GEM230-CT Input Registers

Address (Register)	Input Register Parameter				Modbus Protocol Start Address Hex	
	Description	Length (bytes)	Data Format	Units	Hi Byte	Lo Byte
30001	Phase 1 line to neutral volts.	4	Float	V	00	00
30007	Phase 1 current.	4	Float	A	00	06
30013	Phase 1 power.	4	Float	W	00	0C
30019	Phase 1 apparent power.	4	Float	VA	00	12
30025	Phase 1 reactive power.	4	Float	VAr	00	18
30031	Phase 1 power factor	4	Float	None	00	1E
30037	Phase 1 phase angle.	4	Float	Degrees	00	24
30071	Frequency of supply voltages.	4	Float	Hz	00	46
30073	Import active energy since last reset	4	Float	kWh	00	48
30075	Export active energy since last reset	4	Float	kWh	00	4A
30077	Import reactive energy since last reset	4	Float	kVArh	00	4C
30079	Export reactive energy since last reset	4	Float	kVArh	00	4E
30085		4	Float	W	00	54
30087		4	Float	W	00	56
30089		4	Float	W	00	58
30091		4	Float	W	00	5A
30093		4	Float	W	00	5C
30095		4	Float	W	00	5E
30259		4	Float	A	01	02
30265		4	Float	A	01	08
30343	Total active energy	4	Float	kWh	01	56
30345	Total reactive energy	4	Float	kVArh	01	58
30385	Current resettable total active energy	4	Float	kWh	01	80
30387	Current resettable total reactive energy	4	Float	kWh	01	82
34877	Total active Energy Rate 1	4	Float	kWh	13	0C
34879	Total active Energy Rate 2	4	Float	kWh	13	0E
34885	Import active Energy Rate 1	4	Float	kWh	13	14
34887	Import active Energy Rate 2	4	Float	kWh	13	16
34893	Export active Energy Rate 1	4	Float	kWh	13	1C
34895	Export active Energy Rate 2	4	Float	kWh	13	1E
34901	Total reactive Energy Rate 1	4	Float	kVArh	13	24
34903	Total reactive Energy Rate 2	4	Float	kVArh	13	26
34909	Import reactive Energy Rate 1	4	Float	kVArh	13	2C
34911	Import reactive Energy Rate 2	4	Float	kVArh	13	2E
34917	Export reactive Energy Rate 1	4	Float	kVArh	13	34
34919	Export reactive Energy Rate 2	4	Float	kVArh	13	36
35075	Maximum system positive active power demand of Tarrif 1	4	Float	kW	13	D2
35077	Maximum system positive active power demand of Tarrif 2	4	Float	kW	13	D4
35205	Maximum system reverse active power demand of Tarrif 1	4	Float	kW	14	54
35207	Maximum system reverse active	4	Float	kW	14	56

	power demand of Tarrif 2					
35473	Tariff1 total active Maximum Power Demand	4	Float	W	15	60
35479	Tariff 1 L1 Maximum Current Demand	4	Float	A	15	66
35487	Tariff 2 total active Maximum Power Demand	4	Float	W	15	6E
35493	Tariff 2 L1 Maximum Current Demand	4	Float	A	15	74

1.3 Modbus Protocol Holding Registers and Digital meter set up

Holding registers are used to store and display instrument configuration settings. All holding registers not listed in the table below should be considered as reserved for manufacturer use and no attempt should be made to modify their values.

The holding register parameters may be viewed or changed using the Modbus Protocol. Each parameter is held in two consecutive 4X registers. Modbus Protocol Function Code 03 is used to read the parameter and Function Code 16 is used to write. Write to only one parameter per message.

Address Register	**Parameter**	**Modbus Protocol Start Address Hex**		**Valid range**	**Mode**
		High Byte	**Low Byte**		
40013	Pulse output 1 Width	00	OC	Write pulse on period in milliseconds: 60, 100 or 200, default 100. **Length : 4 byte** **Data Format : Float**	r/w
40019	Network Parity Stop	00	12	Write the network port parity/stop bits for MODBUS Protocol, where: 0 = One stop bit and no parity, default. 1 = One stop bit and even parity. 2 = One stop bit and odd parity.3 = Two stop bits and no parity. **Length : 4 byte** **Data Format : Float**	r/w
40021	Network Node	00	14	Write the network port node address: 1 to 247 for MODBUS Protocol, default 1. **Length : 4 byte** **Data Format : Float**	r/w
40029	Network Baud Rate	00	1C	Write the network port baud rate for MODBUS Protocol, where: 0 = 2400 baud. 1 = 4800 baud. 2 = 9600 baud, default. 5=1200 baud . **Length : 4 byte** **Data Format : Float**	r/w
40087	Pulse 1 Energy Type	00	56	Write MODBUS Protocol input parameter for pulse out 1: 1: import active energy 2: total active energy 4: export active energy, default 5: import reactive energy	r/w

				6: total reactive energy 8: export reactive energy **Length : 4 byte** **Data Format : Float**	
461457					wo
462721	Demand interval , slide time, automatic scroll display interval(scroll Time)，Backlight time	F5	00	数据格式：需量时间-滑差时间-轮显时间-背光点亮时间，单位是 min-min-s-min，默认 15-01-00-60 Note： Demand interval =0：需量不更新 scroll time=0：the display does not scroll automatically. Backlight time=0 Backlight always on **Length : 4 byte** **Data Format : BCD**	r/w
463761	Pulse 1 Divisor	F9	10	Write pulse divisor index: n = 0 to 3 00 00:0.001 kWh(KVArh)/imp,default 00 01:0.01 kWh(KVArh)/imp 00 02:0.1 kWh(KVArh)/imp 00 03:1 kWh(KVArh)/imp **Length : 2 byte** **Data Format : Hex**	r/w
463777	Measurement mode	F9	20	00 01：total = import 00 02：total = import + export 00 03：total = import – export **Length : 2 byte** **Data Format : Hex**	r/w
463793	Running time	F9	30	Continuous working period--hour **Length : 4 byte** **Data Format : Float**	r/w

Note：
Mode 1: Measure imported energy, Total energy=Imported energy.
Mode 2: Measure imported energy and exported energy, Total energy=Import energy + export energy(default).
Mode 3: Measure imported energy and exported energy, Total energy=Imported energy- exported energy.

2 RS485 General Information

Some of the information in this section relates to other Eetarp Digital meter product families, and is included to assist where a mixed network is implemented.RS485 or EIA (Electronic Industries Association) RS485 is a balanced line, half-duplex transmission system allowing transmission distances of up to 1.2 km. The following table summarizes the RS-485 Standard:

PARAMETER	
Mode of Operation	Differential
Number of Drivers and Receivers	32 Drivers, 32 Receivers
Maximum Cable Length	1200 m
Maximum Data Rate	10 M baud
Maximum Common Mode Voltage	12 V to –7 V

Minimum Driver Output Levels (Loaded)	+/– 1.5 V
Minimum Driver Output Levels (Unloaded)	+/– 6 V
Drive Load	Minimum 60 ohms
Driver Output Short Circuit Current Limit	150 mA to Gnd, 250 mA to 12 V 250 mA to –7 V
Minimum Receiver Input Resistance	12 kohms
Receiver Sensitivity	+/– 200 mV

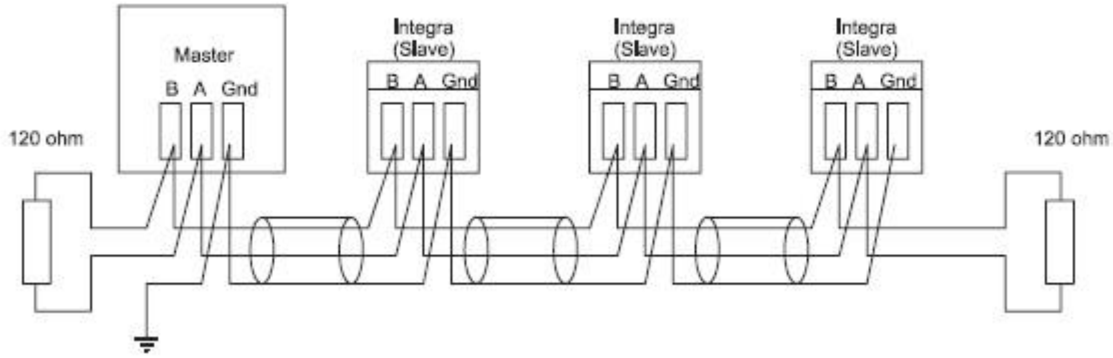
Further information relating to RS485 may be obtained from either the EIA or the various RS485 device manufacturers, for example Texas Instruments or Maxim Semiconductors. This list is not exhaustive.

2.1 Half Duplex

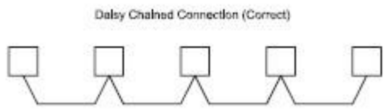
Half duplex is a system in which one or more transmitters (talkers) can communicate with one or more receivers (listeners) with only one transmitter being active at any one time. For example, a “conversation” is started by asking a question, the person who has asked the question will then listen until he gets an answer or until he decides that the individual who was asked the question is not going to reply.
In a 485 network the “master” will start the “conversation” with a “query” addressed to a specific “slave”, the “master” will then listen for the “slave’s” response. If the “slave” does not respond within a pre-defined period, (set by control software in the “master”), the “master” will abandon the “conversation”.

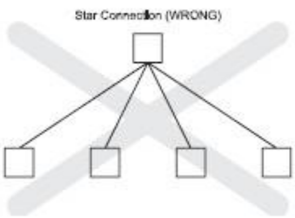
2.2 Connecting the Instruments

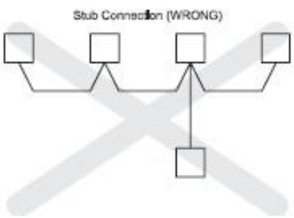
If connecting an RS485 network to a PC use caution if contemplating the use of an RS232 to 485 converter together with a USB to RS485 adapter. Consider either an RS232 to RS485 converter, connected directly to a suitable RS232 jack on the PC, or use a USB to RS485 converter or, for desktop PCs a suitable plug in RS485 card. (*Many 232:485 converters draw power from the RS232 socket. If using a USB to RS232 adapter, the adapter may not have enough power available to run the 232:485 converter*.)
Screened twisted pair cable should be used. For longer cable runs or noisier environments, use of a cable specifically designed for RS485 may be necessary to achieve optimum performance. All “A” terminals should be connected together using one conductor of the twisted pair cable, all “B” terminals should be connected together using the other conductor in the pair. The cable screen should be connected to the “Gnd” terminals.
A Belden 9841 (Single pair) or 9842 (Two pair) or similar cable with a characteristic impedance of 120 ohms is recommended. The cable should be terminated at each end with a 120 ohm, quarter watt (or greater) resistor. Note: Diagram shows wiring topology only. Always follow terminal identification on Eetarp Digital meter product label.



There must be no more than two wires connected to each terminal, this ensures that a “Daisy Chain or “straight line” configuration is used. A “Star” or a network with “Stubs (Tees)” is not recommended as reflections within the cable may result in data corruption.

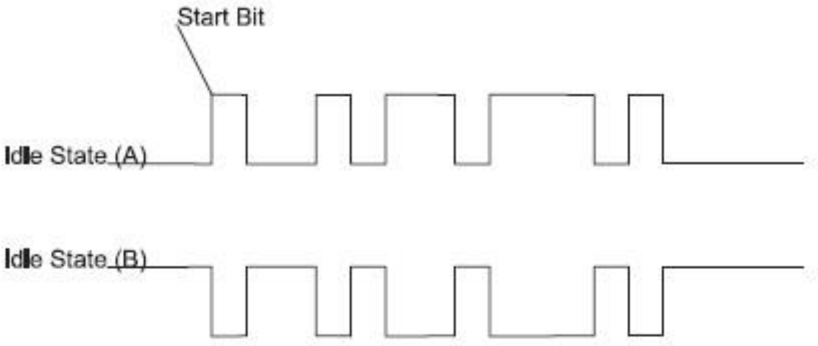






2.3 A and B terminals

The A and B connections to the Eetarp Digital meter products can be identified by the signals present on them whilst there is activity on the RS485 bus:

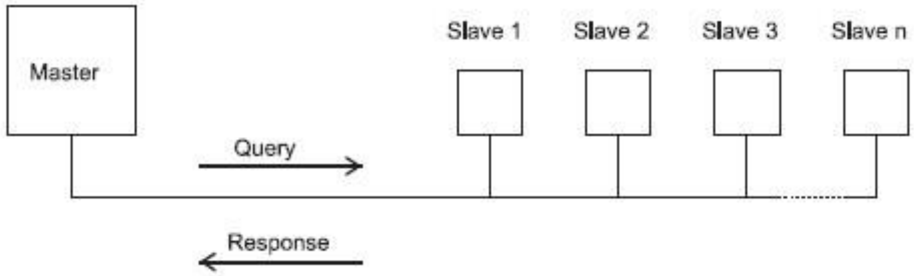


2.4 Troubleshooting

•Start with a simple network, one master and one slave. With Eetarp Digital meter products this is easily achieved as the network can be left intact whilst individual instruments are disconnected by removing the RS485 connection from the rear of the instrument.
·Check that the network is connected together correctly. That is all of the “A’s” are connected together, and all of the “B’s” are connected together, and also that all of the “Gnd’s” are connected together.
·Confirm that the data “transmitted” onto the RS485 is not echoed back to the PC on the RS232 lines.(This facility is sometimes a link option within the converter). Many PC based packages seem to not perform well when they receive an echo of the message they are transmitting. SpecView and PCView (PC software) with a RS232 to RS485 converter are believed to include this feature.
·Confirm that the Address of the instrument is the same as the “master” is expecting.
·If the “network” operates with one instrument but not more than one check that each instrument has a unique address.
· Each request for data must be restricted to 40 parameters or less. Violating this requirement will impact the performance of the instrument and may result in a response time in excess of the specification.
·Check that the MODBUS Protocol mode (RTU or ASCII) and serial parameters (baud rate, number of data bits, number of stop bits and parity) are the same for all devices on the network.
·Check that the “master” is requesting floating-point variables (pairs of registers placed on floating point boundaries) and is not “splitting” floating point variables.
·Check that the floating-point byte order expected by the “master” is the same as that used by Eetarp Digital meter products.
·If possible obtain a second RS232 to RS485 converter and connect it between the RS485 bus and an additional PC equipped with a software package, which can display the data on the bus. Check for the existence of valid requests.

3 MODBUS Protocol General Information

Communication on a MODBUS Protocol Network is initiated (started) by a “Master” sending a query to a “Slave”. The “Slave“, which is constantly monitoring the network for queries addressed to it, will respond by performing the requested action and sending a response back to the ”Master”. Only the “Master” can initiate a query.



In the MODBUS Protocol the master can address individual slaves, or, using a special “Broadcast” address, can initiate a broadcast message to all slaves. The Eetarp Digital meter do not support the broadcast address.

3.1 MODBUS Protocol Message Format

The MODBUS Protocol defines the format for the master’s query and the slave’s response.
The query contains the device (or broadcast) address, a function code defining the requested action, any data to be sent, and an error-checking field.
The response contains fields confirming the action taken, any data to be returned, and an error-checking field. If an error occurred in receipt of the message then the message is ignored, if the slave is unable to perform the requested action, then it will construct an error message and send it as its response. The MODBUS Protocol functions used by the Eetarp Digital meters copy 16 bit register values between master and slaves. However, the data used by the Eetarp Digital meter is in 32 bit IEEE 754 floating point format. Thus each instrument parameter is conceptually held in two adjacent MODBUS Protocol registers. Query
The following example illustrates a request for a single floating point parameter i.e. two 16-bit Modbus Protocol Registers.

First Byte Last Byte

Slave Address	Function Code	Start Address (Hi)	Start Address (Lo)	Number of Points (Hi)	Number of Points (Lo)	Error Check (Lo)	Error Check (Hi)

Slave Address: 8-bit value representing the slave being addressed (1 to 247), 0 is reserved for the broadcast address. The Eetarp Digital meters do not support the broadcast address.
Function Code: 8-bit value telling the addressed slave what action is to be performed. (3, 4, 8 or 16 are valid for Eetarp Digital meter)
Start Address (Hi): The top (most significant) eight bits of a 16-bit number specifying the start address of the data being requested.
Start Address (Lo): The bottom (least significant) eight bits of a 16-bit number specifying the start address of the data being requested. As registers are used in pairs and start at zero, then this must be an even number.
Number of Points (Hi): The top (most significant) eight bits of a 16-bit number specifying the number of registers being requested.
Number of Points (Lo): The bottom (least significant) eight bits of a 16-bit number specifying the number of registers being requested. As registers are used in pairs, then this must be an even number.
Error Check (Lo): The bottom (least significant) eight bits of a 16-bit number representing the error check value.
Error Check (Hi): The top (most significant) eight bits of a 16-bit number representing the error check value.

Response
The example illustrates the normal response to a request for a single floating point parameter i.e. two 16-bit Modbus Protocol Registers.

First Byte Last Byte

Slave Address	Function Code	Byte Count	First Register (Hi)	First Register (Lo)	Second Register (Hi)	Second Register (Lo)	Error Check (Lo)	Error Check (Hi)

Slave Address: 8-bit value representing the address of slave that is responding.
Function Code: 8-bit value which, when a copy of the function code in the query, indicates that the slave recognised the query and has responded. (See also Exception Response).
Byte Count: 8-bit value indicating the number of data bytes contained within this response
First Register (Hi)*: The top (most significant) eight bits of a 16-bit number representing the first register requested in the query.
First Register (Lo)*: The bottom (least significant) eight bits of a 16-bit number representing the first register requested in the query.
Second Register (Hi)*: The top (most significant) eight bits of a 16-bit number representing the second register requested in the query.
Second Register (Lo)*: The bottom (least significant) eight bits of a 16-bit number representing the second register requested in the query.
Error Check (Lo): The bottom (least significant) eight bits of a 16-bit number representing the error check value.
Error Check (Hi): The top (most significant) eight bits of a 16-bit number representing the error check value.
*These four bytes together give the value of the floating point parameter requested.

Exception Response
If an error is detected in the content of the query (excluding parity errors and Error Check mismatch), then an error response (called an exception response), will be sent to the master. The exception response is identified by the function code being a copy of the query function code but with the most-significant bit set. The data contained in an exception response is a single byte error code.

First Byte Last Byte

Slave Address	Function Code	Error Code	Error Check (Lo)	Error Check (Hi)

Slave Address: 8-bit value representing the address of slave that is responding.
Function Code: 8 bit value which is the function code in the query OR'ed with 80 hex, indicating that the slave either does not recognise the query or could not carry out the action requested.
Error Code: 8-bit value indicating the nature of the exception detected. (See “Table Of Exception Codes“ later).
Error Check (Lo): The bottom (least significant) eight bits of a 16-bit number representing the error check value.
Error Check (Hi): The top (most significant) eight bits of a 16-bit number representing the error check value.

3.2 Serial Transmission Modes

There are two MODBUS Protocol serial transmission modes, ASCII and RTU. Eetarp Digital meters do not support the ASCII mode.
In RTU (Remote Terminal Unit) mode, each 8-bit byte is used in the full binary range and is not limited to ASCII characters as in ASCII Mode. The greater data density allows better data throughput for the same baud rate, however each message must be transmitted in a continuous stream. This is very unlikely to be a problem for modern communications equipment.

Coding System: Full 8-bit binary per byte. In this document, the value of each byte will be shown as two hexadecimal characters each in the range 0-9 or A-F.
Line Protocol: 1 start bit, followed by the 8 data bits. The 8 data bits are sent with least significant bit first.

User Option Of Parity No Parity and 2 Stop Bits
And Stop Bits: No Parity and 1 Stop Bit
Even Parity and 1 Stop Bit
Odd Parity and 1 Stop Bit.
User Option of Baud 1200，2400，4800 ，9600

The baud rate, parity and stop bits must be selected to match the master’s setting**s.**

3.3 MODBUS Protocol Message Timing (RTU Mode)

A MODBUS Protocol message has defined beginning and ending points. The receiving devices recognizes the start of the message, reads the “Slave Address” to determine if they are being addressed and knowing when the message is completed they can use the Error Check bytes and parity bits to confirm the integrity of the message. If the Error Check or parity fails then the message is discarded.
In RTU mode, messages starts with a silent interval of at least 3.5 character times.
The first byte of a message is then transmitted, the device address.
Master and slave devices monitor the network continuously, including during the ‘silent’ intervals.
When the first byte (the address byte) is received, each device checks it to find out if it is the addressed device. If the device determines that it is the one being addressed it records the whole message and acts accordingly, if it is not being addressed it continues monitoring for the next message.
Following the last transmitted byte, a silent interval of at least 3.5 character times marks the end of the message. A new message can begin after this interval.
The entire message must be transmitted as a continuous stream. If a silent interval of more than 1.5 character times occurs before completion of the message, the receiving device flushes the incomplete message and assumes that the next byte will be the address byte of a new message.
Similarly, if a new message begins earlier than 3.5 character times following a previous message, the receiving device may consider it a continuation of the previous message. This will result in an error, as the value in the final CRC field will not be valid for the combined messages.

3.4 How Characters are Transmitted Serially

When messages are transmitted on standard MODBUS Protocol serial networks each byte is sent in this order (left to right):

Transmit Character = Start Bit + Data Byte + Parity Bit + 1 Stop Bit (11 bits total):

Least Significant Bit (LSB) ↓ (1)　　Most Significant Bit (MSB) ↓ (8)

Start	1	2	3	4	5	6	7	8	Party	Stop

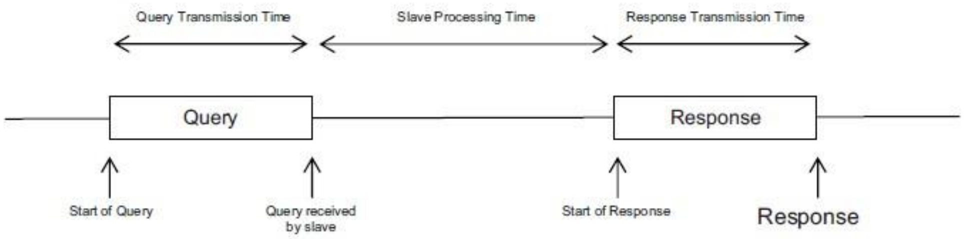
Transmit Character = Start Bit + Data Byte + 2 Stop Bits (11 bits total):

Start	1	2	3	4	5	6	7	8	Stop	Stop

Eetarp Digital meters additionally support No parity, 1 stop bit.
Transmit Character = Start Bit + Data Byte + 1 Stop Bit (10 bits total):

Start	1	2	3	4	5	6	7	8	Stop

The master is configured by the user to wait for a predetermined timeout interval. The master will wait for this period of time before deciding that the slave is not going to respond and that the transaction should be aborted. Care must be taken when determining the timeout period from both the master and the slaves’ specifications. The slave may define the ‘response time’ as being the period from the receipt of the last bit of the query to the transmission of the first bit of the response. The master may define the ‘response time’ as period between transmitting the first bit of the query to the receipt of the last bit of the response. It can be seen that message transmission time, which is a function of the baud rate, must be included in the timeout calculation.



3.5 Error Checking Methods

Standard MODBUS Protocol serial networks use two error checking processes, the error check bytes mentioned above check message integrity whilst Parity checking (even or odd) can be applied to each byte in the message.

3.5.1 Parity Checking

If parity checking is enabled – by selecting either Even or Odd Parity - the quantity of “1’s” will be counted in the data portion of each transmit character. The parity bit will then be set to a 0 or 1 to result in an Even or Odd total of “1’s”.

Note that parity checking can only detect an error if an odd number of bits are picked up or dropped in a transmit character during transmission, if for example two 1’s are corrupted to 0’s the parity check will not find the error.

If No Parity checking is specified, no parity bit is transmitted and no parity check can be made. Also, if No Parity checking is specified and one stop bit is selected the transmit character is effectively shortened by one bit.

3.5.2 CRC Checking

The error check bytes of the MODBUS Protocol messages contain a Cyclical Redundancy Check (CRC) value that is used to check the content of the entire message. The error check bytes must always be present to comply with the MODBUS Protocol, there is no option to disable it.

The error check bytes represent a 16-bit binary value, calculated by the transmitting device. The receiving device must recalculate the CRC during receipt of the message and compare the calculated value to the value received in the error check bytes. If the two values are not equal, the message should be discarded.

The error check calculation is started by first pre-loading a 16-bit register to all 1’s (i.e. Hex (FFFF)) each successive 8-bit byte of the message is applied to the current contents of the register. Note: only the eight bits of data in each transmit character are used for generating the CRC, start bits, stop bits and the parity bit, if one is used, are not included in the error check bytes.

During generation of the error check bytes, each 8-bit message byte is exclusive OR'ed with the lower half of the 16 bit register. The register is then shifted eight times in the direction of the least significant bit (LSB), with a zero filled into the most significant bit (MSB) position. After each shift the LSB prior to the shift is extracted and examined. If the LSB was a 1, the register is then exclusive OR'ed with a pre-set, fixed value. If the LSB was a 0, no exclusive OR takes place.

This process is repeated until all eight shifts have been performed. After the last shift, the next 8-bit message byte is exclusive OR'ed with the lower half of the 16 bit register, and the process repeated.

The final contents of the register, after all the bytes of the message have been applied, is the error check value. In the following pseudo code “Error Word” is a 16-bit value representing the error check values.

```
BEGIN
    Error Word = Hex (FFFF)
    FOR Each byte in message
        Error Word = Error Word XOR byte in message
        FOR Each bit in byte
            LSB = Error Word AND Hex (0001)
            IF LSB = 1 THEN Error Word = Error Word – 1
            Error Word = Error Word / 2
            IF LSB = 1 THEN Error Word = Error Word XOR Hex (A001)
        NEXT bit in byte
NEXT Byte in message
END
```

3.6 Function Codes

The function code part of a MODBUS Protocol message defines the action to be taken by the slave. Eetarp Digital meters support the following function codes:

Code	MODBUS Protocol name	Description
03	Read Holding Registers	Read the contents of read/write location(4X references)
04	Read Input Registers	Read the contents of read only location(3X references)

3.7 IEEE floating point format

The MODBUS Protocol defines 16 bit “Registers” for the data variables. A 16-bit number would prove too restrictive, for energy parameters for example, as the maximum range of a 16-bit number is 65535. However, there are a number of approaches that have been adopted to overcome this restriction. Eetarp Digital meters use two consecutive registers to represent a floating-point number, effectively expanding the range to +/- $1x10^{37}$.

The values produced by Eetarp Digital meters can be used directly without any requirement to “scale” the values, for example, the units for the voltage parameters are volts, the units for the power parameters are watts etc.

What is a floating point Number?

A floating-point number is a number with two parts, a mantissa and an exponent and is written in the form 1.234×10^5. The mantissa (1.234 in this example) must have the decimal point moved to the right with the number of places determined by the exponent (5 places in this example) i.e. $1.234x\ 10^5 = 123400$. If the exponent is negative the decimal point is moved to the left.

What is an IEEE 754 format floating-point number?

An IEEE 754 floating point number is the binary equivalent of the decimal floating-point number shown above. The major difference being that the most significant bit of the mantissa is always arranged to be 1 and is thus not needed in the representation of the number. The process by which the most significant bit is arranged to be 1 is called normalization, the mantissa is thus referred to as a “normal mantissa”. During normalization the bits in the mantissa are shifted to the left whilst the exponent is decremented until the most significant bit of the mantissa is one. In the special case where the number is zero both mantissa and exponent are zero.

The bits in an IEEE 754 format have the following significance:

Data Hi Reg, Hi Byte.	Data Hi Reg, Lo Byte.	Data Lo Reg, Hi Byte.	Data Lo Reg, Lo Byte.
SEEE EEEE	**EMMM MMMM**	**MMMM MMMM**	**MMMM MMMM**

Where:

S represents the sign bit where 1 is negative and 0 is positive

E is the 8-bit exponent with an offset of 127 i.e. an exponent of zero is represented by 127, an exponent of 1 by 128 etc.

M is the 23-bit normal mantissa. The 24th bit is always 1 and, therefore, is not stored.

Using the above format the floating point number 240.5 is represented as 43708000 hex:

Data Hi Reg, Hi Byte	Data Hi Reg, Lo Byte	Data Lo Reg, Hi Byte	Data Lo Reg, Lo Byte
43	**70**	**80**	**00**

The following example demonstrates how to convert IEEE 754 floating-point numbers from their hexadecimal form to decimal form. For this example, we will use the value for 240.5 shown above Note that the floating-point storage representation is not an intuitive format. To convert this value to decimal, the bits should be separated as specified in the floating-point number storage format table shown above.

For example:

Data Hi Reg, Hi Byte	Data Hi Reg, Lo Byte	Data Lo Reg, Hi Byte	Data Lo Reg, Lo Byte
0100 0011	**0111 0000**	**1000 0000**	**0000 0000**

From this you can determine the following information.
·The sign bit is 0, indicating a positive number.
·The exponent value is 10000110 binary or 134 decimal. Subtracting 127 from 134 leaves 7, which is the actual exponent.
·The mantissa appears as the binary number 11100001000000000000000
There is an implied binary point at the left of the mantissa that is always preceded by a 1. This bit is not stored in the hexadecimal representation of the floating-point number. Adding 1 and the binary point to the beginning of the mantissa gives the following:
1.11100001000000000000000
Now, we adjust the mantissa for the exponent. A negative exponent moves the binary point to the left. A positive exponent moves the binary point to the right. Because the exponent is 7, the mantissa is adjusted as follows:
11110000.1000000000000000
Finally, we have a binary floating-point number. Binary bits that are to the left of the binary point represent
the power of two corresponding to their position. For example, 11110000 represents (1 x 2^7) + (1 x 2^6) + (1x 2^5) + (1 x 2^4) + (0 x 2^3)+ (0 x 2^2) + (0 x 2^1)+ (0 x 2^0) = 240.
Binary bits that are to the right of the binary point also represent a power of 2 corresponding to their position. As the digits are to the right of the binary point the powers are negative. For example: .100 represents (1 x 2^{-1}) + (0 x 2^{-2})+ (0 x 2^{-3}) + … which equals 0.5.
Adding these two numbers together and making reference to the sign bit produces the number +240.5.

For each floating point value requested two MODBUS Protocol registers (four bytes) must be requested. The received order and significance of these four bytes for Eetarp Digital meters is shown below:

Data Hi Reg, Hi Byte	Data Hi Reg, Lo Byte	Data Lo Reg, Hi Byte	Data Lo Reg, Lo Byte

3.8 MODBUS Protocol Commands supported

All Eetarp Digital meters support the “Read Input Register” (3X registers), the “Read Holding Register” (4X registers) and the “Pre-set Multiple Registers” (write 4X registers) commands of the MODBUS Protocol RTU protocol. All values stored and returned are in floating point format to IEEE 754 with the most significant register first.

3.8.1 Read Input Registers

MODBUS Protocol code 04 reads the contents of the 3X registers.

Example
The following query will request ‘Volts 1’ from an instrument with node address 1:

Field Name	**Example(Hex)**
Slave Address	01
Function	04
Starting Address High	00
Starting Address Low	00
Number of Points High	00
Number of Points Low	02
Error Check Low	71
Error Check High	CB

Note: Data must be requested in register pairs i.e. the “Starting Address“ and the “Number of Points” must be even numbers to request a floating point variable. If the “Starting Address” or the “Number of

points” is odd then the query will fall in the middle of a floating point variable the product will return an error message.
The following response returns the contents of Volts 1 as 230.2. But see also “Exception Response” later.

Field Name	**Example (Hex)**
Slave Address	01
Function	04
Byte Count	04
Data, High Reg, High Byte	43
Data, High Reg, Low Byte	66
Data, Low Reg, High Byte	33
Data, Low Reg, Low Byte	34
Error Check Low	1B
Error Check High	38

3.9 Holding Registers

3.9.1 Read Holding Registers

MODBUS Protocol code 03 reads the contents of the 4X registers.
Example
The following query will request the prevailing ‘Pulse output 1 Width’:

Field Name	**Example (Hex)**
Slave Address	01
Function	03
Starting Address High	00
Starting Address Low	0C
Number of Points High	00
Number of Points Low	02
Error Check Low	04
Error Check High	08

Note: Data must be requested in register pairs i.e. the “Starting Address“ and the “Number of Points” must be even numbers to request a floating point variable. If the “Starting Address” or the “Number of points” is odd then the query will fall in the middle of a floating point variable the product will return an error message.

The following response returns the contents of Demand Time as 1, But see also “Exception Response” later.

Field Name	**Example (Hex)**
Slave Address	01
Function	03
Byte Count	04
Data, High Reg, High Byte	42
Data, High Reg, Low Byte	C8
Data, Low Reg, High Byte	00
Data, Low Reg, Low Byte	00
Error Check Low	6F
Error Check High	B5

3.9.2 Write Holding Registers

MODBUS Protocol code 10 (16 decimal) writes the contents of the 4X registers.
Example

The following query will set the Pulse output 1 Width to 60 ms, which effectively resets the Demand Time:

Field Name	**Example (Hex)**
Slave Address	01
Function	10
Starting Address High	00
Starting Address Low	0C
Number of Registers High	00
Number of Registers Low	02
Byte Count	04
Data, High Reg, High Byte	42
Data, High Reg, Low Byte	70
Data, Low Reg, High Byte	00
Data, Low Reg, Low Byte	00
Error Check Low	E6
Error Check High	59

Note: Data must be written in register pairs i.e. the “Starting Address“ and the “Number of Points” must be even numbers to write a floating point variable. If the “Starting Address” or the “Number of points” is odd then the query will fall in the middle of a floating point variable the product will return an error message. In general only one floating point value can be written per query

The following response indicates that the write has been successful. But see also “Exception Response”later.

Field Name	**Example (Hex)**
Slave Address	01
Function	10
Starting Address High	00
Starting Address Low	02
Number of Registers High	00
Number of Registers Low	02
Error Check Low	E0
Error Check High	08

Eetarp Engineering Pte Ltd
200001617K
Electrical Safety Power Quality Energy Management

EETARP Engineering Pte Ltd
11, Woodlands Close,
#08-13 Woodlands 11,
Singapore 737853

Tel: +65 63393651
Fax: +65 63393667
Email: info@eetarp.com
www.eetarp.com

Eetarp Power (M) Sdn. Bhd.
(1205228 P)
(Formerly known as EESB Engineering Sdn. Bhd.)
Electrical Safety Power Quality Energy Management

EETARP Power (M) Sdn Bhd
(formerly known as EESB Engineering Sdn Bhd)
No A-5-11, Block Allamanda
10 Boulevard, Lebuhraya Sprint
PJU 6A, 47400, Petaling Jaya
Selangor, Malaysia

Tel: +603 77293973
Fax: +603 77298973
Email: Info@eetarp.com
www.eetarp.com