Software User Manual

eNod4-B

Digital Process Transmitter

NU-soft-eNod4B-E-0716_236704-C.doc

SCAIME SAS – Technosite Altéa – 294, Rue Georges Charpak – 74100 JUVIGNY - FRANCE
Tél. : +33 (0)4 50 87 78 64 – www.scaime.com

1 ENOD4 PRODUCT RANGE

1.1 General presentation

eNod4 is a high speed digital process transmitter with programmable functions and powerful signal processing capabilities. ***eNod4*** offers operating modes for advanced process control both static and dynamic.

Quick and accurate:

- Analog to digital conversion rate up to 1920 meas/s with maximum scaled resolution of ±500 000 points.
- Digital filtering and measurement scaling.
- Measurement transmission up to 1 000 meas/s.

Easy to integrate into automated system:

- **USB**, **RS485** and **CAN** communication interfaces supporting ***ModBus RTU***, ***CANopen®*** and ***PROFIBUS-DPV1*** (depending on version) communication protocols.
- Digital Inputs/Outputs for process control.
- Setting of node number by rotary switches and communication baud rate by dip switches.
- Integrated selectable network termination resistors.
- Wiring by plug-in terminal blocs.

1.2 Versions

1.2.1 Communication protocol versions

- Strain gauges load-cell conditioner with ***CANopen®*** and ***ModBus RTU*** communication.
- Strain gauges load-cell conditioner with ***Profibus DP-V1*** and ***ModBus RTU*** communication.
- Strain gauges load-cell conditioner with ***Modbus TCP*** and ***ModBus RTU*** communication.
- Strain gauges load-cell conditioner with ***EtherNet/IP*** and ***ModBus RTU*** communication.
- Strain gauges load-cell conditioner with ***Profinet IO*** and ***ModBus RTU*** communication.
- Strain gauges load-cell conditioner with ***EtherCAT*** and ***ModBus RTU*** communication.

EDS, GSD, ESI and GSDML configuration files for above protocols can be downloaded from our web site: http://www.scaime.com

1.2.2 IO+ version

In conjunction with all communication protocol versions, ***eNod4*** can supports an opto-insulated board fitted with:

- 2 additional digital inputs and 1 speed sensor dedicated input.
- 0-5V or 0-10V analog output voltage.
- 4-20mA, 0-24mA, 0-20mA or 4-20mA with alarm at 3.6mA analog output current.

1.3 eNodView Software

So as to configure ***eNod4***, SCAIME provides eNodView software tool. ***eNodView*** is the software dedicated to eNod devices and digital load cell configuration from a PC. This simple graphical interface allows accessing the whole functionalities of ***eNod4*** for a complete setting according to the application.

eNodView features and functions:

- eNod4 control from a PC
- Calibration system
- Modification/record of all parameters
- Measure acquisition with graphical display
- Numerical filters simulation
- Frequential analysis FFT
- Process control
- Network parameter

eNodView software is available in English and French version and can be downloaded from our web site: http://www.scaime.com or ordered to our sales department on a CD-ROM support.

2 COMMUNICATION AND FUNCTIONING MODES

Name	Modbus address	CANopen® Index/sub-index	Profibus cyclic IN/OUT	Acyclic DPV1 slot/index	Type	Access
Functioning mode / Serial protocol	0x003E	0x2000/0x00	R : 0x02E8 W: 0x02E9	0x07 / 0x39	Uint	RW
HMI name	0x0034	0x3701/0x00		0x0B / 0x00	String	RW

2.1 Communication protocols Modbus RTU and SCMBus

Modbus RTU, SCMBus, and fast SCMBus communication protocols are accessible through AUX, USB. Modbus RTU or Profibus only depending on version on DB9 connection.

The protocol can be changed via the « Functioning mode/ serial protocol » register (see below).

bits b9b8	Protocol
00	*SCMBus*
01	*Modbus RTU*
11	*Fast SCMBus*

Note: To be applied, any modification of this setting must be followed by an EEPROM back up and device reboots (hardware or software).

2.2 Functioning mode

The « Functioning mode/ serial protocol » register offers the possibility to change the eNod4 application according to the following list:

bits b_1b_0	Functioning mode				
	eNod4-T	eNod4-C	eNod4-D	eNod4-F	eNod4-B
00	*Transmitter*	*Transmitter*	*Transmitter*	*Transmitter*	*Transmitter*
01	/	*Checkweigher transmitter on request*	*Dosing by filling*	*Dosing*	*Belt scale*
10	/	/	*Dosing by unfilling*	/	*Belt weigh feeder*

Note: To be applied, any modification of this setting must be followed by an EEPROM back up and device reboots (hardware or software).

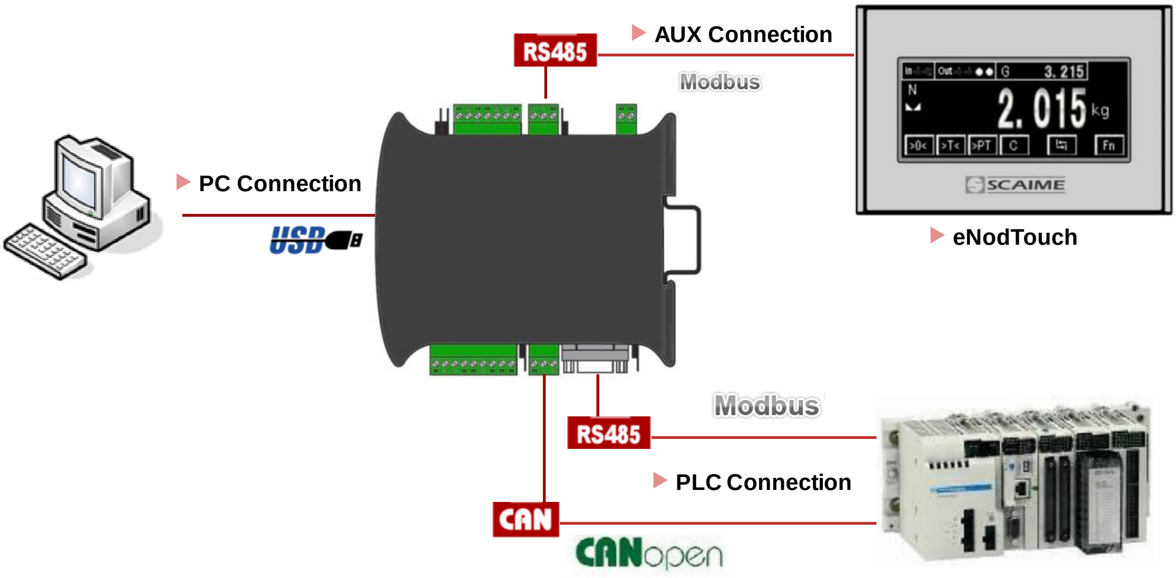
2.3 HMI name

The “HMI name” is a string of 4 characters freely usable to identify the node on any HMI connected to eNod.

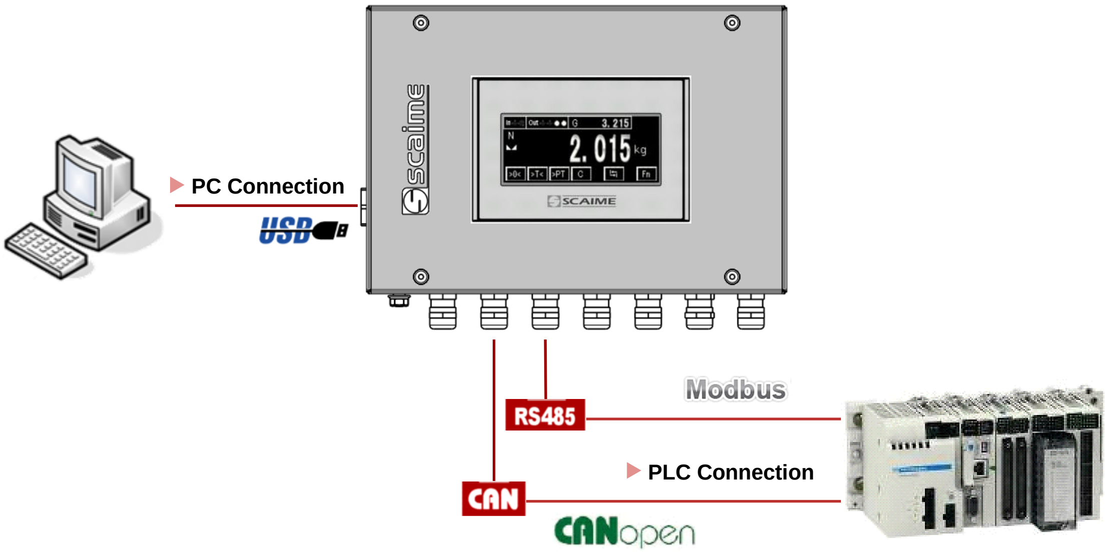
2.4 Simultaneous functioning of communications

2.4.1 Standard version

- **DIN Version**



- **BOX Version**

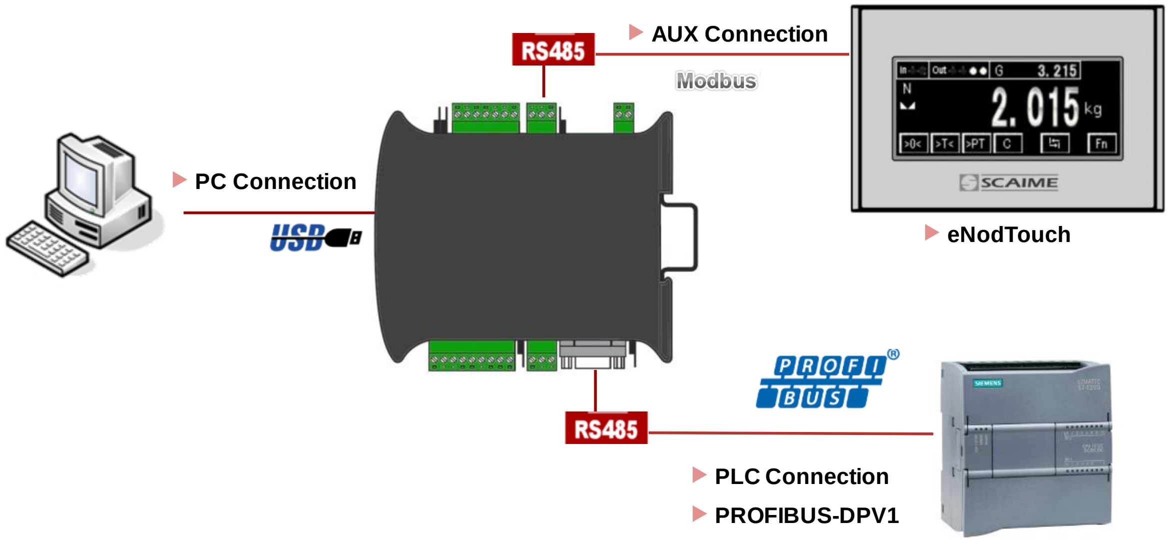


Simultaneous Communication	*RS485 PLC*	*RS485 AUX*	*CAN*
USB	*Yes**	*No*	*Yes**
RS485 PLC		*Yes*	*No*
RS485 AUX			*Yes**

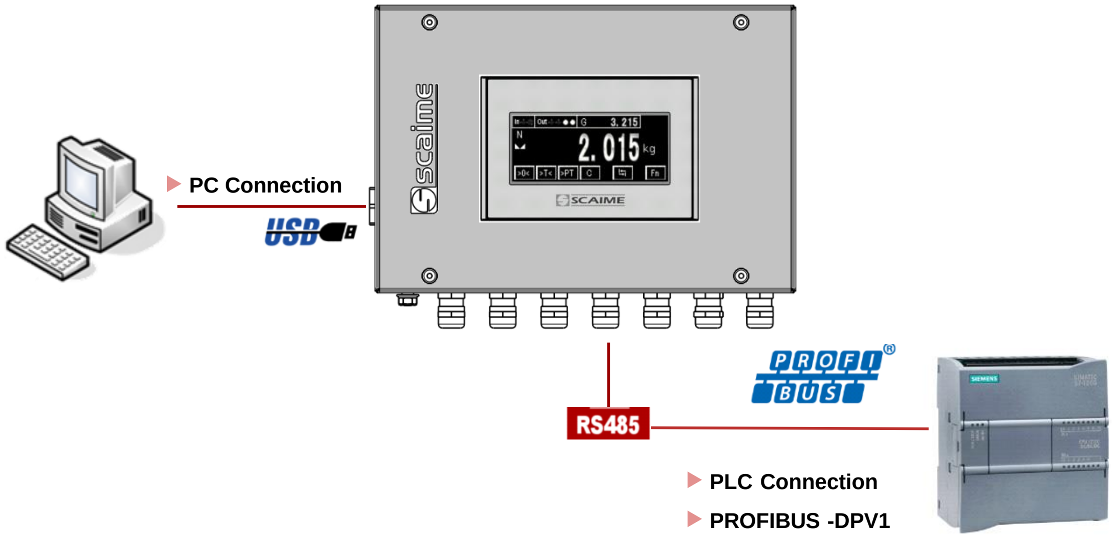
(*)Simultaneous use of CAN or RS485 PLC communication with USB or RS485 AUX can reduce performance of this interface.

2.4.2 Profibus version

- **DIN Version**



- **BOX Version**

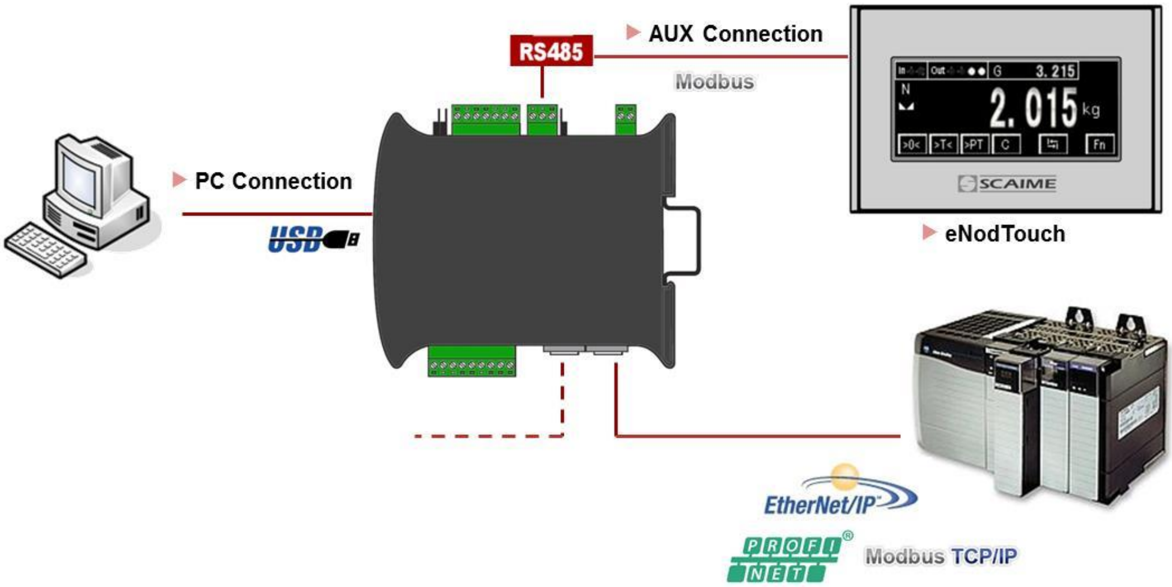


Simultaneous Communication	*Profibus*	*RS485 AUX*
USB	*Yes**	*No*
Profibus		*Yes**

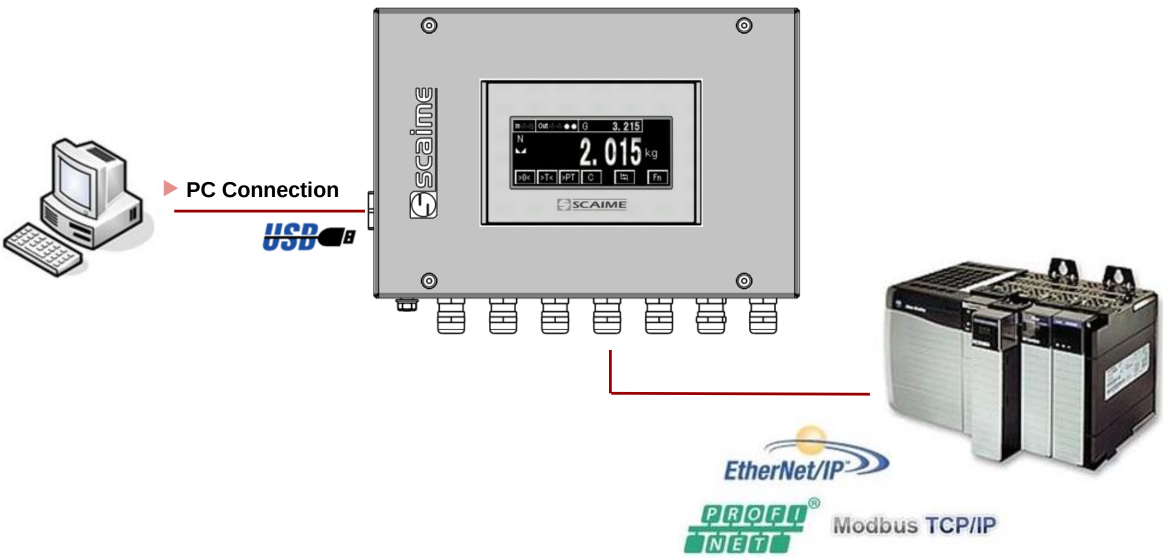
(*)Simultaneous use of Profibus with USB or RS485AUX can reduce performance of this interface.

2.4.3 Ethernet versions

- **DIN Version**



- **BOX Version**



Simultaneous Communication	*Ethernet*	*RS485 AUX*
USB	*Yes**	*No*
Ethernet		*Yes**

(*)Simultaneous use of Ethernet with USB or RS485 AUX can reduce performance of this interface.

3 MODBUS RTU

3.1 Physical interfaces

Modbus RTU communication protocol can be used either through ***eNod4*** USB port, AUX port. Modbus RTU or Profibus only depending on version on DB9 connection.

USB port behaves as a full duplex interface whereas the DB9 and AUX ports support half-duplex RS485 communication. Supported baud rates are 9600, 19200, 38400, 57600, and 115200.

For a complete description of the recommendations about ***eNod4*** RS485 connection, please refer to the user manual "characteristics and functioning" of the ***eNod4***.

Note: using ***eNod4*** through USB requires installing first the necessary USB drivers available on the website http://www.scaime.com.

3.2 Byte format

Data transmitted to ***eNod4*** thanks to Modbus RTU communication protocol must respect following format:

- 1 start bit
- 8 data bits
- no parity
- 2 stop bits

Every Modbus RTU frame is ended by a CRC-16 2-bytes code whose polynomial generator is

$$G(x) = x^{16} + x^{15} + x^2 + 1$$

(cf. CRC-16 calculation algorithm).

3.3 Modbus RTU supported functions

As a Modbus RTU slave, ***eNod4*** supports following Modbus RTU functions:

Function	*Code*
read N registers*	*03_H / 04_H*
write 1 register*	*06_H*
write N registers*	*10_H*

** 1 register = 2 bytes, maximum admitted value for N is 30.*

Note: Broadcast addressing is not allowed by ***eNod4***.

3.4 Frames structure

During a read or write transaction, the two bytes of a register are transmitted MSB first then LSB.

If a data is coded on **4 bytes** (that means it requires two registers), **the two LSB are stored in the low address register and the two MSB are stored in the high address register.**

3.4.1 Function (03H/04H) – read N input registers (N = 30 max)

- request command sent to the slave :

slave address	*03_H or 04_H*	*starting register offset*	*N registers*	*CRC16*
1 byte	*1 byte*	*2 bytes*	*2 bytes*	*2 bytes*

- slave response :

slave address	03_H or 04_H	NB *	data 1	...	CRC16
1 byte	1 byte	1 byte	2 bytes	2 bytes	2 bytes

* NB: number of read bytes (= N*2)

3.4.2 Function (06H) – write single register

- request command sent to the slave :

slave address	06_H	register offset	data	CRC16
1 byte	1 byte	2 bytes	2 bytes	2 bytes

- slave response :

slave address	06_H	register offset	data	CRC16
1 byte	1 byte	2 bytes	2 bytes	2 bytes

3.4.3 Function (10H) – preset multiple registers (N = 30 max)

- request command sent to the slave :

slave address	10_H	starting register offset	N registers	NB	Data 1	...	CRC16
1 byte	1 byte	2 bytes	2 bytes	1 byte	2 bytes	2 bytes	2 bytes

- slave response :

slave address	10_H	starting register offset	N registers	CRC16
1 byte	1 byte	2 bytes	2 bytes	2 bytes

3.4.4 Error frames

- frame format in case of a transaction error :

slave address	Function code + 80_H	error code	CRC16
1 byte	1 byte	1 byte	2 bytes

- Error codes meaning :

Error code	Meaning	description
01_H	*illegal function*	*Modbus-RTU function not supported by eNod4*
02_H	*illegal data address*	*register address requested out of eNod4 register table*
03_H	*illegal data value*	*forbidden data values for the requested register*
04_H	*eNod4 not ready*	*eNod4 is not ready to answer (for example measurement request during a taring operation)*

3.5 Address and Baud rate

Address Modbus RTU	Meaning	Access	Type
0x0001	Address and Baud rate	RO	Uint

Reads the address and baud rate selected on the front panel via the rotary switches and dipswitches.

3.6 Product identification

Software and product versions of the ***eNod4*** are accessible via Modbus RTU.

Address Modbus RTU	Meaning	Access	Type
0x0000	*SW and product version*	*RO*	*Uint*

The 12 LSB bits define the software version (073_H = 115) and the 4 MSB bits define the product version (6_H for the ***eNod4***).

3.7 Measurement transmission

As a master/slave protocol, measurement transmission in Modbus protocol is only done on master request.

3.8 EEPROM error management

Functioning and calibration parameters are stored in EEPROM. After every reset the entireness of parameters stored in EEPROM is checked. If a default appears, measurements are set to 0xFFFF and default is pointed out in measurement status.

4 SCMBUS / FAST SCMBUS

4.1 Physical interfaces

SCMBus and fast SCMBus communication protocols can be used either through ***eNod4*** USB port and AUX port.

USB port behaves as a full duplex interface whereas the DB9 and AUX ports support half-duplex RS485 communication. Supported baud rates are 9600, 19200, 38400, 57600, and 115200.

For a complete description of the recommendations about ***eNod4*** RS485 connexion, please refer to the user manual "characteristics and functioning" of the ***eNod4***.

Note : using ***eNod4*** through USB requires installing first the necessary USB drivers available on the website http://www.scaime.com.

4.2 SCMBus and fast SCMBus features

SCMBus and its variant fast SCMBus can be imbricate into ModBus RTU protocol if the setting *'communication protocol'* is set to SCMBus or fast SCMBus. That means that ***eNod4*** continues answering Modbus RTU frames but it also allows the device to send frames coded according to SCMBus/fast SCMBus format.

Each protocol has its advantages:

- in SCMBus measurements are transmitted as ASCII with the decimal point and the unit integrated to the frame
- fast SCMBus is dedicated to fast measurement transmission as the frames are the most compact as possible
- both protocols allow to communicate without any master request (continuous transmission or sampling triggered by a logical input)

4.3 Byte format

Data transmitted to ***eNod4*** thanks to SCMBus or fast SCMBus communication protocol must respect following format:

- 1 start bit
- 8 data bits
- no parity
- 2 stop bits

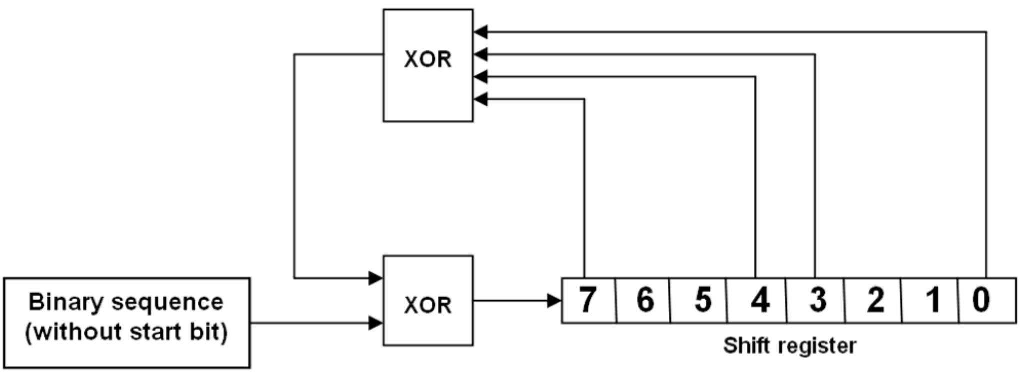
in SCMBus protocol, data is encoded as ASCII numeral characters (30_H ..... 39_H) and ASCII hexadecimal characters ($3A_H$ ..... $3F_H$).

in fast SCMBus protocol, data is encoded as signed hexadecimal (see frame structure paragraph) below.

SCMBus CRC-8 byte is generated by the following polynomial:

$$G(x) = x^8 + x^7 + x^4 + x^3 + 1$$

The CRC-8 polynomial result can be determined by programming the algorithm corresponding to the following diagram:



Note: The frame error detection can be ignored. Value **0xFF** of the CRC-8 always is admitted by ***eNod4*** and a received frame which is ended by such CRC-8 is considered as a frame without any error.

- Fast SCMBus checksum byte is obtained by summing all the frame previous bytes then setting b7 bit to 1.

4.4 Frames structure

4.4.1 Transmission organization

- frame : ***eNod4*** address first
- byte : lsb first
- multi-bytes data : MSB first

4.4.2 Reading request

- request

Address	*Command*	*CR*	*CRC*
*1 **Hex** byte*	*1 **Hex** byte (command)*	*1 **ASCII** byte (0D$_H$)*	*1 **Hex** byte*

- SCMBus response

Address	*Status*	*Value*	*CR*	*CRC*
*1 **Hex** byte*	*2 **Hex** bytes*	*N **ASCII** Hex bytes*	*1 **ASCII** byte (0D$_H$)*	*1 **Hex** byte*

If the *'decimal point position'* and the *'unit'* settings are assigned to a non-null value, the response frame when transmitting measurement contains the decimal point character (2E$_H$) and the unit that is separated from the measurement value by a space ASCII character (20$_H$).

- Fast SCMBus response

STX	*Status word*	*Value*	*Cks*	*ETX*
02$_H$	*2 Hex bytes*	*3 signed Hex bytes (2's complement)*	*Σ of previous bytes and b7 bit set to 1*	*03$_H$*

Note: Because values are encoded in signed hexadecimal bytes format (2's complement) some data bytes can be equal to **STX (02$_H$) or ETX (03$_H$) or DLE (10$_H$)** so before those specific bytes values a **DLE (10$_H$)** byte is inserted. The ***eNod4*** address is not transmitted in the frame.

4.4.3 Functional command request (tare, zero…)

- request :

Address	*Command*	*CR*	*CRC*
*1 **Hex** byte*	*1 **Hex** byte (command)*	*1 **ASCII** byte (0D$_H$)*	*1 **Hex** byte*

- response (SCMBus and fast SCMBus) :

Address	*Command*	*CR*	*CRC*
*1 **Hex** byte*	*1 **Hex** byte (command)*	*1 **ASCII** byte (0D$_H$)*	*1 **Hex** byte*

If the command execution is successful, ***eNod4*** sends back the request frame that has been received as an acknowledgement.

4.4.4 Error frame

In case of an error upon reception of a request, ***eNod4*** sends back an error frame that contains an error code:

- response (SCMBus and fast SCMBus) :

Address	*Error code*	*CR*	*CRC*
*1 **Hex** byte*	*1 **Hex** byte (command)*	*1 **ASCII** byte (0D$_H$)*	*1 **Hex** byte*

- The error codes are listed below:

Error code	Meaning	Description
FE$_H$	*unknown command*	*requested command is not supported by **eNod4***
FF$_H$	*error during command execution*	*ex. : tare when gross meas.<0*

4.5 Address and Baud rate

Address and baud rate identical to Modbus RTU (See § Modbus RTU)

4.6 Product identification

Product identification identical to Modbus RTU (See § Modbus RTU)

4.7 Measurement transmission

Measurement transmission can be triggered by a master request but it might also be triggered and used through the following options:

- transmission triggered by a rising or falling edge on a logical input
- transmission at a configurable period (defined in ms) while a logical input is maintained at a given logical level
- continuous transmission at a configurable period (defined in ms) after a master request. The transmission is then stopped by another master instruction, be careful not to use this mode in half-duplex at a too high rate.

4.8 Continuous transmission

SCMBus and fast SCMBus communication protocols allow ***eNod4*** to transmit measurements at a user-defined rate without the need for successive master queries. To perform this measurement acquisition mode, it is necessary to set first the *'sampling period'* (in ms):

Address SCMBus	Description	Accès	Type
0x003F	*SCMBus Measurement transmission period*	*RW*	*Uint*

A value of 0 implies that measurement transmission is synchronized on the A/N conversion rate. The continuous transmission is triggered and stopped by reception of the following commands:

SCMBus/fast SCMBus functional command	Command code
start net measurement transmission	*E0$_H$*
start factory calibrated points transmission	*E1$_H$*
start brut measurement transmission	*E2$_H$*
stop continuous transmission	*E3$_H$*

Note 1: the measurement transmission rate also depends on the baud rate. So, to achieve the fastest transmission, it is necessary to use the highest baud rate.

Note 2: as RS485 is a half-duplex communication medium, it can be a little hard to transmit the '*stop continuous transmission'* query if the bandwidth is saturated. Therefore, prefer USB communication channel to reach the highest measurement transmission rate.

4.9 EEPROM error management

EEPROM management identical to Modbus RTU (See § Modbus RTU)

5 CANOPEN®

5.1 Physical interface

eNod4 is equipped with a CAN 2.0A compatible interface supporting **CANopen® communication protocol**. The device can be connected to a CAN bus using **CANH** and **CANL** connections. A REF pin can also be connected. Supported baud rates are 50000, 125000, 250000, 500000 and 1000000.

For a complete description of the recommendations about ***eNod4*** CAN connexion, please refer to documentation *"characteristics and functioning"*.

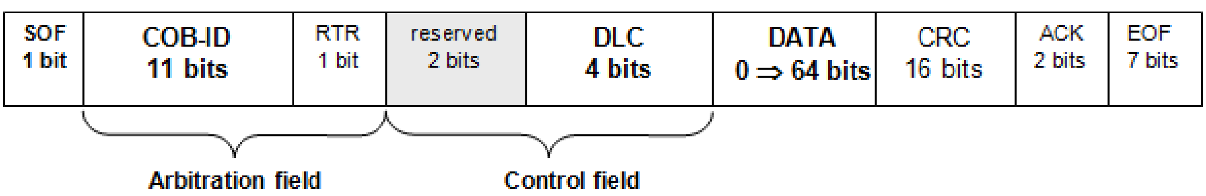
5.2 LED CANopen

The subsequent table describes the meaning of the LEDs for the CAN:

Color	*State*	*Meaning*
Red	*Single Flash*	*At least one of the error counters of the CAN controller has reached or exceed the warning level (too many error frame)*
	Double Flash	*A guard event (NMT-slave or NMT-master) or a heartbeat event (heartbeat consumer) has occurred*
	On	*The CAN controller is bus off*
	Flash	*Self-test: while the device is performing its power up testing, the LED shall be flashing red*
Green	*Blinking*	*The device is in state PRE OPERATIONAL*
	Single Flash	*The device is in state STOPPED*
	On	*The device is in state OPERATIONAL*

5.3 Frame format

Every data frame sent on the CAN bus has the following structure:



- **Start of frame (SOF) : 1 bit**

The beginning of a request or a data frame is indicated by the transmission of one dominant bit.

- **Arbitration field : 12 bits**

This field contains the message COB-ID on 11 bits and the RTR bit, dominant for data frames and recessive for remote frames.

- **Control field : 6 bits**

  The first two bits are reserved and must be transmitted as dominant. The four remaining bits encode the size of the transmitted data in bytes. This is called **«Data length code» (DLC)** with **0 ≤ DLC ≤ 8**.

- **Data : de 8 à 64 bits**

For each byte, the most significant bit (MSB) is transmitted first.

- **Cyclic Redundancy Check (CRC) : 16 bits**

The result of the CRC calculation is made up of 15 bits that guarantee the integrity of the transmitted message. The last bit is used to delimit the field and always is transmitted as dominant.

- **Acknowledgement (ACK) : 2 bits**

During two bus clock periods, the bus is available for acknowledgement of the message. All the nodes that received the message without error generate a dominant bit. Else, an error frame is generated. The second bit is always recessive.

- **End of frame (EOF) : 7 bits**

The end of the frame is represented by a sequence of 7 consecutive recessive bits.

The CANopen® layer defines particularly the content of the arbitration and the control fields and the data field structure.

5.4 Messages transfers hierarchy

CANopen® is a communication protocol especially dedicated to industrial applications. It allows connecting up to 127 different devices on a same bus giving them the possibility to access the bus at any time. Simultaneous emissions are managed by an arbitration system that uses priority levels.

This control hierarchy of data transfers guarantees that there is no frame collision on the bus while ensuring a high level of reliability in communications. The low priority messages are cancelled and reissued after a delay.

The protocol defines several message types characterized by their COB-ID (Communication Object Identifier) that determines the message priority level. The COB-ID is composed of a function code and the node identifier (between 1 and 127).

The node identifier is the device's address on the network. The function code specifies the priority and the purpose of the message. **Assignment of a particular identifier to each device connected to the bus is mandatory.**

eNod4 supports 6 different message types :

- read/write requests : **SDO** (Service Data Object)
- real time transfers : **PDO** (Process Data Object)
- nodes state management : **NMT** (Network Management)
- warnings : **EMCY** (Emergency)
- synchronization events : **SYNC** (Synchronization)
- node status indications : **Boot-up/Heartbeat** and **Node guarding**

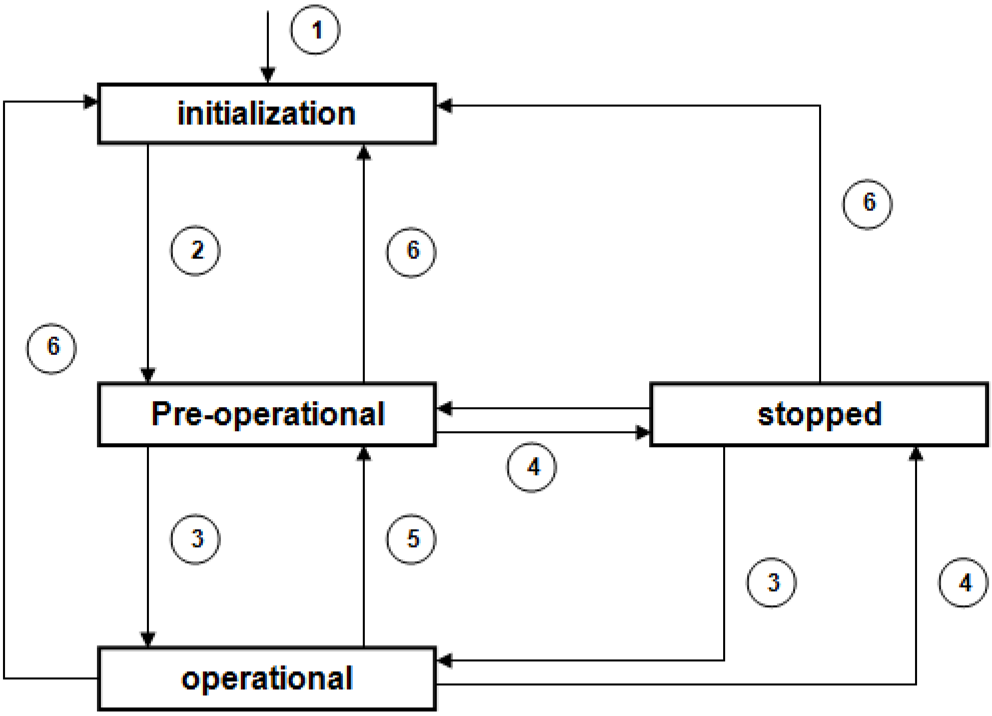
CANopen® messages	*COB-ID (hex)*
NMT	*0*
SYNC	*80*
EMCY	*81-FF*
TPDO1	*181 – 1FF*
RPDO1	*201 – 280*
RPDO2	*301 – 380*
RPDO3	*401 – 480*
RPDO4	*501 – 580*
RPDO5(IO+ version)	*681 – 6FF*
TPDO2	*281 – 2FF*
TPDO3	*381 – 3FF*
SDO (Tx)	*581 – 5FF*
SDO (Rx)	*601 – 67F*
Heartbeat/Boot-up	*701 – 77F*

5.5 eNod4 status remote management

For the CANopen® network, ***eNod4*** is considered as a **NMT slave**. It means that its state can be modified by a **NMT master** present on the bus.

As other CANopen® nodes, ***eNod4*** can be set into one of the four existing states, allowing or forbidding the reception/emission of CAN messages.

These four states constitute the following NMT state machine:



- ⇨ 1 : ***eNod4*** device power-up
- ⇨ 2 : automatic transition after the end of initialization
- ⇨ 3 : reception of a **'Start Node'** indication
- ⇨ 4 : reception of a **'Stop Node'** indication
- ⇨ 5 : reception of an **'Enter pre-operational mode'** indication
- ⇨ 6 : reception of a **'Reset node'** or a **'Reset communications'** indication

eNod4 communication capacities for each state are given in the following table :

	Initialization	*Pre-operational*	*Operational*	*Stopped*
SDO		***X***	***X***	
PDO			***X***	
SYNC		***X***	***X***	
Emergency		***X***	***X***	
NMT		***X***	***X***	***X***
Boot-up	***X***			
Heartbeat		***X***	***X***	***X***

5.5.1 NMT commands

Except during the initialization phase, ***eNod4*** is able to handle any NMT master's requests for changing its current state. All these network management messages are constituted the same way: **a two-byte data frame with a COB-ID equal to zero:**

COB-ID	DLC	byte 1	byte 2
0	2	NMT code	Node identifier

The 2nd byte of the data field contains the node identifier of the device concerned by the request. Its value must be between 0_H and $7F_H$. The 0_H value means that the NMT command concern all the nodes of the network.

The 1st byte codes the command sent to the node. There are five existing commands supported:

- ⇨ **« Start node »**: 01_H. ***eNod4*** is set into **operational** state
- ⇨ **« Stop node »:** 02_H. ***eNod4*** is set into **stopped** state
- ⇨ **« Reset node »**: 81_H. Resets ***eNod4*** (with the same effects as a power-up), back into **initialization** state.
- ⇨ **« Reset communication »**: 82_H. Back into **initialization** state and communication parameters reset.
- ⇨ **« Enter pre-operational mode »**: 80_H. ***eNod4*** is set into **pre-operational** state

5.5.2 Synchronization messages

SYNC messages are emitted on the bus by a producer node (generally the NMT master). This service is unconfirmed so the consumer nodes do not have to respond to SYNC messages. A SYNC message does not carry any data (DLC = 0). ***eNod4*** is only seen as a SYNC messages consumer whose COB-ID is stored at index 1005_H, sub-index 00_H of the object dictionary.

5.5.3 Emergency messages

eNod4 internal errors are reported via emergency frames. Two types of errors can trigger the transmission of an emergency message:

- communication errors
- A/D converter input signal range exceeded

Every emergency frame is built as follows:

COB-ID	*DLC*	*byte 0*	*byte 1*	*byte 2*	*byte 3*	*byte 4*	*byte 5*	*byte 6*	*byte 7*
*80_H + ID **eNod4***	*8*	*emergency code*		*error register content*	*additional information*				

Emergency message is an unconfirmed service. A frame is emitted when a new error occurs and when it is acknowledged. The table below describes the emergency standard codes supported by ***eNod4-T*** and the translation of the additional information bytes (in ASCII):

Emergency codes (hex.)	*Meaning*
0	*error acknowledged*
3200	*voltage error*
8120	*CAN bus communication error*
8130	*life guard error*
Additional information's (hex.)	
4B4F	*no error*

474C	*life time has elapsed or Heart Beat not received*
564F	*sensor signal outside of the input signal range*
5054	*CAN transmitter in error passive state*
5052	*CAN receiver in error passive state*

The error register value is also part of the emergency telegram so as to indicate if other internal errors have been detected.

The number of reported errors is given by an error counter in the **pre-defined error field** located at index 1003_H, sub-index 00_h and the last reported error can be read from the same entry at sub-index 01_H.

5.6 Error control services

CANopen® uses smart mechanisms to control permanently the nodes state on the bus. ***eNod4*** supports **Boot-up** and **Heartbeat** messages and **Node guarding protocol**. Using both services is not allowed. If both are configured so as to be functional, only the Heartbeat mechanism is used.

5.6.1 Heartbeat and boot-up

eNod4 state control can be achieved through the use of Heartbeat and boot-up mechanisms :

⇨ **Boot-up**: this message sent by ***eNod4*** means that its initialization phase is complete and that the node has entered into **pre-operational** state. It consists in the following frame :

COB-ID	DLC	byte 1
700_H + ID eNod4	**1**	**0**

⇨ **Heartbeat** :

- **producer mode** : if a Heartbeat period (in ms) different from 0 is set in the entry *'producer heartbeat time'* of the object dictionary, ***eNod4*** generates at this period a frame containing its state coded on one byte. The corresponding frame is similar to the **Boot-up** mechanism frame:

COB-ID	DLC	byte 1
700_H + ID eNod4	**1**	**eNod4 NMT state**

eNod4 **NMT state byte** can take the different following values :

⇨ 04_H : the node is in the **«stopped»** state
⇨ 05_H : the node is in the **«operational»** state
⇨ $7F_H$: the node is in the **«pre-operational»** state

Using Heartbeat protocol allows a NMT master to check that all nodes connected to the bus are working correctly.

- **Consumer mode**: ***eNod4*** also can be configured so as to monitor the NMT state of one particular node of the network (generally the NMT master). The node ID and a period are defined in the entry *'consumer heartbeat time'* of the object dictionary

If corresponding heartbeat is not received within this time, then ***eNod4*** sends an emergency telegram and switches to pre-operational state.

5.6.2 Node guarding

Node guarding protocol is another way to check the nodes state. But unlike Heartbeat protocol, it needs requests from a NMT master. In this case, the NMT master sends periodically a remote transmit request (remote frame) to the node with **COB-ID 700$_H$ +** ID ***eNod4***. ***eNod4*** has to respond by sending a single-byte data frame with its coded state.

This frame is similar to Heartbeat frame but there is an important difference. Most significant bit of the state byte is a toggle-bit. The value of this bit must alternate between two consecutive responses from the NMT slave. The value of the toggle-bit of the first response after the Guarding Protocol becomes active is 0.

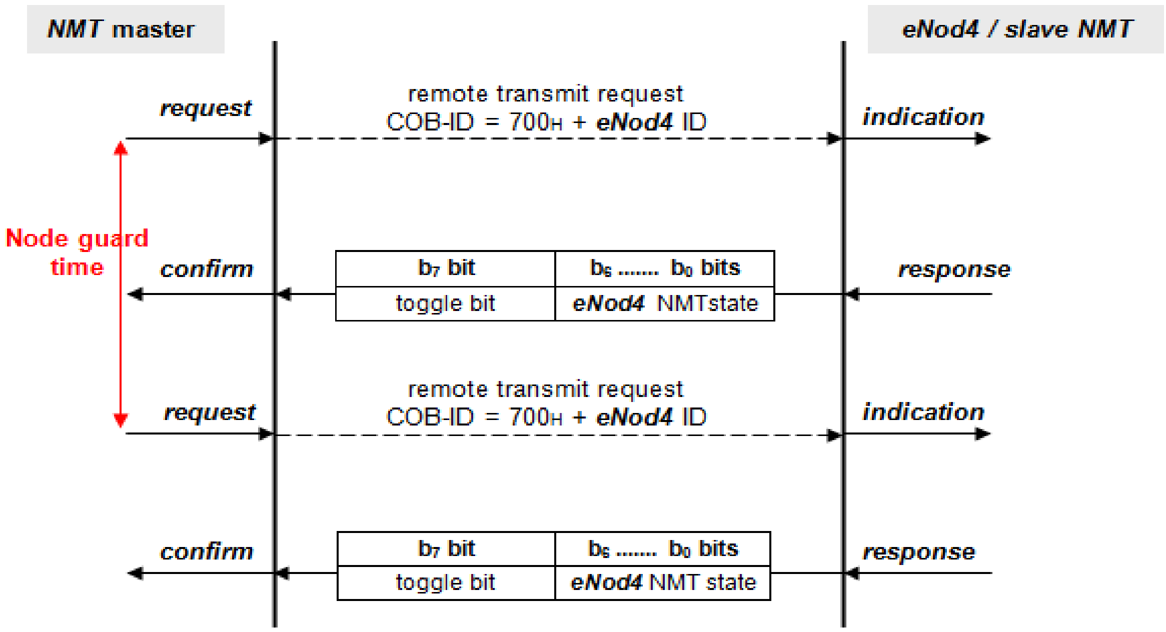
It is only reset to 0 when a *'reset communications'* or a *'reset node'* command is received. If two consecutive responses have the same value of the toggle-bit, then the new response should be handled as if it was not received by the NMT master.

Two parameters of the object dictionary are necessary to set and define node guarding protocol: the **'guard time'** and the **'life time factor'**:

- ⇨ **Guard time**: this parameter expressed in milliseconds indicates the period with which the node is being polled by the NMT master. This value can be different from one node to another.
- ⇨ **Life time factor**: when node guarding protocol is active, **node life time** is given by multiplication of the guard time and the life time factor.

Node guarding activation is effective when guard time has been set (and if Heartbeat protocol is not used) and after reception of the first remote transmit request. If life time factor is also configured and if no remote transmit request is handled within the node life time, ***eNod4* sends an emergency telegram then switches to pre-operational state**. The life guarding error is acknowledged when the state is changed by a NMT command and after reception of a new remote transmit request.

Switching to the stopped NMT state because of a node guarding error may cause ***eNod4*** to be set into a configurable safety mode where parts of its functioning are inhibited



5.7 Access to the object dictionary

The most important element of a CANopen® compatible device is its **object dictionary (OD)**. Each node object that can be accessed via the bus is part of a table called object dictionary. The dictionary entries can be addressed by a couple of an index (2 bytes) and a sub-index (1 byte) with the following organization:

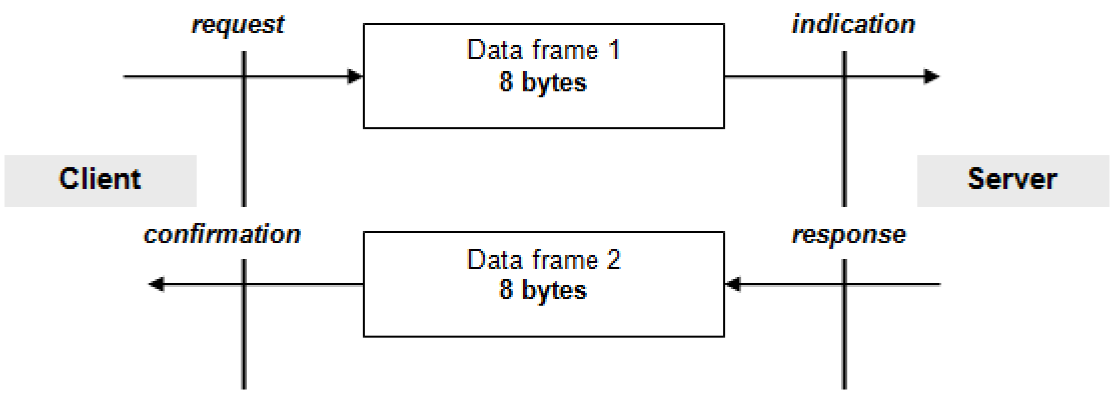
Index (hex.)	*Object type*
0000	*reserved*
0001 ⇒ 001F	*static data types*
0020 ⇒ 003F	*complex data types*
0040 ⇒ 005F	*manufacturer specific complex data bytes*
0060 ⇒ 007F	*device profile specific static data types*
0080 ⇒ 009F	*device profile specific complex data types*
00A0 ⇒ 0FFF	*reserved*
1000 ⇒ 1FFF	***communication profile area***
2000 ⇒ 5FFF	***manufacturer specific profile area***
5FFF ⇒ 9FFF	*standardized device profile area*
A000 ⇒ FFFF	*reserved*

Only the greyed elements of the table are accessible through ***eNod4*** OD.

The whole object dictionary is accessible and can be configured from usual CANopen® configuration tools. This can be done using ***eNod4*** **available EDS file.**

5.7.1 SDO communication

The model for SDO communication is a client/server model as described below:



The node that sends the request is the client application whereas ***eNod4*** only behaves as the server application. There are two types of requests, write and read requests. Both have the same structure:

COB-ID	DLC	byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
11 bits	**1 byte**	**Command byte**	**Index**		**sub-index**	**Data**			
*580_H or 600_H + ID **eNod4***	*8*	*see table*	*LSB*	*MSB*	*/*	*LSB*	*-*	*-*	*MSB*

The client request uses the SDO(Rx) COB-ID (600_H + ID ***eNod4*** and the server uses the SDO(Tx) COB-ID (580_H + ID ***eNod4***).

The **command byte** depends on the requested data length:

Client request	Server response
read data ⇒ 40_H	***43_H ⇒ 4-bytes data***
	$4B_H$ ⇒ 2-bytes data
	$4F_H$ ⇒ 1-byte data
write 4-bytes data ⇒ 23_H	***60_H***
write 2-bytes data ⇒ $2B_H$	
write 1-byte data ⇒ $2F_H$	

For a read request, the value of the four last bytes of the frame (data) does not matter.

If an error occurs during a SDO communication ***eNod4*** responds with the command byte 80_H and the four data bytes contain one of the following SDO abort codes. The data transfer is aborted.

SDO abort codes (hex.)	Description
5040001	*SDO command specifier not supported*
6010001	*unsupported access to an object*
6010002	*attempt to write a read-only object*
6020000	*the object does not exist in the object dictionary*
6040042	*the number and length of the objects to be mapped would exceed PDO length*
6040047	*impossible operation (for example reading a net/gross value during a tare or a zero)*
6070012	*data type does not match, length of service parameter too high*
6070013	*data type does not match, length of service parameter too low*
6090011	*Sub index object does not exist.*
6090030	*value range of parameter exceeded*
6090031	*value of parameter written too high*
6090032	*value of parameter written too low*
8000020	*data cannot be stored to the application*

8000022	*data cannot be transferred or store to the application because of the present device state*

5.7.2 PDO communications

SDO protocol is not the only way to access the object dictionary. PDO allow to transfer data without including their index and sub-index in the frame. Both are stored in an OD specific field called PDO mapping.

The model used for PDO transmissions also is different. It is a Producer/Consumer model in which data are sent by a producer node (TPDO) to a consumer node (RPDO) without any confirmation.

Each PDO is described by a combination of two parameters of the OD: the **PDO communication parameters** and the **PDO mapping**. The PDO communication parameters describe the functioning of the PDO and the PDO mapping describes its content. ***eNod4*** uses **3 TPDO** (2 are programmable) and **5 RPDO**.

The PDO transmission mode can be set in the corresponding object with the following attributes:

⇨ **Synchronous:** PDO transmission/reception is triggered by the reception of one or more SYNC messages. Several options are available :

- cyclic: PDO is sent/received after reception of n ($1 \leq n \leq 240$) SYNC messages.
- acyclic: PDO is sent at reception of the first SYNC message following a specific device event (activation of a logical input assigned to *'send TPDO'* or data variation superior to +/- delta)
- on remote transmit request : PDO is sent after the first SYNC message following a remote transmit request frame with the PDO COB-ID.

⇨ **Asynchronous:** PDO transmission/reception does not depend on the SYNC messages on the CAN bus. Several options are available :

- on remote transmit request : PDO is sent at reception of a remote transmit request frame with the PDO COB-ID.
- activation of a logical input assigned to *'send TPDO'* or data variation superior to +/- delta)
- on a timer event : PDO is sent periodically (with an adjustable period).

The following table recaps the trigger modes that can be chosen by entering the hexadecimal code in the PDO communication parameter:

Code (hex)	*cyclic*	*acyclic (event)*	*synchronous*	*asynchronous*	*remote transmit request*	*Effect*
00		***X***	***X***		***X***	***PDO transmission/reception after a SYNC message following one of these events :*** *- activation of a logical input assigned to 'send TPDO'* *- mapped object variation superior to +/- delta* *- Receipt of remote transmit request.*
01 – F0 (= n)	***X***		***X***		***X***	***PDO transmission after n SYNC messages*** *- Or after receipt of remote transmit request following at less one SYNC.*
F1 - FB	*reserved*					
FC			***X***		***X***	***data update at reception of a remote transmit request and PDO transmission after reception of a SYNC message***
FD				***X***	***X***	***data update and PDO transmission at reception of a remote transmit request***
FE				***X***	***X***	***PDO transmission is triggered by one of these events :*** *- activation of a logical input assigned to 'send TPDO'* *- mapped object variation superior to +/- delta* *- receipt of remote transmit request* *more for TPDO2 and 3, functioning is identical to code FF*
FF				***X***	***X***	***Periodic TPDO emission****. Period can be configured (min = 1 ms).* *RPDO handled upon reception* *TPDO emission after receipt of remote transmit request*

- **Note 1:** for RPDO1-2-3-4-5 and TPDO1, only the transmission types FF_H and 00_H are supported. That means data are updated either immediately upon reception (FF_H) or after next **SYNC** following the RPDO reception (00_H). For TPDO1, the FE_H transmission type means that the TPDO1 is emitted by ***eNod4*** every time it's mapped value changes. The 00_H transmission type is similar but the emission is triggered by the reception of a **SYNC** object.

- **Note 2:** RPDO1-2-3-4-5 and TPDO1 mapping are configured with default values that cannot be modified.

5.8 CANopen command and response registers

In CANopen® communication protocol, the *'command register'* is mapped into **RPDO1** and the *'response register'* is mapped into **TPDO1**. When in operational NMT state, ***eNod4*** is able to handle the **functional commands** received through **RPDO1** and the *'response register'* value changes are automatically transmitted through **TPDO1**.

5.9 Communication objects

Some settings are specific as defined by the CANopen® communication specification.

5.9.1 0x1001 / 0x00 : error register

- **access : RO**
- **data type : unsigned8**
- **default value : 0**
- **mappable ? : N**
- **admitted values : see table below**

Description: The device internal errors are indicated by flag bits of this byte. b_0 bit (generic error) is set to 1 if at least one error is detected.

Bit se to 1	*Meaning*
b_0	*generic error detected*
b_1	*reserved (0)*
b_2	*A/D converter input voltage error*
b_3	*reserved (0)*
b_4	*CAN bus communication error*
b_5	*reserved (0)*
b_6	*reserved (0)*
b_7	*EEPROM error*

5.9.2 0x1003 : Pre-defined error field

Sub-index	*Description*	*Access*	*Default value*	*Mappable (PDO) ?*	*Type*
0x00	*reported errors counter*	*R/W*	*0*	*N*	*unsigned8*
0x01	*last reported error*	*RO*	*0*	*N*	*unsigned32*

Description: This entry of the object dictionary stores the errors that have been reported by emergency telegrams. The reported errors counter (sub-index 00_H) is accessible through write or read request but 0 is the only allowed value for writing transactions. By writing a zero to this sub-index, the error counter is reset and the last reported error (sub index 01_H) is erased. An attempt to write another value is ignored and ***eNod4*** answers the SDO abort code 0x06090030.

5.9.3 0x1005 / 0x00 : synchronization messages COB-ID

- **access : R/W**
- **data type : unsigned32**
- **default value : 0x80**
- **mappable ? : N**
- **admitted values : 0x80 or from 0x7E0 up to to 0x7E3**

Description: This object contains the message COB-ID value supported by ***eNod4*** as synchronization messages (used for PDO activating).

5.9.4 0x100C / 0x00 : guard time

- **access : R/W**
- **data type : unsigned16**
- **default value : 0**
- **mappable ? : N**
- **admitted values : from 0 up to 65535**

Description: The *'life guard'* is one of the two parameters used by the node guarding protocol (errors detection). When *'Heartbeat time'* is inactive and *'life guard'* is different from 0, ***eNod4*** responds to NMT master periodic (period equal to life guard) remote transmit requests.

5.9.5 0x100D / 0x00 : life time factor

- **access : R/W**
- **data type : unsigned8**
- **default value : 0**
- **mappable ? : N**
- **admitted values : from 0 up to 255**

Description: By multiplying the *'life guard'* by the *'life time factor'*, the node life time (cf. §4) can be determined. When node guarding is active, if the node has not be polled within this duration (in ms), ***eNod4*** state is set to stopped. ***eNod4*** behavior while stopped can be configured via the object at index 0x4800.

5.9.6 0x1010 : Store parameters

Sub-index	*Description*	*Access*	*Default value*	*Mappable (PDO) ?*	*Type*
0x00	*largest sub-index*	*RO*	*0x01*	*N*	*unsigned8*
0x01	*save all parameters*	*R/W*	*0x01*	*N*	*unsigned32*

In CANoppen® communication protocol, storing all settings into ***eNod4*** EEPROM memory requires writing through SDO the ASCII string « save » (65766173H) to sub-index 0x01 of entry 0x1010 of the object dictionary (called *'save all parameters'* in the EDS file).

- **0x65 ⇒ e**
- **0x76 ⇒ v**
- **0x61 ⇒ a**
- **0x73 ⇒ s**

When accessing to sub-index 1 with a SDO read request, ***eNod4*** cell responds with a value of 1 that means that parameters are stored in non-volatile memory only on request.

Note: Restore to default settings functional command is not available in CANopen® communication protocol.

5.9.7 0x1014 / 0x00 : Emergency COB-ID

- **access : RO**
- **data type : unsigned32**
- **default value : 0x81**
- **mappable ? : N**
- **admitted values : from 0x81 up to 0xFF**

Description: The COB-ID of emergency messages transmitted by ***eNod4*** is stored at this index. Its value automatically is updated if the node identifier is modified.

5.9.8 0x1016 : Heartbeat consumer time

Sub-index	*Description*	*Access*	*Default value*	*Mappable (PDO) ?*	*Type*
0x00	*max. sub-index*	*RO*	*1*	*N*	*unsigned8*
0x01	*heartbeat consumer time*	*RW*	*0*	*N*	*unsigned32*

Description: eNod4 is able to monitor the Heartbeat generated by another node (see §4) of the network (in general the master). Two settings must be defined through the object at sub-index 0x01:

- ⇨ **the heartbeat time period**, coded on the 16 LSB bits
- ⇨ **the node ID** of the heartbeat producer to monitor, coded on the 16 MSB bits (from 0x01 up to 0x7F)

5.9.9 0x1017 / 0x00 : Heartbeat producer time

- **access : R/W**
- **data type : unsigned16**
- **default value : 0**
- **mappable ? : N**
- **admitted values : from 0 up to 65535**

Description: If a period different from 0 is written into this index, ***eNod4*** periodically generates a Heartbeat frame. It is expressed in ms and must be comprised between 1 and 65535.

5.9.10 0x4800 : Safety mode

- **access : R/W**
- **data type : unsigned8**
- **default value : 0H**
- **admitted values : see table below**

Description: This entry defines ***eNod4*** functioning when in stopped NMT state. The safety mode is used when the bit b0 of the byte is set to 1. The functioning mode is then inhibited and outputs logical level are given by b1, b2, b3 and b4 bits. The ***eNod4*** leaves the safety mode upon reception of a new NMT command.

Warning: In safety mode, when the ***eNod4*** is in stopped NMT state, functioning is also inhibited using Modbus and SCMBus communication. For example, it is not possible to tare or make a zero using RS 485 AUX port, or eNodView connected via USB.

b0	Effect	Notes
0	*safety mode disabled*	***only valid in stopped state***
1	*safety mode enabled*	***No action possible also using Modbus or SCMBus communication***
b1		
0	*output 1 inhibited*	*depending on the chosen logic*
1	*output 1 set active*	
b2		
0	*output 2 inhibited*	*depending on the chosen logic*
1	*output 2 set active*	
b3		
0	*output 3 inhibited*	*depending on the chosen logic*
1	*output 3 set active*	
b2		
0	*output 4 inhibited*	*depending on the chosen logic*
1	*output 4 set active*	

5.9.11 Error behavior

Sub-index	Description	Access	Default value	Mappable (PDO) ?	Type
0x00	*Number of error classes*	*LS*	*1*	*N*	*unsigned 8*
0x01	*communication error*	*L/ E*	*0*	*N*	*unsigned 8*

Description: If a communication error occurs in operational mode, ***eNod4*** shall enter by default autonomously the pre-operational mode.

Alternatively, by error class coding in object 0x1029 ***eNod4*** can enter in the following states:

- **0: Pre-operational**
- **1: No state change**
- **2: Stopped. refer to '*Safety mode*'**

5.10 PDO-related communication objects

5.10.1 RPDO default mapping

- *0x1600 : RPDO1 mapping parameters*

Sub-index	Description	Access	Default value	Type
0x00	*number of supported objects*	*RO*	*1*	*unsigned8*

0x01	*1st object mapping*	*RO*	*0x20030008* *(command register)*	*unsigned32*

When in operational NMT state, ***eNod4*** is able to handle the **functional commands** received through **RPDO1** and the ***'response register'*** value changes are automatically transmitted through **TPDO1**.

- **0x1601 : RPDO2 mapping parameters**

Sub-index	Description	Access	Default value	Type
0x00	*number of supported objects*	*RO*	*1*	*unsigned8*
0x01	*1st object mapping*	*RO*	*0x30010120* *(calibration load 1)*	*unsigned32*

- **0x1602 : RPDO3 mapping parameters**

Sub-index	Description	Access	Default value	Type
0x00	*number of supported objects*	*RO*	*2*	*unsigned8*
0x01	*1st object mapping*	*RO*	*0x25000020* *(zero offset)*	*unsigned32*
0x02	*2nd object mapping*	*RO*	*T, C : 0x30050120* *D, F, B : 0x30060120 (span adjusting coefficient)*	*unsigned32*

- **0x1603 : RPDO4 mapping parameters**

Sub-index	Description	Access	Default value	Type
0x00	*number of supported objects*	*RO*	*2*	*unsigned8*
0x01	*1st object mapping*	*RO*	*0x30020020* *(maximum capacity)*	*unsigned32*
0x02	*2nd object mapping*	*RO*	*0x30040020 (sensitivity)*	*unsigned32*

- **0x1604 : RPDO5 mapping parameters**

Sub-index	Description	Access	Default value	Type
0x00	*number of supported objects*	*RO*	*1*	*unsigned8*
0x01	*1st object mapping*	*RO*	*0x50500010* *(External value to control analog output)*	*unsigned32*

Note: RPDO5 is default deactivated. To use it you have first to activate it setting 0 to the 32nd bit of the RPDO5 COB-ID object (0x1404/0x01).

5.11 Product identification

The device identification settings are a part of the communication profile area of the object dictionary.

- ***0x1009 : Manufacturer hardware version***

Sub-index	Description	Access	Default value	Type
0x00	*manufacturer hardware version*	*RO*	*0x32302E31* *(1.04)*	*visible string*

- ***0x100A : Manufacturer software version***

Sub-index	Description	Access	Default value	Type
0x00	*manufacturer software version*	*RO*	*0x30302E31* *(1.00)*	*visible string*

- ***0x1018 : Identity object***

Sub-index	Description	Access	Default value	Type
0x00	*max. sub-index*	*RO*	*0x04*	*unsigned8*
0x01	*vendor ID*	*RO*	*0x00000142*	*unsigned32*
0x02	*product code*	*RO*	*0x000816E0*	*unsigned32*
0x03	*revision number*	*RO*	*0x00010000*	*unsigned32*
0x04	*serial number*	*RO*	*0x000186A1*	*unsigned32*

5.12 Measurement transmission

CANopen® includes smart transmission mechanisms that are presented in § "PDO communication". All the measurements can be exchanged either through SDO read requests or through TPDO.

TPDO trigger sources are described in a table of § "PDO communication" and all these communication modes are possible for measurement transmission. Thanks to the various TPDO transmission types, ***eNod4*** offers possibility to have a high measurement transmission rate (up to 1000 meas/s) or to limit the bus occupation by causing the exchange on an event.

5.13 EEPROM error management

Functioning and calibration parameters are stored in EEPROM NOV RAM. After every reset the entireness of parameters stored in EEPROM is checked. If a default appears, measurements are set to 0xFFFF and default is pointed out to object dictionary (see bit 6 of object 0x5003/0x00).

6 CANOPEN TPDO MAPPING

- ## 0x1A00 : TPDO1 mapping parameters

Sub-index	Description	Access	Default value	Type
0x00	number of supported objects	RO	5	unsigned8
0x01	1st object mapping	RO	0x20040008 (response register)	unsigned32
0x02	2nd object mapping	RO	0x50070210 (Belt status)	unsigned32
0x03	3rd object mapping	RO	0x51000008 (logical inputs level)	unsigned32
0x04	4st object mapping	RO	0x52000008 (logical outputs level)	unsigned32
0x05	5st object mapping	RO	0x50070110 (Belt alarms)	unsigned32

- ## 0x1A01 : TPDO2 mapping parameters

Sub-index	Description	Access	Default value	Type
0x00	number of supported objects	R/W	2	unsigned8
0x01	1st object mapping	R/W	0x50050120 (Instantaneous flow rate)	unsigned32
0x02	2nd object mapping	R/W	0x50050420 (Belt speed)	unsigned32
0x03	3rd object mapping	R/W	0	unsigned32

- ## *0x1A02 : TPDO3 mapping parameters*

Sub-index	Description	Access	Default value	Type
0x00	number of supported objects	R/W	2	unsigned8
0x01	1st object mapping	R/W	0x50060120 (Totalizer value in weight unit x1000)	unsigned32
0x02	2nd object mapping	R/W	0x50060210 (Complementary totalizer value in weight unit)	unsigned32
0x03	3rd object mapping	R/W	0	unsigned32

- **Note:** TPDO2 and TPDO3 are programmable. TPDO1 is not programmable.
  To set a new mapping, the procedure is as following :
  - **Set *eNod4-B* in '*pre-operational mode*'.**
  - **Disable current TPDO mapping setting to zero number of supported objects (sub-index 0x00).**
  - **Write new mapping.**
  - **Write in sub-index 0x00 number of mapped objects**
  - **Save in EEPROM.**

7 PROFIBUS DPV1

7.1 Physical interface

An ***eNod4*** device compatible version can be connected to a Profibus DPV1 network thanks to the SUBD 9-pin female connector. ***eNod4*** supports baud rates between 9600 kbps and 12 Mbps with automatic detection.

For a complete description of the recommendations about ***eNod4*** Profibus DPV1 connection, please refer to "*Characteristics and functioning*" documentation.

7.2 GSD file

eNod4 capabilities for Profibus communication are described in the corresponding GSD file (that can be easily read as an ASCII text file). Here are the main information's contained in this file:

- *product and vendor identifications*
- *hardware and software versions*
- *supported baud rates*
- *description of the IN/OUT cyclic modules*
- *DPV0/DPV1 supported functions*
- *name of the associated bitmap*

7.3 Cyclic exchanges

7.3.1 Cyclic inputs modules

eNod4 GSD file defines several modules that can be integrated to the Profibus DP cyclic frames. All types of measurements processed by ***eNod4*** can be selected (net/gross/factory calibrated measurement). Moreover it is possible to add the data that contain information about the measurement or the device status.

7.3.2 Cyclic inputs/outputs modules

eNod4 cyclic outputs might be used in combination with cyclic inputs so as to request specific commands to ***eNod4*** (like tare, calibration...) or to modify the values of some settings (for example, set point values). A simple protocol described below allows writing or reading data by writing commands into ***eNod4*** cyclic outputs.

- The module "**Reg. Command/Response**" uses the eNod4 functional command mechanism defined in another chapter. The only difference concerns the "reset" and "restore default parameters" commands which cannot be sent via cyclic exchanges immediately after an eNod4 reboot. To use them with this cyclic exchange module, another command should be used before (cancel tare for example).

  The *'command/response register'* described by ***eNod4 GSD*** file is constituted by 2 IN/OUT bytes and is working almost the same way. Its functioning will be detailed later in this document.
- **The *'read/write request register'*** described by ***eNod4*** GSD file is constituted by 6 IN/OUT bytes :

IN	*OUT*
Transaction status (2 bytes)	***Transaction request (2 bytes)***
Data read/written (4 bytes)	***Data to be written (4 bytes)***

⇨ the Profibus DP master can transmit a read or write request to ***eNod4*** by writing a specific code (see the codes listed in the appendix) into the transaction request register.
⇨ for a write request, the 4 following OUT bytes can be used so as to enter the new value of the accessed setting
⇨ ***eNod4*** IN are then updated :

- Transaction status is set to 0xFFFF in case of an error otherwise it takes the same value as the one entered in the transaction request word.
- For a read transaction, the value of the requested setting is copied into the four IN following bytes.
- For a write transaction the value of the data to be written is copied into the four IN following bytes.
- For 16-bits size data, the data is read/written through the 16 last bits. Ignore the 16 upper bits.

❑ **Note**: the transaction request register **must** be set to 0x0000 before every new transaction.

7.4 Acyclic exchanges

For DPV1 class 1 and class 2 masters, another mechanism is available to write/read data into/from ***eNod4*** device. Acyclic exchanges are supported authorizing a class 1 master to send acyclic requests so as to read or write data into memory area defined by a couple of slot/index.

A class 2 master can also send the same requests after initiating a connexion (up to 2) with ***eNod4***. The communication happens without disturbing the cyclic data exchange established between the device and the class 1 master.

❑ **Note:** be careful when using acyclic read/write requests. The 4-bytes size data are transmitted as 2x16 bits blocks that need to be inverted so as to keep their consistency. For example 500 000_d ($0007A120_H$) is read/written through acyclic requests as $A1200007_H$.

7.5 eNod4 Profibus DP features

7.5.1 Sync

The *Sync* command can be transmitted from a master to one or more slaves. When receiving this command, ***eNod4*** device is set into *Synchro* mode. Thus, the cyclic outputs of all the addressed slaves are maintained in their current state. The cyclic outputs state is not updated until a new *Sync* command is received. The *Synchro mode* stops upon reception of a *Unsync* command.

7.5.2 Freeze

The reception of a *Freeze* command causes ***eNod4*** cyclic inputs to be frozen. Their state is not updated until a new *Freeze* command is received. The *Frezze mode* stops upon reception of an *Unfreeze* command.

7.5.3 Fail-safe

eNod4 supports the Profibus DP *Fail-safe* mode. When the Profibus DP master asks ***eNod4*** to switch to *Fail-Safe* mode, ***eNod4*** functioning changes according to following rules:

⇨ ***eNod4*** Profibus DP cyclic inputs goes on being refreshed
⇨ ***eNod4*** Profibus DP cyclic outputs are no more received

eNod4 remains as a “read-only device” until the master goes back to a normal working mode.

7.5.4 Profibus DP standard and extended diagnoses

eNod4 diagnosis frame is composed of a standard and a specific (called extended) parts that allow to inform the Profibus DP master about the device functioning on the communication bus and about device internal errors. The Profibus diagnosis frame has the following structure:

bytes 1-4				*bytes 5-6*		*byte 7*	*bytes 8-9*
standard diagnosis						***length of extended diagnosis***	***extended diagnosis content***
status 1	*status 2*	*status 3*	*Address*	*Ident Hi*	*Ident Low*	***03***	*XX XX (see table below)*

⇨ **status 1 & status 2 bytes** : both bytes describe ***eNod4*** current state from the Profibus point of view. Bit b_3 of status 1 byte is set to 1 if the extended diagnosis contains one or several errors.

⇨ **status 3 byte** : always 00_H

⇨ **Address** : Profibus address if the DP master that parameterized ***eNod4*** device

⇨ **Ident. High/Ident. Low bytes** : contain ***eNod4*** ident number ($0D2D_H$)

⇨ **extended diagnosis content** :

bits	*meaning*	*note*
b_0		
0	*/*	
1	*input analog signal out of the A/D conversion range (negative quadrant)*	
b_1		
0	*/*	
1	*input analog signal out of the A/D conversion range (positive quadrant)*	
b_2		
0	*/*	*see §8 for the 'maximum capacity' setting description*
1	*gross meas.< (- max capacity)*	
b_3		
0		*see §8 for the 'maximum capacity' setting description*
1	*gross meas. > (max capacity)*	
b_4		
0	*EEPROM OK*	
1	*Default EEPROM*	
b_{15}..... b_5		
0	*reserved*	

7.6 Product identification

As a DPV1 compatible device, ***eNod4*** supports record 0 of Identification and Maintenance (I&M0). A Profibus master can access its content through standard DPV1 request.

7.7 Measurement transmission

In Profibus DPV1 communication protocol, measurements (except for tare value) can be included in the cyclic input frame. The modular slave structure allows the user to select the variables according to his needs. These data cannot be accessed through acyclic requests.

7.8 EEPROM error management

Functioning and calibration parameters are stored in EEPROM NOV RAM. After every reset the entireness of parameters stored in EEPROM is checked. If a default appears, measurements are set to 0xFFFF and default is pointed out into module 1 and bit b_4 of extended diagnoses register.

8 PROFIBUS MODULES LIST

Name	Input data size in byte	Data provided	Output data size in byte	Data consumed	Module ID
Status+Gross Meas. (6 bytes IN)	6	Measurement status (2 bytes) Gross measurement (4 bytes)	0	NA	1
StatIO+Net+Fact (10 bytes IN)	10	Inputs – Outputs levels (2 bytes) Net measurement (4 bytes) A/D converter points (4 bytes)	0	NA	2
Flow rates/Speed. (12 b IN)	12	Instantaneous flow rate (4 bytes) Average flow rate (4 bytes) Average belt speed (4 bytes)	0	NA	3
Output control (6 bytes IN)	6	Flow rate control output (4 bytes) Control output value (2 bytes)	0	NA	4
Totalization (10 bytes IN)	10	Totalization per belt revolution (4 bytes) Totalizer value in weight unit x1000 (4 bytes) Complementary totalizer value (2 bytes)	0	NA	5
Cmd/Resp Reg (2 b IN/OUT)	2	Response register	2	Command register (see functionals commands)	6
R/W req Reg. (6 b IN/OUT)	6	Transaction status (2 bytes) Data read/written (4 bytes)	6	Transaction request (2 octets) Data to be written (4 octets)	7
Status/Errors (12 b IN)	12	Belt alarms register (2 bytes) Belt status register (2 bytes) Errors counter (2 bytes) Batch progression % (2 bytes) Dosing weight deviation	0	NA	8

Name	Input data size in byte	Data provided	Output data size in byte	Data consumed	Module ID
		(4 bytes)			
Other totals (8 b IN)	*8*	*Great total in weight unit x1000 (4 bytes)*	*0*	*NA*	*9*
		General total in weight unit x1000 (4 bytes)			
Ana. Output (2 b IN/OUT)	*2*	*External value to control analog output (2 bytes)*	*2*	*External value to control analog output (2 bytes)*	*10*

The module "Cmd/Resp Reg" uses the mechanism of ***eNod4*** functional commands defined in another chapter. The only difference is for "reset" and "Restore default settings" commands which cannot be sent via cyclic exchanges immediately after a restart of ***eNod4***. To be able to use these commands, it must first be processed another command ("cancel Tare" for example).

Note: The "Command register" data **must be** set to 0x0000 before each new command.

The "Ana. Output(2 b IN/OUT)" module allows writing directly the analog output value. This is only possible when the analog output function assignment is set to « Level on request ».

9 MEASUREMENT AND STATUS

Name	Modbus address	CANopen index/sub-index	Profibus Cyclic IN/OUT	Acyclic DPV1 slot/index	Type	Access
Measurement status	0x007D	0x5003 / 0x00 (M)	Module 1 (2 first bytes)	/	Uint	RO
Gross measurement	0x007E	0x5001 / 0x00 (M)	Module 1 (4 last bytes)	/	Long	RO
Tare value	0x0080	0x5004 / 0x01 (M)	R: 0x0470	/	Long	RO
Net measurement	0x0082	0x5000 / 0x00 (M)	Module 2 (4 bytes from 3th)	/	Long	RO
Factory calibrated points	0x0084	0x5002 / 0x00 (M)	Module 2 (4 last bytes)	/	Long	RO
Preset Tare	0x0097	0x5004 / 0x02 (M)	R: 0x04C4 W: 0x04C5	0x03 / 0x08	Ulong	RW
Defective measurement debounced time	0x0A48	0x4509/0x06	R:0x02EC W:0x02ED	0x06 / 0x0D	Uint	RW
Defective measurement alarm activation time	0x0A49	0x4509/0x07	R:0x02EE W:0x02EF	0x06 / 0x0E	Uint	RW
Sensor input control reference	0x0A44	0x5004 / 0x03 (M)	R : 0x044C W: 0x044D	0x0A / 0x00	long	RW
Sensor input control result	0x0A46	0x5004 / 0x04 (M)	R : 0x024E	0x0A / 0x01	Int	RO
Sensor input control result max. tolerance	0x0A47	0x5004 / 0x05	R: 0x02F8 W: 0x02F9	0x0A / 0x07	Uint	RW

9.1 Measurement transmission

The ***eNod4*** transmits measurement after signal and data processing through different protocols available. The accessible variables are:

9.1.1 Gross measurement

The *'gross measurement'* stands for the digital value after measurement scaling. It is affected by all the *'zero'* functions (power-up zero, zero tracking and zero requests).

9.1.2 Net measurement

The *'net measurement'* stands for the digital value after measurement scaling and tare subtraction.

9.1.3 Tare value

The *'tare value'* stores the calibrated value that is subtracted from the *'gross measurement'* so as to give the *'net measurement'*.

9.1.4 Factory calibrated points

The *'factory calibrated points"* contains the measurement value without the user calibration layer. It is directly linked to the analog input voltage.

9.1.5 Preset Tare value

A previous calculated tare can be restored using this variable.

9.1.6 Measurement status

The measurement status contains information on eNod4 measurement parameters.

The *'measurement status'* bytes contain information about every measurement processed by ***eNod4***. See the flags meaning in the table below:

bits	Meaning	Note
b_1 b_0		
00	*gross measurement*	*only in SCMBus/fast communication protocols* *not significant otherwise (00)*
01	*net measurement*	
10	*factory calibrated measurement*	
11	*tare value*	
b_3 b_2		
00	*measurement OK*	
01	*Defect: sensor input control result out of tolerances OR Sensor input control command in progress OR failed (timeout) OR Sensor input reference command in progress*	*causes a logical output assigned to the 'defective measurement' function to be set active. Causes the analog output assigned to a weight or flow rate image to be set in error mode.*
10	*gross meas.< (- max capacity) OR gross meas. > (max capacity)*	
11	*analog signal out of the A/D converter input range*	
b_4		
0	*motion*	*causes an output assigned to the 'motion' function to be set active*
1	*no motion*	
b_5		
0	*measurement out of the ¼ of division*	
1	*zero in the ¼ of division*	
b_6		
0	*EEPROM OK*	*See Note 1*
1	*EEPROM failure*	
b_7		
0	*reserved*	*1 in SCMBus and fast SCMBus, 0*

bits	Meaning	Note
1		*otherwise*
b_8		
0	*IN1 logical level*	
1		
b_9		
0	*IN2 logical level*	
1		
b_{10}		
0	*OUT1 logical level*	
1		
b_{11}		
0	*OUT2 logical level*	
1		
b_{12}		
0	*OUT3 logical level*	
1		
b_{13}		
0	*OUT4 logical level*	
1		
b_{14}		
0	*no tare*	
1	*at least a tare has been processed*	
b_{15}		
0	*reserved*	*1 in SCMBus and fast SCMBus, 0 otherwise*
1		

Note 1: Functioning and calibration parameters are stored in EEPROM. After every reset the entireness of parameters stored in EEPROM is checked. If a defect appears, measurements are set to 0xFFFF and defect is pointed out in measurement status. Causes a logical output assigned to the 'defective measurement' function to be set active. Causes the analog output assigned to a weight or flow rate image to be set in error mode.

9.2 Weighing diagnosis

9.2.1 Global weighing diagnosis

An internal alarm flag reflects the integrity of the whole measurement chain. It's used to set logical output active or optional analog output in an error mode in order to warn about any defection on the measurement chain (defective measurement).

This variable is set active when at least one of the followings conditions occurs:

- all that set bit2 or bit3 of ***Measurement status:***
  - sensor input control result out of tolerances
  - sensor input control command in progress
  - sensor input control command failed (timeout)
  - sensor input reference command in progress
  - gross meas. < (- max capacity)
  - gross meas. > (max capacity)
  - analog signal out of the A/D converter input range
- the one that set bit6 of ***Measurement status:*** EEPROM failure

This internal alarm flag is featured with adjustable specific de-bounced time and minimal activation time:

9.2.1.1 Defective measurement debounced time

The internal alarm flag is set active only after error conditions have always been true during this de-bounced time. It's expressed in ms.

9.2.1.2 Defective measurement alarm activation time

The internal alarm flag remains active for this minimal *"defective measurement alarm activation time"* when it come to be active and whatever the error conditions are during activation. It is expressed in ms.

9.2.2 Sensor input control

eNod4 features a weighing diagnosis system allowing to check the integrity of analog sensor input by electrically simulating a load, resulting to a simulated weight value. This diagnostic system can be used together with the others defects detection systems in order to achieve overall integrity check of the measurement chain. This system involves two phases initiated by the user:

- The first, just after user calibration, allows taking a simulated reference weight value when the measuring chain integrity is OK.
- The second, when the user wants to check the integrity of the system, allows to make the difference between a new simulated weight value and the reference. Then this difference can be compared with a dedicated maximum tolerance value.

9.2.2.1 Sensor input control reference

Reference value expressed in factory calibrated points for the sensor(s) input control test. The value is automatically determined and stored after executing the ***sensor input reference*** command. When the ***sensor input reference*** command is in progress the bits b3b2 in the ***Measurement status*** are set to 0b01. Its default value is zero.

9.2.2.2 Sensor input control result

Result of sensor(s) input control test expressed in 1/10 of user weight unit. Its value is automatically determined and stored after executing the ***sensor input control*** command. This test result represents the weight difference between the reference value and the current test value. It is set to **-1** when the ***sensor input control*** command is in progress or the command failed, these conditions cause the bits b3b2 in the ***Measurement status*** to be set to 0b01. Its default value is zero.

9.2.2.3 Sensor input control result max. tolerance

The ***Sensor input control result*** variable is compared with the ***Sensor input control result max. tolerance*** parameter which is expressed in 1/10 of user weight unit and has a default value of 30. If the ***sensor input control result*** value is greater than or equal to ***Sensor input control result max. tolerance*** then the bits b3b2 in the ***Measurement status*** are set to 0b01.

10 PROCESSING FUNCTIONAL COMMANDS

Name	*Modbus address*	*CANopen index/sub-index*	*Profibus Cyclic IN/OUT*	*Acyclic DPV1 slot/index*	*Type*	*Access*
Command register	*0x0090*	*0x2003 / 0x00 (M)*	*Module 6*	*/*	*Uint*	*RW*
Response register	*0x0091*	*0x2004 / 0x00 (M)*	*Module 6*	*/*	*Uint*	*RO*

10.1 Principles

eNod4 device is able to handle several functional commands thanks to a couple of registers (except in SCMBus protocols):

***the command register** : dedicated to accept the functional commands*

***the response register :** gives the state of the command currently being processed by **eNod4** (no command, in progress, finished, failed)*

- **00_H** ⇒ free to accept a new command
- **01_H** ⇒ command execution in progress
- **02_H** ⇒ command execution complete
- **03_H** ⇒ error during command execution

Note 1: IMPORTANT except in SCMBus/fast SCMBus protocols, to accept a new command, the command register **must be set to 00_H** first. This causes the response register to be set back to **00_H.**

10.2 Functional commands list

Functional command	Command code	Note
Set to idle (00_H) response register	00_H	See § above note 1
reset*	$D0_H$	
EEPROM storage	$D1_H$	
Restore default settings	$D2_H$	
Zero*	$D3_H$	
Tare*	$D4_H$	
Cancel tare*	$D5_H$	
Cancel current command	$D6_H$	
Theoretical scaling	$D7_H$	
Zero adjustment	$D8_H$	
Start physical calibration	$D9_H$	Physical calibration procedure
Calibration zero acquisition	DA_H	
Segment 1 acquisition	DB_H	
Segment 2 acquisition	DC_H	
Segment 3 acquisition	DD_H	
Store calibration	DE_H	end of calibration (physical/theoretical) procedure
OUT1 activation/deactivation*	$E6_H$	only possible if the outputs are assigned to the associated function
OUT2 activation/deactivation*	$E7_H$	
OUT3 activation/deactivation*	$E8_H$	
OUT4 activation/deactivation*	$E9_H$	
zero offset adjustment	$F0_H$	
Dynamic zero	$F1_H$	
Preset tare*	$F2_H$	
Init speed sensor calibration	$F3_H$	
End speed sensor calibration	$F4_H$	
Init belt length calibration	$F5_H$	
End belt length calibration	$F6_H$	
Calibration of flow rate	$F8_H$	
Flow rate correction	$F7_H$	
PID parameters auto-adjustment	$F9_H$	
STOP PID	$C9_H$	
RESTART PID	CA_H	

Functional command	Command code	Note
Sensor input reference	EF_H	
Sensor input control	FD_H	
Dosing / batch start/resume	$E4_H$	
Dosing / batch stop	$E5_H$	
Clear totalization and error counter	DF_H	
Clear great total	ED_H	
Clear general total	EE_H	

Note: Only the commands with a * can be handled by ***eNod4*** in SCMBus and fast SCMBus protocols.

10.3 Functional commands description

10.3.1 Reset

The *'reset'* functional command execution is similar to the device power-up. This reboot phase is necessary if the address or/and the baud rate are modified and some settings changes are only taken into account after an EEPROM storage followed by a reset.

10.3.2 EEPROM storage

eNod4 configuration and calibration are stored in a non-volatile memory (EEPROM). If changes are made in the device configuration, sending to ***eNod4*** the *'EEPROM storage'* functional command will allow ***eNod4*** to keep these modifications after a power shutdown or the reception a *'reset'* functional command.

Moreover the settings listed below need to be stored and will only be taken into account at the next device reboot:

- span adjusting coefficient
- calibration place **g** value
- place of use **g** value
- stability criterion
- legal for trade activation switch
- power-up zero
- A/D conversion rate
- Functioning mode and communication protocol

10.3.3 Restore default settings

The *'restore default settings'* command causes ***eNod4*** to be set back to its default configuration. The default configuration corresponds to the one on delivery that means with factory settings. Be careful when using this command, all the default settings are recovered including the stored calibration and the legal for trade indicators.

Note: this functional command is not available in CANopen® communication protocol.

10.3.4 Zero

When receiving a *'zero'* functional command**,** ***eNod4*** acquires a volatile zero (gross measurement is set to 0) value if the following conditions are respected:

- measurement is stable
- Current gross measurement is within a ±10% (±2% if the legal for trade option is enabled) range of the *'maximum capacity'*.

Otherwise, after five seconds the command is cancelled and an execution error is reported.

10.3.5 Tare

When receiving a *'tare'* functional command***, eNod4*** acquires a volatile tare (net measurement is set to 0) value if the measurement is stable otherwise, after five seconds the command is cancelled and an execution error is reported. If the tare acquisition is successful b_{14} bit of the *'measurement status'* is set to 1.

10.3.6 Cancel tare

This command erases the current tare value if at least one tare has been previously processed. It also causes b_{14} bit of the *'measurement status'* to be set back to 0.

10.3.7 Cancel last command

This command sets the response register to **00_H** and allows ***eNod4*** to ignore the functional command previously received (for example to exit a sequential procedure like a physical calibration).

10.3.8 Theoretical scaling

The *'theoretical scaling'* functional command involves the *'maximum capacity'* and *the 'sensor sensitivity'* settings. When used, this command realizes an automatic scaling to migrate from the factory calibration to the user calibration (see §8). This calibration must then be saved by sending to ***eNod4*** the '*store calibration'* functional command. Using the *'zero adjustment'* functional command is also recommended so as to completely adapt ***eNod4*** to the application.

10.3.9 Zero adjustment

The *'zero adjustment'* functional command allows the user to set his calibration zero value by asking ***eNod4*** to acquire the current factory calibrated measurement. This acquisition duration depends on the measurement stability; if stability is not reach after 10 seconds, '*zero adjustment*' command is cancelled and an execution error is reported. If it is correctly achieved, this calibration zero modification must then be saved by sending to ***eNod4*** the '*store calibration'* functional command. This functional command can be used any time and has no effect on the user-span that can have been previously configured through a physical or a theoretical calibration procedure.

10.3.10 Start physical calibration

In order to handle a physical calibration with 1 up to 3 know references, ***eNod4*** first must be told to enter the physical calibration mode. It is the first step of a sequential procedure.

10.3.11 Calibration zero acquisition

The *'calibration zero acquisition'* is the second step of the physical calibration procedure. It can only be used if the *'start physical calibration'* functional command has been previously received. This acquisition duration depends on the measurement stability; if stability is not reach after 10 seconds, '*calibration zero acquisition*' command is cancelled and an execution error is reported.

Note: In specific cases (silo for example), this step is not mandatory because it is possible to command a "zero adjustment" when the silo is empty.

10.3.12 Segment 1 acquisition

It consists in applying a known reference on the sensor then sending the *'segment 1 acquisition'* functional command. This acquisition duration depends on the measurement stability; if stability is not reach after 10 seconds, *'actual segment acquisition'* command is cancelled and an execution error is reported.

10.3.13 Segment 2/3 acquisition

Only if the *'calibration zero acquisition'* and *"Segment 1 acquisition"* are successful, next step consists in applying a known reference on the sensor then sending the 'segment X acquisition' functional command where X depends on the value stored in the 'number of calibration segments' register. This acquisition duration depends on the measurement stability; if stability is not reach after 10 seconds, *'actual segment acquisition'* command is cancelled and an execution error is reported.

10.3.14 Store calibration

Only if the *'segment 1/2/3 acquisition'* is successful, next step consists in validating the new calibration by storing the zero and the span that have been determined in EEPROM.

Note: This functional command has to be transmitted at the end of a physical calibration, after a *'zero adjustment'*, a *'theoretical scaling'* or a *'zero offset'*.

10.3.15 Logical outputs 1-4 activation/deactivation

If the corresponding logical outputs are assigned to the *'level on request'* function, they can be enabled/disabled by transmitting one of these functional commands. Upon first reception, the corresponding output is enabled and on next reception it will be disabled. If the requesting logical output is assigned to the wrong function, ***eNod4*** reports an error.

10.3.16 Zero offset

It is also possible to adjust the calibration zero value without acquiring a new one. By entering a positive or negative value into the *'delta zero'* register, the user can quantify the offset (in factory calibrated points) that has to be added or subtracted from the actual calibration zero. This calibration zero modification must then be saved by sending to ***eNod4*** the *'store calibration'* functional command.

10.3.17 Dynamic zero

In belt mode and when the system is running, after receiving a "*dynamic zero"* command ***eNod4*** calculates the average of integrated weight per length during the belt revolution time. A new zero is then used if it is within a "*dynamic zero band"* parameter range. For more accurate result, dynamic zero can be done within several belt revolutions specify by *"number of revolutions"* parameter.

- **Note 1:** Stability is not necessary
- **Note 2:** Dynamic zero acquisition can also be launch by an input assigned to this function.
- **Note 3:** Dynamic zero procedure can be interrupt by sending *"cancel current command"*.

10.3.18 Preset tare

With this command it is possible to retrieve a tare value defined previously.

Important: Preset tare value must be stored in corresponding parameter before to send this command.

10.3.19 Sensor input reference

Sensor input reference command will cause ***eNod4*** to handle special sequence to acquire ***sensor input control reference*** value of the load cell sensor input. This command must not be realized when any process cycle that use weight is in progress (because weight variables do not reflect the real weight whilst command is in progress). This command can fail (error in response register) in case of stability timeout on sensor input. The execution time of this command depends on the weight filtering settings. For any further information about this functionality and result variables see "Weighing diagnosis" § in the MEASUREMENT AND STATUS §.

10.3.20 Sensor input control

Sensor input control command will cause eNod4 to handle special test on sensor input and to deliver a test result. This command must not be realized when any process cycle that use weight is in progress (because weight variables do not reflect the real weight whilst command is in progress). This command can fail (error in response register) in case of stability timeout on sensor input. The execution time of this command depends on the weight filtering settings. For any further information about this functionality and result variables see "Weighing diagnosis" § in the MEASUREMENT AND STATUS §.

10.3.21 Init speed sensor calibration

When the system is running empty, this command will cause ***eNod4*** to initiate belt speed calibration sequence.

An internal speed sensor pulses counter is restarting and the calibration procedure duration will depend of the current belt speed, the "*total belt length"* and the "*number of revolutions"* to handle.

Important: Before initiated speed calibration procedure, use has to configure the following parameters:

- *total belt length*
- *number of revolutions*

10.3.22 End speed sensor calibration

When the configured number of revolution is realized, this command will cause ***eNod4*** to achieve and to exit speed calibration sequence. The new ***"Pulses factor for speed"*** is calculated and the speed given by ***eNod4*** will match the real belt speed. Also, ***eNod4*** will know how long for the belt revolution.

10.3.23 Init belt length calibration

When the system is running empty, this command will cause ***eNod4*** to initiate belt length calibration sequence.

An internal timer is restarting and the calibration procedure duration will depend of the current belt speed, the belt length and the *number of revolutions* to handle.

Important: Before initiated belt length calibration procedure, user has to configure the following parameters:

- *Pulses factor for speed(could be found in speed sensor user manual or determinate by* ***eNod4****)*
- *number of revolutions*

10.3.24 End belt length calibration

When the configured number of revolution is realized, this command will cause ***eNod4*** to achieve and to exit belt length calibration sequence. The new ***"total belt length"*** is calculated and ***eNod4*** will know how long for the belt revolution.

10.3.25 Flow rate correction

Correction factor can be used to correct deviations in the total dosed amount by compensating for mechanical variations. At receiving a "*flow rate correction"* command ***eNod4*** recalculate a correction factor by calculating:

New Correction = Checked Batch Total * Correction / Last batch total.

The next batch the Batch Total and Checked Batch Total should be closer together.

Important: Before initiated this command, user has to configure the "*Checked Batch*" parameter.

10.3.26 Start (batch/system)

In *batch mode* this command launches or resumes a cycle. A cycle can also be launch by an input assigned to this function. If the start is given by the *command register*, ***eNod4*** responds through the *response register*, the state of the control cycle start. If the cycle has been started, then the bits of "*belt status word"* will be positioned (see § belt status word).

In *weigh feeder mode* and if ***eNod4*** analog output is assigned to *"flow rate control output"* and drives the *"conveyor speed"*, this command will start the belt system at *"nominal speed"* configured. In that case, for security reason a logical output can be assigned to "*buzzer"* function and the duration is configured through "conveyor starting alarm duration" parameter.

10.3.27 Stop (batch/system)

In *batch mode* this command allows stopping the batch cycle. It is also possible to order a 'stopping of cycle' with an input set to this function. At receiving of this command, the scale material flow stops automatically.

In *weigh feeder mode* and if ***eNod4*** analog output is assigned to *"flow rate control output"* and drives the *"conveyor speed"*, this command will also stop the belt system.

10.3.28 Clear totalization & errors counter

This command triggers the main totalizer reset. The two parts of the totalizer, the main in ***weight unit*** x 1000, and the complementary part in ***weight unit*** are then reset.
The variable ***errors counter*** is also reset.

10.3.29 Calibration of flow rate

So that ***eNod4*** can carry out an expected flow rate dosing in the best conditions possible, the flow rate output control calibration is required. This also applies when ***eNod4*** is used both as belt scale or belt weigh feeder.
From this calibration will depend the accuracy of the flow rate obtained and on the actuation time delay, if a PID regulator controls it. This calibration is carried out in minimum two segments by the variable ***segments number for the calibration curve of flow rate***. In case the extraction device has a nonlinear response it is recommended to define maximum segments for the flow rate calibration.
If the control of extraction device is directly provided by ***eNod4*** through a control analog output in *current* or *voltage*, analog output of ***eNod4*** must be allocated to ***level on request*** function first.
For each calibration point of the variable ***control output value,*** read the appropriate *average flow rate.* Then provide each of the ***Calibration of control output point n / analog output*** and ***Calibration of flow rate point n matching with control output value.*** Validate the flow rate calibration by sending "***calibration of flow rate" command***.
Allocate in the end the *current* or *voltage* analog output of ***eNod4*** to ***flow rate control output*** function***.***

10.3.30 PID parameters auto-adjustment

The configuration of the plan of regulation can be made in a totally automatic way. The behavior of PID regulation (slow, fast or stable) must beforehand be configured. Also, you have to configure reference flow rate used in PID auto-adjustment.

So that this plan of regulation works, it is necessary that weight calibration indication as well as flowrate calibration is beforehand realized.

"PID parameters auto-adjustment" command will cause ***eNod4*** to perform special sequence on belt system and to calculate optimized PID coefficients ***Kp,*** ***Ti*** and ***Td***.
It is strongly recommended that ***Td*** parameter does not exceeded value **5**.

10.3.31 STOP & RESTART PID

We can STOP and RESTART PID at each cycle step. When PID is stopped, analog output pilots belt to nominal value.

10.3.32 Clear great total

"Clear great total" command allows only resetting *"**Grand total (in weight unit x1000)"*** totalization result.

10.3.33 Clear general total

"Clear general total" command allows only resetting *"**General total (in weight unit x1000)"*** totalization result.

11 CALIBRATION SETTINGS AND PROCEDURES

Name	Modbus address	CANopen index/sub-index	Profibus Cyclic IN/OUT	Acyclic DPV1 slot/index	Type	Access
Maximum capacity	*0x000C*	*0x3002 / 0x00 (M)*	*R : 0x0420* *W: 0x0421*	*0x02 / 0x00*	*Ulong*	*RW*
Number of calibration segments	*0x000E*	*0x3000 / 0x00*	*R : 0x0222* *W: 0x0223*	*0x02 / 0x01*	*Uint*	*RW*
Calibration load 1	*0x000F*	*0x3001 / 0x01 (M)*	*R : 0x0424* *W: 0x0425*	*0x02 / 0x02*	*Ulong*	*RW*
Calibration load 2	*0x0011*	*0x3001 / 0x02*	*R : 0x0426* *W: 0x0427*	*0x02 / 0x03*	*Ulong*	*RW*
Calibration load 3	*0x0013*	*0x3001 / 0x03*	*R : 0x0428* *W: 0x0429*	*0x02 / 0x04*	*Ulong*	*RW*
Sensor sensitivity	*0x0015*	*0x3004 / 0x00 (M)*	*R : 0x042A* *W: 0x042B*	*0x02 / 0x05*	*Ulong*	*RW*
Scale interval	*0x0017*	*0x3003 / 0x00*	*R : 0x022C* *W: 0x022D*	*0x02 / 0x06*	*Uint*	*RW*
Zero calibration	*0x0018*	*0x3005 / 0x00*	*R : 0x0434* *W: 0x0435*	*0x03 / 0x04*	*Long*	*RW*
Span coefficient 1	*0x001A*	*0x3006 / 0x04*	*R : 0x0436* *W: 0x0437*	*0x03 / 0x05*	*Float*	*RW*
Span coefficient 2	*0x001C*	*0x3006 / 0x05*	*R : 0x0438* *W: 0x0439*	*0x03 / 0x06*	*Float*	*RW*
Span coefficient 3	*0x001E*	*0x3006 / 0x06*	*R : 0x043A* *W: 0x043B*	*0x03 / 0x07*	*Float*	*RW*
Span adjusting coefficient	*0x0020*	*0x3006 / 0x01 (M)*	*R : 0x042E* *W: 0x042F*	*0x03 / 0x00*	*Ulong*	*RW*
Calibration place g value	*0x0022*	*0x3006 / 0x02*	*R : 0x0430* *W: 0x0431*	*0x03 / 0x01*	*Ulong*	*RW*
Place of use g value	*0x0024*	*0x3006 / 0x03*	*R : 0x0432* *W: 0x0433*	*0x03 / 0x02*	*Ulong*	*RW*
Zero offset	*0x0092*	*0x2500 / 0x00 (M)*	*R : 0x0472* *W: 0x0473*	*0x03 / 0x03*	*Long*	*RW*

11.1 Principles

eNod4 is factory calibrated to deliver **500 000 counts for 2mV/V** with a load cell on the **A3** input connector.

The measurement scaling in ***eNod4*** can be adapted to his application by the user. Some settings and the 2 calibration methods allow the user to define his specific span according to his sensors characteristics.

When using eNod4 for legal for trade purpose, it is imperatively required to activate the legal for trade switch BEFORE any calibration procedure (cf § legal for trade switch).

11.2 Calibration methods

Measurement scaling can be defined using one of the two following methods:

- **Theoretical calibration** involving the sensitivity of the sensor and a user-defined corresponding capacity
- **Physical calibration** involving 1, 2 or 3 know loads (for a load cell)

Both can be achieved thanks to the functional commands.

11.3 Settings description

11.3.1 Maximum capacity

The *'maximum capacity'* stands for the maximum sensor/load cell signal range. When the absolute value of the gross measurement exceeds its value plus 9 divisions, the b_3 bit (positive overloading) or the b_2 bit (negative overloading) of the measurement status is set to 1 (it can activate a logical output if it is assigned to the *'defective measurement'* function).

The zero acquisition (on request or at power-up) is done only if the gross measurement value is contained between a ±10% range of the *'maximum capacity'* (±2% if the *legal for trade* option is active).

The *'maximum capacity'* setting also allows calibrating ***eNod4*** in case of a theoretical calibration in association with the sensor sensitivity. Measurement scaling will be automatically adapted so as to deliver a gross measurement value equivalent to the *'maximum capacity'* for an analog signal corresponding to the sensor sensitivity.

After a theoretical calibration, the maximum capacity can be changed to fit to the application.

Admitted values : from 1 up to 10000000.

11.3.2 Number of calibration segments

The *'number of calibration segments'* defines how many calibration segments are used during the physical calibration procedure. Linear installations only need one segment.

Admitted values : from 1 up to 3.

11.3.3 Calibration loads 1/2/3

Before starting a physical calibration procedure, each calibration segment must be given a corresponding user value (for example, 1000 points for a 1 kg load).

Admitted values : from 1 up to 10000000.

11.3.4 Sensor sensitivity

The *'sensor sensitivity'* setting is used to achieve a theoretical calibration. The stored value for this parameter is the load cell sensitivity in **mV/V**.

The user can adapt the value delivered by ***eNod4*** for the associated signal using the *'maximum capacity'* and the *'sensor sensitivity'*.

This setting is expressed with a 10^{-5} factor (197500 is equivalent to a 1.975 mV/V load cell sensitivity or a 1.975 V input voltage).

Admitted values : from 1 up to 1000000.

11.3.5 Scale interval

The *'scale interval'* is the minimal difference between two consecutive indicated values (either gross or net).

Modification of scale interval is taking into account after a new calibration.

Admitted values : 1/2/5/10/20/50/100

11.3.6 Zero calibration

Zero calibration value corresponds to the A/D converter points measured during the '*zero acquisition*' step of a physical calibration.

For a theoretical calibration this value must be set. It can be set automatically with the '*zero adjustment*' command.

Note: To be applied, any modification of this setting must be followed by an EEPROM back up and device reboots (hardware or software).

Admitted values : from 0 up to +-10000000

11.3.7 Span coefficients 1/2/3

These coefficients are computed and written during calibration process. Writing these coefficients could be done if you want to restore a previous calibration.

Note: To be applied, any modification of this setting must be followed by an EEPROM back up and device reboots (hardware or software).

Admitted values : different from 0.

11.3.8 Span adjusting coefficient

The *'span adjusting coefficient'* allows adjusting initial calibration. Adjustment applies linearly on the whole calibration curve. This coefficient has a 10^{-6} factor (1000000 is equivalent to a span adjusting coefficient that is equal to 1).

Note: To be applied, any modification of this setting must be followed by an EEPROM back up and device reboots (hardware or software).

Admitted values : from 900000 up to 1100000.

11.3.9 Calibration place g value / place of use g value

When the calibration place and the place of use of a measuring chain are different, a deviation can appear due to the difference of g (gravity) between the 2 places.

The ***eNod4*** calculates a ratio applied to the measure which compensates the difference of gravity between the 2 places.

The g value are expressed in 10^{-6} $m.s^{-2}$ (9805470 is equivalent to g = 9.805470 $m.s^{-2}$).

The ***eNodView*** software can help to determine the g value of a place.

Note: To be applied, any modification of this setting must be followed by an EEPROM back up and device reboots (hardware or software).

Admitted values : different from 0.

11.3.10 Zero offset

The *'Zero offset'* value contains the offset in factory calibrated points that can be added/subtracted (if its value is positive or negative) to the zero calibration value when using the *'zero offset'* functional command. Once the command has been successfully achieved, this register is set to 0.

Note: The 'Zero offset' value is not stored into EEPROM memory and is always equal to 0 after a device power-up or a software reset

Admitted values: different from 0.

12 FILTERS

Name	Modbus address	CANopen index/sub-index	Profibus Cyclic IN/OUT	Acyclic DPV1 slot/index	Type	Access
A/D conversion rate	*0x0036*	*0x4000 / 0x00*	*R : 0x0240* *W: 0x0241*	*0x04 / 0x00*	*Uint*	*RW*
filters activation	*0x0037 LSB*	*0x4001 / 0x01 (byte)*	*R : 0x0242* *W: 0x0243*	*0x04 / 0x01*	*Byte*	*RW*
Low-pass order	*0x0037 MSB*	*0x4001 / 0x02 (byte)*			*Byte*	
Low-pass cut-off frequency	*0x0038*	*0x4001 / 0x03*	*R : 0x0244* *W: 0x0245*	*0x04 / 0x02*	*Uint*	*RW*
Band-stop high cut-off frequency	*0x0039*	*0x4001 / 0x04*	*R : 0x0246* *W: 0x0247*	*0x04 / 0x03*	*Uint*	*RW*
Band-stop low cut-off frequency	*0x003A*	*0x4001 / 0x05*	*R : 0x0248* *W: 0x0249*	*0x04 / 0x04*	*Uint*	*RW*
Average weight filter depth	*0x0058*	*0x4001 / 0x06*	*R : 0x0274* *W: 0x0275*	*0x07 / 0x00*	*Uint*	*RW*
Average process data filter depth	*0x0059*	*0x4703 / 0x09*	*R : 0x0278* *W: 0x0279*	*0x07 / 0x02*	*Uint*	*RW*

12.1 Principles

eNod4 contains 4 filtering layers that are user-configurable :

- filtering related to the A/D conversion rate (with rejection of the mains frequency)
- a low-pass Bessel-type filter
- a band-stop filter
- a Moving average weight filter

Except for the A/D conversion rate that is always enabled, none of these filters is mandatory. However, to perform accurate measurements we recommend setting a combination of filters. ***eNodView*** software may be helpful in designing the best filter configuration for the application.

12.2 Settings list

Here is the list of the settings that have an impact on the filters configuration:

12.3 Settings description

12.3.1 A/D conversion rate

It contains a code which represents the A/D conversion rate and the rejection. See table below:

b_4	Rejection	
0	*60 Hz*	
1	*50 Hz*	
$b_3\ b_2\ b_1\ b_0$	A/D conversion rate (measures/s)	
	50-Hz rejection	60-Hz rejection
0000	*100*	*120*
0001	*50*	*60*
0010	*25*	*30*
0011	*12.5*	*15*
0100	*6.25*	*7.5*
1001	*1600*	*1920*
1010	*800*	*960*
1011	*400**	*480**
1100	*200*	*240*

(*) In ***belt mode***, *A/D conversion rate* is fixed to ***400 measures/s*** and cannot be modified.

Note: To be applied, any modification of this setting must be followed by an EEPROM back up and device reboots (hardware or software).

12.3.2 Filters activation & order

This setting allows to define what filters are enabled in ***eNod4*** signal processing chain.

Note: the filters activation & order setting can be accessed through a 16-bits register except in CANopen® communication protocol where this word is divided into 2 8-bits registers:

b_0	Meaning
0	*band-stop filter disabled*
1	*band-stop filter enabled*
b_1	
0	*Reserved(not used)*
$b_{10}\ b_9\ b_8$	
000	*low-pass filter disabled*
010	*2nd order low-pass filter*
011	*3rd order low-pass filter*

Note: In CANopen® communication protocol (according to version), this word is divided into 2 bytes of 8-bits registers. Bits b8 to b15 are therefore equivalent to bits b0 to b7 of the corresponding address (see CANopen® Register table).

12.3.3 Low-pass filter cut-off frequency

This register contains the low-pass filter cut-off frequency expressed in Hz and multiplied by 100. That means that 690 is equivalent to 6.90 Hz. The value must be compliant with the table shown below.

Admitted values: from 10 up to 20000.

12.3.4 Limitations

Recursive filters like ***eNod4*** low-pass filters are computed according to the filter order, the desired cut-off frequency and the sampling rate. There are some limitations to respect in order to ensure a safe functioning of the signal processing. They are listed in the table below:

A/D conversion rate (meas/s)	*min low-pass cut-off frequency (Hz)*		*A/D conversion rate (meas/s)*	*min low-pass cut-off frequency (Hz)*	
	50 Hz rejection			***60 Hz rejection***	
	2nd order	***3rd order***		***2nd order***	***3rd order***
6.25	*0.10*	*0.10*	***7.5***	*0.10*	*0.10*
12.5	*0.10*	*0.10*	***15***	*0.10*	*0.15*
25	*0.10*	*0.15*	***30***	*0.15*	*0.20*
50	*0.15*	*0.25*	***60***	*0.20*	*0.30*
100	*0.25*	*0.50*	***120***	*0.30*	*0.60*
200	*0.50*	*1.00*	***240***	*0.60*	*1.20*
400*	*1.00*	*2.00*	***480****	*1.20*	*2.40*
800	*2.00*	*4.00*	***960***	*2.40*	*4.80*
1600	*4.00*	*8.00*	***1920***	*4.80*	*9.60*

(*) In ***belt mode***, *A/D conversion rate* is fixed to ***400 meas./s*** and cannot be modified.

12.3.5 Band-stop filter high cut-off frequency

This register contains the band-stop filter high cut-off frequency expressed in Hz and multiplied by 100. That means that 690 is equivalent to 6.90 Hz. **The value must be higher than the band-stop filter low cut-off frequency.**

Admitted values: from 10 up to 20000.

12.3.6 Band-stop filter low cut-off frequency

This register contains the band-stop filter low cut-off frequency expressed in Hz and multiplied by 100. That means that 690 is equivalent to 6.90 Hz. **The value must be lower than the band-stop filter high cut-off frequency.**

Admitted values: from 10 up to 20000.

12.3.7 Moving average weight filter depth

This filter can be set in cascade after previous filters. The Moving average filter is used to smooth the weight value in case of random interferences. This sliding average computes the mean of the 'n' last measures which are output of the previous filters, if enabled. A high filter depth will give a better stability, with a longer response time.

Filter depth admitted values: 0(disabled) up to 128.

12.3.8 Moving average flow rate/speed filter depth

Specific average filter might be applied on belt process data to have stable display. A high filter depth will give a better stability, with a longer response time.

Filter depth admitted values: 0(disabled) up to 128.

13 CONFIGURATION OF INPUT/OUTPUT

Name	Modbus address	CANopen index/sub-index	Profibus Cyclic IN/OUT	Acyclic DPV1 slot/index	Type	Access
Analog output functioning (IO+ version)	0x0040	0x4509 / 0x05	R:0x026C W:0x026D	0x06 / 0x0B	Uint	RW
Logical input 3 functioning (IO+ version)	0x0041 LSB	0x4501 / 0x04	R:0x026A W:0x026B	0x05 / 0x02	Byte	RW
Logical input 4 functioning (IO+ version)	0x0041 MSB	0x4501 / 0x05			Byte	RW
Logical input 1 functioning	0x0042 LSB	0x4501 / 0x02	R:0x0250 W:0x0251	0x05 / 0x00	Byte	RW
Logical input 2 functioning	0x0042 MSB	0x4501 / 0x03			Byte	RW
holding time	0x0043	0x4501 / 0x01	R:0x0252 W:0x0253	0x05 / 0x01	Uint	RW
Output 1 functioning	0x0044 LSB	0x4509 / 0x01	R:0x0254 W:0x0255	0x06 / 0x00	Byte	RW
Output 2 functioning	0x0044 MSB	0x4509 / 0x02			Byte	RW
Output 3 functioning	0x0045 LSB	0x4509 / 0x03	R:0x0256 W:0x0257	0x06 / 0x01	Byte	RW
Output 4 functioning	0x0045 MSB	0x4509 / 0x04			Byte	RW
Set point 1 high value	0x0046	0x4601 / 0x03	R:0x045A W:0x045B	0x06 / 0x02	Long	RW
Set point 1 low value	0x0048	0x4601 / 0x02	R:0x045C W:0x045D	0x06 / 0x03	Long	RW
Set point 2 high value	0x004A	0x4601 / 0x05	R:0x045E W:0x045F	0x06 / 0x04	Long	RW
Set point 2 low value	0x004C	0x4601 / 0x04	R:0x0460 W:0x0461	0x06 / 0x05	Long	RW
Set point 3 high value	0x004E	0x4609 / 0x03	R:0x0462 W:0x0463	0x06 / 0x06	Long	RW
Set point 3 low value	0x0050	0x4609 / 0x02	R:0x0464 W:0x0465	0x06 / 0x07	Long	RW
Set point 4 high value	0x0052	0x4609 / 0x05	R:0x0466 W:0x0467	0x06 / 0x08	Long	RW
Set point 4 low value	0x0054	0x4609 / 0x04	R:0x0468 W:0x0469	0x06 / 0x09	Long	RW
1&2 Set points functioning	0x0056 LSB	0x4601 / 0x01	R:0x0258 W:0x0259	0x06 / 0x0A	Byte	RW
3&4 Set points functioning	0x0056 MSB	0x4609 / 0x01			Byte	RW
External value to control analog output	0x0032	0x5050 / 0x00 (M)	R:0x023C W:0x023D + See modules list	0x06 / 0x0C	Uint	RW
Logical Inputs level	0x0094(LSB)	0x5100/0x00 (M)	Module 2	/	Byte	RO
Logical Outputs level	0x0094(MSB)	0x5200/0x00 (M)	Module 2	/	Byte	RO

13.1 Principles

eNod4 device is fitted with 2 logical inputs (4 logical inputs for IO+ version) and 4 logical outputs that are fully configurable.

13.1.1 Logical inputs

Each input can work individually in positive or negative logic. A holding time (de-bounced time) attached to all inputs can be configured.

Available functions see hereunder table:

Function	*Operating mode*		
	transmitter	*Belt scale*	*Belt weigh feeder*
none	•	•	•
tare	•	•	•
cancel tare	•	•	•
zero	•	•	•
transmit measurement	•		
measurement window	•		
dynamic zero		•	•
Start/Stop		•	•
Belt running detection		•	•
Clear totalization and errors counter		•	•
Sensor input control	•	•	•
Belt fault		•	•

Note: Most of functions set to logical inputs can also be sent by '*functional commands*'; for a precise description see § 'functional commands'.

- ## *None*

The input has no function

- ## *Tare*

A rising (positive logic) or a falling edge (negative logic) causes a tare function to be triggered.

- ## *Cancel tare*

A rising (positive logic) or a falling edge (negative logic) causes the current stored tare to be erased.

- ## *Zero*

A rising (positive logic) or a falling edge (negative logic) causes a zero function to be triggered.

- ## *Transmit measurement*

Only available in CANopen® and SCMBus/fast SCMBus protocols.

A rising (positive logic) or a falling edge (negative logic) triggers a measurement transmission.

- ***Measurement window***

Only available in SCMBus/fast SCMBus protocols. Measurements are continuously transmitted at a rate defined by the 'sampling period' while the input is maintained at the chosen level.

- ***Start/Stop (batch/system)***

If all the starting conditions are respected, a rising or a falling edge (according to the configured logic) on this input causes a new totalization cycle to be started.

In batch mode and when cleared totalization at starting new cycle option is enabled, the totalization result is set to zero at each start cycle.

If dosing cycle is running, the second edge of this command will stop the process.

- ***Belt running motion detection***

If speed sensor is broken or if no speed sensor is connected to ***eNod4*** device, this logical input might be assigned to ***motion detection*** function to enable totalization function.

- ***Clear/Reset totalization and errors counter***

Cleared totalization function can be assigned to logical input. You may clear total amount dosed and dosing errors counter at any time. At this input activation on rising or falling edge (according to the configured logic), the ***main total*** in ***weight unit x1000*** and the ***complementary weight*** value are reset to zero.

Also, ***dosing errors counter*** parameter is set to zero.

- ***Dynamic zero***

When the belt is running empty, this input activation will cause ***eNod4*** to perform conveyor zero function by measuring rate of flow of material.

Run the conveyor for several minutes to ensure the belt is empty and supple. The conveyor should be operating at ***normal speed*** throughout the calibration. Dynamic zero function duration will depend upon the ***number of revolution*** chosen, and belt length and speed. The procedure can be left to complete its cycles or send ***exit calibration*** command to interrupt.

- ***Sensor input control***

A rising (positive logic) or a falling edge (negative logic) triggers a test routine of the sensor input and produces a test result.

- ***Belt fault***

Belt Fault function should be assigned to logical input. At this input activation on rising or falling edge (according to the configured logic), **Belt fault alarm** is activated.

13.1.2 Analog output (IO+ version)

An optional analog board in *current* and *voltage* might be used with ***eNod4*** to provide IO+ version. This must be asked when ordering ***eNod4*** product.
Voltage output might be set either 0-5V or 0-10V, and the current output to 4-20mA, 0-24mA , 0-20mA or 4-20mA with alarm at 3.6mA. Both output (current and voltage) might separately be enable. Settings are effective after ***eNod4*** reset.

Analog output affectation function is common to both *current* and *voltage* output and might be assigned to followings:

function	*Operating mode*		
	transmitter	***Belt scale***	***Belt weigh feeder***
none	●	●	●
gross measurement	●	●	●
net measurement	●	●	●
level on request	●	●	●
flow rate control output		●	●
instant flow rate	●	●	●
average flow rate	●	●	●
average belt speed	●	●	●

- **None :** analog outputs have no function.
- **Gross measurement :** analog outputs can be assigned to gross measurement copy. Maximal level value is related to ***Maximum Capacity*** parameter and works in mono-quadrant functioning. Bi-quadrant option can only be applied to gross measurement copy. When this option is activated, the lowest value of current and voltage levels corresponds to -***MC*** and the highest value to +***MC***.
- **Net measurement :** analog outputs can be assigned to net measurement copy. Maximal value is related to ***Maximum Capacity*** parameter and works in mono-quadrant functioning only. The highest value of current and voltage levels corresponds to +***MC*** in only one quadrant.
- **Level on request :** analog outputs are driven by master requests through the ***external value to control analog output*** variable (in 0.01% of the full scale of current or voltage analog outputs).
- **Flow rate control output :** PID controller output data (activated or not) drives current or voltage outputs generally coupled to the extraction device. Data is expressed in 0.01% of the high value of current or voltage analog outputs. Maximal level output value corresponds to the maximal flowrate (see § flow rate calibration).
- **Instant flow rate :** analog outputs can be assigned to instant flow rate copy. Maximal level output value corresponds to the maximal flowrate (see § flow rate calibration).
- **Average flow rate :** analog outputs can be assigned to average flow rate copy. Maximal level output value corresponds to the maximal flowrate (see § flow rate calibration).
- **Average belt speed :** analog outputs can be assigned to average belt speed copy. Maximal level output value corresponds to the maximal belt speed (see § BELT OPERATING MODES).

When analog output is assigned to “Gross measurement”, “Net measurement”, “Instant flow rate” or “Average flow rate” its value jumps to a special error value when the internal alarm flag described in “Weighing diagnosis” § in the MEASUREMENT AND STATUS § is activated. This allows to warn about defection of the measurement chain.

The error value on analog output is defined depending on voltage or current settings as described in following table:

Setting	*Analog output error mode value*
0 - 5V	***5.5 V***
0 - 10V	***11 V***
4 - 20mA	***no output current***
0 - 20mA*	***no output current***
0 - 24mA*	***no output current***
4 mA - 20 mA with alarm at 3.6 mA	***3.6 mA, voltage output is deactivated (High-Z state)***

* no error detection possible in this setting

13.1.3 Logical outputs

Each output can work individually in positive or negative logic

The available functions are:

function	*Operating mode*		
	transmitter	***Belt scale***	***Belt weigh feeder***
none	•	•	•
set point	•	•	•
motion	•	•	•
defective measurement	•	•	•
input image	•	•	•
level on request	•	•	•
belt alarms		•	•
external totalizer		•	•
belt system running		•	•
batch in progress		•	•
batch result available		•	•
conveyor starting alarm		•	•
material TOR gate		•	•

- ***None***

The output has no function.

- ***Motion***

The output is dedicated to copying the stability flag level.

- ***Defective measurement***

The output level is set when the internal alarm flag described in "Weighing diagnosis" § in the MEASUREMENT AND STATUS § is activated. This allows to warn about defection of the measurement chain. Flowrate alarms are not considered as defective measurements.

- ***Set point***

Each output can be assigned to a configurable set point (set point 1 corresponds to output 1, set point 2 to output 2, set point 3 to output 3 and set point 4 to output 4).

- ***Input X image***

The output is dedicated to copying a logical input level (outputs 1 and 3 correspond to inputs 1 and 3, outputs 2 and 4 correspond to input 2 and 4).

- ***Level on request***

The input level is driven by master requests.

- ***Belt alarms***

The output is dedicated to copying alarm flags level. Alarms might be flow rate or belt speed or belt load or control output level or external totalizer overflow.

- ***External totalizer***

This output is dedicated to deliver pulses for external totalizer device.

If a logical output is assigned to *external totalizer* function, ***eNod4*** will send a pulse every time totalization result will reach a multiple of weight value defined in "***weight quantity per pulse on logical output"*** parameter.

- ***Belt system running***

In belt mode, indicates that a totalization function is activated.

- ***Batch in progress***

In batch mode, indicates that a batch cycle is in progress.

- ***Batch result available***

In batch mode, indicates the end of a cycle (batch target is reached).

- ***Conveyor starting alarm***

eNod4 analog output in current or voltage might be used to drive conveyor speed engine through invertor.

So that for security reason a logical output can be assigned to "*Conveyor starting alarm"* function and the duration is configured through "*conveyor starting alarm duration*" parameter.

- ***Material TOR gate***

Depending application, this output is dedicated to open and to close the belt material two states gate.

13.2 Settings description

13.2.1 Logical inputs assignment

The following table describes the possible assignments.

bits	*meaning*	*note*
$b_3 b_2 b_1 b_0$	***input 1&3 assignment***	
0000	*none*	*the input has no function*
0001	*tare*	*equivalent to the functions described in § functional commands*
0010	*zero*	
0011	*cancel tare*	
0100	*transmit measurement*/send TPDO2***	*data is transmitted on the bus at every rising or falling edge (depending on the chosen logical)* ***Transmitter mode only***
0101	*measurement window**	*Data is transmitted on the bus while the input is maintained at the right level (depending on the chosen logical). Transmission rate is fixed by the 'sampling rate' setting*
0110	*Dynamic zero*	*Same like equivalent functional command describe in § functional commands*
0111	*Start / Stop*	*Belt mode*
1000	*Belt running detection*	
1001	*Clear totalization and errors counter*	*Same like equivalent functional command describe in § functional commands*
1010	*Sensor input control*	*Same like equivalent functional command describe in § functional commands*
1011	*Belt fault*	*In this case, the input is linked to fault sensor. This function triggers alarm.*
b_4	***input 1&3 logical***	
0	*negative logic*	*defines the edge (or level) that triggers input function*
1	*positive logic*	
$b_6 b_5$	***measurement to be transmitted***	
00	*gross*	*only for SCMBus/fast SCMBus protocols, no effect otherwise*
01	*net*	
10	*factory calibrated measurement*	
bits	*meaning*	*note*
$b_{11} b_{10} b_9 b_8$ (or $b_3 b_2 b_1 b_0$ in CANopen®)	***input 2&4 assignment***	
0000	*none*	*the input has no function*

bits	meaning	note
0001	*tare*	*equivalent to the functions described in § functional commands*
0010	*zero*	
0011	*cancel tare*	
0100	*transmit measurement*/send TPDO3***	*data is transmitted on the bus at every rising or falling edge (depending on the chosen logical)*
0101	*measurement window**	*data is transmitted on the bus while the input is maintained at the right level (depending on the chosen logical). Transmission rate is fixed by the 'sampling rate' setting*
0110	*Dynamic zero*	*Same like equivalent functional command describe in § functional commands*
0111	*Start / Stop*	*Belt mode*
1000	*Belt running detection*	*Same like equivalent functional command describe in § functional commands*
1001	*Clear totalization and errors counter*	*Same like equivalent functional command describe in § functional commands*
1010	*Sensor input control*	*Same like equivalent functional command describe in § functional commands*
1011	*Belt fault*	*In this case, the input is linked to fault sensor. This function triggers alarm.*
b_{12} (or b_4 in CANopen®)	***input 2&4 logical***	
0	*negative logic*	*defines the edge (or level) that triggers input function*
1	*positive logic*	
$b_{14}b_{13}$ (or b_6b_5 in CANopen®)	***measurement to be transmitted***	
00	*gross*	*only for SCMBus/fast SCMBus protocols, no effect otherwise*
01	*net*	
10	*factory calibrated measurement*	

- **Note 1:** the functions with a * only are possible in SCMBus and fast SCMBus protocols.
- **Note 2:** in CANopen® communication protocol, to use the *'send TPDOX'* function, it is necessary to configure the TPDO communication parameters (and particularly the communication type) and the mapping for the corresponding TPDO.

13.2.2 Holding time (debounced time)

The holding time (de-bounced time) corresponds to the minimum required stabilization time of the logical inputs before their activation. If the input level varies within this interval, it is ignored.

13.2.3 Analog output(s) assignment (IO+ version)

The following tables describe the possible assignments.

bits	*meaning*	*note*
$b_3 b_2 b_1 b_0$	***analog output assignment***	
0000	*none*	*the output level does not vary*
0001	*copy gross weight*	*Adjustable polarity*
0010	*copy net weight*	
0011	*level on request*	*parameter* ***External value to control analog output*** *will drive analog output*
0100	*flow rate control output*	*in loss in weight feeder mode (for extraction device control)*
0101	*copy instantaneous flow rate*	
0110	*copy average flow rate*	
0111	*copy average speed*	
b_4	***polarity***	
0	*unipolar*	*could be set only with gross measurement*
1	*bipolar*	
$b_7 b_6 b_5$	***output voltage settings***	
000	*disable*	
001	*0 V - 5 V*	
010	*0 V - 10 V*	
$b_{10} b_9 b_8$	***output current settings***	
000	*disable*	
001	*4 mA - 20 mA*	
010	*0 mA - 20 mA*	
011	*0 mA - 24 mA*	
100	*4 mA - 20 mA alarm 3.6 mA*	voltage output is inactive (High-Z state)

13.2.4 External value to control analog output (IO+ version)

If an external device (e.g. PLC) would like to control extraction command through ***eNod4*** analog output, so that output must be set on ***level on request*** function. In this configuration ***eNod4*** will copy ***external value to control analog output*** parameter on analog output in current and voltage.

The external value parameter is expressed in 0.01% of full scale of analog output current or voltage.

13.2.5 Logical outputs 1&2 assignment

The following table describes the possible assignments.

bits	*meaning*	*note*
$b_3 b_2 b_1 b_0$	***output 1 assignment***	
0000	*none*	*the output level does not vary*
0001	*set point 1*	*functioning described by the 'set point functioning' setting and by the 'set point 1 high and low values'*
0010	*motion*	*copies the motion flag of the status bytes*
0011	*defective measurement*	*reflect the internal alarm flag described in "Weighing diagnosis" § in the MEASUREMENT AND STATUS §*
0100	*input 1 image*	*copies input 1 level*
0101	*level on request*	*output 1 level is driven by the 'OUT1 activation/deactivation' functional command*
0110	*belt alarms*	*Belt mode (alarm flag representing the OR logical operation between the error bits of the alarms registers word)*
0111	*external totalizer*	*Belt mode*
1000	*belt system running*	*Belt mode*
1001	*batch in progress*	*Belt mode*
1010	*batch result available*	*Belt mode*
1011	*conveyor starting alarm*	*Belt mode*
1100	*material TOR gate*	*Belt mode*
b_4	***output 1 logical***	
0	*negative logic*	*defines the output level when enabled*
1	*positive logic*	
$b_{11} b_{10} b_9 b_8$ (or $b_3 b_2 b_1 b_0$ in CANopen®)	***output 2 assignment***	
0000	*none*	*the output level does not vary*
0001	*set point 2*	*functioning described by the 'set point functioning' setting and by the 'set point 2 high and low values'*
0010	*motion*	*copies the motion flag of the status bytes*
0011	*defective measurement*	*reflect the internal alarm flag described in "Weighing diagnosis" § in the MEASUREMENT AND STATUS §*
0100	*input 2 image*	*copies input 2 level*
0101	*level on request*	*output 2 level is driven by the 'OUT2 activation/deactivation' functional command*
0110	*belt alarms*	*Belt mode (alarm flag representing the OR logical operation between the error bits of the alarms registers word)*

bits	***meaning***	***note***
0111	*external totalizer*	*Belt mode*
1000	*belt system running*	*Belt mode*
1001	*batch in progress*	*Belt mode*
1010	*batch result available*	*Belt mode*
1011	*conveyor starting alarm*	*Belt mode*
1100	*material TOR gate*	*Belt mode*
b_{12} (or b_4 in CANopen®)	***output 2 logical***	
0	*negative logic*	*defines the output level when enabled*
1	*positive logic*	*defines the output level when enabled*

13.2.6 Logical outputs 3&4 assignment

Similar to the outputs 1&2 configuration parameter, see previous paragraph (replacing all references to output 1 by output 3 and all references to output 2 by output 4).

13.2.7 Weight quantity per pulse on logical output

When a logical output is assigned to ***external totalizer***, a pulse is generated every time the totalization increases a multiple of a ***weight quantity per pulse on logical output***. The maximum pulse frequency is 10 Hz and pulse duration is fixed at 50ms. The output pulse is not rounded (i.e. the pulse is issued only once ***weight quantity per pulse on logical output*** is strictly exceeded).

13.2.8 Set points functioning

The following table describes the possible assignments.

bits	*meaning*	*note*
b_0	***set point 1 commutation mode***	
0	*window*	*only if output 1 assigned to the 'set point' function*
1	*hysteresis*	
$b_2 b_1$	***set point 1 comparison measurement***	
00	*gross*	
01	*net*	
10	*Sensor input control result*	
11	*Batch*	
b_3	***reserved (0)***	
b_4	***set point 2 commutation mode***	
0	*window*	*only if output 2 assigned to the 'set point' function*
1	*hysteresis*	
$b_6 b_5$	***set point 2 comparison measurement***	
00	*gross*	
01	*net*	
10	*Sensor input control result*	
11	*Batch*	
b_7	***reserved (0)***	
b_8 (or b_0 in CANopen®)	***set point 3 commutation mode***	
0	*window*	*only if output 3 assigned to the 'set point' function*
1	*hysteresis*	
$b_{10} b_9$ (or $b_2 b_1$ in CANopen®)	***set point 3 comparison measurement***	
00	*gross*	
01	*net*	
10	*Sensor input control result*	
11	*Batch*	
b_{11} (or b_3 in CANopen®)	***reserved (0)***	

bits	*meaning*	*note*
b_{12} (or b_4 in CANopen®)	***set point 4 commutation mode***	
0	*window*	*only if output 4 assigned to the 'set point' function*
1	*hysteresis*	
$b_{14}b_{13}$ (or b_6b_5 in CANopen®)	***set point 4 comparison measurement***	
00	*gross*	
01	*net*	
10	*Sensor input control result*	
11	*Batch*	
b_{15} (or b_7 in CANopen®)	***reserved (0)***	

13.2.9 Set points high and low values

Each set point is described by its commutation mode (hysteresis/window) and by a couple of values that are constantly compared to the gross or net measurement or to dosing result or dosing running total (depending on the configuration the set point has been given) in order to define the corresponding output logical level. For more details about the set points functioning, please refer to documentation ref. 236722.

Admitted values: from -1000000 to 1000000.

13.3 Input/output level

The level of the ***eNod4*** Input/output can be read according to the following table:

Bits	*Meaning*	*Note*
b0		
0	*low*	*IN1 level*
1	*high*	
b1		
0	*low*	*IN2 level*
1	*high*	
b2	*With IO+ version only, else 0*	
0	*low*	*IN3 level*
1	*high*	
b3	*With IO+ version only, else 0*	
0	*low*	*IN4 level*
1	*high*	
b7 ... b4		
0	*reserved (0)*	
b8 (note 1)		
0	*low*	*OUT1 level*
1	*high*	
b9 (note 1)		
0	*low*	*OUT2 level*
1	*high*	
b10 (note 1)		
0	*low*	*OUT3 level*
1	*high*	
b11 (note 1)		
0	*low*	*OUT4 level*
1	*high*	
b15 ... b12 (note 1)		
0	*reserved (0)*	

Note 1: In CANopen® communication protocol (according to version), this word is divided into 2 bytes of 8-bits registers. Bits b8 to b15 are therefore equivalent to bits b0 to b7 of the corresponding address (see CANopen® Register table).

14 LEGAL FOR TRADE OPTIONS

Name	Modbus address	CANopen index/sub-index	Profibus Cyclic IN/OUT	Acyclic DPV1 slot/index	Type	Access
Legal for trade version	*0x0004 LSB*	*0x3600 / 0x02*	*R : 0x0210* *W: 0x0211*	*0x01 / 0x00*	*Byte*	*RO*
Legal for trade switch	*0x0004 MSB*	*0x3600 / 0x01*			*Byte*	*RW*
Legal for trade counter	*0x0005*	*0x3600 / 0x03*	*R : 0x0212*	*0x01 / 0x01*	*Uint*	*RO*
Legal for trade checksum	*0x0006*	*0x3600 / 0x04*	*R : 0x0214*		*Uint*	*RO*
Zero functions	*0x0007*	*0x3501 / 0x01*	*R : 0x0216* *W: 0x0217*	*0x01 / 0x02*	*Byte*	*RW*
Stability criterion	*0x0008 LSB*	*0x3605 / 0x00*	*R : 0x0218* *W: 0x0219*	*0x01 / 0x03*	*Byte*	*RW*
decimal point position	*0x0008 MSB*	*0x3700 / 0x02*			*Byte*	*RW*
Weight unit	*0x0009*	*0x3700 / 0x01*	*R : 0x041A* *W: 0x041B*	*0x01 / 0x04*	*String*	*RW*
Flow rate time unit	*0x005A*	*0x4701 / 0x01*	*R : 0x021E* *W: 0x021F*	*0x01 / 0x06*	*String*	*RW*

14.1 Principles

The legal for trade options are a set of functions and indicators that are generally used in weighing applications. They have an impact on the device behavior regarding the metrological requirements and track every configuration change that may affect the measurement determination.

14.2 Settings description

14.2.1 Legal for trade switch

This setting activates (b_0 bit set to 1) or deactivates (b_0 bit set to 0) criteria and parameters related to the use of ***eNod4*** in OIML compliance.

The *'legal for trade'* option activation leads to the following changes:

- the *'legal for trade counter'* is incremented every time a storage into EEPROM is requested if one or several metrological settings have been modified.
- a new *'legal for trade checksum'* value is calculated every time a storage into EEPROM is requested if one or several metrological settings have been modified.
- taring is now impossible if gross measurement is negative.
- the measurement value variations cannot be read during the 15 seconds that follow the device reset (error frame in Modbus RTU, value set to -1 in CANopen® and in Profibus DP) and during zero and tare acquisitions

14.2.2 Legal for trade software version

This RO value identifies the version of the part of the software that is dedicated to the metrology and the measurement exploitation.

14.2.3 Legal for trade counter

If the *'legal for trade'* option is enabled, the legal for trade counter is incremented every time a backup into EEPROM is requested if at least one (or several) of these settings has been modified:

- legal for trade switch
- stability criterion
- decimal point position
- maximum capacity

- number of calibration segments
- calibration loads 1/2/3
- scale interval
- span adjusting coefficient
- calibration place/place of use g values
- sensitivity
- A/D conversion rate
- filtering configuration (activation option, order and cut-off frequencies)
- weight unit
- flow rate time unit
- zero functions

14.2.4 Legal for trade checksum

If the 'legal for trade' option is enabled, a new legal for trade checksum is calculated every time a backup into EEPROM is requested if at least one (or several) of the settings listed above has been modified.

14.2.5 Zero functions

The zero tracking and the initial zero setting can be respectively enabled by setting b_0 bit or b_1 bit to 1. When activated, both options are effective on a ±10% range of the *'maximum capacity'* (±2% if the *'legal for trade'* option is enabled).

When the initial zero is used, you must use a stability criterion other than 0 to be not affected by transient effects at power-up.

14.2.6 Stability criterion

The stability criterion defines the interval on which measurements are considered as stable. Motion is indicated by b_4 bit of the measurement status register. A measurement is stable if X consecutive measurements following the reference measurement are included in the stability interval (see following table) else the current measurement becomes the new reference measurement. X depends on the A/D conversion rate.

Bits b2 b1 b0	*Stability criterion*	*Note*
000	*no motion detection (always stable)*	
001	*0,25d*	*1d = 1 scale interval*
010	*0,5d*	
011	*1d*	
100	*2d*	

A/D conversion rate (meas/s)		X
50-Hz rejection	60-Hz rejection	
6,25	*7,5*	*1*
12,5	*15*	*2*
25	*30*	*3*
50	*60*	*5*
100	*120*	*9*
200	*240*	*17*
400	*480*	*33*
800	*960*	*65*
1600	*1920*	*129*

14.2.7 Decimal point position

Although ***eNod4*** measurements are integer values it is however possible to store a '*decimal point position'* so as to design a display related to the application. Its value represents the number of decimal digits. If the variable is set to Zero, it means that decimal point is not used.

Note: the decimal point is directly integrated to SCMBus protocol frames (see § SCMBus).

Admitted values: from 0 up to 7.

14.2.8 Weight unit

It is possible to store the display weight unit into the ***eNod4***.

In ***transmitter mode***, weight unit is a combination of 4 characters and there is no automatic calculation to adjust weight values if modified.

In ***belt mode***, following values are permitted for weight unit parameter:

- gram (**g**)
- kilogram (**kg**)
- ton (**t**)

Note: the unit is directly integrated to SCMBus protocol frames (see § SCMBus).

14.2.9 Flow rate time unit

It is possible to store the display *flow rate time unit* into the ***eNod4***.

Flow rate time unit is a combination of 2 characters and there is no automatic calculation to adjust flow rate value if modified.

In belt mode, following values are permitted for *flow rate time unit* parameter:

- second (**s**)
- hour (**h**)

Note: *weight and flow rate time units* lead ***eNod4*** to handle the following flow rate units:

- Gram per second (**g/s**)
- Kilogram per second(**kg/s**)
- Kilogram per hour (**kg/h**)
- Ton per hour (**t/h**)

14.2.10 Save Tare and Zero in non-volatile memory

There are two options for saving TARE or ZERO request value in non-volatile memory. These options are accessible through the “dosing cycle options” Object.

15 BELT OPERATING MODES

Name	Modbus address	CANopen index/sub-index	Profibus Cyclic IN/OUT	Acyclic DPV1 slot / index	Type	Access
Weight quantity per pulse on logical output	*0x0057*	*0x4700 /0x02*	*R : 0x02AE* *W : 0x02AF*	0x07 / 0x1D	*Uint*	*RW*
Average flow rate/speed determination depth	*0x0059*	*0x4703 /0x09*	*R : 0x0278* *W : 0279*	0x07 / 0x02	*Uint*	*RW*
Minimum weight to totalize x100	*0x005B*	*0x4700 /0x06*	*R : 0x027A* *W : 0x027B*	0x07 / 0x03	*Uint*	*RW*
Conveyor starting alarm duration	*0x005C*	*0x4702 /0x04*	*R : 0x027C* *W : 0x027D*	0x07 / 0x04	*Uint*	*RW*
Conveyor routing material time	*0x005D*	*0x4702 /0x05*	*R : 0x027E* *W : 0x027F*	0x07 / 0x05	*Uint*	*RW*
Conveyor speed stabilization time	*0x005E*	*0x4702 /0x06*	*R : 0x0282* *W : 0x0283*	0x07 / 0x07	*Uint*	*RW*
Cycle and alarm options	*0x005F*	*0x4700 /0x01*	*R : 0x0280* *W : 0x0281*	0x07 / 0x06	*Uint*	*RW*
Nominal flow rate	*0x0060*	*0x4701 /0x02*	*R : 0x04AC* *W : 0x04AD*	0x07 / 0x1C	*Float*	*RW*
Min permissible flow rate	*0x0062*	*0x4701 /0x04*	*R : 0x0298* *W : 0x0299*	0x07 / 0x12	*Uint*	*RW*
Max permissible flow rate	*0x0063*	*0x4701 /0x05*	*R : 0x029A* *W : 0x029B*	0x07 / 0x13	*Uint*	*RW*
Instant flow rate correction factor	*0x002C*	*0x4701 /0x03*	*R : 0x0476* *W : 0x0477*	0x07 / 0x01	*Float*	*RW*
Dynamic conveyor zero band	*0x0064*	*0x4702 /0x08*	*R : 0x0200* *W : 0x0201*	0x07 / 0x3A	*Uint*	*RW*
User fixed belt speed x100	*0x0067*	*0x4703 /0x01*	*R : 0x028E* *W : 0x028F*	0x07 / 0x0D	*Uint*	*RW*
Weight section length	*0x0068*	*0x4702 /0x07*	*R : 0x0286* *W : 0x0287*	0x07 / 0x09	*Uint*	*RW*
Belt number of revolutions	*0x0069*	*0x4702 /0x03*	*R : 0x0288* *W : 0x0289*	0x07 / 0x0A	*Uint*	*RW*
Belt total length	*0x006A*	*0x4702 /0x01*	*R : 0x0484* *W : 0x0485*	0x07 / 0x08	*Float*	*RW*
Max permissible flow rate control output	*0x006C*	*0x4701 /0x09*	*R : 0x029E* *W : 0x029F*	0x07 / 0x15	*Uint*	*RW*
Min permissible flow rate control output	*0x006D*	*0x4701 /0x08*	*R : 0x029C* *W : 0x029D*	0x07 / 0x14	*Uint*	*RW*
Belt inclination x100	*0x006E*	*0x4702 /0x02*	*R : 0x028A* *W : 0x028B*	0x07 / 0x0B	*Uint*	*RW*
Flow rate/belt load control inhibit time at start	*0x006F*	*0x4701 /0x06*	*R : 0x02A0* *W : 0x02A1*	0x07 / 0x16	*Uint*	*RW*
Flow rate/belt load control inhibit time in service	*0x0070*	*0x4701 /0x07*	*R : 0x02A2* *W : 0x02A3*	0x07 / 0x17	*Uint*	*RW*
Speed sensor maximum number of pulses per meter	*0x0071*	*0x4703 /0x02*	*R : 0x02B0* *W : 0x02B1*	0x07 / 0x1E	*Uint*	*RW*
Speed determination time factor	*0x0072*	*0x4703 /0x03*	*R : 0x02A4* *W : 0x02A5*	0x07 / 0x18	*Uint*	*RW*
Nominal belt speed x100	*0x0073*	*0x4703 /0x04*	*R : 0x028C* *W : 0x028D*	0x07 / 0x0C	*Uint*	*RW*
Min permissible belt speed	*0x0074*	*0x4703 /0x05*	*R : 0x0290* *W : 0x0291*	0x07 / 0x0E	*Uint*	*RW*
Max permissible belt speed	*0x0075*	*0x4703 /0x06*	*R : 0x0292* *W : 0x0293*	0x07 / 0x0F	*Uint*	*RW*
Belt speed control inhibit time at start	*0x0076*	*0x4703 /0x07*	*R : 0x0294* *W : 0x0295*	0x07 / 0x10	*Uint*	*RW*
Belt speed control inhibit time in service	*0x0077*	*0x4703 /0x08*	*R : 0x0296* *W : 0x0297*	0x07 / 0x11	*Uint*	*RW*
Weight to totalize (in weight unit x1000)	*0x0078*	*0x4700 /0x03*	*R : 0x04A6* *W : 0x04A7*	0x07 / 0x19	*Ulong*	*RW*

Name	Modbus address	CANopen index/sub-index	Profibus Cyclic IN/OUT	Acyclic DPV1 slot / index	Type	Access
Complementary weight to totalize (in weight unit)	*0x007A*	*0x4700 /0x04*	*R : 0x02A8* *W : 0x02A9*	0x07 / 0x1A	*Uint*	*RW*
Weight to totalize inflight value (in weight unit)	*0x007B*	*0x4700 /0x05*	*R : 0x04AA* *W : 0x04AB*	0x07 / 0x1B	*long*	*RW*
Calibration point 1 for the control output (output value)	*0x0A00*	*0x470A /0x01*	*R : 0x02B4* *W : 0x02B5*	0x07 / 0x20	*Uint*	*RW*
Calibration point 2 for the control output (output value)	*0x0A01*	*0x470A /0x02*	*R : 0x02B6* *W : 0x02B7*	0x07 / 0x21	*Uint*	*RW*
Calibration point 3 for the control output (output value)	*0x0A02*	*0x470A /0x03*	*R : 0x02B8* *W : 0x02B9*	0x07 / 0x22	*Uint*	*RW*
Calibration point 4 for the control output (output value)	*0x0A03*	*0x470A /0x04*	*R : 0x02BA* *W : 0x02BB*	0x07 / 0x23	*Uint*	*RW*
Calibration point 5 for the control output (output value)	*0x0A04*	*0x470A /0x05*	*R : 0x02BC* *W : 0x02BD*	0x07 / 0x24	*Uint*	*RW*
Calibration point 6 for the control output (output value)	*0x0A05*	*0x470A /0x06*	*R : 0x02BE* *W : 0x02BF*	0x07 / 0x25	*Uint*	*RW*
Calibration point 7 for the control output (output value)	*0x0A06*	*0x470A /0x07*	*R : 0x02C0* *W : 0x02C1*	0x07 / 0x26	*Uint*	*RW*
Calibration point 8 for the control output (output value)	*0x0A07*	*0x470A /0x08*	*R : 0x02C2* *W : 0x02C3*	0x07 / 0x27	*Uint*	*RW*
Calibration point 9 for the control output (output value)	*0x0A08*	*0x470A /0x09*	*R : 0x02C6* *W : 0x02C7*	0x07 / 0x28	*Uint*	*RW*
Calibration point 10 for the control output (output value)	*0x0A09*	*0x470A /0x0A*	*R : 0x02C8* *W : 0x02C9*	0x07 / 0x29	*Uint*	*RW*
Calibration point 1 of the control output (flow rate value) corresponding to value of the control output	*0x0A0A*	*0x470B /0x01*	*R : 0x04CA* *W : 0x04CB*	0x07 / 0x2A	*Float*	*RW*
Calibration point 2 of the control output (flow rate value) corresponding to value of the control output	*0x0A0C*	*0x470B /0x02*	*R : 0x04CC* *W : 0x04CD*	0x07 / 0x2B	*Float*	*RW*
Calibration point 3 of the control output (flow rate value) corresponding to value of the control output	*0x0A0E*	*0x470B /0x03*	*R : 0x04CE* *W : 0x04CF*	0x07 / 0x2C	*Float*	*RW*
Calibration point 4 of the control output (flow rate value) corresponding to value of the control output	*0x0A10*	*0x470B /0x04*	*R : 0x04D0* *W : 0x04D1*	0x07 / 0x2D	*Float*	*RW*
Calibration point 5 of the control output (flow rate value) corresponding to value of the control output	*0x0A12*	*0x470B /0x05*	*R : 0x04D2* *W : 0x04D3*	0x07 / 0x2E	*Float*	*RW*
Calibration point 6 of the control output (flow rate value) corresponding to value of the control output	*0x0A14*	*0x470B /0x06*	*R : 0x04D4* *W : 0x04D5*	0x07 / 0x2F	*Float*	*RW*
Calibration point 7 of the control output (flow rate value) corresponding to value of the control output	*0x0A16*	*0x470B /0x07*	*R : 0x04D6* *W : 0x04D7*	0x07 / 0x30	*Float*	*RW*
Calibration point 8 of the control output (flow rate value) corresponding to value of the control output	*0x0A18*	*0x470B /0x08*	*R : 0x04D8* *W : 0x04D9*	0x07 / 0x31	*Float*	*RW*
Calibration point 9 of the control output (flow rate value) corresponding to value of the control output	*0x0A1A*	*0x470B /0x09*	*R : 0x04DA* *W : 0x04DB*	0x07 / 0x32	*Float*	*RW*
Calibration point 10 of the control output (flow rate value) corresponding to value of the control output	*0x0A1C*	*0x470B /0x0A*	*R : 0x04DC* *W : 0x04DD*	0x07 / 0x33	*Float*	*RW*
Segments number for the Calibration curve of the control output	*0x0A1E*	*0x4709 /0x00*	*R : 0x02B2* *W : 0x02B3*	0x07 / 0x1F	*Uint*	*RW*
Weigh frame zero reference	*0x0A31*	*0x4702 /0x0B*	*R : 0x044A* *W : 0x044B*	0x0A / 0x02	*Float*	*RW*
Min permissible belt load	*0x0A33*	*0x4702 /0x09*	*R : 0x0202* *W : 0x0203*	0x07 / 0x3B	*Uint*	*RW*
Max permissible belt load	*0x0A34*	*0x4702 /0x0A*	*R : 0x0204* *W : 0x0205*	0x07 / 0x3C	*Uint*	*RW*
Kp	*0x0A35*	*0x470C /0x01*	*R : 0x04DE* *W : 0x04DF*	0x07 / 0x34	*Float*	*RW*
Ti	*0x0A37*	*0x470C /0x02*	*R : 0x04E0* *W : 0x04E1*	0x07 / 0x35	*Ulong*	*RW*
Td	*0x0A39*	*0x470C /0x03*	*R : 0x04E2* *W : 0x04E3*	0x07 / 0x36	*Ulong*	*RW*

Name	Modbus address	CANopen index/sub-index	Profibus Cyclic IN/OUT	Acyclic DPV1 slot / index	Type	Access
PID behavior	*0x0A3B*	*0x470C /0x04*	*R : 0x02E4 W : 0x02E5*	0x07 / 0x37	*Uint*	*RW*
Reference flow rate for PID adjustment	*0x0A3C*	*0x470C /0x05*	*R : 0x04E6 W : 0x04E7*	0x07 / 0x38	*Float*	*RW*
Checked batch	*0x0A3E*	*0x4700 /0x07*	*R : 0x040C W : 0x040D*	0x07 / 0x3D	*Ulong*	*RW*
Belt fault control inhibit time at start	*0x0A50*	*0x4702 / 0x0C*	*R : 0x02FA W : 0x02FB*	0x0B / 0x01	*Uint*	*RW*
Belt fault control inhibit time in service	*0x0A51*	*0x4702 / 0x0D*	*R : 0x02FC W : 0x02FD*	*0x0B / 0x02*	*Uint*	*RW*
Cut-off frequency for average flow rate (x100)	*0x0A52*	*0x4701 / 0x0A*		*0x0B / 0x03*	*Uint*	*RW*
Time unit for average flow rate	*0x0A53*	*0x4701 / 0x0B*		*0x0B / 0x04*	*Uint*	*RW*
Weight unit for average flow rate	*0x0A54*	*0x4701 / 0x0C*		*0x0B / 0x05*	*Ulong*	*RW*
Instant flow rate	*0x0086*	*0x5005 /0x01 (M)*	*R: 0x04F4 W: / + See modules list*	0xA0 / 0x06	*Float*	*RO*
Average flow rate	*0x0088*	*0x5005 /0x02 (M)*	*See modules list*	/	*Float*	*RO*
Average belt speed	*0x008A*	*0x5005 /0x04 (M)*	*See modules list*	/	*Float*	*RO*
Belt alarms registers	*0x008C*	*0x5007 /0x01 (M)*	*See modules list*	/	*Uint*	*RO*
Totalizer value (Great WU)	*0x008D*	*0x5006 /0x01 (M)*	*R: 0x04F0 W: / + See modules list*	0xA0 / 0x04	*Ulong*	*RO*
Complementary totalizer value	*0x008F*	*0x5006 /0x02 (M)*	*R: 0x02F2 W: / + See modules list*	0xA0 / 0x05	*Uint*	*RO*
Dosing weight deviation	*0x0095*	*0x5007 /0x04 (M)*	*See modules list*	/	*Float*	*RO*
Flow rate control output	*0x009A*	*0x5005 /0x05 (M)*	*See modules list*	/	*Float*	*RO*
Dosing quality factor	*0x009C*	*0x5005 /0x07 (M)*	/	/	*Float*	*RO*
Errors counter	*0x009E*	*0x5007 /0x03 (M)*	*See modules list*	/	*Uint*	*RO*
Weigh frame load	*0x009F*	*0x5005 /0x03 (M)*	*R: 0x04F6 W: /*	0xA0 / 0x03	*Float*	*RO*
Control output value	*0x00A1*	*0x5005 /0x06 (M)*	*See modules list*	/	*Uint*	*RO*
Belt status register	*0x00A3*	*0x5007 /0x02 (M)*	*See modules list*	/	*Uint*	*RO*
Total per belt revolution	*0x00A4*	*0x5006 /0x03 (M)*	*See modules list*	/	*Float*	*RO*
Average flow rate Great Unit	*0x00A6*	*0x5005 /0x08 (M)*	/	/	*Float*	*RO*
Batch progression in percent	*0x00A8*	*0x5006 /0x04 (M)*	*See modules list*	/	*Uint*	*RO*
Grand total	*0x00A9*	*0x5006 /0x05 (M)*	*See modules list*	/	*Uint*	*RO*
General total	*0x00AB*	*0x5006 /0x06 (M)*	*See modules list*	/	*Uint*	*RO*

15.1 Settings description

- **Weight quantity per pulse on logical output**: For external totalization purpose, ***eNod4*** sends a pulse on logical output when the total value reaches multiple of this parameter. When an overflow is occurred on pulses output an alarm is set.
- **Average flow rate/speed determination depth:** Defines the samples “n” numbers for moving average filter on flow rate and belt speed.
- **Minimum load to totalize:** ***eNod4*** enables totalization when “**minimum weight to totalize x100**” divided by “**weigh section length**” is greater than the **minimum load to totalize**.
- **Conveyor starting alarm duration:** For security reason, ***eNod4*** provides buzzer function on logical output at belt start. The buzzer duration depends on this parameter value in second.
- **Conveyor routing material time:** Defines time for materials to travel distance from shear gate to weigh section at normal speed.
- **Conveyor speed stabilization time:** In weigh feeder mode and if PID function is done through the belt speed, this is the stabilization time before PID activation.
- **Cycle and alarm options:** This register defines cycle and alarms functioning. The table below shows the bits definitions.

• bits	Function	Note
bit b0	***PID activation ON/OFF***	
0	*PID function inhibited*	*In weigh feeder mode if batch option*
1	*PID function activated*	
bits b2 b1	***feeder regulation function***	
01	*Through materials flow*	*In weigh feeder mode*
10	*Through conveyor speed*	
bit b3	***Batch activation ON/OFF***	
0	*Batch dosing not activated*	
1	*Batch dosing activated*	
bit b4	***Clear totalization at start cycle option***	
0	*Clear totalization at start cycle not activated*	
1	*Clear totalization at start cycle activated*	
bit b5	***Auto-stop conveyor at end batch***	
0	*Auto-stop conveyor at end batch not activated*	
1	*Auto-stop conveyor at end batch activated*	
bit b6	***Dynamic zero auto-correction***	
0	*Dynamic zero auto-correction not activated*	
1	*Dynamic zero auto-correction activated*	
bit b7	***Reserved(do not used)***	
/	***/***	
bit b8	***Totalization function if alarm occurred***	
0	*Totalization continuously running if alarm occurred*	
1	*Stop totalization function if alarm occurred*	
bit b9	***Save Tare in non-volatile memory***	
0	*Save tare in volatile memory (RAM)*	
1	*Save tare in non-volatile memory (FRAM)*	
bit b10	***Save Zero in non-volatile memory***	
0	*Save zero in volatile memory (RAM)*	
1	*Save zero in non-volatile memory (FRAM)*	
bits b15b11	***Reserved***	
/	***/***	

- **Nominal flow rate:** Set point flow rate expressed in **weight unit** per **time unit.** ***eNod4*** determines **nominal belt load** from **nominal speed** and **nominal flow rate**.

- **Min permissible flow rate:** Minimum value for flow rate for alarm function. Expressed in 0.1% of nominal flow rate.

- **Max permissible flow rate:** Maximum value for flow rate for alarm function. Expressed in 0.1% of nominal flow rate.
- **Instant flow rate correction factor:** Correction factor to apply after material test.
- **Dynamic conveyor zero band:** Defines limit values for dynamic zero correction function. Zero function is cancelled when the weight is out of these limits.
- **User fixed belt speed x100:** Defines the belt speed in **m/s** if no speed sensor is connected to ***eNod4***. The value of 500 corresponds to 5 m/s.
- **Weight section length:** Represents the effective belt weigh frame length expressed in millimeters.
- **Belt number of revolutions:** Specifies the real belt revolutions to realize during speed/length calibrations and dynamic zero procedure.
- **Belt total length:** Defines the total length of the belt in meters.
- **Max permissible flow rate control output:** Maximum value for control output for alarm function. Expressed in 0.1% of nominal flow rate.
- **Min permissible flow rate control output:** Minimum value for control output for alarm function. Expressed in 0.1% of nominal flow rate.
- **Belt inclination x100:** Defines the belt title angle in degrees. The value of 1500 corresponds with inclination of 15 degrees.
- **Flow rate/belt load controls inhibit time at start:** Monitoring the flow rate or belt load is only activated after this delay time when the belt is started. It is expressed in second.
- **Flow rate/belt load controls inhibit time in service:** When the flow rate or belt load is below/above the min/max value, the alarm is activated after this delay elapsed. It is expressed in second.
- **Speed sensor maximum number of pulses per meter:** Defines constant maximum pulses number of speed sensor.
- **Speed determination time factor:** Specifies the belt speed determination time in multiple of 250ms.
- **Nominal belt speed:** Defines the nominal speed in **m/s** and the value of 500 corresponds to 5 m/s. ***eNod4*** determines **nominal belt load** from **nominal speed** and **nominal flow rate**.
- **Min permissible belt speed:** Minimum value for belt speed for alarm function.
- **Max permissible belt speed:** Maximum value for belt speed for alarm function.
- **Belt speed control inhibit time at start:** Inhibition time at belt start of speed monitoring in second.
- **Belt speed control inhibit time in service:** Inhibition time in service of speed monitoring in second.
- **Weight to totalize (Great WU):** The main part of batch target in **weight unit** x1000**.**
- **Complementary weight to totalize:** The complementary part of batch target in **weight unit.**
- **Weight to totalize inflight value:** The batch target inflight value in **weight unit.**
- **Calibration point n for the control output (output value):** Expressed in 0.01% of maximum output current or voltage. Up to 10 calibrations points can be configured. See calibration of flow rate section.
- **Calibration point n of the control output (flow rate value):** Expressed in **weight unit** per **time unit**. Up to 10 calibrations points can be configured. See calibration of flow rate section.
- **Segments number for the Calibration curve of the control output:** Defines calibration number of points of flow rate and output value.

- **Weigh frame zero reference:** reference value for zero per length unit. ***eNod4*** determines this value during dynamic zero procedure.
- **Min permissible belt load:** Minimum value for belt load for alarm function.
- **Max permissible belt load:** Maximum value for belt load for alarm function.
- **Kp:** A proportional controller (Kp) will have the effect of reducing the rise time and will reduce but never eliminate the steady-state error.
- **Ti:** An integral control (Ti) will have the effect of eliminating the steady-state error for a constant or step input, but it may make the transient response slower.
- **Td:** A derivative control (Td) will have the effect of increasing the stability of the system, reducing the overshoot, and improving the transient response.
- **PID behavior:** ***eNod4*** allows auto-adjustment of PID parameters (kp, Ti and Td). The PID behavior (slow, fast or stable) must be set before to realize an auto-adjustment.
- **Reference flow rate for PID adjustment:** This is the reference flow rate value for auto-adjustment. It must be set before sending adjustment command.
- **Checked batch:** After material test, user must fill in this register the real batch measuring by measurement instrument before sending flow rate correction command.
- **Instant flow rate:** Instantaneous flow rate expressed in **weight unit** per **time unit** and based on current belt load and speed.
- **Average flow rate:** The flow rate result of moving average filter on "n" Instantaneous flow rate samples and low-pass filter. It has its own unity for external display purpose (remote display or HMI).
- **Average belt speed:** The speed result of moving average filter on "n" Instantaneous belt speed samples.
- **Belt alarms registers:** This register defines the status of alarms. The table below shows the bits definitions.

b_{15} … b_0 *bits set to 1*	*Meaning*
b_0	*Instantaneous flow rate > Max permissible flow rate*
b_1	*Instantaneous flow rate < Min permissible flow rate*
b_2	*Instantaneous belt speed > Max permissible speed*
b_3	*Instantaneous belt speed < Min permissible speed*
b_4	*Instantaneous belt load > Max permissible belt load*
b_5	*Instantaneous belt load < Min permissible belt load*
b_6	*Control output value > Max permissible control output*
b_7	*Control output value < Min permissible control output*
b_8	*Conveyor starting alarm flag*
b_9	*Conveyor calibration in progress*
b_{10}	*External totalizer output overflow*
b_{15} … b_{11}	***Reserved (0)***

- **Totalizer value (Great WU):** Main totalization result in **weight unit** x1000.
- **Complementary totalizer value:** Complementary totalization result in **weight unit.**
- **Dosing weight deviation:** This represents the weight deviation of belt system at current flow rate and at nominal flow rate.

- **Flow rate control output:** This is the flow rate to control extraction module (feeder or conveyor motors). Expressed in **weight unit** per **time unit**, it can be the fixed value or output of PID function if enabled.

- **Dosing quality factor:** Represents the standard deviation on "n" successive instantaneous flow rate

- **Errors counter:** Each error occurred during totalization cycle is recognized in this register. Errors counter is set to zero at power up or at receiving of clear totalization command.

- **Weigh frame load:** The weigh frame load unit is the effective load on the **weight section**.

- **Control output value:** This value expressed in 0.01% of maximum output current or voltage is to control the analog output.

- **Belt status register:** This register defines the status of belt system. The table below shows the bits definitions.

b_{15} … b_0	Meaning
b_7 … b_0 (LSB)	**Dosing step**
1	*Stop*
3	*Material routing time*
5	*Conveyor starting alarm initialization*
7	*Conveyor starting alarm running*
9	*Flow rate stabilization time*
10	*Conveyor last revolution to stop*
12	*Batch in progress*
13	*Batch initialization*
32	*Batch suspended*
128	*PID coefficients auto-adjustment*
b_{15} … b_8 (MSB) bits set to 1	**Status**
b_8	*Belt system is running*
b_9	*Material TOR gate*
b_{10}	*Minimum totalization load targeted*
b_{11}	*Conveyor in dynamic zero band*
b_{12}	*Dynamic zero function in progress*
b_{13}	*Faulty dynamic zero function*
b_{14}	*Batch in progress*
b_{15}	*Batch result available*

- **Total per belt revolution:** Represents the amount of materials going through the belt conveyor scale at each revolution. The value is expressed in **weight unit** and is refreshed at each revolution. In perfect belt system, after dynamic zero procedure, the total per revolution is set to zero when the conveyor is running empty.

- **Average flow rate Great Unit:** This is the average flow rate value in 0.001% of weight unit per time unit.

- **Batch progression in percent:** Represents the global view progression in percent of batch process.

16 MODBUS RTU REGISTERS TABLE

Chapter	Name	Modbus Address	Type	Access
Modbus	Firmware revision	0x0000	Uint	RO
Modbus	Node number / baud rate	0x0001	Uint	RO
Legal for trade	Legal for trade version	0x0004 LSB	Byte	RO
Legal for trade	Legal for trade switch	0x0004 MSB	Byte	RW
Legal for trade	Legal for trade counter	0x0005	Byte	RO
Legal for trade	Legal for trade checksum	0x0006	Uint	RO
Legal for trade	Zero functions	0x0007	Uint	RW
Legal for trade	Stability criterion	0x0008 LSB	Byte	RW
Legal for trade	decimal point position	0x0008 MSB	Byte	RW
Legal for trade	Weight unit	0x0009	String	RW
Calibration	Maximum capacity	0x000C	Ulong	RW
Calibration	Number of calibration segments	0x000E	Uint	RW
Calibration	Calibration load 1	0x000F	Ulong	RW
Calibration	Calibration load 2	0x0011	Ulong	RW
Calibration	Calibration load 3	0x0013	Ulong	RW
Calibration	Sensor sensitivity	0x0015	Ulong	RW
Calibration	Scale interval	0x0017	Uint	RW
Calibration	Zero calibration	0x0018	Long	RW
Calibration	Span coefficient 1	0x001A	Float	RW
Calibration	Span coefficient 2	0x001C	Float	RW
Calibration	Span coefficient 3	0x001E	Float	RW
Calibration	Span adjusting coefficient	0x0020	Ulong	RW
Calibration	Calibration place g value	0x0022	Ulong	RW
Calibration	Place of use g value	0x0024	Ulong	RW
Calibration	Number of divisions at maximum capacity	0x0028	Ulong	RW
Calibration	Instant flow rate correction factor	0x002C	Float	RW
I/O	External value to control analog output	0x0032	Uint	RW
HMI	HMI name	0x0034	String	RW
Filter	A/D conversion rate	0x0036	Uint	RW
Filter	filters activation	0x0037 LSB	Byte	RW
Filter	Low-pass order	0x0037 MSB	Byte	RW
Filter	Low-pass cut-off frequency	0x0038	Uint	RW

Chapter	Name	Modbus Address	Type	Access
Filter	Band-stop high cut-off frequency	0x0039	Uint	RW
Filter	Band-stop low cut-off frequency	0x003A	Uint	RW
Protocol	Functioning mode / Serial protocol	0x003E	Uint	RW
SCMBus	SCMBus transmission period	0x003F	Uint	RW
I/O	Analog output functioning (optional)	0x0040	Uint	RW
I/O	Logical inputs 3 functioning (optional)	0x0041 LSB	Byte	RW
I/O	Logical inputs 4 functioning (optional)	0x0041 MSB	Byte	RW
I/O	Logical input 1 functioning	0x0042 LSB	Byte	RW
I/O	Logical input 2 functioning	0x0042 MSB	Byte	RW
I/O	holding time	0x0043	Uint	RW
I/O	Output 1 functioning	0x0044 LSB	Byte	RW
I/O	Output 2 functioning	0x0044 MSB	Byte	RW
I/O	Output 3 functioning	0x0045 LSB	Byte	RW
I/O	Output 4 functioning	0x0045 MSB	Byte	RW
I/O	Set point 1 high value	0x0046	Long	RW
I/O	Set point 1 low value	0x0048	Long	RW
I/O	Set point 2 high value	0x004A	Long	RW
I/O	Set point 2 low value	0x004C	Long	RW
I/O	Set point 3 high value	0x004E	Long	RW
I/O	Set point 3 low value	0x0050	Long	RW
I/O	Set point 4 high value	0x0052	Long	RW
I/O	Set point 4 low value	0x0054	Long	RW
I/O	1&2 Set points functioning	0x0056 LSB	Byte	RW
I/O	3&4 Set points functioning	0x0056 MSB	Byte	RW
Belt	Weight quantity per pulse on logical output	0x0057	Uint	RW
Filter	Depth of moving average filter on weights	0x0058	Uint	RW
Filter	Average flow rate/Speed determination depth	0x0059	Uint	RW
Belt	Flow rate time unit	0x005A	Uint	RW
Belt	Minimum weight to totalize x100	0x005B	Uint	RW
Belt	Conveyor starting alarm duration	0x005C	Uint	RW
Belt	Conveyor routing material time	0x005D	Uint	RW
Belt	Conveyor speed stabilization time	0x005E	Uint	RW
Belt	Cycle and alarm options	0x005F	Uint	RW
Belt	Nominal flow rate	0x0060	Float	RW

Chapter	Name	Modbus Address	Type	Access
Belt	Min permissible flow rate	0x0062	Uint	RW
Belt	Max permissible flow rate	0x0063	Uint	RW
Belt	Dynamic conveyor zero band	0x0064	Uint	RW
Belt	User fixed belt speed x100	0x0067	Uint	RW
Belt	Weight section length	0x0068	Uint	RW
Belt	Belt number of revolutions	0x0069	Uint	RW
Belt	Belt total length	0x006A	Float	RW
Belt	Max permissible flow rate control output	0x006C	Uint	RW
Belt	Min permissible flow rate control output	0x006D	Uint	RW
Belt	Belt inclination x100	0x006E	Uint	RW
Belt	Flow rate/load control inhibit time at start	0x006F	Uint	RW
Belt	Flow rate/load control inhibit time in service	0x0070	Uint	RW
Belt	Speed sensor maximum number of pulses per meter	0x0071	Uint	RW
Belt	Speed determination time factor	0x0072	Uint	RW
Belt	Nominal belt speed	0x0073	Uint	RW
Belt	Min permissible belt speed	0x0074	Uint	RW
Belt	Max permissible belt speed	0x0075	Uint	RW
Belt	Belt speed control inhibit time at start	0x0076	Uint	RW
Belt	Belt speed control inhibit time in service	0x0077	Uint	RW
Belt	Weight to totalize (Great WU)	0x0078	Ulong	RW
Belt	Complementary weight to totalize	0x007A	Uint	RW
Belt	Weight to totalize inflight value	0x007B	long	RW
State Register	Measurement status	0x007D	Uint	RO
State Register	Gross measurement	0x007E	Long	RO
State Register	Tare value	0x0080	Long	RO
State Register	Net measurement	0x0082	Long	RO
State Register	Factory calibrated points	0x0084	Long	RO
Belt	Instantaneous flow rate	0x0086	Float	RO
Belt	Average flow rate	0x0088	Float	RO
Belt	Average belt speed	0x008A	Float	RO
Belt	Belt alarms registers	0x008C	Uint	RO
Belt	Totalizer value (weight unit x1000)	0x008D	ULong	RO
Belt	Complementary totalizer value	0x008F	Uint	RO
Command	command register	0x0090	Uint	RW

Chapter	Name	Modbus Address	Type	Access
Command	response register	0x0091	Uint	RO
Calibration	Zero offset	0x0092	Long	RW
State Register	Input levels	0x0094 LSB	Byte	RO
I/O	Input levels	0x0094 LSB	Byte	RO
State Register	output levels	0x0094 MSB	Byte	RO
I/O	output levels	0x0094 MSB	Byte	RO
Belt	Dosing weight deviation	0x0095	Float	RO
State Register	Preset tare value	0x0097	Ulong	RW
Belt	Flow rate control output	0x009A	Float	RO
Belt	Dosing quality factor	0x009C	Float	RO
Belt	Errors counter	0x009E	Uint	RO
Belt	Weigh frame load	0x009F	Float	RO
Belt	Control output value	0x00A1	Uint	RO
Belt	Belt status register	0x00A3	Uint	RO
Belt	Total per belt revolution	0x00A4	Float	RO
Belt	Average flow rate Great Unit	0x00A6	Float	RO
Belt	Batch progression in percent	0x00A8	Uint	RO

*************************** *JUMP TO EXTENSION MODBUS REGISTERS* *******************************

Chapter	Name	Modbus address	Type	Access
Belt	Calibration point 1 for the control output (output value)	0x0A00	Uint	RW
Belt	Calibration point 2 for the control output (output value)	0x0A01	Uint	RW
Belt	Calibration point 3 for the control output (output value)	0x0A02	Uint	RW
Belt	Calibration point 4 for the control output (output value)	0x0A03	Uint	RW
Belt	Calibration point 5 for the control output (output value)	0x0A04	Uint	RW
Belt	Calibration point 6 for the control output (output value)	0x0A05	Uint	RW
Belt	Calibration point 7 for the control output (output value)	0x0A06	Uint	RW
Belt	Calibration point 8 for the control output (output value)	0x0A07	Uint	RW
Belt	Calibration point 9 for the control output (output value)	0x0A08	Uint	RW
Belt	Calibration point 10 for the control output (output value)	0x0A09	Uint	RW
Belt	Calibration point 1 of the control output (flow rate value) corresponding to value of the control output	0x0A0A	Float	RW
Belt	Calibration point 2 of the control output (flow rate value) corresponding to value of the control output	0x0A0C	Float	RW
Belt	Calibration point 3 of the control output (flow rate value) corresponding to value of the control output	0x0A0E	Float	RW
Belt	Calibration point 4 of the control output (flow rate value) corresponding to value of the control output	0x0A10	Float	RW
Belt	Calibration point 5 of the control output (flow rate value) corresponding to value of the control output	0x0A12	Float	RW
Belt	Calibration point 6 of the control output (flow rate value) corresponding to value of the control output	0x0A14	Float	RW
Belt	Calibration point 7 of the control output (flow rate value) corresponding to value of the control output	0x0A16	Float	RW
Belt	Calibration point 8 of the control output (flow rate value) corresponding to value of the control output	0x0A18	Float	RW
Belt	Calibration point 9 of the control output (flow rate value) corresponding to value of the control output	0x0A1A	Float	RW
Belt	Calibration point 10 of the control output (flow rate value) corresponding to value of the control output	0x0A1C	Float	RW
Belt	Segments number for the Calibration curve of the control output	0x0A1E	Uint	RW
Belt	Weigh frame zero reference	0x0A31	Float	RW
Belt	Min permissible belt load	0x0A33	Uint	RW
Belt	Max permissible belt load	0x0A34	Uint	RW
Belt	Kp	0x0A35	Float	RW
Belt	Ti	0x0A37	Ulong	RW
Belt	Td	0x0A39	Ulong	RW
Belt	PID behavior	0x0A3B	Uint	RW
Belt	Reference flow rate for PID adjustment	0x0A3C	Float	RW
Belt	Checked batch	0x0A3E	Ulong	RW
State Register	Sensor input control reference	0x0A44	long	RW
State Register	Sensor input control result	0x0A46	Int	RO
State Register	Sensor input control result max. tolerance	0x0A47	Uint	RW

Chapter	Name	Modbus address	Type	Access
State Register & I/O	Defective measurement debounced time	0x0A48	Uint	RW
State Register & I/O	Defective measurement alarm activation time	0x0A49	Uint	RW
Belt	Belt fault control inhibit time at start	0x0A50	Uint	RW
Belt	Belt fault control inhibit time in service	0x0A51	Uint	RW
Belt	Cut-off frequency for average flow rate (x100)	0x0A52	Uint	RW
Belt	Time unit for average flow rate	0x0A53	Uint	RW
Belt	Time weight for average flow rate	0x0A54	Ulong	RW

17 CRC-16 CALCULATION ALGORITHM