		Sub	Data byte
580+ID	60	01	30	00	00

Master → Encoder (**Reset node**)

COB-ID	Cmd	Slave ID
000	81	old ID

Encoder → Master (Boot-up with new Node-ID)

COB-ID	Cmd
700+ID	00

NOTE

To save the new Node-ID value execute the store parameters function (see the **1010-01 Store parameters** object).
When the power is turned off, parameters not saved will be lost.

3005-00 Velocity format

[Unsigned8, rw]
This attribute defines the engineering units for the velocity value shown in the next **3006-00 Velocity value** object.

00h = steps/s:	number of steps per second (default);
01h = rpm:	revolutions per minute.

Default = 00h

3006-00 Velocity value

[Unsigned32, ro]
This attribute shows the current output speed value detected by the position sensor and calculated every 100 ms.
The value can be expressed in either steps per second or revolutions per minute according to the setting in the previous **3005-00 Velocity format** object.
The value is transmitted according to the settings in the **1803 PDO4 parameters** object.

6.8.3 Device Profile Area objects (DS 406)

6000-00 Operating parameters

[Unsigned16, rw]

Bit	Function	bit = 0	bit = 1
0	**Code sequence**	**CW (clockwise)**	CCW (counter clockwise)
1	not used		
2	**Scaling function**	**disabled**	enabled
3 ... 11	not used		
12	**Limit switch min.**	**disabled**	enabled
13	**Limit switch max.**	**disabled**	enabled
14 and 15	not used		

Default values are highlighted in bold
Default = 0000h

Code sequence

It defines whether the position value outputted by the transducer increases when the encoder shaft rotates clockwise (CW) or counter-clockwise (CCW). If **Code sequence** = 0, the position value increases when the encoder shaft rotates clockwise; on the contrary, if **Code sequence** = 1, the position value increases when the encoder shaft rotates counter-clockwise. CW and CCW rotations are viewed from shaft end.

To know whether the **Code sequence** is currently set to CW or CCW, you can read the bit 0 **Code sequence** of the 6500-00 Operating status object, see on page 71.

Scaling function

If this option is disabled, the device uses the physical resolution values (see the 6501-00 Hardware counts per revolution and 6502-00 Hardware number of turns objects) to provide the position information; if it is enabled, it uses the custom resolution values set in the 6001-00 Measuring units per revolution and 6002-00 Total resolution objects in accordance with the following relation:

Transmitted position =

$$\frac{\text{6001-00 Measuring units per revolution}}{\text{6501-00 Hardware counts per revolution}} * \text{real position} \leq \text{6002-00 Total resolution}$$

To know whether the **Scaling function** is currently enabled, you can read the bit 2 **Scaling function** of the 6500-00 Operating status object, see on page 71.

WARNING
When you enable the scaling function (bit 2 **Scaling function** = 1), please enter scaled values next to the 6001-00 Measuring units per revolution and 6002-00 Total resolution objects that are consistent with the physical values.

WARNING
Every time you enable the scaling function and/or change the scaling values (see the 6001-00 Measuring units per revolution and 6002-00 Total resolution objects), then you are required to set a new preset value (see the 6003-00 Preset value object) and finally save the new parameters (see the 1010-01 Store parameters object).

Limit switch min.
Limit switch max.
They allow to enable / disable the function of the 2104-00 Limit switch min. and 2105-00 Limit switch max. objects. For further information see on page 60.

To know whether the **Limit switch min.** / **Limit switch max.** is currently enabled, you can read the bit 12 **Limit switch min.** and the bit 13 **Limit switch max.** of the 6500-00 Operating status object, see on page 71.

6001-00 Measuring units per revolution

[Unsig32, rw]

WARNING
This object is active only if the bit 2 **Scaling function** in the 6000-00 Operating parameters object is set to "=1"; otherwise it is ignored and the system uses the physical values (6501-00 Hardware counts per revolution and 6502-00 Hardware number of turns) to calculate the position information.

This object sets a custom number of distinguishable steps per revolution custom singleturn resolution).
The set value must range between 1 and 6501-00 Hardware counts per revolution (see encoder label). If you enter out-of-range values, the number of information per revolution is forced to the physical singleturn resolution.

To avoid counting errors, check that

$$\frac{\text{6501-00 Hardware counts per revolution}}{\text{6001-00 Measuring units per revolution}} = \text{integer value.}$$

Allowed values are equal to or less than **6501-00 Hardware counts per revolution** (see encoder label).

Default =	4,096	for Ax5812/...
	8,192	for Ax5813/...

Setting the resolution per revolution 6001-00 Measuring units per revolution (2^{10} = 0000 0400h)

Master → Encoder (Set request)

COB-ID	Cmd	Index		Sub	Process data			
600+ID	23	01	60	00	00	40	00	00

Encoder → Master (Set confirmation)

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	01	60	00	00	00	00	00

WARNING
When you set a new value next to the **6001-00 Measuring units per revolution** object, please always check also the **6002-00 Total resolution** object value and be sure that the resulting number of revolutions complies with the physical number of revolutions of the device (see the **6502-00 Hardware number of turns** object, it can be 1 or 4,096, see the order code).

Let's suppose that the AM5813/4096... encoder is programmed as follows:
6001-00 Measuring units per revolution = 8,192 cpr
6002-00 Total resolution = 33,554,432 = 8,192 cpr * 4,096 revolutions
Let's set a new singleturn resolution, for instance: **6001-00 Measuring units per revolution** = 360 cpr

If we do not change the **6002-00 Total resolution** value at the same time, we will get the following result:

Number of revolutions =	33,554,432 (**6002-00 Total resolution**)	= 93,206.755...
	360 (**6001-00 Measuring units per revolution**)	

As you can see, the encoder is required to carry out more than 93,000 revolutions, this cannot be as the hardware number of revolutions is, as stated, 4,096 (see the **6502-00 Hardware number of turns** object). When this happens, the encoder falls into an error.

WARNING
Every time you enable the scaling function (bit 2 **Scaling function** in the 6000-00 Operating parameters object) and/or change the value in the scaled values (6001-00 Measuring units per revolution and 6002-00 Total resolution objects), then you are required to set a new preset value (see the 6003-00 Preset value object) and finally save the new parameters (see the 1010-01 Store parameters object).

NOTE
Please refer to the "7 – Setting-up" section on page 81 for an example of how the 6001-00 Measuring units per revolution is to be set.

6002-00 Total resolution

[Unsigned32, rw]

WARNING
This object is active only if the bit 2 **Scaling function** in the 6000-00 Operating parameters object is set to "=1"; otherwise it is ignored and the system uses the physical values (6501-00 Hardware counts per revolution and 6502-00 Hardware number of turns) to calculate the position information.

This object sets a custom number of distinguishable steps over the total measuring range. The total resolution of the encoder results from the product of 6001-00 Measuring units per revolution by the **required number of revolutions**.
Allowed values are equal to or less than **Total hardware resolution** (6501-00 Hardware counts per revolution * 6502-00 Hardware number of turns, see encoder label).

Default =	4,096	for AS5812/...
	8,192	for AS5813/...
	16,777,216	for AM5812/4096...
	33,554,432	for AM5813/4096...

Setting the total resolution 6002-00 Total resolution (2^{24}=0100 0000h)

Master → Encoder (Set request)

COB-ID	Cmd	Index		Sub	Process data			
600+ID	23	02	60	00	00	00	00	01

Encoder → Master (Set confirmation)

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	02	60	00	00	00	00	00

WARNING

When you set a new value next to the 6002-00 Total resolution object, please always check also the 6001-00 Measuring units per revolution object value and be sure that the resulting number of revolutions complies with the **physical number of revolutions** of the device (see the 6502-00 Hardware number of turns object, it can be 1 or 4,096, see the order code).

Let's suppose that the AM5813/4096... encoder is programmed as follows:
6001-00 Measuring units per revolution = 8,192 cpr
6002-00 Total resolution = 33,554,432 = 8,192 cpr * 4,096 revolutions
Let's set a new singleturn resolution, for instance: 6001-00 Measuring units per revolution = 360 cpr
As the 6002-00 Total resolution must be greater than or equal to the 6001-00 Measuring units per revolution, the above setting is not allowed. When this happens, the encoder falls into an error.

EXAMPLE

Multiturn encoder AM58**13/4096**CB with "CC-CB-C" connection cap

The physical values are as follows:

- Hardware counts per revolution: 6501-00 Hardware counts per revolution = 8,192 (2^{13})
- Hardware number of turns: 6502-00 Hardware number of turns = 4,096 (2^{12})
- Total hardware resolution: = 33,554,432 (2^{25})

The specific installation requires 2,048 counts/rev. ∗ 1,024 turns:

- Enable the scaling function: bit 2 **Scaling function** in the 6000-00 Operating parameters object = "1"
- Custom resolution per revolution: 6001-00 Measuring units per revolution = 2,048 (0000 0800h)
- Total resolution: 6002-00 Total resolution = 2,048 * 1,024 = 2,097,152 (0020 0000h)

NOTE

To avoid counting errors we suggest values which are power of 2 (2^n: 2, 4, ..., 2048, 4096, 8192,...) to be set in the 6001-00 Measuring units per revolution and 6002-00 Total resolution objects.

WARNING

Every time you enable the scaling function (bit 2 **Scaling function** in the 6000-00 Operating parameters object) and/or change the value in the scaled values (6001-00 Measuring units per revolution and 6002-00 Total resolution objects), then you are required to set a new preset value (see the

6003-00 Preset value object) and finally save the new parameters (see the 1010-01 Store parameters object).

NOTE
Please refer to the "7 – Setting-up" section on page 81 for an example of how the 6002-00 Total resolution is to be set.

6003-00 Preset value

[Unsigned32, rw]
This object allows to set the encoder position to a Preset value. The Preset function is meant to assign a desired value to a physical position of the encoder shaft. The chosen position will get the value set next to this object and all the previous and the following positions will get a value according to it. This function is useful, for example, when the zero position of the encoder and the zero position of the axis need to match. The preset value will be set for the position of the encoder in the moment when the preset value is transmitted. We suggest setting the preset value when the encoder is in stop.
Default = 0000 0000h

EXAMPLE
Let's take a look at the following example to better understand the preset function and the meaning and use of the related objects: 6003-00 Preset value and 6509-00 Offset value.
The encoder position which is transmitted results from the following calculation:
Transmitted value = **read position** (it does not matter whether the position is physical or scaled) + 6003-00 Preset value - 6509-00 Offset value.
If you never set the 6003-00 Preset value and you never performed the preset setting, then the transmitted value and the read position are necessarily the same as 6003-00 Preset value = 0 and 6509-00 Offset value = 0.
When you set the 6003-00 Preset value and then execute the preset setting, the system saves the current encoder position in the 6509-00 Offset value object. It follows that the transmitted value and the 6003-00 Preset value are the same as **read position** - 6509-00 Offset value = 0; in other words, the value set next to the 6003-00 Preset value object is paired with the current position of the encoder as you wish.

For example, let's assume that the value "50" is set next to the 6003-00 Preset value object and you execute the preset setting when the encoder position is "1000". In other words, you want to receive the value "50" when the encoder reaches the position "1000".

We will obtain the following information sequence:
Transmitted value = **read position** (="1000") + 6003-00 Preset value (="50") - 6509-00 Offset value (="1000") = 50.

The following transmitted value will be:
Transmitted value = **read position** (="1001") + 6003-00 Preset value (="50") - 6509-00 Offset value (="1000") = 51.
And so on.

To set the preset value you must send the following command:
Set the Preset value 6003-00 Preset value (preset = 1000 = 03E8h)

Master → Encoder (Set request)

COB-ID	Cmd	Index		Sub	Process data			
600+ID	23	03	60	00	E8	03	00	00

Encoder → Master (Set confirmation)

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	03	60	00	00	00	00	00

NOTE

- If the scaling function is <u>disabled</u> (see the bit 2 **Scaling function** in the 6000-00 Operating parameters object), 6003-00 Preset value must be less than or equal to the **Total hardware resolution** - 1 (6501-00 Hardware counts per revolution * 6502-00 Hardware number of turns - 1).
- If the scaling function is <u>enabled</u> (see the bit 2 **Scaling function** in the 6000-00 Operating parameters object), 6003-00 Preset value must be less than or equal to 6002-00 Total resolution - 1.

WARNING

Check the value in the 6003-00 Preset value object and perform the preset operation every time you set a new **Code sequence** and/or change the scaled values (6001-00 Measuring units per revolution and/or 6002-00 Total resolution).

NOTE

Please refer to the "7 – Setting-up" section on page 81 for an example of how the 6003-00 Preset value is to be set.

6004-00 Position value

[Unsigned32, ro]
This object contains the position value of the encoder.
The output value is scaled according to the scaling parameters (if the scaling function is enabled), see the bit 2 **Scaling function** of the 6000-00 Operating parameters object.
The position value is transmitted cyclically or synchronously according to the settings in the 1800 PDO1 parameters, 1801 PDO2 parameters, and 1802 PDO3 parameters objects (see on page 52 ff).

6200-00 Cyclic timer

[Unsigned16, rw]
The cyclic timer value is used in asynchronous transmission mode (**Transmission Type** = FEh) to set the interval between two following PDO transmissions during a cyclic communication.
If the value next to this 6200-00 Cyclic timer object $\neq$ 0, the PDO message is sent cyclically and the interval between two messages is the time set in this object; otherwise, if the value next to this 6200-00 Cyclic timer object = 0, the PDO message is not sent.
The value is expressed in milliseconds. See on pages 45 and 52 ff.
Default = 0000h

Setting the cyclic time: 6200-00 Cyclic timer (100 ms = 64h)

Master → Encoder

COB-ID	Cmd	Index		Sub	Process data			
600+ID	2B	00	62	00	64	00	-	-

Encoder → Master

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	00	62	00	00	00	-	-

NOTE
Please refer to the "7 – Setting-up" section on page 81 for an example of how the 6200-00 Cyclic timer is to be set.

6500-00 Operating status

[Unsigned16, ro]

Bit	Function	bit = 0	bit = 1
0	**Code sequence**	CW clockwise	CCW counter-clockwise
1	not used		
2	**Scaling function**	Disabled	Enabled
3...11	not used		
12	**Limit switch min.**	position > **2104-00 Limit switch min.**	position < **2104-00 Limit switch min.**
13	**Limit switch max.**	position < **2105-00 Limit switch max.**	position > **2105-00 Limit switch max.**
14	not used		
15	**Current operating state**	Stopped / Pre-operational	Operational

Code sequence

It shows the value that is currently set through the bit 0 **Code sequence** in the **6000-00 Operating parameters** object. If the bit is "=0" the output encoder position value has been set to increase (count-up information) when the encoder shaft turns clockwise (CW).; otherwise, if the bit is "=1" the output encoder position value has been set to increase when the encoder shaft turns counter-clockwise (CWW). CW and CCW rotations are viewed from the shaft end. To set the code sequence to either CW or CCW you must set the bit 0 **Code sequence** of the **6000-00 Operating parameters** object to 0 / 1. For any further information on setting and using the counting direction refer to the **6000-00 Operating parameters** object on page 63.

Scaling function

It shows the value that is currently set through the bit 2 **Scaling function** of the **6000-00 Operating parameters** object. In other words, it is intended to show whether the scaling function is disabled or enabled. If the value is "=0" the scaling function is disabled; if the value is "=1" instead the scaling function is enabled. To disable / enable the scaling function you must set the bit 2 **Scaling function** of the **6000-00 Operating parameters** object to 0 / 1. For any further information on setting and using the scaling function refer to the **6000-00 Operating parameters** object on page 63.

Limit switch min.

If the encoder position is greater than the value set in the **2104-00 Limit switch min.** object, the bit 12 of this object is set to "0".

If the encoder position is less than the value set in the 2104-00 Limit switch min. object, the bit 12 of this object is set to "1".
To enable this function set the bit 12 **Limit switch min.** of the 6000-00 Operating parameters object to "1". For any further information on setting and using the lower limit switch refer to the 2104-00 Limit switch min. object on page 60.

Limit switch max.
If the encoder position is less than the value set in the 2105-00 Limit switch max. object, the bit 13 of this object is set to "0".
If the encoder position is greater than the value set in the 2105-00 Limit switch max. object, the bit 13 of this object is set to "1".
To enable this function set the bit 13 **Limit switch max.** of the 6000-00 Operating parameters object to "1". For any further information on setting and using the higher limit switch refer to the 2105-00 Limit switch max. object on page 60.

Current operating state
It shows the current operating state of the unit. For further information on the available states see the "6.2 State machine" section on page 42.
bit 15 = 0: Stopped or Pre-operational state;
bit 15 = 1: Operational state.

6501-00 Hardware counts per revolution

[Unsigned32, ro]

WARNING
This object is active only if the bit 2 **Scaling function** in the 6000-00 Operating parameters object is set to "=0"; otherwise it is ignored and the system uses the custom values (6001-00 Measuring units per revolution and 6002-00 Total resolution) to calculate the position information.

This object is intended to show the number of physical distinguishable steps per each revolution provided by the hardware (physical singleturn resolution).
If the bit 2 **Scaling function** in the 6000-00 Operating parameters object is set to "=0" the system uses the value in this object (and the value in the 6502-00 Hardware number of turns object) to calculate the position information. If the bit 2 **Scaling function** in the 6000-00 Operating parameters object is set to "=1" the system uses the custom values (6001-00 Measuring units per revolution and 6002-00 Total resolution) to calculate the position information.
If you want to set a custom singleturn resolution see the 6001-00 Measuring units per revolution object.

NOTE
Please refer to the "7 – Setting-up" section on page 81 for an example of how the **6501-00 Hardware counts per revolution** object can be read.

6502-00 Hardware number of turns

[Unsig16, ro]

WARNING
This object is active only if the bit 2 **Scaling function** in the **6000-00 Operating parameters** object is set to "=0"; otherwise it is ignored and the system uses the custom values (**6001-00 Measuring units per revolution** and **6002-00 Total resolution**) to calculate the position information.

This object is intended to show the number of physical revolutions provided by the hardware.
The **Total hardware resolution** results from **6501-00 Hardware counts per revolution** * **6502-00 Hardware number of turns**.
If the bit 2 **Scaling function** in the **6000-00 Operating parameters** object is set to "=0" the system uses the value in this object (and the value in the **6501-00 Hardware counts per revolution** object) to calculate the position information. If the bit 2 **Scaling function** in the **6000-00 Operating parameters** object is set to "=1" the system uses the custom values (**6001-00 Measuring units per revolution** and **6002-00 Total resolution**) to calculate the position information.
If you want to set a custom number of revolutions see the **6001-00 Measuring units per revolution** and **6002-00 Total resolution** objects.

NOTE
Please refer to the "7 – Setting-up" section on page 81 for an example of how the **6502-00 Hardware number of turns** object can be read.

6504-00 Supported alarms

[Unsigned16, ro]
This object contains the information on the alarms supported by the encoder. No alarms are supported in this encoder.
Default = 0000h (Alarms not supported).

6506-00 Supported warnings

[Unsigned16, ro]
This object contains the information on the warnings supported by the encoder. No warnings are supported in this encoder.
Default = 0000h (Warnings not supported).

6507-00 Profile and software version

[Unsig32, ro]
It shows the version of both the profile and the software.
Profile version for encoders = 3.1
Software version = 1.1
Default = 0301 0101h

6508-00 Operating time

[Unsigned32, ro]
This object contains the information on the operating time. The operating time monitor stores the operating time for the encoder expressed in operating hours. The operating time is stored in the encoder non-volatile memory as long as the encoder is power supplied.
This object is currently not used in this encoder.
Default = FFFF FFFFh (not used)

6509-00 Offset value

[Integer32, ro]
As soon as you activate the preset, the current position value of the encoder is saved in this object. The offset value is then used in the preset function in order to calculate the encoder position value to be transmitted. To zero set the value in this object you must upload the factory default values (see the **1011-01 Restore default parameters** object on page 50).
For any further information on the preset function and the meaning and use of the related objects and commands **6003-00 Preset value** and **6509-00 Offset value** refer to page 68.
Default = 0000 0000h

650A-01 Manufacturer offset value

[Integer32, ro]
This object contains the manufacturer offset value. This is the difference between the physical zero position of the encoder (zero set mechanically) and the zero position set by the manufacturer (zero set via software).
Default = 0000 0000h

650B-00 Serial number

[Unsigned32, ro]
This object contains the serial number of the encoder.
This object is currently not used in this encoder.
Default = FFFF FFFFh (not used)

NOTE
To save the new parameters execute the store parameters function (see the **1010-01 Store parameters** object).
When the power is turned off or in case of **Reset node** and **Restore node** commands, parameters not saved are lost.

6.9 SDO abort codes

Here following is the list and the meaning of the SDO abort codes indicated by CANopen but not necessarily supported by the manufacturer. For complete information please refer to the "SDO abort transfer protocol" section in the "CiA Draft Standard 301" document available at the address www.can-cia.org.

Abort code	Description
0503 0000h	Toggle bit not alternated.
0504 0000h	SDO protocol timed out.
0504 0001h	Client/server command specifier not valid or unknown.
0504 0002h	Invalid block size (block mode only).
0504 0003h	Invalid sequence number (block mode only).
0504 0004h	CRC error (block mode only).
0504 0005h	Out of memory.
0601 0000h	Unsupported access to an object.
0601 0001h	Attempt to read a write only object.
0601 0002h	Attempt to write a read only object.
0602 0000h	Object does not exist in the object dictionary.
0604 0041h	Object cannot be mapped to the PDO.
0604 0042h	The number and length of the objects to be mapped would exceed PDO length.
0604 0043h	General parameter incompatibility reason.
0604 0047h	General internal incompatibility in the device.
0606 0000h	Access failed due to an hardware error.
0607 0010h	Data type does not match, length of service parameter does not match
0607 0012h	Data type does not match, length of service parameter too high
0607 0013h	Data type does not match, length of service parameter too low
0609 0011h	Sub-index does not exist.
0609 0030h	Invalid value for parameter (download only).
0609 0031h	Value of parameter written too high (download only).
0609 0032h	Value of parameter written too low (download only).
0609 0036h	Maximum value is less than minimum value.
060A 0023h	Resource not available: SDO connection
0800 0000h	General error
0800 0020h	Data cannot be transferred or stored to the application.
0800 0021h	Data cannot be transferred or stored to the application because of local control.
0800 0022h	Data cannot be transferred or stored to the application because of the present device state.
0800 0023h	Object dictionary dynamic generation fails or no object dictionary is present (e.g. object dictionary is generated from file and generation fails

	because of an file error).
0800 0024h	No data available

6.10 Emergency objects

Emergency (EMCY) messages are issued by the device when an internal error occurs.

Structure of the EMCY message:

IDENTIFIER	**CAN Data Byte**			
COB-ID(hex)	0	1	2	3 ... 7
see the **1014-00 COB-ID EMCY** object	Error code		Error Sub-register	Specific code
	LSB	MSB	01	00...00

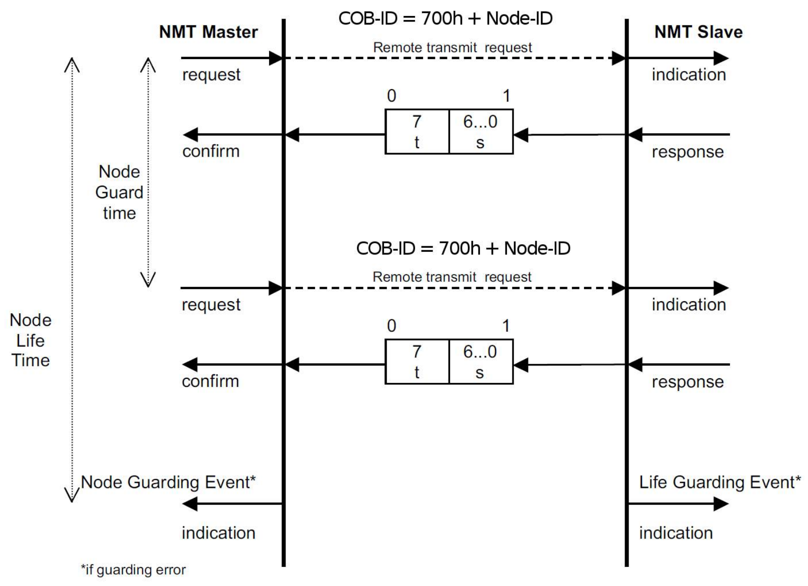
Available error codes indicated by CANopen but not necessarily supported by the manufacturer:

Error code	Description
0000h	Error reset or no error
1000h	Generic error, **Node guarding error**
2000h	Current – generic error
2100h	Current, CANopen device input side – generic
2200h	Current inside the CANopen device – generic
2300h	Current, CANopen device output side – generic
3000h	Voltage – generic error
3100h	Mains voltage – generic
3200h	Voltage inside the CANopen device – generic
3300h	Output voltage – generic
4000h	Temperature – generic error
4100h	Ambient temperature – generic
4200h	Device temperature – generic
5000h	CANopen device hardware – generic error
5530h	**Flash memory error**
6000h	CANopen device software – generic error
6100h	Internal software – generic
6200h	User software – generic
6300h	Data set – generic
7000h	Additional modules – generic error

8000h	Monitoring – generic error
8100h	Communication – generic
8110h	CAN overrun (objects lost)
8120h	CAN in error passive mode
8130h	Life guard error or heartbeat error
8140h	Recovered from bus off
8150h	CAN-ID collision
8200h	Protocol error - generic
8210h	PDO not processed due to length error
8220h	PDO length exceeded
8230h	DAM MPDO not processed, destination object not available
8240h	Unexpected SYNC data length
8250h	RPDO timeout
9000h	External error – generic error
F000h	Additional functions – generic error
FF00h	Device specific – generic error

6.11 Node guarding protocol

This protocol is used to detect remote error in the network. Each NMT Slave uses one remote COB for the Node Guarding protocol. This protocol implements the provided initiated Error Control services.



S: the state of the NMT Slave

4:	STOPPED
5:	OPERATIONAL

127: PRE-OPERATIONAL

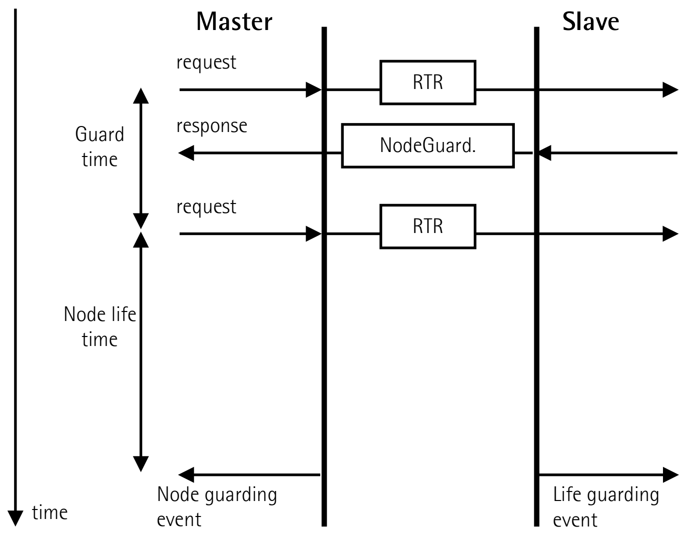
t: Toggle bit. The value of this bit must alternate between two consecutive responses from the NMT Slave. The value of the Toggle bit of the first response after the Node Guarding protocol becomes active is 0. The Toggle bit in the Node Guarding protocol is only reset to 0 when reset_communication is passed (no other change of the state resets the Toggle bit). If a response is received with the same value of the Toggle bit as in the preceding response then the new response is handled as if it was not received.

The NMT Master polls each NMT Slave at regular time intervals. This time-interval is called the guard time (see the **100C-00 Guard time** object) and may be different for each NMT Slave. The response of the NMT Slave contains the state of that NMT Slave. The **node life time** is given by the **100C-00 Guard time** object value multiplied by the **100D-00 Life time factor** object value. The node life time can be different for each NMT Slave. If the NMT Slave has not been polled during its life time, a remote node error is indicated through the 'Life Guarding Event' service.
A remote node error is indicated through the 'Node guarding event' service if:

- the remote transmit request is not confirmed within the node life time;
- the reported NMT Slave state does not match the expected state.

If it has been indicated that a remote error has occurred and the errors in the guarding protocol have disappeared, it will be indicated that the remote error has been resolved through the 'Node Guarding Event' and 'Life Guarding Event' services.

At system boot the "Node guarding protocol" is disabled; this protocol is enabled automatically as soon as the Master device sends an RTR message (Remote Transmission Request) the first time.



100C-00 Guard time:	interval between two RTR messages.
Node life time:	maximum time available for the encoder to receive an RTR message.

Node life time = **100C-00 Guard time** ∗ **100D-00 Life time factor**.

"Node guarding" is enabled if **Node life time** $\neq 0$.

If the Slave does not receive an RTR message before the **Node life time** has expired, it warns activating a "Life Guarding Event". Furthermore the red LED starts flashing so indicating the Node guarding error. The **1001-00 Error register** and **1003 Pre-defined error field** objects are updated and an error message is sent.
To reset the error send a **Reset node** command.

7 – Setting-up

Here following are some examples of transmission between Master and Slave devices.
A generic "ID" value is used to indicate the encoder address; the address of the Master is always 0. All values are expressed in hexadecimal notation.

7.1 Setting the Operational, Pre-operational state

NMT message — Master → Slave

	COB-ID	Cmd	Node
Operational:	000	01	ID
Pre-operational:	000	80	ID

7.2 Reading the value of the physical resolution per revolution

6501-00 Hardware counts per revolution

Master → Encoder

COB-ID	Cmd	Index		Sub	Process data			
601	40	01	65	00	-	-	-	-

Encoder → Master

COB-ID	Cmd	Index		Sub	Process data			
581	43	01	65	01	A0	A1	A2	A3

steps/rev. = ((A3<<24) | (A2<<16) | (A1<<8) | A0)

7.3 Reading the number of physical revolutions

6502-00 Hardware number of turns

Master → Encoder

COB-ID	Cmd	Index		Sub	Process data			
601	40	02	65	00	-	-	-	-

Encoder → Master

COB-ID	Cmd	Index		Sub	Process data			
581	43	02	65	01	B0	B1	B2	B3

N. rev. =((B3<<24) | (B2<<16) | (B1<<8) | B0)

7.4 Setting the resolution per revolution

6001-00 Measuring units per revolution (2^{10}=0000 0400h)

Master → Encoder (Set request)

COB-ID	Cmd	Index		Sub	Process data			
600+ID	23	01	60	00	00	40	00	00

Encoder → Master (Set confirmation)

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	01	60	00	00	00	00	00

7.5 Setting the total resolution

6002-00 Total resolution (2^{24}=0100 0000h)

Master → Encoder (Set request)

COB-ID	Cmd	Index		Sub	Process data			
600+ID	23	02	60	00	00	00	00	01

Encoder → Master (Set confirmation)

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	02	60	00	00	00	00	00

7.6 Setting the operating parameters

6000-00 Operating parameters

(**Code sequence**: 0 = CW, **Scaling function**: 1 = enabled, **Limit switch min.** / **Limit switch max.**: 0 = disabled) = $00000000\ 00000100_2$ = 0004h

Master → Encoder (Set request)

COB-ID	Cmd	Index		Sub	Process data			
600+ID	2B	00	60	00	04	00	-	-

Encoder → Master (Set confirmation)

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	00	60	00	00	00	-	-

7.7 Setting the preset value

6003-00 Preset value (preset = 1000_{10} = 03E8h)

Master → Encoder (Set request)

COB-ID	Cmd	Index		Sub	Process data			
600+ID	23	03	60	00	E8	03	00	00

Encoder → Master (Set confirmation)

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	03	60	00	00	00	00	00

7.8 Setting the Sync counter

1801 PDO2 parameters, sub 2 (n = 5 = 05h)

Master → Encoder (Set request)

COB-ID	Cmd	Index		Sub	Process data			
600+ID	2F	01	18	02	05	-	-	-

Encoder → Master (Set confirmation)

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	01	18	02	00	-	-	-

7.9 Disabling the Sync mode

1801 PDO2 parameters, sub 1

Read COB-ID used by PDO2:

Master → Encoder (Req request)

COB-ID	Cmd	Index		Sub	Process data			
600+ID	40	01	18	01	-	-	-	-

Encoder → Master (Req reply)

COB-ID	Cmd	Index		Sub	Process data			
580+ID	43	01	18	01	B0	B1	B2	B3

COB-ID used by PDO2 = ((B3<<24) | (B2<<16) | (B1<<8) | B0)

set the most significant bit to 1:

```
B3 |= 0x80;
```

Set new COB-ID used by PDO2 (1801 PDO2 parameters, sub 1):

Master → Encoder (Set request)

COB-ID	Cmd	Index		Sub	Process data			
600+ID	23	01	18	01	B0	B1	B2	B3

Encoder → Master (Set confirmation)

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	01	18	01	00	00	00	00

7.10 Enabling the Cyclic mode

Set cyclic time 6200-00 Cyclic timer (100 ms = 64h)

Master → Encoder (Set request)

COB-ID	Cmd	Index		Sub	Process data			
600+ID	2B	00	62	00	64	00	-	-

Encoder → Master (Set confirmation)

COB-ID	Cmd	Index		Sub	Process data			
580+ID	60	00	62	00	00	00	-	-

Read COB-ID used by PDO1 (**1800 PDO1 parameters**, sub 1):

Master → Encoder (Req request)

COB-ID	**Cmd**	**Index**		**Sub**	**Process data**			
600+ID	40	00	18	01	-	-	-	-

Encoder → Master (Req reply)

COB-ID	**Cmd**	**Index**		**Sub**	**Process data**			
580+ID	43	00	18	01	B0	B1	B2	B3

COB-ID used by PDO1 = ((B3<<24) | (B2<<16) | (B1<<8) | B0)
set the most significant bit to 0:

```
B3 &= 0x7F;
```

Set new COB-ID used by PDO1 (**1800 PDO1 parameters**, sub 1):

Master → Encoder (Set request)

COB-ID	**Cmd**	**Index**		**Sub**	**Process data**			
600+ID	23	00	18	01	B0	B1	B2	B3

Encoder → Master (Set confirmation)

COB-ID	**Cmd**	**Index**		**Sub**	**Process data**			
580+ID	60	00	18	01	00	00	00	00

NOTE
To save the new parameters execute the store parameters function (see the **1010-01 Store parameters** object).
When the power is turned off or in case of **Reset node** and **Restore node** commands, parameters not saved are lost.

8 – Default parameters list

Default values are expressed in hexadecimal notation.

Parameters list	Default values		
1000-00 Device type	0001 0196 0002 0196		
1001-00 Error register	00		
1003 Pre-defined error field	-		
1005-00 COB-ID SYNC message	0000 0080		
1008-00 Manufacturer device name	LIKA SRL [1]		
1009-00 Manufacturer hardware version	-		
100A-00 Manufacturer software version	-		
100C-00 Guard time	0000		
100D-00 Life time factor	00		
1014-00 COB-ID EMCY	NODEID+0000 0080		
1015-00 Inhibit time EMCY	0000		
1018 Identity object	-		
1800 PDO1 parameters, sub 1	NODEID+4000 0180		
1800 PDO1 parameters, sub 2	FE		
1801 PDO2 parameters, sub 1	NODEID+4000 0280		
1801 PDO2 parameters, sub 2	01		
1802 PDO3 parameters, sub 1	NODEID+C000 0380		
1802 PDO3 parameters, sub 2	01		
1803 PDO4 parameters, sub 1	NODEID+C000 0480		
1803 PDO4 parameters, sub 2	FE		
1A00-01 PDO1 mapping parameter, sub 1	6004 0020		
1A01-01 PDO2 mapping parameter, sub 1	6004 0020		
1A02-01 PDO3 mapping parameter, sub 1	6004 0020		
1A03-01 PDO4 mapping parameter, sub 1	3600 0020		
2104-00 Limit switch min.	0000 0010		
2105-00 Limit switch max.	003F FFF0		
3000-00 Baud rate	05		
3001-00 Node-ID	01		
3005-00 Velocity format	00		
6000-00 Operating parameters Code sequence Scaling function Limit switch min. Limit switch max.	0000 0 0 0 0		
6001-00 Measuring units per revolution	4096 [2] for Ax5812/... 8192 [2] for Ax5813/...		
6002-00 Total resolution	4096 [2] for AS5812/... 8192 [2] for AS5813/... 16,777,216 [2] for AM5812/4096... 33,554,432 [2] for AM5813/4096...		
6003-00 Preset value	0000 0000		

6200-00 Cyclic timer	0000		
6500-00 Operating status	-		
6504-00 Supported alarms	0000		
6506-00 Supported warnings	0000		
6507-00 Profile and software version	0301 0101		
6508-00 Operating time	FFFF FFFF		
6509-00 Offset value	0000 0000		
650A-01 Manufacturer offset value	0000 0000		
650B-00 Serial number	FFFF FFFF		

1	Text string
2	Decimal value

This page intentionally left blank

Document release	Release date	Description	HW	SW	EDS file version
1.0	18.01.2001	First issue	-	-	V1
1.1	06.02.2003	General review	-	-	V1
1.2	13.03.2004	General review	-	-	V1
1.3	24.03.2006	General review	-	-	V1
2.0	27.10.2006	SW and HW of CANopen® interface updated, general review	-	-	V1
2.1	28.11.2006	Cable output added (section 4)	-	-	V1
2.2	20.04.2007	DIP A and DIP B DIP switches added Bit 15 function for object 6500h added	-	-	V1
2.3	24.05.2007	Objects 100C and 100D updated	-	-	V2
2.4	20.05.2008	M12 electrical connections added	-	-	V2
2.5	15.01.2009	General review	-	-	V2
2.6	29.05.2009	Section 3.1 added	-	-	V2
2.7	14.10.2010	Section 4 updated	-	-	V2
3.0	12.10.2011	SW interface updated (velocity value added), general review	-	-	V2
3.1	13.02.2013	General review, section 4 updated	-	-	V2
3.2	07.02.2014	Objects 1014.00 – 1800.01 – 1801.01 – 1802.01 - 1803.01 updated, section 8 added, general review, Italian / English separate edition	-	-	V2
3.3	04.03.2015	Information about objects 1800h, 1801h, 1802h, 1803h, 6200h updated	-	-	V3.1, V4
3.4	24.06.2022	General review	-	-	V4

This device is to be supplied by a Class 2 Circuit or Low-Voltage Limited Energy or Energy Source not exceeding 30 Vdc. Refer to the order code for supply voltage rate.
Ce dispositif doit être alimenté par un circuit de Classe 2 ou à très basse tension ou bien en appliquant une tension maxi de 30Vcc. Voir le code de commande pour la tension d'alimentation.

Dispose separately

Lika Electronic
Via S. Lorenzo, 25 • 36010 Carrè (VI) • Italy

Tel. +39 0445 806600
Fax +39 0445 806699

info@lika.biz • www.lika.biz