GW EIP/MODBUS... with ControlLogix

Integration guide for the GW EIP/MODBUS… into ControlLogix using Polling communication

Quick Reference Guide

QRG_923_EN_01_GW-EIP-MODBUS-Polling-ControlLogix.docx

Pos.	Qty.	Order-No.	Type-Description	Description
1	1	1062540	GW EIP/MODBUS 1E/1DB9	One RJ45 port and one D-SUB 9 port
		1062423	GW EIP/MODBUS 1E/2BD9	One RJ45 port and two D-SUB 9 ports
		1062380	GW EIP/MODBUS 2E/2DB9	Two RJ45 ports and two D-SUB 9 ports
		1062388	GW EIP/MODBUS 2E/4DB9	Two RJ45 ports and four D-SUB 9 ports
2	1			Modbus device
3	1			ControlLogix Controller
4	1		RSLogix5000	Logix Designer version 29.00

1 Overview

This document provides guidance to configure the GW EIP/MODBUS… for Modbus RTU and Modbus TCP communication with a ControlLogix PLC.

With polling, the PLC will request data on a periodic basis. It provides the ability to control the received data flow. However, it does require periodic data requests and the request rate must be fast enough to ensure that the serial port RX queues on the GW EIP/Modbus… do not overflow. The serial or socket data will return in response to the data request message.

This document assumes the user understands basic electrical concepts such as Modbus, serial and Ethernet communication, and is proficient in programming using RSLogix 5000.

2 Login

Set the IP address of the connected PC to the subnetwork of the GW EIP/MODBUS...: for example, IP = 192.168.254.10, subnetwork = 255.255.255.0.

Open a web browser and enter the IP address of the GW EIP/MODBUS... in the “Address” field. The default IP address is 192.168.254.254.

If the web server does not load, first check the IP parameters of the PC. If everything is set correctly, check to see if there are any proxy settings loaded in the web browser. The proxy setting must be set to “Load automatically” or “Deactivated” to properly establish communication.

Enter the credentials to access the web server configuration pages. The default credentials are:

- User name: Admin
- Default password: admin

Make sure you always use the latest documentation.
It can be downloaded at: www.phoenixcontact.net/products

3 GW EIP/MODBUS… Configuration

3.1 Configure serial communication

If connecting a serial Modbus device, navigate to Port Configuration under the Serial Settings. Set the serial settings of the GW EIP/MODBUS… to match the serial settings of the Modbus device. Refer to Figure 1: Serial settings.

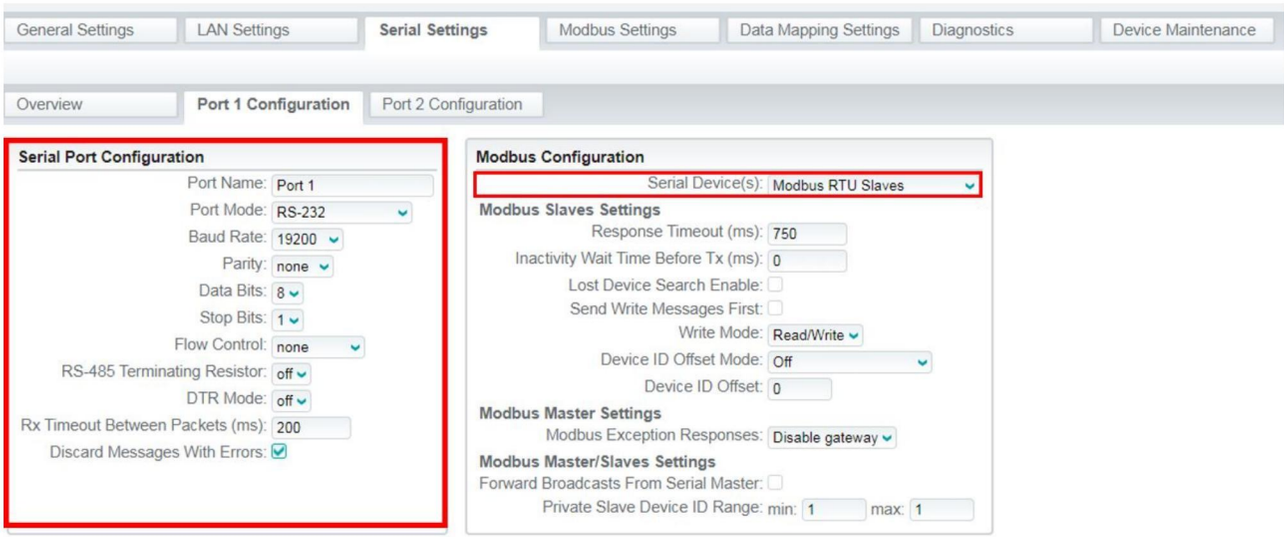

Figure 1: Serial settings

3.2 Configure Modbus TCP

If the GW EIP/MODBUS… will poll a Modbus TCP server, navigate to the Remote Modbus Addressing under Modbus Settings. Enter the Device ID, IP address of the Modbus TCP server, the Modbus TCP Port and other relevant settings. Refer to Figure 2: Remote Modbus Addressing.

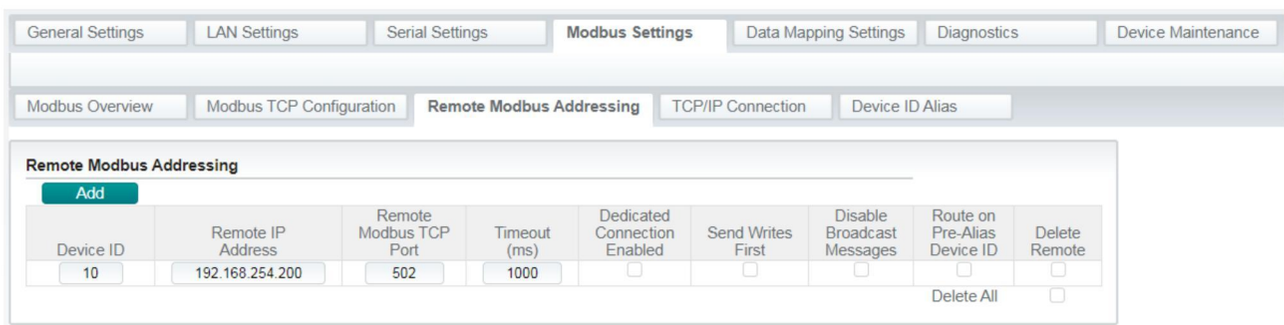

Figure 2: Remote Modbus Addressing

3.3 Shared memory configuration

Navigate to the Shared Memory tab under the Data Mapping Settings tab. The Shared Memory configuration options are broken into three sections, Shared Memory Configuration, Shared Holding Registers and Shared Coils. Refer to Figure 3: Shared memory configuration.

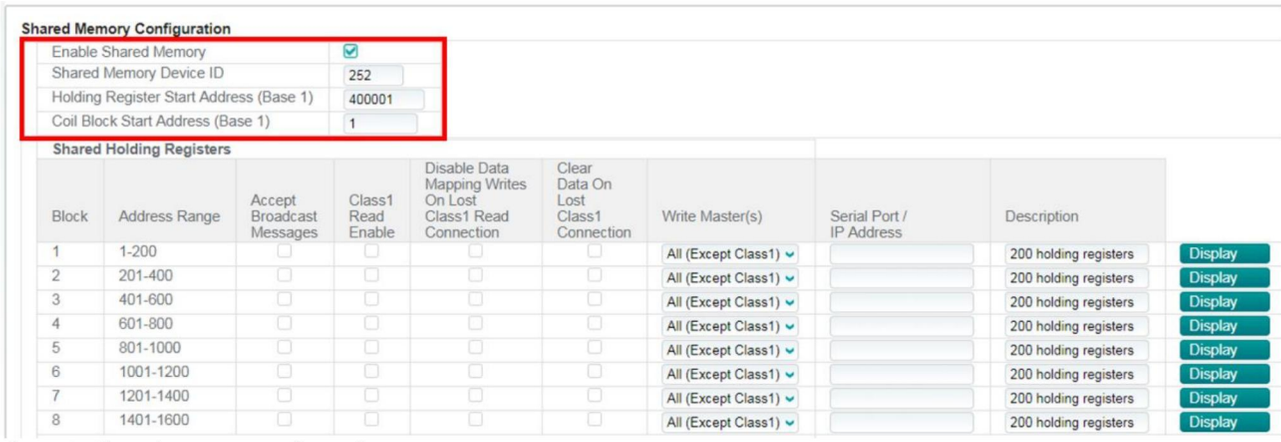

Figure 3: Shared memory configuration

The Shared Memory Configuration section is to enable the Shared Memory, define the Slave ID and define the starting address for the holding registers and coils.

- Enable Shared Memory: Click the Enable Shared Memory check box.
- Shared Memory Device ID: Enter a Modbus device ID for the GW EIP/MODBUS.... The device ID must be unique within the Modbus network. The valid range of values is 1 to 255 although Modbus org recommends using values between 1-247.
- Holding Register Start Address (Base 1): Enter the starting address of the GW EIP/MODBUS… holding registers. The GW EIP/MODBUS… supports extended holding register addressing. Therefore, the valid starting address range is 40001 to 463935.
- Coil Block Start Address (Base 1): Enter the starting address of the GW EIP/MODBUS... coils. The GW EIP/MODBUS… supports extended coil addressing. Therefore, the valid starting address range is 1 to 64255.

This example will use the settings in Table 1: Shared Memory Configuration.

Table 1: Shared Memory Configuration

Shared Memory Device ID	252
Holding Register Start Address	400001
Coil Block Start Address	1

4 Modbus Master

This section describes polling the GW EIP/MODBUS… shared memory with a Modbus master.

4.1 Connection

If connecting a Modbus RTU or Modbus ASCII PLC, refer to section 3.1 Configure serial communication. If connecting a Modbus TCP PLC enter the IP address (default: 192.168.254.254) of the GW EIP/MODBUS… and Modbus TCP port 502.

4.2 Read from Shared Memory

Reference the settings from Table 1: Shared Memory Configuration for the Device ID. Use function code 03 Read Holding Registers to read the Shared Holding Registers and function code 01 Read Coils to read the Shared Coils.

Following the Shared Memory Configuration set previously, the Ethernet/IP controller writes to Block 1. Therefore, the Modbus PLC reads from holding registers 40201…40400 and coils 321….640.

4.3 Write from Shared Memory

Reference the settings from Table 1: Shared Memory Configuration. for the Device ID. Use function code 06 Write Single Holding Register to write to the Shared Holding Registers and function code 05 Write Single Coil to write to the Shared Coils.

Following the Shared Memory Configuration set previously, the Ethernet/IP controller reads from Block 2. Therefore, the Modbus PLC writes to holding registers 40001…40200 and coils 1….320.

5 Modbus Slave

This section describes configuring a Modbus-to-Modbus command using the GW EIP/MODBUS… to read and write to a Modbus slave.

5.1 Modbus-to-Modbus read from slave, write to Shared Memory

In the web manager of the GW EIP/MODBUS… navigate to the Modbus-to-Modbus tab under Data Mapping Settings. Create a Modbus-to-Modbus configuration to read from the slave device and then write to shared memory. Refer to Table 2: Modbus-to-Modbus Configuration.

Table 2: Modbus-to-Modbus Configuration

Modbus (Read)	Device ID	Device ID of the Modbus slave to read from
	Function Code	Modbus function code
	Address (base 1)	Starting address on Modbus slave device
	Length (Regs/Coils)	Number of registers or coils to read from slave device
	Poll Rate (ms)	Rate to read from slave device
Modbus (Write)	Device ID	Device ID of the shared memory to write to reference Table 1: Shared Memory Configuration
	Function code	Corresponding function code to write to shared memory
	Address (base 1)	Starting address of shared memory

Figure 4: Modbus-to-Modbus configuration - read from slave, write to shared memory, depicts an example configuration with the Modbus slave device at Device ID 1 and the shared memory at Device ID 252. Select the active check box to the left of the configuration to activate the configuration. The GW /MODBUS… will then begin polling Modbus slave Device ID 1, register 40001 through 40004 and writing the values to the Shared Memory Block 1 40001 through 40004.

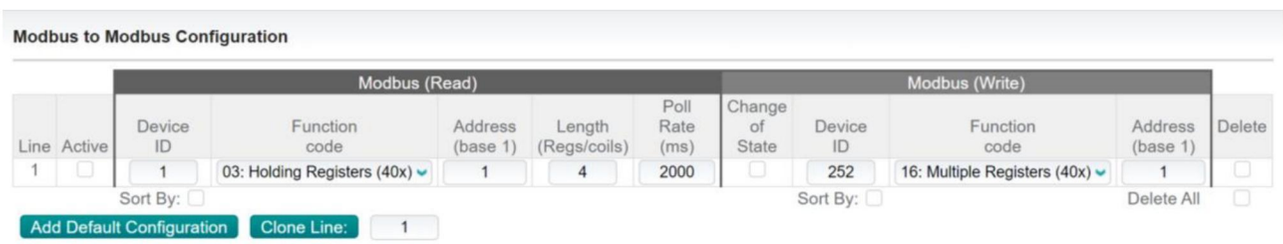

Figure 4: Modbus-to-Modbus configuration - read from slave, write to shared memory

To verify proper communication, navigate to the block of shared memory configured for the read in the web manager for the GW EIP/MODBUS.... It can be found under the Modbus Diagnostics tab in the shared memory tab.

5.2 Modbus-to-Modbus read from shared memory, write to slave

In the web manager of the GW EIP/MODBUS… navigate to the Modbus-to-Modbus tab under Data Mapping Settings. Create a Modbus-to-Modbus configuration to read from the shared memory and then write to a slave device. Refer to Table 3: Modbus-to-Modbus configuration settings.

Table 3: Modbus-to-Modbus configuration settings

Modbus (Read)	Device ID	Device ID of the shared memory reference Table 1: Shared Memory Configuration
	Function Code	Modbus function code
	Address (base 1)	Starting address of the shared memory location
	Length (Regs/Coils)	Number of registers/coils to read from shared memory
	Poll Rate (ms)	Rate of read from shared memory
Modbus (Write)	Device ID	Device ID of the Modbus slave to write to
	Function code	Desired format of write to Modbus slave
	Address (base 1)	Starting address of Modbus slave

To read data from shared memory and write the data to a Modbus slave device, navigate to the Modbus-to-Modbus tab under the Data Mapping Settings in the GW EIP/MODBUS…. Ensure the configuration is active by checking the “Active” box. Refer to Figure 5: Modbus to Modbus configuration.

Figure 5: Modbus to Modbus configuration – read from shared memory, write to slave, depicts an example configuration with the shared memory Device ID 252 and a Modbus slave device at Device ID 250. Select the active check box to the left of the configuration if you would like to activate the configuration. The GW EIP/MODBUS… will then begin polling the GW EIP/MODBUS… shared memory register 40001 through 40004 and writing the values to the Modbus slave device ID 250 from registers 40001 through 40004.

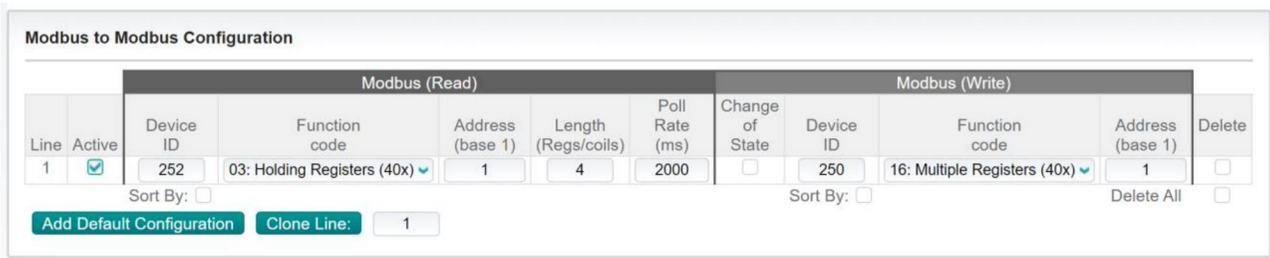

Figure 5: Modbus to Modbus configuration

To verify proper communication, navigate to the block of shared memory configured for the read in the web manager for the GW EIP/MODBUS…. It can be found under the Modbus Diagnostics tab in the shared memory tab.

6 ControlLogix Configuration

6.1 Read Data

Open RSLogix5000 and create a new program. In the Controller Organizer navigate to “Main Routine” under the “Tasks” folder. Refer to Figure 6: Main Routine.

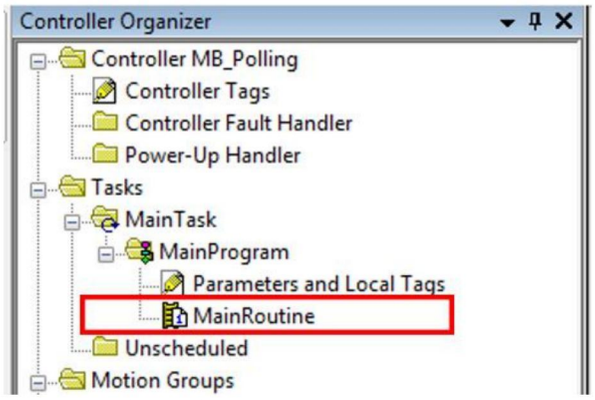

Figure 6: Main Routine

In the main routine, add a normally open contact and a MSG block to rung 0. The MSG block is in the input/output folder. Refer to Figure 7: Read rung.

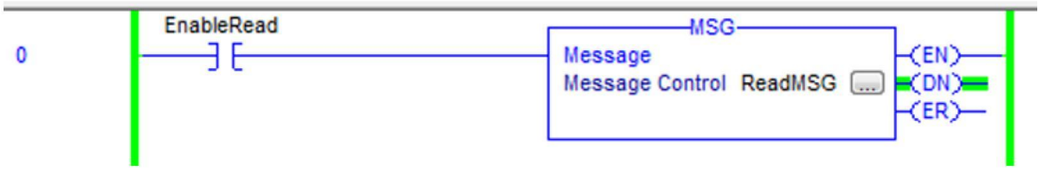

Figure 7: Read rung

In the controller organizer, right-click on “Controller Tags” and create a new tag for the normally open contact. Refer to Figure 8: Create new tag.

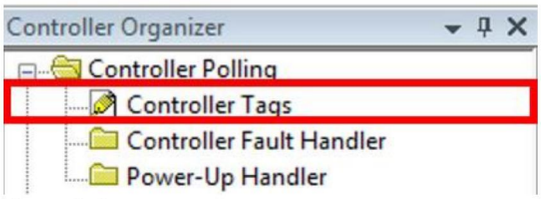

Figure 8: Create new tag

In the pop-up window, configure the tag for the normally open contact. Ensure it is of type BOOL. Refer to Figure 9: Normally open contact for read. Click OK. Next, double-click on the question mark above the normally open contact in the Main Routine window and set it to the tag just created.

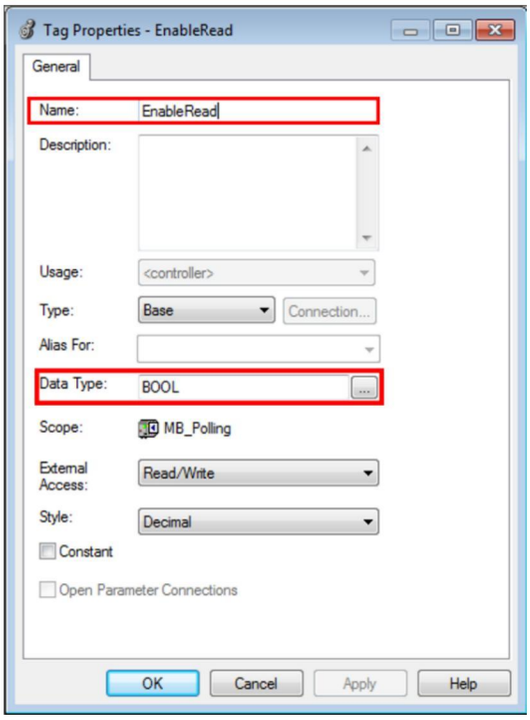

Figure 9: Normally open contact for read

To communicate using polling with the GW EIP/MODBUS… create new data types. In the controller organizer, right-click on “Data Types” and select create new data type. Refer to Figure 10: Create new data type.

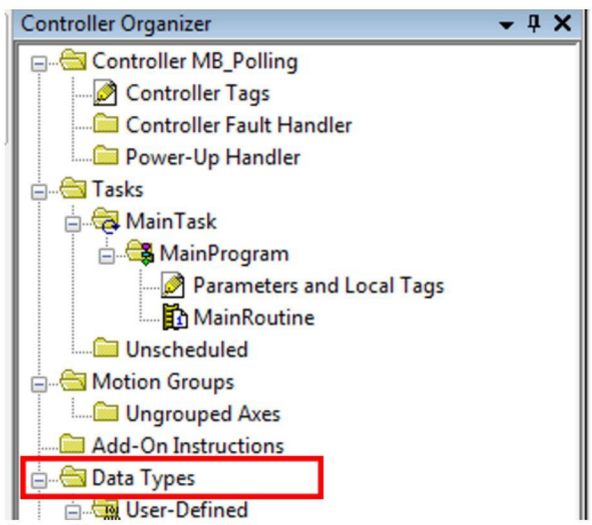

Figure 10: Create new data type

Create a data type titled “RdHoldingRegsReq”. Within this data type, create members titled “startAddress” and “numRegisters”. Both members should be INT types. When completed, select OK. Refer to Figure 11: Read request data type.

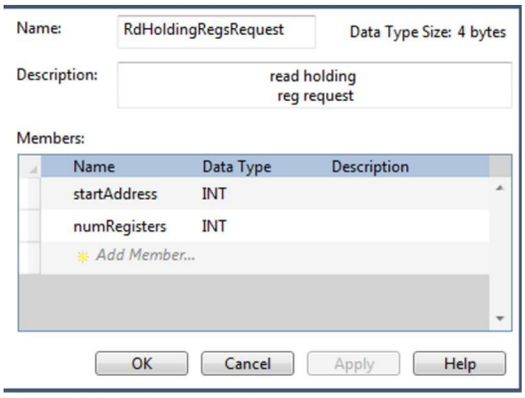

Figure 11: Read request data type

Next, create a data type titled “RdHoldingRegsResponse”. Within this data type create a member titled “registerData” as an INT with a length of 125. When finished, select OK. Refer to Figure 12: Read response data type.

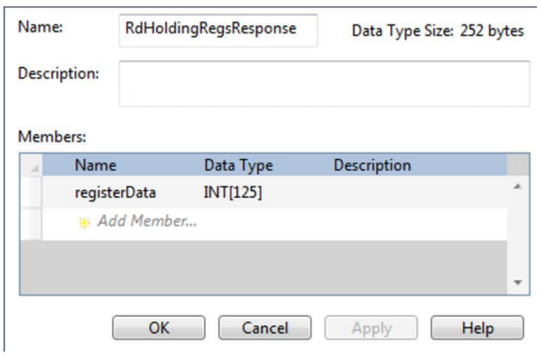

Figure 12: Read response data type

After creating both new data types, right-click on controller tags to create new tags. Create a tag for the read request data. Ensure that the tag’s data type is also for read requests. Click OK to create the tag. Refer to Figure 13: Read request tag.

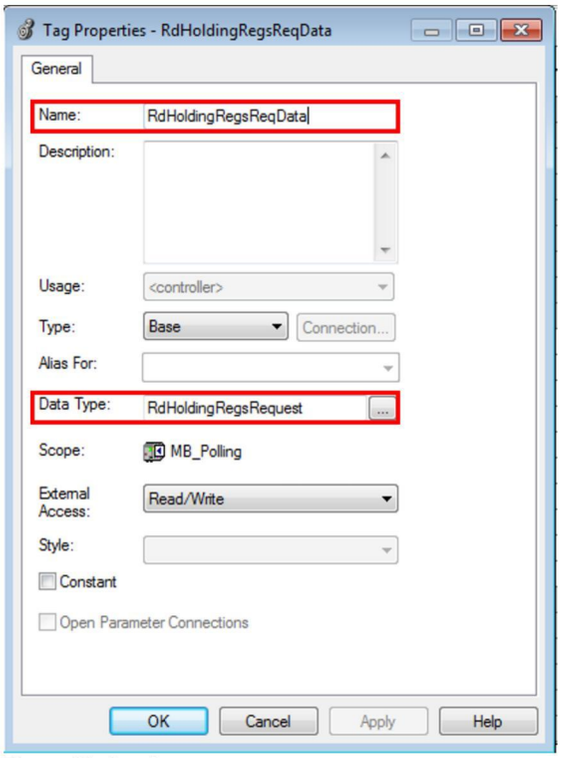

Figure 13: Read request tag

Repeat this process to create a tag for the read response data. Ensure the tag's data type is also for read responses. Click OK to create the tag. Refer to Figure 14: Read response tag.

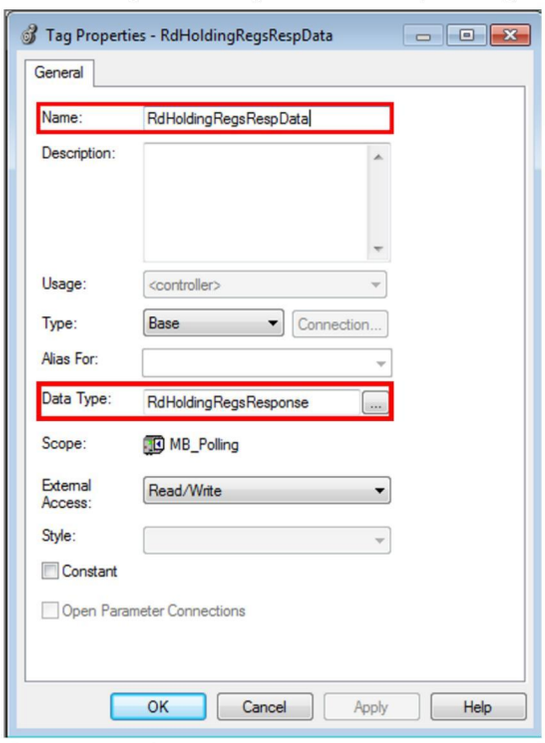

Figure 14: Read response tag

Navigate to the “Main Routine” and double-click on the question marks in the MSG block on rung 0. Configure a new tag for the overall read message. Configure the internal settings of the MSG block for the read. Refer to Table 4: Modbus Read of Holding Registers for a description of the parameters needed for the MSG block.

Table 4: Modbus Read of Holding Registers

Parameter	Value	Description
Service Code	4e	Directly corresponds to Modbus function code: Read Input Registers (Function code 3)
Instance	252	Device ID of shared memory (Valid range: 1-255).
Class	44	Modbus Object
Attribute	0	Not used
Source Element	RdHoldingRegsReqData	Tag of structure type “RdHoldingRegsRequest”
Source Length	4	Length in bytes of RdHoldingRegsReqData
Destination	RdHoldingRegsRespData	Tag of structure type “RdHoldingRegsResponse”

Ensure the “Message type” is “CIP Generic” and the service type is “Custom”. Refer to Figure 15: Read MSG configuration.

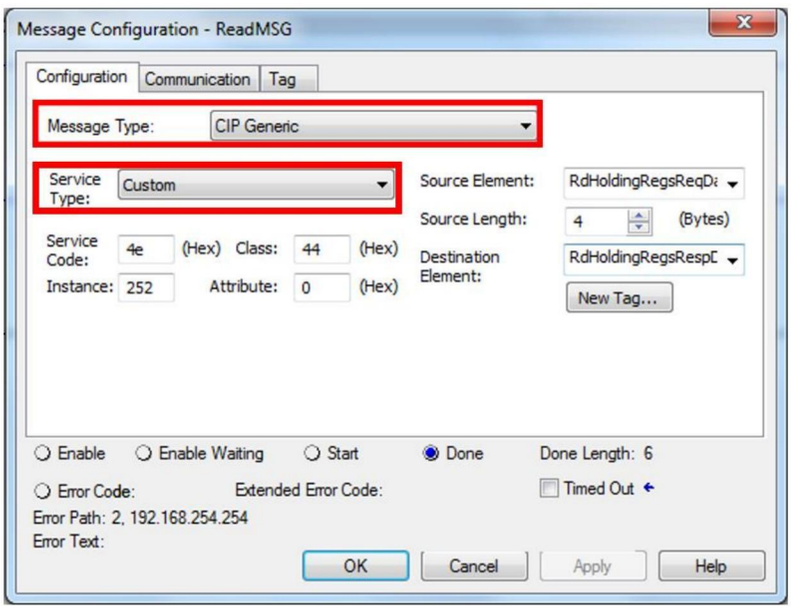

Figure 15: Read MSG configuration

In the Message Configuration pop-up, navigate to the “Communication” tab. In the communication tab, set the path of communication. The path is configured as follows: “2,xxx.xxx.xxx.xxx” where the 2 corresponds to an ethernet connection and the IP address refers to the address of the GW EIP/MODBUS…. Refer to Figure 16: Read MSG path configuration.

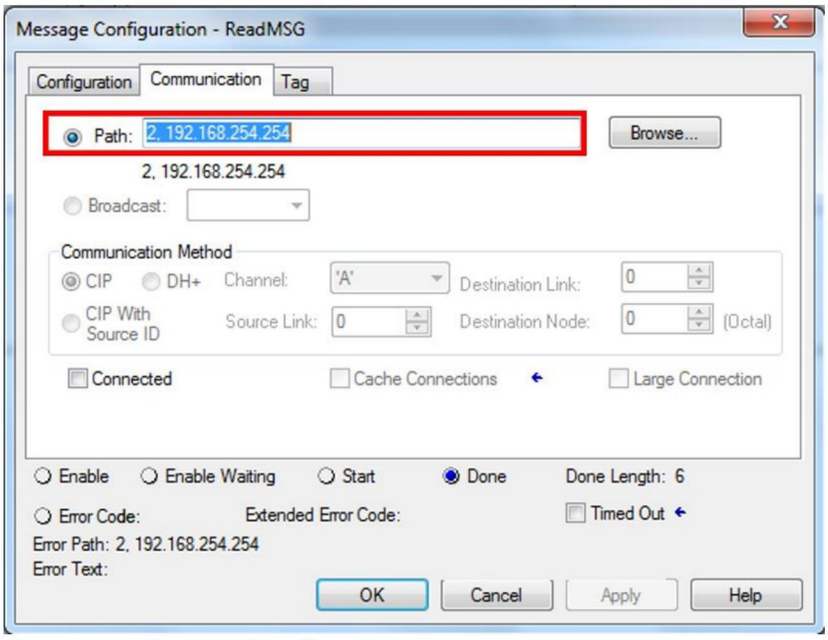

Figure 16: Read MSG path configuration

In the toolbar of RSLogix5000, set the path of the controller. Click on the “Who Active” icon and select the controller from the list. Refer to Figure 17: Set path.

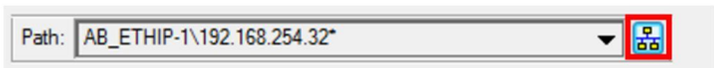

Figure 17: Set path

In the toolbar, set controller status to run mode. Refer to Figure 18: Controller in Run Mode.

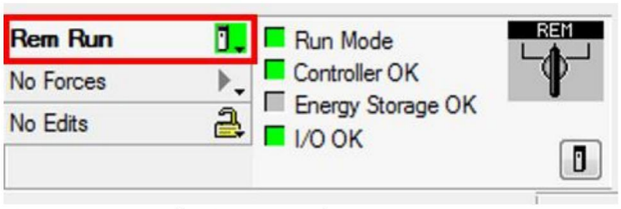

Figure 18: Controller in Run Mode

Navigate to the “Main Routine”. Right click on the normally open contact for the read and select “toggle bit”. This reads the data from the shared memory in GW EIP/MODBUS…. The shared memory in the GW EIP/MODBUS… is in the Shared Memory tab under “Modbus Diagnostics”. Refer to Figure 19: Read data in shared memory.

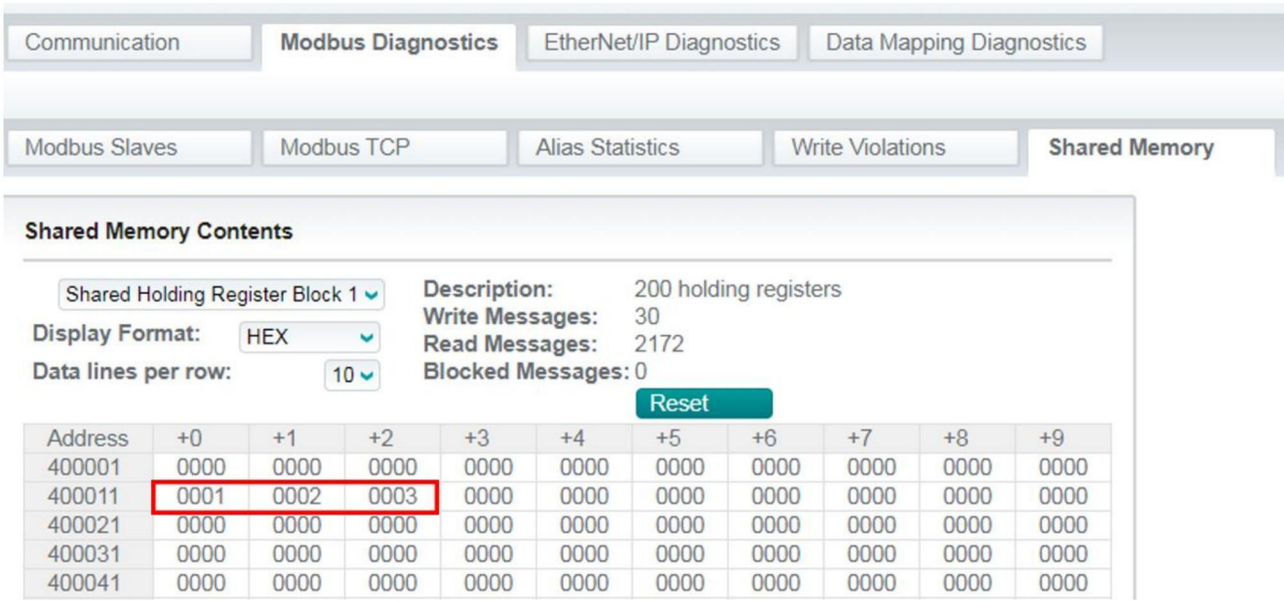

Figure 19: Read data in shared memory

After toggling the normally open contact for the read, the controller reads the block of shared memory. The user must specify how many registers to read and where the data starts. Refer to Figure 20: Read data in the controller.

RdHoldingRegsReqData	{...}	{...}		RdHoldingRegsRequest
RdHoldingRegsReqData.numRegisters	3		Decimal	INT
RdHoldingRegsReqData.startAddress	10		Decimal	INT
RdHoldingRegsRespData	{...}	{...}		RdHoldingRegsResponse
RdHoldingRegsRespData.registerData	{...}	{...}	Decimal	INT[125]
RdHoldingRegsRespData.registerData[0]	1		Decimal	INT
RdHoldingRegsRespData.registerData[1]	2		Decimal	INT
RdHoldingRegsRespData.registerData[2]	3		Decimal	INT
RdHoldingRegsRespData.registerData[3]	0		Decimal	INT
RdHoldingRegsRespData.registerData[4]	0		Decimal	INT

Figure 20: Read data in the controller

6.2 Write Data

Open RSLogix5000 and create a new program. In the Controller Organizer navigate to “Main Routine” under the “Tasks” folder. Refer to Figure 6: Main Routine.

In the main routine, add a normally open contact and a MSG block to rung 1. The MSG block is in the input/output folder. Refer to Figure 21: Write rung.

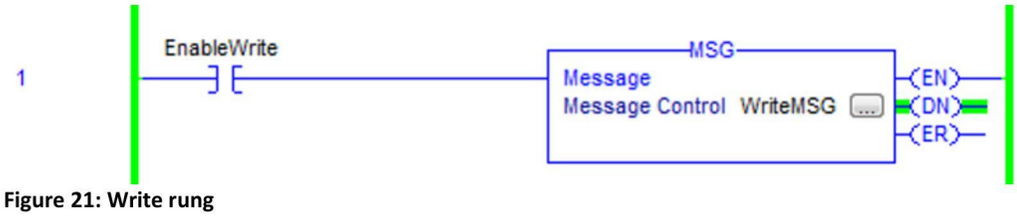

Figure 21: Write rung

In the controller organizer, right-click on “Controller Tags” and create a new tag for the normally open contact. Refer to Figure 8: Create new tag.

In the pop-up window, configure the tag for the normally open contact. Ensure it is of type Boolean. Refer to Figure 22: Normally open contact for write. Click OK. Next, double-click on the question mark above the normally open contact in the Main Routine window and set it to the tag just created.

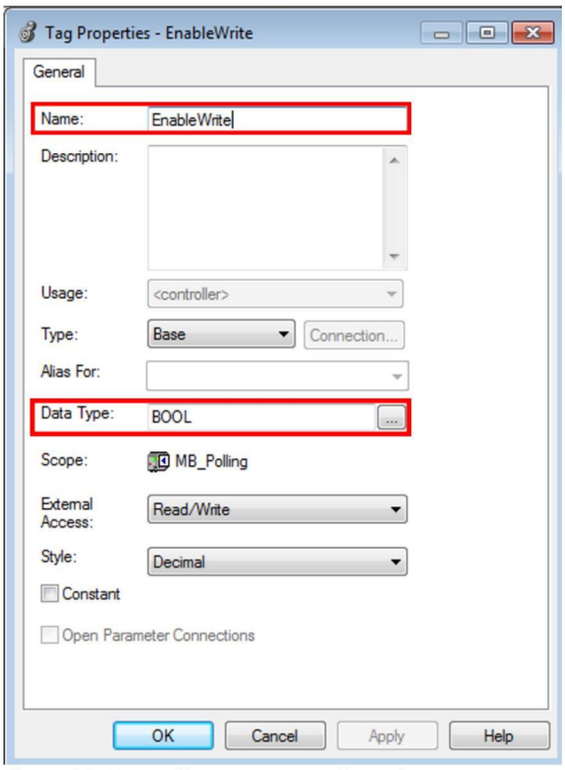

Figure 22: Normally open contact for write

To communicate using polling with the GW EIP/MODBUS… create new data types. In the controller organizer, right-click on “Data Types” and select create new data type. Refer to Figure 6: Create new data type.

Create a data type titled “WrHoldingRegsRequest”. Within this data type, create members titled “startAddresses”, “numRegs”, and “registerData”. All members should be INT types, while the “registerData” member should be an INT of size 123. When completed, select OK. Refer to Figure 23: Write holding register request data type.

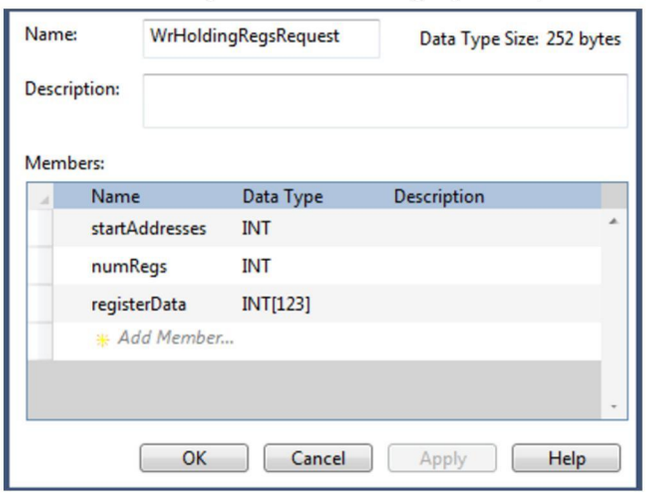

Figure 23: Write holding register request data type

Next, create a data type titled “WrHoldingRegsResponse”. Within this data type create a member titled “startAddress” and a new member titled “numRegs”. Both members should be of data type INT. When finished, select OK. Refer to Figure 24: Write holding register response data type.

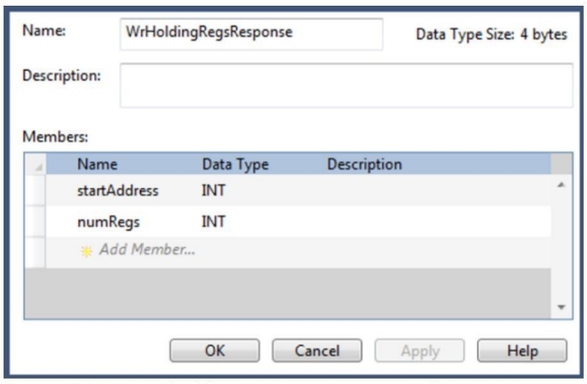

Figure 24: Write holding register response data type

After creating both new data types, right-click on controller tags to create new tags. Create a tag for the write request data. Ensure that the tag’s data type is also for write requests. Click OK to create the tag. Refer to Figure 25: Write holding registers request tag.

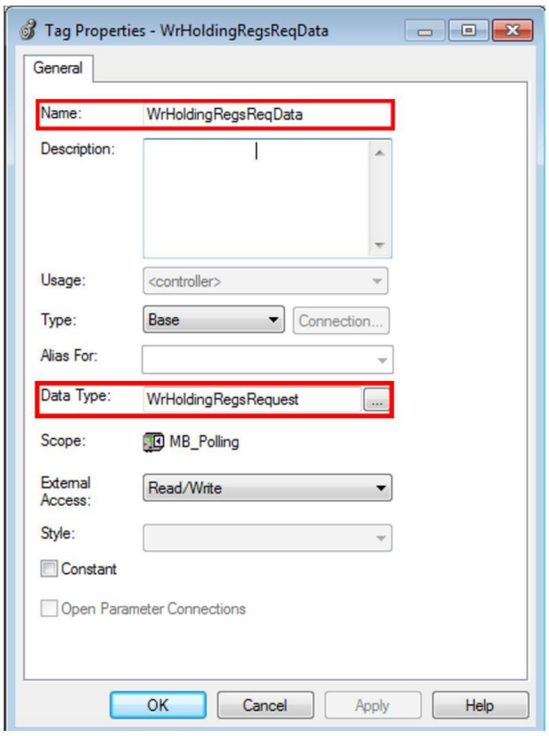

Figure 25: Write holding registers request tag

Create a tag for the write response data. Ensure the tag’s data type is also for write responses. Click OK to create the tag. Refer to Figure 26: Write holding register response tag.

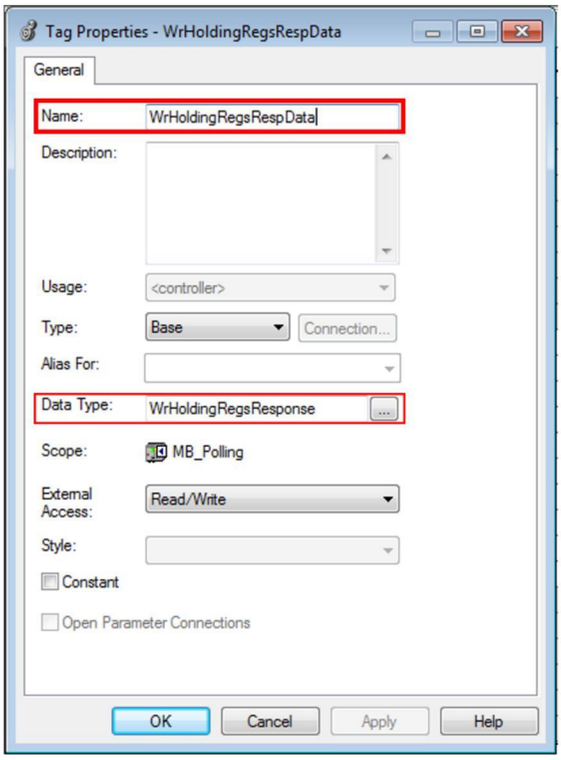

Figure 26: Write holding register response tag

Navigate to the “Main Routine” and double-click on the question marks in the MSG block on rung 1. Configure a new tag for the overall write message. Configure the internal settings of the MSG block for the write. Refer to Table 5: Modbus write of holding registers for a description of the parameters needed for the write MSG block.

Table 5: Modbus write of holding registers

Parameter	Value	Description
Service Code	50	Directly corresponds to Modbus function code: Write Holding Registers (Function code 16)
Instance	252	Device ID of shared memory (Valid range: 1-255).
Class	44	Modbus Object
Attribute	0	Not used
Source Element	WrHoldingRegsReqData	Tag of structure type “WrHoldingRegsRequest”
Source Length	250	Length in bytes of WrHoldingRegsReqData
Destination	WrHoldingRegsRespData	Tag of structure type “WrHoldingRegsResponse”

Ensure the “Message type” is “CIP Generic” and the service type is “Custom”. Refer to Figure 27: Write MSG configuration.

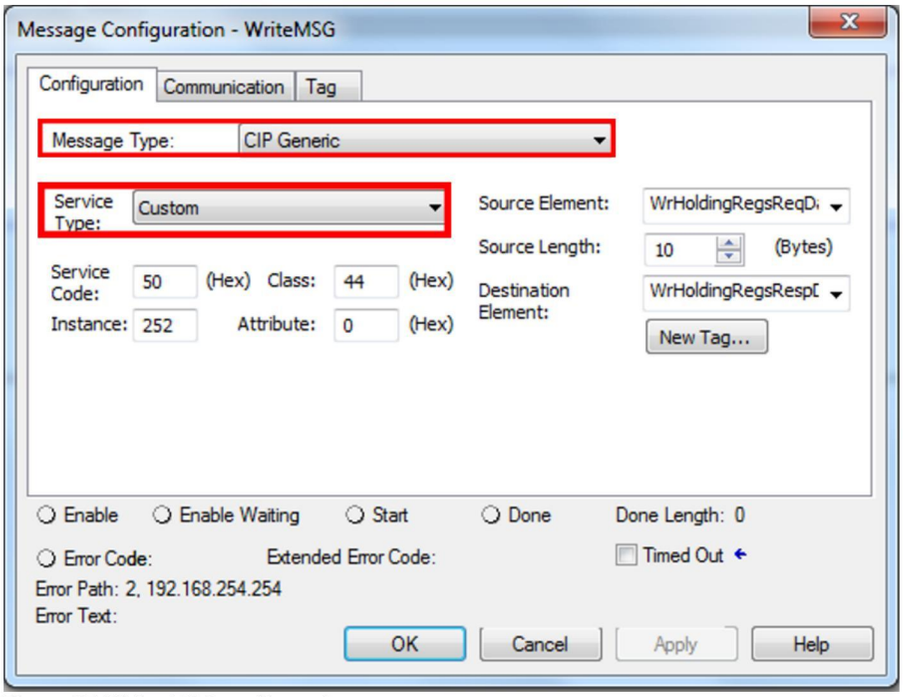

Figure 27: Write MSG configuration

In the Message Configuration pop-up, navigate to the “Communication” tab. In the communication tab, set the path of communication. The path is configured as follows: “2,xxx.xxx.xxx.xxx” where the 2 corresponds to an ethernet connection and the IP address refers to the address of the GW EIP/MODBUS…. Refer to Figure 28: Write MSG communication path.

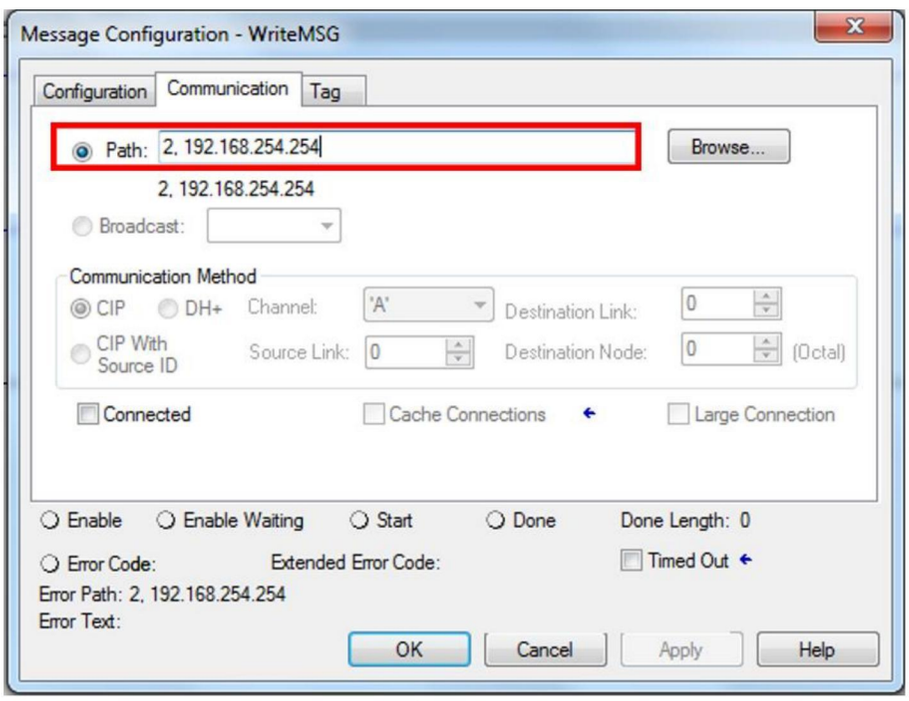

Figure 28: Write MSG communication path

In the toolbar of RSLogix5000, set the path of the controller. Click on the “Who Active” icon and select the controller from the list. Refer to Figure 17: Set path.

In the toolbar, set controller status to run mode. Refer to Figure 18: Controller in Run Mode.

In RSLogix5000, navigate to the controller tags and edit the “WrHoldingRegsReqData” tag. Specify the number of registers to write and add data to that number of registers. The example writes to 3 registers and puts “6” in the first register, “5” in the second register, and “4” in the third register. Refer to Figure 29: Write data in controller.

WrHoldingRegsReqData	{...}	{...}		WrHoldingRegsRequest
WrHoldingRegsReqData.numRegs	3		Decimal	INT
WrHoldingRegsReqData.registerData	{...}	{...}	Decimal	INT[123]
WrHoldingRegsReqData.registerData[0]	6		Decimal	INT
WrHoldingRegsReqData.registerData[1]	5		Decimal	INT
WrHoldingRegsReqData.registerData[2]	4		Decimal	INT
WrHoldingRegsReqData.registerData[3]	0		Decimal	INT
WrHoldingRegsReqData.registerData[4]	0		Decimal	INT
WrHoldingRegsReqData.registerData[5]	0		Decimal	INT

Figure 29: Write data in controller

Next navigate to the “Main Routine” and right-click on the normally open contact for the write and select “toggle bit” to trigger the write. To confirm the action, navigate to the web manager for the GW EIP/MODBUS…. In the web manager navigate to the block of shared memory under the “Modbus Diagnostics” tab. Refer to Figure 30: Data written to shared memory.

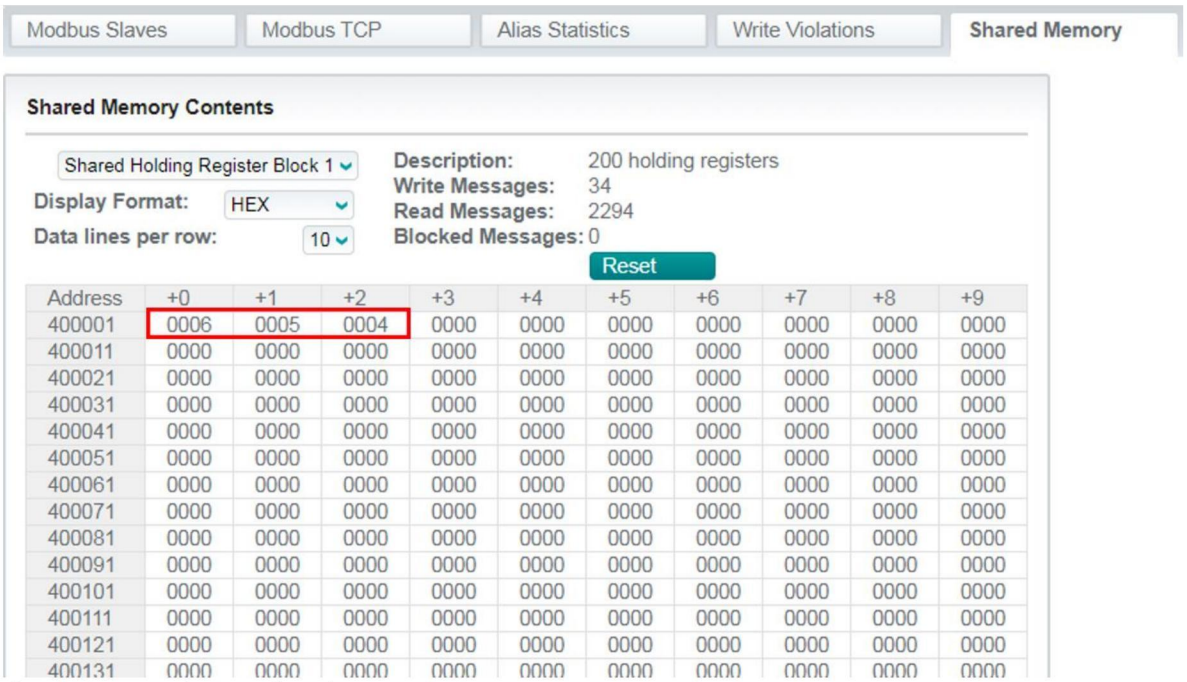

Figure 30: Data written to shared memory