GEFRAN

LM-C

Contactless linear position transducer with TWIIST technology (CANopen output)

CANopen

Cod. 80721 Edit.12/2023 - ENG

REVISION HISTORY		
Rev. 0	31-03-2021	First release
Rev. 1	15-03-2022	Second release including bootloader functionality
Rev. 2	22-12-2023	Third release, firmware 2.x revision

SUMMARY

1. INTRODUCTION

The Gefran LM-C is a digital multivariable position sensor with CANopen interface.

It implements the standard CANopen communications protocol defined by CiA (CAN in Automation).

The CANopen standards supported by the device are listed in Table 1.

Table 1. Supported CANopen standards

CiA standard	Description	Version
DS 301	CANopen application layer and communication profile	4.2.0
DS 302	Additional application layer functions	4.1.0
DS 303	Recommendation	1.8.0
DS 305	Layer setting services (LSS) and protocols	3.0.1
DS 406	Device profile for encoders (linear multi-sensor encoder)	4.1.0
DS 410	Device profile for inclinometers (class C2)	2.0.0

This document describes the CANopen implementation on the Gefran LM-C CANopen device.

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 CiA. The details of aspects defined by CANopen do not pertain to the purpose of this text. For further information on the CANopen protocol see www.can-cia.de.

1.1. Working principle Working principle

1.1.1. Position and speed measurement

Gefran LM-C sensor exploits the patented TWIIST technology. The main components of this technology are the helical magnet and the triaxial Hall effect IC, as shown in Figure 1: the helical magnet, that composes the magnetic core, and the triaxial Hall effect IC, included in the sensor rod.

From the mathematical description of the helix, a unique pair of magnetic field values Bx, Bz is identified for each measurement position. The arctangent of the ratio between Bx and Bz identifies the rotation angle of the helix, corresponding to the measured position p.

A speed estimation algorithm, based on the discrete derivative of the position with respect to the sampling time, is embedded in the sensor firmware without requiring any computational contribution to the user control unit.

As described in Figure 2, the Gefran LM-C sensor is proposed in half-redundant architecture which allows to obtain two independent position measurements. Therefore, the block diagram of the sensor includes two independent Hall-effect primary elements that share the same electronic board, microcontroller, bus transceiver, and power management circuit.

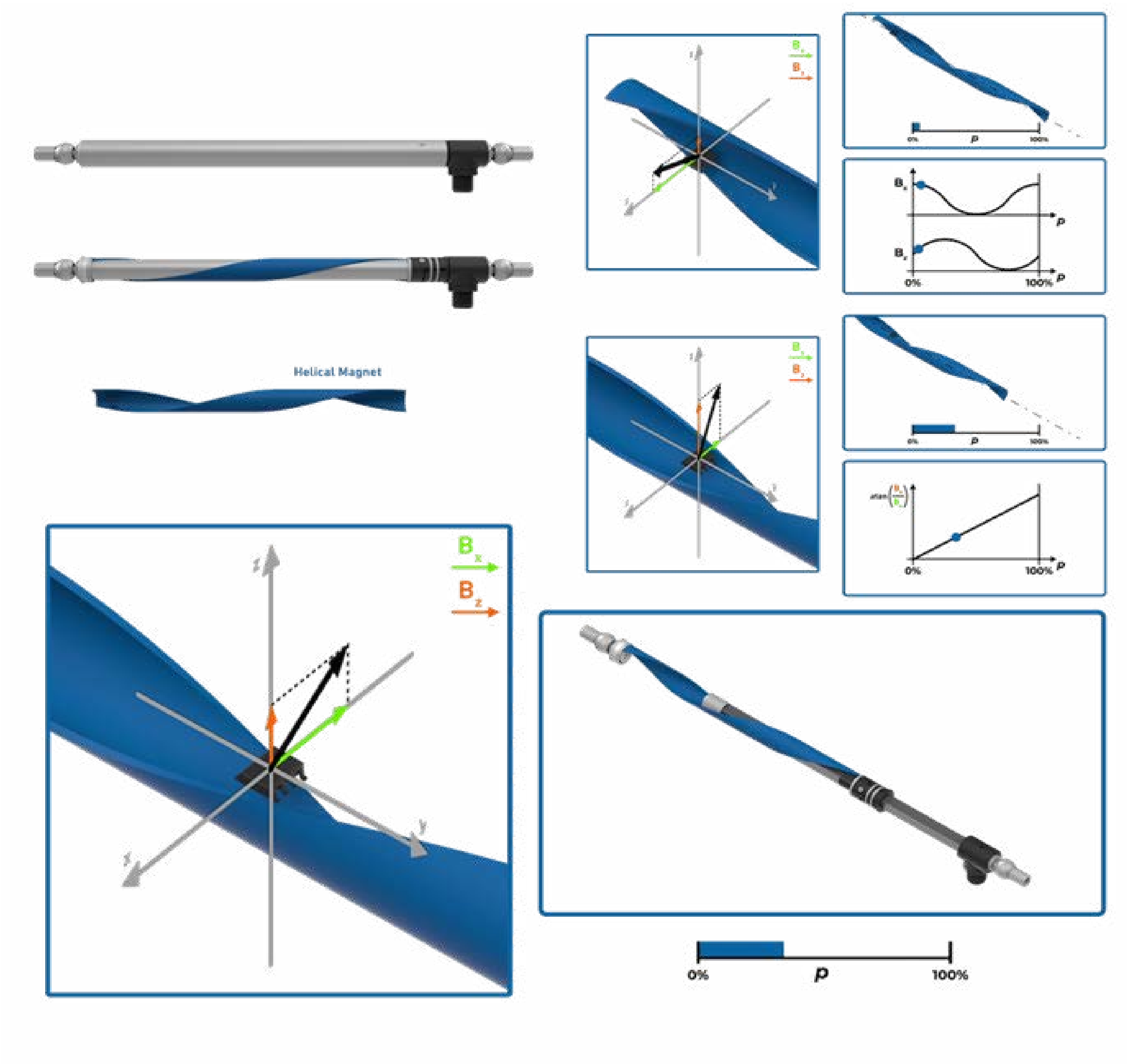

Figure 1. Working principle of position measurement

In Figure 3 an example of position measurement performed with the Gefran LM-C sensor is presented.

The sensor is fixed at both ends to the process to be measured. It does not matter which end moves and which one remains fixed. The sensor will detect the sliding of the magnetic core on the rod, connected at the end with the electrical connection, as an absolute position measurement. The magnetic core can slide on the sensor rod for a length equal to the stroke (FS).

For safety reasons about 1.5 mm of overtravel have been included. Below the ZERO position (0 mm), the sensor will provide negative measurement values up to -1 mm, position that triggers an emergency object. Similarly, above FS the sensor will provide positive measurement values up to FS + 1 mm, position that triggers an emergency object. In order not to irreparably damage the sensor, please do not move the cursor beyond the overtravel thresholds.

Position measurements can be low pass filtered by selecting the filter tap according to object 2010h sub-index 02h.

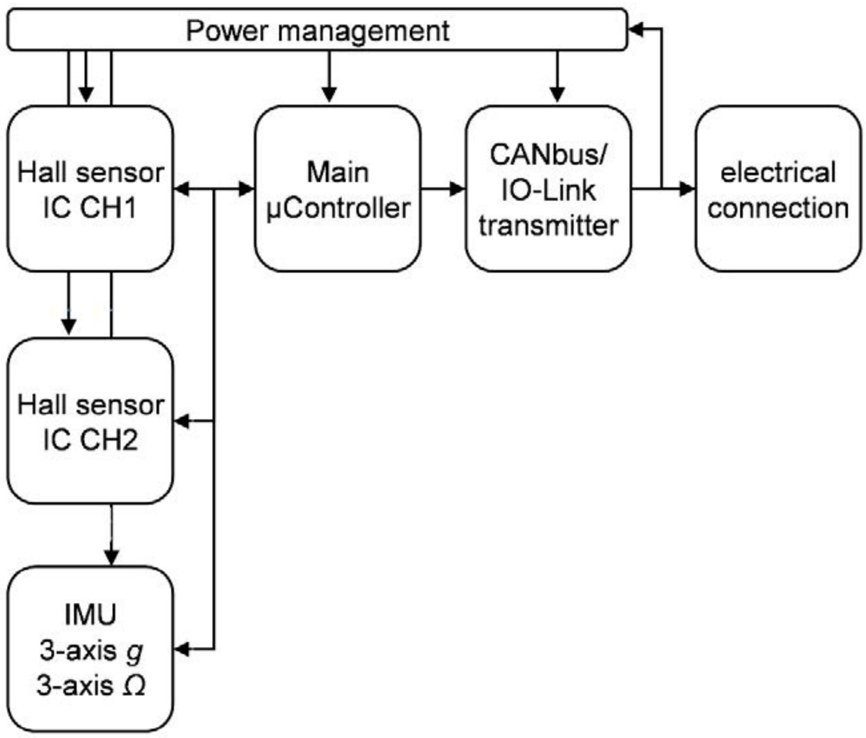

Figure 2. Architecture description: block diagram

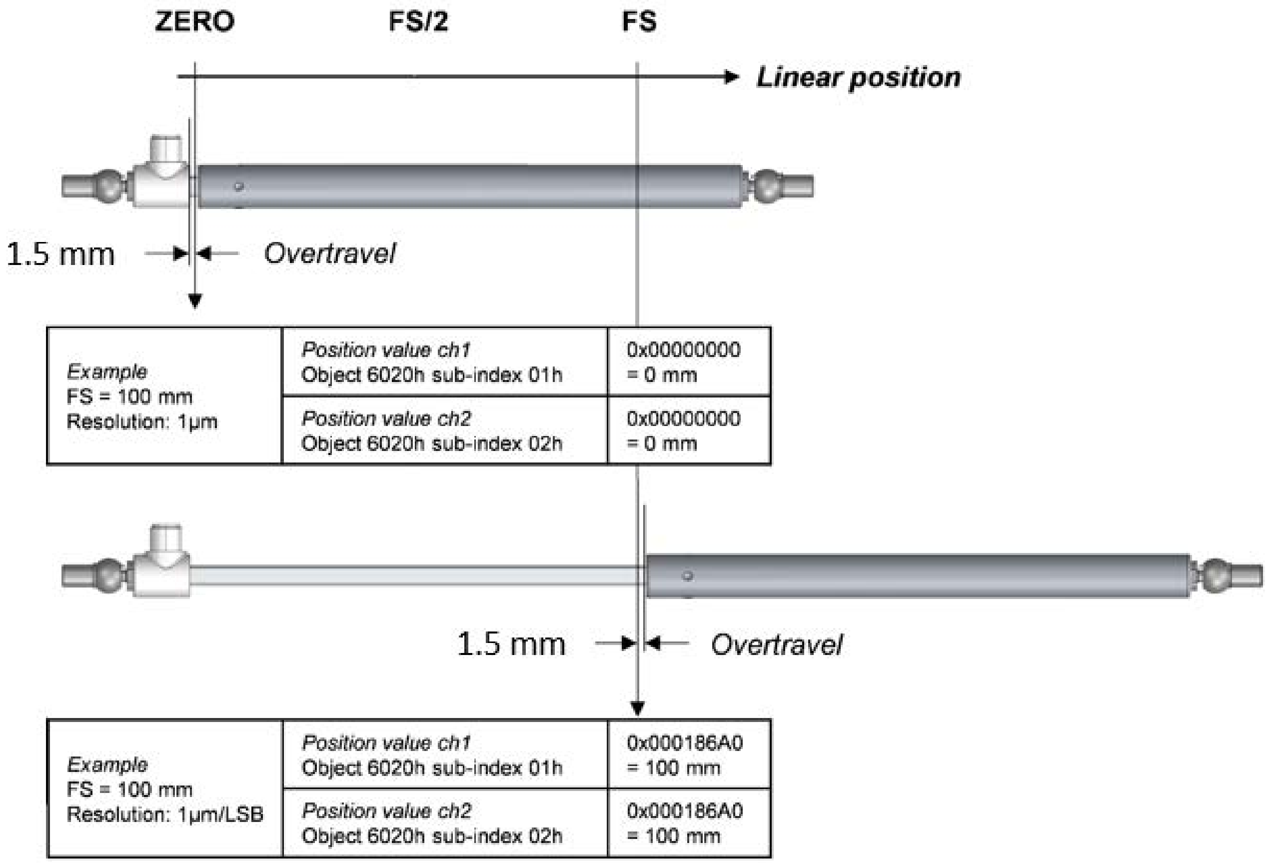

Figure 3. Examples of position measurement

1.1.2. Tilt sensing

The Gefran LM-C sensor in the “multivariable” version includes an inertial module composed of a triaxial accelerometer and a triaxial gyroscope. The sensing axes orientation is described in Figure 4: X, Y, and Z are related to the accelerometer, while Ωx, Ωy and Ωz are related to the gyroscope. These 6-D raw measurements are transmitted by the sensor to the user control unit to permit custom processing.

A sensor fusion algorithm based on the Kalman filter, suitable for the estimation of 3-D orientation in space, is also embedded in the sensor firmware. It acquires data from the accelerometer and gyroscope and provides information about the device position, i.e., quaternions and Euler angles (pitch, roll, yaw) as shown in Figure 5. Pitch and roll angles are mapped, according to CiA DS 410, as slope lateral and slope longitudinal.

Pitch ranges from -180 deg to +180 deg, while roll ranges from -90 deg to +90 deg as describes in Figure 6 and Figure 8.

Pitch angle sweeps the entire measurement range only if roll is around the 0-deg position. The farther roll is from the 0-deg position, the greater the pitch angle measurement error will be. Furthermore, for roll angles close to the edges of the measurement range it is not possible to perform an accurate pitch measurement. For this reason, the user is advised to select the correct IMU orientation parameter (object 2010h sub-index 03h) according to the sensor mounting orientation shown in Figure 9.

Yaw calculation is provided by integration of gyroscope data, as a relative angle position measurement respect to the power-on sensor position that is considered as 0-deg yaw position.

Tilt angle measurements can be low pass filtered by selecting the filter tap according to object 2010h sub-index 03h.

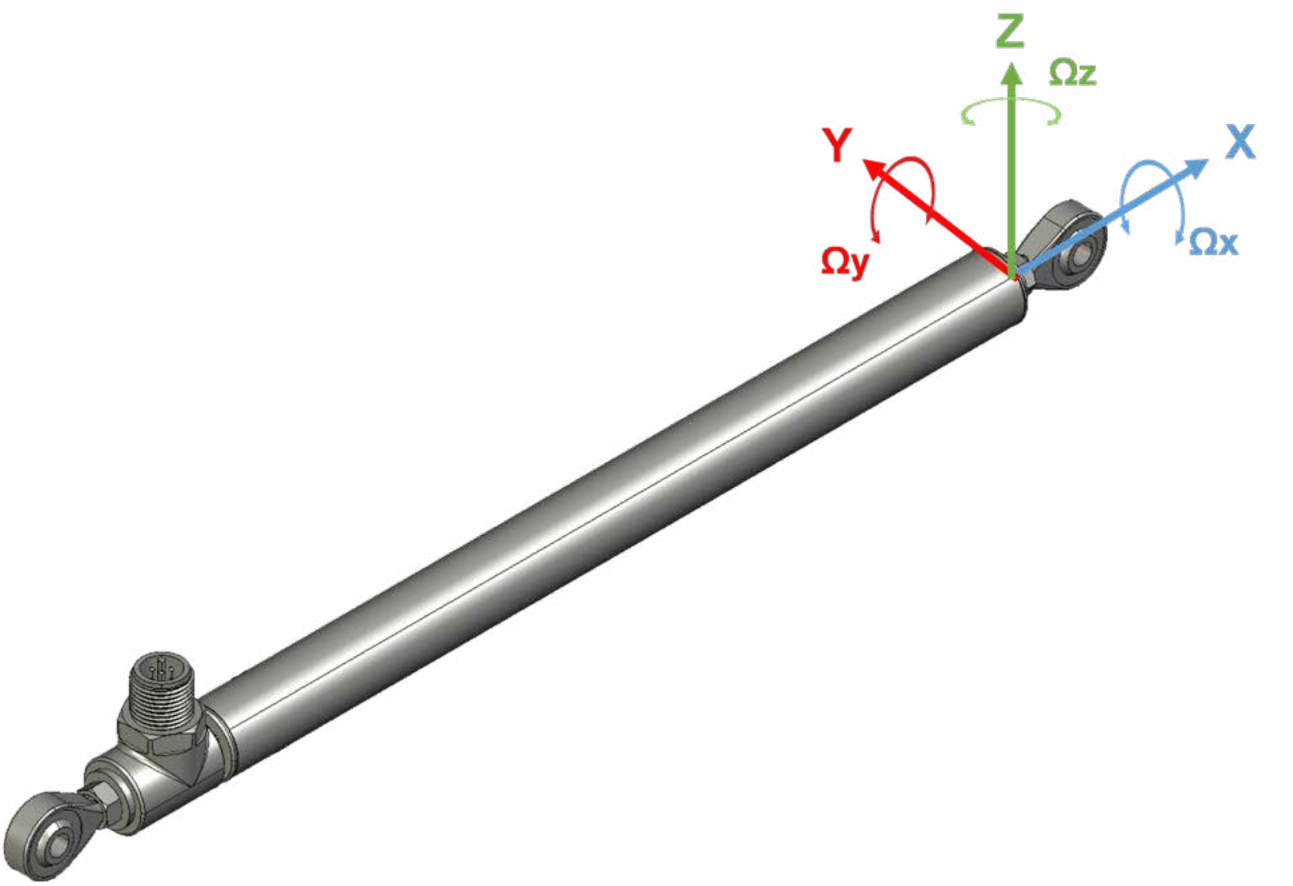

Figure 4. Inertial module sensing axes orientations

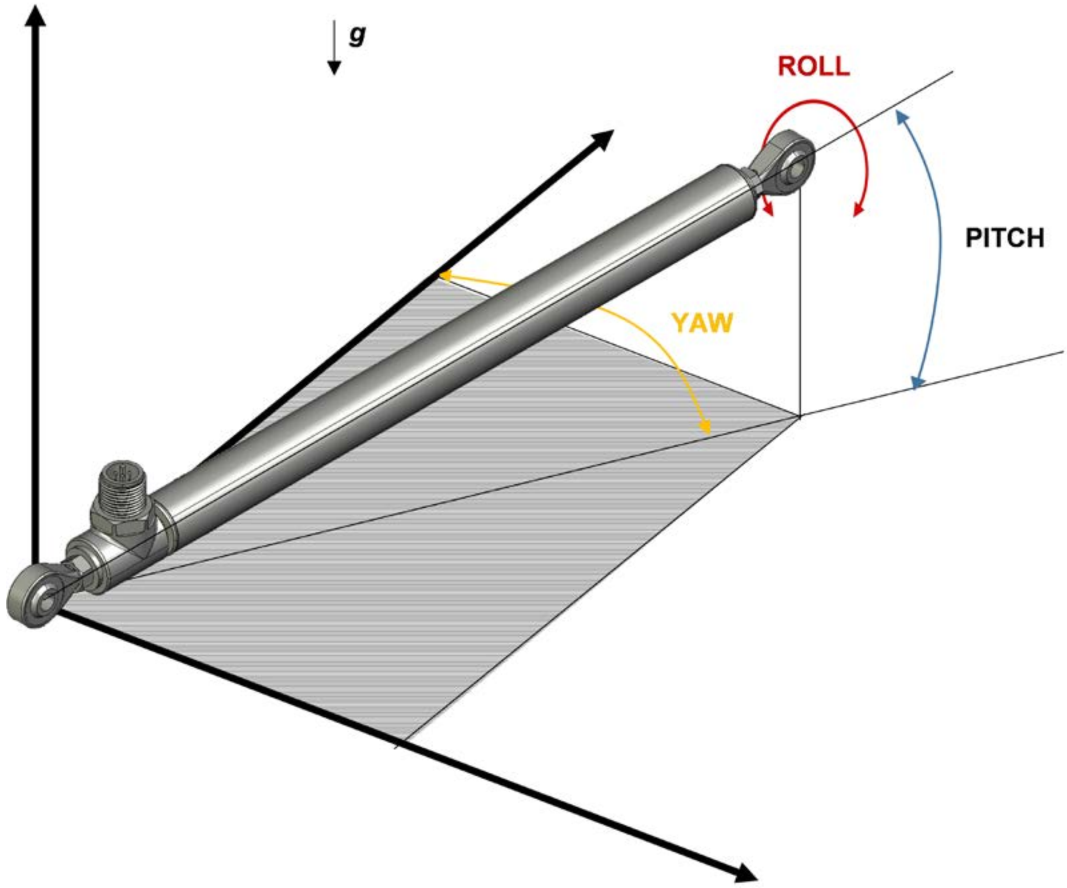

Figure 5. Tilt sensing axes orientations

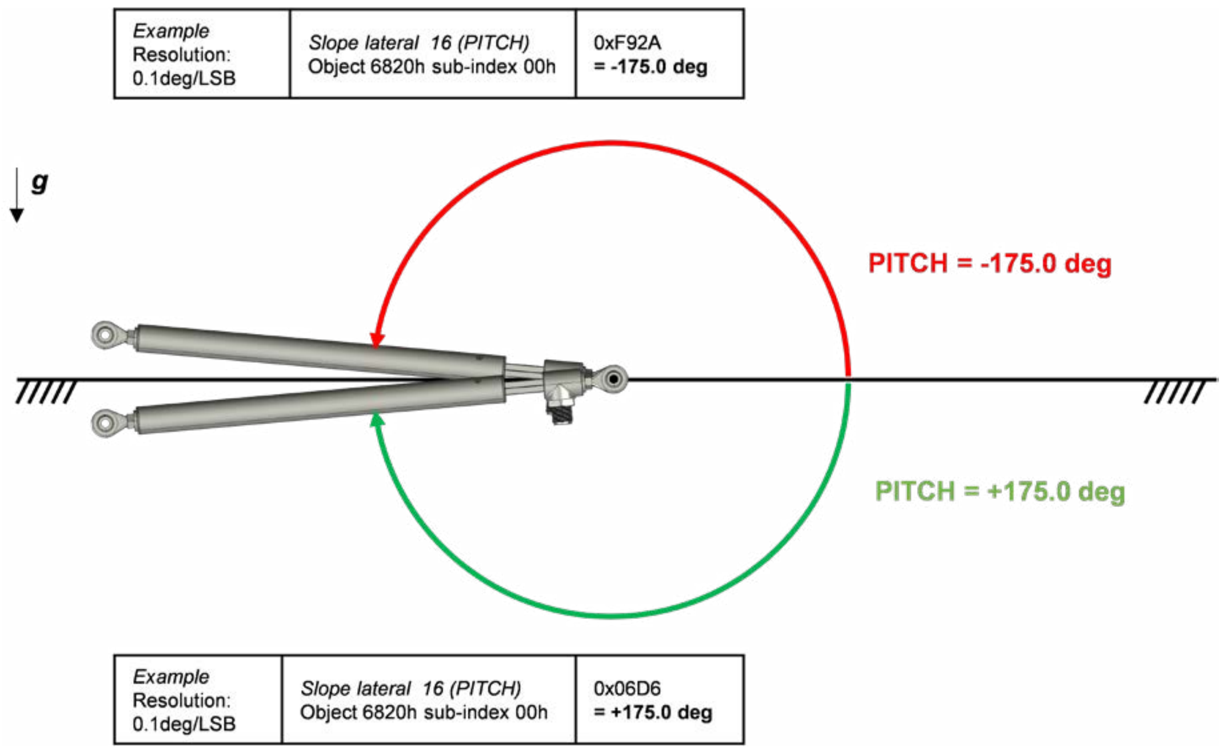

Figure 6. Example of position measurement: pitch = +/-175 deg

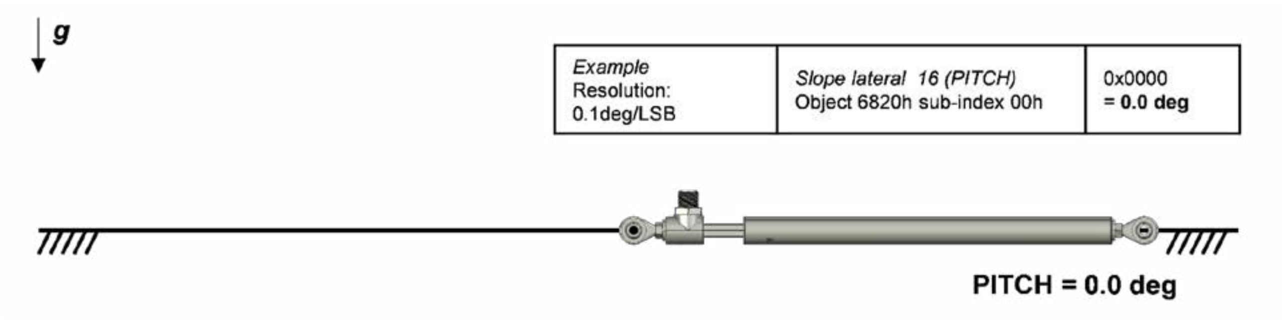

Figure 7. Example of position measurement: pitch = 0 deg

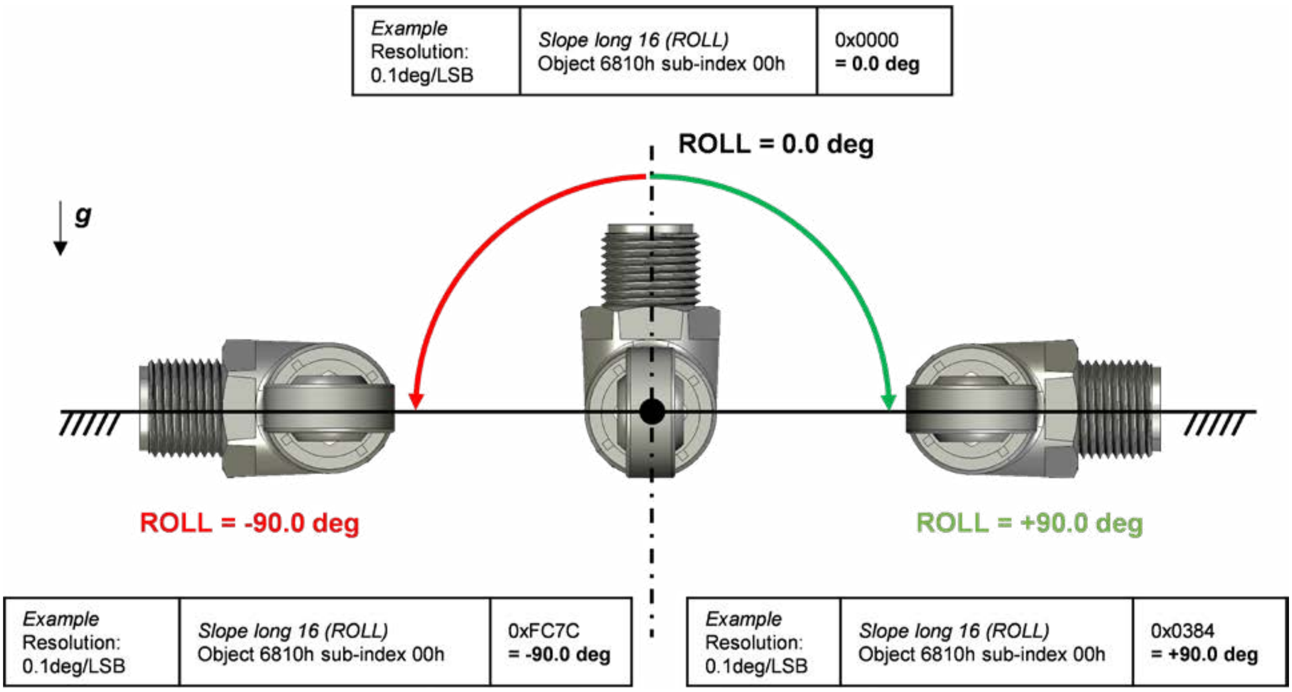

Figure 8. Example of position measurement: roll =-90…0…+90 deg

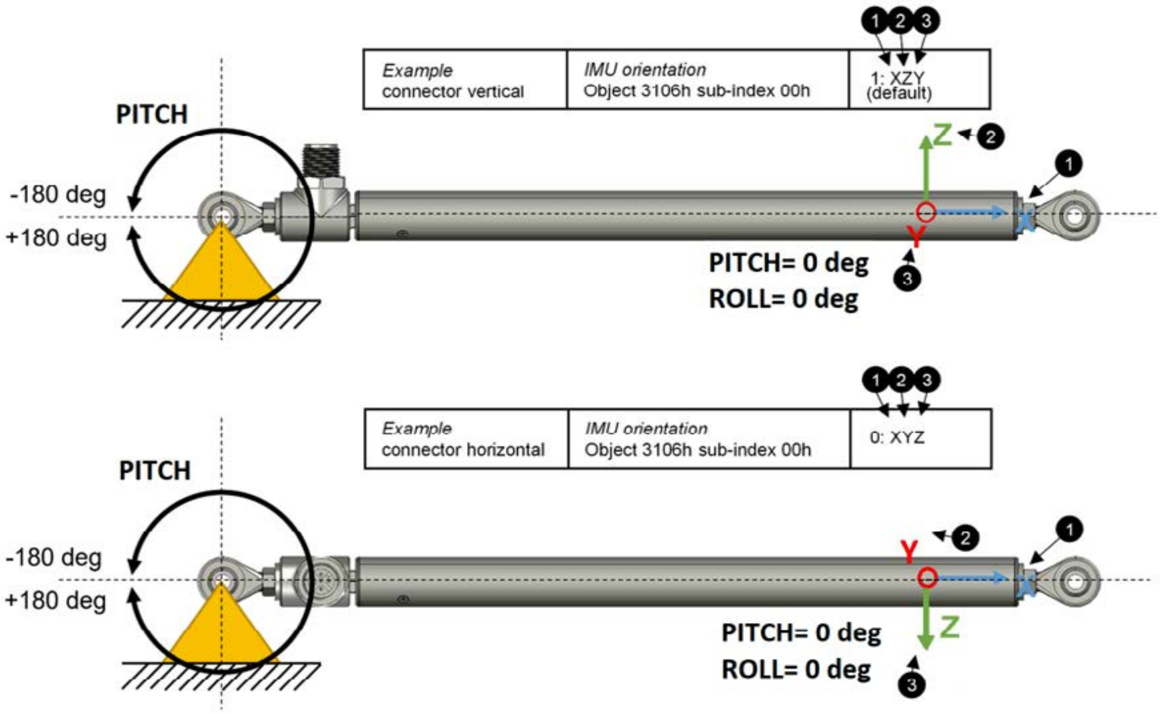

Figure 9. IMU orientation parameter according to the sensor mounting orientation

1.2. PDO measure reading

1.2.1. Position and speed measurement

In this section some examples of how to read the position measurement, obtained by the PDO1 transmitted by the sensor and mapped as described in 7.1, are presented.

Please, note that position measurement of channel one can also be read through SDO request.

- **ZERO:** Figure 10 and Trace 1 show the mechanical position of the sensor and PDO1, which contains the associated measurement, respectively.

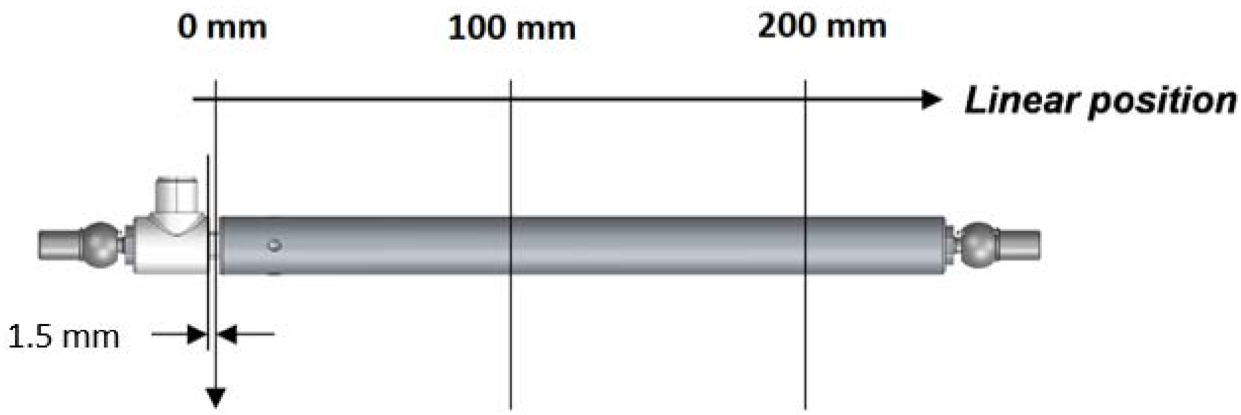

Figure 10. Mechanical position at 0 mm

COB-ID (hex)	Rx/Tx	DLC	Data (hex)	Comment
1FF	Tx	6	00 00 00 00 00 00	PDO 1

Trace 1. Position reading at 0 mm

Byte 0 (LSB) =00h

Byte 1 =00h

Byte 2 =00h

Byte 3 (MSB) =00h

=> Position channel 1= 000000h to decimal 0d = 0 μm

Byte 4 (LSB) =00h

Byte 5 (MSB) =00h

=> Speed channel 1= 0000h to decimal 0d = 0 mm/s

- **FS:** Figure 11 and Trace 2 show the mechanical position of the sensor and PDO1, which contains the associated measurement, respectively.

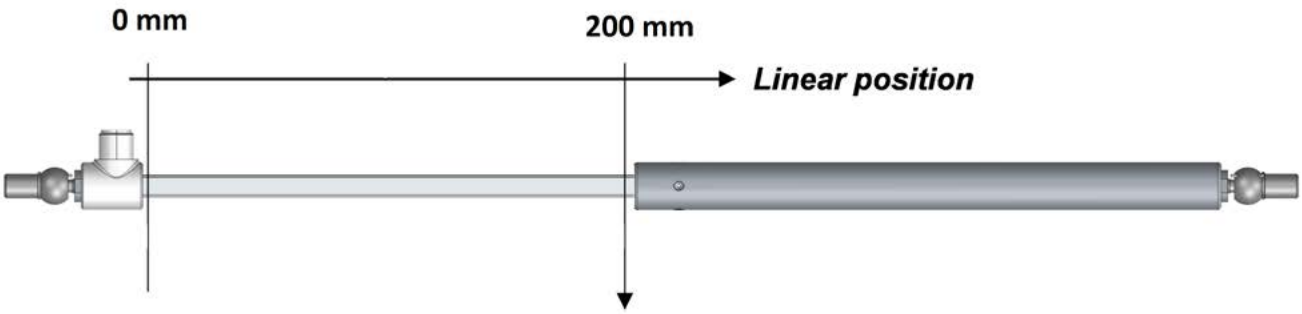

Figure 11. Mechanical position at 200 m

COB-ID (hex)	Rx/Tx	DLC	Data (hex)	Comment
1FF	Tx	6	40 0D 03 00 32 00	PDO 1

Trace 2. Position reading at 200 m

Byte 0 (LSB) =40h

Byte 1 =0Dh

Byte 2 =03h

Byte 3 (MSB) =00h

=> Position channel 1= 00030D40h to decimal 200 000d = 200 000 μm= 200 mm

Byte 4 (LSB) =00h

Byte 5 (MSB) =32h

=> Speed channel 1= 0032h to decimal 50d = 5 mm/s

According to standard CiA DS 406, position measure is calculated as descripted in Figure 12 considering default configuration parameters described in object dictionary:

- Operating parameters (object 6000h, sub-index 00h) = 4d => Scaling enable, direction forward
- Position resolution (object 6005h, sub-index 01h) = 1000d => 1 μm
- Preset (object 6010h, sub-index 01h) = 0
- Offset (object 650Ch, sub-index 01h) = 0
- Total measuring range (object 6002h, sub-index 00h) = 200 000d => 200 000 μm

According to standard CiA DS 406, speed measure is calculated considering default configuration parameters described in object dictionary:

- Speed resolution (object 6005h, sub-index 02h) = 10d => 0.1 mm/s

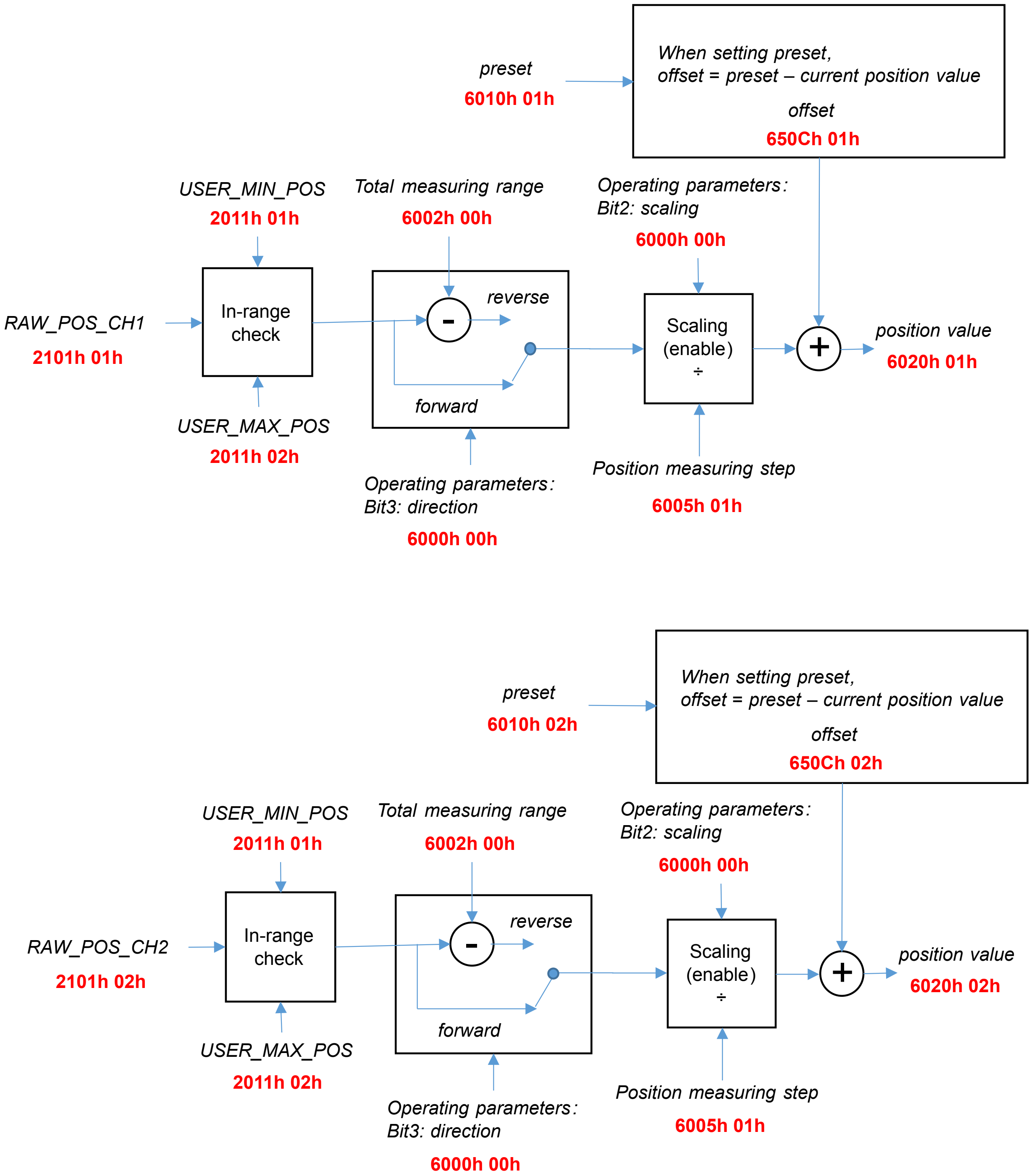

Figure 12. Block diagram of position calculation

1.2.2. Tilt measurement

In this section some examples of how to read the tilt measurement, obtained by the PDO2 transmitted by the sensor and mapped as described in 7.1, are presented. Please, note that pitch and roll measurement can also be read through SDO request.

- **ROLL:** Figure 13 and Trace 3 show the mechanical position of the sensor and PDO2, which contains the associated roll measurement, respectively.

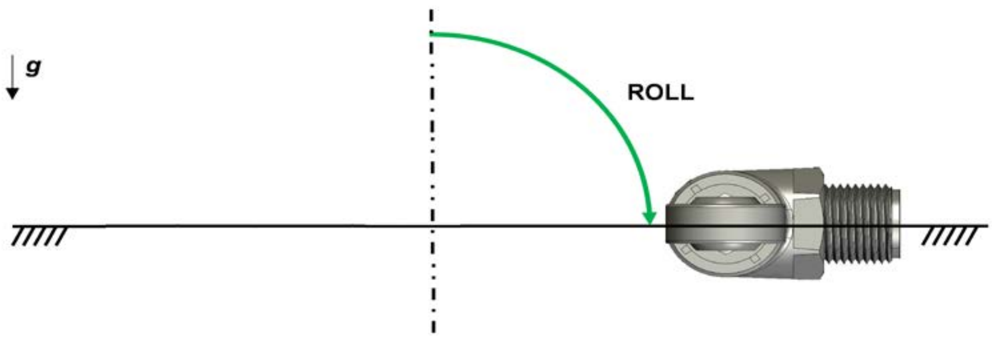

Figure 13. Roll at 90 deg

COB-ID (hex)	Rx/Tx	DLC	Data (hex)	Comment
2FF	Tx	6	1D 84 03 00 00 00	PDO 2

Trace 3. Roll reading at 90 deg

Byte 1 (LSB) =84h

Byte 2 (MSB) =03h

=> Roll= 0384h to decimal 900d = 90.0 deg

- **PITCH:** Figure 14 and Trace 4 show the mechanical position of the sensor and PDO1, which contains the associated pitch measurement, respectively.

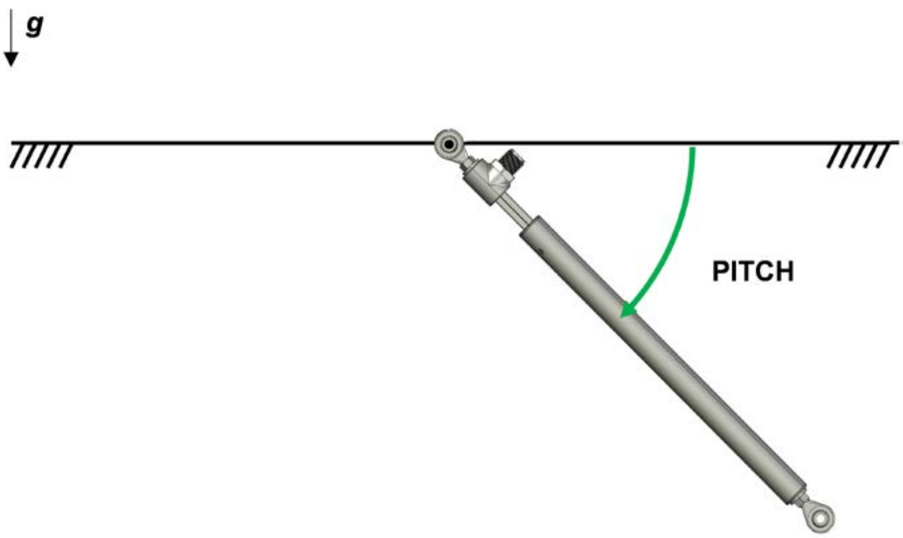

Figure 14. Pitch at 45 deg

COB-ID (hex)	Rx/Tx	DLC	Data (hex)	Comment
2FF	Tx	5	1D 00 00 C2 01	PDO 2

Trace 4. Pitch reading at 45 deg

Byte 3 (LSB) =C2h

Byte 4 (MSB) =01h

=> Pitch= 01C2h to decimal 450d = 45.0 deg

According to standard CiA DS 410, roll and pitch measure are calculated as descripted in Figure 15 and Figure 16, considering default configuration parameters described in object dictionary:

- Slope long16 operating parameters (object 6811h, sub-index 00h) = 2d => Scaling enable, inversion disable
- Slope lateral16 operating parameters (object 6821h, sub-index 00h) = 2d => Scaling enable, inversion disable
- Resolution (object 6800h, sub-index 00h) = 100d => 0.1 deg
- Slope long16 preset (object 6812h, sub-index 00h) = 0
- Slope lateral16 preset (object 6822h, sub-index 00h) = 0
- Slope long16 offset (object 6813Ch, sub-index 00h) = 0
- Slope lateral16 offset (object 6823Ch, sub-index 00h) = 0
- Differential slope long16 offset (object 6814Ch, sub-index 00h) = 0
- Differential slope lateral16 offset (object 6824Ch, sub-index 00h) = 0

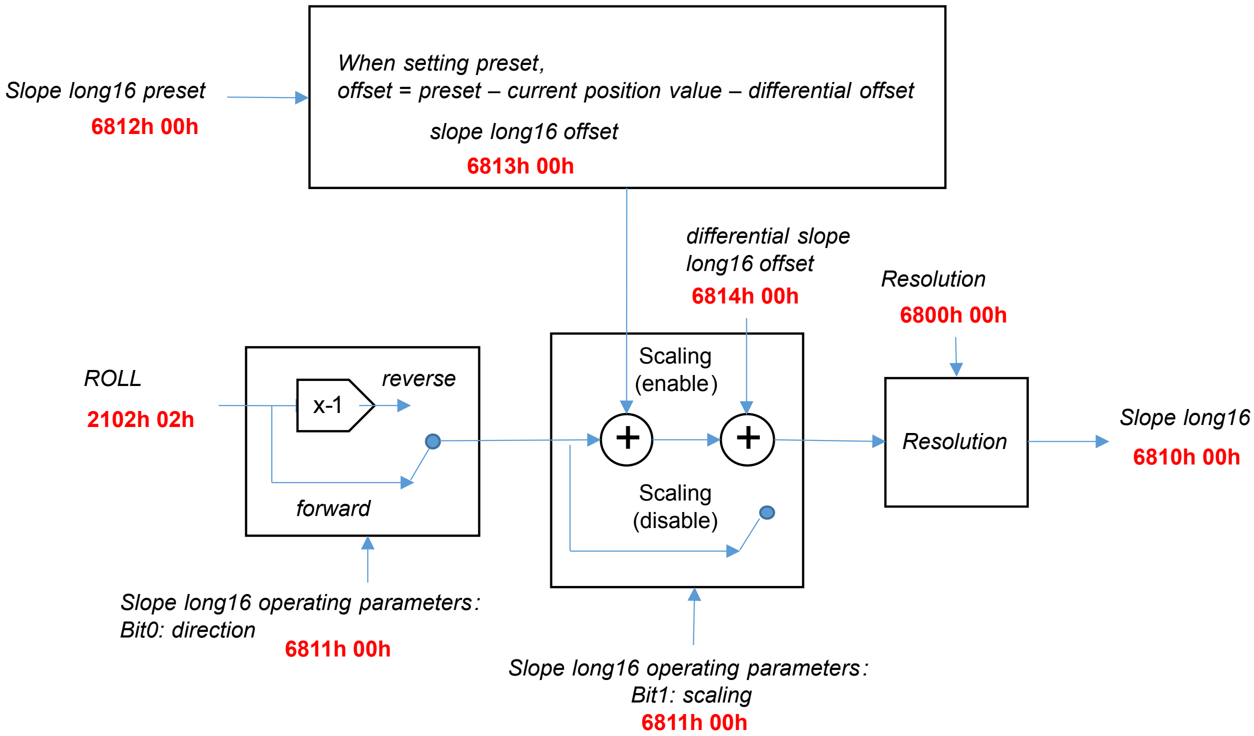

Figure 15. Block diagram of roll calculation

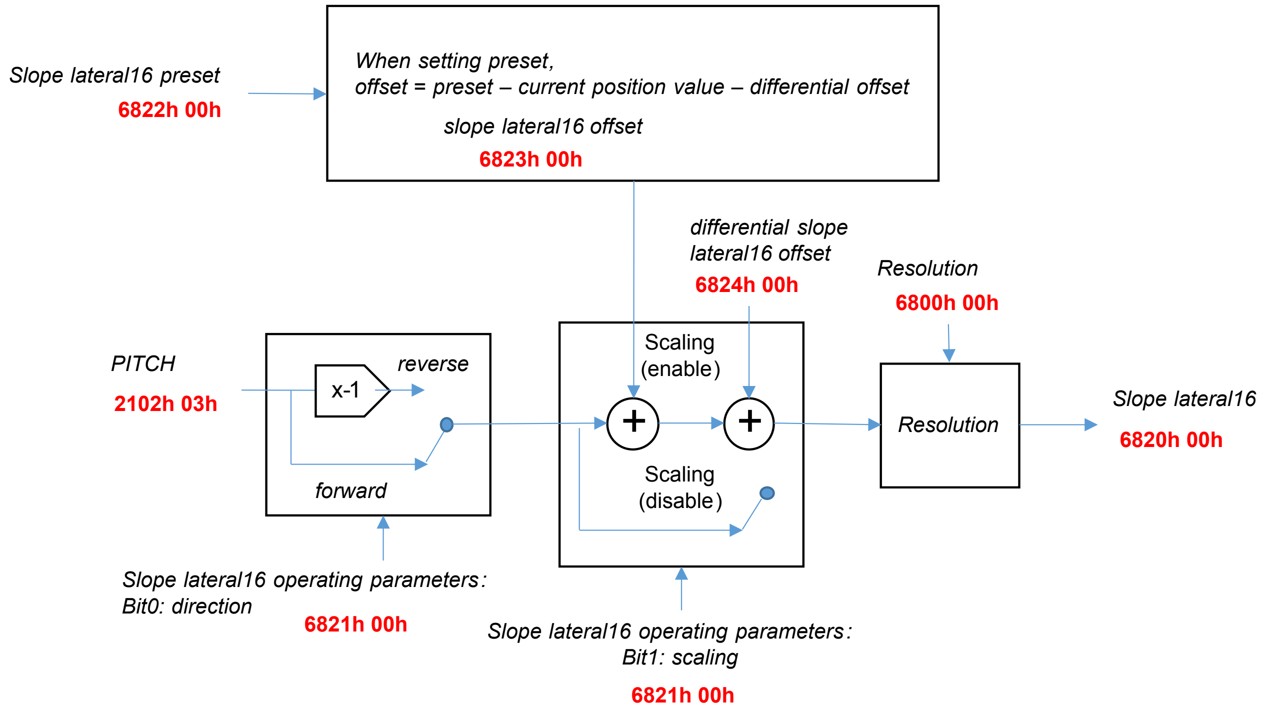

Figure 16. Block diagram of pitch calculation

2. ELECTRICAL CONNECTIONS

The M12 5 poles connector, referred to the standard CiA 303, is described in Figure 17.

PIN	M12 5 pole connector
1	N.C. (not internally connected)
2	V+
3	V-
4	CAN_H
5	CAN_L

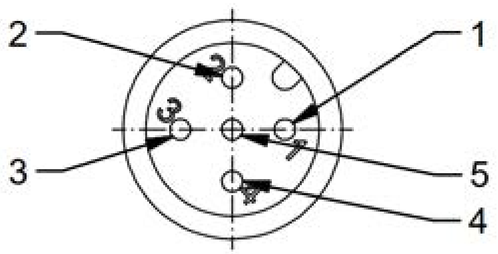

Figure 17. M12 5poles connections

The impedance measured between CAN-H and CAN-L must be 60 Ω that means the cable must be connected to a 120-Ω resistor on each ends of the bus line.

The sensor is not internally terminated, therefore a 120 Ω resistor must be connected. Do not confuse the signal lines of the CANbus, otherwise communication with the transducer is impossible.

3. GET START PROCEDURE

3.1. Default parameters

In this section the principal parameters of Gefran LM-C sensor and the associated default values are presented:

- **Node-ID:** 0x7F
- **Baud rate:** 250 kbit/s
- **Automatic NMT operational state after power-on:** disable (Object 2010h sub-index 01h)
- **Linear position resolution:** 1 μm (Object 6005h sub-index 01h for channel 1, Object 6005h sub-index 02h for channel 2)
- **Speed resolution:** 0.1 mm/s (Object 6005h sub-index 01h for channel 1, Object 6005h sub-index 02h for channel 2)
- **Tilt angles resolution:** 0.1 deg (Object 6800h sub-index 00h)
- **Raw acceleration data resolution:** 1/4096 g
- **Raw angular rate data resolution:** 1/16 deg/s
- **TPDO1 parameters:**
    - COB-ID: 180h + Node-ID
    - Event timer: 4 ms (Object 1800h sub-index 05h)
- **TPDO1 mapping:**
    - Byte 1, 2, 3, 4: Position value channel 1 (Object 6020h sub-index 01h) signed integer 32bit
    - Byte 5, 6: Speed value channel 1 (Object 6030h sub-index 01h) signed integer 16bit.
- **TPDO2 parameters:**
    - COB-ID: 280h + Node-ID
    - Event timer: 12 ms (Object 1801h sub-index 05h)
- **TPDO2 mapping:**
    - Byte 1, 2: Temperature (Object 2100h sub-index 00h) signed integer 16bit
    - Byte 3, 4: Slope longitudinal 16 ROLL (Object 6810h sub-index 00h) signed integer 16bit
    - Byte 5, 6, Slope lateral 16 PITCH (Object 6820h sub-index 00h) signed integer 16bit
- **TPDO3 parameters:**
    - COB-ID: 80000380h + Node-ID (the TPDO is set disabled as default)
    - Event timer: 12 ms (Object 1802h sub-index 05h)
- **TPDO3 mapping:**
    - Byte 1, 2: Raw data output accelerometer sensor X axis (Object 2104h sub-index 01h) signed integer 16bit
    - Byte 3, 4: Raw data output accelerometer sensor Y axis (Object 2104h sub-index 02h) signed integer 16bit
    - Byte 5, 6, Raw data output accelerometer sensor Z axis (Object 2104h sub-index 03h) signed integer 16bit
- **TPDO4 parameters:**
    - COB-ID: 80000480h + Node-ID (the TPDO is set disabled as default)
    - Event timer: 4 ms (Object 1803h sub-index 05h)
- **TPDO4 mapping:**
    - Byte 1, 2, 3, 4: Position value channel 2 (Object 6020h sub-index 02h) signed integer 32bit
    - Byte 5, 6: Speed value channel 2 (Object 6030h sub-index 02h) signed integer 16bit.

3.2. Node parameters setting

Before connecting the GEFRAN LM-C sensor to a fully configured and working CAN bus, some basic configuration actions have to be performed. The configuration involves the node-ID and the baud rate of the CANopen device.

The configuration is mandatory if at least one of these conditions is true:

a) The node-ID of the Gefran LM-C sensor is identical to the node-ID of another device connected to the CAN bus.

b) The Gefran LM-C sensor operates with a baud rate different from the CAN bus baud rate.

If the condition at point b) in not verified, the configuration can also be performed on that CAN bus, but all the other CANopen devices on the CAN bus should be taken in power-off state during the configuration process to avoid errors or conflicts.

If the baud rate configuration has to be performed, the Gefran LM-C sensor must be connected to a CAN bus that works at the same baud rate of the sensor. The baud rate of the actual CAN bus can also be temporary set equal to the sensor baud rate until configuration is done. The configuration is made using LSS (Layer Setting Services).

Switching to LSS configuration mode

The first operation is to switch the sensor into LSS configuration mode. If the sensor is the only device on the CAN bus (with the LSS master), the LSS Switch State Global command can be used.

Table 2. LSS Switch State Global command

Source	COB-ID	DLC	Data	Destination
Controller	7E5h	08h	04h; 01h; 00h; 00h; 00h; 00h; 00h; 00h	Sensor

If there are other devices on the CAN bus (except the LSS master), the LSS Switch State Selective command must be used. Refer to the LSS Services section for details.

Setting the Node-ID

If the node-ID of the sensor has to be changed, the LSS Configure node-ID command must be used, the slave sends the response message:

Table 3. LSS Configure Node-ID command

Source	COB-ID	DLC	Data	Destination
Controller	7E5h	08h	11h; **7Eh***; 00h; 00h; 00h; 00h; 00h; 00h	Sensor
Sensor	7E4h	08h	11h; **00h****; 00h; 00h; 00h; 00h; 00h; 00h	Controller

* the node-ID value to be configured, within 1..127 (126=7Eh in this example).

** if value is 1, it means node-ID out of range, i.e., the command was not accepted.

Setting the baud rate

If the baud rate of the sensor has to be changed, the LSS Configure Bit Timing Parameters command must be used, the slave sends the response message:

Table 4. LSS Configure Bit Timing Parameters command

Source	COB-ID	DLC	Data	Destination
Controller	7E5h	08h	13h; 00h; **02h***; 00h; 00h; 00h; 00h; 00h	Sensor
Sensor	7E4h	08h	13h; **00h****; 00h; 00h; 00h; 00h; 00h; 00h	Controller

* The table-index of the corresponding bit rate (500kbit/s in this example). Refer to Table 15 in the LSS Configure Bit Timing Parameters section for details.

** If the value is 1 means that the bit timing is not supported; the command was not accepted.

Storing configuration settings

To save the previously configured Node-ID and Baud rate permanently (into Flash memory of the device) the LSS Store Configuration command must be used, the slave sends the response message:

Table 5. LSS Store Configuration command

Source	COB-ID	DLC	Data	Destination
Controller	7E5h	08h	17h; 00h; 00h; 00h; 00h; 00h; 00h; 00h	Sensor
Sensor	7E4h	08h	17h; 00h*; 00h; 00h; 00h; 00h; 00h; 00h	Controller

* Value other than 0, means store operation failed.

Verifying configuration setting

To check if the configuration settings of the device have been correctly executed and stored, proceed as follows:

- power off the device
- set the baud rate of the CAN bus to the correct value
- power on the device

If the boot-up message is received, it means that the device baud rate setting is correct. The node-ID of the device is contained inside the COB-ID of the message (boot-up COB-ID = 700h + Node-ID).

The format of the boot-up message is specified in Table 6.

Table 6. Boot-up message format

Source	COB-ID	DLC	Data	Destination
Controller	700h + Node-ID	01h	00h	Controller

4. NMT SERVICES

The device supports CANopen network management functionality NMT, all possible states and transitions are shown in Figure 18.

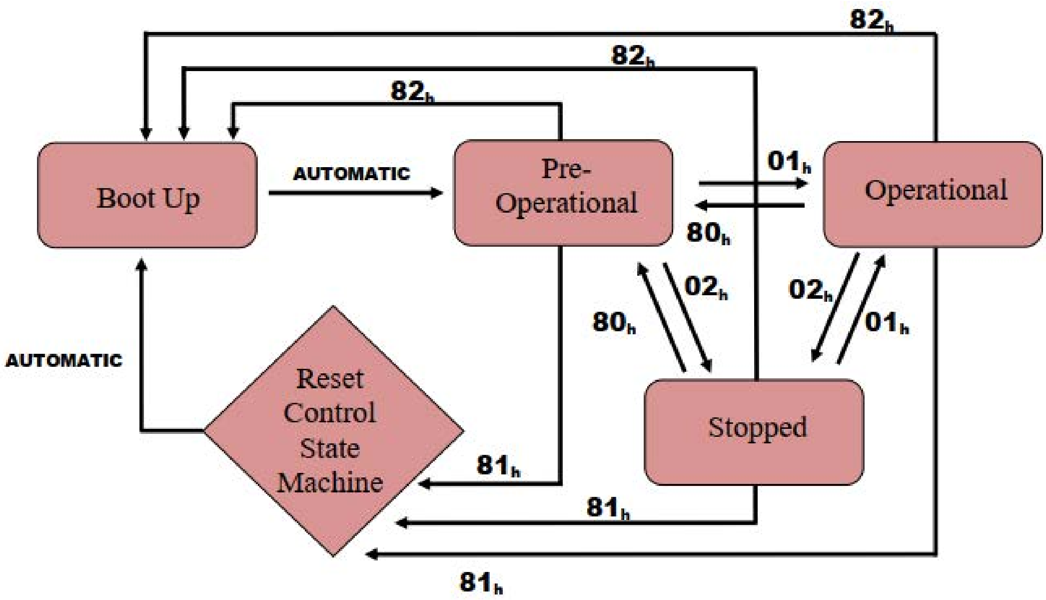

Figure 18. NMT states and state transition

Initialization state

In the NMT state initialization the CANopen device is initialized. The CANopen device parameters are set to their default values (last stored parameters in flash memory).

The NMT state initialization is composed by the sub-states Reset application and Reset communication, which are processed automatically one after the other:

- Reset application: the CANopen device resets all application-related CANopen device parameters and initializes the CANopen node-ID.
- Reset communication: the CANopen device reset all communication-related CANopen device parameters and set the CANopen node-ID.

Pre-operational state

In the pre-operational state, the behaviour of the CANopen device at its communication interface can be configured. This can take place by SDO or LSS services. PDO communication is not available.

Operational state

In the operational state all communication objects are active. Object Dictionary Access via SDO is possible and the node can handle PDO communication.

Stopped state

In the stopped state the device stops the communication. In this state no communication object is supported, except of Error control services and the reception of NMT commands.

4.1. NMT node control

After power-on, the CANopen device initializes; the initialization state terminates with the transmission of the boot-up message, after which the device enters autonomously the pre-operational state.

In order to change the NMT state of a CANopen device, the NTM master sends the message shown in the following tables.

Table 7. NMT messages

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
0	Tx	2	CS	Node-ID	-	-	-	-	-	-

Table 8. NMT messages bit field

Bit field	Value range	Description
CS	01h	Start. Enter NMT Operational state
	02h	Stop. Enter NMT Stopped state
	80h	Enter NMT Pre-operational state
	81h	Enter NMT Reset application state
	82h	Enter NMT Reset communication state
Node-ID	00h	All devices must perform the commanded transition
	01h to 7Fh	Only the device that claims the indicated Node-ID must execute the commanded transition

Specific services can only be executed if the devices involved in the communication are in the appropriate communication states. The relationship between communication states and communication objects is shown in Table 9.

Table 9. NMT states and communication objects

Object	Reset application	Reset communication	Pre-operational	Operational	Stopped
PDO				X	
SDO			X	X	
Boot up transmission		X			
SYNC			X	X	
EMCY			X	X	
NMT error control (Heartbeat)			X	X	X
NMT node control			X	X	

5. LSS SERVICES

LSS protocols are used to inquire or to change the settings of three parameters of the CANopen device:

- Node-ID of the CANopen device
- Bit timing parameters of the physical layer (bit rate)
- LSS address compliant to the identity object (1018h)

5.1. LSS switch state services

ISS switch state global

By means of this service, the LSS master device switches all LSS slave devices in the network into LSS waiting state or LSS configuration state.

The LSS master sends this message to switch the LSS slave(s) into configuration state:

Table 10. LSS switch state global - configuration state - message

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
7E5h	Rx	8	04h	01h	00h	00h	00h	00h	00h	00h

The LSS master sends this message to switch back the LSS slave(s) to waiting state:

Table 11. LSS switch state global - waiting state - message

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
7E5h	Rx	8	04h	00h	00h	00h	00h	00h	00h	00h

LSS switch state selective

By means of this service, the LSS master device switches the LSS slave device, whose LSS address equals the LSS address specified by the messages, into LSS configuration state.

The transmitted LSS address shall be equal to the identity object (object 1018h) of the related LSS slave. The LSS address for the Gefran LM-C CANopen device is specified in Table 12.

Table 12. LM-C LSS Address

	Address Field	Value
	Vendor-ID	00000093h
	Product code	00434D4Ch
	Revision Number	Actual LM-C r.n.*
LSS Address	Serial Number	LM-C S.N. (printed on the label) **

* Actual Revision number can vary.

** Serial number is device specific. It is printed on the label attached to the Gefran LM-C transducer case.

The LSS master sends this message sequence to switch the Gefran LM-C CANopen device into configuration state (the slave sends the response message):

Table 13. LSS switch state selective message sequence

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
7E5h	Rx	8	40h	93h	00h	00h	00h	00h	00h	00h
7E5h	Rx	8	41h	52h	4Bh	35h	53h	00h	00h	00h
7E5h	Rx	8	42h	01h*	00h*	00h*	00h*	00h	00h	00h
7E5h	Rx	8	43h	31h**	5Fh**	51h**	01h**	00h	00h	00h
7E4h	Tx	8	44h	00h	00h	00h	00h	00h	00h	00h

* The Revision number used for this example is 00000001h

** The Serial number used for this example is: 01515F31h=22110001

The Serial Number is assigned by Gefran to the LM-C sensor in accordance with the following scheme:

SERIAL NUMBER: YY WW NNNN

where:

YY: year of production

WW: week of production

NNNN: progressive number inside the week, starting from 1.

5.2. LSS configuration services

ISS configure node-ID

By means of this service, the LSS master device configures the pending node-ID of the LSS slave device. The LSS slave device confirms the success or the failure of the service execution.

The allowed node-ID values are in the range 1..127 (01h..7Fh).

The LSS master sends this message to configure the value of the node-ID, the slave sends the response message:

Table 14. LSS configure node-ID message

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
7E5h	Rx	8	11h	Node ID	00h	00h	00h	00h	00h	00h
7E4h	Tx	8	11h	Error code	00h	00h	00h	00h	00h	00h

Error code can assume the values: 00h (Protocol successfully completed) or 01h (node-ID out of range).

The pending node-ID becomes active only after the master sends a NMT reset communication command. The node-ID is not automatically saved in the Flash memory of the slave device. In order to save the node-ID, refer to the LSS store configuration service. When the pending node-ID becomes active, or when the node-ID is stored in Flash memory, the following COB-IDs are automatically updated according to their default values:

- COB-ID SYNC (1005h)
- COB-ID EMCY (1014h)
- COB-ID SDO rx (1200h, sub 01h)
- COB-ID SDO tx (1200h, sub 02h)
- COB-ID TPDO (1800h, sub 01h)

LSS configure bit timing parameters

By means of this service, the LSS master device configures the pending bit rate of the LSS slave device. The LSS slave device confirms the success or the failure of the service execution. The allowed bit rate values with the associated table index, are specified in Table 15.

Table 15. Table index for bit timing table

Table index	Bit rate (kbit/s)
0	1000
1	Not Supported
2	500
3	250
4	125
5	Reserved
6	50
7	20
8	10

The LSS master sends this message to configure the bit rate, the slave sends the response message:

Table 16. LSS configure bit timing message

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
7E5h	Rx	8	13h	00h	Table index	00h	00h	00h	00h	00h
7E4h	Tx	8	13h	Error code	00h	00h	00h	00h	00h	00h

Error code can assume the values: 00h (Protocol successfully completed) or 01h (Bit timing not supported).

The pending bit rate becomes active only after the master sends the LSS activate bit timing parameter service, or with the next power-on after the execution of the LSS store configuration service.

The bit rate is not automatically saved to the Flash memory of the slave device. In order to save the bit rate configuration, refer to the LSS store configuration service.

LSS activate bit timing parameters

By means of this service, the LSS master activates simultaneously the bit rate at the LSS communication interface of all CANopen devices in the network. Therefore, the reception of this command triggers at the LSS slave the copying process of the currently pending bit rate to the active bit rate.

The LSS master sends this message to activate the bit timing parameters, the slave sends the response message:

Table 17. LSS activate bit timing parameters message

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
7E5h	Rx	8	15h	Switch delay		00h	00h	00h	00h	00h

The Switch delay parameter specifies the length of two delay periods of equal length, which are necessary to avoid operating the network with different bit rates.

After “Switch delay” has elapsed the first time after service indication, the slave device stops communicating on the bus. After “Switch delay” has elapsed one more time, the slave device resumes the communication on the bus using the new active bit rate.

LSS store configuration

By means of this service, the LSS master device requests the LSS slave device to store the configured local layer settings (node-ID and bit rate) to non-volatile memory. On execution of this command the pending node-ID and bit rate are copied to the persistent node-ID and bit rate.

The LSS master sends this message to store the LSS configuration, the slave sends the response message:

Table 18. LSS store configuration message

COB-ID	Rx/Tx	DLC	Data							
			D0	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**
7E5h	Rx	8	17h	00h	00h	00h	00h	00h	00h	00h
7E4h	Tx	8	17h	Error code	00h	00h	00h	00h	00h	00h

Error code can assume the values: 00h (Protocol successfully completed) or 02h (Storage media access error).

5.3. LSS inquiry services

ISS inquire node-ID

By means of this service, the LSS master device inquires the active node-ID of the LSS slave device that is in LSS configuration state. The LSS slave device responds indicating his active node-ID.

The LSS master sends this message to inquire the node-ID, the slave sends the response message:

Table 19. Table 19 - LSS inquire node-ID message

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
7E5h	Rx	8	5Eh	00h	00h	00h	00h	00h	00h	00h
7E4h	Tx	8	5Eh	Node ID	00h	00h	00h	00h	00h	00h

LSS inquire LSS address

By means of this service, the LSS master device inquires the LSS address of the LSS slave device. The LSS slave device responds indicating his LSS address.

The LSS master sends this message to inquire the Vendor-ID, the slave sends the response message:

Table 20. LSS inquire identity Vendor-ID message

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
7E5h	Rx	8	5Ah	00h	00h	00h	00h	00h	00h	00h
7E4h	Tx	8	5Ah	Vendor ID				00h	00h	00h

The LSS master sends this message to inquire the Product-code, the slave sends the response message:

Table 21. TLSS inquire identity Product-code message

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
7E5h	Rx	8	5Bh	00h	00h	00h	00h	00h	00h	00h
7E4h	Tx	8	5Bh	Product code				00h	00h	00h

where Product-code is the LSS slave's identity Product-code (little-endian format byte ordering).

The LSS master sends this message to inquire the Revision number, the slave sends the response message:

Table 22. Table 22 - LSS inquire identity Revision number message

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
7E5h	Rx	8	5Ch	00h	00h	00h	00h	00h	00h	00h
7E4h	Tx	8	5Ch	Revision number				00h	00h	00h

where Revision number is the LSS slave's identity Revision number (little-endian format byte ordering).

The LSS master sends this message to inquire the Serial number, the slave sends the response message:

Table 23. LSS inquire identity Serial number message

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
7E5h	Rx	8	5Dh	00h	00h	00h	00h	00h	00h	00h
7E4h	Tx	8	5Dh	Serial number				00h	00h	00h

where Serial number is the LSS slave's identity Serial number (little-endian format byte ordering).

6. SDO SERVICES

Service Data Object (S.D.O.) service provides direct access to the object entries of a CANopen device's object dictionary. The device initiating the SDO transfer is called the SDO client. The CANopen device hosting the accessed object dictionary is called the SDO server.

6.1. SDO download

The SDO client uses this service for transferring data to the object dictionary of the SDO server. SDO download service is therefore used to configure (write) communication, device and manufacturer parameters of the Gefran LM-C CANopen sensor.

The client (master) sends the download message:

Table 24. Table 24 - Structure of SDO download request by the Master

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + node-ID	Rx	8	Cs	Index		Sub index	Data			

Where:

- Cs is the command specifier of the SDO download request and its value depends on the number of bytes of Data field, following the description of Table 25.
- Data is the data to write in the object dictionary parameter (little-endian format byte ordering).
- Index is the object dictionary parameter index (little-endian format byte ordering).
- Sub index is the object dictionary parameter sub-index.

Table 25. Command specifier for download messages

Cs	Data byte length
2Fh	1 byte of Data
2Bh	2 bytes of Data
27h	3 bytes of Data
23h	4 bytes of Data

The server (sensor) answers to the download message:

Table 26. Structure of SDO download answer by the slave (sensor)

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
580h + node-ID	Tx	8	Res	Index		Sub index	00h	00h	00h	00h

Where Res field determines the correct/incorrect answer of the slave, as indicated in Table 27.

Table 27. SDO download: Res field of slave answer

Res	Description
60h	Data sent successfully
80h	Error

6.2. SDO upload

The SDO client uses this service to transfer the data from the slave, the sensor, to the master. SDO upload service is therefore used to check (read) communication, device, and manufacturer parameters of the GEFRAN LM-C CANopen device. The client (master) sends the download message:

Table 28. Structure of SDO upload request by the Master

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + node-ID	Tx	8	40h	Index		Sub index	00h	00h	00h	00h

The server (sensor) answers to the download message:

Table 29. Structure of SDO answer by the slave (sensor)

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
580h + node-ID	Rx	8	Res	Index		Sub index	Data			

Where Res field determines the correct/incorrect answer of the slave, as indicated in Table 30.

Table 30. SDO upload: Res field of slave answer

Res	Description
4Fh	1 byte of Data
4Bh	2 bytes of Data
47h	3 bytes of Data
43h	4 bytes of Data
80h	Error

The error code indicates an SDO download or SDO upload abort, the following table contains the abort codes provided by the protocol SDO abort transfer of the Gefran LM-C CANopen device.

Table 31. SDO abort codes

Abort code	Description
05040001h	Client/server command specifier not valid or unknown
05040005h	Out of memory
06010000h	Unsupported access to object
06010001h	Attempt to read a write only object
06010002h	Attempt to write a read only object
06020000h	Object does not exist in the object dictionary
06040041h	Object cannot be mapped to the PDO
06040042h	Mapping not correct (number of bytes mismatch)
06060000h	Access failed due to a hardware error
06070010h	Data type does not match, length of service parameter does not match
06090011h	Sub-index does not exist
06090030h	Invalid value for parameter - download only
08000020h	Data cannot be transferred or stored to the application
08000022h	Requested action can currently not be performed

6.3. Object dictionary

The object dictionary of the Gefran LM-C CANopen device, specified in the following tables, is composed of communication, device and manufacturer profiles.

Communication Profile Area

Index	Sub index	Name	Type	Access	Default value	Comment
1000h	00h	Device type	Unsigned32	RO	FFFF0196h	Multiple logical device with ds406 device profile as the first logical device
1001h	00h	Error register	Unsigned8	RO	-	00h: No error 01h: Error Occurs
1003h	00h	Number of Errors	Unsigned32	RO	-	Number of Errors Occurred, if different from zero, the relative subindex objects can be read.
1005h	00h	COB-ID SYNC	Unsigned32	RW	00000080h	Configured COB-ID of the synchronization object (SYNC)
1008h	00h	Manufacturer device name	Visible string	RO	Twiist LM-C	Name of the device
100Ch	00h	Guard Time	Unsigned16	RW	0	The value shall be given in multiple of ms, the 0 value disable the life guarding
100Dh	00h	Life Time Factor	Unsigned8	RW	0	The life time factor multiplied with the guard time gives the life time for the life guarding protocol
1010h	00h	Store parameters	Unsigned8	RO	4	Highest sub-index supported
	01h		Unsigned32	RW	00000001h	Writing the signature “evas” stores all parameters in retentive memory
	02h		Unsigned32	RW	00000001h	Writing the signature “evas” stores communication parameters in retentive memory
	03h		Unsigned32	RW	00000001h	Writing the signature “evas” stores application parameters in retentive memory
	04h		Unsigned32	RW	00000001h	Writing the signature “evas” stores manufacturer parameters in retentive memory
1010h	00h	Restore default parameters	Unsigned8	RO	4	Highest sub-index supported
	01h		Unsigned32	RW	00000001h	Writing the signature “daol” reset all parameters to factory defaults
	02h		Unsigned32	RW	00000001h	Writing the signature “daol” reset communication parameters to factory defaults
	03h		Unsigned32	RW	00000001h	Writing the signature “daol” reset application parameters to factory defaults
	04h		Unsigned32	RW	00000001h	Writing the signature “daol” reset manufacturer parameters to factory defaults
1014h	00h	COB-ID EMCY	Unsigned32	RW	00000080h + Node-ID	Configured COB-ID for the EMCY write service
1015h	00h	Inhibit time EMCY	Unsigned16	RW	0000h	Configured inhibit time for the EMCY service
1017h	00h	Producer heartbeat time	Unsigned16	RW	0000h	Configured cycle time of the heartbeat (ms)

Index	Sub index	Name	Type	Access	Default value	Comment
1018h	00h	Identity object	Unsigned8	RO	4	Highest sub-index supported
	01h		Unsigned32	RO	00000093h	Vendor-ID
	02h		Unsigned32	RO	00434D4Ch	Product code
	03h		Unsigned32	RO	-	Revision number
	04h		Unsigned32	RO	-	Serial number (Indicated on sensor label)
1029h	00h	Error Behaviour	Unsigned8	RO	2	Highest sub-index supported
	01h		Unsigned32	RW	0	Communication Error
	02h		Unsigned32	RW	0	Specific Error Class
1200h	00h	SDO1 server parameter	Unsigned8	RO	2	Highest sub-index supported
	01h		Unsigned32	RO	00000600h + Node-ID	COB-ID client --> server (rx)
	02h		Unsigned32	RO	00000580h + Node-ID	COB-ID server <-- client (tx)
1800h	00h	TPDO1 communication parameter	Unsigned8	RO	5	Highest sub-index supported
	01h		Unsigned32	RW	00000180h + Node-ID	COB-ID of the TPDO1
	02h		Unsigned8	RW	FEh	Transmission type
	03h		Unsigned16	RW	0	Inhibit Time
	05h		Unsigned16	RW	0004h	Event-timer
1800h	00h	TPDO2 communication parameter	Unsigned8	RO	5	Highest sub-index supported
	01h		Unsigned32	RW	00000280h + Node-ID	COB-ID of the TPDO2
	02h		Unsigned8	RW	FEh	Transmission type
	03h		Unsigned16	RW	0	Inhibit Time
	05h		Unsigned16	RW	000Ch	Event-timer
1800h	00h	TPDO3 communication parameter*	Unsigned8	RO	5	Highest sub-index supported
	01h		Unsigned32	RW	80000380h + Node-ID	COB-ID of the TPDO3
	02h		Unsigned8	RW	FEh	Transmission type
	03h		Unsigned16	RW	0	Inhibit Time
	05h		Unsigned16	RW	000h	Event-timer
1800h	00h	TPDO4 communication parameter*	Unsigned8	RO	5	Highest sub-index supported
	01h		Unsigned32	RW	80000480h + Node-ID	COB-ID of the TPDO4
	02h		Unsigned8	RW	FEh	Transmission type
	03h		Unsigned16	RW	0	Inhibit Time
	05h		Unsigned16	RW	0004h	Event-timer
1A00h	00h	TPDO1 mapping parameter	Unsigned8	RW	2	Number of mapped application objects in TPDO1
	01h		Unsigned32	RW	60200120h	1st application object (position)
	02h		Unsigned32	RW	60300110h	2nd application object (speed)
	03h		Unsigned32	RW	-	3rd application object (null)
	04h		Unsigned32	RW	-	4th application object (null)
1A01h	00h	TPDO2 mapping parameter	Unsigned8	RW	3	Number of mapped application objects in TPDO2
	01h		Unsigned32	RW	21000010h	Device temperature
	02h		Unsigned32	RW	68100010h	Slope long 16
	03h		Unsigned32	RW	68200010h	Slope lateral 16
	04h		Unsigned32	RW	-	4th application object (null)

Index	Sub index	Name	Type	Access	Default value	Comment
1A02h	00h	TPDO3 mapping parameter*	Unsigned8	RW	3	Number of mapped application objects in TPDO3
	01h		Unsigned32	RW	21040110h	Acceleration x axis (in 1/4096 g/LSB)
	02h		Unsigned32	RW	21040210h	Acceleration y axis (in 1/4096 g/LSB)
	03h		Unsigned32	RW	21040310h	Acceleration z axis (in 1/4096 g/LSB)
	04h		Unsigned32	RW	-	4th application object (null)
1A03h	00h	TPDO4 mapping parameter*	Unsigned8	RW	2	Number of mapped application objects in TPDO4
	01h		Unsigned32	RW	60200220h	Position value channel 2
	02h		Unsigned32	RW	60300210h	Speed value channel 2
	03h		Unsigned32	RW	-	Angular rate z axis (in 7/800 deg/s/LSB)
	04h		Unsigned32	RW	-	4th application object (null)

* The TPDO3 and TPDO4 are set disabled as default

Manufacturer Profile Area

Index	Sub index	Name	Type	Access	Default value	Comment
2000h	00h	Number of channels	Unsigned8	RO	2	Number of channels for linear position
2001h	00h	Special Execution TAG	String	RO	0	Tag for special execution (e.g.: “1AX”)
2002h	00h	User Device Name	String	RW	0	USER_DEVICE_NAME: User string for device name
2010h	00h	Sensor Parameters	Unsigned8	RO	03h	Highest sub-index supported
	01h		Unsigned8	RW	00h	Auto-operational mode: 00h: Disabled 01h: After boot-up the device enters the NMT Operational state automatically
	02h		Unsigned8	RW	00h	Position - First Order IIR Filter: 00h: Disable From 01h to 19h: Time Constant expressed in multiples of 10ms
	03h		Unsigned8	RW	00h	TILT - First Order IIR Filter: 00h: Disable From 01h to 19h: Time Constant expressed in multiples of 10ms
	04h		Unsigned8	RW	00h	TILT - OUTPUT_TYPE 0: ENU (xyz) 1: NED (yx-z)
2011h	00h	Sensor Diagnostic	Unsigned8	RO	06h	Highest sub-index supported
	01h		Integer32	RW	-	USER_MIN_POS: triggers alarm/ EMCY if RAW_POS_CH1 or RAW_ POS_CH2 is lower than USER_MIN_ POS (μm)
	02h		Integer32	RW	-	USER_MAX_POS: triggers alarm/ EMCY if RAW_POS_CH1 or RAW_ POS_CH2 is greater than USER_ MAX_POS (μm)
	03h		Unsigned8	RW	00h	ENABLE_ERROR_POSITION: If an error occurs, the data position (CH1 and CH2) is forced to the ERROR_ POSITION value. 0: Disabled 1: Enabled
	04h		Integer32	RW	00h	ERROR_POSITION: Position value read in case of error (valid only if ENABLE_ERROR_ POSITION is set enabled)
	05h		Unsigned8	RW	00h	ENABLE_ERROR_ GYRACC: if an error occurs, the gyro and accelerometer data are forced to the ERROR_GYRACC value. 0: Disabled 1: Enabled
	06h		Integer16	RW	00h	ERROR_ GYRACC: Gyro and accelerometer value read in case of error (valid only if ENABLE_ERROR_ GYRACC is set enabled)
2100h	00h	Temperature Sensor	Integer16	RO	-	TEMPERATURE: Internal temperature sensor (in 0.1 °C/LSB)

Index	Sub index	Name	Type	Access	Default value	Comment
2101h	00h	Raw Position Data	Unsigned8	RO	02h	Highest sub-index supported
	01h		Integer32	RO	-	RAW_POS_CH1: raw position value from the first position primary element (CH1) (μm)
	02h		Integer32	RO	-	RAW_POS_CH2: raw position value from the second position primary element (CH2) (μm)
2102h	00h	Raw Euler Angles	Unsigned8	RO	03h	Highest sub-index supported
	01h		Integer16	RO	-	YAW: Euler angle YAW (0.1 deg/LSB)
	02h		Integer16	RO	-	ROLL: Euler angle Roll (0.1 deg/LSB)
	03h		Integer16	RO	-	PITCH: Euler angle Pitch (0.1 deg/ LSB)
2103h	00h	Raw Quaternions	Unsigned8	RO	04h	Highest sub-index supported
	01h		Integer16	RO	-	QUAT_W: Quaternion scalar part w (in 1/30000)
	02h		Integer16	RO	-	QUAT_X: Quaternion vector part x (in 1/30000)
	03h		Integer16	RO	-	QUAT_Y: Quaternion vector part y (in 1/30000)
	04h		Integer16	RO	-	QUAT_Z: Quaternion vector part z (in 1/30000)
2104h	00h	Accelerometers -Raw Data	Unsigned8	RO	03h	Highest sub-index supported
	01h		Integer16	RO	-	OUTX_A: Raw data output accelerometer sensor x axis (in 1/4096 g/LSB)
	02h		Integer16	RO	-	OUTY_A: Raw data output accelerometer sensor y axis (in 1/4096 g/LSB)
	03h		Integer16	RO	-	OUTZ_A: Raw data output accelerometer sensor z axis (in 1/4096 g/LSB)
2105h	00h	Gyroscope Raw Angles (xyz)	Unsigned8	RO	3	Highest sub-index supported
	01h		Integer16	RO	-	OUTX_G: Raw data output angular rate sensor x axis (in 1/16 dps/LSB)
	02h		Integer16	RO	-	OUTY_G: Raw data output angular rate sensor y axis (in 1/16 dps/LSB)
	03h		Integer16	RO	-	OUTZ_G: Raw data output angular rate sensor z axis (in 1/16 dps/LSB)
3000h	00h	Program execution	Unsigned16	RW	01h	Program Execution: Default value 01h (application execution) Write FFh to switch to bootloader execution

Note: The objects in blooded can be mapped in a TPDO object.

Device Profile Area

Index	Sub index	Name	Type	Access	Default value	Commet
6000h	00h	Operating parameters	Unsigned16	RW	4	Configuration of the operating parameters of the encoder: Bit2 sfc: Scaling function control Bit 3 md: measuring direction
6002h	00h	Total Measuring range in measuring units	Unsigned32	RO	-	Nominal span [um]
6005h	00h	Linear encoder measuring step settings	Unsigned8	RO	2	Highest sub-index supported
	01h		Unsigned32	RW	1000	Position measuring step given in multiples of 0,001 µm Minimum value 1000 (1 µm)
	02h		Unsigned32	RW	10	Speed measuring step given in multiples of 0,01mm/s Minimum value 10 (0.1 mm/s)
6010h	00h	Preset values for multi-sensor devices	Unsigned8	RO	2	Highest sub-index supported
	01h		Integer32	RW	-	Preset value channel 1
	02h		Integer32	RW	-	Preset value channel 2
6020h	00h	Position values for multi-sensor devices	Unsigned8	RO	2	Highest sub-index supported
	01h		Integer32	RO	-	Position value channel 1
	02h		Integer32	RO	-	Position value channel 2
6030h	00h	Speed value	Unsigned8	RO	2	Highest sub-index supported
	01h		Integer16	RO	-	Speed value channel 1
	02h		Integer16	RO	-	Speed value channel 2
6500h	00h	Operating status	Unsigned16	RO	-	Operating status of the encoder functions configured in the object 6000h
6501h	00h	Measuring step	Unsigned32	RO	1000	Position measuring step given in multiples of 0,001µm
6502h	00h	Number of distinguishable revolutions	Unsigned32	RO	0	number of distinguishable revolutions that the encoder is able to output.
650Ch	00h	Offset values for multi-sensor devices	Unsigned8	RO	2	Highest sub-index supported
	01h		Integer32	RO	-	Offset value channel 1
	02h		Integer32	RO	-	Offset value channel 2
650Eh	00h	Device capability	Unsigned16	RO	01h	Encoder class: 001 class 1 Resolution: 0 normal Safety: 0 not supported
67FFh	00h	Device type	Unsigned32	RO	7080196h	Multi-sensor encoder interface with DS 406 device profile
6800h	00h	Resolution	Unsigned16	RW	0x0064	Resolution of Slope long16 (object 6810h) and Slope lateral16 (object 6820h) objects based on 0.001 deg.
6810h	00h	Slope long16	Integer16	RO	-	Slope value of the longitudinal axis (ROLL) with the resolution given in object 6800h.

Index	Sub index	Name	Type	Access	Default value	Commet
6811h	00h	Slope long16 operating parameter	Unsigned8	RW	0x02	If scaling is enabled, the Slope long16 value shall be calculated accordingly to the following equation: Slope long16 = physically measured angle + Differential slope long16 offset + Slope long16 offset If scaling is disabled, the Slope long16 value shall be equal to the physical measured angle.
6812h	00h	Slope long16 preset value	Integer16	RW	0x0000	The preset for the longitudinal slope is calculated with respect to object 6814h with the resolution given in object 6800h.
6813h	00h	Slope long16 offset	Integer16	RW	0x0000	Application-offset of the longitudinal axis with the resolution given in object 6800h.
6814h	00h	Differential slope long16 offset	Integer16	RW	0x0000	This object shall shift the Slope long16 value (object 6810h) independent of Slope long16 preset value (object 6812h) and Slope long16 offset (object 6813h). The value shall be given in degree (angle) with the resolution given in object 6800h.
6820h	00h	Slope lateral16	Integer16	RO	-	Slope value of the lateral axis (PITCH) with the resolution given in object 6800h.
6821h	00h	Slope lateral16 operating parameter	Unsigned8	RW	0x02	If scaling is enabled, the Slope latrtal16 value shall be calculated accordingly to the following equation: Slope lateral16 = physically measured angle + Differential slope lateral16 offset + Slope lataral16 offset If scaling is disabled, the Slope lataral16 value shall be equal to the physical measured angle.
6822h	00h	Slope lataral16 preset value	Integer16	RW	0x0000	The preset for the lateral slope is calculated with respect to object 6814h with the resolution given in object 6800h.
6823h	00h	Slope lateral16 offset	Integer16	RW	0x0000	Application-offset of the lateral axis with the resolution given in object 6800h.
6824h	00h	Differential slope lateral16 offset	Integer16	RW	0x0000	This object shall shift the Slope lateral16 value (object 6820h) independent of Slope lateral16 preset value (object 6822h) and Slope lateral16 offset (object 6823h). The value shall be given in degree (angle) with the resolution given in object 6800h.
6FFFh	00h	Device type	Unsigned32	RO	0002019Ah	Inclinometer class C2 with DS 410 device profile

7. PDO SERVICES

The real-time data transfer is performed by means of "Process Data Objects (PDO)".

Data type and mapping of application objects into PDO is determined by a corresponding default PDO mapping structure within the object dictionary. In particular, mapping parameters of PDO1, PDO2, PDO3 and PDO4 are set in object 1A00h, 1A01h, 1A02h and 1A03h respectively.

Communication parameters of PDOs, as COB-ID, transmission mode and transmission rate, are also specified in the object dictionary. In particular, communication parameters of PDO1, PDO2, PDO3 and PDO4 are set in object 1800h, 1801h, 1802h and 1803h respectively.

7.1. PDO messages format

Gefran LM-C presents four different PDO, mapped as described from Table 32 to Table 35. In operational state, each PDO will be transmitted with its event timer as described in PDO communication parameter (sub-index 05h of object 1800h, 1801h, 1802h and 1803h for PDO1, PDO2, PDO3 and PDO4 respectively).

Table 32. Transmit PDO1 message format

COB-ID	Rx/Tx	DLC	Data					
			D0	D1	D2	D3	D4	D5
180h + Node-ID	Tx	6	Pos CH1 LSB	Pos CH1	Pos CH1	Pos CH1 MSB	Speed Ch1 LSB	Speed Ch1 MSB

Table 33. Transmit PDO2 message format

COB-ID	Rx/Tx	DLC	Data					
			D0	D1	D2	D3	D4	D5
280h + Node-ID	Tx	5	temp	Roll LSB	Roll MSB	Pitch LSB	Pitch MSB	-

Table 34. Transmit PDO3 message format

COB-ID	Rx/Tx	DLC	Data					
			D0	D1	D2	D3	D4	D5
380h + Node-ID	Tx	6	Acceleration X LSB	Acceleration X MSB	Acceleration Y LSB	Acceleration Y MSB	Acceleration Z LSB	Acceleration Z MSB

Table 35. Transmit PDO4 message format

COB-ID	Rx/Tx	DLC	Data					
			D0	D1	D2	D3	D4	D5
480h + Node-ID	Tx	6	Pos CH2 LSB	Pos CH2	Pos CH2	Pos CH2 MSB	Speed Ch2 LSB	Speed Ch2 MSB

Note: The TPDO3 and TPDO4 are set disabled as default.

7.2. PDO mapping editing

PDO re-mapping is supported by the Gefran LM-C CANopen sensor. Object supported for PDO mapping are listed in Table 36. Depending on object length (number of bytes), up to 4 objects can be mapped for each TPDO considering maximum PDO length is 8 bytes.

In accordance with standard defined by CiA, PDO mapping can be changed from the standard following the procedure:

- Destroy TPDO by setting bit valid to 1b of sub-index 01h of the according TPDO communication parameter.
- Disable mapping by setting sub-index 00h to 00h value.
- Modify mapping by changing the values of the corresponding sub-indices.
- Enable mapping by setting sub-index 00h to the number mapped objects.
- Create TPDO by setting bit valid to 0b of sub-index 01h of the according TPDO.

Table 36. Object supporting PDO mapping

Value	Description
21020110h	Raw Euler angle Pitch (0.1 deg)
21020210h	Raw Euler angle Roll (0.1 deg)
21020310h	Raw Euler angle Yaw (0.1 deg)
21030110h	Quaternion scalar part w (in 1/30000)
21030210h	Quaternion scalar part x (in 1/30000)
21030310h	Quaternion scalar part y (in 1/30000)
21030410h	Quaternion scalar part z (in 1/30000)
21040110h	Raw data output accelerometer sensor x axis (1/4096 g)
21040210h	Raw data output accelerometer sensor y axis (1/4096 g)
21040310h	Raw data output accelerometer sensor z axis (1/4096 g)
21050110h	Raw data output angular rate sensor x axis (1/16 deg/s)
21050210h	Raw data output angular rate sensor y axis (1/16 deg/s)
21050310h	Raw data output angular rate sensor z axis (1/16 deg/s)
21000010h	Temperature (0.1 °C)
21010120h	Raw position value from the first position primary element (CH1) (μm)
21010220h	Raw position value from the second position primary element (CH2) (μm)
60200120h	Position value channel 1
60300110h	Speed value channel 1
60200220h	Position value channel 2
60300210h	Speed value channel 2
68100010h	Slope long 16
68200010h	Slope lateral 16

8. SYNC SERVICES

The SYNC object can be broadcasted periodically by the SYNC producer. The SYNC object provides the basic network synchronization mechanism.

If the CANopen devices operates synchronously (see object 1800, sub-index 2), it uses the SYNC object to synchronize its own timing, as the PDO transmission, with that of the synchronization object producer.

The format of the SYNC object is explained in Table 37.

Table 37. SYNC message format

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
80h	Rx	0	-	-	-	-	-	-	-	-

9. EMCY SERVICES

Emergency objects are triggered by the occurrence of the CANopen device internal error situation. An emergency object is transmitted only once per 'error event'. No further emergency objects are transmitted as long as no new error types occur on the CANopen device. If one or more error conditions change, the CANopen device transmits the emergency object with the updated error code. The error register value inside the EMCY object is also updated.

For the Gefran LM-C CANopen sensor two types of error conditions are defined: device hardware error and Data set error.

The possible EMCY error codes are shown in Table 38.

Table 38. EMCY error codes for the LM-C CANopen device

Error code	Description
0000h	Error reset or no error
5062h	Multiple sensor elements failure (CiA 406)
6200h	Out of Range Error (over travel)
6300h	Data set
FF03h	Longitudinal sensor and Lateral sensor Error (CiA 410)

The format of the EMCY message is explained in Table 39. About the content of the error register see the description of relative object (Error register, 1001h).

Table 39. EMCY message format

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
80h + Node-ID	Tx	8	EMCY error code LSB	EMCY error code MSB	Error register (1001h)	00h	00h	00h	00h	00h

To understand the error cause, user can also perform a SDO upload of manufacture status register (1002h).

10. BOOTLOADER SERVICE

Bootloader functionality allows to upgrade firmware of the Gefran LM-C sensor; this document describes the functionality of the CANopen Bootloader protocol stack and the interaction between the CANopen Bootloader and the CANopen Application of LM-C sensor.

Bootloader provides functionality for the CANopen standards CiA 301, CiA 302 and CiA 305, the following services are disabled:

- EMCY producer
- LSS server

10.1. Bootloader object dictionary

SDO services provide direct access to the object entries of a CANopen Bootloader object dictionary.

In compliance with CiA 302-3, the following bootloader specific object dictionary entries are implemented.

Index	Sub index	Name	Type	Access	Default value	Comment
100h	00h	Device Profile	Unsigned32	RO	0000 0000h	Since the specification CiA 302-2 is not a device profile, any other value than 0 is not allowed.
1001h	00h	Error register	Unsigned8	RO	-	Error register: Bit 0: generic error Bit 1: Current Bit 2: Voltage Bit 3: Temperature Bit 4: Communication error Bit 5: Device profile specific Bit 6: Reserved Bit 7: Manufacturer-specific
1002h	00h	Manufacturer status	Unsigned32	RO	-	Common status register for manufacturer-specific purposes
1008h	00h	Manufacturer device name	Visible String	RO	Boot	Name of the device
1009h	00h	Manufacturer hardware version	Visible String	RO	-	Hardware version description
100Ah	00h	Manufacturer firmware version	Visible String	RO	-	Software version description
1014h	00h	COB-ID emergency	Unsigned32	RW	00000080h + Node-ID	Configured COB-ID for the EMCY write service
1017h	00h	Heartbeat producer time	Unsigned16	RW	0	Configured cycle time of the heartbeat (ms)
1018h	00h	Identity object: Highest sub-index	Unsigned8	RO	4	Highest sub-index supported
	01h	Vendor ID	Unsigned32	RO	00000093h	Vendor-ID
	02h	Product Code	Unsigned32	RO	00434D4Ch	Product code
	03h	Revision Number	Unsigned32	RO	-	Revision number
	04h	Serial Number	Unsigned32	RO	-	Serial number
1F50h	00h	Program data: Highest sub-index	Unsigned8	RO	1	This object provides the download to the CANopen device. If the download of a program fails for any reason the transfer shall be responded with the SDO abort message. The SDO abort code 0606 0000h shall indicate any Flash memory related error
	01h	Program Data 1	Domain	WO	-	

Index	Sub index	Name	Type	Access	Default value	Comment
1F51h	00h	Program control: Highest sub-index	Unsigned8	RO	1	This object allows the control of the programs downloaded to the CANopen device: 00h Stop program 01h Start program 02h Reset program 03h Clear program 04h…7Fh Reserved 80h…FFh Reserved The SDO abort code 0609 0030h shall indicate a not supported action. The SDO abort code 0800 0022h shall indicate that a requested action can currently not be performed
	01h	Program Control 1	Unsigned8	RW	00h	
1F56h	00h	Program software identification: Highest sub-index	Unsigned8	RO	1	This object provides a unique identification per the application program software. The calculation method is a checksum over the Flash memory reserved to the application code.
	01h	Program software identification 1	Unsigned32	RO	-	
1F57h	00h	Flash status identification: Highest sub-index	Unsigned8	RO	1	This object provides the current Flash memory status: 00h Status OK 01h Flash operations in progress 02h No valid application available
	01h	Flash status identification 1	Unsigned32	RO	00000000h	

10.2. Firmware update

Bootloader functionality allows to upgrade firmware of the Gefran LM-C sensor.

This document describes the functionality of the CANopen Bootloader and the interaction between the CANopen Bootloader and the CANopen Application of LM-C sensor.

At the power on of the sensor, the Bootloader firmware code is executed; the Bootloader is silent and the execution is autonomously switched to the Application LM-C code.

Figure 19 describes the flow chart execution of the complete update process and the interactions between the CANopen Bootloader and the LM-C Application.

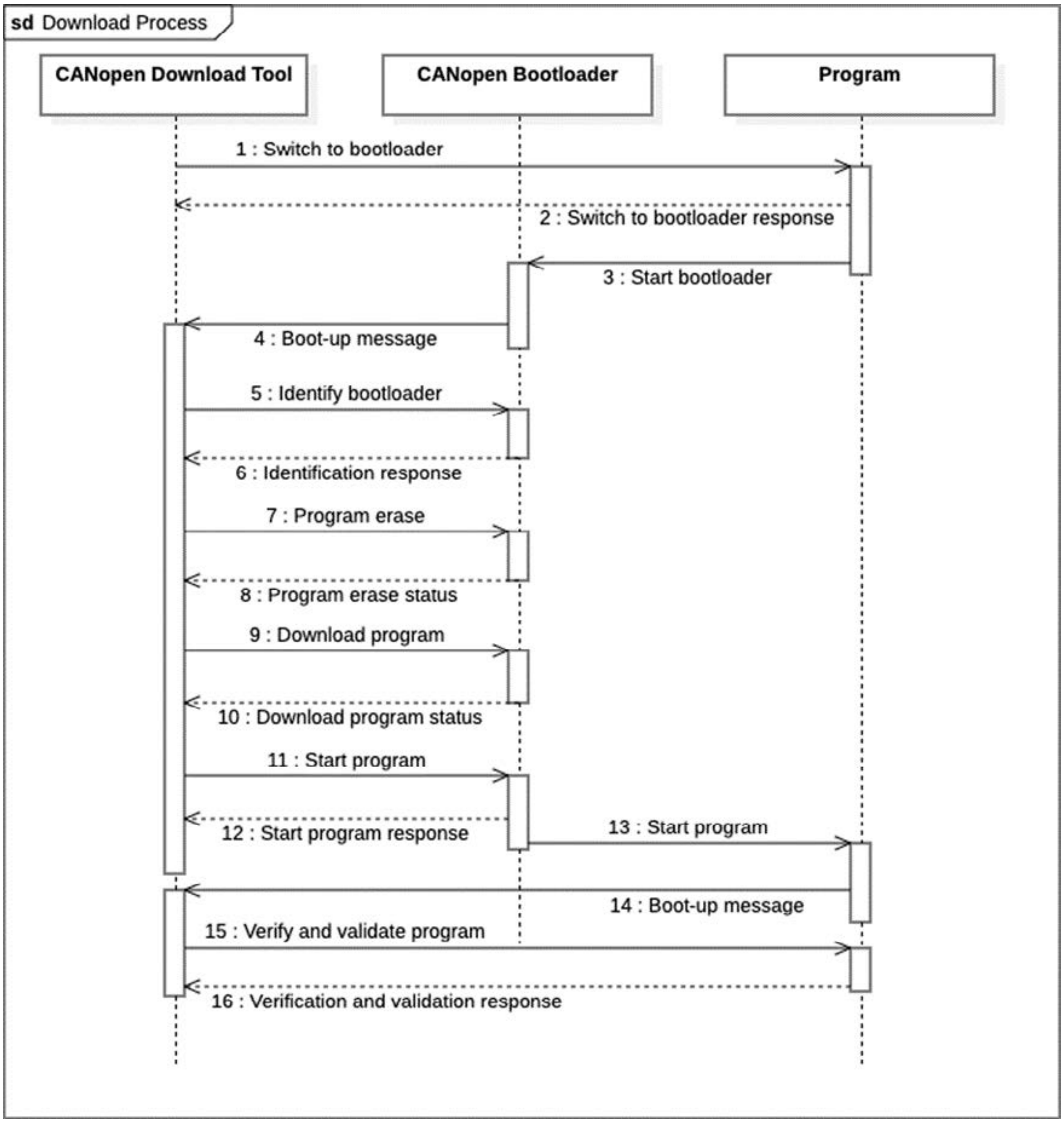

Figure 19. Firmware update flow chart

Bootloader activation

At power on, a boot up message from the application code is received, so the user is totally unconscious of the presence of the Bootloader.

To update the firmware and download a new application code, it is necessary to transmit a SDO request to a manufacture specific object (3000h, as described in 4.2), to switch to Bootloader execution.

After receiving the boot-up message from the Application, transmit an SDO download request to object 3000h with value FFh to switch from Application execution to Bootloader execution. An example of the sequence of transmitted and received messages is showed in Trace 5.

Note: In the following Traces SDO requests and response are reported, considering Node-ID equal to 7Fh. Please consider that standard Node-ID of LM-C sensor is 7Fh, but it can be changed be customer through LSS services as described in the User Manual and as defined by CiA (CAN in Automation, DS 305).

ID (hex)	DLC	Data (hex)	Comment	
77F	1	00	Boot-up message from Application	
67F	8	2B 00 30 00 FF 00 00 00	Write Execution	
5FF	8	60 00 30 00 00 00 00 00	Correct response	Note 0
77F	1	00	Boot-up message from Bootloader	

Trace 5. Bootloader activation

Bootloader identification (steps 5 – 6)

After reception of a boot-up message the identification of the CANopen Bootloader shall be executed.

Following actions can be performed to identify bootloader execution and verify the absence of errors:

- Verify transmission of Boot-up message from Bootloader.
- SDO upload request to object 1000h to verify the Device Profile. Value for Bootloader execution is 0000 0000h.
- SDO upload request to object 1001h to verify the Error Register. Value 00h indicates no errors.
- SDO upload request to object 1008h to verify the Manufacture Device Name.
- SDO upload request to object 1018h 01h to verify the Vendor ID.
- SDO upload request to object 1008h 02h to verify the Product Code.
- SDO upload request to object 1008h 03h to verify the Revision Number.
- SDO upload request to object 1008h 04h to verify the Serial Number.

Example of messages to transmit and receive in Trace 6.

ID (hex)	DLC	Data (hex)	Comment
77F	1	00	Boot-up message from Bootloader
67F	8	40 00 10 00 00 00 00 00	Read Device Profile
5FF	8	43 00 10 00 00 00 00 00	Response: 00 00 00 00h
67F	8	40 01 10 00 00 00 00 00	Read Error register
5FF	8	4F 01 10 00 00 00 00 00	Response: 00h
67F	8	40 08 10 00 00 00 00 00	Read Device Name
5FF	8	43 08 10 00 42 6F 6F 74	Response: Boot
67F	8	40 18 10 01 00 00 00 00	Read Vendor ID
5FF	8	43 18 10 01 93 00 00 00	Response: 93h
67F	8	40 18 10 02 00 00 00 00	Read Product Code
5FF	8	43 18 10 02 00 4C 4D 43	Response: “LM-C”
67F	8	40 18 10 03 00 00 00 00	Read Revision Number
5FF	8	43 18 10 03 XX XX XX XX	Response: XX XX XX XXh
67F	8	40 18 10 04 00 00 00 00	Read Serial Number
5FF	8	43 18 10 04 XX XX XX XX	Response: XX XX XX XXh

Trace 6. Bootloader identification

Device Profile values depends on which program is in execution, possible values:

Value	Description
00000000h	Bootloader
FFFF0196h	Application

Program erase (steps 7 – 8)

The program erase sequence is performed by writing the “clear program“ into the program control object 1F51h.

Afterwards, the Flash Status Identification value, object 1F57h, shall be repeatedly read until the response denotes the status “flash empty“ (corresponding to “No valid program available”) or any possible error code.

Example of messages to transmit and receive in Trace 7.

ID (hex)	DLC	Data (hex)	Comment
67F	8	22 51 1F 01 03 00 00 00	Write “Clear Program” in Program Control
57F	8	60 51 1F 01 XX XX XX XX	Response: XX XX XX XXh * Note1
67F	8	40 57 1F 01 00 00 00 00	Read Flash Status idenfification
57F	8	43 57 1F 01 00 00 00 00	Response: busy
		…	Repeat reading 1F57h
67F	8	40 57 1F 01 00 00 00 00	Read Flash Status idenfification
57F	8	43 57 1F 01 02 00 00 00	Response: flash empty

Trace 7. Program Erase

*Note 1: Value corresponding to value contained in object 1F50h 01h

Program download (steps 9 – 10)

The program download sequence shall be initiated by reading the flash status identification value in order to verify that the flash memory is empty.

Afterwards, the binary program file can downloaded to the device by means of the SDO block download protocol.

At the end of the code download read Flash Status Identification object 1F57h 01h until read value is equal to 00h (corresponding to Status OK).
Example of messages to transmit and receive in Trace 8.

ID (hex)	DLC	Data (hex)	Comment	
67F	8	40 57 1F 01 00 00 00 00	Read Flash Status idenfification	
5FF	8	43 57 1F 01 02 00 00 00	Response: flash empty	
67F	8	C2 50 1F 01 XX XX XX XX	Init block download	* Note 2
5FF	8	A0 50 1F 01 78 00 00 00	Init block download resp.	* Note 3
67F	8	01 XX XX XX XX XX XX XX	Write data, block 1	* Note 4
67F	8	02 XX XX XX XX XX XX XX	Write data, block 2	* Note 4
67F	8	03 XX XX XX XX XX XX XX	Write data, block 3	* Note 4
67F	8	…	…	
67F	8	78 XX XX XX XX XX XX XX	Write data, block 120	* Note 4
57F	8	A2 78 78 00 00 00 00 00	Received 78 of 78 blocks	* Note 5
67F	8	01 XX XX XX XX XX XX XX	Write data, block 1	* Note 4
67F	8	02 XX XX XX XX XX XX XX	Write data, block 2	* Note 4
67F	8	03 XX XX XX XX XX XX XX	Write data, block 3	* Note 4
67F	8	…	…	
67F	8	78 XX XX XX XX XX XX XX	Write data, block 120	* Note 4
57F	8	A2 78 78 00 00 00 00 00	Received 78 of 78 blocks	* Note 5
67F	8	01 XX XX XX XX XX XX XX	Write data, block 1	* Note 4
67F	8	02 XX XX XX XX XX XX XX	Write data, block 2	* Note 4
67F	8	03 XX XX XX XX XX XX XX	Write data, block 3	* Note 4
67F	8	…	…	
67F	8	NN XX XX XX XX XX XX XX	Write data, block NN	* Note 6
67F	8	XX 14 00 02 09 3D 00 01	Write data, final block	* Note 7
57F	8	A2 XX 78 00 00 00 00 00	Final block: response	* Note 8
67F	8	C1 00 00 00 00 00 00 00	Block download end	
57F	8	A1 00 00 00 00 00 00 00	Block download end response	
67F	8	40 57 1F 01 00 00 00 00	Read Flash Status idenfification	
57F	8	43 57 1F 01 01 00 00 00	Response: In progress	
		…	Repeat reading 1F57h	
67F	8	40 57 1F 01 00 00 00 00	Read Flash Status idenfification	
57F	8	43 57 1F 01 00 00 00 00	Response: Status OK	
67F	8	40 56 1F 01 00 00 00 00	Read Program software Identification	
57F	8	43 56 1F 01 XX XX XX XX	Response: CRC value	* Note 9

Trace 8. Program download

*Note 2: Value corresponding to length of code to download expressed in hex value. (Example: 107058 bytes, write 00 01 A2 32).

*Note 3: The received value indicates the maximum number of SDO block to transmit. (Example: 78h indicates 120 blocks can be transmitted).

*Note 4: Value corresponding to binary code to write in memory. (.bin file to download)

*Note 5: The received value indicates the number of SDO block received. (Example: 78h indicates 120 of 120 blocks received).

*Note 6: The NN value indicates the number of SDO block to transmit to complete the download minus one. (Example: value 35h indicates 53+1=54 of 120 blocks total blocks to transmit).

*Note 7: The XX value is given by the number of SDO transmitted and plus 80h. (Example: 54 blocks, write 36h+ 80h =B6h).

*Note 8: The NN value indicates the number of SDO block to transmit to complete the download. (Example: 36h indicates 54 of 120 blocks to transmit).

*Note 9: The value indicates the CRC value of the downloaded binary code and it has to be compared with the CRC calculated previously from the download tool. (Example: CRC= 2EBB2CB8h = 784018616 dec).

Program start (steps 11 – 14)

The program start sequence is initiated by writing the “start program“ value into the program control object 1F51h. The application program will send a boot-up message.

Example of messages to transmit and receive in Trace 9.

ID (hex)	DLC	Data (hex)	Comment	
67F	8	22 51 1F 01 01 00 00 00	Write “Start Program” in Program Control	
57F	8	60 51 1F 01 XX XX XX XX	Response: XX XX XX XXh	* Note 1
77F	1	00	Boot-up message from Application	

Trace 9. Program start

Program validation (steps 15 – 16)

Following steps shall be performed to validate the downloaded code:

- SDO upload request to object 1000h to identify Application execution. Value for Application execution is FFFF0196h.
- Transmit a SDO download request to object 5F01h 00h with the code “done” (656E6F64h) to mark the downloaded application as valid. An SDO correct answer should be received.
- Send a SDO download request to object 5F00h 00h with code FFh to jump to Bootloader execution.
- Wait for the Boot-up message.
- Read the Device Profile to verify the Bootloader execution is in on progress. Value for Application execution is 00000000h.
- Write a SDO download request to object 1F51 with code 80h to set the app as valid and set the Application auto-start at power-on. A correct SDO answer should be received.

Example of messages to transmit and receive in Trace 10.

ID (hex)	DLC	Data (hex)	Comment	
77F	1	00	Boot-up message from Application	
67F	8	40 00 10 00 00 00 00 00	Read Device Profile	
5FF	8	43 00 10 00 96 01 FF FF	Response: FF FF 01 96h	
67F	8	23 01 5F 00 64 6F 6E 65	Write Valid manufacturer password	
5FF	8	60 01 5F 00 00 00 00 00	Correct Response	
67F	8	2B 00 5F 00 FF 00 00 00	Write Execution	
5FF	8	60 00 5F 00 00 00 00 00	Correct response	
77F	1	00	Boot-up message from Application	
67F	8	40 00 10 00 00 00 00 00	Read Device Profile	
5FF	8	43 00 10 00 00 00 00 00	Response: 00 00 00 00h	
67F	8	22 51 1F 01 80 00 00 00	Write 80h in Program Control	
57F	8	60 51 1F 01 XX XX XX XX	Response: XX XX XX XXh	* Note 1

Trace 10. Program validation

Final verification

In order to verify all steps were performed correctly and the code was downloaded and fully functioning, next checks shall be performed:

- Swich off and on the power to perform the sensor reset.
- Wait for the boot-up.
- Verify the Node-ID, which should be 7Fh.
- Verify application is in execution by reading the Device profile, object with index 1000h.

Example of messages to transmit and receive in Trace 11.

ID (hex)	DLC	Data (hex)	Comment
77F	1	00	Boot-up message from Application
67F	8	40 00 10 00 00 00 00 00	Read Device Profile
5FF	8	43 00 10 00 96 01 FF FF	Response: FF FF 01 96h

Trace 11. Final verification

11. COMMUNICATION EXAMPLES

11.1. How to change the baud rate setting

In accordance with standard protocol defined by CiA, Gefran LM-C sensor Baud rate can be modified through LSS services; in chapter 3.2 an example of how change the baud rate setting from 250 kbit/s (default value) to 500 kbit/s is presented.

11.2. How to change the node-ID

In accordance with standard protocol defined by CiA, Gefran LM-C sensor node-ID can be modified through LSS services, in chapter 3.2 an example of how change the node-ID from 7Fh (standard value) to 7Eh is presented.

11.3. How to change the PDO rate (event timer)

In accordance with standard protocol defined by CiA, Gefran LM-C sensor PDO rate can be modified through SDO request (described in 6.1), following the described procedure.

In particular, to change the PDO1 event timer from 4 ms (standard value) to 10ms, write the new event timer in the object 1800h sub-index 05h by using the following SDO message:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + node-ID	Rx	8	2Bh	00h	18h	05h	0Ah	00h	00h	00h

The answers after successful storing you will receive is:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
580h + node-ID	Tx	8	60h	00h	18h	05h	00h	00h	00h	00h

It's also possible to permanently save the new communication parameters by sending the following SDO message:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + node-ID	Rx	8	23h	10h	10h	02h	73h	61h	76h	65h

The answers after successful storing you will receive is:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
580h + node-ID	Tx	8	60h	10h	10h	02h	00h	00h	00h	00h

To change the PDO2, PDO3 and PDO4 event timer from the standard, write the new event timer in the object 1801h, 1802h and 1803h sub-index 05h respectively.

11.4. How to activate/deactivate the automatic operational mode

To change the automatic NMT operational state after power on configuration and make the sensor automatically go in operational state after initialization, write the enable value in the object 2010h sub-index 01h (0=disable, 1=automatic enable) by using the following SDO message.

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + node-ID	Rx	8	2Fh	10h	20h	01h	01h	00h	00h	00h

The answers after successful storing you will receive is:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
580h + node-ID	Tx	8	60h	10h	20h	01h	00h	00h	00h	00h

It's also possible to permanently save the new customized parameters by sending the following SDO message:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + node-ID	Rx	8	23h	10h	10h	04h	73h	61h	76h	65h

The answers after successful storing you will receive is:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
580h + node-ID	Tx	8	60h	10h	10h	04h	00h	00h	00h	00h

11.5. How to change the position step setting (resolution)

In accordance with standard protocol defined by CiA, Gefran LM-C sensor resolution can be modified through SDO request (described in 6.1), following the described procedure.

In particular, to change the resolution from 1 μm (standard value) to 1 mm, write the new resolution in the object 6005h sub-index 01h by using the following SDO message:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + node-ID	Rx	8	23h	05h	60h	01h	40h	42h	0Fh	00h

The answers after successful storing you will receive is:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
580h + node-ID	Tx	8	60h	05h	60h	01h	00h	00h	00h	00h

It's also possible to permanently save the new application parameters by sending the following SDO message:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + node-ID	Rx	8	23h	10h	10h	03h	73h	61h	76h	65h

The answers after successful storing you will receive is:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
580h + node-ID	Tx	8	60h	10h	10h	03h	00h	00h	00h	00h

Remember that, as defined by CiA and as specified in the Object dictionary, position measuring step (object 6005h) is given in multiples of 0,001 µm. For this reason, writing F4240h = 1 000 000 dec means setting the following resolution:

1 000 000 + 0.001 µm=1 000 µm=1 mm

11.6. How to preset the zero-position value

In accordance with standard protocol defined by CiA, in Gefran LM-C sensor position can be preset through SDO request (described in 6.1), following the described procedure.

In particular, to preset the position of channel 1, which means setting a new zero position, move the sensor to the desired position and then write the value 00000000h in the relative preset object (object 6010h sub-index 01h) by using the following SDO message:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + node-ID	Rx	8	23h	10h	60h	01h	00h	00h	00h	00h

The answers after successful storing you will receive is:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
580h + node-ID	Tx	8	60h	10h	60h	01h	00h	00h	00h	00h

It's also possible to permanently save the new application parameters by sending the following SDO message:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + node-ID	Rx	8	23h	10h	10h	03h	73h	61h	76h	65h

The answers after successful storing you will receive is:

COB-ID	Rx/Tx	DLC	Data							
			D0	D1	D2	D3	D4	D5	D6	D7
580h + node-ID	Tx	8	60h	10h	10h	03h	00h	00h	00h	00h

When writing the preset object, the sensor will internally calculate the offset that guaranties output zero at the desired position and will automatically write it in offset object (object 650Ch sub-index 01h, relative to channel 1).

In the same way, to preset the position of channel 2 move the sensor to the desired position and then write the value 00000000h in the relative preset object (object 6010h sub-index 02h). The sensor will internally calculate the offset that guaranties output zero at the desired position and will automatically write it in offset object (object 650Ch sub-index 02h).

11.7. How to reset to Factory defaults

It's possible to easily restore the factory defaults by sending the following SDO message:

COB-ID	Rx/Tx	DLC	Dati							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + Node-ID	Rx	8	23h	11h	10h	XX	6Ch	6Fh	61h	64h

Where XX can have the following values:

- 01h: Restore all CAN objects to factory defaults
- 02h: Restore only Communication CAN objects to factory defaults (e.g. PDO parameters)
- 03h: Restore only Application CAN objects to factory defaults (e.g. offset and resolution values)
- 04h: Restore only Manufacturer-specific CAN objects to factory defaults (e.g. filters)

The answer after successful restoring you will receive is:

COB-ID	Rx/Tx	DLC	Dati							
			D0	D1	D2	D3	D4	D5	D6	D7
580h + Node-ID	Tx	8	60h	11h	10h	XX	00h	00h	00h	00h

To permanently save the parameters restored, send the following SDO message:

COB-ID	Rx/Tx	DLC	Dati							
			D0	D1	D2	D3	D4	D5	D6	D7
600h + node-ID	Rx	8	23h	10h	10h	XX	73h	61h	76h	65h

GEFRAN spa
GEFRAN
via Sebina, 74 - 25050 PROVAGLIO D'ISEO (BS) - ITALIA
tel. 0309888.1 - fax. 0309839063 Internet: http://www.gefran.com