MODBUS
Communication Protocol

for RS485 MODBUS and LAN GATEWAY modules
for counters with integrated MODBUS or ETHERNET interface

PROTOCOL MANUAL

v011 - August edition 2021

Limitation of Liability

The Manufacturer reserves the right to modify the specifications in this manual without previous warning. Any copy of this manual, in part or in full, whether by photocopy or by other means, even of electronic nature, without the manufacture giving written authorization, breaches the terms of copyright and is liable to prosecution.

It is absolutely forbidden to use the device for different uses other than those for which it has been devised for, as inferred to in this manual. When using the features in this device, obey all laws and respect privacy and legitimate rights of others.

EXCEPT TO THE EXTENT PROHIBITED BY APPLICABLE LAW, UNDER NO CIRCUMSTANCES SHALL THE MANUFACTURER BE LIABLE FOR CONSEQUENTIAL DAMAGES SUSTAINED IN CONNECTION WITH SAID PRODUCT AND THE MANUFACTURER NEITHER ASSUMES NOR AUTHORIZES ANY REPRESENTATIVE OR OTHER PERSON TO ASSUME FOR IT ANY OBBLIGATION OR LIABILTY OTHER THAN SUCH AS IS EXPRESSLY SET FORTH HEREIN.

All trademarks in this manual are property of their respective owners.

The information contained in this manual is for information purposes only, is subject to changes without previous warning and cannot be considered binding for the Manufacturer. The Manufacturer assumes no responsibility for any errors or incoherence possibly contained in this manual.

Index

1. DESCRIPTION

MODBUS ASCII/RTU is a master-slave communication protocol, able to support up to 247 slaves connected in a bus or a star network. The protocol uses a simplex connection on a single line. In this way, the communication messages move on a single line in two opposite directions.

MODBUS TCP is a variant of the MODBUS family. Specifically, it covers the use of MODBUS messaging in an "Intranet" or "Internet" environment using the TCP/IP protocol on a fixed port 502.

Master-slave messages can be:

- Reading (Function codes $01, $03, $04): the communication is between the master and a single slave. It allows to read information about the queried counter
- Writing (Function code $10): the communication is between the master and a single slave. It allows to change the counter settings
- Broadcast (not available for MODBUS TCP): the communication is between the master and all the connected slaves. It is always a write command (Function code $10) and required logical number $00

In a multi-point type connection (MODBUS ASCII/RTU), slave address (called also logical number) allows to identify each counter during the communication. Each counter is preset with a default slave address (01) and the user can change it.

In case of MODBUS TCP, slave address is replaced by a single byte, the Unit identifier.

Communication frame structure - ASCII mode

Bit per byte: 1 Start, 7 Bit, Even, 1 Stop (7E1)

Name	Length	Function
START FRAME	1 char	Message start marker. Starts with colon ":" ($3A)
ADDRESS FIELD	2 chars	Counter logical number
FUNCTION CODE	2 chars	Function code ($01 / $03 / $04 / $10)
DATA FIELD	n chars	Data + length will be filled depending on the message type
ERROR CHECK	2 chars	Error check (LRC)
END FRAME	2 chars	Carriage return - line feed (CRLF) pair ($0D & $0A)

Communication frame structure - RTU mode

Bit per byte: 1 Start, 8 Bit, None, 1 Stop (8N1)

Name	Length	Function
START FRAME	4 chars idle	At least 4 character time of silence (MARK condition)
ADDRESS FIELD	8 bits	Counter logical number
FUNCTION CODE	8 bits	Function code ($01 / $03 / $04 / $10)
DATA FIELD	n x 8 bits	Data + length will be filled depending on the message type
ERROR CHECK	16 bits	Error check (CRC)
END FRAME	4 chars idle	At least 4 character time of silence between frames

Communication frame structure - TCP mode

Bit per byte: 1 Start, 7 Bit, Even, 2 Stop (7E2)

Name	Length	Function
TRANSACTION ID	2 bytes	For synchronization between messages of server & client
PROTOCOL ID	2 bytes	Zero for MODBUS TCP
BYTE COUNT	2 bytes	Number of remaining bytes in this frame
UNIT ID	1 byte	Slave address (255 if not used)
FUNCTION CODE	1 byte	Function code ($01 / $04 / $10)
DATA BYTES	n bytes	Data as response or command

1.1 LRC Generation

The Longitudinal Redundancy Check (LRC) field is one byte, containing an 8–bit binary value. The LRC value is calculated by the transmitting device, which appends the LRC to the message. The receiving device recalculates an LRC during receipt of the message, and compares the calculated value to the actual value it received in the LRC field. If the two values are not equal, an error results. The LRC is calculated by adding together successive 8–bit bytes in the message, discarding any carries, and then two's complementing the result. The LRC is an 8–bit field, therefore each new addition of a character that would result in a value higher than 255 decimal simply 'rolls over' the field's value through zero. Because there is no ninth bit, the carry is discarded automatically.

A procedure for generating an LRC is:

1. Add all bytes in the message, excluding the starting 'colon' and ending CR LF. Add them into an 8–bit field, so that carries will be discarded.
2. Subtract the final field value from $FF, to produce the ones–complement.
3. Add 1 to produce the twos–complement.

Placing the LRC into the Message

When the the 8–bit LRC (2 ASCII characters) is transmitted in the message, the high–order character will be transmitted first, followed by the low–order character. For example, if the LRC value is $52 (0101 0010):

Colon ':'	Addr	Func	Data Count	Data	Data	....	Data	LRC Hi '5'	LRC Lo'2'	CR	LF

C-function to calculate LRC

```
*pucFrame – pointer on "Addr" of message
usLen – length message from "Addr" to end "Data"
UCHAR prvucMBLRC( UCHAR * pucFrame, USHORT usLen )
{
    UCHAR           ucLRC = 0;  /* LRC char initialized */
    while( usLen-- )
    {
        ucLRC += *pucFrame++;   /* Add buffer byte without carry */
    }
    /* Return twos complement */
    ucLRC = ( UCHAR ) ( -( ( CHAR ) ucLRC ) );
    return ucLRC;
}
```

1.2 CRC Generation

The Cyclical Redundancy Check (CRC) field is two bytes, containing a 16–bit value. The CRC value is calculated by the transmitting device, which appends the CRC to the message. The receiving device recalculates a CRC during receipt of the message, and compares the calculated value to the actual value it received in the CRC field. If the two values are not equal, an error results.

The CRC is started by first preloading a 16–bit register to all 1's. Then a process begins of applying successive 8–bit bytes of the message to the current contents of the register. Only the eight bits of data in each character are used for generating the CRC. Start and stop bits, and the parity bit, do not apply to the CRC.

During generation of the CRC, each 8–bit character is exclusive ORed with the register contents. Then the result is shifted in the direction of the least significant bit (LSB), with a zero filled into the most significant bit (MSB) position. The LSB is extracted and examined. If the LSB was a 1, the register is then exclusive ORed with a preset, fixed value. If the LSB was a 0, no exclusive OR takes place.

This process is repeated until eight shifts have been performed. After the last (eighth) shift, the next 8–bit character is exclusive ORed with the register's current value, and the process repeats for eight more shifts as described above. The final contents of the register, after all the characters of the message have been applied, is the CRC value.

A calculated procedure for generating a CRC is:

1. Load a 16–bit register with $FFFF. Call this the CRC register.
2. Exclusive OR the first 8–bit byte of the message with the low–order byte of the 16–bit CRC register, putting the result in the CRC register.
3. Shift the CRC register one bit to the right (toward the LSB), zero–filling the MSB. Extract and examine the LSB.
4. (If the LSB was 0): Repeat Step 3 (another shift). (If the LSB was 1): Exclusive OR the CRC register with the polynomial value $A001 (1010 0000 0000 0001).
5. Repeat Steps 3 and 4 until 8 shifts have been performed. When this is done, a complete 8–bit byte will have been processed.
6. Repeat Steps 2 through 5 for the next 8–bit byte of the message. Continue doing this until all bytes have been processed.
7. The final contents of the CRC register is the CRC value.
8. When the CRC is placed into the message, its upper and lower bytes must be swapped as described below.

Placing the CRC into the Message

When the 16–bit CRC (two 8–bit bytes) is transmitted in the message, the low-order byte will be transmitted first, followed by the high-order byte.

For example, if the CRC value is $35F7 (0011 0101 1111 0111):

Addr	Func	Data Count	Data	Data	....	Data	CRC lo F7	CRC Hi 35

CRC generation functions - With Table

All of the possible CRC values are preloaded into two arrays, which are simply indexed as the function increments through the message buffer. One array contains all of the 256 possible CRC values for the high byte of the 16–bit CRC field, and the other array contains all of the values for the low byte. Indexing the CRC in this way provides faster execution than would be achieved by calculating a new CRC value with each new character from the message buffer.

```
/*CRC table for calculate with polynom 0xA001 with init value 0xFFFF, High half word*/
rom unsigned char CRC_Table_Hi[] = {
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
    0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
    0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
    0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81,
    0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,
    0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
    0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
    0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
    0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
    0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,
    0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,  0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
    0x40
};
/*CRC table for calculate with polynom 0xA001 with init value 0xFFFF, Low half word*/
rom unsigned char  CRC_Table_Lo[] = {
    0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7, 0x05, 0xC5, 0xC4,
    0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09,
    0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE, 0xDF, 0x1F, 0xDD,
    0x1D, 0x1C, 0xDC, 0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,
    0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32, 0x36, 0xF6, 0xF7,
    0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A,
```

```
    0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B, 0x2A, 0xEA, 0xEE,
    0x2E, 0x2F, 0xEF, 0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
    0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1, 0x63, 0xA3, 0xA2,
    0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F,
    0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8, 0xB9, 0x79, 0xBB,
    0x7B, 0x7A, 0xBA, 0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
    0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0, 0x50, 0x90, 0x91,
    0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C,
    0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59, 0x58, 0x98, 0x88,
    0x48, 0x49, 0x89, 0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
    0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83, 0x41, 0x81, 0x80,
    0x40
};

unsigned short ModBus_CRC16( unsigned char * Buffer, unsigned short Length )
{
    unsigned char CRCHi = 0xFF;
    unsigned char CRCLo = 0xFF;
    int           Index;
        unsigned short ret;

    while( Length-- )
    {
        Index = CRCLo ^ *Buffer++ ;
        CRCLo = CRCHi ^ CRC_Table_Hi[Index];
        CRCHi = CRC_Table_Lo[Index];
    }
        ret=((unsigned short)CRCHi << 8);
    ret|= (unsigned short)CRCLo;
    return ret;
}
```

CRC generation functions - Without Table

```
unsigned short ModBus_CRC16( unsigned char * Buffer, unsigned short Length )
{
/* ModBus_CRC16 Calculatd CRC16 with polynome 0xA001 and init value 0xFFFF
Input *Buffer - pointer on data
Input Lenght - number byte in buffer
Output - calculated CRC16
*/
        unsigned int cur_crc;

        cur_crc=0xFFFF;
        do
        {
                unsigned int i = 8;
                cur_crc = cur_crc ^ *Buffer++;
                        do
                {
                        if (0x0001 & cur_crc)
                        {
                                cur_crc >>= 1;
                                cur_crc ^= 0xA001;
                        }
                        else
                        {
                                cur_crc >>= 1;
                        }
                }
                while (--i);
        }
        while (--Length);

        return cur_crc;
}
```

2. READING COMMAND STRUCTURE

In case of module combined with counter: The master communication device can send commands to the module to read its status and setup or to read the measured values, status and setup relevant to the counter.

In case of counter with integrated communication: The master communication device can send commands to the counter to read its status, setup and the measured values.

More registers can be read, at the same time, sending a single command, only if the registers are consecutive (see chapter 5). According to the used MODBUS protocol mode, the read command is structured as follows.

2.1 Modbus ASCII/RTU

Values contained both in Query or Response messages are in hex format.

Query example in case of MODBUS RTU: 01030002000265CB

Example	Byte	Description	No. of bytes
01	-	Slave address	1
03	-	Function code	1
00	High	Starting register	2
02	Low		
00	High	No. of words to be read	2
02	Low		
65	High	Error check (CRC)	2
CB	Low		

Response example in case of MODBUS RTU: 01030400035571F547

Example	Byte	Description	No. of bytes
01	-	Slave address	1
03	-	Function code	1
04	-	Byte count	1
00	High	Requested data	4
03	Low		
55	High		
71	Low		
F5	High	Error check (CRC)	2
47	Low		

2.2 Modbus TCP

Values contained both in Query or Response messages are in hex format.

Query example in case of MODBUS TCP: 010000000006010400020002

Example	Byte	Description	No. of bytes
01	-	Transaction identifier	1
00	High	Protocol identifier	4
00	Low		
00	High		
00	Low		
06	-	Byte count	1
01	-	Unit identifier	1
04	-	Function code	1
00	High	Starting register	2
02	Low		
00	High	No. of words to be read	2
02	Low		

Response example in case of MODBUS TCP: 01000000000701040400035571

Example	Byte	Description	No. of bytes
01	-	Transaction identifier	1
00	High	Protocol identifier	4
00	Low		
00	High		
00	Low		
07	-	Byte count	1
01	-	Unit identifier	1
04	-	Function code	1
04	-	No. of byte of requested data	2
00	High	Requested data	4
03	Low		
55	High		
71	Low		

2.3 Floating Point as per IEEE Standard

The basic format allows a IEEE standard floating-point number to be represented in a single 32 bit format, as shown below:

$$\mathbf{N.n = (-1)^{S}\ 2^{e'-127}\ (1.f\,)}$$

where **S** is the sign bit, **e'** is the first part of the exponent and f is the decimal fraction placed next to 1. Internally the exponent is 8 bits in length and the stored fraction is 23 bits long.

A round to nearest method is applied to the calculated value of floating point.

The floating-point format is shown as follows:

```
=================================
 |  S  |  e + 127  |          f          |
=================================
31        30                 23 22                          0 ←
                                                          bit number
```

where:

	bit length
Sign	1
Exponent	8
Fraction	23 + (1)
Total	m = 32 + (1)

Exponent	
Min e'	0
Max e'	255
Bias	127

NOTE: Fractions (decimals) are always shown while the leading 1 (hidden bit) is not stored.

Example of conversion of value shown with floating point

Value read with floating point:
$45AACC00_{(16)}$

Value converted in binary format:

0	10001011	$01010101100110000000000_{(2)}$
sign	exponent	fraction

sign = 0
exponent = $10001011_{(2)} = 139_{(10)}$
fraction = $01010101100110000000000_{(2)}$ / $8388608\ _{(10)}$ =
= $2804736\ _{(10)}$ / $8388608\ _{(10)}$ = $0.334350585\ _{(10)}$

$N.n = (-1)^{S}\ 2^{e'-127}\ (1+f) =$
$= (-1)^{0}\ 2^{139-127}\ (1.334350585) =$
$= (+1)\ (4096)\ (1.334350585) =$
$= 5465.5$

3. WRITING COMMAND STRUCTURE

In case of module combined with counter: The master communication device can send commands to the module to program itself or to program the counter.

In case of counter with integrated communication: The master communication device can send commands to the counter to program it.

More settings can be carried out, at the same time, sending a single command, only if the relevant registers are consecutive (see chapter 5). According to the used MODBUS protocol type, the write command is structured as follows.

3.1 Modbus ASCII/RTU

Values contained both in Request or Response messages are in hex format.

Query example in case of MODBUS RTU: 011005150001020008F053

Example	Byte	Description	No. of bytes
01	-	Slave address	1
10	-	Function code	1
05	High	Starting register	2
15	Low		
00	High	No. of words to be written	2
01	Low		
02	-	Data byte counter	1
00	High	Data for programming	2
08	Low		
F0	High	Error check (CRC)	2
53	Low		

Response example in case of MODBUS RTU: 01100515000110C1

Example	Byte	Description	No. of bytes
01	-	Slave address	1
10	-	Function code	1
05	High	Starting register	2
15	Low		
00	High	No. of written words	2
01	Low		
10	High	Error check (CRC)	2
C1	Low		

3.2 Modbus TCP

Values contained both in Request or Response messages are in hex format.

Query example in case of MODBUS TCP: 010000000009011005150001020008

Example	Byte	Description	No. of bytes
01	-	Transaction identifier	1
00	High	Protocol identifier	4
00	Low		
00	High		
00	Low		
09	-	Byte count	1
01	-	Unit identifier	1
10	-	Function code	1
05	High	Starting register	2
15	Low		
00	High	No. of words to be written	2
01	Low		
02	-	Data byte counter	1
00	High	Data for programming	2
08	Low		

Response example in case of MODBUS TCP: 010000000006011005150001

Example	Byte	Description	No. of bytes
01	-	Transaction identifier	1
00	High	Protocol identifier	4
00	Low		
00	High		
00	Low		
06	-	Byte count	1
01	-	Unit identifier	1
10	-	Function code	1
05	High	Starting register	2
15	Low		
00	High	Command successfully sent	2
01	Low		

4. EXCEPTION CODES

In case of module combined with counter: When the module receives a not-valid query, an error message (exception code) is sent.

In case of counter with integrated communication: When the counter receives a not-valid query, an error message (exception code) is sent.

According to the used MODBUS protocol mode, possible exception codes are as follows.

4.1 Modbus ASCII/RTU

Values contained in Response messages are in hex format.

Response example in case of MODBUS RTU: 01830131F0

Example	Byte	Description	No. of bytes
01	-	Slave address	1
83	-	Function code (80+03)	1
01	-	Exception code	1
31	High	Error check (CRC)	2
F0	Low		

Exception codes for MODBUS ASCII/RTU are following described:

$01 **ILLEGAL FUNCTION:** the function code received in the query is not an allowable action.
$02 **ILLEGAL DATA ADDRESS:** the data address received in the query is not an allowable address (i.e. the combination of register and transfer length is invalid).
$03 **ILLEGAL DATA VALUE:** a value contained in the query data field is not an allowable value.
$04 **ILLEGAL RESPONSE LENGTH:** the request would generate a response with size bigger than that available for MODBUS protocol.

4.2 Modbus TCP

Values contained in Response messages are in hex format.

Response example in case of MODBUS TCP: 010000000003018302

Example	Byte	Description	No. of bytes
01	-	Transaction identifier	1
00	High	Protocol identifier	4
00	Low		
00	High		
00	Low		
03	-	No. of byte of next data in this string	1
01	-	Unit identifier	1
83	-	Function code (80+03)	1
02	-	Exception code	1

Exception codes for MODBUS TCP are following described:

$01 **ILLEGAL FUNCTION:** the function code is unknown by the server.
$02 **ILLEGAL DATA ADDRESS:** the data address received in the query is not an allowable address for the counter (i.e. the combination of register and transfer length is invalid).
$03 **ILLEGAL DATA VALUE:** a value contained in the query data field is not an allowable value for the counter.
$04 **SERVER FAILURE:** the server failed during the execution.
$05 **ACKNOWLEDGE:** the server accepted the server invocation but the service requires a relatively long time to execute. The server therefore returns only an aknowledgement of the service invocation receipt.
$06 **SERVER BUSY:** the server was unable to accept the MB request PDU. The client application has the responsability of deciding if and when re-sending the request.
$0A **GATEWAY PATH UNAVAILABLE:** the communication module (or the counter, in case of counter with integrated communication) is not configured or cannot communicate.
$0B **GATEWAY TARGET DEVICE FAILED TO RESPOND:** the counter is not available in the network.

5. GENERAL INFORMATION ON REGISTER TABLES

NOTE: Highest number of registers (or bytes) which can be read with a single command:
- 63 registers in ASCII mode
- 127 registers in RTU mode
- 256 bytes in TCP mode

NOTE: Highest number of registers which can be programmed with a single command:
- 13 registers in ASCII mode
- 29 registers in RTU mode
- 1 register in TCP mode

NOTE: The register values are in hex format ($).

Table HEADER	Meaning
PARAMETER	Symbol and description of the parameter to be read/written.
+/-	Positive or negative sign on the read value. The sign representation changes according to communication module or counter model: **Sign Bit Mode**: If this column is checked, the read register value can have positive or negative sign. Convert a signed register value as shown in the following instructions: The Most Significant Bit (MSB) indicates the sign as follows: 0=positive (+), 1=negative (-). Negative value example: **MSB** $8020 = **1**000000000100000 = -32 **| hex |** **bin** **| dec |** **2's Complement Mode**: If this column is checked, the read register value can have positive or negative sign. The negative values are represented with 2's complement.
INTEGER	INTEGER register data. It shows the Unit of measure, the RegSet type and the corresponding Word number and the Address in hex format. Two RegSet types are available: **RegSet 0:** even / odd word registers. **RegSet 1:** even word registers. <u>Not available for LAN GATEWAY modules.</u> Available only for: ▪ Counters with integrated MODBUS ▪ Counters with integrated ETHERNET ▪ RS485 modules with firmware release 2.00 or higher To identify the RegSet in use, please refer to $0523/$0538 registers.
IEEE	IEEE standard register data. It shows the Unit of measure, the Word number and the Address in hex format.
REGISTER AVAILABILITY BY MODEL	Availability of the register according to the model. If checked (●), the register is available for the corresponding model: **3ph 6A/63A/80A SERIAL:** 6A, 63A and 80A 3phase counters with serial communication. **1ph 80A SERIAL:** 80A 1phase counters with serial communication. **1ph 40A SERIAL:** 40A 1phase counters with serial communication. **3ph integrated ETHERNET TCP:** 3phase counters with integrated ETHERNET TCP communication. **1ph integrated ETHERNET TCP:** 1phase counters with integrated ETHERNET TCP communication. **LANG TCP (according to model):** counters combined with LAN GATEWAY module.
DATA MEANING	Description of data received by a response of a reading command.
PROGRAMMABLE DATA	Description of data which can be sent for a writing command.

6. READING REGISTERS (FUNCTION CODES $03, $04)

PARAMETER		+/-	INTEGER					IEEE			REGISTER AVAILABILITY BY MODEL					
			RegSet 0		RegSet 1											
Symbol	Description	Signed	Words	Address	Words	Address	Unit of measure	Words	Address	Unit of measure	3ph 6A/63A/80A SERIAL	1ph 80A SERIAL	1ph 40A SERIAL	3ph Integrated ETHERNET TCP	1ph Integrated ETHERNET TCP	LANG TCP (according to model)
REALTIME VALUES																
U1N	Ph 1-N Voltage		2	0000	2	0000	mV	2	1000	V	●			●		●
U2N	Ph 2-N Voltage		2	0002	2	0002	mV	2	1002	V	●			●		●
U3N	Ph 3-N Voltage		2	0004	2	0004	mV	2	1004	V	●			●		●
U12	L 1-2 Voltage		2	0006	2	0006	mV	2	1006	V	●			●		●
U23	L 2-3 Voltage		2	0008	2	0008	mV	2	1008	V	●			●		●
U31	L 3-1 Voltage		2	000A	2	000A	mV	2	100A	V	●			●		●
U∑	System Voltage		2	000C	2	000C	mV	2	100C	V	●	●	●	●	●	●
A1	Ph1 Current	●	2	000E	2	000E	mA	2	100E	A	●			●		●
A2	Ph2 Current	●	2	0010	2	0010	mA	2	1010	A	●			●		●
A3	Ph3 Current	●	2	0012	2	0012	mA	2	1012	A	●			●		●
AN	Neutral Current	●	2	0014	2	0014	mA	2	1014	A	●			●		●
A∑	System Current	●	2	0016	2	0016	mA	2	1016	A	●	●	●	●	●	●
PF1	Ph1 Power Factor	●	1	0018	2	0018	0.001	2	1018	-	●			●		●
PF2	Ph2 Power Factor	●	1	0019	2	001A	0.001	2	101A	-	●			●		●
PF3	Ph3 Power Factor	●	1	001A	2	001C	0.001	2	101C	-	●			●		●
PF∑	Sys Power Factor	●	1	001B	2	001E	0.001	2	101E	-	●	●	●	●	●	●
P1	Ph1 Active Power	●	3	001C	4	0020	mW	2	1020	W	●			●		●
P2	Ph2 Active Power	●	3	001F	4	0024	mW	2	1022	W	●			●		●
P3	Ph3 Active Power	●	3	0022	4	0028	mW	2	1024	W	●			●		●
P∑	Sys Active Power	●	3	0025	4	002C	mW	2	1026	W	●	●	●	●	●	●
S1	Ph1 Apparent Power	●	3	0028	4	0030	mVA	2	1028	VA	●			●		●
S2	Ph2 Apparent Power	●	3	002B	4	0034	mVA	2	102A	VA	●			●		●
S3	Ph3 Apparent Power	●	3	002E	4	0038	mVA	2	102C	VA	●			●		●
S∑	Sys Apparent Power	●	3	0031	4	003C	mVA	2	102E	VA	●	●	●	●	●	●
Q1	Ph1 Reactive Power	●	3	0034	4	0040	mvar	2	1030	var	●			●		●
Q2	Ph2 Reactive Power	●	3	0037	4	0044	mvar	2	1032	var	●			●		●
Q3	Ph3 Reactive Power	●	3	003A	4	0048	mvar	2	1034	var	●			●		●
Q∑	Sys Reactive Power	●	3	003D	4	004C	mvar	2	1036	var	●	●	●	●	●	●
F	Frequency		1	0040	2	0050	mHz	2	1038	Hz	●			●		●
PH SEQ	Phase Sequence		1	0041	2	0052	-	2	103A	-	●			●		●
	Meaning of read data: INTEGER: $00=123-CCW, $01=321-CW, $02=not defined IEEE for Counters with Integrated Communication and RS485 Modules: $3DFBE76D=123-CCW, $3E072B02=321-CW, $0=not defined IEEE for LAN GATEWAY Modules: $0=123-CCW, $3F800000=321-CW, $40000000=not defined															
-	Reserved		3	0042	-	-	-	-	-	-	R	R	R	R	R	R
TOTAL COUNTERS																
+kWh1	Ph1 Imp. Active En.		3	0100	4	0100	0.1Wh	2	1100	Wh	●			●		●
+kWh2	Ph2 Imp. Active En.		3	0103	4	0104	0.1Wh	2	1102	Wh	●			●		●
+kWh3	Ph3 Imp. Active En.		3	0106	4	0108	0.1Wh	2	1104	Wh	●			●		●
+kWh∑	Sys Imp. Active En.		3	0109	4	010C	0.1Wh	2	1106	Wh	●	●	●	●	●	●
-kWh1	Ph1 Exp. Active En.		3	010C	4	0110	0.1Wh	2	1108	Wh	●			●		●
-kWh2	Ph2 Exp. Active En.		3	010F	4	0114	0.1Wh	2	110A	Wh	●			●		●
-kWh3	Ph3 Exp. Active En.		3	0112	4	0118	0.1Wh	2	110C	Wh	●			●		●
-kWh ∑	Sys Exp. Active En.		3	0115	4	011C	0.1Wh	2	110E	Wh	●	●	●	●	●	●
+kVAh1-L	Ph1 Imp. Lag. Apparent En.		3	0118	4	0120	0.1VAh	2	1110	VAh	●			●		●
+kVAh2-L	Ph2 Imp. Lag. Apparent En.		3	011B	4	0124	0.1VAh	2	1112	VAh	●			●		●
+kVAh3-L	Ph3 Imp. Lag. Apparent En.		3	011E	4	0128	0.1VAh	2	1114	VAh	●			●		●
+kVAh∑-L	Sys Imp. Lag. Apparent En.		3	0121	4	012C	0.1VAh	2	1116	VAh	●	●	●	●	●	●
-kVAh1-L	Ph1 Exp. Lag. Apparent En.		3	0124	4	0130	0.1VAh	2	1118	VAh	●			●		●
-kVAh2-L	Ph2 Exp. Lag. Apparent En.		3	0127	4	0134	0.1VAh	2	111A	VAh	●			●		●
-kVAh3-L	Ph3 Exp. Lag. Apparent En.		3	012A	4	0138	0.1VAh	2	111C	VAh	●			●		●
-kVAh∑-L	Sys Exp. Lag. Apparent En.		3	012D	4	013C	0.1VAh	2	111E	VAh	●	●	●	●	●	●
+kVAh1-C	Ph1 Imp. Lead. Apparent En.		3	0130	4	0140	0.1VAh	2	1120	VAh	●			●		●
+kVAh2-C	Ph2 Imp. Lead. Apparent En.		3	0133	4	0144	0.1VAh	2	1122	VAh	●			●		●
+kVAh3-C	Ph3 Imp. Lead. Apparent En.		3	0136	4	0148	0.1VAh	2	1124	VAh	●			●		●
+kVAh∑-C	Sys Imp. Lead. Apparent En.		3	0139	4	014C	0.1VAh	2	1126	VAh	●	●	●	●	●	●
-kVAh1-C	Ph1 Exp. Lead. Apparent En.		3	013C	4	0150	0.1VAh	2	1128	VAh	●			●		●
-kVAh2-C	Ph2 Exp. Lead. Apparent En.		3	013F	4	0154	0.1VAh	2	112A	VAh	●			●		●
-kVAh3-C	Ph3 Exp. Lead. Apparent En.		3	0142	4	0158	0.1VAh	2	112C	VAh	●			●		●
-kVAh∑-C	Sys Exp. Lead. Apparent En.		3	0145	4	015C	0.1VAh	2	112E	VAh	●	●	●	●	●	●
+kvarh1-L	Ph1 Imp. Lag. Reactive En.		3	0148	4	0160	0.1varh	2	1130	varh	●			●		●
+kvarh2-L	Ph2 Imp. Lag. Reactive En.		3	014B	4	0164	0.1varh	2	1132	varh	●			●		●

PARAMETER		+/-	INTEGER					IEEE			REGISTER AVAILABILITY BY MODEL					
			RegSet 0		RegSet 1											
Symbol	Description	Signed	Words	Address	Words	Address	Unit of measure	Words	Address	Unit of measure	3ph 6A/63A/80A SERIAL	1ph 80A SERIAL	1ph 40A SERIAL	3ph Integrated ETHERNET TCP	1ph Integrated ETHERNET TCP	LANG TCP (according to model)
TOTAL COUNTERS																
+kvarh3-L	Ph3 Imp. Lag. Reactive En.		3	014E	4	0168	0.1varh	2	1134	varh	●			●		●
+kvarh∑-L	Sys Imp. Lag. Reactive En.		3	0151	4	016C	0.1varh	2	1136	varh	●	●	●	●	●	●
-kvarh1-L	Ph1 Exp. Lag. Reactive En.		3	0154	4	0170	0.1varh	2	1138	varh	●			●		●
-kvarh2-L	Ph2 Exp. Lag. Reactive En.		3	0157	4	0174	0.1varh	2	113A	varh	●			●		●
-kvarh3-L	Ph3 Exp. Lag. Reactive En.		3	015A	4	0178	0.1varh	2	113C	varh	●			●		●
-kvarh∑-L	Sys Exp. Lag. Reactive En.		3	015D	4	017C	0.1varh	2	113E	varh	●	●	●	●	●	●
+kvarh1-C	Ph1 Imp. Lead. Reactive En.		3	0160	4	0180	0.1varh	2	1140	varh	●			●		●
+kvarh2-C	Ph2 Imp. Lead. Reactive En.		3	0163	4	0184	0.1varh	2	1142	varh	●			●		●
+kvarh3-C	Ph3 Imp. Lead. Reactive En.		3	0166	4	0188	0.1varh	2	1144	varh	●			●		●
+kvarh∑-C	Sys Imp. Lead. Reactive En.		3	0169	4	018C	0.1varh	2	1146	varh	●	●	●	●	●	●
-kvarh1-C	Ph1 Exp. Lead. Reactive En.		3	016C	4	0190	0.1varh	2	1148	varh	●			●		●
-kvarh2-C	Ph2 Exp. Lead. Reactive En.		3	016F	4	0194	0.1varh	2	114A	varh	●			●		●
-kvarh3-C	Ph3 Exp. Lead. Reactive En.		3	0172	4	0198	0.1varh	2	114C	varh	●			●		●
-kvarh∑-C	Sys Exp. Lead. Reactive En.		3	0175	4	019C	0.1varh	2	114E	varh	●	●	●	●	●	●
-	Reserved		3	0178	2	01A0	-	2	1150	-	**R**	**R**	**R**	**R**	**R**	**R**
TARIFF 1 COUNTERS																
+kWh1-T1	Ph1 Imp. Active En.		3	0200	4	0200	0.1Wh	2	1200	Wh	●					●
+kWh2-T1	Ph2 Imp. Active En.		3	0203	4	0204	0.1Wh	2	1202	Wh	●					●
+kWh3-T1	Ph3 Imp. Active En.		3	0206	4	0208	0.1Wh	2	1204	Wh	●					●
+kWh∑-T1	Sys Imp. Active En.		3	0209	4	020C	0.1Wh	2	1206	Wh	●	●				●
-kWh1-T1	Ph1 Exp. Active En.		3	020C	4	0210	0.1Wh	2	1208	Wh	●					●
-kWh2-T1	Ph2 Exp. Active En.		3	020F	4	0214	0.1Wh	2	120A	Wh	●					●
-kWh3-T1	Ph3 Exp. Active En.		3	0212	4	0218	0.1Wh	2	120C	Wh	●					●
-kWh∑-T1	Sys Exp. Active En.		3	0215	4	021C	0.1Wh	2	120E	Wh	●	●				●
+kVAh1-L-T1	Ph1 Imp. Lag. Apparent En.		3	0218	4	0220	0.1VAh	2	1210	VAh	●					●
+kVAh2-L-T1	Ph2 Imp. Lag. Apparent En.		3	021B	4	0224	0.1VAh	2	1212	VAh	●					●
+kVAh3-L-T1	Ph3 Imp. Lag. Apparent En.		3	021E	4	0228	0.1VAh	2	1214	VAh	●					●
+kVAh∑-L-T1	Sys Imp. Lag. Apparent En.		3	0221	4	022C	0.1VAh	2	1216	VAh	●	●				●
-kVAh1-L-T1	Ph1 Exp. Lag. Apparent En.		3	0224	4	0230	0.1VAh	2	1218	VAh	●					●
-kVAh2-L-T1	Ph2 Exp. Lag. Apparent En.		3	0227	4	0234	0.1VAh	2	121A	VAh	●					●
-kVAh3-L-T1	Ph3 Exp. Lag. Apparent En.		3	022A	4	0238	0.1VAh	2	121C	VAh	●					●
-kVAh∑-L-T1	Sys Exp. Lag. Apparent En.		3	022D	4	023C	0.1VAh	2	121E	VAh	●	●				●
+kVAh1-C-T1	Ph1 Imp. Lead. Apparent En.		3	0230	4	0240	0.1VAh	2	1220	VAh	●					●
+kVAh2-C-T1	Ph2 Imp. Lead. Apparent En.		3	0233	4	0244	0.1VAh	2	1222	VAh	●					●
+kVAh3-C-T1	Ph3 Imp. Lead. Apparent En.		3	0236	4	0248	0.1VAh	2	1224	VAh	●					●
+kVAh∑-C-T1	Sys Imp. Lead. Apparent En.		3	0239	4	024C	0.1VAh	2	1226	VAh	●	●				●
-kVAh1-C-T1	Ph1 Exp. Lead. Apparent En.		3	023C	4	0250	0.1VAh	2	1228	VAh	●					●
-kVAh2-C-T1	Ph2 Exp. Lead. Apparent En.		3	023F	4	0254	0.1VAh	2	122A	VAh	●					●
-kVAh3-C-T1	Ph3 Exp. Lead. Apparent En.		3	0242	4	0258	0.1VAh	2	122C	VAh	●					●
-kVAh∑-C-T1	Sys Exp. Lead. Apparent En.		3	0245	4	025C	0.1VAh	2	122E	VAh	●	●				●
+kvarh1-L-T1	Ph1 Imp. Lag. Reactive En.		3	0248	4	0260	0.1varh	2	1230	varh	●					●
+kvarh2-L-T1	Ph2 Imp. Lag. Reactive En.		3	024B	4	0264	0.1varh	2	1232	varh	●					●
+kvarh3-L-T1	Ph3 Imp. Lag. Reactive En.		3	024E	4	0268	0.1varh	2	1234	varh	●					●
+kvarh∑-L-T1	Sys Imp. Lag. Reactive En.		3	0251	4	026C	0.1varh	2	1236	varh	●	●				●
-kvarh1-L-T1	Ph1 Exp. Lag. Reactive En.		3	0254	4	0270	0.1varh	2	1238	varh	●					●
-kvarh2-L-T1	Ph2 Exp. Lag. Reactive En.		3	0257	4	0274	0.1varh	2	123A	varh	●					●
-kvarh3-L-T1	Ph3 Exp. Lag. Reactive En.		3	025A	4	0278	0.1varh	2	123C	varh	●					●
-kvarh∑-L-T1	Sys Exp. Lag. Reactive En.		3	025D	4	027C	0.1varh	2	123E	varh	●	●				●
+kvarh1-C-T1	Ph1 Imp. Lead. Reactive En.		3	0260	4	0280	0.1varh	2	1240	varh	●					●
+kvarh2-C-T1	Ph2 Imp. Lead. Reactive En.		3	0263	4	0284	0.1varh	2	1242	varh	●					●
+kvarh3-C-T1	Ph3 Imp. Lead. Reactive En.		3	0266	4	0288	0.1varh	2	1244	varh	●					●
+kvarh∑-C-T1	Sys Imp. Lead. Reactive En.		3	0269	4	028C	0.1varh	2	1246	varh	●	●				●
-kvarh1-C-T1	Ph1 Exp. Lead. Reactive En.		3	026C	4	0290	0.1varh	2	1248	varh	●					●
-kvarh2-C-T1	Ph2 Exp. Lead. Reactive En.		3	026F	4	0294	0.1varh	2	124A	varh	●					●
-kvarh3-C-T1	Ph3 Exp. Lead. Reactive En.		3	0272	4	0298	0.1varh	2	124C	varh	●					●
-kvarh∑-C-T1	Sys Exp. Lead. Reactive En.		3	0275	4	029C	0.1varh	2	124E	varh	●	●				●
-	Reserved		3	0278	-	-	-	-	-	-	**R**	**R**	**R**	**R**	**R**	**R**

PARAMETER		+/-	INTEGER					IEEE			REGISTER AVAILABILITY BY MODEL					
			RegSet 0		RegSet 1											
Symbol	Description	Signed	Words	Address	Words	Address	Unit of measure	Words	Address	Unit of measure	3ph 6A/63A/80A SERIAL	1ph 80A SERIAL	1ph 40A SERIAL	3ph Integrated ETHERNET TCP	1ph Integrated ETHERNET TCP	LANG TCP (according to model)
TARIFF 2 COUNTERS																
+kWh1-T2	Ph1 Imp. Active En.		3	0300	4	0300	0.1Wh	2	1300	Wh	●					●
+kWh2-T2	Ph2 Imp. Active En.		3	0303	4	0304	0.1Wh	2	1302	Wh	●					●
+kWh3-T2	Ph3 Imp. Active En.		3	0306	4	0308	0.1Wh	2	1304	Wh	●					●
+kWh∑-T2	Sys Imp. Active En.		3	0309	4	030C	0.1Wh	2	1306	Wh	●	●				●
-kWh1-T2	Ph1 Exp. Active En.		3	030C	4	0310	0.1Wh	2	1308	Wh	●					●
-kWh2-T2	Ph2 Exp. Active En.		3	030F	4	0314	0.1Wh	2	130A	Wh	●					●
-kWh3-T2	Ph3 Exp. Active En.		3	0312	4	0318	0.1Wh	2	130C	Wh	●					●
-kWh∑-T2	Sys Exp. Active En.		3	0315	4	031C	0.1Wh	2	130E	Wh	●	●				●
+kVAh1-L-T2	Ph1 Imp. Lag. Apparent En.		3	0318	4	0320	0.1VAh	2	1310	VAh	●					●
+kVAh2-L-T2	Ph2 Imp. Lag. Apparent En.		3	031B	4	0324	0.1VAh	2	1312	VAh	●					●
+kVAh3-L-T2	Ph3 Imp. Lag. Apparent En.		3	031E	4	0328	0.1VAh	2	1314	VAh	●					●
+kVAh∑-L-T2	Sys Imp. Lag. Apparent En.		3	0321	4	032C	0.1VAh	2	1316	VAh	●	●				●
-kVAh1-L-T2	Ph1 Exp. Lag. Apparent En.		3	0324	4	0330	0.1VAh	2	1318	VAh	●					●
-kVAh2-L-T2	Ph2 Exp. Lag. Apparent En.		3	0327	4	0334	0.1VAh	2	131A	VAh	●					●
-kVAh3-L-T2	Ph3 Exp. Lag. Apparent En.		3	032A	4	0338	0.1VAh	2	131C	VAh	●					●
-kVAh∑-L-T2	Sys Exp. Lag. Apparent En.		3	032D	4	033C	0.1VAh	2	131E	VAh	●	●				●
+kVAh1-C-T2	Ph1 Imp. Lead. Apparent En.		3	0330	4	0340	0.1VAh	2	1320	VAh	●					●
+kVAh2-C-T2	Ph2 Imp. Lead. Apparent En.		3	0333	4	0344	0.1VAh	2	1322	VAh	●					●
+kVAh3-C-T2	Ph3 Imp. Lead. Apparent En.		3	0336	4	0348	0.1VAh	2	1324	VAh	●					●
+kVAh∑-C-T2	Sys Imp. Lead. Apparent En.		3	0339	4	034C	0.1VAh	2	1326	VAh	●	●				●
-kVAh1-C-T2	Ph1 Exp. Lead. Apparent En.		3	033C	4	0350	0.1VAh	2	1328	VAh	●					●
-kVAh2-C-T2	Ph2 Exp. Lead. Apparent En.		3	033F	4	0354	0.1VAh	2	132A	VAh	●					●
-kVAh3-C-T2	Ph3 Exp. Lead. Apparent En.		3	0342	4	0358	0.1VAh	2	132C	VAh	●					●
-kVAh∑-C-T2	Sys Exp. Lead. Apparent En.		3	0345	4	035C	0.1VAh	2	132E	VAh	●	●				●
+kvarh1-L-T2	Ph1 Imp. Lag. Reactive En.		3	0348	4	0360	0.1varh	2	1330	varh	●					●
+kvarh2-L-T2	Ph2 Imp. Lag. Reactive En.		3	034B	4	0364	0.1varh	2	1332	varh	●					●
+kvarh3-L-T2	Ph3 Imp. Lag. Reactive En.		3	034E	4	0368	0.1varh	2	1334	varh	●					●
+kvarh∑-L-T2	Sys Imp. Lag. Reactive En.		3	0351	4	036C	0.1varh	2	1336	varh	●	●				●
-kvarh1-L-T2	Ph1 Exp. Lag. Reactive En.		3	0354	4	0370	0.1varh	2	1338	varh	●					●
-kvarh2-L-T2	Ph2 Exp. Lag. Reactive En.		3	0357	4	0374	0.1varh	2	133A	varh	●					●
-kvarh3-L-T2	Ph3 Exp. Lag. Reactive En.		3	035A	4	0378	0.1varh	2	133C	varh	●					●
-kvarh∑-L-T2	Sys Exp. Lag. Reactive En.		3	035D	4	037C	0.1varh	2	133E	varh	●	●				●
+kvarh1-C-T2	Ph1 Imp. Lead. Reactive En.		3	0360	4	0380	0.1varh	2	1340	varh	●					●
+kvarh2-C-T2	Ph2 Imp. Lead. Reactive En.		3	0363	4	0384	0.1varh	2	1342	varh	●					●
+kvarh3-C-T2	Ph3 Imp. Lead. Reactive En.		3	0366	4	0388	0.1varh	2	1344	varh	●					●
+kvarh∑-C-T2	Sys Imp. Lead. Reactive En.		3	0369	4	038C	0.1varh	2	1346	varh	●	●				●
-kvarh1-C-T2	Ph1 Exp. Lead. Reactive En.		3	036C	4	0390	0.1varh	2	1348	varh	●					●
-kvarh2-C-T2	Ph2 Exp. Lead. Reactive En.		3	036F	4	0394	0.1varh	2	134A	varh	●					●
-kvarh3-C-T2	Ph3 Exp. Lead. Reactive En.		3	0372	4	0398	0.1varh	2	134C	varh	●					●
-kvarh∑-C-T2	Sys Exp. Lead. Reactive En.		3	0375	4	039C	0.1varh	2	134E	varh	●	●				●
-	Reserved		3	0378	-	-	-	-	-	-	R	R	R	R	R	R
PARTIAL COUNTERS																
+kWh∑-P	Sys Imp. Active En.		3	0400	4	0400	0.1Wh	2	1400	Wh	●	●	●	●	●	●
-kWh∑-P	Sys Exp. Active En.		3	0403	4	0404	0.1Wh	2	1402	Wh	●	●	●	●	●	●
+kVAh∑-L-P	Sys Imp. Lag. Apparent En.		3	0406	4	0408	0.1VAh	2	1404	VAh	●	●	●	●	●	●
-kVAh∑-L-P	Sys Exp. Lag. Apparent En.		3	0409	4	040C	0.1VAh	2	1406	VAh	●	●	●	●	●	●
+kVAh∑-C-P	Sys Imp. Lead. Apparent En.		3	040C	4	0410	0.1VAh	2	1408	VAh	●	●	●	●	●	●
-kVAh∑-C-P	Sys Exp. Lead. Apparent En.		3	040F	4	0414	0.1VAh	2	140A	VAh	●	●	●	●	●	●
+kvarh∑-L-P	Sys Imp. Lag. Reactive En.		3	0412	4	0418	0.1varh	2	140C	varh	●	●	●	●	●	●
-kvarh∑-L-P	Sys Exp. Lag. Reactive En.		3	0415	4	041C	0.1varh	2	140E	varh	●	●	●	●	●	●
+kvarh∑-C-P	Sys Imp. Lead. Reactive En.		3	0418	4	0420	0.1varh	2	1410	varh	●	●	●	●	●	●
-kvarh∑-C-P	Sys Exp. Lead. Reactive En.		3	041B	4	0424	0.1varh	2	1412	varh	●	●	●	●	●	●
BALANCE COUNTERS																
kWh∑-B	Sys Active En.	●	3	041E	4	0428	0.1Wh	2	1414	Wh	●	●		●	●	●
kVAh∑-L-B	Sys Lag. Apparent En.	●	3	0421	4	042C	0.1VAh	2	1416	VAh	●	●		●	●	●
kVAh∑-C-B	Sys Lead. Apparent En.	●	3	0424	4	0430	0.1VAh	2	1418	VAh	●	●		●	●	●
kvarh∑-L-B	Sys Lag. Reactive En.	●	3	0427	4	0434	0.1varh	2	141A	varh	●	●		●	●	●
kvarh∑-C-B	Sys Lead. Reactive En.	●	3	042A	4	0438	0.1varh	2	141C	varh	●	●		●	●	●
-	Reserved		3	042D	-	-	-	-	-	-	R	R	R	R	R	R

PARAMETER		INTEGER				DATA MEANING	REGISTER AVAILABILITY BY MODEL					
		RegSet 0		RegSet 1								
Symbol	Description	Words	Address	Words	Address	Values	3ph 6A/63A/80A SERIAL	1ph 80A SERIAL	1ph 40A SERIAL	3ph Integrated ETHERNET TCP	1ph Integrated ETHERNET TCP	LANG TCP (according to model)
INFORMATION ON ENERGY COUNTER AND COMMUNICATION MODULE												
EC SN	Counter Serial Number	5	0500	6	0500	10 ASCII chars. ($00…$FF)	●	●	●	●	●	●
EC MODEL	Counter Model	1	0505	2	0506	$03=6A 3phases, 4wires $08=80A 3phases, 4wires $0C=80A 1phase, 2wires $10=40A 1phase, 2wires $12=63A 3phases, 4wires	●	●	●	●	●	●
EC TYPE	Counter Type	1	0506	2	0508	$00=NO MID, RESET $01=NO MID $02=MID $03=NO MID, Wiring selection $05=MID no varh $09=MID, Wiring selection $0A=MID no varh, Wiring selection $0B=NO MID, RESET, Wiring selection	●	●	●	●	●	●
EC FW REL1	Counter Firmware Release 1	1	0507	2	050A	Convert the read Hex value in Dec value. e.g. $66=102 => rel. 1.02	●	●	●	●	●	●
EC HW VER	Counter Hardware Version	1	0508	2	050C	Convert the read Hex value in Dec value. e.g. $64=100 => ver. 1.00	●	●	●	●	●	●
-	Reserved	2	0509	2	050E	-	**R**	**R**	**R**	**R**	**R**	**R**
T	Tariff in use	1	050B	2	0510	$01=tariff 1 $02=tariff 2	●	●				●
PRI/SEC	Primary/Secondary Value Only 6A model. Reserved and fixed to 0 for other models.	1	050C	2	0512	$00=primary $01=secondary	●			●		●
ERR	Error Code	1	050D	2	0514	Bit field coding: - bit0 (LSb)=Phase sequence - bit1=Memory - bit2=Clock (RTC)-Only ETH model - other bits not used Bit=1 means error condition, Bit=0 means no error	●	●	●	●	●	●
CT	CT Ratio Value Only 6A model. Reserved and fixed to 1 for other models.	1	050E	2	0516	$0001…$2710	●			●		●
-	Reserved	2	050F	2	0518	-	**R**	**R**	**R**	**R**	**R**	**R**
FSA	FSA Value	1	0511	2	051A	$00=1A $01=5A $02=80A $03=40A $06=63A	●	●	●	●	●	●
WIR	Wiring Mode	1	0512	2	051C	$01=3phases, 4 wires, 3 currents $02=3phases, 3 wires, 2 currents $03=1phase $04=3phases, 3 wires, 3 currents	●	●	●	●	●	●
ADDR	MODBUS Address	1	0513	2	051E	$01…$F7	●	●	●	●	●	●
MDB MODE	MODBUS Mode	1	0514	2	0520	$00=7E2 (ASCII) $01=8N1 (RTU)	●	●	●			
BAUD	Communication Speed	1	0515	2	0522	$01=300 bps $02=600 bps $03=1200 bps $04=2400 bps $05=4800 bps $06=9600 bps $07=19200 bps $08=38400 bps $09=57600 bps	●	●	●			
-	Reserved	1	0516	2	0524	-	**R**	**R**	**R**	**R**	**R**	**R**
INFORMATION ON ENERGY COUNTER AND COMMUNICATION MODULE												
EC-P STAT	Partial Counter Status	1	0517	2	0526	Bit field coding: - bit0 (LSb)= +kWhΣ PAR - bit1=-kWhΣ PAR - bit2=+kVAhΣ-L PAR - bit3=-kVAhΣ-L PAR - bit4=+kVAhΣ-C PAR - bit5=-kVAhΣ-C PAR - bit6=+kvarhΣ-L PAR - bit7=-kvarhΣ-L PAR - bit8=+kvarhΣ-C PAR - bit9=-kvarhΣ-C PAR - other bits not used Bit=1 means counter active, Bit=0 means counter stopped	●	●	●	●	●	●

PARAMETER		INTEGER				DATA MEANING	REGISTER AVAILABILITY BY MODEL					
		RegSet 0		RegSet 1								
Symbol	**Description**	**Words**	**Address**	**Words**	**Address**	**Values**	**3ph 6A/63A/80A SERIAL**	**1ph 80A SERIAL**	**1ph 40A SERIAL**	**3ph Integrated ETHERNET TCP**	**1ph Integrated ETHERNET TCP**	**LANG TCP** (according to model)
MOD SN	Module Serial Number	5	0518	6	0528	10 ASCII chars. ($00…$FF)	●	●				●
SIGN	Signed Value Representation	1	051D	2	052E	$00=sign bit $01=2's complement	●	●	●	●	●	
-	Reserved	1	051E	2	0530	-	**R**	**R**	**R**	**R**	**R**	**R**
MOD FW REL	Module Firmware Release	1	051F	2	0532	Convert the read Hex value in Dec value. e.g. $66=102 => rel. 1.02	●	●				●
MOD HW VER	Module Hardware Version	1	0520	2	0534	Convert the read Hex value in Dec value. e.g. $64=100 => ver. 1.00	●	●				●
-	Reserved	2	0521	2	0536	-	**R**	**R**	**R**	**R**	**R**	**R**
REGSET	RegSet in use	1	0523	2	0538	$00=register set 0 $01=register set 1	●	●		●	●	
		2	0538	2	0538	$00=register set 0 $01=register set 1			●			
FW REL2	Counter Firmware Release 2	1	0600	2	0600	Convert the read Hex value in Dec value. e.g. $C8=200 => rel. 2.00	●	●	●	●	●	●
RTC-DAY	Ethernet interface RTC day	1	2000	1	2000	Convert the read Hex value in Dec value. e.g. $1F=31 => day 31				●	●	
RTC-MONTH	Ethernet interface RTC month	1	2001	1	2001	Convert the read Hex value in Dec value. e.g. $0C=12 => december				●	●	
RTC-YEAR	Ethernet interface RTC year	1	2002	1	2002	Convert the read Hex value in Dec value. e.g. $15=21 => year 2021				●	●	
RTC-HOURS	Ethernet interface RTC hours	1	2003	1	2003	Convert the read Hex value in Dec value. e.g. $0F=15 => 15 hours				●	●	
RTC-MIN	Ethernet interface RTC minutes	1	2004	1	2004	Convert the read Hex value in Dec value. e.g. $1E=30 => 30 minutes				●	●	
RTC-SEC	Ethernet interface RTC seconds	1	2005	1	2005	Convert the read Hex value in Dec value. e.g. $0A=10 => 10 seconds				●	●	

NOTE: the RTC registers ($2000…$2005) are available only for energy meters with Ethernet Firmware rel. 1.15 or higher.

7. COILS READING (FUNCTION CODE $01)

<table>
<tr><th colspan="2">PARAMETER</th><th colspan="2">INTEGER</th><th>DATA MEANING</th><th colspan="6">REGISTER AVAILABILITY BY MODEL</th></tr>
<tr><th>Symbol</th><th>Description</th><th>Bits</th><th>Address</th><th>Values</th><th>3ph 6A/63A/80A SERIAL</th><th>1ph 80A SERIAL</th><th>1ph 40A SERIAL</th><th>3ph Integrated ETHERNET TCP</th><th>1ph Integrated ETHERNET TCP</th><th>LANG TCP (according to model)</th></tr>
<tr><td>AL</td><td>Alarms</td><td>40</td><td>0000</td><td>Bit sequence bit 39 (MSb) … bit 0 (LSb):
|U3N-L|U2N-L|U1N-L|UΣ-L|U3N-H|U2N-H|U1N-H|UΣ-H|
COM	RES	U31-L	U23-L	U12-L	U31-H	U23-H	U12-H
RES	RES	RES	RES	RES	RES	AN-L	A3-L
A2-L	A1-L	AΣ-L	AN-H	A3-H	A2-H	A1-H	AΣ-H
RES	RES	RES	RES	RES	RES	RES	f-O	 LEGEND L=Under the Threshold (Low) H=Over the Threshold (High) O=Out of Range COM=Communication on IR port OK. Do not consider in case of models with integrated SERIAL communication RES=Bit Reserved to 0 NOTE: Voltage, Current and Frequency Threshold Values can change according to the counter model. Please refer to the tables shown below.</td><td>•</td><td>•</td><td></td><td>•</td><td>•</td><td>•</td></tr>
</table>

<table>
<tr><th rowspan="2">VOLTAGE AND FREQUENCY RANGES ACCORDING TO MODEL</th><th colspan="4">PARAMETER THRESHOLDS</th></tr>
<tr><th>PHASE-NEUTRAL VOLTAGE</th><th>PHASE-PHASE VOLTAGE</th><th>CURRENT</th><th>FREQUENCY</th></tr>
<tr><td>3x230/400V 50Hz</td><td>ULN-L=230V-20%=184V
ULN-H=230V+20%=276V</td><td>ULL-L=230V x $\sqrt{3}$ -20%=318V
ULL-H=230V x $\sqrt{3}$ +20%=478V</td><td rowspan="2">I-L=Starting Current (I_{st})
I-H=Current Full Scale (I_{FS})</td><td rowspan="2">f-L=45Hz
f-H=65Hz</td></tr>
<tr><td>3x230/400…3x240/415V 50/60Hz</td><td>ULN-L=230V-20%=184V
ULN-H=240V+20%=288V</td><td>ULL-L=398V-20%=318V
ULL-H=415V+20%=498V</td></tr>
</table>

8. WRITING REGISTERS (FUNCTION CODE $10)

PARAMETER		INTEGER				PROGRAMMABLE DATA	REGISTER AVAILABILITY BY MODEL					
		RegSet 0		RegSet 1								
Symbol	Description	Words	Address	Words	Address	Values	3ph 6A/63A/80A SERIAL	1ph 80A SERIAL	1ph 40A SERIAL	3ph Integrated ETHERNET TCP	1ph Integrated ETHERNET TCP	LANG TCP (according to model)
PROGRAMMABLE DATA FOR ENERGY COUNTER AND COMMUNICATION MODULE												
ADDR	MODBUS Address	1	0513	2	051E	$01…$F7	●	●	●	●	●	●
MDB MODE	MODBUS Mode	1	0514	2	0520	$00=7E2 (ASCII) $01=8N1 (RTU)	●	●				
BAUD	Communication Speed *300, 600, 1200, 57600 values not available for 40A model.	1	0515	2	0522	$01=300 bps* $02=600 bps* $03=1200 bps* $04=2400 bps $05=4800 bps $06=9600 bps $07=19200 bps $08=38400 bps $09=57600 bps*	●	●	●			
EC RES	Reset Energy Counters Only type with RESET function	1	0516	2	0524	$00=TOTAL Counters $03=ALL Counters	●	●	●	●	●	●
						$01=TARIFF 1 Counters $02=TARIFF 2 Counters	●	●				●
EC-P OPER	Partial Counter Operation	1	0517	2	0526	For RegSet1, set the MS word always to 0000. The LS word must be structured as follows: Byte 1 – PARTIAL Counter Selection $00=+kWhΣ PAR $01=-kWhΣ PAR $02=+kVAhΣ-L PAR $03=-kVAhΣ-L PAR $04=+kVAhΣ-C PAR $05=-kVAhΣ-C PAR $06=+kvarhΣ-L PAR $07=-kvarhΣ-L PAR $08=+kvarhΣ-C PAR $09=-kvarhΣ-C PAR $0A=ALL Partial Counters Byte 2 – PARTIAL Counter Operation $01=start $02=stop $03=reset **e.g. Start +kWhΣ PAR Counter** **00=+kWhΣ PAR** **01=start** **Final value to be set:** **-RegSet0=0001** **-RegSet1=00000001**	●	●	●	●	●	●
REGSET	RegSet switching	1	100B	2	1010	$00=switch to RegSet 0 $01=switch to RegSet 1	●	●		●	●	
		2	0538	2	0538	$00=switch to RegSet 0 $01=switch to RegSet 1			●			
RTC-DAY	Ethernet interface RTC day	1	2000	1	2000	$01…$1F (1…31)				●	●	
RTC-MONTH	Ethernet interface RTC month	1	2001	1	2001	$01…$0C (1…12)				●	●	
RTC-YEAR	Ethernet interface RTC year	1	2002	1	2002	$01…$25 (1…37=2001…2037) **e.g. to set 2021, write $15**				●	●	
RTC-HOURS	Ethernet interface RTC hours	1	2003	1	2003	$00…$17 (0…23)				●	●	
RTC-MIN	Ethernet interface RTC minutes	1	2004	1	2004	$00…$3B (0…59)				●	●	
RTC-SEC	Ethernet interface RTC seconds	1	2005	1	2005	$00…$3B (0…59)				●	●	

NOTE: the RTC registers ($2000…$2005) are available only for energy meters with Ethernet Firmware rel. 1.15 or higher.

NOTE: if the RTC writing command contains inappropriate values (e.g. 30th February), the value will not be accepted and the device replies with exception code (Illegal Value).

NOTE: in case of RTC lost due to long time power off, set again the RTC value (day, month, year, hours, min, sec) to restart the recordings.