# Autometers Modbus Protocol Specification V6.3.23

# Title page 1

## Autometers MODBUS Protocol

***by Autometers***

*This document details the MODBUS comunication protocol for the Autometers product range.*

# Autometers Modbus Protocol Specification



Printed: December 2017 in England

# Table of Contents

# Part I

# 1 Introduction

## 1.1 Scope of this document

This document provides generic information for Autometers devices implementing the Modbus RTU and Modbus ASCII Serial Communications protocol. Configuration information relating to specific devices is supplied in separate user manuals.

## 1.2 Protocol overview

For a detailed description of the Modbus protocol please vis the web site www.modbus.org where the latest specs can be found.

## 1.3 Contacts

For further help and assistance please use the following contacts:

Autometers Systems
Tel: +44(0)161 861 9056
www.autometers.co.uk

## 1.4 Glossary

2W The Two-Wire configuration defined in the “Electrical Interface” chapter, or one of its interfaces.

4W The Four-Wire configuration defined in the “Electrical Interface” chapter, or one of its interfaces.

PDU Protocol Data Unit

## 1.5 Document Changes

| Document modifications | | |
|---|---|---|
| **Rev** | **Date** | **Modification** |
| 6.3.0 | 25/11/2009 | New Release |
| 6.3.1 | 09/07/2010 | Added new fields to the Misc section |
| 6.3.2 | 05/10/2010 | Added new fields to the Energy Rate Registers |
| 6.3.3 | 28/10/2010 | Added new Device ID |
| 6.3.4 | 11/11/2010 | Changed IC5-8 to IC5-7 |
| 6.3.5 | 12/01/2011 | Added new Device ID's |
| 6.3.6 | 13/01/2011 | Added new Device ID |
| 6.3.7 | 21/01/2011 | Minor Text Changes |
| 6.3.8 | 01/09/2011 | Added example to read the Energy Register |
| 6.3.9 | 18/11/2011 | Added new Device ID for IC-5-1 |
| 6.3.10 | 18/11/2011 | Added new Device ID for AD1-125MBIC |
| 6.3.11 | 09/07/2012 | Added registers for heatmeters |
| 6.3.12 | 11/07/2012 | Added new ID for AD1-80MBIC |
| 6.3.13 | 22/10/2012 | Added registers and device ID for Modbus HCC |
| 6.3.14 | 19/12/2012 | Removed old device ID's |
| 6.3.15 | 31/07/2013 | Added new device ID's |
| 6.3.16 | 14/08/2014 | Added new device ID's . Changed AM34 to P2000-D and P2000-T |
| 6.3.17 | 09/09/2014 | Added new device IDs |
| 6.3.18 | 26/11/2014 | Added new device IDs |
| 6.3.19 | 19/01/2015 | Changed device ID 0x27 for C2000 |
| 6.3.20 | 16/06/2015 | Added ID's for new meters |
| 6.3.21 | 29/10/2015 | Added power direction register |
| 6.3.22 | 07/12/2017 | Updated register size for power direction |
| 6.3.23 | 20/12/2017 | Added generic device id for future MBus meters |

# Part II

# 2 MODBUS Transmission Modes

Two different serial transmission modes are defined : The RTU mode and the ASCII mode.

It defines the bit contents of message fields transmitted serially on the line. It determines how information is packed into the message
fields and decoded.

The transmission mode (and serial port parameters) must be the same for all devices on a MODBUS Serial Line.

Although the ASCII mode is required in some specific applications, interoperability between MODBUS devices can be reached only if
each device has the same transmission mode :
All devices must implement the RTU Mode. The ASCII transmission mode is an option.

Devices should be set up by the users to the desired transmission mode, RTU or ASCII.
Default setup must be the RTU mode.

## 2.1 RTU Transmission mode

When devices communicate on a MODBUS serial line using the RTU (Remote Terminal Unit) mode, each 8–bit byte in a message contains two 4–bit hexadecimal characters. The main advantage of this mode is that its greater character density allows better data throughput than ASCII mode for the same baud rate. Each message must be transmitted in a continuous stream of characters.

**The format ( 11 bits ) for each byte in RTU mode is :**

| | |
|---|---|
| Coding System: | 8–bit binary |
| Bits per Byte: | 1 start bit |
| | 8 data bits, least significant bit sent first |
| | 1 bit for parity completion |
| | 1 stop bit |

**Even parity is required**, other modes ( odd parity, no parity ) may also be used. In order to ensure a maximum compatibility with other products, it is recommended to support also No parity mode. The default parity mode must be even parity.
Remark : the use of no parity requires 2 stop bits.

**How Characters are Transmitted Serially :**
Each character or byte is sent in this order (left to right):
Least Significant Bit (LSB) . . . Most Significant Bit (MSB)

Devices may accept by configuration either Even, Odd, or No Parity checking. If No Parity is implemented, an additional stop bit is
transmitted to fill out the character frame to a full 11-bit asynchronous character :

**Frame Checking Field :**
Cyclical Redundancy Checking (CRC)

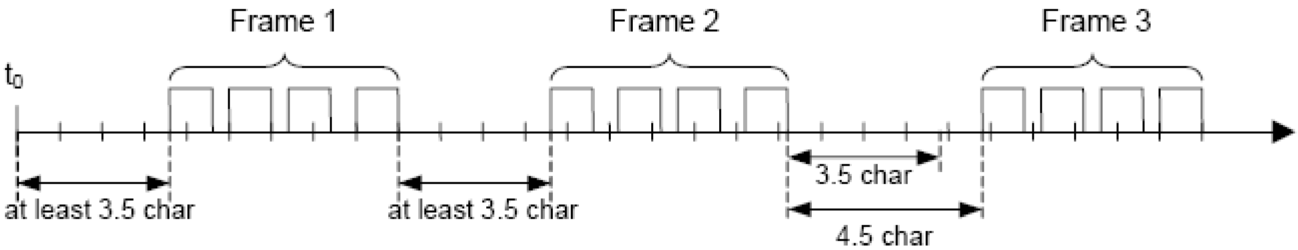
**Frame description :**

| Slave Address | Function Code | Data | CRC |
|---|---|---|---|
| 1 byte | 1 byte | 0 up to 252 byte(s) | 2 bytes<br>CRC Low, CRC Hi |

The maximum size of a MODBUS RTU frame is 256 bytes.

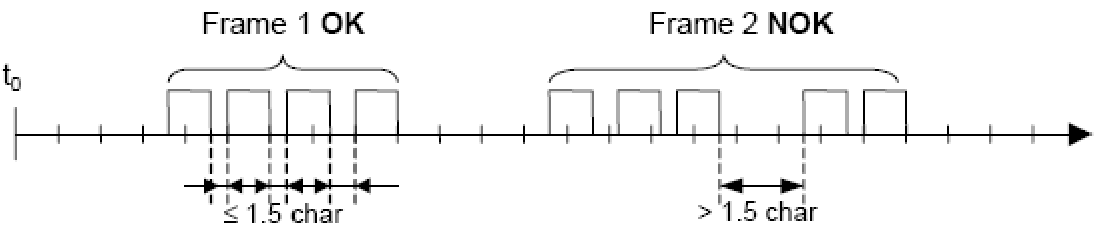
## 2.1.1 RTU Message Framing

A MODBUS message is placed by the transmitting device into a frame that has a known beginning and ending point. This allows
devices that receive a new frame to begin at the start of the message, and to know when the message is completed. Partial messages must be detected and errors must be set as a result.
In RTU mode, message frames are separated by a silent interval of at least 3.5 character times. In the following sections, this time interval is called t3,5.




| Start | MODBUS message: Address | Function | Data | CRC Check | End |
|---|---|---|---|---|---|
| ≥ 3.5 char | 8 bits | 8 bits | N x 8 bits | 16 bits | ≥ 3.5 char |

The entire message frame must be transmitted as a continuous stream of characters.
If a silent interval of more than 1.5 character times occurs between two characters, the message frame is declared incomplete and should be discarded by the receiver.




**Note:**
The implementation of RTU reception driver may imply the management of a lot of interruptions due to the t1.5 and t3.5 timers. With
high communication baud rates, this leads to a heavy CPU load. Consequently these two

timers must be strictly respected when the
baud rate is equal or lower than 19200 Bps. For baud rates greater than 19200 Bps, fixed values for the 2 timers should be used: it is
recommended to use a value of 750µs for the inter-character time-out (t1.5) and a value of 1.750ms for inter-frame delay (t3.5).

### 2.1.2 RTU CRC Checking

The RTU mode includes an error–checking field that is based on a Cyclical Redundancy Checking (CRC) method performed on the
message contents.

The CRC field checks the contents of the entire message. It is applied regardless of any parity checking method used for the individual characters of the message.

The CRC field contains a 16–bit value implemented as two 8–bit bytes.

The CRC field is appended to the message as the last field in the message. When this is done, the low–order byte of the field is appended first, followed by the high–order byte. The CRC high–order byte is the last byte to be sent in the message.

The CRC value is calculated by the sending 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 calculation is started by first pre-loading 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 (eight) shift, the next 8–bit byte is exclusive ORed with the register's current value, and the process repeats for eight more shifts as described above. The final content of the register, after all the bytes of the message have been applied, is the CRC value.

When the CRC is appended to the message, the low-order byte is appended first, followed by the high-order byte. A detailed example
of CRC generation is given below:

#### 2.1.2.1 CRC Generation

The Cyclical Redundancy Checking (CRC) field is two bytes, containing a 16–bit binary value. The CRC value is calculated by the transmitting device, which appends the CRC to the message. The device that receives 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 content of the register, after all the characters of the message have been applied, is the CRC value.

**A procedure for generating a CRC is:**

1. Load a 16–bit register with FFFF hex (all 1's). 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 0xA001 (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 content 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 highorder byte.

### Example

An example of a C language function performing CRC generation is shown on the following pages. 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.

Note: This function performs the swapping of the high/low CRC bytes internally. The bytes are already swapped in the CRC value that
is returned from the function.

Therefore the CRC value returned from the function can be directly placed into the message for transmission.
The function takes two arguments:

unsigned char *puchMsg; A pointer to the message buffer containing binary data to be used for generating the CRC
unsigned short usDataLen; The quantity of bytes in the message buffer.

### CRC Generation Function

```
unsigned short CRC16 ( puchMsg, usDataLen ) /* The function returns the CRC as a unsigned short typ
unsigned char *puchMsg ; /* message to calculate CRC upon */
unsigned short usDataLen ; /* quantity of bytes in message */
{
	unsigned char uchCRCHi = 0xFF ; /* high byte of CRC initialized */
	unsigned char uchCRCLo = 0xFF ; /* low byte of CRC initialized */
	unsigned uIndex ; /* will index into CRC lookup table */
	while (usDataLen--) /* pass through message buffer */
	{
		uIndex = uchCRCLo ^ *puchMsg++ ; /* calculate the CRC */
		uchCRCLo = uchCRCHi ^ auchCRCHi[uIndex] ;
		uchCRCHi = auchCRCLo[uIndex] ;
	}

	return (uchCRCHi << 8 | uchCRCLo) ;
}
```

### High-Order Byte Table

```
/* Table of CRC values for high–order byte */
static unsigned char auchCRCHi[] = {
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
} ;
```

## Low-Order Byte Table

```
/* Table of CRC values for low–order byte */
static char auchCRCLo[] = {
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
};
```

## 2.2 ASCII Transmission mode

Interchange) mode, each 8–bit byte in a message is sent as two ASCII characters. This mode is used when the physical
communication link or the capabilities of the device does not allow the conformance with RTU mode requirements regarding timers
management.

**Note:** this mode is less efficient than RTU since each byte needs two characters.

**Example:**
The byte 0X5B is encoded as two characters : 0x35 and 0x42 ( 0x35 ="5", and 0x42 ="B" in ASCII ).

**The format ( 10 bits ) for each byte in ASCII mode is :**
Coding System: Hexadecimal, ASCII characters 0–9, A–F
One hexadecimal character contains 4-bits of data within each ASCII character of the message
Bits per Byte: 1 start bit
7 data bits, least significant bit sent first
1 bit for parity completion;
1 stop bit

**Even parity is required**, other modes ( odd parity, no parity ) may also be used. In order to ensure a maximum compatibility with
other products, it is recommended to support also No parity mode. The default parity mode must be Even parity.

**Note :** the use of no parity requires 2 stop bits.

**How Characters are Transmitted Serially :**
Each character or byte is sent in this order (left to right):
Least Significant Bit (LSB) . . . Most Significant Bit (MSB)

Devices may accept by configuration either Even, Odd, or No Parity checking. If No Parity is implemented, an additional stop bit is
transmitted to fill out the character frame :

**Frame Checking Field:** Longitudinal Redundancy Checking (LRC)

### 2.2.1 ASCII Message Framing

A MODBUS message is placed by the transmitting device into a frame that has a known beginning and ending point. This allows devices that receive a new frame to begin at the start of the message, and to know when the message is completed. Partial messages must be detected and errors must be set as a result.

The address field of a message frame contains two characters.

In ASCII mode, a message is delimited by specific characters as Start-of-frames and End-of-frames. A message must start with a ‘colon’ ( : ) character (ASCII 3A hex), and end with a ‘carriage return – line feed’ (CRLF) pair (ASCII 0D and 0A hex).

**Note :** The LF character can be changed using a specific MODBUS application command ( see MODBUS application protocol  specification).

The allowable characters transmitted for all other fields are hexadecimal 0–9, A–F (ASCII coded). The devices monitor the bus continuously for the ‘colon’ character. When this character is received, each device decodes the next character until it detects the End-Of-Frame.

Intervals of up to one second may elapse between characters within the message. Unless the user has configured a longer timeout, an interval greater than 1 second means an error has occurred. Some Wide-Area-Network application may require a timeout in the 4 to 5 second range.

A typical message frame is shown below.

| Start | Address | Function | Data | LRC | End |
|---|---|---|---|---|---|
| 1 char<br>: | 2 chars | 2 chars | 0 up to 2x252 char(s) | 2 chars | 2 chars<br>CR,LF |

**Note :** Each data byte needs two characters for encoding. Thus, to ensure compatibility at MODBUS application level between ASCII mode and RTU mode, the maximum data size for ASCII data field (2x252) is the double the maximum data size for RTU data field (252).

Consequently, the maximum size of a MODBUS ASCII frame is 513 characters.

### 2.2.2 ASCII LRC Checking

In ASCII mode, messages include an error–checking field that is based on a Longitudinal Redundancy Checking (LRC) calculation that is performed on the message contents, exclusive of the beginning ‘colon’ and terminating CRLF pair characters. It is applied regardless of any parity checking method used for the individual characters of the message.

The LRC field is one byte, containing an 8–bit binary value. The LRC value is calculated by the device that emits, which appends the LRC to the message. The device that receives calculates 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 of the message, discarding any carries, and then two’s complementing the result. It is performed on the bytes of the message, before the encoding of each byte in the two ASCII characters corresponding to the hexadecimal representation of each nibble. The computation does not include the 'colon' character that begins the message, and does not include the CRLF pair at the end of the message.

The resulting LRC is ASCII encoded into two bytes and placed at the end of the ASCII mode frame before the CRLF.

A detailed example of LRC generation is given in LRC Generation.

#### 2.2.2.1 LRC Generation

The Longitudinal Redundancy Checking (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 device that receives 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 CRLF. Add them into an 8–bit field, so that carries will be discarded.
2. Subtract the final field value from FF hex (all 1's), to produce the ones–complement.
3. Add 1 to produce the twos–complement.

**Placing the LRC into the Message**

When 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.

**Example:** an example of a C language function performing LRC generation is shown below.

The function takes two arguments:
unsigned char *auchMsg; A pointer to the message buffer containing binary data to be used for generating the LRC,
unsigned short usDataLen; The quantity of bytes in the message buffer.

**LRC Generation Function**

```
static unsigned char LRC(auchMsg, usDataLen) /* the function returns the LRC as a type unsigned cha
unsigned char *auchMsg ; /* message to calculate LRC upon */
unsigned short usDataLen ; /* quantity of bytes in message */
{
	unsigned char uchLRC = 0 ; /* LRC char initialized */
	while (usDataLen––) /* pass through message buffer */
	{
		uchLRC += *auchMsg++ ; /* add buffer byte without carry */
	}
	return ((unsigned char)(–((char)uchLRC))) ; /* return twos complement */
}
```

## 2.3 Data Signal Rates

An Autometers Slave device supports the following baud rates

| Baud Rate | Comments |
|---|---|
| 1200 | |
| 2400 | |
| 4800 | |
| 9600 | |
| 19200 | |
| 38400 | |
| 56000 | |
| 115000 | |

Every implemented baud rate must be respected better than 1% in transmission situation, and must accept an error of 2% in reception situation.

# Part III

# 3 Data Formats

## 3.1 unsigned 16-bit integer word Format

The Modbus applications support 16 bit integer information for several of the function codes.

A read or write to a modbus register comprises a 2 x 8 bit byte.

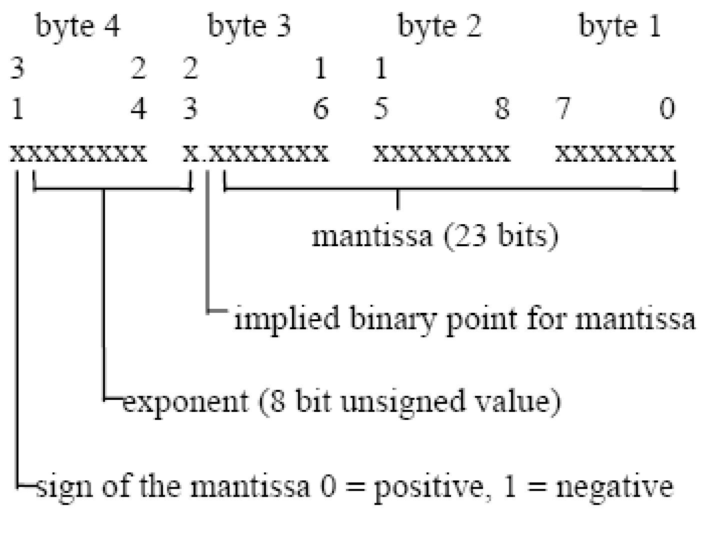
## 3.2 IEEE 32-bit Floating-Point Register Format

The Modbus applications support IEEE 32-bit floating-point information for several of the function codes.

**IEEE Floating-Point Data Format**

The formula for calculating the floating-point number is:

$$\text{mantissa} \times 2^{(\text{exponent} - 127)}$$




**Byte Format**

**Mantissa and Sign**

The mantissa is defined by a sign bit (31), and a 23-bit binary fraction. This binary fraction is combined with an “implied” value of 1 to create a mantissa value, which is greater than or equal to 1.0 and less than 2.0.

The mantissa is positive if the sign bit is zero (reset), and negative if the sign bit is one (set). For example:

| DECIMAL | HEXADECIMAL | BINARY |
|---|---|---|
| 100 | 42C80000 | 01000010 11001000 00000000 00000000 |

The sign bit (31) is zero, indicating a positive mantissa. Removing the sign bits and exponent bits, the mantissa becomes:

| HEXADECIMAL | BINARY |
|---|---|
| 480000 | xxxxxxxx x1001000 00000000 00000000 |

Add an "implied" value of one to the left of the binary point:

BINARY
1.1001000 00000000 00000000

Using positioned notation, this binary number is equal to:

$$1.0 + (1x2^{-1}) + (0x2^{-2}) + (0x2^{-3}) + (1x2^{-4}) = 1.0 + 0.5 + 0.0 + 0.0 + 0.0625 = 1.5625$$

**Exponent**

The exponent is defined by an unsigned 8-bit binary value (bits 23 through 30). The value of the exponent is derived by performing a signed subtraction of 127 (decimal) from the 8-bit exponent value.

| DECIMAL | HEXADECIMAL | BINARY |
|---|---|---|
| 100 | 42C80000 | 01000010 11001000 00000000 00000000 |

removing the sign and mantissa bits, the exponent becomes:

| DECIMAL | HEXADECIMAL | BINARY |
|---|---|---|
| 133 | 85 | x1000010 1xxxxxxx xxxxxxxx xxxxxxxx |

or:

$$1x2^7 + 0x2^6 + 0x2^5 + 0x2^4 + 0x2^3 + 1x2^2 + 0x2^1 + 1x2^0$$

Subtract a bias of 127 (decimal) from the exponent to determine its value: 133 – 127 = 6.

**Mantissa and Exponent Combination**

Combining the mantissa and exponent from the two previous examples:

$$\text{float number} = \text{mantissa x } 2^{\text{exponent}}$$

$$\text{float number} = 1.5625 \text{ x } 2^6 = 1.5625 \text{ x } 64 = 100.0$$

**Reserved Operands**

Per the Standard certain exceptional forms of floating-point operands are excluded from the numbering system. These are as follows:

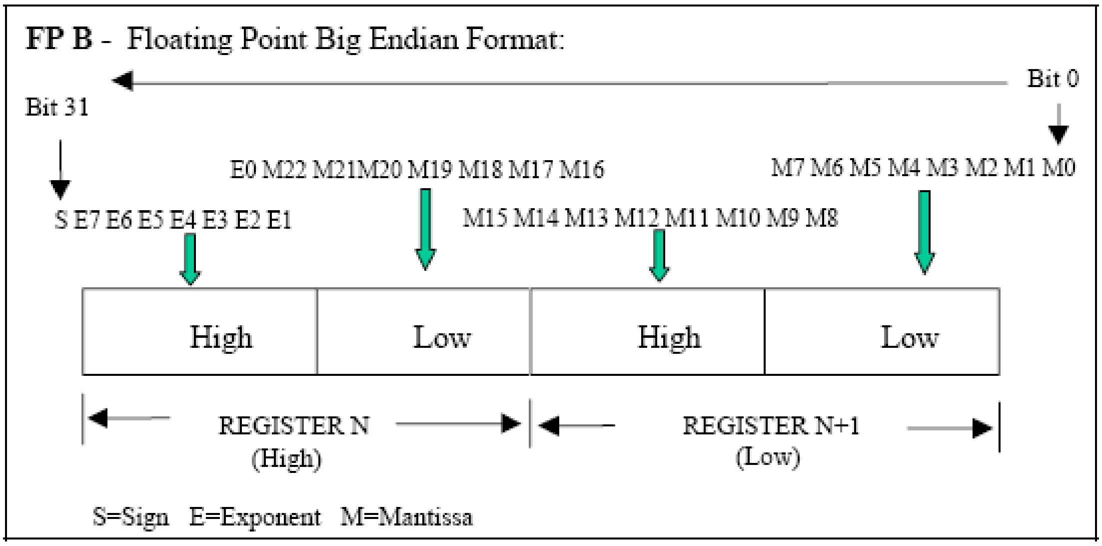
| EXCEPTION | EXPONENT | MANTISSA |
|---|---|---|
| +/- Infinity | All 1's | All 0's |
| Not-a-Number (NAN) | All 1's | Other than 0's |
| Denormalized Number | All 0's | Other than 0's |
| Zero | All 0's | All 0's |

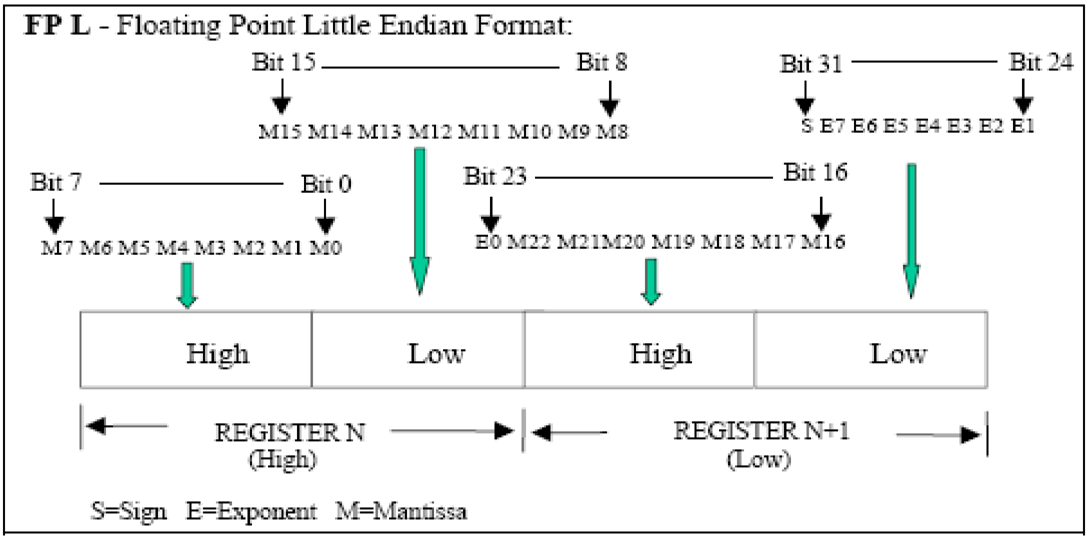
### 3.2.1 MODBUS Double Register Format

Each IEEE 32-bit floating point number requires two consecutive registers (four bytes) starting with the register defined as the starting register for the information. The stuffing order of the bytes into the two registers differs among Modbus hosts.

The selections are:

| Selection | Description | Byte Order See byte format | Notes |
|---|---|---|---|
| High Word First | Floating Point Big Endian Format | 4, 3, 2, 1 | |
| Low Word First | Floating Point Little Endian Format | 1, 2, 3, 4 | |




FP L - Floating Point Little Endian Format:
Bit 15 Bit 8
Bit 31 Bit 24
M15 M14 M13 M12 M11 M10 M9 M8
S E7 E6 E5 E4 E3 E2 E1
Bit 7 Bit 0
Bit 23 Bit 16
M7 M6 M5 M4 M3 M2 M1 M0
E0 M22 M21M20 M19 M18 M17 M16
High
Low
High
Low
REGISTER N
(High)
REGISTER N+1
(Low)
S=Sign E=Exponent M=Mantissa

# Part IV

# 4 MODBUS Function Codes

The Autometers Modbus RTU and Modbus ASCII protocols uses a subset of the standard Modbus function codes to provide access to measurement and information registers. These standard function codes provide basic support for IEEE 32-bit floating point numbers, 16 bit integer and 8 bit char values.

| Function Code | Name | Usage |
|---|---|---|
| 0x04 | Read Input Registers | Used for reading floating point and 16 bit integer measurements |
| 0x06 | Write Single Register | Used for writing floating point and 16 bit integer values to a single register |
| 0x10 | Write Multiple Registers | Used for writing floating point and 16 bit integer values to multiple registers |
| 0x11 | Report Device ID | Used for reading device information including device ID, description, software version etc |
| 0x2B / 0x0E | Read Device Identification | Used for reading device identification and additional information |

## 4.1 Function Code 0x04

This function code is used to read from 1 to 125 contiguous input registers in a remote device. The Request PDU specifies the starting register address and the number of registers.

In the PDU Registers are addressed starting at zero. Therefore input registers numbered 1-16 are addressed as 0-15.

The register data in the response message are packed as two bytes per register, with the binary contents right justified within each byte. For each register, the first byte contains the high order bits and the second contains the low order bits.

### Request

| | | |
|---|---|---|
| **Function Code** | 1 byte | **0x04** |
| **Starting Address** | 2 bytes | 0x0000 to 0xFFFF |
| **Quantity of Input Registers** | 2 bytes | 0x0001 to 0x007D |

### Response

| | | |
|---|---|---|
| **Function Code** | 1 byte | **0x04** |
| **Byte Count** | 1 bytes | 2 * N(1) |
| **Input Registers** | N(1) * 2 bytes | |

Where N = Quantity of Input Registers

### Error

| | | |
|---|---|---|
| **Error Code** | 1 byte | **0x84** |
| **Exception Code** | 1 bytes | 0x01 or 0x02 or 0x03 or 0x04 |

An Example of a request to read input register 9 from slave address 2 using RTU format, where the register contains the 16 bit hex value 0x55AA.

### Request

| Field Name | (Hex) |
|---|---|
| **Slave Address** | 0x02 |
| **Function Code** | 0x04 |
| **Starting Address Hi** | 0x00 |
| **Starting Address Lo** | 0x08 |
| **Quantity of Input Register Hi** | 0x00 |
| **Quantity of Input Register Lo** | 0x01 |
| **Check Sum** | CRC |
| **Check Sum** | CRC |

### Response

| Field Name | (Hex) |
|---|---|
| Slave Address | 0x02 |
| Function Code | 0x04 |
| Byte Count | 0x02 |
| Input Register Hi | 0x55 |
| Input Register Lo | 0xAA |
| Check Sum | CRC |
| Check Sum | CRC |

Where CRC is the check sum for the message see RTU CRC Checking for details

An Example of a request to read input register 9 from slave address 3 using ASCII format, where the register contains the 16 bit hex value 0x1234.

### Request

| Field Name | Char |
|---|---|
| Start Char | : |
| Slave Address | 0 |
| Slave Address | 3 |
| Function Code | 0 |
| Function Code | 4 |
| Starting Address Hi | 0 |
| Starting Address Hi | 0 |
| Starting Address Lo | 0 |
| Starting Address Lo | 8 |
| Quantity of Input Register Hi | 0 |
| Quantity of Input Register Hi | 0 |
| Quantity of Input Register Lo | 0 |
| Quantity of Input Register Lo | 1 |
| Check Sum | LRC |
| Check Sum | LRC |
| End | CR |
| End | LF |

### Response

| Field Name | Char |
|---|---|
| Start Char | : |
| Slave Address | 0 |
| Slave Address | 3 |
| Function Code | 0 |
| Function Code | 4 |

| | |
|---|---|
| **Byte Count** | 0 |
| **Byte Count** | 2 |
| **Input Register Hi** | 1 |
| **Input Register Hi** | 2 |
| **Input Register Lo** | 3 |
| **Input Register Lo** | 4 |
| **Check Sum** | LRC |
| **Check Sum** | LRC |
| **End** | CR |
| **End** | LF |

Where LRC is the check sum for the message see ASCII LRC Checking for details.

## 4.2 Function Code 0x06

This function code is used to write a single holding register in a remote device.

The Request PDU specifies the address of the register to be written. Registers are addressed starting at zero. Therefore register numbered 1 is addressed as 0.

The normal response is an echo of the request, returned after the register contents have been written.

### Request

| | | |
|---|---|---|
| **Function Code** | 1 byte | **0x06** |
| **Register Address** | 2 bytes | 0x0000 to 0xFFFF |
| **Register Value** | 2 bytes | 0x0000 to 0xFFFF |

### Response

| | | |
|---|---|---|
| **Function Code** | 1 byte | **0x06** |
| **Register Address** | 2 bytes | 0x0000 to 0xFFFF |
| **Register Value** | 2 bytes | 0x0000 to 0xFFFF |

### Error

| | | |
|---|---|---|
| **Error Code** | 1 byte | **0x86** |
| **Exception Code** | 1 bytes | 0x01 or 0x02 or 0x03 or 0x04 |

### Example

An Example of a writing to register 30001 (Primary VT Ratio) the value 400, to slave address 5 in RTU mode

### Request

| Field Name | (Hex) |
|---|---|
| **Slave Address** | 0x05 |
| **Function Code** | 0x06 |
| **Register Address Hi** | 0x00 |
| **Register Address Lo** | 0x00 |
| **Register Value Hi** | 0x01 |
| **Register Value Lo** | 0x90 |
| **Check Sum** | CRC |
| **Check Sum** | CRC |

### Response

| Field Name | (Hex) |
|---|---|
| **Slave Address** | 0x05 |
| **Function Code** | 0x06 |
| **Register Address Hi** | 0x00 |
| **Register Address Lo** | 0x00 |
| **Register Value Hi** | 0x01 |
| **Register Value Lo** | 0x90 |
| **Check Sum** | CRC |
| **Check Sum** | CRC |

## 4.3 Function Code 0x10

This function code is used to write multiple holding register in a remote device.

The Request PDU specifies the address of the register to be written. Registers are addressed starting at zero. Therefore register numbered 1 is addressed as 0.

The normal response is an echo of the request, returned after the register contents have been written.

### Request

| | | |
|---|---|---|
| **Function Code** | 1 byte | **0x10** |
| **Start Register Address** | 2 bytes | 0x0000 to 0xFFFF |
| **Quantity of Register** | 2 bytes | 0x0000 to 0xFFFF |
| **Byte Count** | 1 bytes | 2 x N* |
| **Register Value** | N* x 2 bytes | value |

***N = Quantity of Registers**

### Response

| | | |
|---|---|---|
| **Function Code** | 1 byte | **0x10** |
| **Start Register Address** | 2 bytes | 0x0000 to 0xFFFF |
| **Quantity of Register** | 2 bytes | 1 to 123 (0x7B) |

### Error

| | | |
|---|---|---|
| **Error Code** | 1 byte | **0x90** |
| **Exception Code** | 1 bytes | 0x01 or 0x02 or 0x03 or 0x04 |

### Example

An Example of a writing to register 30915 (Pulse value for power) the value 1.0, to slave address 5 in RTU mode

### Request

| Field Name | (Hex) |
|---|---|
| **Slave Address** | 0x05 |
| **Function Code** | 0x10 |
| **Start Register Address Hi** | 0x03 |
| **Start Register Address Lo** | 0x92 |
| **Quantity Of Registers Hi** | 0x00 |
| **Quantity Of Registers Lo** | 0x02 |
| **Byte Count** | 0x04 |
| **Register Value Hi** | 0x3F |

| | |
|---|---|
| **Value** | 0x80 |
| **Value** | 0x00 |
| **Register Value Lo** | 0x00 |
| **Check Sum** | 0x77 |
| **Check Sum** | 0x26 |

## Response

| Field Name | (Hex) |
|---|---|
| **Slave Address** | 0x05 |
| **Function Code** | 0x10 |
| **Start Register Address Hi** | 0x03 |
| **Start Register Address Lo** | 0x92 |
| **Quantity Of Registers Hi** | 0x00 |
| **Quantity Of Registers Lo** | 0x02 |
| **Check Sum** | 0xe1 |
| **Check Sum** | 0xe5 |

## 4.4 Function Code 0x11

This function code is used to read the description of the type, the current status, and other information specific to a remote device.

The format of a normal response is shown in the following example. The data contents are specific to each type of device.

### Request

| | | |
|---|---|---|
| **Function Code** | 1 byte | **0x11** |

### Response

| | | |
|---|---|---|
| **Function Code** | 1 byte | **0x11** |
| **Byte Count** | 1 byte | 0x1A |
| **Device ID** | 1 byte | See Device ID Table |
| **Run Indicator(1)** | 1 byte | 0x00 = OFF<br>0xFE = Limited Function<br>0xFF = Fully Functional |
| **Description** | 16 bytes | See Save Description Table |
| **Serial Number** | 4 bytes | 0 to 4294967295 |
| **Hardware Version Engine** | 2 bytes | |
| **Hardware Version Coms** | 2 bytes | |
| **Hardware Version Display** | 2 bytes | |

### Error

| | | |
|---|---|---|
| **Error Code** | 1 byte | **0x91** |
| **Exception Code** | 1 bytes | 0x01 or 0x04 |

### NOTES

**1**: RUN Indicator: on meters with two modbus channels, if the Main Modbus channel is disabled the run indicator will be set to 0xFE

### Device ID Table

| Device Type | Device ID (Hex) |
|---|---|
| HC1 | 0x01 |
| IC970 (V6 Only) | 0x02 |
| IC990 | 0x03 |
| IC200 | 0x04 |
| Reserved | 0x05 |
| SU8 | 0x06 |
| IC-5-7 | 0x08 |

| Device Type | Device ID (Hex) |
|---|---|
| IC-5-9 | 0x09 |
| AMS12 | 0x0A |
| AMS34 | 0x0B |
| Reserved | 0x0C |
| Reserved | 0x0D |
| IC995 | 0x0E |
| Display-Small | 0x0F |
| Display-Intermediate | 0x10 |
| Smart Box | 0x11 |
| AD1-80C | 0x12 |
| AD1-80MC | 0x13 |
| AD1-125MC | 0x14 |
| AD3-5C | 0x15 |
| AD3-5MC | 0x16 |
| AD3-80C | 0x17 |
| AD3-80MC | 0x18 |
| AD3-125MC | 0x19 |
| ADP3-5C | 0x1A |
| ADP3-80C | 0x1B |
| AD1-125C | 0x1C |
| AD3-125C | 0x1F |
| AD3-80MBIC | 0x20 |
| AD3-65MBIC | 0x21 |
| AD3-5MBIC | 0x22 |
| AD1-125MBIC | 0x23 |
| AD1-80MBIC | 0x24 |
| AD Res | 0x25 |
| IC-5-1 | 0x26 |
| C2000 | 0x27 |
| P2000-D | 0x28 |
| HCCv1 | 0x29 |
| P2000-T | 0x2A |
| HCCv2 | 0x2B |
| P2000-D3 | 0x2C |
| P2000-T3 | 0x2D |
| AD1-45MBIC | 0x2E |
| Cold Water Meter | 0x2F |
| Heat Meter | 0x30 |
| Cooling Meter | 0x31 |
| SDM-220MT | 0x32 |
| DDSD285 | 0x33 |
| A300-MT | 0x34 |
| A300-M | 0x35 |
| A300-P - not used | 0x36 |

| Device Type | Device ID (Hex) |
|---|---|
| A35-M | 0x37 |
| A35-MT | 0x38 |
| A35-MB - not used | 0x39 |
| A35-P - not used | 0x3A |
| A-100M | 0x3F |
| A-100MT | 0x40 |
| IC1070 | 0x41 |
| IC1095 | 0x42 |
| IC1075 | 0x4D |
| Reserved | 0x4E |
| Hot Water Meter | 0x4F |
| Gas Meter | 0x50 |
| Oil Meter | 0x51 |
| Other Meter | 0x52 |

## Slave Description Table

This is a 16 Character ASCII Message. The value is right justified and padded with 0x00 (null) to make up the 16 Characters

| Device Type | Slave Description (16 chars) |
|---|---|
| IC970 | "IC970 xxx.yy" |
| IC200 | "IC200 xxx.yy" |
| SU8 | "SU8 xxx.yy" |
| IC200MF | "IC200MF xxx.yy" |
| IC-5-7 | "IC-5-7 xxx.yy" |
| IC-5-9 | "IC-5-9 xxx.yy" |
| AMS12 | "AMS12 xxx.yy" |
| AMS34 | "AMS34 xxx.yy" |
| IC995 | "IC995 xxx.yy" |
| Display-Small | "Display xxx.yy" |
| Display-Intermediate | "Display xxx.yy" |
| Smart Box | "Smart Box xxx.yy" |
| AD1-80C | "AD1-80C xxx.yy" |
| AD1-80MC | "AD1-80MC xxx.yy" |
| AD1-125MC | "AD1-125MC xxx.yy" |
| AD3-5C | "AD3-5C xxx.yyy" |
| AD3-5MC | "AD3-5MC xxx.yy" |
| AD3-80C | "AD3-80C xxx.yy" |
| AD3-80MC | "AD3-80MC xxx.yy" |
| AD3-125MC | "AD3-125MC xxx.yy" |
| ADP3-5C | "ADP3-5C xxx.yy" |
| ADP3-80C | "ADP3-80C xxx.yy" |

| Device Type | Slave Description (16 chars) |
|---|---|
| AD1-125C | "AD1-125C xxx.yy" |
| AD3-80MBIC | "AD3-80MBIC xxx.yyy" |
| IC-5-1 | "IC-5-1 xxx.yy" |
| AMS12M | "AMS12M xxx.yy" |
| P2000-D | "P2000-D xxx.yy" |
| HCC | "HCC xxx.yy" |
| P2000-T | "P2000-T xxx.yy" |
| HCC2 | "HCC2 xxx.yy" |
| P2000-D3 | "P2000-D3 xxx.yy" |
| P2000-T3 | "P2000-T3 xxx.yy" |
| AD1-45MBIC | "AD1-45MBIC  xxx.yy" |
| A12-10M | "A12-10M xxx.yy" |

where **xxx.yy** is the software version of the device

An example of a request to report the ID and status from an RTU slave 3.
Device is a IC990 version 003.01
With Software version 10.84
With Software version Comms 1.34
With Software version Display 0.0 (not fitted)
Serial number 789123

### Request

| Field Name | (Hex) |
|---|---|
| **Slave Address** | 0x03 |
| **Function Code** | 0x11 |
| **Check Sum** | CRC |
| **Check Sum** | CRC |

### Response

| Field Name | (Hex) |
|---|---|
| **Slave Address** | 0x03 |
| **Function Code** | 0x11 |
| **Byte Count** | 0x1A |
| **Device ID** | 0x01 |
| **Run Indicator** | 0xFF |
| **Description[0]** | 'I' |
| **Description[1]** | 'C' |
| **Description[2]** | '9' |
| **Description[3]** | '9' |
| **Description[4]** | '0' |
| **Description[5]** | ' ' |

| Field Name | (Hex) |
|---|---|
| Description[6] | '0' |
| Description[7] | '0' |
| Description[8] | '3' |
| Description[9] | '.' |
| Description[10] | '0' |
| Description[11] | '1' |
| Description[12] | 0x00 |
| Description[13] | 0x00 |
| Description[14] | 0x00 |
| Description[15] | 0x00 |
| Serial Number Hi | 0x00 |
| Serial Number Hi | 0x0C |
| Serial Number Lo | 0x0A |
| Serial Number Lo | 0x83 |
| Software Version Engine | 0x03 |
| Software Version Engine | 0x01 |
| Software Version Coms | 0x0A |
| Software Version Coms | 0x54 |
| Software Version Display | 0x00 |
| Software Version Display | 0x00 |
| Check Sum | CRC |
| Check Sum | CRC |

## 4.5 Function Code 0x2B/0x0E

This function code allows reading the identification and additional information relative to the physical and functional description of a remote device, only.
The Read Device Identification interface is modeled as an address space composed of a set of addressable data elements. The data elements are called objects and an object Id identifies them.

The interface consists of 3 categories of objects :

- Basic Device Identification. All objects of this category are mandatory : VendorName, Product code, and revision number.

- Regular Device Identification. In addition to Basic data objects, the device provides additional and optional identification and description data objects. All of the objects of this category are defined in the standard but their implementation is optional .

- Extended Device Identification. In addition to regular data objects, the device provides additional and optional identification and description private data about the physical device itself. All of these data are device dependent.

| Object ID | Object Name / Description | Type | M/O | Category |
|---|---|---|---|---|
| 0x00 | VendorName | ASCII String | Mandatory | Basic |
| 0x01 | ProductCode | ASCII String | Mandatory | |
| 0x02 | MajorMinorRevision | ASCII String | Mandatory | |
| 0x03 | VendorUrl | ASCII String | Optional | Regular |
| 0x04 | ProductName | ASCII String | Optional | |
| 0x05 | ModelName | ASCII String | Optional | |
| 0x06 | UserApplicationName | ASCII String | Optional | |
| 0x80 | DeviceSerialNumber | ASCII String | Optional | Extended |
| 0x81 | DeviceEngineFirmwareVersion | ASCII String | Optional | |
| 0x82 | DeviceCommsFirmwareVersion | ASCII String | Optional | |
| 0x83 | DeviceDisplayFirmwareVersion | ASCII String | Optional | |

### Request

| | | |
|---|---|---|
| **Function Code** | 1 byte | **0x2B** |
| **MEI Type** | 1 byte | 0x0E |
| **Read Device ID Code** | 1 byte | 0x01 / 0x0 / 0x03 / 0x04 |
| **Object ID** | 1 byte | 0x00 to 0xFF |

### Response

| | | |
|---|---|---|
| **Function Code** | 1 byte | **0x2B** |
| **MEI Type** | 1 byte | 0x0E |
| **Read Device ID Code** | 1 byte | 0x01 / 0x0 / 0x03 / 0x04 |
| **Conformity Level** | 1 byte | 0x01 / 0x0 / 0x03 / 0x04<br>0x81 / 0x82 / 0x83 |
| **More Follows** | 1 byte | 0x00 / 0xFF |
| **Next object ID** | 1 byte | Object ID number |
| **Number of objects** | 1 byte | |
| **List of** | | |
| **Object ID** | 1 byte | |
| **Object Length** | 1 byte | |
| **Object Data** | Object Length | Depending on the object Data |

### Error

| | | |
|---|---|---|
| **Error Code** | 1 byte | **0xAB** |
| **Exception Code** | 1 bytes | 0x01 / 0x02 / 0x03 / 0x04 |

### Request parameters description :

A MODBUS Encapsulated Interface assigned number 14 identifies the Read identification request.

The parameter " Read Device ID code " allows to define four access types

| Read Device Code | Description |
|---|---|
| 0x01 | Request to get the Basic device identification (Stream access) |
| 0x02 | Request to get the Regular device identification (Stream access) |
| 0x03 | Request to get the Extended device identification (Stream access) |
| 0x04 | Request to get one specific identification object (individual access) |

An exception code 03 is sent back in the response if the Read device ID code is illegal. In case of a response that does not fit into a single response, several transactions (request/response ) must be done. The Object Id byte gives the identification of the first object to obtain. For the first transaction, the client must set the Object Id to 0 to obtain the beginning of the device identification data. For the following transactions, the client must set the Object Id to the value returned by the server in its previous response.

**Note:** An object is indivisible, therefore any object must have a size consistent with the size of transaction response.

If the Object Id does not match any known object, the server responds as if object 0 were pointed out (restart at the beginning).
In case of an individual access: ReadDevId code 04, the Object Id in the request gives

the identification of the object to obtain, and if the Object Id doesn't match to any known object, the server returns an exception response with exception code = 02 (Illegal data address).

If the server device is asked for a description level ( readDevice Code )higher that its conformity level , It must respond in accordance with its actual conformity level.

**Response parameter description :**

| | |
|---|---|
| **Function code** | Function code 43 (decimal) 0x2B (hex) |
| **MEI Type** | 14 (0x0E) MEI Type assigned number for Device Identification Interface |
| **ReadDevId code** | Same as request ReadDevId code : 01, 02, 03 or 04 |
| **Conformity Level** | Identification conformity level of the device and type of supported access.<br>• 0x01 basic identification (stream access only)<br>• 0x02 regular identification (stream access only)<br>• 0x03 extended identification (stream access only)<br>• 0x81 basic identification (stream access and individual access)<br>• 0x82 regular identification (stream access and individual access)<br>• 0x83 extended identification(stream access and individual access) |
| **More Follows** | In case of ReadDevId codes 01, 02 or 03 (stream access),<br>If the identification data doesn't fit into a single response, several request/response transactions may be required.<br>• 0x00 No more objects are available<br>• 0xFF other identification Object are available and further MODBUS transactions are required<br><br>**In case of ReadDevId code 04 (individual access), this field must be set to 00**. |
| **Next Object Id** | If "MoreFollows = FF", identification of the next Object to be asked for.<br>If "MoreFollows = 00", must be set to 00 (useless) |
| **Number Of Objects** | Number of identification Object returned in the response<br>(for an individual access, Number Of Objects = 1) |
| **Object0.Id** | Identification of the first Object returned in the PDU (stream access) or the requested Object (individual access) |
| **Object0.Length** | Length of the first Object in byte |
| **Object0.Value** | Value of the first Object (Object0.Length bytes) |
| | |
| **ObjectN.Id** | Identification of the last Object (within the response) |
| **ObjectN.Length** | Length of the last Object in byte |

| | |
|---|---|
| **ObjectN.Value** | Value of the last Object (ObjectN.Length bytes) |

## Example 1

An Example of reading the Device Serial Number request "Basic device Identification" from Slave address 3 in RTU mode

In this example, all the information is sent in one PDU response.

### Request

| Field Name | (Hex) |
|---|---|
| Slave Address | 0x03 |
| Function Code | 0x2B |
| MEI Type | 0x0E |
| Read Device ID code | 0x01 |
| Object ID | 0x00 |
| Check Sum | CRC |
| Check Sum | CRC |

### Response

| Field Name | (Hex) |
|---|---|
| Slave Address | 0x03 |
| Function Code | 0x11 |
| MEI Type | 0x0E |
| Read Device ID code | 0x01 |
| Conformity Level | 0x01 |
| More Follows | 0x00 |
| Next Object Id | 0x00 |
| Number Of Objects | 0x03 |
| Object Id | 0x00 |
| Object Length | 0x0E |
| Object Value | "Autometers Ltd" |
| Object Id | 0x01 |
| Object Length | 0x0C |
| Object Value | "IC990 xxx.yy" |
| Object Id | 0x02 |
| Object Length | 0x05 |
| Object Value | "V5.86" |
| Check Sum | CRC |
| Check Sum | CRC |

## Example 2

An Example of reading the Device Identification request "Extended device Identification" from Slave address 3 in RTU mode

In this example, all the information is sent in one PDU response.

### Request

| Field Name | (Hex) |
|---|---|
| **Slave Address** | 0x03 |
| **Function Code** | 0x2B |
| **MEI Type** | 0x0E |
| **Read Device ID code** | 0x04 |
| **Object ID** | 0x80 |
| **Check Sum** | CRC |
| **Check Sum** | CRC |

### Response

| Field Name | (Hex) |
|---|---|
| **Slave Address** | 0x03 |
| **Function Code** | 0x11 |
| **MEI Type** | 0x0E |
| **Read Device ID code** | 0x04 |
| **Conformity Level** | 0x83 |
| **More Follows** | 0x00 |
| **Next Object Id** | 0x80 |
| **Number Of Objects** | 0x01 |
| **Object Id** | 0x80 |
| **Object Length** | 0x07 |
| **Object Value** | "7891234" |
| **Check Sum** | CRC |
| **Check Sum** | CRC |

# Part V

# 5 MODBUS Exception Codes

### Introduction

When a master device sends a query to a slave device it expects a normal response. One of four possible events can occur from the master's query:

- Slave device receives the query without a communication error and can handle the query normally. It returns a normal response.
- Slave does not receive the query due to a communication error. No response is returned. The master program will eventually process a time-out condition for the query.
- Slave receives the query but detects a communication error (parity, LRC or CRC). No response is returned. The master program will eventually process a time-out condition for the query.
- Slave receives the query without a communication error but cannot handle it (i.e., request is to a non-existent coil or register). The slave will return with an exception response informing the master of the nature of the error (Illegal Data Address.)

The exception response message has two fields that differentiate it from a normal response:

**Function Code Field:**
In a normal response, the slave echoes the function code of the original query in the function code field of the response. All function codes have a most-significant bit (MSB) of 0 (their values are below 80 hex). In an exception response, the slave sets the MSB of the function code to 1. This makes the function code value in an exception response exactly 80 hex higher than the value would be for a normal response.
With the function code's MSB set, the master's application program can recognize the exception response and can examine the data field for the exception code.

**Data Field:**
In a normal response, the slave may return data or statistics in the data field. In an exception response, the slave returns an exception code in the data field. This defines the slave condition that caused the exception.

| Function Code | Name | Usage |
|---|---|---|
| 0x01 | Illegal Function | The message received is not an allowable action for the addressed device. |
| 0x02 | Illegal Data Address | The address referenced in the function-dependent data section of the message is not valid in the addressed device. |
| 0x03 | Illegal Data Value | The value referenced at the addressed device location is no within range. |
| 0x04 | Slave Device Failure | The addressed device has not been able to process a valid message due to a bad device state. |
| 0x06 | Slave Device Busy | The addressed device has ejected a message due to a busy state. Retry later. |

.

### Query

Example: Internal slave error reading 2 registers starting at address 1820h from slave at slave address 02.

02 03 18 20 00 02 CRC CRC

### Response

Example: Return MSB in Function Code byte set with Slave Device Failure (04) in the data field.

83 04 CRC CRC

# Part VI

# 6 MODBUS Register map

This appendix describes all parameters accessible by Function Codes 0x04 and 0x06. Parameters are grouped together according to the measurement been made, to simplify and speed up the reading of the data.

The availability of parameters and functions is depended on the device been accessed. Where a this is the case, the detailed description for each parameters or function code will state which products it is applicable for.

## 6.1 Register Map Overview

The following table describes the global register map for the Function Codes 0x04 (register read) and 0x06 (register write). Details on each address are given in the relevant sections.

**Conversion of address (hex) number to register (decimal) number.**

To convert the address number to the register number, convert the address from hexadecimal to decimal and add 30001.

Registers are addressed starting at zero: registers 1 – 16 are addressed as 0 – 15.

To convert the register number to the address number, subtract 30001 from the register and convert to hex.

| Start Address (hex) | End Address (hex) | Description | See Subsection |
|---|---|---|---|
| 0x0000 | 0x0006 | Transformer Ratios | Transformer Ratios |
| 0x0010 | 0x004F | Voltage Measurements | Voltage |
| 0x0050 | 0x0089 | Current Measurements | Current |
| 0x0090 | 0x0185 | Power Measurements | Power |
| | | Energy Measurements | Energy |
| 0x0200 | 0x02CF | Voltage Harmonics | Voltage Harmonics |
| 0x0300 | 0x040F | Current Harmonics | Current Harmonics |
| 0x0480 | 0x04B4 | Quality | Quality |
| 0x0500 | 0x05FF | Miscellaneous | Miscellaneous |
| 0x0600 | 0X0619 | Total Registers, a continuous block mapping all the total registers | Total Registers |
| 0x0900 | 0x0901 | Digital Input/Output Registers | Digital IO |
| 0x1000 | 0x100C | SOE and other events | SOE and other Events |
| 0x2000 | | HCC setup registers | HCC Setup |
| 0x9C40 | | Heat meters | Heat meters |
| 0xA000 | | Broadcast messages | boradcast messages |

## 6.2 Voltage Measurement

Values for the voltage measurements can be read from these registers.

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0010 | 30017 | 4 | Voltage L1 | R | Float | V |
| 0x0012 | 30019 | 4 | Voltage L2 | R | Float | V |
| 0x0014 | 30021 | 4 | Voltage L3 | R | Float | V |
| 0x0016 | 30023 | 4 | Voltage Max L1 | R | Float | V |
| 0x0018 | 30025 | 4 | Voltage Max L2 | R | Float | V |
| 0x001A | 30027 | 4 | Voltage Max L3 | R | Float | V |
| 0x001C | 30029 | 4 | Voltage Min L1 | R | Float | V |
| 0x001E | 30031 | 4 | Voltage Min L2 | R | Float | V |
| 0x0020 | 30033 | 4 | Voltage Min L3 | R | Float | V |
| 0x0022 | 30035 | 4 | Voltage Max Alarm Setting L1 | R/W | Float | V |
| 0x0024 | 30037 | 4 | Voltage Max Alarm Setting L2 | R/W | Float | V |
| 0x0026 | 30039 | 4 | Voltage Max Alarm Setting L3 | R/W | Float | V |
| 0x0028 | 30041 | 4 | Voltage Min Alarm Setting L1 | R/W | Float | V |
| 0x002A | 30043 | 4 | Voltage Min Alarm Setting L2 | R/W | Float | V |
| 0x002C | 30045 | 4 | Voltage Min Alarm Setting L3 | R/W | Float | V |
| | | | | | | |
| 0x0030 | 30049 | 4 | Voltage L1-L3 | R | Float | V |
| 0x0032 | 30051 | 4 | Voltage L3-L2 | R | Float | V |
| 0x0034 | 30053 | 4 | Voltage L2-L1 | R | Float | V |
| 0x0036 | 30055 | 4 | Voltage Max L1-L3 | R | Float | V |
| 0x0038 | 30057 | 4 | Voltage Max L3-L2 | R | Float | V |
| 0x003A | 30059 | 4 | Voltage Max L2-L1 | R | Float | V |
| 0x003C | 30061 | 4 | Voltage Min L1-L3 | R | Float | V |
| 0x003E | 30063 | 4 | Voltage Min L3-L2 | R | Float | V |
| 0x0040 | 30065 | 4 | Voltage Min L2-L1 | R | Float | V |
| 0x0042 | 30067 | 4 | Voltage Max Alarm Setting L1-L3 | R/W | Float | V |
| 0x0044 | 30069 | 4 | Voltage Max Alarm Setting L3-L2 | R/W | Float | V |
| 0x0046 | 30071 | 4 | Voltage Max Alarm Setting L2-L1 | R/W | Float | V |
| 0x0048 | 30073 | 4 | Voltage Min Alarm Setting L1-L3 | R/W | Float | V |
| 0x004A | 30075 | 4 | Voltage Min Alarm Setting L3-L2 | R/W | Float | V |
| 0x004C | 30077 | 4 | Voltage Min Alarm Setting L2-L1 | R/W | Float | V |
| | | | | | | |
| 0x004E | 30079 | 4 | Frequency | R | Float | Hz |

### 6.2.1 Voltage Harmonics

Values for the Voltage Harmonics

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0200 | 30513 | 4 | Voltage Crest Factor L1 | R | Float | |
| 0x0202 | 30515 | 4 | Voltage Crest Factor L2 | R | Float | |
| 0x0204 | 30517 | 4 | Voltage Crest Factor L3 | R | Float | |
| 0x0206 | 30519 | 4 | Voltage Total Harmonic Distortion L1 | R | Float | % |
| 0x0208 | 30521 | 4 | Voltage Total Harmonic Distortion L2 | R | Float | % |
| 0x020A | 30523 | 4 | Voltage Total Harmonic Distortion L3 | R | Float | % |
| | | | | | | |
| 0x0210 | 30529 | 4 | Voltage 1st Harmonic L1 | R | Float | % |
| 0x0212 | 30531 | 4 | Voltage 3rd Harmonic L1 | R | Float | % |
| 0x0214 | 30533 | 4 | Voltage 5th Harmonic L1 | R | Float | % |
| 0x0216 | 30535 | 4 | Voltage 7th Harmonic L1 | R | Float | % |
| 0x0218 | 30537 | 4 | Voltage 9th Harmonic L1 | R | Float | % |
| 0x021A | 30539 | 4 | Voltage 11th Harmonic L1 | R | Float | % |
| 0x021C | 30541 | 4 | Voltage 13th Harmonic L1 | R | Float | % |
| 0x021E | 30543 | 4 | Voltage 15th Harmonic L1 | R | Float | % |
| 0x0220 | 30545 | 4 | Voltage 17th Harmonic L1 | R | Float | % |
| 0x0222 | 30547 | 4 | Voltage 19th Harmonic L1 | R | Float | % |
| 0x0224 | 30549 | 4 | Voltage 21st Harmonic L1 | R | Float | % |
| 0x0226 | 30551 | 4 | Voltage 23rd Harmonic L1 | R | Float | % |
| 0x0228 | 30553 | 4 | Voltage 25th Harmonic L1 | R | Float | % |
| 0x022A | 30555 | 4 | Voltage 27th Harmonic L1 | R | Float | % |
| 0x022C | 30557 | 4 | Voltage 29th Harmonic L1 | R | Float | % |
| 0x022E | 30559 | 4 | Voltage 31st Harmonic L1 | R | Float | % |
| 0x0230 | 30561 | 4 | Voltage 33rd Harmonic L1 | R | Float | % |
| 0x0232 | 30563 | 4 | Voltage 35th Harmonic L1 | R | Float | % |
| 0x0234 | 30565 | 4 | Voltage 37th Harmonic L1 | R | Float | % |
| 0x0236 | 30567 | 4 | Voltage 39th Harmonic L1 | R | Float | % |
| 0x0238 | 30569 | 4 | Voltage 41st Harmonic L1 | R | Float | % |
| 0x023A | 30571 | 4 | Voltage 43rd Harmonic L1 | R | Float | % |
| 0x023C | 30573 | 4 | Voltage 45th Harmonic L1 | R | Float | % |
| 0x023E | 30575 | 4 | Voltage 47th Harmonic L1 | R | Float | % |
| 0x0240 | 30577 | 4 | Voltage 49th Harmonic L1 | R | Float | % |
| 0x0242 | 30579 | 4 | Voltage 51st Harmonic L1 | R | Float | % |
| 0x0244 | 30581 | 4 | Voltage 53rd Harmonic L1 | R | Float | % |
| 0x0246 | 30583 | 4 | Voltage 55th Harmonic L1 | R | Float | % |
| 0x0248 | 30585 | 4 | Voltage 57th Harmonic L1 | R | Float | % |
| 0x024A | 30587 | 4 | Voltage 59th Harmonic L1 | R | Float | % |
| 0x024C | 30589 | 4 | Voltage 61st Harmonic L1 | R | Float | % |
| 0x024E | 30591 | 4 | Voltage 63rd Harmonic L1 | R | Float | % |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0250 | 30593 | 4 | Voltage 1st Harmonic L2 | R | Float | % |
| 0x0252 | 30595 | 4 | Voltage 3rd Harmonic L2 | R | Float | % |
| 0x0254 | 30597 | 4 | Voltage 5th Harmonic L2 | R | Float | % |
| 0x0256 | 30599 | 4 | Voltage 7th Harmonic L2 | R | Float | % |
| 0x0258 | 30601 | 4 | Voltage 9th Harmonic L2 | R | Float | % |
| 0x025A | 30603 | 4 | Voltage 11th Harmonic L2 | R | Float | % |
| 0x025C | 30605 | 4 | Voltage 13th Harmonic L2 | R | Float | % |
| 0x025E | 30607 | 4 | Voltage 15th Harmonic L2 | R | Float | % |
| 0x0260 | 30609 | 4 | Voltage 17th Harmonic L2 | R | Float | % |
| 0x0262 | 30611 | 4 | Voltage 19th Harmonic L2 | R | Float | % |
| 0x0264 | 30613 | 4 | Voltage 21st Harmonic L2 | R | Float | % |
| 0x0266 | 30615 | 4 | Voltage 23rd Harmonic L2 | R | Float | % |
| 0x0268 | 30617 | 4 | Voltage 25th Harmonic L2 | R | Float | % |
| 0x026A | 30619 | 4 | Voltage 27th Harmonic L2 | R | Float | % |
| 0x026C | 30621 | 4 | Voltage 29th Harmonic L2 | R | Float | % |
| 0x026E | 30623 | 4 | Voltage 31st Harmonic L2 | R | Float | % |
| 0x0270 | 30625 | 4 | Voltage 33rd Harmonic L2 | R | Float | % |
| 0x0272 | 30627 | 4 | Voltage 35th Harmonic L2 | R | Float | % |
| 0x0274 | 30629 | 4 | Voltage 37th Harmonic L2 | R | Float | % |
| 0x0276 | 30631 | 4 | Voltage 39th Harmonic L2 | R | Float | % |
| 0x0278 | 30633 | 4 | Voltage 41st Harmonic L2 | R | Float | % |
| 0x027A | 30635 | 4 | Voltage 43rd Harmonic L2 | R | Float | % |
| 0x027C | 30637 | 4 | Voltage 45th Harmonic L2 | R | Float | % |
| 0x027E | 30639 | 4 | Voltage 47th Harmonic L2 | R | Float | % |
| 0x0280 | 30641 | 4 | Voltage 49th Harmonic L2 | R | Float | % |
| 0x0282 | 30643 | 4 | Voltage 51st Harmonic L2 | R | Float | % |
| 0x0284 | 30645 | 4 | Voltage 53rd Harmonic L2 | R | Float | % |
| 0x0286 | 30647 | 4 | Voltage 55th Harmonic L2 | R | Float | % |
| 0x0288 | 30649 | 4 | Voltage 57th Harmonic L2 | R | Float | % |
| 0x028A | 30651 | 4 | Voltage 59th Harmonic L2 | R | Float | % |
| 0x028C | 30653 | 4 | Voltage 61st Harmonic L2 | R | Float | % |
| 0x028E | 30655 | 4 | Voltage 63rd Harmonic L2 | R | Float | % |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0290 | 30657 | 4 | Voltage 1st Harmonic L3 | R | Float | % |
| 0x0292 | 30659 | 4 | Voltage 3rd Harmonic L3 | R | Float | % |
| 0x0294 | 30661 | 4 | Voltage 5th Harmonic L3 | R | Float | % |
| 0x0296 | 30663 | 4 | Voltage 7th Harmonic L3 | R | Float | % |
| 0x0298 | 30665 | 4 | Voltage 9th Harmonic L3 | R | Float | % |
| 0x029A | 30667 | 4 | Voltage 11th Harmonic L3 | R | Float | % |
| 0x029C | 30669 | 4 | Voltage 13th Harmonic L3 | R | Float | % |
| 0x029E | 30671 | 4 | Voltage 15th Harmonic L3 | R | Float | % |
| 0x02A0 | 30673 | 4 | Voltage 17th Harmonic L3 | R | Float | % |
| 0x02A2 | 30675 | 4 | Voltage 19th Harmonic L3 | R | Float | % |
| 0x02A4 | 30677 | 4 | Voltage 21st Harmonic L3 | R | Float | % |
| 0x02A6 | 30679 | 4 | Voltage 23rd Harmonic L3 | R | Float | % |
| 0x02A8 | 30681 | 4 | Voltage 25th Harmonic L3 | R | Float | % |
| 0x02AA | 30683 | 4 | Voltage 27th Harmonic L3 | R | Float | % |
| 0x02AC | 30685 | 4 | Voltage 29th Harmonic L3 | R | Float | % |
| 0x02AE | 30687 | 4 | Voltage 31st Harmonic L3 | R | Float | % |
| 0x02B0 | 30689 | 4 | Voltage 33rd Harmonic L3 | R | Float | % |
| 0x02B2 | 30691 | 4 | Voltage 35th Harmonic L3 | R | Float | % |
| 0x02B4 | 30693 | 4 | Voltage 37th Harmonic L3 | R | Float | % |
| 0x02B6 | 30695 | 4 | Voltage 39th Harmonic L3 | R | Float | % |
| 0x02B8 | 30697 | 4 | Voltage 41st Harmonic L3 | R | Float | % |
| 0x02BA | 30699 | 4 | Voltage 43rd Harmonic L3 | R | Float | % |
| 0x02BC | 30701 | 4 | Voltage 45th Harmonic L3 | R | Float | % |
| 0x02BE | 30703 | 4 | Voltage 47th Harmonic L3 | R | Float | % |
| 0x02C0 | 30705 | 4 | Voltage 49th Harmonic L3 | R | Float | % |
| 0x02C2 | 30707 | 4 | Voltage 51st Harmonic L3 | R | Float | % |
| 0x02C4 | 30709 | 4 | Voltage 53rd Harmonic L3 | R | Float | % |
| 0x02C6 | 30711 | 4 | Voltage 55th Harmonic L3 | R | Float | % |
| 0x02C8 | 30713 | 4 | Voltage 57th Harmonic L3 | R | Float | % |
| 0x02CA | 30715 | 4 | Voltage 59th Harmonic L3 | R | Float | % |
| 0x02CC | 30717 | 4 | Voltage 61st Harmonic L3 | R | Float | % |
| 0x02CE | 30719 | 4 | Voltage 63rd Harmonic L3 | R | Float | % |

## 6.3 Current Measurement

Values for the current measurements can be read from these registers.

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Unit |
|---|---|---|---|---|---|---|
| 0x0050 | 30081 | 4 | Current L1 | R | Float | A |
| 0x0052 | 30083 | 4 | Current L2 | R | Float | A |
| 0x0054 | 30085 | 4 | Current L3 | R | Float | A |
| 0x0056 | 30087 | 4 | Current Neutral | R | Float | A |
| 0x0058 | 30089 | 4 | Current Total | R | Float | A |
| 0x005A | 30091 | 4 | Current Max L1 | R | Float | A |
| 0x005C | 30093 | 4 | Current Max L2 | R | Float | A |
| 0x005E | 30095 | 4 | Current Max L3 | R | Float | A |
| 0x0060 | 30097 | 4 | Current Max Neutral | R | Float | A |
| 0x0062 | 30099 | 4 | Current Max Total | R | Float | A |
| 0x0064 | 30101 | 4 | Current Min L1 | R | Float | A |
| 0x0066 | 30103 | 4 | Current Min L2 | R | Float | A |
| 0x0068 | 30105 | 4 | Current Min L3 | R | Float | A |
| 0x006A | 30107 | 4 | Current Min Neutral | R | Float | A |
| 0x006C | 30109 | 4 | Current Min Total | R | Float | A |
| 0x006E | 30111 | 4 | Current Max Alarm Setting L1 | R/W | Float | A |
| 0x0070 | 30113 | 4 | Current Max Alarm Setting L2 | R/W | Float | A |
| 0x0072 | 30115 | 4 | Current Max Alarm Setting L3 | R/W | Float | A |
| 0x0074 | 30117 | 4 | Current Max Alarm Setting Neutral | R/W | Float | A |
| 0x0076 | 30119 | 4 | Current Max Alarm setting Total | R/W | Float | A |
| 0x0078 | 30121 | 4 | Current Min Alarm Setting L1 | R/W | Float | A |
| 0x007A | 30123 | 4 | Current Min Alarm Setting L2 | R/W | Float | A |
| 0x007C | 30125 | 4 | Current Min Alarm Setting L3 | R/W | Float | A |
| 0x007E | 30127 | 4 | Min Alarm Setting Neutral | R/W | Float | A |
| 0x0080 | 30129 | 4 | Min Alarm Setting Total | R/W | Float | A |
| 0x0082 | 30131 | 2 | Period for maximum demand amps | R/W | 16 bits | mins |
| 0x0084 | 30133 | 4 | Present maximum demand for amps | R | Float | A |
| 0x0086 | 30135 | 2 | Minutes into maximum demand period | R | 16 bits | mins |
| 0x0088 | 30137 | 2 | Seconds into maximum demand period | R | 16 bits | secs |

### 6.3.1 Current Harmonics

Values for the Current Harmonics

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0300 | 30769 | 4 | Current Crest Factor L1 | R | Float | |
| 0x0302 | 30771 | 4 | Current Crest Factor L2 | R | Float | |
| 0x0304 | 30773 | 4 | Current Crest Factor L3 | R | Float | |
| 0x0306 | 30775 | 4 | Current Crest Factor Neutral | R | Float | |
| 0x0308 | 30777 | 4 | Current Total Harmonic Distortion L1 | R | Float | % |
| 0x030A | 30779 | 4 | Current Total Harmonic Distortion L2 | R | Float | % |
| 0x030C | 30781 | 4 | Current Total Harmonic Distortion L3 | R | Float | % |
| 0x030E | 30783 | 4 | Current Total Harmonic Distortion Neutral | R | Float | % |
| | | | | | | |
| 0x0310 | 30785 | 4 | Current 1st Harmonic L1 | R | Float | % |
| 0x0312 | 30787 | 4 | Current 3rd Harmonic L1 | R | Float | % |
| 0x0314 | 30789 | 4 | Current 5th Harmonic L1 | R | Float | % |
| 0x0316 | 30791 | 4 | Current 7th Harmonic L1 | R | Float | % |
| 0x0318 | 30793 | 4 | Current 9th Harmonic L1 | R | Float | % |
| 0x031A | 30795 | 4 | Current 11th Harmonic L1 | R | Float | % |
| 0x031B | 30797 | 4 | Current 13th Harmonic L1 | R | Float | % |
| 0x031E | 30799 | 4 | Current 15th Harmonic L1 | R | Float | % |
| 0x0320 | 30801 | 4 | Current 17th Harmonic L1 | R | Float | % |
| 0x0322 | 30803 | 4 | Current 19th Harmonic L1 | R | Float | % |
| 0x0324 | 30805 | 4 | Current 21st Harmonic L1 | R | Float | % |
| 0x0326 | 30807 | 4 | Current 23rd Harmonic L1 | R | Float | % |
| 0x0328 | 30809 | 4 | Current 25th Harmonic L1 | R | Float | % |
| 0x032A | 30811 | 4 | Current 27th Harmonic L1 | R | Float | % |
| 0x032C | 30813 | 4 | Current 29th Harmonic L1 | R | Float | % |
| 0x032E | 30815 | 4 | Current 31st Harmonic L1 | R | Float | % |
| 0x0330 | 30817 | 4 | Current 33rd Harmonic L1 | R | Float | % |
| 0x0332 | 30819 | 4 | Current 35th Harmonic L1 | R | Float | % |
| 0x0334 | 30821 | 4 | Current 37th Harmonic L1 | R | Float | % |
| 0x0336 | 30823 | 4 | Current 39th Harmonic L1 | R | Float | % |
| 0x0338 | 30825 | 4 | Current 41st Harmonic L1 | R | Float | % |
| 0x033A | 30827 | 4 | Current 43rd Harmonic L1 | R | Float | % |
| 0x033C | 30829 | 4 | Current 45th Harmonic L1 | R | Float | % |
| 0x033E | 30831 | 4 | Current 47th Harmonic L1 | R | Float | % |
| 0x0340 | 30833 | 4 | Current 49th Harmonic L1 | R | Float | % |
| 0x0342 | 30835 | 4 | Current 51st Harmonic L1 | R | Float | % |
| 0x0344 | 30837 | 4 | Current 53rd Harmonic L1 | R | Float | % |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0346 | 30839 | 4 | Current 55th Harmonic L1 | R | Float | % |
| 0x0348 | 30841 | 4 | Current 57th Harmonic L1 | R | Float | % |
| 0x034A | 30843 | 4 | Current 59th Harmonic L1 | R | Float | % |
| 0x034C | 30845 | 4 | Current 61st Harmonic L1 | R | Float | % |
| 0x034E | 30847 | 4 | Current 63rd Harmonic L1 | R | Float | % |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0350 | 30849 | 4 | Current 1st Harmonic L2 | R | Float | % |
| 0x0352 | 30851 | 4 | Current 3rd Harmonic L2 | R | Float | % |
| 0x0354 | 30853 | 4 | Current 5th Harmonic L2 | R | Float | % |
| 0x0356 | 30855 | 4 | Current 7th Harmonic L2 | R | Float | % |
| 0x0358 | 30857 | 4 | Current 9th Harmonic L2 | R | Float | % |
| 0x035A | 30859 | 4 | Current 11th Harmonic L2 | R | Float | % |
| 0x035C | 30861 | 4 | Current 13th Harmonic L2 | R | Float | % |
| 0x035E | 30863 | 4 | Current 15th Harmonic L2 | R | Float | % |
| 0x0360 | 30865 | 4 | Current 17th Harmonic L2 | R | Float | % |
| 0x0362 | 30867 | 4 | Current 19th Harmonic L2 | R | Float | % |
| 0x0364 | 30869 | 4 | Current 21st Harmonic L2 | R | Float | % |
| 0x0366 | 30871 | 4 | Current 23rd Harmonic L2 | R | Float | % |
| 0x0368 | 30873 | 4 | Current 25th Harmonic L2 | R | Float | % |
| 0x036A | 30875 | 4 | Current 27th Harmonic L2 | R | Float | % |
| 0x036C | 30877 | 4 | Current 29th Harmonic L2 | R | Float | % |
| 0x036E | 30879 | 4 | Current 31st Harmonic L2 | R | Float | % |
| 0x0370 | 30881 | 4 | Current 33rd Harmonic L2 | R | Float | % |
| 0x0372 | 30883 | 4 | Current 35th Harmonic L2 | R | Float | % |
| 0x0374 | 30885 | 4 | Current 37th Harmonic L2 | R | Float | % |
| 0x0376 | 30887 | 4 | Current 39th Harmonic L2 | R | Float | % |
| 0x0378 | 30889 | 4 | Current 41st Harmonic L2 | R | Float | % |
| 0x037A | 30891 | 4 | Current 43rd Harmonic L2 | R | Float | % |
| 0x037C | 30893 | 4 | Current 45th Harmonic L2 | R | Float | % |
| 0x037E | 30895 | 4 | Current 47th Harmonic L2 | R | Float | % |
| 0x0380 | 30897 | 4 | Current 49th Harmonic L2 | R | Float | % |
| 0x0382 | 30899 | 4 | Current 51st Harmonic L2 | R | Float | % |
| 0x0384 | 30901 | 4 | Current 53rd Harmonic L2 | R | Float | % |
| 0x0386 | 30903 | 4 | Current 55th Harmonic L2 | R | Float | % |
| 0x0388 | 30905 | 4 | Current 57th Harmonic L2 | R | Float | % |
| 0x038A | 30907 | 4 | Current 59th Harmonic L2 | R | Float | % |
| 0x038C | 30909 | 4 | Current 61st Harmonic L2 | R | Float | % |
| 0x038E | 30911 | 4 | Current 63rd Harmonic L2 | R | Float | % |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0390 | 30913 | 4 | Current 1st Harmonic L3 | R | Float | % |
| 0x0392 | 30915 | 4 | Current 3rd Harmonic L3 | R | Float | % |
| 0x0394 | 30917 | 4 | Current 5th Harmonic L3 | R | Float | % |
| 0x0396 | 30919 | 4 | Current 7th Harmonic L3 | R | Float | % |
| 0x0398 | 30921 | 4 | Current 9th Harmonic L3 | R | Float | % |
| 0x039A | 30923 | 4 | Current 11th Harmonic L3 | R | Float | % |
| 0x039C | 30925 | 4 | Current 13th Harmonic L3 | R | Float | % |
| 0x039E | 30927 | 4 | Current 15th Harmonic L3 | R | Float | % |
| 0x03A0 | 30929 | 4 | Current 17th Harmonic L3 | R | Float | % |
| 0x03A2 | 30931 | 4 | Current 19th Harmonic L3 | R | Float | % |
| 0x03A4 | 30933 | 4 | Current 21st Harmonic L3 | R | Float | % |
| 0x03A6 | 30935 | 4 | Current 23rd Harmonic L3 | R | Float | % |
| 0x03A8 | 30937 | 4 | Current 25th Harmonic L3 | R | Float | % |
| 0x03AA | 30939 | 4 | Current 27th Harmonic L3 | R | Float | % |
| 0x03AC | 30941 | 4 | Current 29th Harmonic L3 | R | Float | % |
| 0x03AE | 30943 | 4 | Current 31st Harmonic L3 | R | Float | % |
| 0x03B0 | 30945 | 4 | Current 33rd Harmonic L3 | R | Float | % |
| 0x03B2 | 30947 | 4 | Current 35th Harmonic L3 | R | Float | % |
| 0x03B4 | 30949 | 4 | Current 37th Harmonic L3 | R | Float | % |
| 0x03B6 | 30951 | 4 | Current 39th Harmonic L3 | R | Float | % |
| 0x03B8 | 30953 | 4 | Current 41st Harmonic L3 | R | Float | % |
| 0x03BA | 30955 | 4 | Current 43rd Harmonic L3 | R | Float | % |
| 0x03BC | 30957 | 4 | Current 45th Harmonic L3 | R | Float | % |
| 0x03BE | 30959 | 4 | Current 47th Harmonic L3 | R | Float | % |
| 0x03C0 | 30961 | 4 | Current 49th Harmonic L3 | R | Float | % |
| 0x03C2 | 30963 | 4 | Current 51st Harmonic L3 | R | Float | % |
| 0x03C4 | 30965 | 4 | Current 53rd Harmonic L3 | R | Float | % |
| 0x03C6 | 30967 | 4 | Voltage 55th Harmonic L3 | R | Float | % |
| 0x03C8 | 30969 | 4 | Voltage 57th Harmonic L3 | R | Float | % |
| 0x03CA | 30971 | 4 | Current 59th Harmonic L3 | R | Float | % |
| 0x03CC | 30973 | 4 | Current 61st Harmonic L3 | R | Float | % |
| 0x03CE | 30975 | 4 | Current 63rd Harmonic L3 | R | Float | % |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x03D0 | 30977 | 4 | Current 1st Harmonic Neutral | R | Float | % |
| 0x03D2 | 30979 | 4 | Current 3rd Harmonic Neutral | R | Float | % |
| 0x03D4 | 30981 | 4 | Current 5th Harmonic Neutral | R | Float | % |
| 0x03D6 | 30983 | 4 | Current 7th Harmonic Neutral | R | Float | % |
| 0x03D8 | 30985 | 4 | Current 9th Harmonic Neutral | R | Float | % |
| 0x03DA | 30987 | 4 | Current 11th Harmonic Neutral | R | Float | % |
| 0x03DC | 30989 | 4 | Current 13th Harmonic Neutral | R | Float | % |
| 0x03DE | 30991 | 4 | Current 15th Harmonic Neutral | R | Float | % |
| 0x03E0 | 30993 | 4 | Current 17th Harmonic Neutral | R | Float | % |
| 0x03E2 | 30995 | 4 | Current 19th Harmonic Neutral | R | Float | % |
| 0x03E4 | 30997 | 4 | Current 21st Harmonic Neutral | R | Float | % |
| 0x03E6 | 30999 | 4 | Current 23rd Harmonic Neutral | R | Float | % |
| 0x03E8 | 31001 | 4 | Current 25th Harmonic Neutral | R | Float | % |
| 0x03EA | 31003 | 4 | Current 27th Harmonic Neutral | R | Float | % |
| 0x03EC | 31005 | 4 | Current 29th Harmonic Neutral | R | Float | % |
| 0x03EE | 31007 | 4 | Current 31st Harmonic Neutral | R | Float | % |
| 0x03F0 | 31009 | 4 | Current 33rd Harmonic Neutral | R | Float | % |
| 0x03F2 | 31011 | 4 | Current 35th Harmonic Neutral | R | Float | % |
| 0x03F4 | 31013 | 4 | Current 37th Harmonic Neutral | R | Float | % |
| 0x03F6 | 31015 | 4 | Current 39th Harmonic Neutral | R | Float | % |
| 0x03F8 | 31017 | 4 | Current 41st Harmonic Neutral | R | Float | % |
| 0x03FA | 31019 | 4 | Current 43rd Harmonic Neutral | R | Float | % |
| 0x03FC | 31021 | 4 | Current 45th Harmonic Neutral | R | Float | % |
| 0x03FE | 31023 | 4 | Current 47th Harmonic Neutral | R | Float | % |
| 0x0400 | 31025 | 4 | Current 49th Harmonic Neutral | R | Float | % |
| 0x0402 | 31027 | 4 | Current 51st Harmonic Neutral | R | Float | % |
| 0x0404 | 31029 | 4 | Current 53rd Harmonic Neutral | R | Float | % |
| 0x0406 | 31031 | 4 | Voltage 55th Harmonic Neutral | R | Float | % |
| 0x0408 | 31033 | 4 | Voltage 57th Harmonic Neutral | R | Float | % |
| 0x040A | 31035 | 4 | Current 59th Harmonic Neutral | R | Float | % |
| 0x040C | 31037 | 4 | Current 61st Harmonic Neutral | R | Float | % |
| 0x040E | 31039 | 4 | Current 63rd Harmonic Neutral | R | Float | % |

## 6.4 Power Measurement

Values for the Power and Energy measurements can be read from these registers.

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0090 | 30145 | 4 | Power L1 | R | Float | KW |
| 0x0092 | 30147 | 4 | Power L2 | R | Float | KW |
| 0x0094 | 30149 | 4 | Power L3 | R | Float | KW |
| 0x0096 | 30151 | 4 | Power Total | R | Float | KW |
| 0x0098 | 30153 | 4 | Power Max L1 | R | Float | KW |
| 0x009A | 30155 | 4 | Power Max L2 | R | Float | KW |
| 0x009C | 30157 | 4 | Power Max L3 | R | Float | KW |
| 0x009E | 30159 | 4 | Power Max Total | R | Float | KW |
| 0x00A0 | 30161 | 4 | Power Min L1 | R | Float | KW |
| 0x00A2 | 30163 | 4 | Power Min L2 | R | Float | KW |
| 0x00A4 | 30165 | 4 | Power Min L3 | R | Float | KW |
| 0x00A6 | 30167 | 4 | Power Min Total | R | Float | KW |
| 0x00A8 | 30169 | 4 | Power Max Alarm Settings L1 | R/W | Float | KW |
| 0x00AA | 30171 | 4 | Power Max Alarm Settings L2 | R/W | Float | KW |
| 0x00AC | 30173 | 4 | Power Max Alarm Settings L3 | R/W | Float | KW |
| 0x00AE | 30175 | 4 | Power Max Alarm Settings Total | R/W | Float | KW |
| 0x00B0 | 30177 | 4 | Power Min Alarm Settings L1 | R/W | Float | KW |
| 0x00B2 | 30179 | 4 | Power Min Alarm Settings L2 | R/W | Float | KW |
| 0x00B4 | 30181 | 4 | Power Min Alarm Settings L3 | R/W | Float | KW |
| 0x00B6 | 30183 | 4 | Power Min Alarm Settings Total | R/W | Float | KW |
| 0x00B8 | 30185 | 2 | Period for maximum demand total power | R/W | 16 bits | min |
| 0x00BA | 30187 | 4 | Present demand for total power | R | Float | KW |
| 0x00BC | 30189 | 4 | Maximum value for maximum demand total power | R | Float | KW |
| 0x00BE | 30191 | 2 | Minutes into maximum demand period for total power | R | 16 bits | mins |
| 0x00C0 | 30193 | 2 | Seconds into maximum demand period for total power | R | 16 bits | secs |
| 0x00C2 | 30195 | 4 | Time for last Maximum - Maximum demand | R | 32 bits | EPOCH |
| 0x00C4 | 30197 | 2 | Total import active demand Time Year | R | 16bits | YYYY |
| 0x00C5 | 30198 | 2 | Total import active demand Time Month | R | 16bits | 1-12 |
| 0x00C6 | 30199 | 2 | Total import active demand Time Day | R | 16bits | 1-31 |
| 0x00C7 | 30200 | 2 | Total import active demand Time Hour | R | 16bits | 00-23 |
| 0x00C8 | 30201 | 2 | Total import active demand Time Minutes | R | 16bits | 00-59 |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x00D0 | 30209 | 4 | Apparent Power L1 | R | Float | KVA |
| 0x00D2 | 30211 | 4 | Apparent Power L2 | R | Float | KVA |
| 0x00D4 | 30213 | 4 | Apparent Power L3 | R | Float | KVA |
| 0x00D6 | 30215 | 4 | Apparent Power Total | R | Float | KVA |
| 0x00D8 | 30217 | 4 | Apparent Power Max L1 | R | Float | KVA |
| 0x00DA | 30219 | 4 | Apparent Power Max L2 | R | Float | KVA |
| 0x00DC | 30221 | 4 | Apparent Power Max L3 | R | Float | KVA |
| 0x00DE | 30223 | 4 | Apparent Power Max Total | R | Float | KVA |
| 0x00E0 | 30225 | 4 | Apparent Power Min L1 | R | Float | KVA |
| 0x00E2 | 30227 | 4 | Apparent Power Min L2 | R | Float | KVA |
| 0x00E4 | 30229 | 4 | Apparent Power Min L3 | R | Float | KVA |
| 0x00E6 | 30231 | 4 | Apparent Power Min Total | R | Float | KVA |
| 0x00E8 | 30233 | 4 | Apparent Power Max Alarm Settings L1 | R/W | Float | KVA |
| 0x00EA | 30235 | 4 | Apparent Power Max Alarm Settings L2 | R/W | Float | KVA |
| 0x00EC | 30237 | 4 | Apparent Power Max Alarm Settings L3 | R/W | Float | KVA |
| 0x00EE | 30239 | 4 | Apparent Power Max Alarm Settings Total | R/W | Float | KVA |
| 0x00F0 | 30241 | 4 | Apparent Power Min Alarm Settings L1 | R/W | Float | KVA |
| 0x00F2 | 30243 | 4 | Apparent Power Min Alarm Settings L2 | R/W | Float | KVA |
| 0x00F4 | 30245 | 4 | Apparent Power Min Alarm Settings L3 | R/W | Float | KVA |
| 0x00F6 | 30247 | 4 | Apparent Power Min Alarm Settings Total | R/W | Float | KVA |
| 0x00F8 | 30249 | 2 | Period for maximum demand total apparent power | R/W | 16 bits | min |
| 0x00FA | 30251 | 4 | Present demand for total apparent power | R | Float | KVA |
| 0x00FC | 30253 | 4 | Maximum value for maximum demand total apparent power | R | Float | KVA |
| 0x00FE | 30255 | 2 | Minutes into maximum demand period for total apparent power | R | 16 bits | mins |
| 0x0100 | 30257 | 2 | Seconds into maximum demand period for total apparent power | R | 16 bits | secs |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0110 | 30273 | 4 | Reactive Power L1 | R | Float | KVAR |
| 0x0112 | 30275 | 4 | Reactive Power L2 | R | Float | KVAR |
| 0x0114 | 30277 | 4 | Reactive Power L3 | R | Float | KVAR |
| 0x0116 | 30279 | 4 | Reactive Power Total | R | Float | KVAR |
| 0x0118 | 30281 | 4 | Reactive Power Max L1 | R | Float | KVAR |
| 0x011A | 30283 | 4 | Reactive Power Max L2 | R | Float | KVAR |
| 0x011C | 30285 | 4 | Reactive Power Max L3 | R | Float | KVAR |
| 0x011E | 30287 | 4 | Reactive Power Max Total | R | Float | KVAR |
| 0x0120 | 30289 | 4 | Reactive Power Min L1 | R | Float | KVAR |
| 0x0122 | 30291 | 4 | Reactive Power Min L2 | R | Float | KVAR |
| 0x0124 | 30293 | 4 | Reactive Power Min L3 | R | Float | KVAR |
| 0x0126 | 30295 | 4 | Reactive Power Min Total | R | Float | KVAR |
| 0x0128 | 30297 | 4 | Reactive Power Max Alarm Settings L1 | R/W | Float | KVAR |
| 0x012A | 30299 | 4 | Reactive Power Max Alarm Settings L2 | R/W | Float | KVAR |
| 0x012C | 30301 | 4 | Reactive Power Max Alarm Settings L3 | R/W | Float | KVAR |
| 0x012E | 30303 | 4 | Reactive Power Max Alarm Settings Total | R/W | Float | KVAR |
| 0x0130 | 30305 | 4 | Reactive Power Min Alarm Settings L1 | R/W | Float | KVAR |
| 0x0132 | 30307 | 4 | Reactive Power Min Alarm Settings L2 | R/W | Float | KVAR |
| 0x0134 | 30309 | 4 | Reactive Power Min Alarm Settings L3 | R/W | Float | KVAR |
| 0x0136 | 30311 | 4 | Reactive Power Min Alarm Settings Total | R/W | Float | KVAR |
| 0x0138 | 30313 | 2 | Period for maximum demand total reactive power | R/W | 16 bits | min |
| 0x013A | 30315 | 4 | Present demand for total reactive power | R | Float | KVAR |
| 0x013C | 30317 | 4 | Maximum value for maximum demand total reactive power | R | Float | KVAR |
| 0x013E | 30319 | 2 | Minutes into maximum demand period for total reactive power | R | 16 bits | mins |
| 0x0140 | 30321 | 2 | Seconds into maximum demand period for total reactive power | R | 16 bits | secs |
| 0x0141 | | 4 | | | | |
| 0x0142 | 30324 | 2 | Total import reactive demand Time Year | R | 16bits | YYYY |
| 0x0143 | 30325 | 2 | Total import reactive demand Time Month | R | 16bits | 1-12 |
| 0x0144 | 30326 | 2 | Total import reactive demand Time Day | R | 16bits | 1-31 |
| 0x0145 | 30327 | 2 | Total import reactive demand Time Hour | R | 16bits | 00-23 |
| 0x0146 | 30328 | 2 | Total import reactive demand Time Minutes | R | 16bits | 00-59 |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0150 | 30337 | 4 | Power Factor L1 | R | Float | |
| 0x0152 | 30339 | 4 | Power Factor L2 | R | Float | |
| 0x0154 | 30341 | 4 | Power Factor L3 | R | Float | |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0156 | 30343 | 4 | Power Factor Total | R | Float | |
| 0x0158 | 30345 | 2 | Power Direction | R | 16bits | 1 = import<br>2 = export |

## 6.5 Energy Measurements

Values for the Energy measurements can be read from these registers.

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0160 | 30353 | 4 | Import Energy | R | Float | KWHr |
| 0x0162 | 30355 | 4 | Import Reactive Energy | R | Float | KVARHr |
| 0x0164 | 30357 | 4 | Import Apparent Energy | R | Float | KVAHr |
| 0x0166 | 30359 | 4 | Export Energy | R | Float | KWHr |
| 0x0168 | 30361 | 4 | Export Reactive Energy | R | Float | KVARHr |
| 0x016A | 30363 | 4 | Amps Energy | R | Float | Ahr |

### 6.5.1 Energy Rate Meters

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x07D0 | 32001 | 4 | Import Energy Rate 1 | R | Float | Kwh |
| 0x07D2 | 32003 | 4 | Import Energy Rate 2 | R | Float | Kwh |
| 0x07D4 | 32005 | 4 | Import Energy Rate 3 | R | Float | Kwh |
| 0x07D6 | 32007 | 4 | Import Energy Rate 4 | R | Float | Kwh |
| 0x07D8 | 32009 | 4 | Channel 1 Total Import Energy | R | Float | Kwh |
| 0x07DA | 32011 | 4 | Channel 2 Total Import Energy | R | Float | Kwh |
| 0x07DC | 32013 | 4 | Channel 3 Total Import Energy | R | Float | Kwh |
| 0x07DE | 32015 | 4 | Channel 4 Total Import Energy | R | Float | Kwh |
| 0x07E0 | 32017 | 4 | Channel 5 Total Import Energy | R | Float | Kwh |
| 0x07E2 | 32019 | 4 | Channel 6 Total Import Energy | R | Float | Kwh |
| 0x07E4 | 32021 | 4 | Channel 7 Total Import Energy | R | Float | Kwh |
| 0x07E6 | 32023 | 4 | Channel 8 Total Import Energy | R | Float | Kwh |
| 0x07E8 | 32025 | 4 | Total Import Energy All channels and rates | R | Float | Kwh |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x07EA | 32027 | 4 | HCC Channel 1 Energy Rate 1 | R | Float | Kwh |
| 0x07EC | 32029 | 4 | HCC Channel 2 Energy Rate 1 | R | Float | Kwh |
| 0x07EE | 32031 | 4 | HCC Channel 3 Energy Rate 1 | R | Float | Kwh |
| 0x07F0 | 32033 | 4 | HCC Channel 4 Energy Rate 1 | R | Float | Kwh |
| 0x07F2 | 32035 | 4 | HCC Channel 5 Energy Rate 1 | R | Float | Kwh |
| 0x07F4 | 32037 | 4 | HCC Channel 6 Energy Rate 1 | R | Float | Kwh |
| 0x07F6 | 32039 | 4 | HCC Channel 7 Energy Rate 1 | R | Float | Kwh |
| 0x07F8 | 32041 | 4 | HCC Channel 8 Energy Rate 1 | R | Float | Kwh |
| 0x07FA | 32043 | 4 | HCC Channel 9 Energy Rate 1 | R | Float | Kwh |
| 0x07FC | 32045 | 4 | HCC Channel 10 Energy Rate 1 | R | Float | Kwh |
| 0x07FE | 32047 | 4 | HCC Channel 11 Energy Rate 1 | R | Float | Kwh |
| 0x0800 | 32049 | 4 | HCC Channel 12 Energy Rate 1 | R | Float | Kwh |
| 0x0802 | 32051 | 4 | HCC Channel 13 Energy Rate 1 | R | Float | Kwh |
| 0x0804 | 32053 | 4 | HCC Channel 14 Energy Rate 1 | R | Float | Kwh |
| 0x0806 | 32055 | 4 | HCC Channel 15 Energy Rate 1 | R | Float | Kwh |
| 0x0808 | 32057 | 4 | HCC Channel 16 Energy Rate 1 | R | Float | Kwh |
| 0x080A | 32059 | 4 | HCC Channel 1 Energy Rate 2 | R | Float | Kwh |
| 0x080C | 32061 | 4 | HCC Channel 2 Energy Rate 2 | R | Float | Kwh |
| 0x080E | 32063 | 4 | HCC Channel 3 Energy Rate 2 | R | Float | Kwh |
| 0x0810 | 32065 | 4 | HCC Channel 4 Energy Rate 2 | R | Float | Kwh |
| 0x0812 | 32067 | 4 | HCC Channel 5 Energy Rate 2 | R | Float | Kwh |
| 0x0814 | 32069 | 4 | HCC Channel 6 Energy Rate 2 | R | Float | Kwh |
| 0x0816 | 32071 | 4 | HCC Channel 7 Energy Rate 2 | R | Float | Kwh |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0818 | 32073 | 4 | HCC Channel 8 Energy Rate 2 | R | Float | Kwh |
| 0x081A | 32075 | 4 | HCC Channel 9 Energy Rate 2 | R | Float | Kwh |
| 0x081C | 32077 | 4 | HCC Channel 10 Energy Rate 2 | R | Float | Kwh |
| 0x081E | 32079 | 4 | HCC Channel 11 Energy Rate 2 | R | Float | Kwh |
| 0x0820 | 32081 | 4 | HCC Channel 12 Energy Rate 2 | R | Float | Kwh |
| 0x0822 | 32083 | 4 | HCC Channel 13 Energy Rate 2 | R | Float | Kwh |
| 0x0824 | 32085 | 4 | HCC Channel 14 Energy Rate 2 | R | Float | Kwh |
| 0x0826 | 32087 | 4 | HCC Channel 15 Energy Rate 2 | R | Float | Kwh |
| 0x0828 | 32089 | 4 | HCC Channel 16 Energy Rate 2 | R | Float | Kwh |
| 0x082A | 32091 | 4 | HCC Channel 1 Energy Rate 3 | R | Float | Kwh |
| 0x082C | 32093 | 4 | HCC Channel 2 Energy Rate 3 | R | Float | Kwh |
| 0x082E | 32095 | 4 | HCC Channel 3 Energy Rate 3 | R | Float | Kwh |
| 0x0830 | 32097 | 4 | HCC Channel 4 Energy Rate 3 | R | Float | Kwh |
| 0x0832 | 32099 | 4 | HCC Channel 5 Energy Rate 3 | R | Float | Kwh |
| 0x0834 | 32101 | 4 | HCC Channel 6 Energy Rate 3 | R | Float | Kwh |
| 0x0836 | 32103 | 4 | HCC Channel 7 Energy Rate 3 | R | Float | Kwh |
| 0x0838 | 32105 | 4 | HCC Channel 8 Energy Rate 3 | R | Float | Kwh |
| 0x083A | 32107 | 4 | HCC Channel 9 Energy Rate 3 | R | Float | Kwh |
| 0x083C | 32109 | 4 | HCC Channel 10 Energy Rate 3 | R | Float | Kwh |
| 0x083E | 32111 | 4 | HCC Channel 11 Energy Rate 3 | R | Float | Kwh |
| 0x0840 | 32113 | 4 | HCC Channel 12 Energy Rate 3 | R | Float | Kwh |
| 0x0842 | 32115 | 4 | HCC Channel 13 Energy Rate 3 | R | Float | Kwh |
| 0x0844 | 32117 | 4 | HCC Channel 14 Energy Rate 3 | R | Float | Kwh |
| 0x0846 | 32119 | 4 | HCC Channel 15 Energy Rate 3 | R | Float | Kwh |
| 0x0848 | 32121 | 4 | HCC Channel 16 Energy Rate 3 | R | Float | Kwh |
| 0x084A | 32123 | 4 | HCC Channel 1 Energy Rate 4 | R | Float | Kwh |
| 0x084C | 32125 | 4 | HCC Channel 2 Energy Rate 4 | R | Float | Kwh |
| 0x084E | 32127 | 4 | HCC Channel 3 Energy Rate 4 | R | Float | Kwh |
| 0x0850 | 32129 | 4 | HCC Channel 4 Energy Rate 4 | R | Float | Kwh |
| 0x0852 | 32131 | 4 | HCC Channel 5 Energy Rate 4 | R | Float | Kwh |
| 0x0854 | 32133 | 4 | HCC Channel 6 Energy Rate 4 | R | Float | Kwh |
| 0x0856 | 32135 | 4 | HCC Channel 7 Energy Rate 4 | R | Float | Kwh |
| 0x0858 | 32137 | 4 | HCC Channel 8 Energy Rate 4 | R | Float | Kwh |
| 0x085A | 32139 | 4 | HCC Channel 9 Energy Rate 4 | R | Float | Kwh |
| 0x085C | 32141 | 4 | HCC Channel 10 Energy Rate 4 | R | Float | Kwh |
| 0x085E | 32143 | 4 | HCC Channel 11 Energy Rate 4 | R | Float | Kwh |
| 0x0860 | 32145 | 4 | HCC Channel 12 Energy Rate 4 | R | Float | Kwh |
| 0x0862 | 32147 | 4 | HCC Channel 13 Energy Rate 4 | R | Float | Kwh |
| 0x0864 | 32149 | 4 | HCC Channel 14 Energy Rate 4 | R | Float | Kwh |
| 0x0866 | 32151 | 4 | HCC Channel 15 Energy Rate 4 | R | Float | Kwh |
| 0x0868 | 32153 | 4 | HCC Channel 16 Energy Rate 4 | R | Float | Kwh |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0898 | 32201 | 4 | Export Energy Rate 1 | R | Float | Kwh |
| 0x089A | 32203 | 4 | Export Energy Rate 2 | R | Float | Kwh |
| 0x089C | 32205 | 4 | Export Energy Rate 3 | R | Float | Kwh |
| 0x089E | 32207 | 4 | Export Energy Rate 4 | R | Float | Kwh |
| 0x08A0 | 32209 | 4 | Channel 1 Total Export Energy | R | Float | Kwh |
| 0x08A2 | 32211 | 4 | Channel 2 Total Export Energy | R | Float | Kwh |
| 0x08A4 | 32213 | 4 | Channel 3 Total Export Energy | R | Float | Kwh |
| 0x08A6 | 32215 | 4 | Channel 4 Total Export Energy | R | Float | Kwh |
| 0x08A8 | 32217 | 4 | Channel 5 Total Export Energy | R | Float | Kwh |
| 0x08AA | 32219 | 4 | Channel 6 Total Export Energy | R | Float | Kwh |
| 0x08AC | 32221 | 4 | Channel 7 Total Export Energy | R | Float | Kwh |
| 0x08AE | 32223 | 4 | Channel 8 Total Export Energy | R | Float | Kwh |
| 0x08B0 | 32225 | 4 | Total Export Energy All channels and rates | R | Float | Kwh |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x08B2 | 32227 | 4 | Import Reactive Energy Rate 1 | R | Float | Kvarh |
| 0x08B4 | 32229 | 4 | Import Reactive Energy Rate 2 | R | Float | Kvarh |
| 0x08B6 | 32231 | 4 | Import Reactive Energy Rate 3 | R | Float | Kvarh |
| 0x08B8 | 32233 | 4 | Import Reactive Energy Rate 4 | R | Float | Kvarh |
| 0x08BA | 32235 | 4 | Channel 1 Total Import Reactive Energy | R | Float | Kvarh |
| 0x08BC | 32237 | 4 | Channel 2 Total Import Reactive Energy | R | Float | Kvarh |
| 0x08BE | 32239 | 4 | Channel 3 Total Import Reactive Energy | R | Float | Kvarh |
| 0x08C0 | 32241 | 4 | Channel 4 Total Import Reactive Energy | R | Float | Kvarh |
| 0x08C2 | 32243 | 4 | Channel 5 Total Import Reactive Energy | R | Float | Kvarh |
| 0x08C4 | 32245 | 4 | Channel 6 Total Import Reactive Energy | R | Float | Kvarh |
| 0x08C6 | 32247 | 4 | Channel 7 Total Import Reactive Energy | R | Float | Kvarh |
| 0x08C8 | 32249 | 4 | Channel 8 Total Import Reactive Energy | R | Float | Kvarh |
| 0x08CA | 32251 | 4 | Total Import Reactive Energy All channels and rates | R | Float | Kvarh |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x08CC | 32253 | 4 | Export Reactive Energy Rate 1 | R | Float | Kvarh |
| 0x08CE | 32255 | 4 | Export Reactive Energy Rate 2 | R | Float | Kvarh |
| 0x08D0 | 32257 | 4 | Export Reactive Energy Rate 3 | R | Float | Kvarh |
| 0x08D2 | 32259 | 4 | Export Reactive Energy Rate 4 | R | Float | Kvarh |
| 0x08D4 | 32261 | 4 | Channel 1 Total Export Reactive Energy | R | Float | Kvarh |
| 0x08D6 | 32263 | 4 | Channel 2 Total Export Reactive Energy | R | Float | Kvarh |
| 0x08D8 | 32265 | 4 | Channel 3 Total Export Reactive Energy | R | Float | Kvarh |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x08DA | 32267 | 4 | Channel 4 Total Export Reactive Energy | R | Float | Kvarh |
| 0x08DC | 32269 | 4 | Channel 5 Total Export Reactive Energy | R | Float | Kvarh |
| 0x08DE | 32271 | 4 | Channel 6 Total Export Reactive Energy | R | Float | Kvarh |
| 0x08E0 | 32273 | 4 | Channel 7 Total Export Reactive Energy | R | Float | Kvarh |
| 0x08E2 | 32275 | 4 | Channel 8 Total Export Reactive Energy | R | Float | Kvarh |
| 0x08E4 | 32277 | 4 | Total Export Reactive Energy All channels and rates | R | Float | Kvarh |

## 6.5.2 Energy Pulse Output

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x00C2 | 30195 | 4 | Pulse value for Active Energy | R/W | Float | KWHr |
| 0x00C4 | 30197 | 2 | Pulse high period for Active Energy | R/W | 16 bits | ms |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0102 | 30259 | 4 | Pulse value for Apparent Energy | R/W | Float | KVAHr |
| 0x0104 | 30261 | 2 | Pulse high period for Apparent Energy | R/W | 16 bits | ms |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (Read/ Write) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0142 | 30323 | 4 | Pulse value for Reactive Energy | R/W | Float | KVARHr |
| 0x0144 | 30325 | 2 | Pulse high period for Reactive Energy | R/W | 16 bits | ms |

## 6.6 Transformer Ratios

Values for the VT, CT and NT transformers can be read or set using

- VT Voltage Transformer
- CT Current Transformer
- NT Neutral Current Transformer

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0000 | 30001 | 2 | Primary VT ratio | R/W | 16 Bits | n |
| 0x0001 | 30002 | 2 | Secondary VT ratio | R | 16 Bits | n |
| 0x0002 | 30003 | 2 | Primary CT ratio | R/W | 16 Bits | n |
| 0x0003 | 30004 | 2 | Secondary CT ratio | R | 16 Bits | n |
| 0x0004 | 30005 | 2 | Primary NT ratio | R/W | 16 Bits | n |
| 0x0005 | 30006 | 2 | Secondary NT ratio | R | 16 Bits | n |

## 6.7 Quality

Quality Registers

Current K Factors

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0490 | 31169 | 4 | Current K Factor L1 | R | Float | |
| 0x0492 | 31171 | 4 | Current K Factor L2 | R | Float | |
| 0x0494 | 31173 | 4 | Current K Factor L3 | R | Float | |

THFF

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x04A0 | 31185 | 4 | THFF L1 | R | Float | |
| 0x04A2 | 31187 | 4 | THFF L2 | R | Float | |
| 0x04A4 | 31189 | 4 | THFF L3 | R | Float | |

Sequence Voltage and Current

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x04A6 | 31191 | 4 | Positive Sequence Voltage | R | Float | |
| 0x04A8 | 31193 | 4 | Negative Sequence Voltage | R | Float | |
| 0x04AA | 31195 | 4 | Neutral Sequence Voltage | R | Float | |
| 0x04AC | 31197 | 4 | Positive Sequence Current | R | Float | |
| 0x04AE | 31199 | 4 | Negative Sequence Current | R | Float | |
| 0x04B0 | 31201 | 4 | Neutral Sequence Current | R | Float | |
| 0x04B2 | 31203 | 4 | Voltage Unbalance factor | R | Float | |
| 0x04B4 | 31205 | 4 | Current Unbalance factor | R | Float | |

## 6.8 Total Registers

The register can be found else where. They are re-mapped here to allow easy reading of all total registers.

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0600 | 31537 | 4 | Total Amps | R | Float | A |
| 0x0602 | 31539 | 4 | Total Power | R | Float | KW |
| 0x0604 | 31541 | 4 | Total Apparent Power | R | Float | KVA |
| 0x0606 | 31543 | 4 | Total Reactive Power | R | Float | KVAR |
| 0x0608 | 31545 | 4 | Total Power Factor | R | Float | |
| 0x060A | 31547 | 4 | Total Voltage Harmonic Distortion L1 | R | Float | % |
| 0x060C | 31549 | 4 | Total Voltage Harmonic Distortion L2 | R | Float | % |
| 0x060E | 31551 | 4 | Total Voltage Harmonic Distortion L3 | R | Float | % |
| 0x0610 | 31553 | 4 | Total Current Harmonic Distortion L1 | R | Float | % |
| 0x0612 | 31555 | 4 | Total Current Harmonic Distortion L2 | R | Float | % |
| 0x0614 | 31557 | 4 | Total Current Harmonic Distortion L3 | R | Float | % |
| 0x0616 | 31559 | 4 | Total Current Harmonic Distortion Neutral | R | Float | % |
| 0x0618 | 31561 | 4 | Total Active Energy | R | Float | Kwh |

## 6.9 Miscellaneous

Miscellaneous and other reading values.

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0500 | 31281 | 4 | Import Register Gas | R | Float | Kw |
| 0x0502 | 31283 | 4 | Import Register Water | R | Float | $m^3$ |
| 0x0504 | 31285 | 4 | Import Register Heat | R | Float | Kw |
| 0x0506 | 31287 | 30 | Location Field **(1)** | R/W | String | |
| 0x0524 | 31317 | 2 | Change Modbus Slave Address Number**(2)** | R/W | 16 bit | Address |
| 0x0525 | 31318 | 2 | Change Modbus Slave Baud rate | R/W | 16bit | Baud |
| 0x0526 | 31319 | 2 | Modbus Terminal Resistor | R/W | 16bit | State**(3)** |

**(1)**
Note: Use the identification and additional information functions in stead of this field for compatibility with other Modbus devices
(Function code 0x2B/0x0E)

**(2)**
Note: valid range for address is a number between 1 and 255

**(3)**
this is 16 bits, the lower 8 bits is for the main modbus channel and the upper 8 bits is for the display modbus channel
if the value = 0 then the resistor is off (read from meter) will be turned off (write to meter)
if the value = 1 then the resistor is on (read from meter) will be turned on (write to meter)

i.e. hex value 00 01 = main modbus on and display modbus off

not all meters support the control of the resistor on the display modbus channel

### Extra Meter Information

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0550 | 31361 | 2 | Meter Mode | R | 16 bit | MODE |
| 0x0551 | 31362 | 2 | Field Rotation | R | 16 bit | ROTATION |
| 0x0552 | 31363 | 4 | Date and Time | R/W | 32 bit | EPOCH |
| 0x0554 | 31365 | 4 | Run Time | R | 32 bit | SECONDS |
| 0x0555 | 31367 | 2 | Voltage Health | R | 16bits | VHEALTH |

where:- MODE Meter Mode
0 = 3 Phase 4 wire
1 = 3 Phase 3 wire
2 = 3 Phase 3 wire VT
3 = 1 Phase

where:- ROTATION
0 = Right
1 = Left

where :VHEALTH
bit 0 = L1 Voltage, '0' = OFF '1' = ON
bit 1 = L2 Voltage, '0' = OFF '1' = ON
bit 2 = L3 Voltage, '0' = OFF '1' = ON
bit 15 = Rotation, '0' = Right, '1' = Left

## 6.10 Digital IO

Miscellaneous and other reading values.

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x0900 | 32305 | 2 | Digital Inputs - 16 channel | R | 16 BIT | BIT |
| 0x0901 | 32306 | 2 | Digital Outputs - 16 channel | R/W | 16 BIT | BIT |

The digital input and output are 16 bit wide registers, each bit represents one channel: the LSB bit, bit 0 = Channel 1 and the MSB bit = Channel 16

| Digital input or output register | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| Ch 16 | Ch 15 | Ch14 | Ch 13 | Ch 12 | Ch 11 | Ch 10 | Ch 9 | Ch 8 | Ch 7 | Ch 6 | Ch 5 | Ch 4 | Ch 3 | Ch 2 | Ch 1 |

### Digital Input

Channel value '1' = Contact 'closed' or 'made'
Channel value '0' = Contact 'open' or 'not made'

### Digital Output

Channel value '1' = Output 'ON' or Relay Active (contact closed)
Channel value '0' = Output 'OFF' or Relay NOT Active (contacts open)

## 6.11 SOE and other Events

The last 6 events can be read from the units

Miscellaneous and other reading values.

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x1000 | 34097 | 4 | Event Time 1 | R | 32 BIT | EPOCH |
| 0x1002 | 34099 | 2 | Event Code 1 | R | 16 BIT | EVENT |
| 0x1003 | 34100 | 4 | Event Time 2 | R | 32 BIT | EPOCH |
| 0x1005 | 34102 | 2 | Event Code 2 | R | 16 BIT | EVENT |
| 0x1006 | 34103 | 4 | Event Time 3 | R | 32 BIT | EPOCH |
| 0x1008 | 34105 | 2 | Event Code 3 | R | 16 BIT | EVENT |
| 0x1009 | 34106 | 4 | Event Time 4 | R | 32 BIT | EPOCH |
| 0x100B | 34108 | 2 | Event Code 4 | R | 16 BIT | EVENT |
| 0x100C | 34109 | 4 | Event Time 5 | R | 32 BIT | EPOCH |
| 0x100E | 34111 | 2 | Event Code 5 | R | 16 BIT | EVENT |
| 0x100F | 34112 | 4 | Event Time 6 | R | 32 BIT | EPOCH |
| 0x100B | 34114 | 2 | Event Code 6 | R | 16 BIT | EVENT |

### EVENT CODE FORMAT

An event code is 16 bits and is split into two 8 bits. The High 8 bits is the EVENT TYPE and the Low 8 bits gives extra information for the event

| 16 Bit Event Code | | Comment |
|---|---|---|
| **Bits 15..8** | **Bits 7 .. 0** | |
| **Event Type** | **Event Status / Information** | |
| 0x00 | | Reserved |
| 0x01 | Channel Number 1 to 16 | Digital Input Changed to OPEN |
| 0x02 | Channel Number 1 to 16 | Digital Input Changed to CLOSE |
| 0x03 | Channel Number 1 to 16 | Digital Output Changed to OFF |
| 0x04 | Channel Number 1 to 16 | Digital Output Changed to ON |
| 0x05 to 0xFF | | Reserved |

## 6.12 HCC Setup registers

Registers for HCC setup and information

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x2000 | 38193 | 4 | PPU channel 1 | R/W | float | PPU |
| 0x2002 | 38195 | 4 | PPU channel 2 | R/W | float | PPU |
| 0x2004 | 38197 | 4 | PPU channel 3 | R/W | float | PPU |
| 0x2006 | 38199 | 4 | PPU channel 4 | R/W | float | PPU |
| 0x2008 | 38201 | 4 | PPU channel 5 | R/W | float | PPU |
| 0x200A | 38203 | 4 | PPU channel 6 | R/W | float | PPU |
| 0x200C | 38205 | 4 | PPU channel 7 | R/W | float | PPU |
| 0x200E | 38207 | 4 | PPU channel 8 | R/W | float | PPU |
| 0x2010 | 38209 | 4 | PPU channel 9 | R/W | float | PPU |
| 0x2012 | 38211 | 4 | PPU channel 10 | R/W | float | PPU |
| 0x2014 | 38213 | 4 | PPU channel 11 | R/W | float | PPU |
| 0x2016 | 38215 | 4 | PPU channel 12 | R/W | float | PPU |
| 0x2018 | 38217 | 4 | PPU channel 13 | R/W | float | PPU |
| 0x201A | 38219 | 4 | PPU channel 14 | R/W | float | PPU |
| 0x201C | 38221 | 4 | PPU channel 15 | R/W | float | PPU |
| 0x201E | 38223 | 4 | PPU channel 16 | R/W | float | PPU |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x2020 | 38225 | 2 | Debounce Time Channel 1 | R/W | U16 | ms |
| 0x2021 | 38226 | 2 | Debounce Time Channel 2 | R/W | U16 | ms |
| 0x2022 | 38227 | 2 | Debounce Time Channel 3 | R/W | U16 | ms |
| 0x2023 | 38228 | 2 | Debounce Time Channel 4 | R/W | U16 | ms |
| 0x2024 | 38229 | 2 | Debounce Time Channel 5 | R/W | U16 | ms |
| 0x2025 | 38230 | 2 | Debounce Time Channel 6 | R/W | U16 | ms |
| 0x2026 | 38231 | 2 | Debounce Time Channel 7 | R/W | U16 | ms |
| 0x2027 | 38232 | 2 | Debounce Time Channel 8 | R/W | U16 | ms |
| 0x2028 | 38233 | 2 | Debounce Time Channel 9 | R/W | U16 | ms |
| 0x2029 | 38234 | 2 | Debounce Time Channel 10 | R/W | U16 | ms |
| 0x202A | 38235 | 2 | Debounce Time Channel 11 | R/W | U16 | ms |
| 0x202B | 38236 | 2 | Debounce Time Channel 12 | R/W | U16 | ms |
| 0x202C | 38237 | 2 | Debounce Time Channel 13 | R/W | U16 | ms |
| 0x202D | 38238 | 2 | Debounce Time Channel 14 | R/W | U16 | ms |
| 0x202E | 38239 | 2 | Debounce Time Channel 15 | R/W | U16 | ms |
| 0x202F | 38240 | 2 | Debounce Time Channel 16 | R/W | U16 | ms |

Note: Valid range is 20 to 120 ms

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x2030 | 38241 | 2 | Tariff Type channel 1 | R/W | U16 | |
| 0x2031 | 38242 | 2 | Tariff Type channel 2 | R/W | U16 | |
| 0x2032 | 38243 | 2 | Tariff Type channel 3 | R/W | U16 | |
| 0x2033 | 38244 | 2 | Tariff Type channel 4 | R/W | U16 | |
| 0x2034 | 38245 | 2 | Tariff Type channel 5 | R/W | U16 | |
| 0x2035 | 38246 | 2 | Tariff Type channel 6 | R/W | U16 | |
| 0x2036 | 38247 | 2 | Tariff Type channel 7 | R/W | U16 | |
| 0x2037 | 38248 | 2 | Tariff Type channel 8 | R/W | U16 | |
| 0x2038 | 38249 | 2 | Tariff Type channel 9 | R/W | U16 | |
| 0x2039 | 38250 | 2 | Tariff Type channel 10 | R/W | U16 | |
| 0x203A | 38251 | 2 | Tariff Type channel 11 | R/W | U16 | |
| 0x203B | 38252 | 2 | Tariff Type channel 12 | R/W | U16 | |
| 0x203C | 38253 | 2 | Tariff Type channel 13 | R/W | U16 | |
| 0x203D | 38254 | 2 | Tariff Type channel 14 | R/W | U16 | |
| 0x203E | 38255 | 2 | Tariff Type channel 15 | R/W | U16 | |
| 0x203F | 38256 | 2 | Tariff Type channel 16 | R/W | U16 | |

Note: valid range is 1, 2, 3 or 4

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x2040 | 38257 | 2 | Tariff 1 Time Channel 1 | R/W | U16 | HH:MM |
| 0x2041 | | 2 | Tariff 2 Time Channel 1 | R/W | U16 | HH:MM |
| 0x2042 | | 2 | Tariff 3 Time Channel 1 | R/W | U16 | HH:MM |
| 0x2043 | | 2 | Tariff 4 Time Channel 1 | R/W | U16 | HH:MM |
| 0x2044 | | 2 | Tariff 1 Time Channel 2 | R/W | U16 | HH:MM |
| 0x2045 | | 2 | Tariff 2 Time Channel 2 | R/W | U16 | HH:MM |
| 0x2046 | | 2 | Tariff 3 Time Channel 2 | R/W | U16 | HH:MM |
| 0x2047 | | 2 | Tariff 4 Time Channel 2 | R/W | U16 | HH:MM |
| 0x2048 | | 2 | Tariff 1 Time Channel 3 | R/W | U16 | HH:MM |
| 0x2049 | | 2 | Tariff 2 Time Channel 3 | R/W | U16 | HH:MM |
| 0x204A | | 2 | Tariff 3 Time Channel 3 | R/W | U16 | HH:MM |
| 0x204B | | 2 | Tariff 4 Time Channel 3 | R/W | U16 | HH:MM |
| 0x204C | | 2 | Tariff 1 Time Channel 4 | R/W | U16 | HH:MM |
| 0x204D | | 2 | Tariff 2 Time Channel 4 | R/W | U16 | HH:MM |
| 0x204E | | 2 | Tariff 3 Time Channel 4 | R/W | U16 | HH:MM |
| 0x204F | | 2 | Tariff 4 Time Channel 4 | R/W | U16 | HH:MM |
| 0x2050 | | 2 | Tariff 1 Time Channel 5 | R/W | U16 | HH:MM |
| 0x2051 | | 2 | Tariff 2 Time Channel 5 | R/W | U16 | HH:MM |
| 0x2052 | | 2 | Tariff 3 Time Channel 5 | R/W | U16 | HH:MM |
| 0x2053 | | 2 | Tariff 4 Time Channel 5 | R/W | U16 | HH:MM |
| 0x2054 | | 2 | Tariff 1 Time Channel 6 | R/W | U16 | HH:MM |
| 0x2055 | | 2 | Tariff 2 Time Channel 6 | R/W | U16 | HH:MM |
| 0x2056 | | 2 | Tariff 3 Time Channel 6 | R/W | U16 | HH:MM |
| 0x2057 | | 2 | Tariff 4 Time Channel 6 | R/W | U16 | HH:MM |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x2058 | | 2 | Tariff 1 Time Channel 7 | R/W | U16 | HH:MM |
| 0x2059 | | 2 | Tariff 2 Time Channel 7 | R/W | U16 | HH:MM |
| 0x205A | | 2 | Tariff 3 Time Channel 7 | R/W | U16 | HH:MM |
| 0x205B | | 2 | Tariff 4 Time Channel 7 | R/W | U16 | HH:MM |
| 0x205C | | 2 | Tariff 1 Time Channel 8 | R/W | U16 | HH:MM |
| 0x205D | | 2 | Tariff 2 Time Channel 8 | R/W | U16 | HH:MM |
| 0x205E | | 2 | Tariff 3 Time Channel 8 | R/W | U16 | HH:MM |
| 0x205F | | 2 | Tariff 4 Time Channel 8 | R/W | U16 | HH:MM |
| 0x2060 | | 2 | Tariff 1 Time Channel 9 | R/W | U16 | HH:MM |
| 0x2061 | | 2 | Tariff 2 Time Channel 9 | R/W | U16 | HH:MM |
| 0x2062 | | 2 | Tariff 3 Time Channel 9 | R/W | U16 | HH:MM |
| 0x2063 | | 2 | Tariff 4 Time Channel 9 | R/W | U16 | HH:MM |
| 0x2064 | | 2 | Tariff 1 Time Channel 10 | R/W | U16 | HH:MM |
| 0x2065 | | 2 | Tariff 2 Time Channel 10 | R/W | U16 | HH:MM |
| 0x2066 | | 2 | Tariff 3 Time Channel 10 | R/W | U16 | HH:MM |
| 0x2067 | | 2 | Tariff 4 Time Channel 10 | R/W | U16 | HH:MM |
| 0x2068 | | 2 | Tariff 1 Time Channel 11 | R/W | U16 | HH:MM |
| 0x2069 | | 2 | Tariff 2 Time Channel 11 | R/W | U16 | HH:MM |
| 0x206A | | 2 | Tariff 3 Time Channel 11 | R/W | U16 | HH:MM |
| 0x206B | | 2 | Tariff 4 Time Channel 11 | R/W | U16 | HH:MM |
| 0x206C | | 2 | Tariff 1 Time Channel 12 | R/W | U16 | HH:MM |
| 0x206D | | 2 | Tariff 2 Time Channel 12 | R/W | U16 | HH:MM |
| 0x206E | | 2 | Tariff 3 Time Channel 12 | R/W | U16 | HH:MM |
| 0x206F | | 2 | Tariff 4 Time Channel 12 | R/W | U16 | HH:MM |
| 0x2070 | | 2 | Tariff 1 Time Channel 13 | R/W | U16 | HH:MM |
| 0x2071 | | 2 | Tariff 2 Time Channel 13 | R/W | U16 | HH:MM |
| 0x2072 | | 2 | Tariff 3 Time Channel 13 | R/W | U16 | HH:MM |
| 0x2073 | | 2 | Tariff 4 Time Channel 13 | R/W | U16 | HH:MM |
| 0x2074 | | 2 | Tariff 1 Time Channel 14 | R/W | U16 | HH:MM |
| 0x2075 | | 2 | Tariff 2 Time Channel 14 | R/W | U16 | HH:MM |
| 0x2076 | | 2 | Tariff 3 Time Channel 14 | R/W | U16 | HH:MM |
| 0x2077 | | 2 | Tariff 4 Time Channel 14 | R/W | U16 | HH:MM |
| 0x2078 | | 2 | Tariff 1 Time Channel 15 | R/W | U16 | HH:MM |
| 0x2079 | | 2 | Tariff 2 Time Channel 15 | R/W | U16 | HH:MM |
| 0x207A | | 2 | Tariff 3 Time Channel 15 | R/W | U16 | HH:MM |
| 0x207B | | 2 | Tariff 4 Time Channel 15 | R/W | U16 | HH:MM |
| 0x207C | | 2 | Tariff 1 Time Channel 16 | R/W | U16 | HH:MM |
| 0x207D | | 2 | Tariff 2 Time Channel 16 | R/W | U16 | HH:MM |
| 0x207E | | 2 | Tariff 3 Time Channel 16 | R/W | U16 | HH:MM |
| 0x207F | | 2 | Tariff 4 Time Channel 16 | R/W | U16 | HH:MM |

Note format:

16 bits = upper 8 bits : lower 8 bits

upper 8 bits = hours. valid range is 0 to 23
lower 8 bits = minutes. valid range is 0 to 59

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x2090 | 38337 | 20 | Meter Type Channel 1 | R/W | char[20] | |
| 0x209A | 38347 | 20 | Meter Type Channel 2 | R/W | char[20] | |
| 0x20A4 | 38357 | 20 | Meter Type Channel 3 | R/W | char[20] | |
| 0x20AE | 38367 | 20 | Meter Type Channel 4 | R/W | char[20] | |
| 0x20B8 | 38377 | 20 | Meter Type Channel 5 | R/W | char[20] | |
| 0x20C2 | 38387 | 20 | Meter Type Channel 6 | R/W | char[20] | |
| 0x20CC | 38397 | 20 | Meter Type Channel 7 | R/W | char[20] | |
| 0x20B6 | 38407 | 20 | Meter Type Channel 8 | R/W | char[20] | |
| 0x20E0 | 38417 | 20 | Meter Type Channel 9 | R/W | char[20] | |
| 0x20EA | 38427 | 20 | Meter Type Channel 10 | R/W | char[20] | |
| 0x20F4 | 38437 | 20 | Meter Type Channel 11 | R/W | char[20] | |
| 0x20FE | 38447 | 20 | Meter Type Channel 12 | R/W | char[20] | |
| 0x2108 | 38457 | 20 | Meter Type Channel 13 | R/W | char[20] | |
| 0x2112 | 38467 | 20 | Meter Type Channel 14 | R/W | char[20] | |
| 0x211C | 38477 | 20 | Meter Type Channel 15 | R/W | char[20] | |
| 0x2126 | 38487 | 20 | Meter Type Channel 16 | R/W | char[20] | |
| 0x2130 | 38497 | 20 | Meter Description Channel 1 | R/W | char[20] | |
| 0x213A | 38507 | 20 | Meter Description Channel 2 | R/W | char[20] | |
| 0x2144 | 38517 | 20 | Meter Description Channel 3 | R/W | char[20] | |
| 0x214E | 38527 | 20 | Meter Description Channel 4 | R/W | char[20] | |
| 0x2158 | 38537 | 20 | Meter Description Channel 5 | R/W | char[20] | |
| 0x2162 | 38547 | 20 | Meter Description Channel 6 | R/W | char[20] | |
| 0x216C | 38557 | 20 | Meter Description Channel 7 | R/W | char[20] | |
| 0x2176 | 38567 | 20 | Meter Description Channel 8 | R/W | char[20] | |
| 0x2180 | 38577 | 20 | Meter Description Channel 9 | R/W | char[20] | |
| 0x218A | 38587 | 20 | Meter Description Channel 10 | R/W | char[20] | |
| 0x2194 | 38597 | 20 | Meter Description Channel 11 | R/W | char[20] | |
| 0x219E | 38607 | 20 | Meter Description Channel 12 | R/W | char[20] | |
| 0x21A8 | 38617 | 20 | Meter Description Channel 13 | R/W | char[20] | |
| 0x21B2 | 38627 | 20 | Meter Description Channel 14 | R/W | char[20] | |
| 0x21BC | 38637 | 20 | Meter Description Channel 15 | R/W | char[20] | |
| 0x21C6 | 38647 | 20 | Meter Description Channel 16 | R/W | char[20] | |

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x21D0 | 38657 | 20 | HCC Description | R/W | char[20] | |
| 0x21DA | 38667 | 4 | HCC Serial Number | R/W | U32 | |
| 0x21DB | 38668 | 1 | HCC Command | W | U16 | function |
| 0x21DC | 38669 | 2 | HCC Version | R | U32 | |

## 6.13 Heat meters

Values for the Heat meter measurements can be read from these registers.

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0x9C40 | 40001<br>40002 | 4 | Serial Number, value between<br>0 and 4294967296 | R | 32bit unsigned | |
| 0x9C42 | 40003<br>40004 | 4 | Energy, scaled as 0.01 * value<br>0.00 to 42949672.96 KWh | R | 32bit unsigned | KWh |
| 0x9C44 | 40005 | 2 | Flow Temperature, scaled as 0.01 * value<br>0.00 to 655.35 | R | 16bit unsigned | ºC |
| 0x9C45 | 40006 | 2 | Return Temperature, scaled as 0.01 * value<br>0.00 to 655.35 | R | 16bit unsigned | ºC |
| 0x9C46 | 40007 | 2 | Flow, scaled as 0.01 * value<br>0.00 to 655.35 litres per hour (l/h) | R | 16bit unsigned | l/h |
| 0x9C47 | 40008 | 2 | Volume, scaled as 0.01 * value<br>0.00 to 655.35 meters^3 | R | 16bit unsigned | m^3 |
| | | | | | | |

## 6.14 Broadcast messages

Registers for setting time values. If the modbus address is set to 0, then it is a broadcast message to all nodes that can accept the time messages

| Address (hex) | Register (decimal) | Length (bytes) | Parameter Name | Access (R/W) | Data Format | Units |
|---|---|---|---|---|---|---|
| 0xA000 | | 4 | GMT Time | W | U32 | EPOCH |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Part VII

# 7 Examples

This section gives examples for some of the modbus features:

## 7.1 Modbus Termination Resistor

Commands to turn on or off the modbus resistor, the value is 16 bits and allows the control of the terminal resistor on the main modbus channel or the display channel (Note: most meters will only have one resistor that can be controlled on the main modbus, the display modbus will be fixed)

modbus resistor command

read or write of 16 bits from 0x0526

read request from meter with Modbus Address of 01

01 04 05 26 00 01 CRC CRC

response

01 04 02 00 00 CRC CRC

where

01 = Modbus ID
04 = Request Type
02 = Number of bytes
00 = State of Display modbus resistor (off in this case)
00 = state of main modbus resistor (off in this case)
CRC
CRC

if the response has been

01 04 02 00 01 CRC CRC

where
01 = Modbus ID
04 = Request Type
02 = Number of bytes
00 = State of Display modbus resistor (off in this case)
01 = state of main modbus resistor (on in this case)
CRC
CRC

Changing the modbus resistor

you need to write a new value to register 0x0526 using function code 0x06

example to set the main modbus resistor on

sent to meter

```
01 06 05 26 00 01 CRC CRC

response
01 06 05 26 00 01 CRC CRC


example to set main modbus resistor off
sent to meter
01 06 05 26 00 00 CRC CRC

response
01 06 05 26 00 00 CRC CRC
```

## 7.2 Request Energy

Commands to request the Energy Register, 30353, the return value is a 32 bit floating point number. See the IEEE section for the format.

modbus resistor command

Read 32 bits from 30353 (0x0160)

read request from meter with Modbus Address of 01

```
01 04 01 60 00 02 CRC CRC
```

response

```
01 04 04 00 00 00 00 CRC CRC
```

where

```
01 = Modbus ID
04 = Request Type
04 = Number of bytes
00 = Floating point number in 32 bits, the value is 0
00 =
00 =
00 =
CRC
CRC
```

if the energy register had been 1234.56 the 32 bit IEEE hex format would be 0x449A51EC and the response would have been

```
01 04 04 44 9A 51 EC F3 46
```

Back Cover