



Sample EZ-ZONE® RM 32-bit Modbus RTU Packet

RM Configuration Note:

The user can specify the units of temperature measurement over serial independently from the display in addition to other Modbus settings. The default is Modbus Data Map 1. Watlow suggest use of Modbus Data Map 2 allowing more parameters accessed which the examples are based upon. The Modbus Word Order is settable between Low/High and High/Low. The default is Low/High. If set to High/Low, change the decoding accordingly. Navigate on keypad to Setup Page, Communications Menu (1 or 2), Modbus Address, Baud Rate, Parity, Display Units, Modbus Word Order, Data Map selections and Non-Volatile Save settings.

Modbus RTU Configuration Note:

The user specifies the PC/PLC timeout setting to determine when a typical response on the serial network is received. Ensure the entered value in the software driver is greater than the typical response time to prevent timeout errors. The Modbus standard does not specify a maximum response time.

Sent to RM - Read (32-bit) Analog Input 1 value

| Binary | Hex | Decimal | Purpose |
|----------|------|---------|--|
| 00000001 | 0x01 | 1 | Controller Address |
| 00000011 | 0x03 | 3 | Function Code - Read Holding Registers |
| 00000001 | 0x01 | 1 | Read Starting at register High Byte (Analog Input 1 value is contained in registers 360 & 361) |
| 11010000 | 0x68 | 104 | Read Starting at register Low Byte (Analog Input 1 value is contained in registers 360 & 361) |
| 00000000 | 0x00 | 0 | Read number of consecutive registers - High Byte (Always 0) |
| 00000010 | 0x02 | 2 | Read number of consecutive registers - Low Byte (1 to 125) |
| 01000100 | 0x44 | 68 | Low byte of CRC |
| 00101011 | 0x2B | 43 | High byte of CRC |

The CRC (also a 16-bit wide value) is sent in reverse order, low byte then high byte.

Received from PM - Read (32-bit) Analog Input 1 value

| Binary | Hex | Decimal | Purpose |
|----------|------|---------|--|
| 00000001 | 0x01 | 1 | Controller Address |
| 00000011 | 0x03 | 3 | Function Code - Read Holding Registers |
| 00000100 | 0x04 | 4 | Number of data bytes returned |
| 10010111 | 0x97 | 151 | Data High byte of 1 st register read – MSB of LSW |
| 01111101 | 0x7D | 125 | Data Low byte of 1 st register read – LSB of LSW |
| 01000010 | 0x42 | 66 | Data High byte of 2 nd register read – MSB of MSW |
| 10011100 | 0x9C | 156 | Data Low byte of 2 nd register read – LSB of MSW |
| 01110110 | 0x76 | 118 | Low byte of CRC |
| 10010110 | 0x96 | 150 | High byte of CRC |

Description of example above:

Analog Input 1 value is contained in two 16-bit registers. Register 360 contains the two lower bytes (least significant word, LSW) while register 361 contains the two higher bytes (most significant word, MSW). The 32-bit answer is an IEEE 754, 32-bit float data type.



Sample EZ-ZONE® RM 32-bit Modbus RTU Packet

The packet described is assembled and sent to the RM as one continuous stream of bits per the Modbus standard. The packet returned from the RM is decoded per the Modbus standard.

In this example, we extract 0x977D 0x429C from the packet for the answer. Changing the Modbus Word Order to High Word, Low Word we see the answer is 0x429C977D. Converting the raw data 0X429C977D to a 32-bit floating point value shows the analog input is reading +78.29588 degrees.

General steps to read registers are:

Assemble a packet to send the controller:

1. Determine controller address to read.
2. Determine function code for read. Example: Function Code (0x03) – Read Holding Registers or Function Code. (0x04) – Read Input Registers. The PM responds to both command with the same information.
3. Determine Modbus relative registers to read.
4. Convert number to hexadecimal.
5. Determine number of registers to read.
6. Enter 0x00 for number of registers to read high byte.
7. Enter number of registers to read low byte from previous step into packet. As many as 125 registers may be read with one read command. Example: Use 0x02 registers to retrieve one 32-bit value. Use 0x04 to retrieve four consecutive registers which might contain two 32-bit values or four 16-bit values or other combinations.
8. Calculate the CRC on the packet. Use the Internet to locate free programs to demonstrate how CRCs are calculated.
9. Enter the Low byte of CRC calculation into packet.
10. Enter the High byte of CRC calculation into packet.
11. Send packet as one continuous stream.
12. Wait for response from controller.
13. If no response or exception code returned, enter error routine per standard.

Process the packet received based on these steps:

14. Process packet for accuracy by comparing CRC to calculated value.
15. If CRC is correct, proceed else enter error routine per standard.
16. Parse answer from packet based on number of bytes returned.
17. Convert answers to appropriate data type. Some data is 32-bit floating point values while enumerated data is 16-bit unsigned integer values. In very few registers, the data type is 32-bit unsigned integer values. A column in the RM User's Guide provides the relative register address, the data type and whether the register is read only or read/write capable.

Please note that the difference between a Modbus RTU packet and a Modbus TCP packet is that Modbus TCP uses the same Modbus RTU packet without the CRC shown above and encapsulates the packet into an Ethernet packet using the Modbus TCP protocol.

MSB = Most significant byte

LSB = Least significant byte

MSW = Most significant word

LSW = Least significant word

CRC = Cyclic Redundancy Check



Sample EZ-ZONE® RM 32-bit Modbus RTU Packet

Sent to RM - Write (32-bit) Set Point 1 of 75.0 °F

| Binary | Hex | Decimal | Purpose |
|----------|------|---------|--|
| 00000001 | 0x01 | 1 | Controller Address |
| 00010000 | 0x10 | 16 | Function Code – Write Multiple Registers |
| 00001001 | 0x09 | 9 | Write Starting at register High Byte (Set Point 1 is register 2500 & 2501) |
| 11000100 | 0xC4 | 196 | Write Starting at register Low Byte (Set Point 1 is register 2500 & 2501) |
| 00000000 | 0x00 | 0 | Write number of consecutive registers - High Byte (Always 0) |
| 00000010 | 0x02 | 2 | Write number of consecutive registers - Low Byte (1 to 123) |
| 00000100 | 0x04 | 4 | Number of Bytes to Write (always 2 x write number) |
| 00000000 | 0x00 | 0 | Data High byte of 1 st register to write – MSB of LSW |
| 00000000 | 0x00 | 0 | Data Low byte of 1 st register to write – LSB of LSW |
| 01000010 | 0x42 | 66 | Data High byte of 2 nd register to write – MSB of MSW |
| 10010110 | 0x96 | 150 | Data Low byte of 2 nd register to write – LSB of MSW |
| 00100100 | 0x24 | 36 | Low byte of CRC |
| 10010010 | 0x92 | 146 | High byte of CRC |

The CRC (also a 16-bit wide value) is sent in reverse order, low byte then high byte.

Received from RM - Writing Set Point 1 of 75.0 °F

| Binary | Hex | Decimal | Purpose |
|----------|------|---------|--|
| 00000001 | 0x01 | 1 | Controller Address |
| 00010000 | 0x10 | 16 | Function Code – Write Multiple Registers |
| 00001001 | 0x09 | 9 | High Byte of Register 2500 decimal – Start writing at register |
| 11000100 | 0xC4 | 196 | Low Byte of Register 2500 decimal – Start writing at register |
| 00000000 | 0x00 | 0 | High Byte – number of registers written |
| 00000010 | 0x02 | 2 | Low Byte – number of registers written |
| 00000011 | 0x03 | 3 | Low byte of CRC |
| 10101001 | 0xA9 | 169 | High byte of CRC |

Description of example above:

The Set Point 1 of the RM is contained in two 16-bit registers. Register 2500 contains the two lower bytes (least significant word, LSW) while register 2501 contains the two higher bytes (most significant word, MSW). The 32-bit answer is an IEEE 754, 32-bit float data type. Floating point writes must always be accomplished using a Function Code - Multiple Write Registers command.

The packet described is assembled and sent to the RM as one continuous stream of bits per the Modbus standard. The packet returned from the RM is decoded per the Modbus standard.

In this example, we write a set point of 75.0.

0x42960000 = 75.0 degrees when read/written as a 32-bit float data type

0000 4296 is in Low Word (LSW), High Word (MSW) Order.

Register 2500 is written with LSW of 0x0000

Register 2501 is written with MSW of 0x4296



Sample EZ-ZONE® RM 32-bit Modbus RTU Packet

General steps to write registers are:

Assemble a packet to send the controller:

1. Determine controller address to write.
2. Determine function code for write. Example: Function Code (0x10) – Write Multiple Registers or Function Code. (0x06) – Write Single Register. The RM uses Write Multiple Registers for all 32-bit values.
3. Determine starting Modbus relative registers to write.
4. Convert register number in hexadecimal.
5. Enter 0x00 for number of consecutive registers to write high byte.
6. Enter 0x02 for number of consecutive registers to write low byte.
7. Enter 0x04 for the number of bytes to write – 4 bytes is for a 32-bit value.
8. Enter the data to write. For 32-bit floating values, use low word/high word order.
9. Calculate the CRC on the packet.
10. Enter the Low Byte of CRC calculation into packet.
11. Enter the High Byte of CRC calculation into packet.
12. Send packet as one continuous stream.
13. Wait for response from controller.

Process the packet received based on these steps:

14. Process packet for accuracy by comparing CRC to calculated value.
15. If CRC is correct, proceed else enter error routine per standard.
16. If no response or exception code returned, enter error routine per standard.
17. Validate response matches what was sent.



Additional details –

Some process values may be rounded off to fit into the four-character display of the RM.

Full floating-point process values are readable via Modbus. The displayed units of measurement are independent of the units of measurement sent via communications.

- **Example:** The controller may be set to display in °C on the RUI LED display but utilize °F in communication exchanged values. For Modbus RTU settings, see ‘Setup Page’, ‘Communications Menu’ to configure Modbus Address, Baud Rate, Parity, Display Units, Modbus Word Order and Data Map settings. The Display Units in the Network settings affects the communications exchanged and the Display Units in the ‘Setup Page’, ‘Global Menu’ affect the values on the RUI LED display.

All temperature parameters exchanged via communications are in °F through Modbus by default. Modbus Word Order is Low High by default. Baud Rate is 9600 with no parity by default. See the RM User’s Guide for Modbus register assignment and additional information. To prevent unintended programming changes never write values until you have read the desired register and validated it as the correct register assignment.

By default, the low register number contains the two lower bytes (least significant word); high register numbers contain the two higher bytes (most significant word) of the four-bytes for 32-bit floating-point values.

- To change the word order, set parameter Modbus Word Order ‘Low High’ to ‘High Low’ in ‘Setup Page’, ‘Communications Menu’ using the RUI keypad or EZ-ZONE Configurator.

The only function codes supported in the RM are –

- Function Code (0x03) – Read Input Registers
- Function Code (0x04) – Read Holding Registers
- Function Code (0x06) – Write Single Register
- Function Code (0x10) – Write Multiple Registers

Visit <http://www.watlow.com/literature/software.cfm> a spreadsheet showing Modbus register assignment for EZ-ZONE All Register list in Excel.

Visit <http://www.modbus.org> for a free download of the Modbus RTU and Modbus TCP implementation specifications.

Visit <http://www.watlow.com/literature/software.cfm> and locate ModbusTest for a free sample program to test communication with Modbus RTU.

Visit http://www.modbusdriver.com/shop/product_info.php?products_id=66 for a third-party Modbus software driver. The software may be purchased from ModbusDRIVER.com and is an excellent buy to get your software quickly talking to Modbus devices when writing a .NET application. The software driver includes their technical support assistance. See the documentation for converting between relative versus absolute Modbus addressing for a given driver.