Sample EHG CL and EHG SL10 16-bit Modbus RTU Packet

Sent to EHG - Read (16-bit) Process Value Controller

Binary	Hex	Decimal	Purpose
00000001	0x01	1	Controller Address
00000011	0x03	3	Function Code - Read Holding Registers
00000000	0x00	0	Read starting at register High byte (Process Value Controller is contained in register 20)
00010100	0x14	20	Read starting at register Low byte (Process Value Controller is contained in register 20)
00000000	0x00	0	Read number of consecutive registers - High byte (Always 0)
00000001	0x01	1	Read number of consecutive registers - Low byte (1 to 125)
00001110	0x0E	14	Low byte of CRC
11000100	0xC4	196	High byte of CRC

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

Received from EHG - Read Process Value Controller of 78 °F (16-bit)

Binary	Hex	Decimal	Purpose
00000001	0x01	1	Controller Address
00000011	0x03	3	Function Code - Read Holding Registers
00000010	0x02	2	Number of data bytes returned
00000000	0x00	0	High byte of 1st register read
01001110	0x4E	78	Low byte of 1st register read
01110000	0x70	112	Low byte of CRC
00111000	0x38	56	High byte of CRC

Description of example above:

Analog Input 1 value is contained in one 16-bit register. Register 20 contains the two bytes. If register 20 contains 0x4E, the 16-bit answer is +78 degrees as a signed integer data type.

The packet described is assembled and sent to the EHG as one continuous stream of bits per the Modbus standard. The packet returned from the EHG is decoded per the Modbus standard. The process values may be in degrees Fahrenheit or Celsius. Modbus relative registers 17 will read a value of 4 for °F or 5 for °C. You may write either value (4 for °F or 5 for °C) to change the units of temperature measurement.

1241 Bundy Blvd
Winona, MN 55987
Telephone (507) 494-5656

General steps to read registers are:
Assemble a packet to send the controller:

1. Determine controller address to read. Example: Address 0x01
2. Determine function code for read. Example: Function Code (0x03) – Read Holding Registers or Function Code (0x04) – Read Input Registers. The EHG responds to both command with the same information.
3. Determine Modbus relative register to read (20 decimal for Process Value Controller)
4. Convert register numbers to hexadecimal. Example: 20 decimal = 0x14
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 0x01 register to retrieve one 16-bit value. Use 0x04 to retrieve four consecutive registers which contain four 16-bit values.
8. Calculate the CRC on the packet. Use the Internet to find 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, enter into 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 into 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 16-bit signed integer values while enumerated data is 16-bit unsigned integer values. A column in the EHG User's Guide provides the relative register address, the data type and whether the register is read only or read/write capable.

1241 Bundy Blvd
Winona, MN 55987
Telephone (507) 494-5656

Sample EHG CL and EHG SL10 16-bit Modbus RTU Packet

Sent to EHG - Write (16-bit) Closed Set Point of 75 °F

Binary	Hex	Decimal	Purpose
00000001	0x01	1	Controller Address
00000110	0x06	6	Function Code – Write Single Register
00000000	0x00	0	Write register High byte (Closed Loop Set Point is register 34)
00100010	0x22	34	Write register Low byte (Closed Loop Set Point is register 34)
00000000	0x00	0	High byte of Data to write into register (new set point of 75)
01001011	0x4B	75	Low byte of Data to write into register (new set point of 75)
11110111	0xF7	247	Low byte of CRC
01101001	0x69	105	High byte of CRC

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

Received from EHG - Writing Closed Loop Set Point of 75 °F

Binary	Hex	Decimal	Purpose
00000001	0x01	1	Controller Address
00000110	0x06	6	Function Code – Write Single Register
00000000	0x00	0	Write register High byte (Closed Loop Set Point is register 34)
00100010	0x22	34	Write register Low byte (Closed Loop Set Point is register 34)
00000000	0x00	0	High byte of Data to write into register (new set point of 75)
01001011	0x4B	75	Low byte of Data to write into register (new set point of 75)
11110111	0xF7	247	Low byte of CRC
01101001	0x69	105	High byte of CRC

Description of example above:

The Closed Loop Set Point of the EHG is contained in relative register 34 as a 16-bit value. The 16-bit answer is a signed integer data type.

The packet described is assembled and sent to the EHG as one continuous stream of bits per the Modbus standard. The packet returned from the EHG is decoded per the Modbus standard. The response is an echo of the packet send after the controller has acknowledged the command for Function Code of Write Single Register. The Closed Loop Set Point value may be in degrees Fahrenheit or Celsius. Modbus relative registers 17 will read a value of 4 for °F or 5 for °C. You may write either value (4 for °F or 5 for °C) to change the units of temperature measurement.

In this example, we write a Closed Loop Set Point of 75.

0x004B = 75 degrees when read/written as a 16-bit signed integer data type

1241 Bundy Blvd
Winona, MN 55987
Telephone (507) 494-5656

General steps to write registers are:
Assemble a packet to send the controller:

1. Determine controller address to write. Example: Address 0x01
2. Determine function code for write. Example: Function Code (0x06) – Write Single Register. The EHG does not support multiple registers write.
3. Determine starting Modbus relative register to write (34 decimal for Closed Loop Set Point)
4. Convert register number to hexadecimal. Example: 34 decimal = 0x0022
5. Calculate the CRC on the packet.
6. Convert data to write in to hexadecimal. Example: 75 decimal = 0x004B
7. Enter the Low byte of CRC calculation into packet
8. Enter the High byte of CRC calculation into packet
9. Send packet as one continuous stream
10. Wait for response from controller

Process the packet received based on these steps:

11. Process packet for accuracy by comparing CRC to calculated value
12. If CRC is correct, proceed else enter into error routine per standard
13. Validate response matches what was sent.
14. If response does not match, enter into error routine per standard

1241 Bundy Blvd
Winona, MN 55987
Telephone (507) 494-5656

Additional details –

Baud Rate is 9600 with no parity by default. See the EHG 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.

The only function codes supported in the EHG are –
Function Code (0x03) – Read Input Registers
Function Code (0x04) – Read Holding Registers
Function Code (0x06) – Write Single Register

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 include their technical support assistance. See the documentation for converting between relative versus absolute Modbus addressing for a given driver.

1241 Bundy Blvd
Winona, MN 55987
Telephone (507) 494-5656