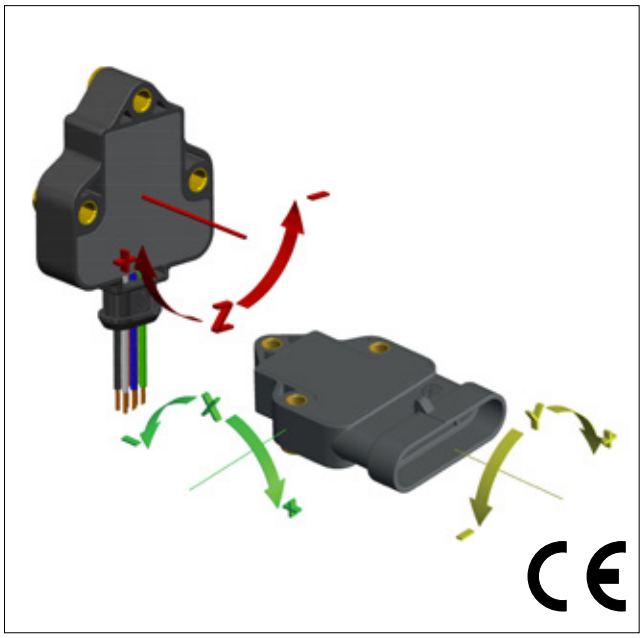
GEFRAN

CANopen GIB

digital output



Code 85200B **Edition** 03-2019

SUMMARY

1. INTRODUCTION 2
2. ELECTRICAL CONNECTIONS 3
3. NETWORK MANAGEMENT (NMT) 5
4. BAUD RATE 6
5. Node-ID 6
6. PARAMETER SETTINGS 6
7. RESTORE DEFAULT PARAMETERS 6
8. HEARTBEAT 7
9. ERROR HANDLING 7
10. SDO COMMUNICATION AND READ/WRITE COMMANDS 9
11. PDO COMMUNICATION AND ANGLE CALCULATION 10
12. CANOPEN FEATURES SUMMARY 21
13. DIGITAL FILTER SETTING 25
14. COMMUNICATION EXAMPLES 26

1. INTRODUCTION

The GIB AMP or cable output 2-axis (from ±15° to ±90°) or 1-axis (-180°...+180° = 0°...360°) inclination sensor with CANopen interface enables angle levelling and position detection in many applications. The sensor is based on state-of-the-art MEMS capacitive technology implementing the functions of a CAN BUS network slave device conforming to standard CANopen protocol proposed by C.i.A. (Can in Automation) and described in the document entitled “CANOpen Application Layer and Communication Profile DS 301 v. 4.2” and in other documents mentioned below. Other reference documents used are C.i.A. DS-410 Device Profile for Encoders V3.1 (not completely implemented) and C.i.A. DSP-305 Layer Setting Services and Protocol V1.1.1.

This document describes the standard CANopen implementations created.

This document describes the standard CANopen implementations created. It is addressed to CANopen network system integrators and to CANopen device designers who already know the content of the above-mentioned standards defined by C.i.A..

The details of aspects defined by CANopen do not pertain to the purpose of this text. For further information on the protocol you can also contact us via e-mail: at http://www.gefran.com/it/it/messages/new or contact the GEFRAN office nearest to you.

Definition and Shortening

CAN: Controller Area Network.
Describes a serial communication bus that implements the “physical” level 1 and the “data link” level 2 of the ISO/OSI reference model.

CAL: CAN Application Layer.
Describes implementation of the CAN in the level 7 “application” of the ISO/OSI reference model from which CANopen derives.

CMS: CAN Message Specification.
CAL service element. Defines the CAN Application Layer for the various industrial applications.

COB: Communication Object.
Unit of transport of data in a CAN network (a CAN message). A maximum of 2048 COBs may be present in a CAN network, each of which may transport from 0 to a maximum of 8 bytes.

COB-ID: COB Identifier.
Identifying element of a CAN message. The identifier determines the priority of a COB in case of multiple messages in the network.

D1 – D8: Data from 1 to 8.
Number of bytes in the data field of a CAN message.

DLC: Data Length code.
Number of data bytes transmitted in a single frame.

ISO: International Standard Organization.
International authority providing standards for various merchandise sectors.

NMT: Network Management.
CAL service element. Describes how to configure, initialize, manage errors in a CAN network.

PDO: Process Data Object.
Process data communication objects (with high priority).

RXSDO: Receive SDO.
SDO objects received from the remote device.

SDO: Service Data Object.
Service data communication objects (with low priority). The value of this data is contained in the “Objects Dictionary” of each device in the CAN network.

TXPDO: Transmit PDO.
PDO objects transmitted by the remote device.

TXSDO: Transmit SDO.
SDO objects transmitted by the remote device.

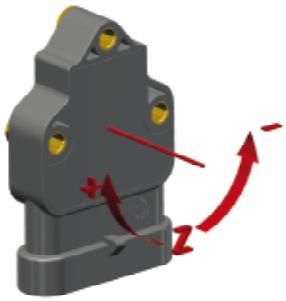
N.B.: The numbers followed by the suffix “h” represent a hexadecimal value, with suffix “b” a binary value, and with suffix “d” a decimal value. The value is decimal unless specified otherwise.

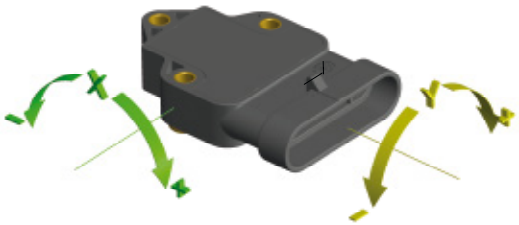
2. ELECTRICAL CONNECTIONS

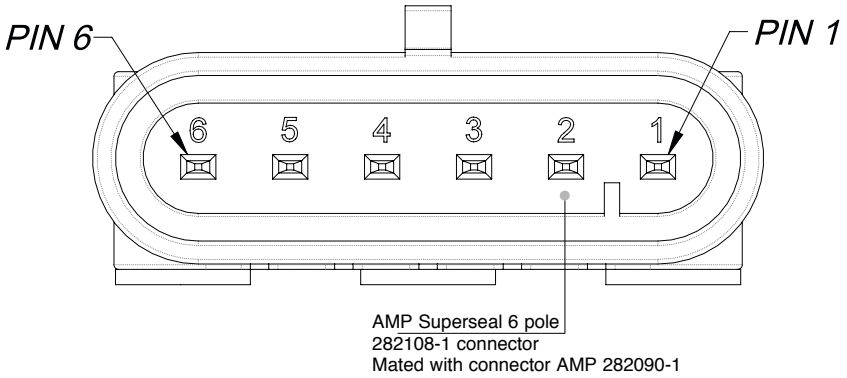
For the connections refer to the table below:

AMP Superseal 6 P 282108-1	Meaning
1	OV (GND)
2	+Vs (+10 … +36 Vdc)
3	NC
4	NC
5	CAN-L
6	CAN-H

Note: please make sure that the CANbus is terminated.
The impedance measured between CAN H and CAN L must be 60 ohm that means the cable must be connected to a 120 ohm resistor on each ends of the bus line. Internally the transducer is not terminated with the resistor of 120 ohm.
Do not confuse the signal lines of the CAN bus, otherwise communication with the transducer is impossible.



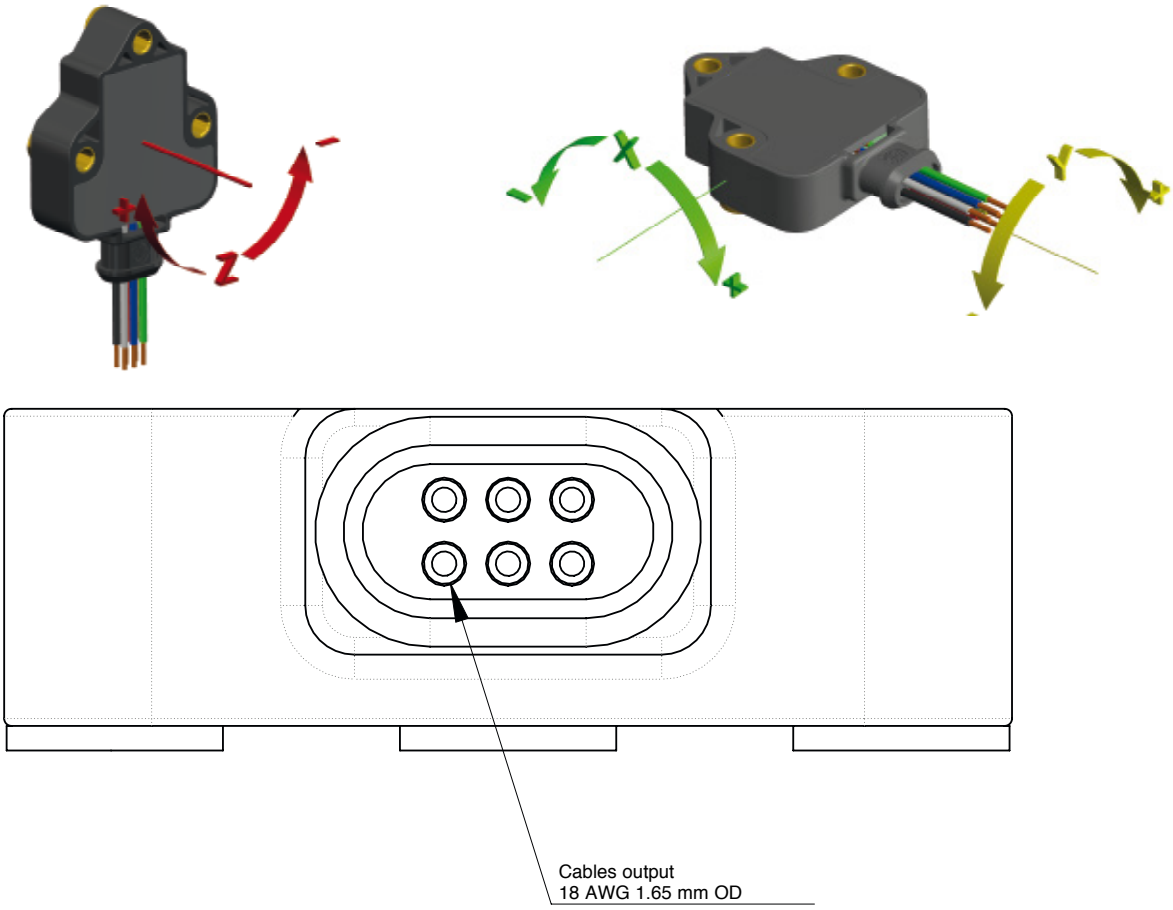




For the connections refer to the table below:

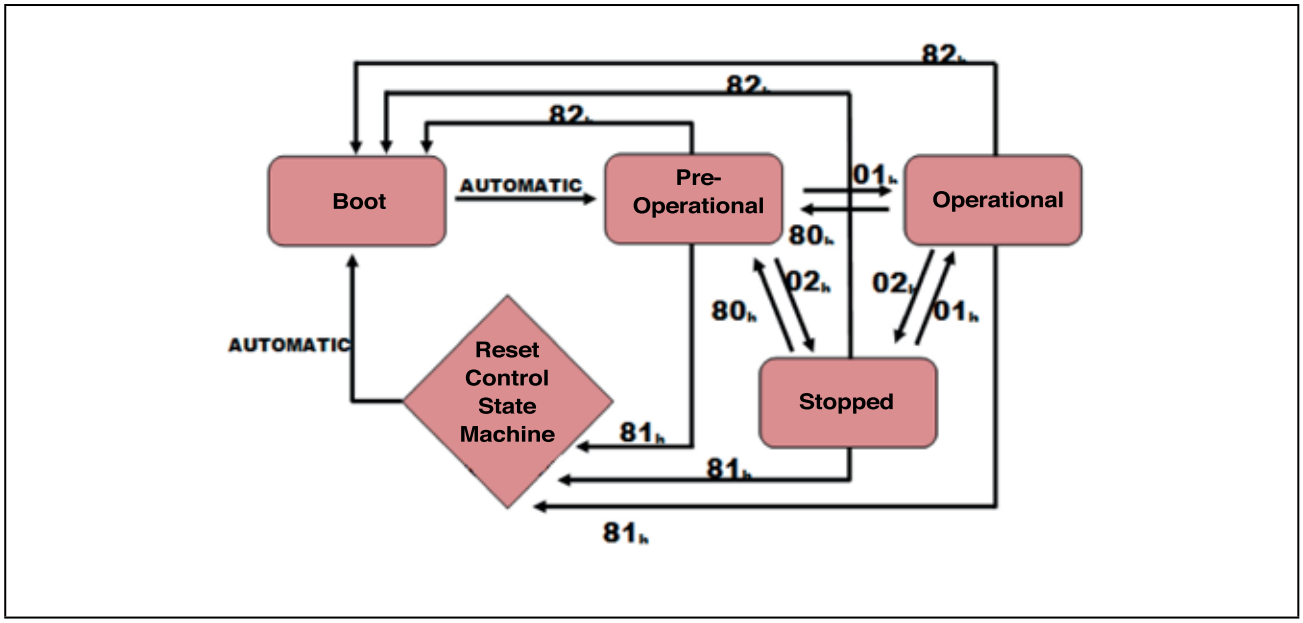
6 wires output 18 AWG 1.65mm OD	Meaning
BLACK	OV (GND)
RED	+Vs (+10 … +36 Vdc)
YELLOW	NC
GREEN	NC
BLUE	CAN-L
WHITE	CAN-H

Note: please make sure that the CANbus is terminated.
The impedance measured between CAN H and CAN L must be 60 ohm that means the cable must be connected to a 120 ohm resistor on each ends of the bus line. Internally the transducer is not terminated with the resistor of 120 ohm.
Do not confuse the signal lines of the CAN bus, otherwise communication with the transducer is impossible.



3. NETWORK MANAGEMENT (NMT)

The device supports CANopen network management functionality NMT Slave (Minimum Boot Up).



Every CANopen device contains an internal Network Management server that communicates with an external NMT master. One device in a network, generally the host, may act as the NMT master.

Through NMT messages, each CANopen device's network management server controls state changes within its built-in **Communication State Machine**.

This is independent from each node's operational state machine, which is device dependant and described in **Control State Machine**.

It is important to distinguish a CANopen device's operational state machine from its Communication State Machine.

CANopen sensors and I/O modules, for example, have completely different operational state machines than servo drives.

The "**Communication State Machine**" in all CANopen devices, however, is identical as specified by the DS301.

NMT messages have the highest priority. The 5 NMT messages that control the Communication State Machine each contain 2 data bytes that identify the node number and a command to that node's state machine.

Table 1 shows the 5 NMT messages supported, and **Table 2** shows the correct message construction for sending these messages.

Table 1

NMT Message	COB-ID	Data Byte 1	Data Bytes 2
Start Remote Node	0	01h	Node-ID*
Stop Remote Node	0	02h	Node-ID*
Pre-operational State	0	80h	Node-ID*
Reset Node	0	81h	Node-ID*
Reset Communication	0	82h	Node-ID*
* Node-ID = Drive address (from 1 to 7Fh)			

Table 2

Arbitration Field	Data Field								
COB-ID	RTR	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
000h	0	See table 1	See table 2	These bytes not sent					

4. BAUD RATE

Node ID can be configurable via SDO communication object 0x20F2 and 0x20F3 (see communication examples at the end of this document).

The default Baud rate is 250kbit/s.

Important Note:
Changing this parameter can disturb the network! Use this service only if one device is connected to the network!

5. Node-ID

Node ID can be configurable via SDO communication object 0x20F0 and 0x20F1 (see communication examples at the end of this document).

The default Node-ID is 7F.

Important Note:
Changing this parameter can disturb the network! Use this service only if one device is connected to the network!

6. PARAMETER SETTINGS

All object dictionary parameters (objects with marking PARA) can be saved in a special section of the internal EEPROM and secured by checksum calculation. The special LSS parameters (objects with marking LSS-PARA), also part of the object dictionary, will be also saved in a special section of the internal EEPROM and secured by checksum calculation. Due to the internal architecture of the microcontroller the parameter write cycles are limited to 100,000 cycles.

7. RESTORE DEFAULT PARAMETERS

All object dictionary parameters (objects with marking PARA) can be restored to factory default values via SDO communication (index 0x1011).

8. HEARTBEAT

The heartbeat mechanism for this device is established through cyclic transmission of the heartbeat message done by the heartbeat producer.
One or more devices in the network are aware of this heartbeat message. If the heartbeat cycle fails from the heartbeat producer the local application on the heartbeat consumer will be informed about that event.
The implementation of either guarding or heartbeat is mandatory.

The device supports **Heartbeat Producer** functionality. The producer heartbeat time is defined in object 0x1017.

Heartbeat Message

COB-ID	Byte	0
700+Nodo-ID	Content	NMT State

9. ERROR HANDLING

Principle

Emergency messages (EMCY) shall be triggered by internal errors on device and they are assigned the highest possible priority to ensure that they get access to the bus without delay (**EMCY Producer**). By default, the EMCY contains the error field with pre-defined error numbers and additional information.

Error Behavior (object 0x4000)

If a serious device failure is detected the object 0x4000 specifies, to which state the module shall be set:

0: pre-operational
1: no state change (default)
2: stopped

EMCY Message

The EMCY COB-ID is defined in object 0x1014. The EMCY message consists of 8 bytes. It contains an emergency error code, the contents of object 0x1001 and 5 byte of manufacturer specific error code. This device uses only the 1st byte as manufacturer specific error code.

Byte	Byte1 Byte2	Byte3	Byte4	Byte5	Byte6 Byte7 Byte8
Description	Emergency Error Code [1]	Error Register (object 0x1001[2])	Manufacturer specific error code (always 0x00)	Manufacturer specific error code (object 0x4001)	Manufacturer specific error code NOT IMPLEMENTED (always 0x00)
[1] **Error Code**		0x0000 Error Reset or no Error (Error Register = 0 0x1000 Generic error			
[2] **Always 0**					

Supported Manufacturer Specific Error Codes (object 0x4001)

Manufacturer Specific Error Code (bit field)	Description
0bxxxxxxx1[(a)]	Sensor Error TYPE GIB-Z-360 (e.g. angle under/above limits, self-test failure, MEMS IC communication error)
0bxxxxxxx1[(a)]	Sensor Error X-axis TYPE GIB-XY-0xx (e.g. angle under/above limits, self-test failure, MEMS IC communication error)
0bxxxxxx1x[(a)]	Sensor Error Y-axis TYPE GIB-XY-0xx (e.g. angle under/above limits, self-test failure, MEMS IC communication error)
0bxxx1xxxx	Program checksum error
0bxx1xxxxx	Flash limit reached - error
0bx1xxxxxx	LSS Parameter checksum error

(a) An angle error will be generated if the actual measured angle is under or above limits.
Example of limits for different versions are reported below:
GIB dual axis version ±10° Error limits are ± 11° (± 11° are also the FSO angles STOP)
GIB dual axis version ±15° Error limits are ± 16.5° (± 16.5° are also the FSO angles STOP)
GIB dual axis version ±20° Error limits are ± 22° (± 22° are also the FSO angles STOP)
GIB dual axis version ±30° Error limits are ± 33° (± 33° are also the FSO angles STOP)
GIB dual axis version ±45° Error limits are ± 49.5° (± 49.5° are also the FSO angles STOP)
GIB dual axis version ±60° Error limits are ± 66° (± 66° are also the FSO angles STOP)
GIB dual axis version ±90° Error limits are ± 87° (± 87° are also the FSO angles STOP)

10. SDO COMMUNICATION AND READ/WRITE COMMANDS

The device fulfils the SDO Server functionality.
With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.
Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data	Data	Data	Data

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index	Data	Data	Data	Data

Write Access, Data Transfer from Host to Slave

Each access to the object dictionary is checked by the slave for validity. Any write access to nonexistent objects, to read-only objects or with a non-corresponding data format are rejected and answered with a corresponding error message.

CMD *determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

The slave answers:

RES *Response of the slave:*

60 hex Data sent successfully

80 hex Error,

Read Access, Data Transfer from Slave to Host

Any read access to non-existing objects is answered with an error message.
CMD determines the direction of data transfer:

40 hex read access (in any case)

The slave answers:

RES *Response of the slave:*

42 hex Bytes used by node when replying to read command with 4 or less data

43 hex Bytes 5...8 contain a 32-bit value

4B hex Bytes 5, 6 contain a 16-bit value

4F hex Byte 5 contains an 8-bit value

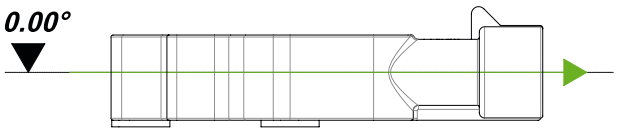
80 hex Error,

11. PDO COMMUNICATION and Angle Calculation

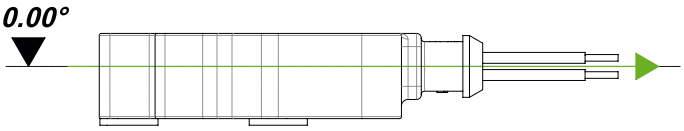
Transmit PDO #0 – Dual axis configuration X-Y (from ± 10° to ± 90°) model GIB-XY-xxx

This PDO transmits asynchronously the position value of the angle sensor. The Tx PDO #0 shall be transmitted cyclically, if the cyclic timer (object 0x1800.5) is programmed > 0. Values between 4ms and 65535 ms shall be selectable by parameter settings. The Tx PDO #0 will be transmitted by entering the "Operational" state.

Byte	Byte1	Byte2	Byte3	Byte4	Byte5 Byte6 Byte7 Byte8
Description	X Axis (object 0x6010) Low-Byte	X Axis (object 0x6010) High-Byte	Y Axis (object 0x6020) Low-Byte	Y Axis (object 0x6020) High-Byte	(0x00)
In the following figures an example of PDO mapping is reported in the case of Angle X = 0.00° and Angle Y = 0.00° (Node-ID = 7Fh and resolution ± 0.01°)					

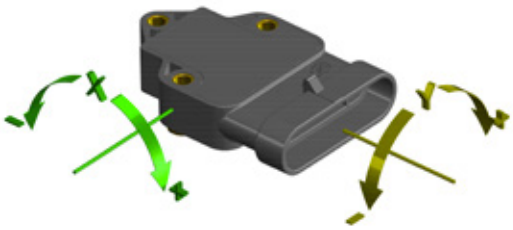


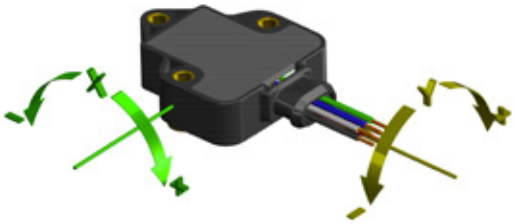
X= 0°



X= 0°

Angle X = 0.00°
Angle Y = 0.00°





ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
1FFh	00h	00h	00h	00h	00h	00h	00h	00h

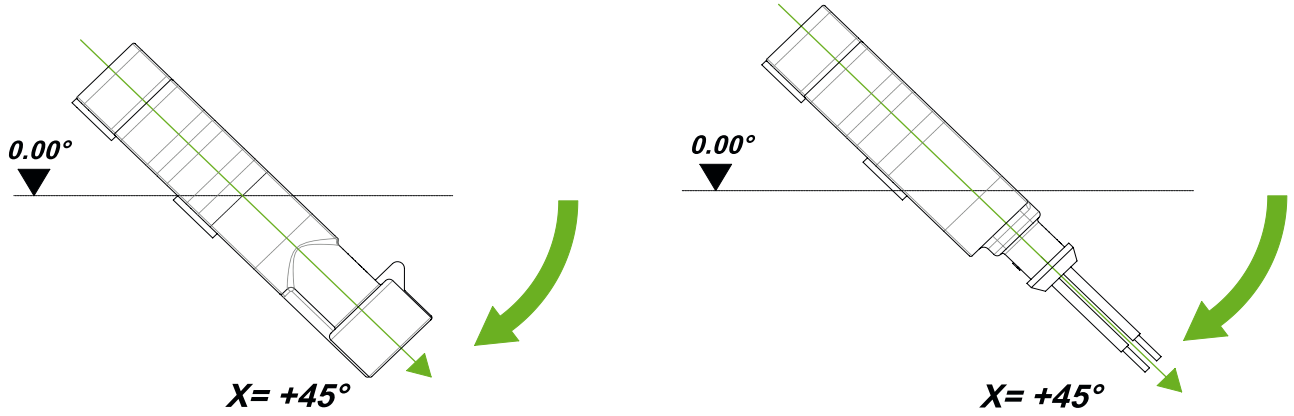
Angle X:

Byte 2 MSB (00h) = 00h
Byte 1 LSB (00h) = 00h
Angle X = 0000h to decimal 0d (resolution ± 0.01°) = 0.00°

Angle Y:

Byte 4 MSB (00h) = 00h
Byte 3 LSB (00h) = 00h
Angle Y = 0000h to decimal 0d (resolution ± 0.01°) = 0.00°

In the following figures an example of PDO mapping is reported in the case of **Angle X = + 45.00°** and **Angle Y = 0.00°** (Node-ID = 7Fh and resolution ± 0.01°)



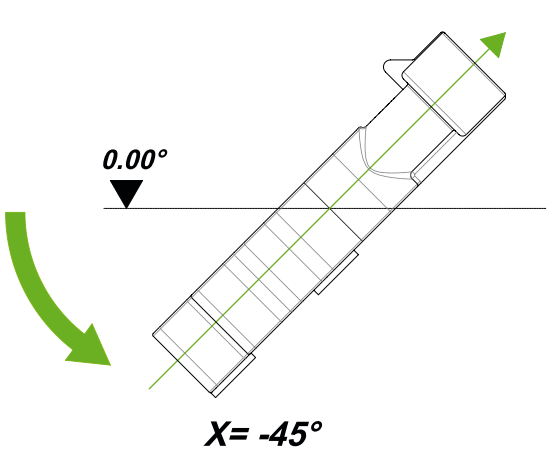
Angle X = +45.00°
Angle Y = +45.00°

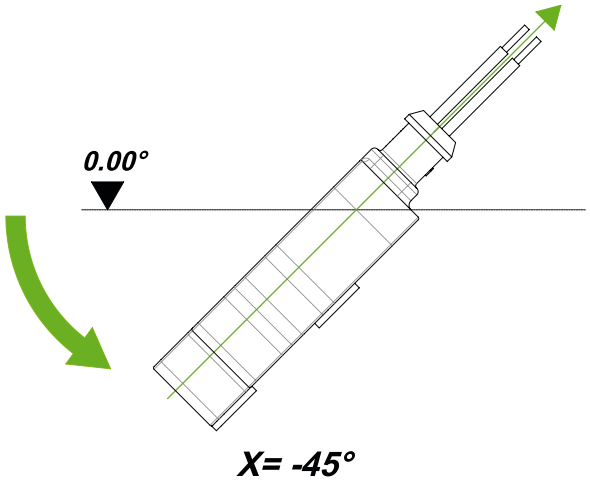
ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
1FFh	94h	11h	00h	00h	00h	00h	00h	00h

Angle X:
Byte 2 MSB (11h) = 11h
Byte 1 LSB (94h) = 94h
Angle X = 1194h to decimal 4500d (resolution ± 0.01°) = +45.00°

Angle Y:
Byte 4 MSB (00h) = 00h
Byte 3 LSB (00h) = 00h
Angle Y = 0000h to decimal 0d (resolution ± 0.01°) = 0.00°

In the following figures an example of PDO mapping is reported in the case of **Angle X = - 45.00°** and **Angle Y = + 0.00°** (Node-ID = 7Fh and resolution ± 0.01°).



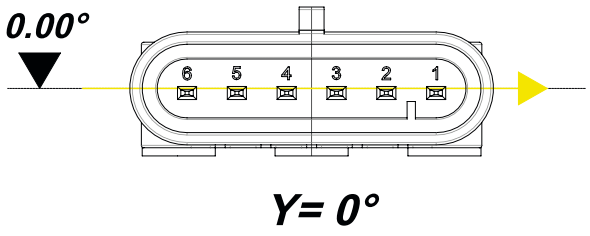


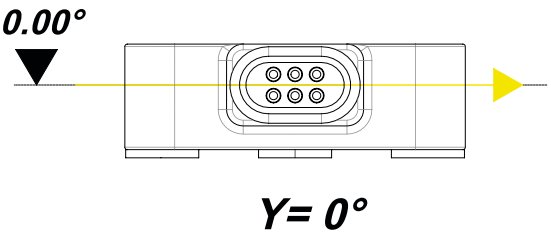
Angle X = -45.00°
Angle Y = 0.00°

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
1FFh	6Bh	EEh	00h	00h	00h	00h	00h	00h

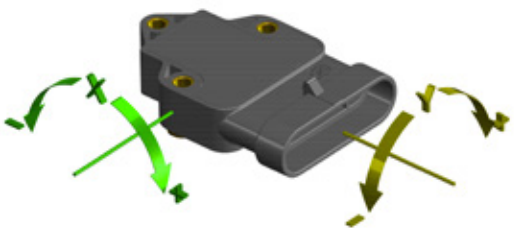
Angle X:
Byte 2 MSB (EEh) = EEh
Byte 1 LSB (6Bh) = 6Bh
Angle X = EE6Bh to decimal 61035d
If the Angle X in decimal is greater than 32768, the Angle X is NEGATIVE and it must be computed as below (resolution ± 0.01°)
Angle X = EE6Bh to decimal 61035d
Angle X = Angle X (in decimal) - 65535d = 61035d - 65535d = -4500d (resolution ± 0.01°) = -45.00°

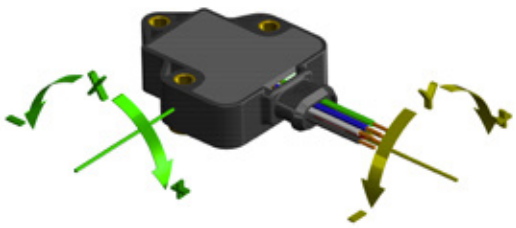
Angle Y:
Byte 4 MSB (00h) = 00h
Byte 3 LSB (00h) = 00h
Angle Y = 0000h to decimal 0d (resolution ± 0.01°) = 0.00°
In the following figures an example of PDO mapping is reported in the case of **Angle X = 0.00**° and **Angle Y = 0.00°** (Node-ID = 7Fh and resolution ± 0.01°)





Angle X = -0.00°
Angle Y = 0.00°

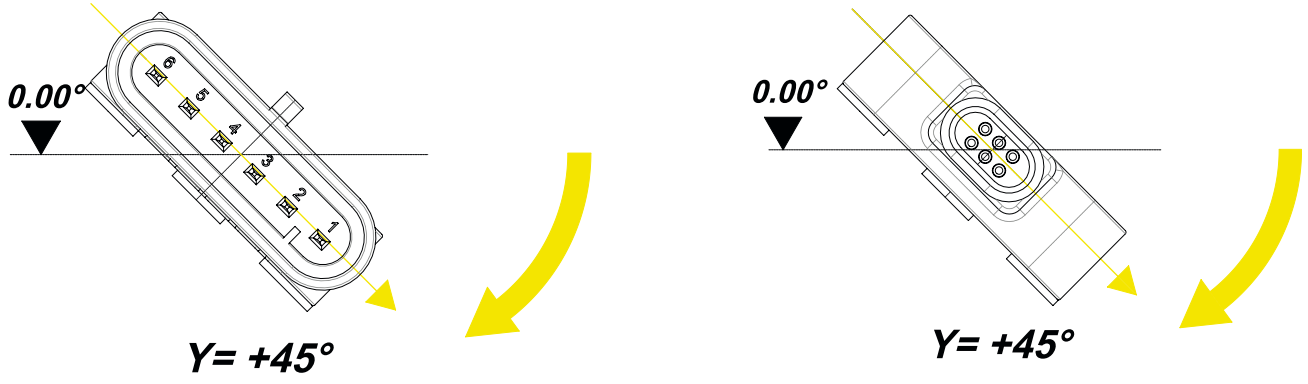




ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
1FFh	00h	00h	00h	00h	00h	00h	00h	00h

Angle X:
Byte 2 MSB (00h) = 00h
Byte 1 LSB (00h) = 00h
Angle X = 0000h to decimal 0d (resolution ± 0.01°) = 0.00°

Angle Y:
Byte 4 MSB (00h) = 00h
Byte 3 LSB (00h) = 00h
Angle Y = 0000h to decimal 0d (resolution ± 0.01°) = 0.00°
In the following figures an example of PDO mapping is reported in the case of **Angle X** = 0.00° and **Angle Y** = +45.00° (Node-ID = 7Fh and resolution ± 0.01°)



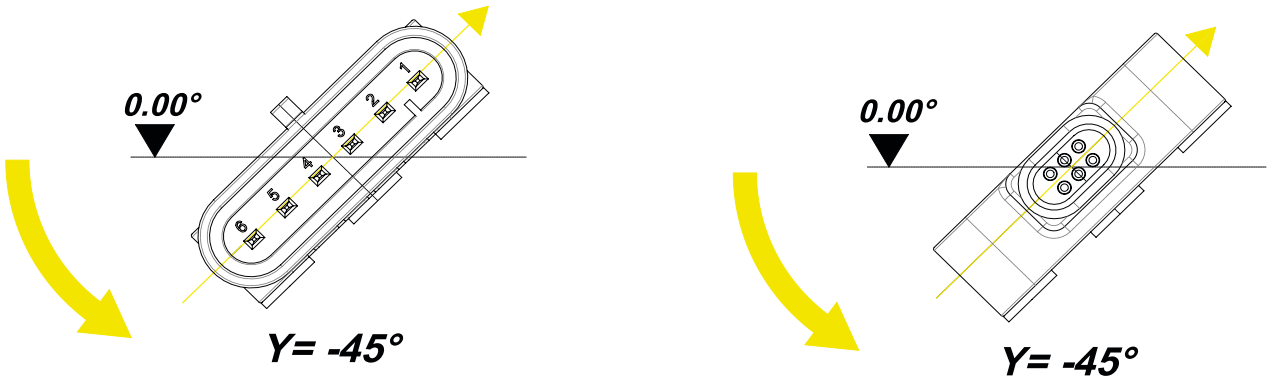
Angle X = -0.00°
Angle Y = +45.00°

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
1FFh	00h	00h	94h	11h	00h	00h	00h	00h

Angle X:
Byte 2 MSB (00h) = 00h
Byte 1 LSB (00h) = 00h
Angle X = 0000h to decimal 0d (resolution ± 0.01°) = 0.00°

Angle Y:
Byte 4 MSB (11h) = 11h
Byte 3 LSB (94h) = 94h
Angle Y = 1194h to decimal 4500d (resolution ± 0.01°) = +45.00°
In the following figures an example of PDO mapping is reported in the case of **Angle X** = 0.00° and **Angle Y** = +45.00° (Node-ID = 7Fh and resolution ± 0.01°)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
1FFh	00h	00h	6Bh	EEh	00h	00h	00h	00h



Angle X = -0.00°
Angle Y = -45.00°

Angle X:
Byte 2 MSB (00h) = 00h
Byte 1 LSB (00h) = 00h
Angle X = 0000h to decimal 0d (resolution ± 0.01°) = 0.00°

Angle Y:
Byte 4 MSB (EEh) = EEh
Byte 3 LSB (6Bh) = 6Bh
Angle Y = EE6Bh to decimal 61035d
If the Angle Y in decimal is greater than 32768, the Angle Y is NEGATIVE and it must be computed as below (resolution ± 0.01°)
Angle Y = EE6Bh to decimal 61035d
Angle Y = Angle Y (in decimal) - 65535d = 61035d - 65535d = -4500d (resolution ± 0.01°) = -45.00°

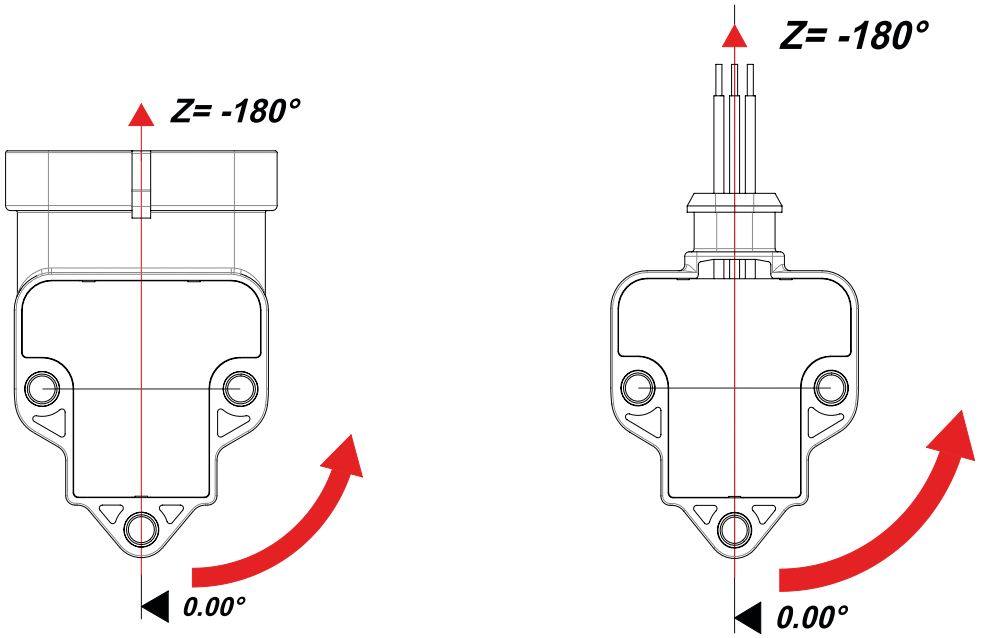
Transmit PDO #0 – Single axis configuration Z (-180°...+180°) model GIB-Z-360

This PDO transmits synchronously the position value of the inclination sensor. The Tx PDO #0 shall be transmitted cyclically,

if the cyclic timer (object 0x1800.5) is programmed > 0. Values between 4ms and 65535 ms shall be selectable by parameter settings. The Tx PDO #0 will be transmitted by entering the "Operational" state.

Byte	Byte1	Byte2	Byte3 Byte4 Byte5 Byte6 Byte7 Byte8
Description	Z Axis (object 0x6010) Low-Byte	Z Axis (object 0x6010) High-Byte	0x00

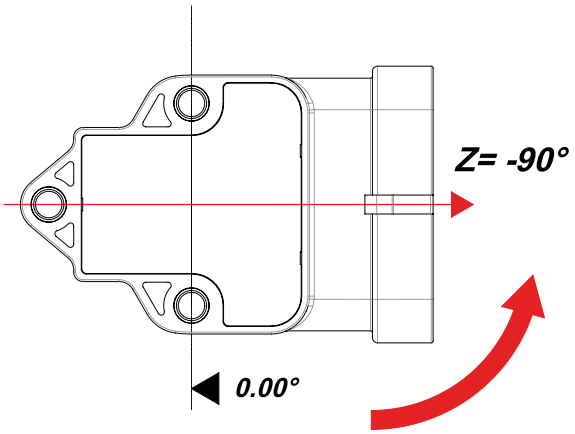
In the following figures an example of PDO mapping is reported in the case of **Angle Z = -180.0°** (in 0...360° configuration the equivalent angle is 0.00°). The Node-ID = 7Fh and resolution ± 0.01°.

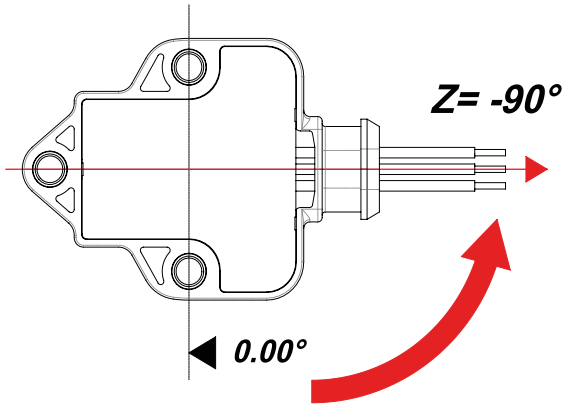


Angle Z = -180.00°

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
1FFh	AFh	B9h	00h	00h	00h	00h	00h	00h

Angle Z:
Byte 2 MSB (B9h) = B9h
Byte 1 LSB (AFh) = AFh
Angle Z = B9AFh to decimal 47535d
If the Angle Z in decimal is greater than 32768, the Angle Z is NEGATIVE and it must be computed as below (resolution ± 0.01°)
Angle Z = B9AFh to decimal 47535d
Angle Z = Angle Z (in decimal) - 65535d = 47535d - 65535d = -18000d (resolution ± 0.01°) = -180.00°
In the following figures an example of PDO mapping is reported in the case of **Angle Z = -90.0°** (in 0...360° configuration the equivalent angle is +90.00°). The Node-ID = 7Fh and resolution ± 0.01°.

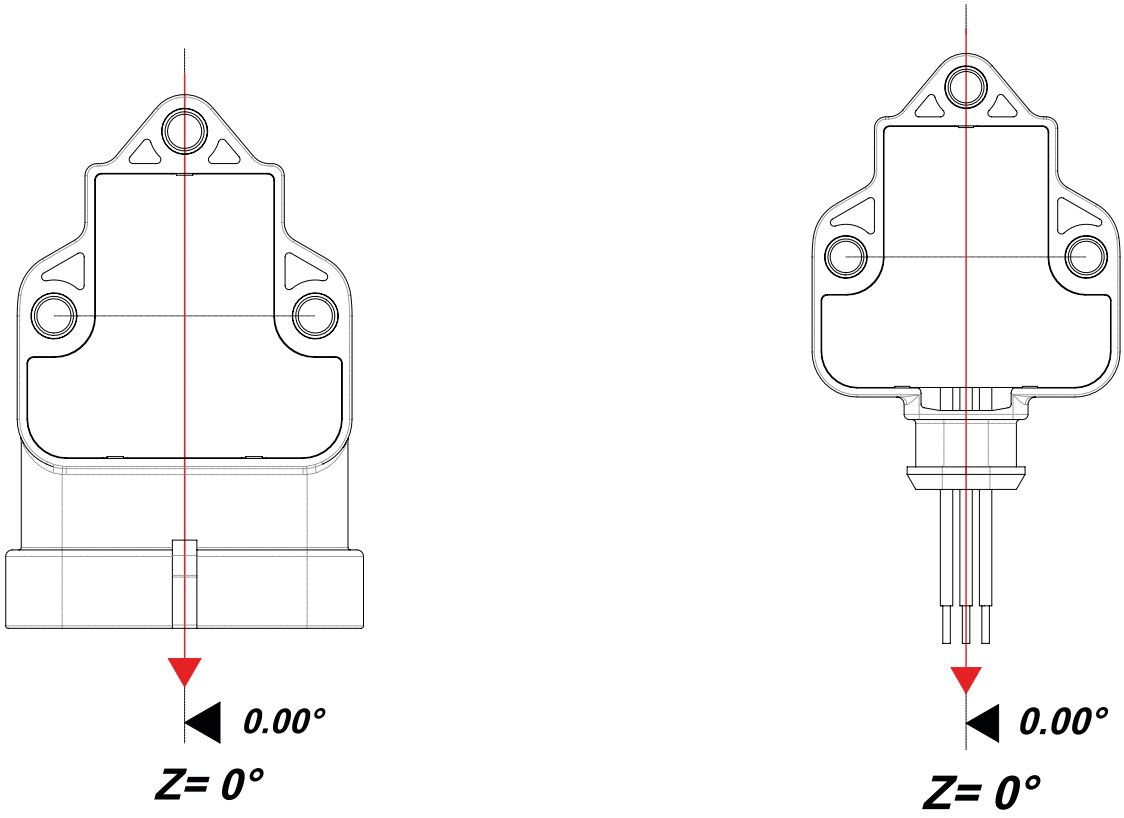




Angle Z = -90.00°

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
1FFh	D7h	DCh	00h	00h	00h	00h	00h	00h

Angle Z:
Byte 2 MSB (DCh) = DCh
Byte 1 LSB (D7h) = D7h
Angle Z = DCD7h to decimal 56535d
If the Angle Z in decimal is greater than 32768, the Angle Z is NEGATIVE and it must be computed as below (resolution ± 0.01°)
Angle Z = B9AFh to decimal 56535d
Angle Z = Angle Z (in decimal) - 65535d = 56535d - 65535d = -9000d (resolution ± 0.01°) = -90.00°
In the following figures an example of PDO mapping is reported in the case of **Angle Z = 0.0°** (in 0...360° configuration the equivalent angle is +180.00°). The Node-ID = 7Fh and resolution ± 0.01°.



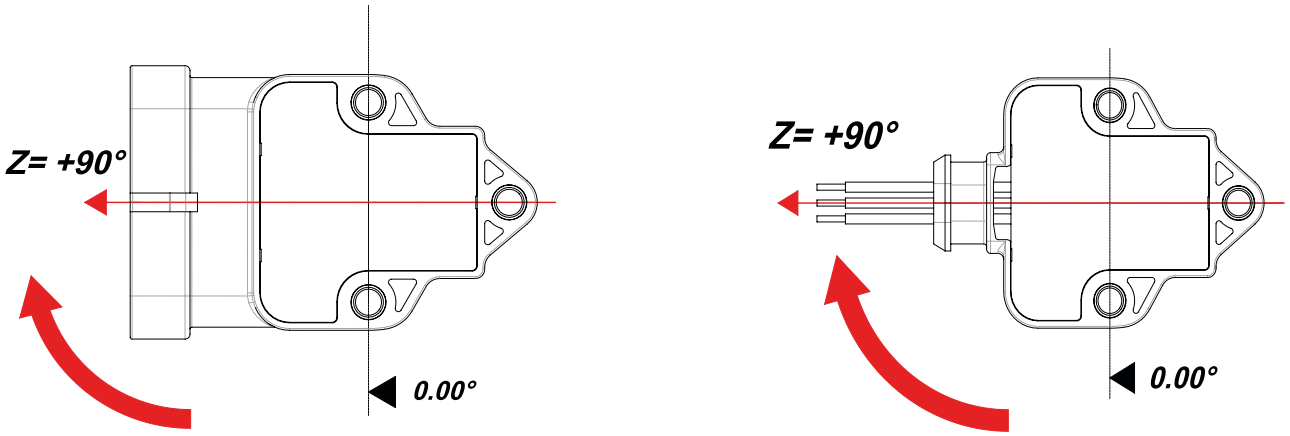
Angle Z = 0.00°

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
1FFh	00h	00h	00h	00h	00h	00h	00h	00h

Angle Z:
Byte 2 MSB (00h) = 00h
Byte 1 LSB (00h) = 00h
Angle Z = 0000h to decimal 0d = 0.00°

In the following page an example of PDO mapping is reported in the case of **Angle Z = + 90.00°** (in 0...360° configuration the equivalent angle is +270.00°). The Node-ID = 7Fh and resolution ± 0.01°.

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
1FFh	28h	23h	00h	00h	00h	00h	00h	00h



Angle Z = +90.00°

Angle Z:
Byte 2 MSB (23h) = 23h
Byte 1 LSB (28h) = 28h
Angle Z = 2328h to decimal 9000d (resolution 0.01°) = +90.00 °

Communication Profile
The parameters which are critical for communication are determined in the Communication profile.
This area is common for all CANopen devices.

12. CANOPEN FEATURES SUMMARY

Index	Sub Index	Name	Type	Access	Default value	Comments
1000h		Device Profile	Unsigned 32	Ro	0x0008019A	Profile 410: Device profile for inclinometer (not fully implemented).
1001h		Error Register	Unsigned 8	Ro	0x00	Always ZERO
1005h		COB-ID SYNC	Unsigned 32	Rw	0x00000080	
1008h		Manufacturer Device Name	String	Const	“GIB”	Refer to GEFRAN products catalogue: GIB: Dual-axis or single-axis inclinometer sensor
1009h		Manufacturer Hardware Version	String	Const	“1.00”	
100Ah		Manufacturer Software Version	String	Const	“1.10”	
1010h	0	Number of Entries	Unsigned 8	Ro	1	“save” (0x65766173) to store all parameters (objects with marking PARA)
	1	Save all Parameters	Unsigned 32	Wo		
1011h	0	Restore Default Parameters	Unsigned 8	Ro		"load" (0x64616F6C) to restore all parameters (objects with marking PARA and LSS-PARA).
	1	Restore all Parameters	Unsigned 32	Rw		
1014h	0	Emergency ID	Unsigned 32	Ro	0x80 + Nodo-ID	
1017h	0	Producer Time / Heart Beat	Unsigned 16	Rw	0	Min= 0 & Max=65535 with unit = 1ms If 0: NOT USED From 1 to 19 NOT ACCEPTED From 20 to 65535 ACCEPTED
1018h	0	Identity Object	Unsigned 8	Ro	4	Refer to “Gefran Product Overview CANopen” Gefran Vendor ID:0x0000093
	1	Vendor ID	Unsigned 32	Ro	0x0000093	
	2	Product Code	Unsigned 32	Ro	0x0000064	
	3	Revision Number	Unsigned 32	Ro	0x0000001	
	4	Serial Number	Unsigned 32	Ro	0x0000000	
1200h		SDO Server Parameter				
	0	Number of Entries	Unsigned 8	Ro	2	
	1	COB-ID Client to Server (Rx)	Unsigned 32	Ro	0x600+ Node-ID	
	2	COB-ID Server to Server (Tx)	Unsigned 32	Ro	0x580+ Node-ID	
1800h	0	1st Transmit PDO Parameter	Unsigned 8	Ro		
	1	COB-ID	Unsigned 32	Ro	180h + Node-ID	
	2	Transmission Type	Unsigned 8	Rw	254	Asynchronous transmission.
	3	Inhibit Time	Unsigned 16	Rw	4	Min= 4 & Max=65535 with unit = 1ms
	4	Reserved	Unsigned 8	Ro		
	5	Event timer - PARA	Unsigned 16	Rw	100	Min= 4 & Max=65535 with unit = 1ms
1A00h		Tx PDO [X] 0 Mapping Parameter				The inclination of longitudinal axis (long; X) is indicated in Idx 6010 in the case of dual axis sensor (±10°…±90°) The inclination of transverse axis (tran; Y) is indicated in Idx 6020 in the case of dual axis sensor (±10°…±90°) The inclination of Z axis is indicated in Idx 6010 in the case of single axis sensor (±180°)
	0	Number of entries	Unsigned 8	Ro	2	
	1	1st Mapped Object	Unsigned 32	Ro	0x60100010	
	2	2nd Mapped Object	Unsigned 32	Ro	0x60200020	

Ro = the parameter can be read only
Rw = the parameter can be read and also written
Wo = the parameter can be written only

Manufacturer Specific Profile Objects

In this section you will find the manufacturer specific profile indices for the transducer.

"Setting the Node-ID"

Index	Sub Index	Name	Type	Access	Default value	Comments
20F0h	0	Setting of the Node ID	Unsigned 8	Rw	0x7F (=127d)	The node ID used to access thesensor in the CANopen network
20F1h	0	Setting of the Node ID	Unsigned 8	Rw	0x7F (=127d)	The node ID used to access thesensor in the CANopen network

A change of the Node ID is only accepted if the entries 20F0 and 20F1 contain the same changed value.
Values below 1 / above 127 are not accepted; the existing setting remains valid. After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

"Setting the Baud Rate"

Index	Sub Index	Name	Type	Access	Default value	Comments
20F2h	0	Setting of the Baud rate	Unsigned 8	Rw	0x03 (250 kBaud)	Baud rate of the CAN network 0 = 1000 kBaud 1 = 800 kBaud 2 = 500 kBaud 3 = 250 kBaud (default) 4 = 125 kBaud 5 = 100 kBaud 6 = 50 kBaud 7 = 20 kBaud
20F3h	0	Setting of the Baud rate	Unsigned 8	Rw	0x03 (250 kBaud)	Baud rate of the CAN network 0 = 1000 kBaud 1 = 800 kBaud 2 = 500 kBaud 3 = 250 kBaud (default) 4 = 125 kBaud 5 = 100 kBaud 6 = 50 kBaud 7 = 20 kBaud

A change of the Baude rate is only accepted if the entries 20F2 and 20F3 contain the same changed value.
Values above 7 are not accepted; the existing setting remains valid.
After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Manufacturer Specific Profile Objects

In this section you will find the manufacturer specific profile indices for the transducer

Index	Sub Index	Name	Type	Access	Default value	Comments
4000h		Error Behavior - PARA	Unsigned 8	Rw	1	0: Pre-operational 1: no state change 2: stopped Min=0 & Max=255
4001h		Error Code	Unsigned 8	Ro	0	0: no error Min=0 & Max=255
5000h		Automatic NMT Start after Power-On - PARA	Unsigned 8	Rw	0	0:not activated 1: activated Min=0 & Max=1

Ro = the parameter can be read only
Rw = the parameter can be read and also written

Setting the Digital Filter

Index	Sous-Index	Nom	Type	Accès	Valeur par défaut	Remarques
2001h	0	Filter Setting - PARA	Unsigned 8	Rw	0	Filter = 0 Slow - Filter = 1 Medium - Filer = 2 Fast See Par. 14 and examples at the end of this manual

A change of the Filter Setting is only accepted after a STORE command (see Store Parameters setting via SDO 0x1010 Sub 1 and examples of Filter setting at the end of this manual)

Manufacturer Specific Profile Objects (according to CiA DS-410)

In this section you will find the manufacturer specific profile indices for the transducer.

Index	Sub Index	Name	Type	Access	Default value	Comments
6000h	0	Resolution	Unsigned 16	Rw	0x32 (50d)	Display resolution of the inclination for both axes(1) 10d = Inclination is indicated as signed int in 0.01° 50d = Inclination is indicated as signed int in 0.05° 100d = Inclination is indicated as signed int in 0.1° 500d = Inclination is indicated as signed int in 0.5° 1000d = Inclination is indicated as signed int in 1.0° **Note**: If the display resolution is changed all offset values or zero point values which may have been entered are deleted. Therefore the sensor must be set before it is aligned! (1) A change of the display resolution in Idx 6000 is only accepted, if the scaling in Idx 6011 and Idx 6021 is activated.
6010h	0	Slope Longitudinal	Signed 16	Ro	-	Inclination of the longitudinal axis X (long; X)
6011h	0	Slope Longitudinal Operating Parameter	Unsigned 8	Rw	0b000000xx	Inverting the sign 0b 0000 00x0 deactivated 0b 0000 00x1 activated Scaling of the measured value 0b 0000 000x deactivated 0b 0000 001x activated(1) Value output: Slope longitudinal = measured value in dependence of Resolution (Index 6000) + Slope Longitudinal Offset + Differential Slope Longitudinal Offset (1) A change of the display resolution in Idx 6000 is only accepted, if the scaling in Idx 6011 and Idx 6021 is activated.
6012h	0	Slope Longitudinal Preset Value	Signed 16	Rw	0x0000	Corrects the measured sensor value. The displayed value Slop Longitudinal is set to the entered value. The offset is indicated in the index 0x6013
6013h	0	Slope Longitudinal Offset	Signed 16	Ro	0x0000	Offset value calculated from the following objects: Slope Longitudinal Offset = Slope Longitudinal Preset Value t_{acc} – measured value t_{acc} (t_{acc} : istant when the Slope Longitudinal Preset Value is set)
6014h	0	Slope Longitudinal Differential Offset	Signed 16	Rw	0x0000	Shifts the displayed value by the entered value irrespective of "Slope Longitudinal Preset Value".
6020h	0	Slope Lateral	Signed 16	Ro	-	Inclination of the Lateral axis X (long; X)

Index	Sub Index	Name	Type	Access	Default value	Comments
6021h	0	Slope Lateral Operating Parameter	Unsigned 8	Rw	0b000000xx	Inverting the sign 0b 0000 00x0 deactivated 0b 0000 00x1 activated Scaling of the measured value 0b 0000 000x deactivated 0b 0000 001x activated(1) Value output: Slope Lateral = measured value in dependence of Resolution (Index 6000) + Slope Lateral Offset + Differential Slope Lateral Offset (1) A change of the display resolution in Idx 6000 is only accepted, if the scaling in Idx 6011 and Idx 6021 is activated.
6022h	0	Slope Lateral Preset Value	Signed 16	Rw	0x0000	Corrects the measured sensor value. The displayed value Slop Lateral is set to the entered value. The offset is indicated in the index 0x6023
6023h	0	Slope Lateral Offset	Signed 16	Ro	0x0000	Offset value calculated from the following objects: Slope Lateral Offset = Slope Lateral Preset Value t_{acc} – measured value t_{acc} (t_{acc} : istant when the Slope Lateral Preset Value is set)
6024h	0	Slope Lateral Differential Offset	Signed 16	Rw	0x0000	Shifts the displayed value by the entered value irrespective of "Slope Laterall Preset Value".

Ro = the parameter can be read only
Rw = the parameter can be read and also written
Wo = the parameter can be written only

13. DIGITAL FILTER SETTING

The inclination sensor offers the possibility to suppress the influence of external disturbing vibrations.
The internal lowpass digital filters (8th order) are programmable in 3 steps (more steps can be obtained on request and they can be adjusted for any kind of application).
The sensor has digital filters that can be selected according to Table 2 below.
The filter selection is configurable via SDO communication object 0x2001 Sub 0 (see the Manufacturer Specific Profile Objects and communication examples at the end of this document).

Filter Selection (via SDO oggetto 0x2106 Sub 6)	Filter code	Application
Slow	Filter 0	Static inclination measurement with high damping to vibration
Medium	Filter 1	Inclination measurement in applications that requires a certain dynamism, without overshoot at angle changes with good damping
Fast	Filter 2	General application with mediumhigh dynamic

Table – Filter setting

Example 1) How to change the Baud Rate Setting from 250 kbaud (current setting) to 500 kbaud

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data			

CMD *determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

RES *Response of the slave:*

60 hex Data sent successfully

80 hex Error,

A change of the Baud rate is only accepted if the entries 0x20F2 and 0x20F3 contain the same changed value.

With the aim to change the baud rate from 250kBaud (0x03) to 500 kBaud (0x02) write a first SDO (in the example the Node-ID = 0x7F).

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Fh	F2h	20h	00h	02h	00h	00h	00h

A change of the Baud rate is only accepted if the entries 0x20F2 and 0x20F3 contain the same changed value.

With the aim to change the baud rate from 250kBaud (0x03) to 500 kBaud (0x02) write a second SDO (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Fh	F3h	20h	00h	02h	00h	00h	00h

Object:

20F2h	0	Setting of the Baude rate	Unsigned 8	Rw	0x03 (250 kBaud)	Baud rate of the CAN network 0 = 1000 kBaud 1 = 800 kBaud 2 = 500 kBaud 3 = 250 kBaud (default) 4 = 125 kBaud 5 = 100 kBaud 6 = 50 kBaud 7 = 20 kBaud

20F3h	0	Setting of the Baude rate	Unsigned 8	Rw	0x03 (250 kBaud)	Baud rate of the CAN network 0 = 1000 kBaud 1 = 800 kBaud 2 = 500 kBaud 3 = 250 kBaud (default) 4 = 125 kBaud 5 = 100 kBaud 6 = 50 kBaud 7 = 20 kBaud

The supported baudrate are listed in the following table:

Byte5	Baudrate
07h	20 kBaud
06h	50 kBaud
05h	100 kBaud
04h	125 kBaud
03h	250 kBbaud
02h	500 kBbaud
01h	800 kBbaud
00h	1000 kBbaud

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	F2h	20h	00h	00h	00h	00h	00h

and

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	F2h	20h	00h	00h	00h	00h	00h

IMPORTANT NOTE:

A change of the Baud rate is only accepted if the entries 0x20F2 and 0x20F3 contain the same changed value. Values above 7 are not accepted; the existing setting remains valid.

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 2) How to change the ID-Node from 0x7Fh (127d)(current setting) to 0x06h (6d)

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data			

CMD *determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

RES *Response of the slave:*

60 hex Data sent successfully

80 hex Error,

A change of the Node ID is only accepted if the entries 0x20F0 and 0x20F1 contain the same changed value.

With the aim to change the Node ID from 127 (0x7F) to 6 (0x06) write a first SDO (in the example the Node-ID =0x7F).

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Fh	F0h	20h	00h	06h	00h	00h	00h

A change of the Node ID is only accepted if the entries 0x20F0 and 0x20F1 contain the same changed value.

With the aim to change the Node ID from 127 (0x7F) to 6 (0x06) write a second SDO (in the example the Node-ID =0x7F).

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Fh	F1h	20h	00h	06h	00h	00h	00h

Object:

20F0h	0	Setting of the Node ID	Unsigned 8	Rw	0x7F (= 127d)	The Node ID used to access the sensor in the CANopen
20F1h	0	Setting of the Node ID	Unsigned 8	Rw	0x7F (= 127d)	The Node ID used to access the sensor in the CANopen

The supported Node-ID are 0x01 to 0x7F

The answers after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	F0h	20h	00h	00h	00h	00h	00h

and

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	F1h	20h	00h	00h	00h	00h	00h

IMPORTANT NOTE:

A change of the Node ID is only accepted if the entries 0x20F0 and 0x20F1 contain the same changed value. Values below 1 / above 127 are not accepted; the existing setting remains valid. After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 3) How to change the PDO rate (time interval) from 100 ms (current setting) to 20 ms

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data	Data		

CMD *determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

RES *Response of the slave:*

60 hex Data sent successfully

80 hex Error,

With the aim to change the PDO rate from 100 ms (0x64) to 20 ms (0x14) write (in the example the Node-ID=0x7F).

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Bh	00h	18h	05h	14h	00h	00h	00h

Object:

1800h	0	1st Transmit PDO Parameter	Unsigned 8	Ro		
	1	COB-ID	Unsigned 32	Ro	180h+ Node-ID	
	2	Transmission Type	Unsigned 8	Rw	254	Asynchronous transmission.
	3	Inhibit Time	Unsigned 16	Ro	0	Min=0 & Max=65535 with unit=1ms
	4	Reserved	//	//		
	5	Timer	Unsigned 16	Rw	100 (64h)	Min=0 & Max=65535 with unit=1ms

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	00h	18h	05h	00h	00h	00h	00h

With the aim to save functionality write the “save” command as below:

Write (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	23h	10h	10h	01h	73h	61h	76h	65h

Note: save command is given by sending the code:

73h	61h	76h	65h

Where:

73h = ASCII code **“s”**

61h = ASCII code **“a”**

76h = ASCII code **“v”**

65h = ASCII code **“e”**

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	10h	10h	01h	00h	00h	00h	00h

IMPORTANT NOTE:

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 4) How to activate an automatic NMT Start after Power ON (the PDO will be send automatically after power ON)

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data			

CMD *determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

RES *Response of the slave:*

60 hex Data sent successfully

80 hex Error,

With the aim to activate an automatic NMT Start after power ON write (in the example the Node-ID = 0x7F).

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Fh	00h	50h	00h	01h	00h	00h	00h

Object:

5000h	0	Automatic NMT Start after Power-On - PARA	Unsigned 8	Rw	0	0: not activated 1: activated Min=0 & Max=1

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	00h	50h	00h	00h	00h	00h	00h

With the aim to save functionality write the “save” command as below:

Write (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	23h	10h	10h	01h	73h	61h	76h	65h

Note: save command is given by sending the code:

73h	61h	76h	65h

Where:

73h = ASCII code **“s”**

61h = ASCII code **“a”**

76h = ASCII code **“v”**

65h = ASCII code **“e”**

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	10h	10h	01h	00h	00h	00h	00h

IMPORTANT NOTE:

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 5) How to Preset the angle X to 0.00° (in case of dual axis ±10°…±90°).

The values "Preset Value" (Idx 60x2) and "Differential Offset" (Idx 60x4) affects the display of the longitudinal and lateral axis. The value entered in "Preset Value" immediately corrects the measured value of the sensor cell at the instant tacc.

A typical application is the compensation of display errors due to mounting (e.g. sensor zeroing). The sensor must first be brought to a defined position. The value "Differential Offset" shifts the displayed value of the sensor by the entered value. A set "Preset Value" does not affect shifting.

Note that the resolution parameter must be set before aligning the sensor (resolution, Idx 6000)!

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data	Data		

CMD *determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

RES *Response of the slave:*

60 hex Data sent successfully

80 hex Error,

With the aim to preset the X angle to 0.00° write (in the example the Node-ID = 0x7F).

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Bh	12h	60h	00h	00h	00h	00h	00h

Object:

6012h	0	Slop Longitudinal Preset Value	Signed 16	Rw	-	Corrects the measured sensor value. The displayed value Slop Longitudinal is set to the entered value. The offset is indicated in the index 0x6013

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	12h	60h	00h	00h	00h	00h	00h

With the aim to save functionality write the “save” command as below:

Write (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	23h	10h	10h	01h	73h	61h	76h	65h

Note: save command is given by sending the code:

73h	61h	76h	65h

Where:

73h = ASCII code **“s”**

61h = ASCII code **“a”**

76h = ASCII code **“v”**

65h = ASCII code **“e”**

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	10h	10h	01h	00h	00h	00h	00h

IMPORTANT NOTE:

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 6) How to set the angle Y to 0.00° (in case of dual axis ±10°…±90°).

The values "Preset Value" (Idx 60x2) and "Differential Offset" (Idx 60x4) affects the display of the longitudinal and lateral axis. The value entered in "Preset Value" immediately corrects the measured value of the sensor cell at the instant tacc.

A typical application is the compensation of display errors due to mounting (e.g. sensor zeroing). The sensor must first be brought to a defined position. The value "Differential Offset" shifts the displayed value of the sensor by the entered value. A set "Preset Value" does not affect shifting.

⚠ **Note that the resolution parameter must be set before aligning the sensor (resolution, Idx 6000)!**

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data	Data		

***CMD** determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

RES Response of the slave:

60 hex Data sent successfully

80 hex Error,

With the aim to preset the Y angle to 0.00° write (in the example the Node-ID = 0x7F).

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Bh	22h	60h	00h	00h	00h	00h	00h

Object:

6022h	0	Slop Lateral Preset Value	Signed 16	Rw	-	Corrects the measured sensor value. The displayed value Slop Lateral is set to the entered value. The offset is indicated in the index 0x6023

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	22h	60h	00h	00h	00h	00h	00h

With the aim to save functionality write the “save” command as below:

Write (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	23h	10h	10h	01h	73h	61h	76h	65h

Note: save command is given by sending the code:

73h	61h	76h	65h

Where:

73h = ASCII code **“s”**

61h = ASCII code **“a”**

76h = ASCII code **“v”**

65h = ASCII code **“e”**

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	10h	10h	01h	00h	00h	00h	00h

IMPORTANT NOTE:

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 7) How to set the angle Z to 0.00° (in case of single axis ±180°).

The values "Preset Value" (Idx 60x2) and "Differential Offset" (Idx 60x4) affects the display of the longitudinal and lateral axis. The value entered in "Preset Value" immediately corrects the measured value of the sensor cell at the instant tacc.

A typical application is the compensation of display errors due to mounting (e.g. sensor zeroing). The sensor must first be brought to a defined position. The value "Differential Offset" shifts the displayed value of the sensor by the entered value.

A set "Preset Value" does not affect shifting.

Note that the resolution parameter must be set before aligning the sensor (resolution, Idx 6000)!

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data	Data		

CMD *determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

RES *Response of the slave:*

60 hex Data sent successfully

80 hex Error,

With the aim to preset the Z angle to 0.00° write (in the example the Node-ID = 0x7F).

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Bh	12h	60h	00h	00h	00h	00h	00h

Object:

6012h	0	Slop Lateral Preset Value	Signed 16	Rw	-	Corrects the measured sensor value. The displayed value Slop Lateral is set to the entered value. The offset is indicated in the index 0x6013

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	12h	60h	00h	00h	00h	00h	00h

With the aim to save functionality write the “save” command as below:

Write (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	23h	10h	10h	01h	73h	61h	76h	65h

Note: save command is given by sending the code:

73h	61h	76h	65h

Where:

73h = ASCII code **“s”**

61h = ASCII code **“a”**

76h = ASCII code **“v”**

65h = ASCII code **“e”**

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	10h	10h	01h	00h	00h	00h	00h

IMPORTANT NOTE:

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 8) How to invert the direction (from CW to CCW) in angle Z (in case of single axis ±180°).

The values "Preset Value" (Idx 60x2) and "Differential Offset" (Idx 60x4) affects the display of the longitudinal and lateral axis. The value entered in "Preset Value" immediately corrects the measured value of the sensor cell at the instant tacc.

A typical application is the compensation of display errors due to mounting (e.g. sensor zeroing). The sensor must first be brought to a defined position. The value "Differential Offset" shifts the displayed value of the sensor by the entered value.

A set "Preset Value" does not affect shifting.

Note that the resolution parameter must be set before aligning the sensor (resolution, Idx 6000)!

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data			

***CMD** determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

***RES** Response of the slave:*

60 hex Data sent successfully

80 hex Error,

With the aim to invert the direction (from CW to CCW) in angle Z (in the example the Node-ID = 0x7F).

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Fh	11h	60h	00h	03h	00h	00h	00h

Object:

6011h	0	Slop Lateral Preset Value	Signed 8	Rw	0x02 (2d)	Inverting the sign 0b 0000 00x0 deactivated 0b 0000 00x1 activated

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	11h	60h	00h	00h	00h	00h	00h

With the aim to save functionality write the “save” command as below:

Write (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	23h	10h	10h	01h	73h	61h	76h	65h

Note: save command is given by sending the code:

73h	61h	76h	65h

Where:

73h = ASCII code **“s”**

61h = ASCII code **“a”**

76h = ASCII code **“v”**

65h = ASCII code **“e”**

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	10h	10h	01h	00h	00h	00h	00h

IMPORTANT NOTE:

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 9) How to change the resolution from ±0.05° to ±0.01°.

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data	Data		

CMD *determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

RES *Response of the slave:*

60 hex Data sent successfully

80 hex Error,

With the aim to change the resolution from ± 0.05° (0x32) to ±0.01° (0x0A) write (in the example the Node-ID=0x7F).

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Bh	00h	60h	00h	0Ah	00h	00h	00h

Object:

6000h	0	Resolution	Unsigned 16	Rw	0x32 (50d)	Display resolution of the inclination for both axes(1) 10d = Inclination is indicated as signed int in 0.01° 50d = Inclination is indicated as signed int in 0.05° 100d = Inclination is indicated as signed int in 0.1° 500d = Inclination is indicated as signed int in 0.5° 1000d = Inclination is indicated as signed int in 1.0° Note: If the display resolution is changed all offset values or zero point values which may have been entered are deleted. Therefore the sensor must be set before it is aligned! (1) A change of the display resolution in Idx 6000 is only accepted, if the scaling in Idx 6011 and Idx 6021 is activated.

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	00h	60h	00h	00h	00h	00h	00h

With the aim to save functionality write the “save” command as below:

Write (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	23h	10h	10h	01h	73h	61h	76h	65h

Note: save command is given by sending the code:

73h	61h	76h	65h

Where:

73h = ASCII code **“s”**

61h = ASCII code **“a”**

76h = ASCII code **“v”**

65h = ASCII code **“e”**

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	10h	10h	01h	00h	00h	00h	00h

IMPORTANT NOTE:

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 10) How to change the Filter Setting from FAST (Filter = 2) to SLOW (Filter = 0).

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data			

CMD *determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

RES *Response of the slave:*

60 hex Data sent successfully

80 hex Error,

With the aim to change the filter settings from FAST response (0x02) to SLOW response (0x00) write (in the example the Node-ID = 0x7F).

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Fh	01h	20h	00h	00h	00h	00h	00h

Object:

2001h	0	Filter Setting	Unsigned 8	Rw	2	Filter = 0 Slow Filter = 1 Medium Filter = 2 Fast

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	01h	02h	00h	00h	00h	00h	00h

With the aim to save functionality write the “save” command as below:

Write (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	23h	10h	10h	01h	73h	61h	76h	65h

Note: save command is given by sending the code:

73h	61h	76h	65h

Where:

73h = ASCII code **“s”**

61h = ASCII code **“a”**

76h = ASCII code **“v”**

65h = ASCII code **“e”**

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	10h	10h	01h	00h	00h	00h	00h

IMPORTANT NOTE:

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 11) How to send the command RESTORE

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data	Data	Data	Data

CMD *determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index	Data	Data	Data	Data

RES Response of the slave:

60 hex Data sent successfully

80 hex Error,

With the aim to restore all parameters to default write (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	23h	11h	10h	01h	6Ch	6Fh	61h	64h

Object:

1011h	1	Load all parameters	Unsigned 8	Wo		"load" (0x64616F6C) to restore all parameters (objects with marking PARA and LSSPARA).

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	11h	10h	01h	00h	00h	00h	00h

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 12) How to disable the Asynchronous Transmission (Asynchronous TPDO inactive)

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data	Data		

CMD determines the direction of data transfer and the size of the data object:

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

RES *Response of the slave:*

60 hex Data sent successfully

80 hex Error,

With the aim to disable the asynchronous transmission write theSDO (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Bh	00h	18h	05h	00h	00h	00h	00h

Object:

1800h	0	1st Transmit PDO Parameter	Unsigned 8	Ro		
	1	COB-ID Trans PDO	Unsigned 32	Ro	180h+ Node-ID	
	2	Transmission Type Trans PDO - PARA	Unsigned 8	Rw	254 (0xFE)	0x01...0xF0 = synch cyclic Outputs are only updated after "n" synch objects. n = 0x01 (1) ... 0xF0 (240) 0xFC not implemented 0xFD not implemented 0xFE = asynchronous 0xFF = not implemented
	5	Event timer Trans PDO- PARA	Unsigned 16	Rw	100 (0x64)	0 = inactive Min= 4 & Max=65535 with unit = 1ms

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	00h	18h	05h	00h	00h	00h	00h

With the aim to save functionality write the “save” command as below:

Write (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	23h	10h	10h	01h	73h	61h	76h	65h

Note: save command is given by sending the code:

73h	61h	76h	65h

Where:

73h = ASCII code **“s”**

61h = ASCII code **“a”**

76h = ASCII code **“v”**

65h = ASCII code **“e”**

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	10h	10h	01h	00h	00h	00h	00h

IMPORTANT NOTE:

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).

Example 13) How to enable the Synchronous Transmission (Synchronous TPDO active after 1st sync message)

With Service Data Object (S.D.O.) the access to entries of a device Object Dictionary is provided. As these entries may contain data of arbitrary size and data type SDOs can be used to transfer multiple data sets from a client to a server and vice versa.

Structure of SDO-request by the Master

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
600+Node-ID	8	CMD	Index		Sub-Index	Data			

***CMD** determines the direction of data transfer and the size of the data object:*

23 hex Sending of 4-byte data (bytes 5...8 contain a 32-bit value)

2B hex Sending of 2-byte data (bytes 5, 6 contain a 16-bit value)

2F hex Sending of 1-byte data (byte 5 contains an 8-bit value)

Structure of SDO-answer by the Slave

COB-ID	DLC	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
580+Node-ID	8	RES	Index		Sub-Index				

***RES** Response of the slave:*

60 hex Data sent successfully

80 hex Error,

With the aim to enable the synchronous transmission with TPDO active after 1st sync message, write the SDO (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	2Fh	00h	18h	02h	01h	00h	00h	00h

Object:

1800h	0	1st Transmit PDO Parameter	Unsigned 8	Ro		
	1	COB-ID Trans PDO	Unsigned 32	Ro	180h+ Node-ID	
	2	Transmission Type Trans PDO - PARA	Unsigned 8	Rw	254 (0xFE)	0x01...0xF0 = synch cyclic Outputs are only updated after "n" synch objects. n = 0x01 (1) ... 0xF0 (240) 0xFC not implemented 0xFD not implemented 0xFE = asynchronous 0xFF = not implemented
	5	Event timer Trans PDO- PARA	Unsigned 16	Rw	100 (0x64)	0 = inactive Min= 4 & Max=65535 with unit = 1ms

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	00h	18h	02h	00h	00h	00h	00h

With the aim to save functionality write the “save” command as below:

Write (in the example the Node-ID = 0x7F)

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
67Fh	23h	10h	10h	01h	73h	61h	76h	65h

Note: save command is given by sending the code:

73h	61h	76h	65h

Where:

73h = ASCII code **“s”**

61h = ASCII code **“a”**

76h = ASCII code **“v”**

65h = ASCII code **“e”**

The answer after successful storing you will receive is:

ID	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
5FFh	60h	10h	10h	01h	00h	00h	00h	00h

IMPORTANT NOTE:

After setting the new entries a reset must be made so that the new entries become valid (switch off the module for a short time).