DeviceMaster® EIP-MOD

EtherNet®/IP - Modbus®

User Guide

Trademark Notices

Comtrol, DeviceMaster, and PortVision are trademarks owned by Pepperl + Fuchs, GmbH and are used under license by Pepperl + Fuchs Comtrol, Inc.

Concept is a trademark of Schneider Electric.

Modbus is a registered trademark of Schneider Electric.

ControlLogix, PLC-5 and Rockwell Automation are registered trademarks of Rockwell Automation.

MicroLogix, RSLinx, RSLogix 5, RSLogix 500, RSLogix 5000 and SLC are trademarks of Rockwell Automation.

PLC is a registered trademark of Allen-Bradley Company, Inc.

Ethernet is a registered trademark of Digital Equipment Corporation, Intel, and Xerox Corporation.

Windows is a registered trademark of Microsoft Corporation in the United States and/or other countries.

Other product names mentioned herein may be trademarks and/or registered trademarks of their respective owners.

First Edition, April 3, 2019

Pepperl + Fuchs Comtrol, Inc. makes no representations or warranties with regard to the contents of this document or to the suitability of the Pepperl + Fuchs Comtrol, Inc. product for any particular purpose. Specifications subject to change without notice. Some software or features may not be available at the time of publication. Contact your reseller for current product information.

Document Number: 2000664 Rev A

Table of Contents

Chapter 1. Introduction

This document describes how to configure the DeviceMaster EIP-MOD for the EtherNet/IP-Modbus after basic DeviceMaster EIP-MOD installation and configuration procedures.

See *Locating the Latest Software and Documentation* on Page 19 to locate the latest firmware, documentation, and tools.

1.1. EtherNet/IP-Modbus Gateway Overview

The DeviceMaster EtherNet/IP-Modbus gateway provides the following functionality.

EtherNet/IP interface support:

- Supports all fully CIP compliant controllers including Rockwell ControlLogix, CompactLogix, SoftLogix, Schneider Electric, Omron, and GE PLCs, OPC Servers, SCADA systems, HMIs and many more.
- Supports PCCC based PLCs such as MicroLogix, SLC 5/05 and PLC-5.
- EtherNet/IP Interface specifics:
  - Maximum CIP sessions = 128
  - Maximum CIP input (to PLC) connections = 128
  - Maximum CIP output (from PLC) connections = 16
  - Write-to-Tag/File and Read-from-Tag/File support
  - CIP Modbus Object support. Custom service codes support additional Modbus function codes.
- EtherNet/IP controllers can communicate to:
- Shared Memory
- A wide variety of Modbus slave devices:
  - Local serial Modbus/RTU and Modbus/ASCII slave devices
  - Modbus/TCP slaves
  - Remote serial Modbus/RTU and Modbus/ASCII devices connected to other gateways
- A wide variety of Modbus masters via Shared Memory:
  - Modbus/TCP masters
  - Modbus/RTU and Modbus/ASCII masters over Ethernet TCP connections
  - Serial Modbus/RTU and Modbus/ASCII masters
- Other EtherNet/IP controllers via Shared Memory
  - Via Class1 connections to Shared memory
  - CIP Modbus Object messages

  - Via Read-from-Tag/File and Write-to-File/Tag configurations

The most extensive Modbus functionality in the industry:

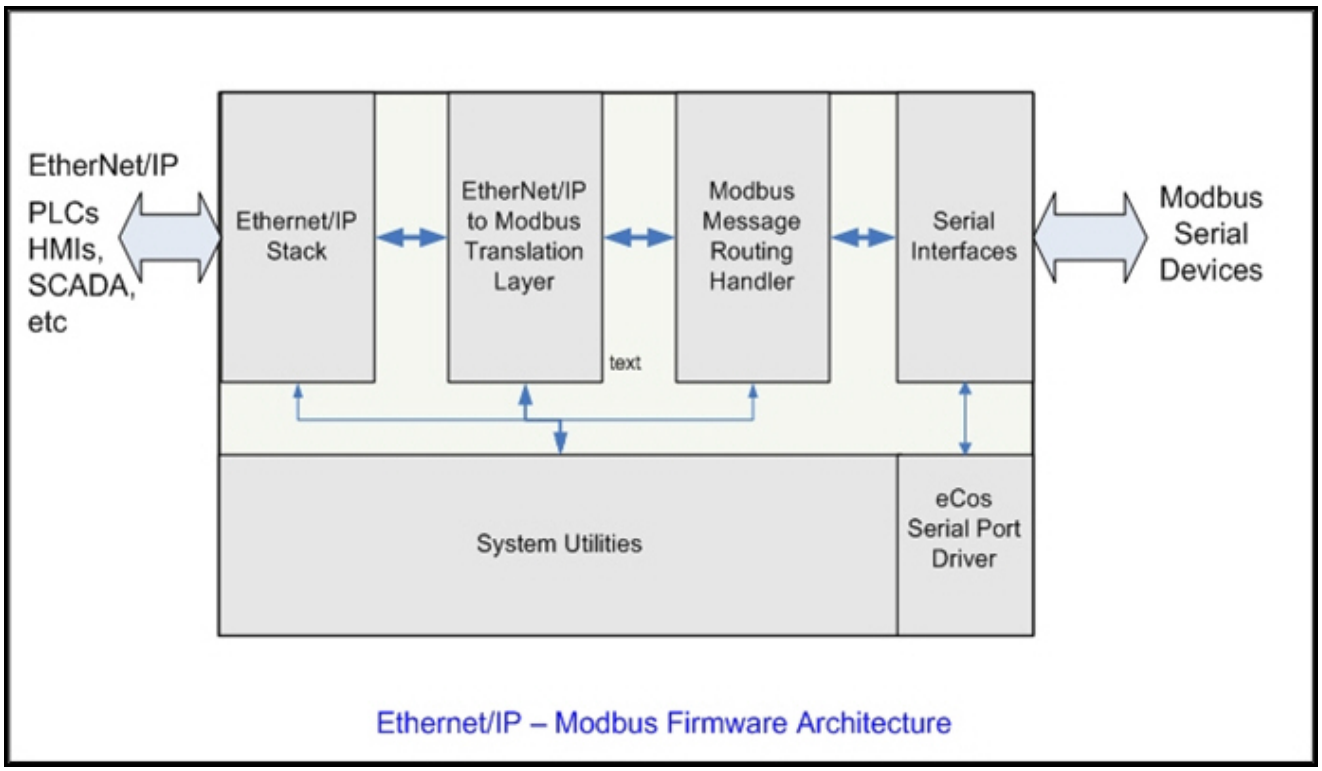
- Up to 96 Modbus/TCP connections
- Support up to eight unique Modbus TCP socket port numbers. (i.e. 502, 503, 505, etc.)
- Multiple Modbus/RTU and/or Modbus/ASCII over Ethernet TCP connections
- Local serial Modbus/RTU and Modbus/ASCII slave devices
- Maximum sized Modbus messages
- Serial Modbus/RTU and Modbus/ASCII masters
- Shared Memory that can be configured to provide communication interfaces between a vast array of Modbus slaves, Modbus masters and EtherNet/IP controllers.
- Modbus Alias functionality
- Remote Modbus device connectivity
- Modbus slave to Modbus slave communication
- Private Modbus serial bus connectivity – A serial port connected to a serial master and one or more serial slave devices

A Data Mapping process that provides:

- EtherNet/IP Tag/File to Modbus connectivity - to either Modbus device(s) or Shared memory
- Modbus to EtherNet/IP Tag/File connectivity - from either Modbus device(s) or Shared memory
- Modbus to Modbus connectivity
  - Can read from multiple Modbus devices into Shared Memory.
  - Can write from Shared memory to multiple Modbus devices.
  - Can read directly from one Modbus slave device to another Modbus slave device anywhere on the Modbus network. These devices can be of any supported type, local or remote.
  - Can convert from Modbus register type to another automatically.
- A Data Mapping verification web page that checks for possible Modbus write conflicts, incorrect configuration and write violations.
- A Shared Memory map web page that displays the Shared Memory read and write memory blocks as defined utilized by the Data Mapping.

1.2. System Architecture

The DeviceMaster EtherNet/IP-Modbus firmware utilizes the following architecture.

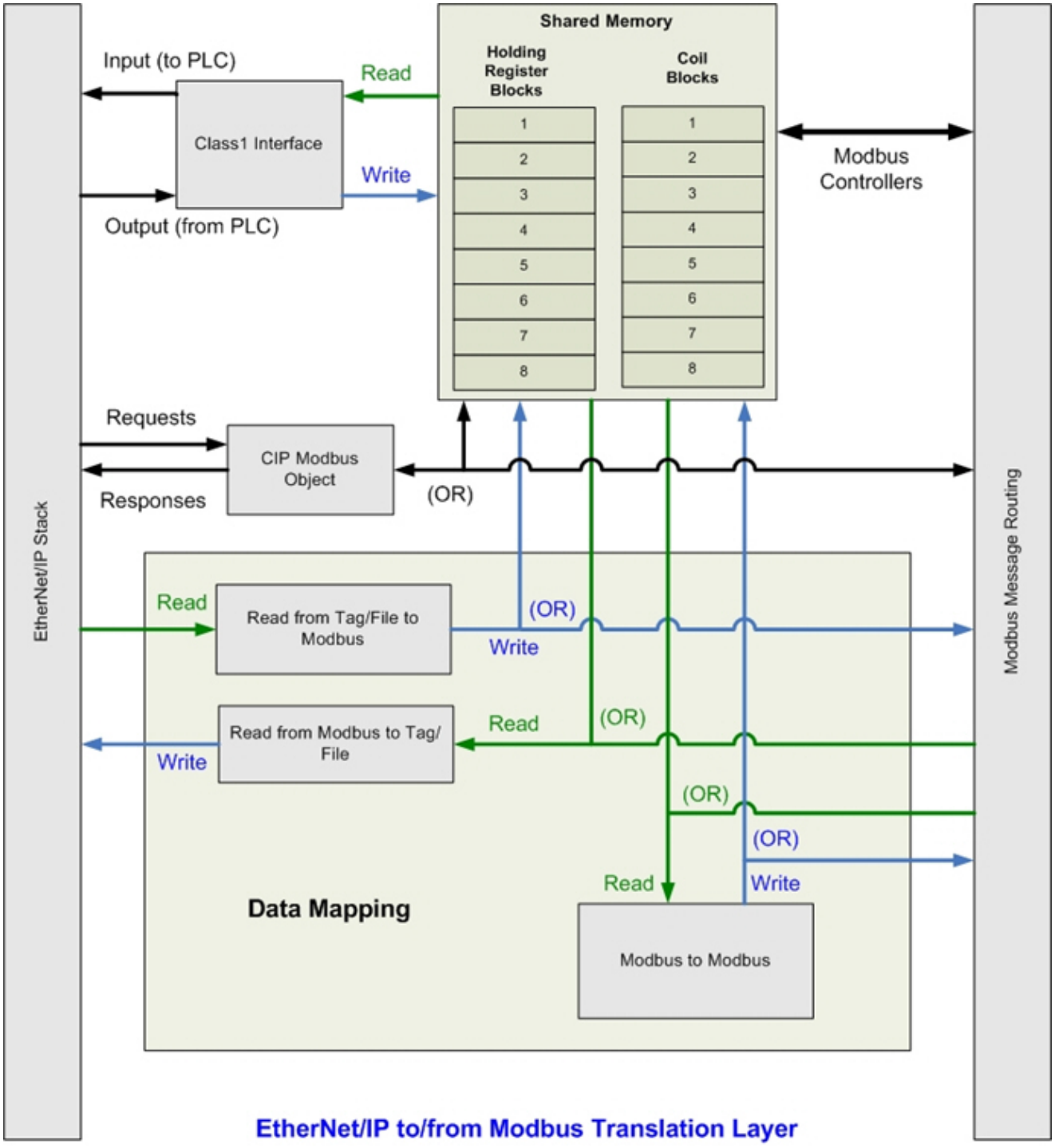


Ethernet/IP – Modbus Firmware Architecture

The firmware architecture is designed to take full advantage of a broad range of EtherNet/IP and Modbus routing functionality supported by the DeviceMaster EtherNet/IP-Modbus gateway.

1.3. EtherNet/IP to Modbus Translation Layer Architecture

The following image illustrates the EtherNet/IP to/from Modbus translation layer.



1.4. Translation Layer Connections

The following connections are possible through the translation layer.

1.4.1. Connection Methods Table

This table lists the possible connections and the methods available for a wide variety of EtherNet/IP and Modbus interfaces.

		Write (To)								
		EtherNet/IP PLC(s) via Class1 Connection	EtherNet/IP PLC(s) via Tag/File	EtherNet/IP PLC(s) via Modbus Object	Modbus/TCP Master(s)	Modbus RTU/ASCII Serial Master(s)	Modbus RTU/ASCII Over Ethernet TCP Master(s)	Modbus RTU/ASCII Serial Slave(s)	Modbus/TCP Slave(s)	Remote Modbus RTU/ASCII Serial Slave(s)
Read (From)	EtherNet/IP PLC(s) via Class1 Connection	2	3	2	2	2	2	3	3	3
	EtherNet/IP PLC(s) via Tag/File	3	3	3	3	3	3	3,4	3,4	3,4
	EtherNet/IP PLC(s) via Modbus Object	2	3	2	2	2	2	1,3	1,3	1,3
	Modbus/TCP Master(s)	2	3	2	2	2	2	1,3	1,3	1,3
	Modbus RTU/ASCII Serial Master(s)	2	3	2	2	2	2	1,3	1,3	1,3
	Modbus RTU/ASCII Over Ethernet TCP Master(s)	2	3	2	2	2	2	1,3	1,3	1,3
	Modbus RTU/ASCII Serial Slave(s)	3	3,4	1,3	1,3	1,3	1,3	3,4	3,4	3,4
	Modbus/TCP Slave(s)	3	3,4	1,3	1,3	1,3	1,3	3,4	3,4	3,4
	Remote Modbus RTU/ASCII Serial Slave(s)	3	3,4	1,3	1,3	1,3	1,3	3,4	3,4	3,4

Where the method is:

1 = via direct message to Modbus slave device

2 = via Shared Memory

3 = via Shared Memory and use of Data Mapping

4 = via use of only Data Mapping

1.4.2. EtherNet/IP Controllers Class1 to/from Shared Memory

The following provides information about EtherNet/IP controllers Class1 to and from shared memory.

- Combined with the Data Mapping Modbus to Modbus functionality:
  - Local serial Modbus/RTU and Modbus/ASCII slave devices.
  - Remote serial Modbus/RTU and Modbus/ASCII slave devices (via additional gateways).
  - Remote Modbus/TCP slaves.
- Modbus Controllers:
  - Modbus/TCP masters
  - Modbus/RTU and Modbus/ASCII over Ethernet TCP masters
  - Serial Modbus/RTU and Modbus/ASCII masters
- Other EtherNet/IP controllers:
  - Utilizing Class1 connections
  - Combined with the Data Mapping Read Tag/File Write Modbus functionality
  - Combined with the Data Mapping Read Modbus Write Tag/File functionality

1.4.3. EtherNet/IP Controllers via Data Mapping to/from Tag/File

The following provides information about EtherNet/IP controllers via data mapping to and from Tag/File.

- Directly to/from Modbus slave devices:
  - Local serial Modbus/RTU and Modbus/ASCII slave devices.
  - Remote serial Modbus/RTU and Modbus/ASCII slave devices (via additional gateways).
  - Modbus/TCP slaves.
- Through Shared Memory:
  - Combined with the Data Mapping Modbus to Modbus functionality:
    - Local serial Modbus/RTU and Modbus/ASCII slave devices.
    - Remote serial Modbus/RTU and Modbus/ASCII slave devices (via additional gateways).
    - Remote Modbus/TCP slaves.
  - Modbus Masters:
    - Modbus/TCP masters
    - Modbus/RTU and Modbus/ASCII over Ethernet TCP masters
    - Serial Modbus/RTU and Modbus/ASCII masters
  - Other EtherNet/IP controllers:
    - Utilizing Class1 connection(s)
    - Combined with the Data Mapping Read Tag/File Write Modbus functionality
    - Combined with the Data Mapping Read Modbus Write Tag/File functionality

1.4.4. EtherNet/IP Controllers CIP Modbus Object via MSG Instructions

The following provides information about EtherNet/IP controller CIP Modbus object via MSG instructions.

- Directly to/from Modbus slave devices:
  - Local serial Modbus/RTU and Modbus/ASCII slave devices.
  - Remote serial Modbus/RTU and Modbus/ASCII slave devices (via additional gateways).
  - Modbus/TCP slaves.
- Through Shared Memory:
  - Combined with the Data Mapping Modbus to Modbus functionality:
    - Local serial Modbus/RTU and Modbus/ASCII slave devices.
    - Remote serial Modbus/RTU and Modbus/ASCII slave devices (via additional gateways).
    - Remote Modbus/TCP slaves.
  - Modbus Masters:
    - Modbus/TCP masters
    - Modbus/RTU and Modbus/ASCII over Ethernet TCP masters
    - Serial Modbus/RTU and Modbus/ASCII masters
  - Other EtherNet/IP controllers:
    - Utilizing Class1 connection(s)
    - Combined with the Data Mapping Read Tag/File Write Modbus functionality
    - Combined with the Data Mapping Read Modbus Write Tag/File functionality

1.5. Data Type Definitions

The following list defines the available data types.

Data Type	Definition
USINT	Unsigned short integer (8-bits)
UINT	Unsigned integer (16-bit)
UDINT	Unsigned double integer (32-bits)
INT	Signed integer (16-bits)
DINT	Signed double integer (32-bits)
BYTE	Bit string (8-bits)
WORD	Bit string (16-bits)
DWORD	Bit string (32-bits)
STRING	Character string (1-byte per character)

1.5.1. Terms and Definitions

This section uses the following terms and definitions.

Term	Definition
Class 1	Otherwise called *implicit* messaging, is a method of communication between EtherNet/IP controllers and devices that: • Uses Ethernet UDP messages. • Is cyclic in nature. Input and/or output data is exchanged between the controllers and devices at regular time intervals.
Class 3	Otherwise called explicit messaging, is a method of communication between EtherNet/IP controllers and devices that: • Uses Ethernet TCP/IP messages. • By itself is not cyclic in nature. The controller and devices must send individual messages to each other.
EtherNet/IP	An Ethernet based industrial communication protocol utilized to communicate between controllers, often times PLCS, and devices.
Ethernet TCP/IP	Standard Ethernet communications protocol utilizing socket communication interfaces that guarantees delivery to the intended device.
Ethernet UDP/IP	Standard Ethernet communications protocol utilizing socket communication interfaces that does not guarantee delivery. The data may or may get to the intended device.
Multicast	Multicast addressing involves Ethernet devices sending messages to each other using a multicast address. Multicast addressing: • Uses a specified IP address range designated for multicast communication. • Allows either one or multiple devices to receive the same messages.
Point-to-Point	Point-to-Point, otherwise called unicast, addressing involves Ethernet devices sending messages directly to each other using their own IP addresses. Messages are sent to only one device.
Master (or Client) Mode	The method of operation when a PLC, a gateway, or an application is operating as a *Master* or the message originator.
Slave (or Server) Mode	The method of operation when a PLC, a gateway, or an application is operating as a *Slave* or the message receiver.
Device ID	The Modbus device identification number. Device IDs include: • 0 = Broadcast • 1-247 = Standard device IDs • 248-255 = Reserved device IDs (generally used for vendor specific gateway functions)
Device ID Offset	An offset applied at the slave serial port interface that changes the message device ID range to match the serial device(s) device ID range.
Received Device ID	The original device ID received in the Modbus message from a Modbus master.
Alias Device ID	The device ID that the original received ID is changed to when an Alias Device ID is configured.
Local Slave Device	A local slave device is one that is connected directly to a serial port on the gateway.
Remote Slave Device	A remote slave device is either a slave Modbus/TCP device or a serial slave device attached to another gateway and accessed as a remote Modbus/TCP device.

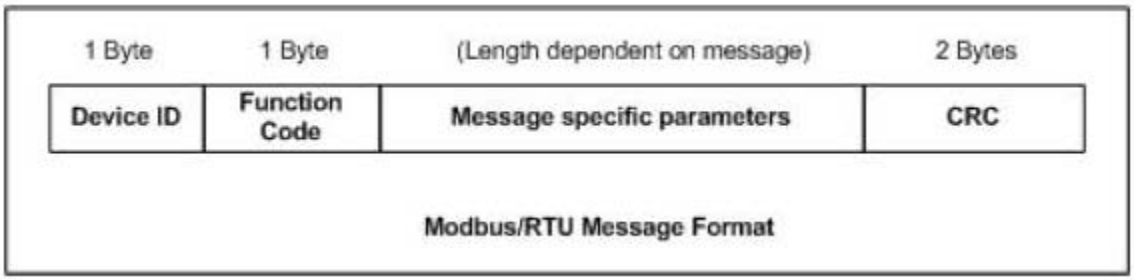
Term	Definition (Continued)
Modbus/RTU	The standard Modbus messages, in hexadecimal format, that are typically transmitted over serial lines but can also be transmitted over other communication methods such as wireless or Ethernet TCP/IP socket connections. ***Note:*** *Modbus/RTU over Ethernet TCP/IP is not the same as Modbus TCP.*
Modbus/ASCII	An ASCII, or character based, form of Modbus. The base message is the same as Modbus/RTU, but the format is somewhat different.
Modbus/TCP	An Ethernet based form of Modbus communication. The base message is the same as Modbus/RTU, but a special Modbus header is included for packet identification and routing purposes.
Polling	The process where a PLC or Application requests data on a continual basis. In this operation the Master sends the request messages while the Slave responds to the messages.
Ethernet TCP/IP	A form of Ethernet connectivity that provides a level of guaranteed delivery and data verification. This is used for many upper layer protocols such as Modbus/TCP and can be also used for transferring Modbus/RTU and Modbus/ASCII. messages.

1.6. What is Modbus?

This subsection discusses:

1.6.1. Modbus/RTU

Modbus/RTU is native Modbus in hexadecimal format. These are the base Modbus messages that contain simple read and write requests. The format is as follows:



Modbus/RTU Message Format

Where:

- The terms **Master** or **Client** are used to identify the sender of the message.
- The terms **Slave** or **Server** are used to identify the devices responding to the message.

Modbus/RTU is used primarily for:

- Serial port connectivity

  RS-485 is the most common serial mode, but RS-232 and RS-422 are also widely used. Commonly used by both Master and Slave devices.
- Ethernet TCP/IP socket connections

  This is not the same as *Modbus/TCP* on Page 18, but does provide a very simple method of interfacing to remote devices. It is used by many applications and some OPC servers.

  Note: *This communication method typically is not supported by PLCs.*

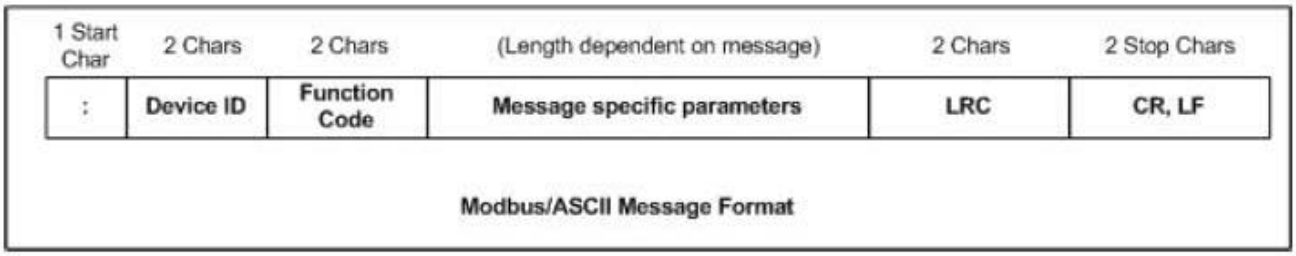
1.6.2. Modbus/ASCII

Modbus/ASCII is native Modbus in ASCII format. This protocol is used primarily by legacy devices and is no longer supported as widely as Modbus/RTU.

Like Modbus/RTU, Modbus/ASCII contains the base Modbus messages that contain simple read and write requests. The differences between Modbus/ASCII and Modbus/RTU are:

1. The message data is sent in ASCII format, so the message length is twice as long. It requires two ASCII characters for each byte of data.
2. An 8-bit LRC is attached to verify the message instead of a 16-bit CRC. The LRC is also transmitted in ASCII format.
3. There are defined starting and ending characters to determine a Modbus/ASCII messages.

The format is as follows:



Modbus/ASCII Message Format

Where:

- The terms **Master** or **Client** are used to identify the sender of the message.
- The terms **Slave** or **Server** are used to identify the devices responding to the message.

Modbus/ASCII is used primarily for:

- Serial port connectivity

  RS-485 is the most common serial mode, but RS-232 and RS-422 are also used. Used primarily by legacy Slave devices.

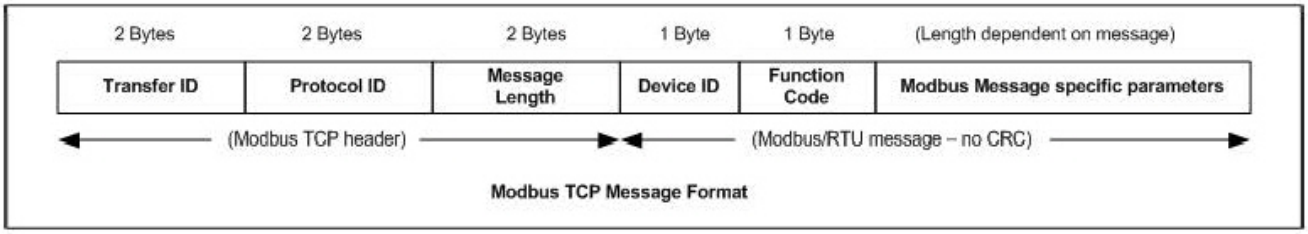
- Ethernet TCP/IP socket connections

  This is not the same as *Modbus/TCP*, but does provide a very simple method of interfacing to remote devices. It is used by some applications and some OPC servers.

  Note: *This communication method typically is not supported by PLCs.*

1.6.3. Modbus/TCP

Modbus/TCP is an Ethernet network based protocol that contains a Modbus/RTU message, with the exception of the 2-byte CRC. The Modbus/TCP message contains a header with information designed to provide message identification and routing information. The format is as follows:



Modbus TCP Message Format

Where:

- The terms **Master** or **Client** are used to identify the sender of the message.
- The terms **Slave** or **Server** are used to identify the devices responding to the message.
- Modbus/TCP messages are typically sent to and received on a defined Ethernet TCP/IP socket of 502.
- Modbus/TCP implementations provide more capability, but also require more processing than simpler Modbus/RTU implementations.

Modbus/TCP is used for connecting advanced Ethernet based devices, such as PLCs, HMIs, SCADA Systems, and most OPC Servers to:

- Other Ethernet devices supporting Modbus/TCP.
- Remote serial Modbus/RTU and/or Modbus/ASCII devices through gateways (such as the DeviceMaster EIP-MOD running the EtherNet/IP-Modbus or Modbus/TCP applications).
- Remote serial or Ethernet TCP/IP ASCII devices (barcode scanners, printers, RFID readers, visions systems, etc) through a gateway (such as the DeviceMaster EIP-MOD running the Modbus/TCP application).

1.7. Locating the Latest Software and Documentation

You can use the links in this table to check for updated software or documentation.

Software and Documentation		FTP
PortVision DX (Windows)	Use PortVision DX to manage Pepperl+Fuchs Comtrol, Inc. Ethernet-attached devices to: • Scan the network for attached devices • View networked devices in real-time • Access product-specific network settings configurations • Assign IP addresses and network settings to one or multiple devices • Upload the latest firmware or Bootloader • Save and load configuration files • Access DeviceMaster EIP-MOD configuration web pages • Access Telnet/SSH sessions • Remotely reboot devices • Download technical documentation • Enable event logging to assist in monitoring and troubleshooting • Create shortcuts to quickly access your favorite applications • Organize devices into folders and create multiple views • Enter notes about a folder or device	
EtherNet/IP-Modbus Firmware	This is the application that may or may not have been loaded on the DeviceMaster EIP-MOD depending on the model that was ordered. You may need to use PortVision DX to load this firmware.	
EtherNet/IP-Modbus Hardware Installation and Configuration Guide	This contains hardware installation, configuration information, and connector information. This includes using PortVision DX to configure the IP address and if necessary, how to update the firmware.	
EtherNet/IP-Modbus User Guide *(this guide)*	This *User Guide* contains detailed information about the EtherNet/IP-Modbus (application) firmware, which includes additional information about the web configuration interface.	

Chapter 2. EtherNet/IP Interface Profile (ControlLogix)

This section describes the EtherNet/IP objects included in the ControlLogix EtherNet/IP interface and supported by the DeviceMaster EIP-MOD.

2.1. Modbus Object (44 Hex, 1 Instance)

The Modbus Object provides an interface to the data and functions within a Modbus device or the Shared memory on the DeviceMaster EtherNet/IP-Modbus gateway. It utilizes CIP object specific services to implement Modbus functions through an explicit CIP message.

Note: *The instance number is used to define the device or unit Id of the Modbus device or Shared memory. There is no support for broadcast Modbus messages, which have a device Id of zero.*

2.1.1. Class Attributes

The following table displays the Class Attributes for Modbus Object (44 hex, 1 Instance).

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Revision	UINT	1	Get
2	Max Instance	UINT	255 (maximum Modbus device Id)	Get
3	Num Instances	UINT	255 (maximum Modbus device Id)	Get
6	Maximum Number Class Attribute	UINT	7	Get
7	Maximum Number Instance Attribute	UINT	0	Get

2.1.2. Instance Attributes

There are no Instance Attributes.

2.1.3. Common Services

This table provides the Common Services for Modbus Object (44 hex, 1 instance).

Service Code	Implemented in Class	Implemented in Instance	Service Name
0E hex	Yes	Yes	Get_Attribute_Single

2.1.4. Object Specific Services (Instance Specific)

These services are provided to implement the supported Modbus functions. All Modbus Object services, as specified in *Volume 7: Integration of Modbus Devices into the CIP Architecture*, are supported as well as several additional services.

Service Code	Service Name	Service Description
4B hex	Read Discrete Inputs	Reads one or more contiguous discrete input(s).
4C hex	Read Coils	Reads one or more contiguous coil(s).
4D hex	Read Input Registers	Reads one or more contiguous input register(s).
4E hex	Read Holding Registers	Reads one or more contiguous holding register(s).
4F hex	Write Coils	Writes one or more contiguous coil(s).
50 hex	Write Holding Registers	Writes one or more contiguous holding register(s).
51 hex	Modbus Pass through	Provides encapsulation of any public or private Modbus function.
70 hex **	Write Single Coil	Writes a single coil.
71 hex **	Write Single Register	Writes a single holding register.
72 hex **	Mask Write Register	Perform a mask write operation on a single holding register.
73 hex **	Read/Write Holding Registers	Writes one or more holding register(s) and reads one or more holding register(s).
** = Vendor specific services provided to support additional Modbus functions.		

Note: *In order to use these services, the Enable Vendor Specific Modbus Object Services option on the EtherNet/IP Stack Configuration web page must be selected.*

2.1.4.0.1. Read Discrete Inputs Service (4B Hex)

This service reads one or more discrete inputs from the Modbus Discrete Inputs table. This service results in the DeviceMaster EIP-MOD issuing a Read Discrete Inputs function (function code 0x02) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3 and/or MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

2.1.4.0.2. Read Coils Service (4C Hex)

This service reads one or more coils from the Modbus Coils table. This service results in the DeviceMaster EIP-MOD issuing a Read Coils function (function code 0x01) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3 and/or MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

2.1.4.0.3. Read Input Registers Service (4D Hex)

This service reads one or more input registers from the Modbus Input Register table. This service results in the DeviceMaster EIP-MOD issuing a Read Input Registers function (function code 0x04) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3 and/or MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

2.1.4.0.4. Read Holding Registers Service (4E Hex)

This service reads one or more holding registers from the Modbus Holding Register table. This service results in the DeviceMaster EIP-MOD issuing a Read Holding Registers function (function code 0x03) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3 and/or MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

2.1.4.0.5. Write Coils Service (4F Hex)

This service writes one or more coils to the Modbus Coils table. This service results in the DeviceMaster EIP-MOD issuing a Write Multiple Coils function (function code 0x0F) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3 and/or MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

2.1.4.0.6. Write Holding Registers Service (50 Hex)

This service writes one or more holding registers to the Modbus Holding Registers table. This service results in the DeviceMaster EIP-MOD issuing a Write Multiple Registers function (function code 0x10) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3 and/or MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

2.1.4.0.7. Modbus Passthrough Service (51 Hex)

The Modbus Passthrough service provides a way for a client to initiate a specific Modbus function to a target Modbus device. The Modbus request and response are encapsulated in the CIP request and response service data fields with no modification. The DeviceMaster EIP-MOD does not attempt to perform any byte endian conversion on the bytes in the data stream (either request or response). Refer to *CIP Standard Vol1_3.22 Section 5B-3 and/or MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

2.1.4.0.8. Vendor Specific Write Single Coil Service (70 Hex)

This service writes one coil to the Modbus Coils table. This service results in the DEVICEMASTER ETHERNET/IP- DeviceMaster EIP-MOD issuing a Write Single Coil function (function code 0x05) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

Note: *In order to use this service, the Enable Vendor Specific Modbus Object Services option on the EtherNet/IP Stack Configuration web page must be selected.*

2.1.4.0.9. Vendor Specific Write Single Register Service (71 Hex)

This service writes one holding register to the Modbus Holding Registers table. This service results in the DeviceMaster EIP-MOD issuing a Write Single Registers function (function code 0x06) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

Note: *In order to use this service, the Enable Vendor Specific Modbus Object Services option on the EtherNet/IP Stack Configuration web page must be selected.*

2.1.4.0.10. Vendor Specific Write Mask Register (72 Hex)

This service performs a write mask operation on one holding register in the Modbus Holding Registers table. This service results in the DeviceMaster EtherNet/IP-Modbus gateway issuing a Write Mask Register function (function code 0x16) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

Note: *In order to use this service, the Enable Vendor Specific Modbus Object Services option on the EtherNet/IP Stack Configuration web page must be selected.*

2.1.4.0.11. Vendor Specific Read/Write Holding Registers (73 Hex)

This service performs first write and then read operations on multiple holding registers in the Modbus Holding Registers table. This service results in the DeviceMaster EtherNet/IP-Modbus gateway issuing a Read/Write Holding Registers function (function code 0x17) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

Note: *In order to use this service, the Enable Vendor Specific Modbus Object Services option on the EtherNet/IP Stack Configuration web page must be selected.*

2.2. Informational Objects

The following object definitions are included for informational purposes only. While some software packages, such as RSLinx make use of these objects, few PLC programmers will have a need to directly access them.

2.2.1. Identity Object (01 Hex, 1 Instance)

The Identity Object provides identification of and general information about the DeviceMaster EIP-MOD.

2.2.1.1. Class Attributes

The following table provides information about the Identity Object (01 hex, 1 Instance) Class Attributes.

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Revision	UINT	1	Get
2	Max Class	UINT	1	Get
3	Max Instance	UINT	1	Get
6	Maximum Number Class Attribute	UINT	7	Get
7	Maximum Number Instance Attribute	UINT	7	Get

2.2.1.2. Instance Attributes

The following table provides information about the Identity Object (01 hex, 1 Instance) Instance Attributes.

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Vendor ID	UINT	909 (Pepperl+Fuchs Comtrol, Inc.)	Get
2	Device Type	UINT	43 (Generic product device)	Get
3	Product Code	UINT	As defined by Pepperl+Fuchs Comtrol, Inc.	Get
4	Revision (product or software release)			
	Structure of:			Get
	Major Revision	USINT	1 to 127	
	Minor Revision	USINT	1 to 999	
5	Status	WORD	See below.	Get
6	Serial Number	UDINT	1-FFFFFFFF hex	Get

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
7	Product Name Structure of: Name Length Name String	 USINT STRING	 Length of string See below	Get

2.2.1.3. Status Word

Refer to Page 52 of *Volume 3.5 of the CIP Common Specification*. The following applies to the Identity Object status word for the DeviceMaster EtherNet/IP-Modbus gateway.

Status Word Bit	Setting	Description
0	0	Ownership Flag. Does not apply to the DeviceMaster EtherNet/IP-Modbus gateway.
1	0	Reserved.
2	0	DeviceMaster EtherNet/IP-Modbus gateway is operating on the default configuration.
	1	The DeviceMaster EtherNet/IP-Modbus gateway has a configuration other than the default configuration.
3	0	Reserved
4-7	0101 (0x50)	Indicates that there is a major fault (either Bit 10 or Bit 11 is set).
	0100 (0x40)	Indicates the stored configuration is invalid.
	0011 (0x30)	Indicates the system is operational and there are no I/O (Class 1) connections.
	0110 (0x60)	Indicates the system is operational and there is at least one active I/O (Class 1) connection.
	0000	Indicates the system is not operational. It may be in any of the following states: • System startup. • Configuration in process. • Idle. • Critical (major) fault.
8	0	No recoverable minor fault. No error history entry reported within the last ten seconds.
	1	Recoverable minor fault. The DeviceMaster EtherNet/IP-Modbus gateway has reported an error within the last ten seconds and a major fault has not been detected.
9	1	Unrecoverable minor fault. Does not apply to the DeviceMaster EtherNet/IP-Modbus gateway.
10	0	No recoverable major fault.
	1	A major recoverable fault exists. This is a fault that the DeviceMaster EIP-MOD may be able to recover from by a system reset. If the system does not recover automatically, a system reset message or a power cycle of the DeviceMaster EtherNet/IP-Modbus gateway may be required.

Status Word Bit	Setting	Description (Continued)
11	0	No major unrecoverable fault.
	1	A major unrecoverable fault has occurred in the DeviceMaster EIP-MOD. If the major fault is not corrected with a system reset or a power cycle, refer to the *Troubleshooting and Technical Support* on Page 155 or call Pepperl+Fuchs Comtrol, Inc. support.
12-15	0	Reserved.

2.2.1.4. Common Services

This table displays Common Services for Identity Object (01 hex, 1 Instance).

Service Code	Implemented in Class	Implemented in Instance	Service Name
01 hex	Yes	Yes	Get_Attribute_All
05 hex	No	Yes	Reset
0E hex	Yes	Yes	Get_Attribute_Single

2.2.2. Message Router Object (02 Hex)

The Message Router Object provides a messaging connection point through which a client may address a service to any object or instance residing in the physical device.

2.2.2.1. Class Attributes

This table displays the Class Attributes for Message Router Object (02 hex).

Attribute ID	Name	Data Type	Data Value	Access Rule
1	Revision	UINT	1	Get
2	Max Class	UINT	1	Get
3	Max Instance	UINT	1	Get
4	Optional Attribute List	UINT	2	Get
5	Option Service List	UINT	1	Get
6	Maximum Number Class Attribute	UINT	7	Get
7	Maximum Number Instance Attribute	UINT	2	Get

2.2.2.2. Instance Attributes

This table displays Instance Attributes for Message Router Object (02 hex).

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Object List Structure of: Number Classes	 UINT Array of UINT	 Number of supported class codes List of supported class codes	Get
2	Max Connections	UINT	128	Get

2.2.2.3. Common Services

This table displays Common Services for Message Router Object (02 hex).

Service Code	Implemented in Class	Implemented in Instance	Service Name
01 hex	Yes	No	Get_Attribute_All
0E hex	Yes	Yes	Get_Attribute_Single
0A hex	No	Yes	Multiple_Service_Req

2.2.3. Connection Manager Object (06 Hex)

The Connection Manager Object provides services for connection and connection-less communications.

2.2.3.1. Class Attributes Object (06 hex)

The following table displays the Class Attributes for the Connection Manager Object (06 hex).

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Revision	UINT	1	Get
2	Max Class	UINT	1	Get
3	Max Instance	UINT	1	Get
4	Optional Attribute List	UINT	8	Get
6	Maximum number Class Attribute	UINT	7	Get
7	Maximum Number Instance Attributes	UINT	8	Get

2.2.3.2. Instance Attributes (06 hex)

This table displays the Instance Attributes for the Connection Manager Object (06 hex).

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Open Requests	UINT	0-0xffffffff	Set/Get
2	Open Format Rejects	UINT	0-0xffffffff	Set/Get
3	Open Resource Rejects	UINT	0-0xffffffff	Set/Get
4	Open Other Rejects	UINT	0-0xffffffff	Set/Get
5	Close Requests	UINT	0-0xffffffff	Set/Get
6	Close Format Requests	UINT	0-0xffffffff	Set/Get
7	Close Other Requests	UINT	0-0xffffffff	Set/Get
8	Connection Time Outs	UINT	0-0xffffffff	Set/Get

2.2.3.3. Common Services Object (06 hex)

This table displays the Common Services for the Connection Manager Object (06 hex).

Service Code	Implemented in Class	Implemented in Instance	Service Name
01 hex	Yes	Yes	Get_Attribute_All
02 hex	No	Yes	Set_Attribute_ALL
0E hex	Yes	Yes	Get_Attribute_Single
10 hex	No	Yes	Set_Attribute_Single
4E hex	N/A	N/A	Forward_Close
52 hex	N/A	N/A	Unconnected_Send
54 hex	N/A	N/A	Forward_Open
5A hex	N/A	N/A	Get_Connection_Owner
5B hex	N/A	N/A	Large_Forward_Open

2.2.4. Port Object (F4 Hex - 1 Instance)

The Port Object enumerates the CIP ports on the DeviceMaster EIP-MOD.

2.2.4.1. Class Attributes

This table displays Class Attributes for Port Object (F4 hex - 1 Instance).

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Revision	UINT	2	Get
2	Max Instance	UINT	2	Get
3	Num Instances	UINT	2	Get
6	Maximum Number Class Attributes	UINT	9	Get
7	Maximum Number Instance Attributes	UINT	10	Get
8	Entry Port	UINT	1	Get
9	All Ports	Array of UINT	[0]=0 [1]=0 [2] = 1 (Vendor Specific) [3] = 1 (Backplane) [4]=TCP_IP_PORT_TYPE (4) [5]=TCP_IP_PORT_NUMBER(2)	Get

2.2.4.2. Instance Attributes (F4 Hex, Instance 1)

This table illustrates the Instance Attributes for the Port Object (F4 hex – Instance 1).

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Port Type	UINT	1	Get
2	Port Number	UINT	1	Get
3	Port Object Structure of: 16-bit word count in path Path	 UINT Array of UINT	 2 [0]=6420 hex [1]=0124	Get
4	Port Name Structure of: String Length Port Name	 USINT Array of USINT	 10 “Backplane”	Get
7	Node Address	USINT[2]	10 hex, 0 hex	Get

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
10	Port Routing Capabilities	UDINT	17 hex: • Bit 0: Routing of incoming Unconnected Messaging supported • Bit 1: Routing of outgoing Unconnected Messaging supported • Bit 2: Routing of incoming Transport Class 0/1 Connections supported • Bit 4: Routing of incoming Transport Class 2/3 Connections supported	Get

2.2.4.3. Instance Attributes (F4 Hex, Instance 2)

This table illustrates the Instance Attributes for the Port Object (F4 hex – Instance 2).

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Port Type	UINT	4 (TCP/IP)	Get
2	Port Number	UINT	2 (TCP/IP)	Get
3	Port Object Structure of: 16-bit word count in path Path	 UINT Array of UINT	 2 [0]=F520 hex [1]=0124 hex	Get
4	Port Name Structure of: String Length Port Name	 USINT Array of USINT	 17 “Ethernet/IP Port”	Get
7	Node Address	USINT[2]	10 hex, 0 hex	Get
10	Port Routing Capabilities	UDINT	17 hex: Bit 0: Routing of incoming Unconnected Messaging supported Bit 1: Routing of outgoing Unconnected Messaging supported Bit 2: Routing of incoming Transport Class 0/1 Connections supported Bit 4: Routing of incoming Transport Class 2/3 Connections supported	Get

2.2.4.4. Common Services

This table displays Common Services for Port Object (F4 hex, Instance 2).

Service Code	Implemented in Class	Implemented in Instance	Service Name
01 hex	Yes	Yes	Get_Attribute_All
0E hex	Yes	Yes	Get Attribute Single

2.2.5. TCP Object (F5 Hex - 1 Instance)

The TCP/IP Interface Object provides the mechanism to retrieve the TCP/IP attributes for the DeviceMaster EIP-MOD.

2.2.5.1. Class Attributes

This table displays the Class Attributes for TCP Object (F5 hex - 1 instance).

Attribute ID	Name	Data Type	Data Value	Access Rule
1	Revision	UINT	4	Get
2	Max Instance	UINT	1	Get
3	Num Instances	UINT	1	Get
4	Optional Attribute List	UINT	4	Get
6	Maximum Number Class Attribute	UINT	7	Get
7	Maximum Number Instance Attribute	UINT	13	Get

2.2.5.2. Instance Attributes

This table displays Instance Attributes for TCP Object (F5 hex - 1 instance).

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Status	DWORD	0 = The Interface Configuration attribute has not been configured. 1 = The Interface Configuration attribute contains configuration obtained from DHCP or nonvolatile storage.	Get
2	Configuration Capability	DWORD	14 hex (DHCP and SETTABLE)	Get
3	Configuration Control	DWORD	0=Use stored IP address (static IP address) 2=DHCP	Get

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
4	Physical Link Object Structure of: Path Size Path	 UINT Array of USINT	 2 [0]=20 hex [1]=F6 hex [2]=24 hex [3]=01 hex	Get
5	Interface Configuration Structure of: IP Address Network Mask Gateway Address Name Server Name Server 2 Domain Name Length Domain Name	 UDINT UDINT UDINT UDINT UDINT UINT STRING	 <IP address> <Network mask> <Gateway Addr> <Name server> <Name server 2> <Length of name> <Domain name>	Set
6	Host Name Structure of: Host Name Length Host Name	 UINT STRING	 0 to 15 <Default = IP address>	Set
8	TTL (Time-to-Live) value for IP multicast packets	USINT	1 to 255 <Default = 1>	Set
9	IP Multicast Address Configuration	Struct of: USINT –Allocation Control USINT – Reserved UINT – Num Mcast UDINT – Start Multicast Address	 Allocation Control 0=Default Algorithm 1 = Configuration Num Multicast Addresses: 1-32 Start Multicast Address: 239.192.1.0 to 239.255.255.255	Set
13	Encapsulation Inactivity Timeout	USINT	Number of seconds of inactivity before TCP connection or DTLS session is closed	Set

2.2.5.3. Common Services

This table shows the Common Services for the TCP Object (F5 hex - I Instance).

Service Code	Implemented in Class	Implemented in Instance	Service Name
01 hex	Yes	Yes	Get_Attribute_All
02 hex	No	Yes	Set_Attribute_All
0E hex	Yes	Yes	Get_Attribute_Single
10 hex	No	Yes	Set_Attribute_Single

2.2.6. Ethernet Link Object (F6 Hex)

The Ethernet Link object maintains link-specific counters and status information for the Ethernet communications on the DeviceMaster EIP-MOD.

2.2.6.1. Class Attributes

This table displays the Class Attributes for the Ethernet Link Object (F6 hex).

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Revision	UINT	4	Get
2	Max Instance	UINT	1 = One Ethernet Port Models 3 = Two Ethernet Port Models	Get
3	Num Instances	UINT	1 = One Ethernet Port Models 3 = Two Ethernet Port Models	Get
4	Optional Attribute List	UINT	4	Get
6	Maximum Number Class Attributes	UINT	7	Get
7	Maximum Number Instance Attributes	UINT	11	Get

2.2.6.2. Instance Attributes

This table displays the Instance Attributes for the Ethernet Link Object (F6 hex).

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Interface speed (Current operational speed)	UDINT	10=10 Mbit 100=100 Mbit	Get
2	Interface Flags (Current operational status)	DWORD	Bit 0 =link status (0=inactive) (1=active) Bit 1=Half/Full Duplex (0=half duplex) (2=full duplex) Bits 2-4: 00 = negotiation in progress 01 = negotiation failed 02 = negotiation failed speed OK 03 = negotiation success	Get
3	Physical Address	Array of 6 USINT	MAC Address	Get
7	Interface Type	USINT	2 = Twisted Pair	Get
8	Interface State	USINT	1 = Interface is enabled and operational	Get
9	Admin State	USINT	1 = Interface enabled	Get
10	Interface Label	USINT16 Array of USINT	Length = 1 to 64 ASCII characters <Default = IP address in “xxx.xxx.xxx.xxx” format>	Get
11	Interface Capability	UDINT Struct of: USINT Array of: UINT USINT	Capability bits - Interface capabilities, other than speed/ duplex Value = 6 Bit 1: Auto-negotiate Bit 2: Auto-MDIX Speed/Duplex Array Count = 1 Interface Speed - 10 or 100 Interface Duplex Mode 0=half duplex 1=full duplex	Get6

2.2.6.3. Common Services

This table displays the Common Services for Ethernet Link Object (F6 hex).

Service Code	Implemented in Class	Implemented in Instance	Service Name
01 hex	Yes	Yes	Get_Attribute_All
0E hex	Yes	Yes	Get_Attribute_Single

2.2.7. Assembly Object (For Class 1 Interface)

The EtherNet/IP specification requires that all Class 1 interfaces be provided through the Assembly Object interface. The Assembly Object interface is used to directly tie Vendor Specific objects to a standard interface, which the EtherNet/IP controller, or PLC, uses to communicate to the device.

For the DeviceMaster EtherNet/IP-Modbus gateway, the Assembly Object corresponds to the Serial and Socket Data Transfer objects. Each instance of the Assembly Object corresponds to one or more of the Serial and/or Socket Data Transfer Object attributes.

The Assembly object defines the interface by which a Class 1 PLC or controller can:

- Request the receive data packets from the DeviceMaster EtherNet/IP-Modbus gateway.
- Write the transmit data to the DeviceMaster EtherNet/IP-Modbus gateway.

2.2.7.1. Class Attributes

This table shows the Class Attributes for the Assembly Object for a Class 1 Interface.

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
1	Revision	UINT	2	Get
2	Max Instance	UINT	132	Get
3	Num Instances	UINT	32	Get
4	Optional Attribute List	UINT	4	Get
6	Maximum Number Class Attribute	UINT	7	Get
7	Maximum Number Instance Attribute	UINT	4	Get

2.2.7.2. Instance Attributes

This table shows the Instance Attributes for the Assembly Object for a Class 1 Interface.

Attribute ID	Name	Data Type	Data Value(s)	Access Rule
3	Data	Array of BYTE	0-255	Get/Set
4	Data Length	UINT	Maximum number of bytes in Attribute 3	Get

2.2.7.2.1. Instance Attribute Definitions: Attribute 3-Request/Write Data

Dependent on the instance number, this is either the PDI data block and/or the PDO data block.

2.2.7.2.2. Instance Attribute Definitions: Attribute 4-Data Length

This is the maximum data length for each Assembly instance.

2.2.7.3. Common Services

This table shows the Common Services for the Assembly Object for a Class 1 Interface.

Service Code	Implemented in Class	Implemented in Instance	Service Name
01 hex	Yes	No	Get_Attributes_All
0E hex	Yes	Yes	Get_Attribute_Single
10 hex	No	Yes	Set_Attribute_Single
02 hex	No	No	Set_Attribute_All

2.2.7.4. Instance Definitions

Each Assembly instance is tied directly to a Shared Memory block of Holding Registers or Coils. Each input instance is enabled only if the corresponding block's Read Class1 Enable has been selected in the Shared Memory configuration. Each output instance is enabled only if the corresponding block's Write Master(s) option has been set to EIP Class1.

2.2.7.4.1. Instance Definitions (Default)

This table shows the Assembly Object instance definitions for the default Shared Memory configuration.

Default Configuration Shared Memory Configuration	
Holding Register Block 1	Read Class1 Enable selected
Coils Block 1	Read Class1 Enable selected
Holding Register Block 2	Write Master(s) set to EIP Class1
Coils Block 2	Write Master(s) set to EIP Class1

For all other Shared Memory blocks:

- Read Class1 Enable not selected
- Write Master(s) not set to EIP Class1

Assembly Input Instances				
Assembly Instance Number	**Description**	**Data Type**	**Data Values**	**Access Rule**
101	Read data from Shared Memory Holding Registers Block 1	WORD Array Length = 200	0-FFFF (hex)	Get
109	Read data from Shared Memory Coils Block 1	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Get

Assembly Output Instances				
Assembly Instance Number	**Description**	**Data Type**	**Data Values**	**Access Rule**
118	Write data to Shared Memory Holding Registers Block 2	WORD Array Length = 200	0-FFFF (hex)	Set
126	Write data to Shared Memory Coils Block 2	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Set

2.2.7.4.2. All Available Instance Definitions

This table displays all available Instance Definitions.

Assembly Input Instances				
Assembly Instance Number	**Description**	**Data Type**	**Data Values**	**Access Rule**
101	Read data from Shared Memory Holding Registers Block 1	WORD Array Length = 200	0-FFFF (hex)	Get
102	Read data from Shared Memory Holding Registers Block 2	WORD Array Length = 200	0-FFFF (hex)	Get
103	Read data from Shared Memory Holding Registers Block 3	WORD Array Length = 200	0-FFFF (hex)	Get
104	Read data from Shared Memory Holding Registers Block 4	WORD Array Length = 200	0-FFFF (hex)	Get
105	Read data from Shared Memory Holding Registers Block 5	WORD Array Length = 200	0-FFFF (hex)	Get
106	Read data from Shared Memory Holding Registers Block 6	WORD Array Length = 200	0-FFFF (hex)	Get
107	Read data from Shared Memory Holding Registers Block 7	WORD Array Length = 200	0-FFFF (hex)	Get
108	Read data from Shared Memory Holding Registers Block 8	WORD Array Length = 200	0-FFFF (hex)	Get
109	Read data from Shared Memory Coils Block 1	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Get
110	Read data from Shared Memory Coils Block 2	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Get
111	Read data from Shared Memory Coils Block 3	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Get

Assembly Input Instances (Continued)				
Assembly Instance Number	**Description**	**Data Type**	**Data Values**	**Access Rule**
112	Read data from Shared Memory Coils Block 4	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Get
113	Read data from Shared Memory Coils Block 5	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Get
114	Read data from Shared Memory Coils Block 6	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Get
115	Read data from Shared Memory Coils Block 7	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Get
116	Read data from Shared Memory Coils Block 8	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Get

Assembly Output Instances				
Assembly Instance Number	**Description**	**Data Type**	**Data Values**	**Access Rule**
117	Write data to Shared Memory Holding Registers Block 1	WORD Array Length = 200	0-FFFF (hex)	Set
118	Write data to Shared Memory Holding Registers Block 2	WORD Array Length = 200	0-FFFF (hex)	Set
119	Write data to Shared Memory Holding Registers Block 3	WORD Array Length = 200	0-FFFF (hex)	Set
120	Write data to Shared Memory Holding Registers Block 4	WORD Array Length = 200	0-FFFF (hex)	Set
121	Write data to Shared Memory Holding Registers Block 5	WORD Array Length = 200	0-FFFF (hex)	Set
122	Write data to Shared Memory Holding Registers Block 6	WORD Array Length = 200	0-FFFF (hex)	Set
123	Write data to Shared Memory Holding Registers Block 7	WORD Array Length = 200	0-FFFF (hex)	Set
124	Write data to Shared Memory Holding Registers Block 8	WORD Array Length = 200	0-FFFF (hex)	Set
125	Write data to Shared Memory Coil Block 1	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Set
126	Write data to Shared Memory Coil Block 2	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Set
127	Write data to Shared Memory Coil Block 3	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Set

Assembly Output Instances (Continued)				
Assembly Instance Number	**Description**	**Data Type**	**Data Values**	**Access Rule**
128	Write data to Shared Memory Coil Block 4	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Set
129	Write data to Shared Memory Coil Block 5	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Set
130	Write data to Shared Memory Coil Block 6	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Set
131	Write data to Shared Memory Coil Block 7	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Set
132	Write data to Shared Memory Coil Block 8	WORD Array Length = 20 Words (320 coils)	0-FFFF (hex)	Set

Chapter 3. Configuration Overview

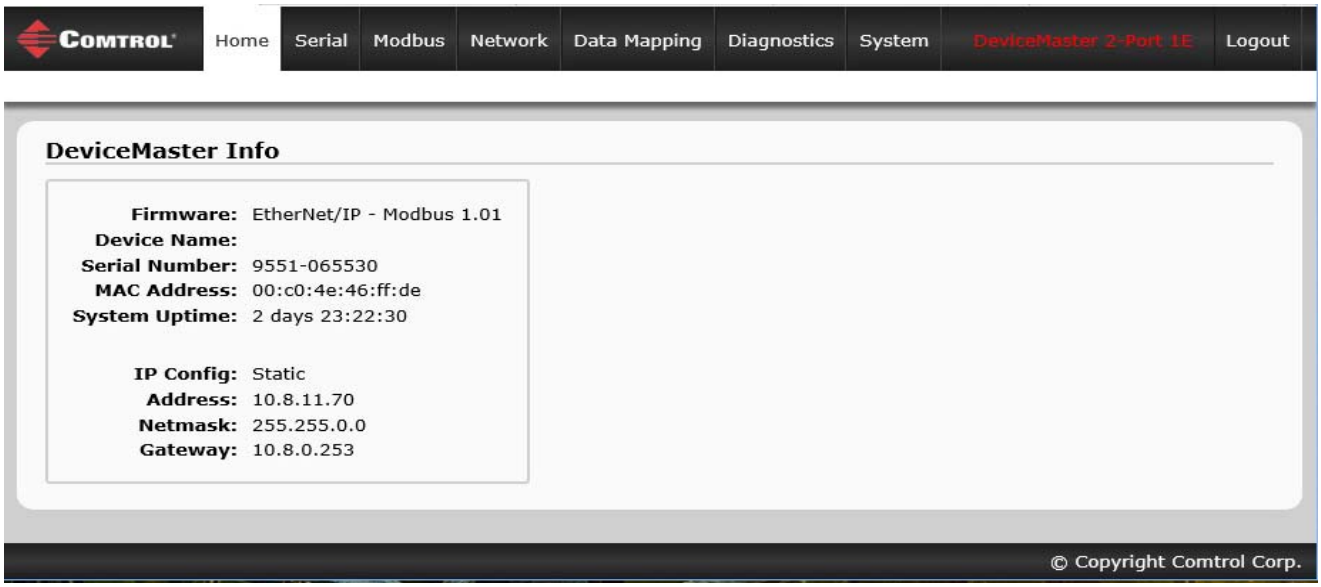
This chapter provides an overview of these topics:

3.1. Home Page

If you have not configured the network information into the DeviceMaster EIP-MOD during initial setup, you must configure the network information before configuring serial/socket port characteristics.

The latest EtherNet/IP-Modbus firmware must be installed before you can configure network or serial/socket port characteristics. For firmware installation and setup information, see the *DeviceMaster EIP-MOD Hardware Installation and Configuration Guide* or the PortVision DX help system.

From PortVision DX, highlight the DeviceMaster EIP-MOD that you want to configure and click **Webpage** or enter the IP address of the DeviceMaster EIP-MOD in the ***Address*** box of your web browser. The **Home** page displays.



EtherNet/IP-Modbus Home Page	
Firmware	EtherNet/IP-Modbus firmware version currently running on the DeviceMaster.
Device Name	You can enter a Device Name in the *Network* | *Configuration* page, which will display in this field.
Serial Number	The DeviceMaster EIP-MOD serial number.
MAC Address	This is the MAC address of this DeviceMaster EIP-MOD, which is located on the compliance label on the DeviceMaster EIP-MOD.
System Uptime	Displays how long this DeviceMaster EIP-MOD has been on-line since powered on or rebooted.
IP Config	Type of IP configuration currently in use (static or DHCP).
IP Address, IP Netmask, IP Gateway	IP address, netmask, and gateway configured in the DeviceMaster EIP-MOD.

Use the appropriate chapter as a reference for the configuration options.

- Chapter 4. Serial Menus on Page 47
- Chapter 5. Modbus Menus on Page 53

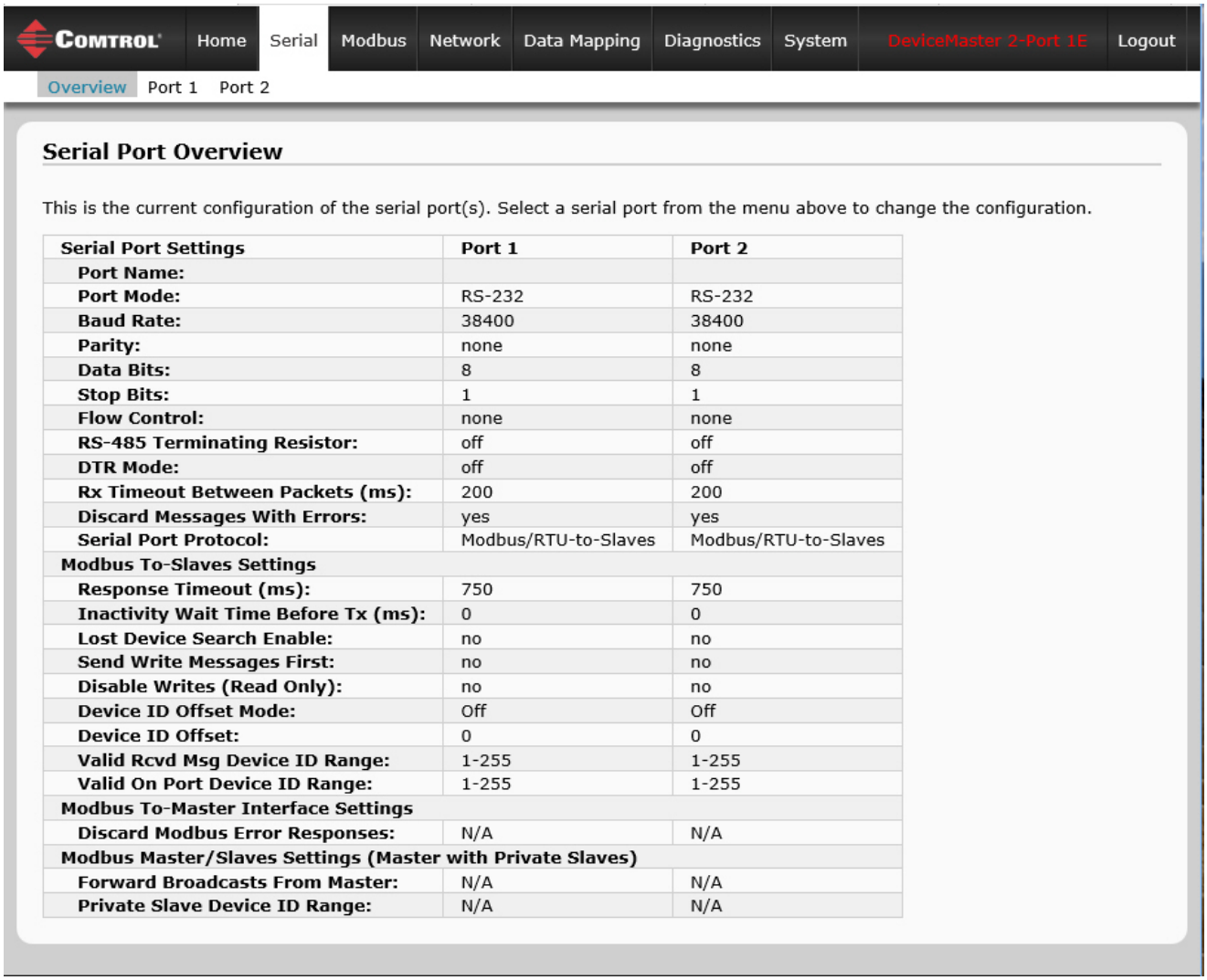
You can review the configuration pages and basic procedures in the following subsections.

- *3.2. Serial Port - Configuration Overview* on Page 43
- *3.3. Modbus Over TCP Overview - Configuration Overview* on Page 45

3.2. Serial Port - Configuration Overview

Use the following steps to access the serial port configuration pages.

1. Open the DeviceMaster EIP-MOD web page using the IP address in your browser or PortVision DX.
2. Click the **Serial** menu, which opens the *Serial Port Overview* page.

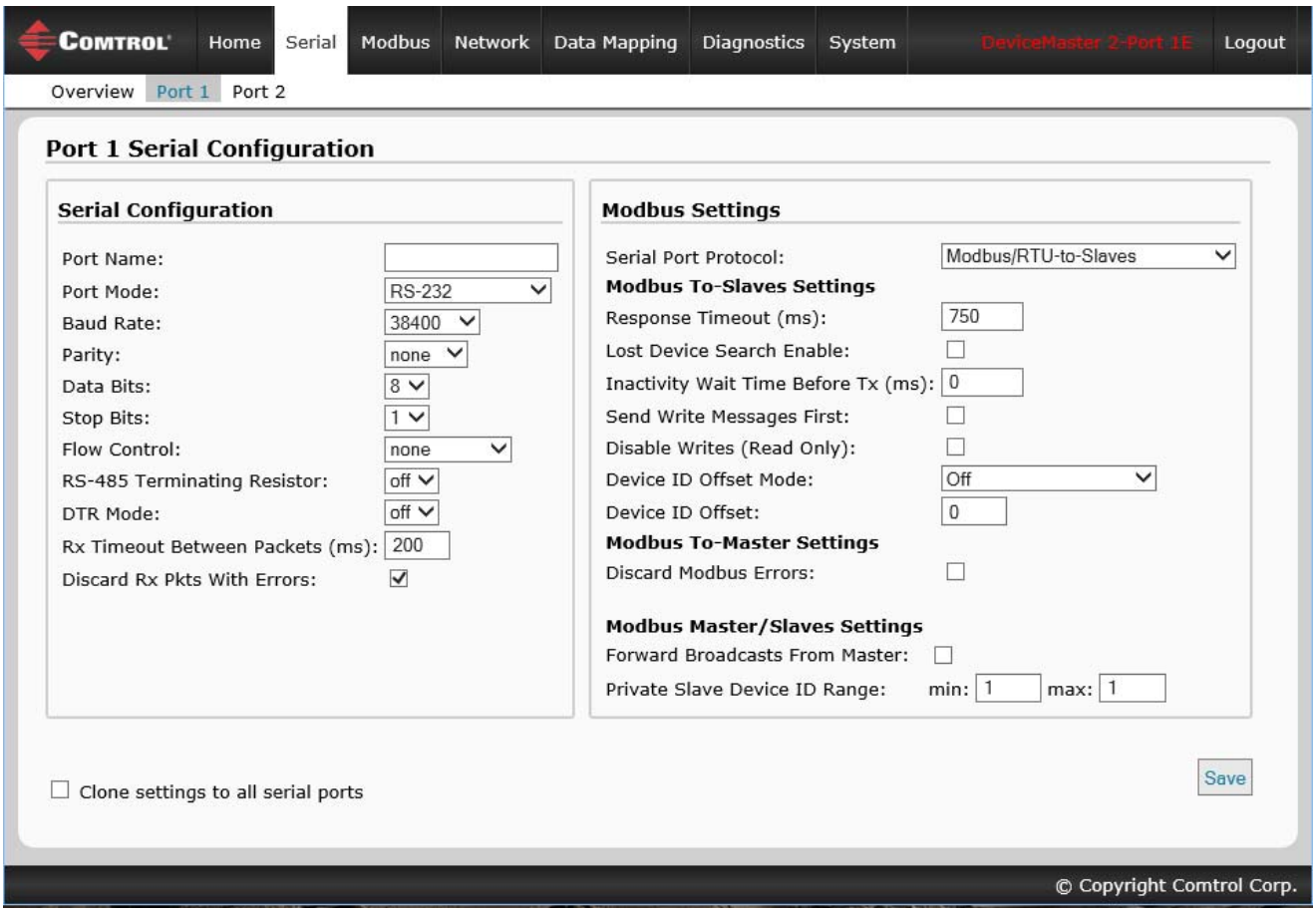


Serial Port Overview

This is the current configuration of the serial port(s). Select a serial port from the menu above to change the configuration.

Serial Port Settings	Port 1	Port 2
Port Name:		
Port Mode:	RS-232	RS-232
Baud Rate:	38400	38400
Parity:	none	none
Data Bits:	8	8
Stop Bits:	1	1
Flow Control:	none	none
RS-485 Terminating Resistor:	off	off
DTR Mode:	off	off
Rx Timeout Between Packets (ms):	200	200
Discard Messages With Errors:	yes	yes
Serial Port Protocol:	Modbus/RTU-to-Slaves	Modbus/RTU-to-Slaves
Modbus To-Slaves Settings		
Response Timeout (ms):	750	750
Inactivity Wait Time Before Tx (ms):	0	0
Lost Device Search Enable:	no	no
Send Write Messages First:	no	no
Disable Writes (Read Only):	no	no
Device ID Offset Mode:	Off	Off
Device ID Offset:	0	0
Valid Rcvd Msg Device ID Range:	1-255	1-255
Valid On Port Device ID Range:	1-255	1-255
Modbus To-Master Interface Settings		
Discard Modbus Error Responses:	N/A	N/A
Modbus Master/Slaves Settings (Master with Private Slaves)		
Forward Broadcasts From Master:	N/A	N/A
Private Slave Device ID Range:	N/A	N/A

3. Click the port number that you want to configure and the *Serial Settings* | *Port Configuration* page appears.

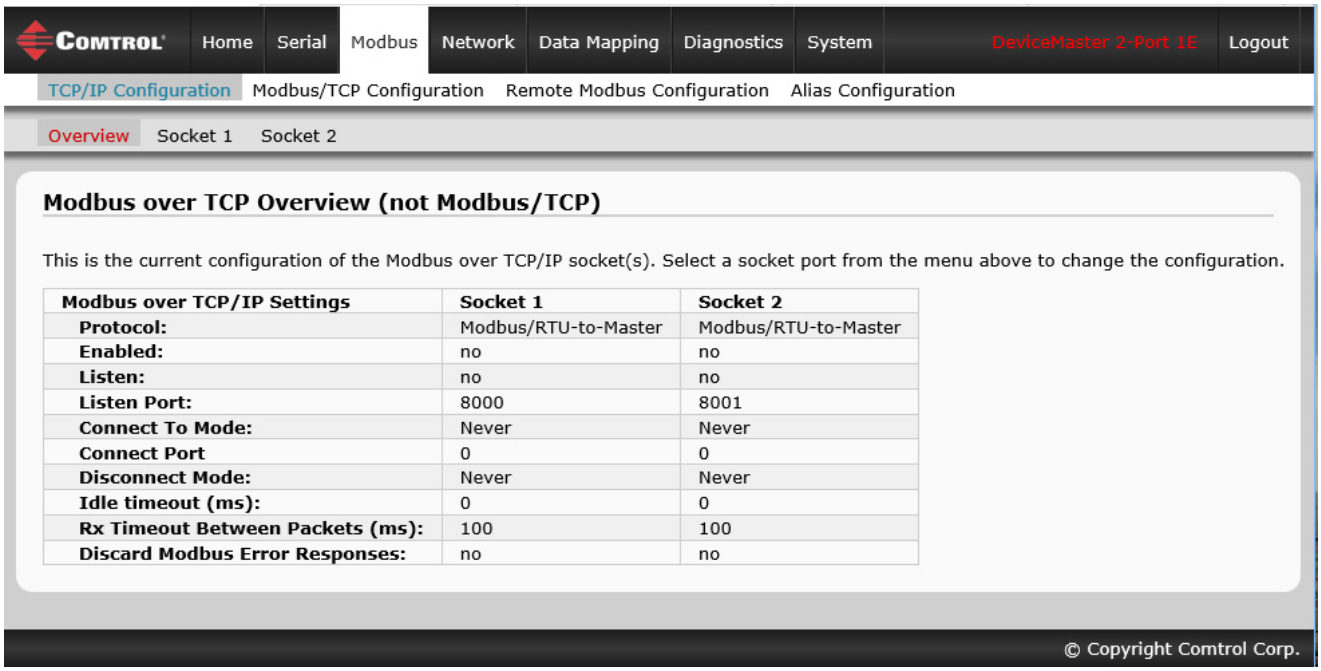


4. Change the serial port configuration properties (Page 48) as required for your site.
5. Click the **Save** button.

3.3. Modbus Over TCP Overview - Configuration Overview

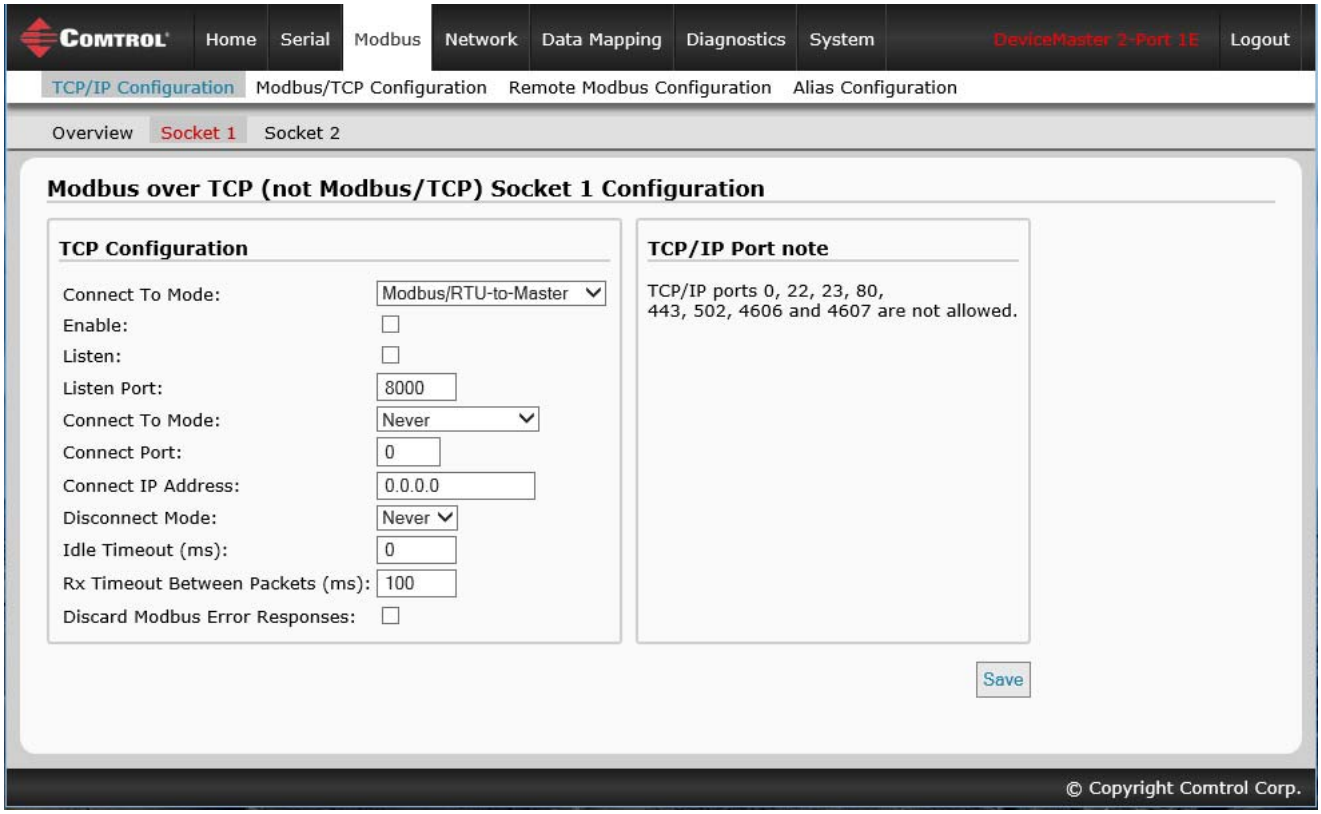
Use the following overview to configure the Modbus device.

1. Click the **Modbus** menu, which opens the *Modbus over TCP Overview (not Modbus/TCP)* page.



2. Click the appropriate **Socket** number to access the *Modbus over TCP (not Modbus/TCP) Socket Configuration* page for that port.

3. Change the socket configuration properties (Page 53) as required for you site.



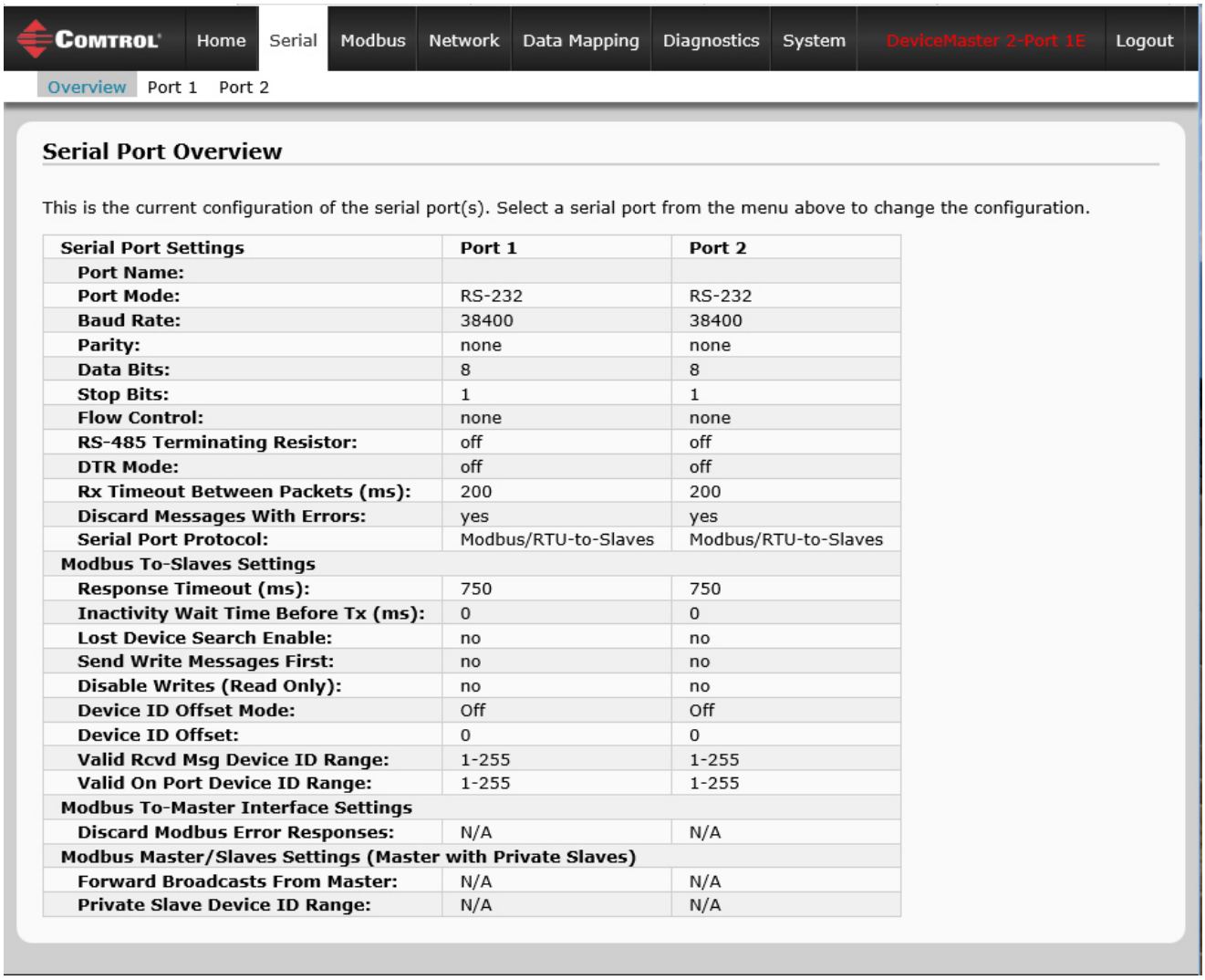
4. Click the **Save** button after completing the changes.
5. Repeat the previous steps for each serial port and Modbus device.

Chapter 4. Serial Menus

This section discusses the *4.1. Serial Port Overview Page* and *4.2. Serial Port Configuration Page* on Page 48, which are located under the **Serial** menu.

4.1. Serial Port Overview Page

The *Serial Port Overview* page displays information about the configured serial settings for each port.



Serial Port Settings	Port 1	Port 2
Port Name:		
Port Mode:	RS-232	RS-232
Baud Rate:	38400	38400
Parity:	none	none
Data Bits:	8	8
Stop Bits:	1	1
Flow Control:	none	none
RS-485 Terminating Resistor:	off	off
DTR Mode:	off	off
Rx Timeout Between Packets (ms):	200	200
Discard Messages With Errors:	yes	yes
Serial Port Protocol:	Modbus/RTU-to-Slaves	Modbus/RTU-to-Slaves
Modbus To-Slaves Settings		
Response Timeout (ms):	750	750
Inactivity Wait Time Before Tx (ms):	0	0
Lost Device Search Enable:	no	no
Send Write Messages First:	no	no
Disable Writes (Read Only):	no	no
Device ID Offset Mode:	Off	Off
Device ID Offset:	0	0
Valid Rcvd Msg Device ID Range:	1-255	1-255
Valid On Port Device ID Range:	1-255	1-255
Modbus To-Master Interface Settings		
Discard Modbus Error Responses:	N/A	N/A
Modbus Master/Slaves Settings (Master with Private Slaves)		
Forward Broadcasts From Master:	N/A	N/A
Private Slave Device ID Range:	N/A	N/A

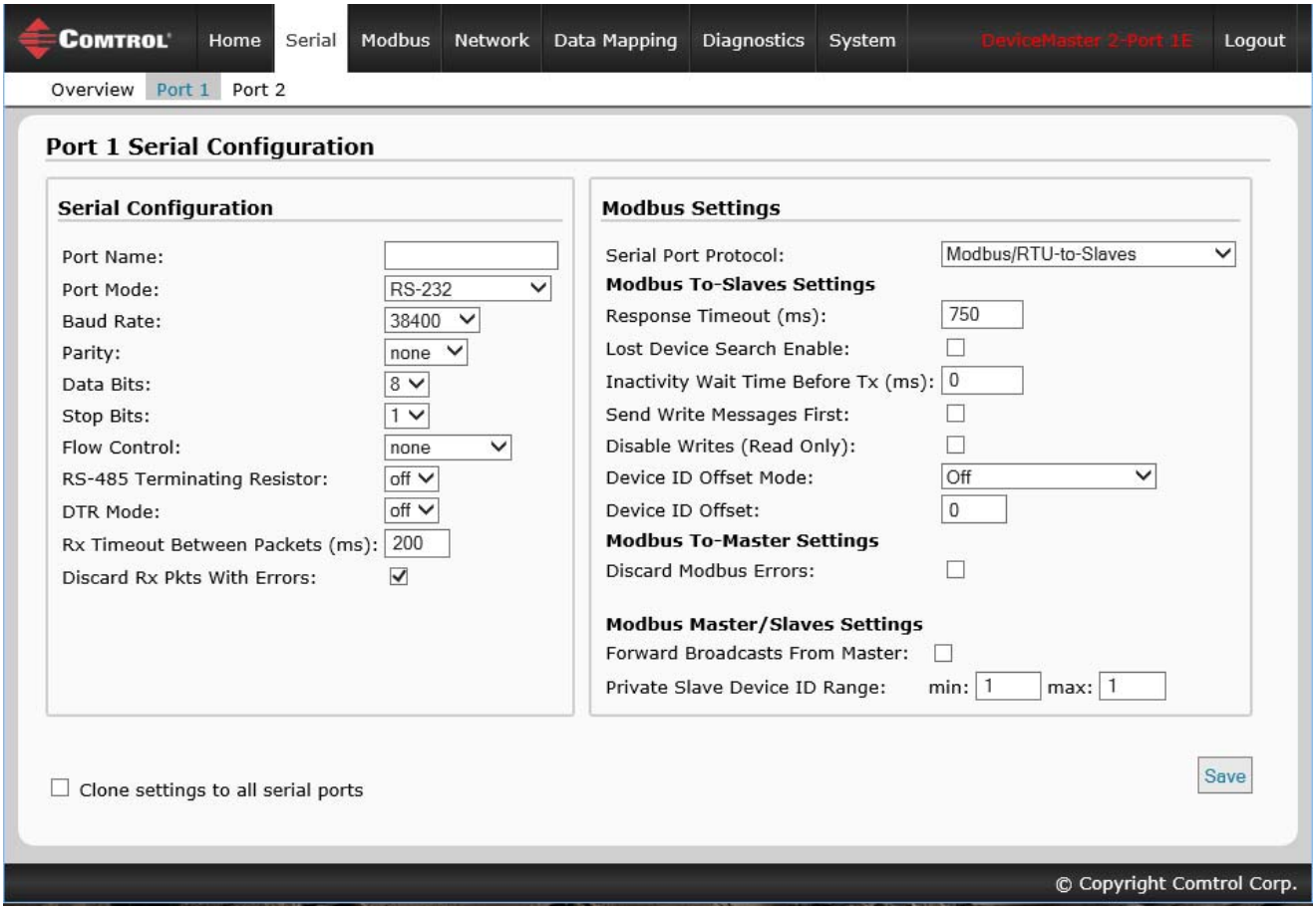
Valid Rcvd Msg Device ID Range (1-255) – (1-255): Displays the valid received message device ID range. Messages with other valid device IDs will not be transmitted out this port. (Informational only.)

Valid On Port Device ID Range (1-255) – (1-255): Displays the valid device ID range for slave devices connected to the serial port. Slave devices with other device IDs will not be accessible on this port. (Informational only.)

For additional diagnostic information, see Chapter 8. Diagnostic Menus on Page 91.

4.2. Serial Port Configuration Page

To access the *Serial Port Configuration* page, click **Serial | Port x**.



The following tables provide information about serial configuration settings.

Serial Configuration Page	
Serial Configuration	
Port Name (Default = blank)	User-definable string used to describe the serial interface. Valid characters include a-z, A-Z, 0-9, underscores, spaces and dashes. All other characters are discarded. The Port Name supports up to an 80 character ASCII string.
Port Mode (Default = RS-232)	All panel or rack mount models: • RS-232 (default) • RS-485 • RS-422 All DIN rail models: • RS-232 (default) • RS-422 • RS-485_2-wire • RS-485_4-wire_Master • RS-485_4-wire_Slave
Baud Rate (Default = 38400)	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, and 230400
Parity (Default = None)	None Even Odd
Data Bits (Default = 8)	5, 6, 7, 8
Stop Bits (Default = 1)	1 or 2
Flow Control (Default = None)	RTS/CTS XON/XOFF Half Duplex
Terminating Resistor (Only displays on applicable models)	To ensure the best signal integrity on a long distance high-speed RS-422 or RS-485 network, sometimes it is required to place a 120Ω termination resistor between each of the transmit differential pair or receive differential pair. Signal reflection is reduced and a reliable communication is ensured by placing the 120Ω termination resistor. Using the software, you can place 120Ω termination resistors between the signals shown below: • RS-422 Mode - Between TxD+ and TxD- (Pin 5 and 7) - Between RxD+ and RxD- (Pin 4 and 2) • RS-485 - Between TxD/RxD+ and TxD/RxD- (Pin 5 and 7)
DTR Mode (Default = Off)	0=Off 1=On

Serial Configuration Page (Continued)	
Rx Timeout Between Packets (ms) (Default = 200)	Receive time-out (0-65535) between packets in msec. This is the maximum spacing between received bytes allowed before the received Modbus serial message is expected to be complete.
Discard Rx Packets With Errors (Default = On)	On or Off: If selected, the DeviceMaster EIP-MOD drops all packets received with parity, framing, or overrun errors. ***Note:*** *Modbus/RTU messages with invalid CRCs and Modbus/ASCII messages without correct start and end of transmission characters are always discarded independent of this setting.*
Modbus Settings	
Serial Port Protocol (Default = Modbus/ RTU-to-Slaves)	The Modbus protocol, setting for this serial port: • **Modbus/RTU-to-Slaves** – Configures the serial port to communicate to Modbus/ RTU slaves. • **Modbus/ASCII-to-Slaves** – Configures the serial port to communicate to Modbus/ASCII slaves. • **Modbus/RTU-to-Master** – Configures the serial port to communicate to a Modbus/RTU master. • **Modbus/ASCII-to-Master** – Configures the serial port to communicate to a Modbus/ASCII master. • **Modbus/RTU-to-Master/Slaves** – Configures the serial port to communicate to a serial bus with a Modbus/RTU master and Modbus/RTU slave(s). • **Modbus/ASCII-to-Master/Slaves** – Configures the serial port to communicate to a serial bus with a Modbus/ASCII master and Modbus/ASCII slaves.
Modbus To-Slaves Settings	
Response Timeout (Default = 750msec)	The maximum allowable time (0 to 65535 msec.) for a slave device to respond to a message before the message is considered timed out.
Lost Device Search Enable *Not supported*: 1-port (Default = Off)	If selected, lost devices that were on this port are searched for on other Modbus/ RTU and Modbus/ASCII slave ports that also have this option set.
Inactivity Wait Time Before Tx (ms) (Default = 0 ms)	The minimum time (0 to 65535 msec.) that the DeviceMaster EIP-MOD waits after receiving a response or transmitting a Modbus request before transmitting the next request.
Send Write Messages First (Default = Off)	If selected, it transmits any write messages before transmitting any read messages that may have already been queued for transmission.
Disable Writes (Read Only) (Default = Off)	If selected, it disables transmission of all standard Modbus write messages.
Device ID Offset Mode (Default = Off)	• **Off** disables Device ID Offset functionality. • **Add-to-Msg-ID** adds the Device Offset to the message device ID. • **Subtract-from-Msg-ID** subtracts the **Device ID Offset** from the message device ID.

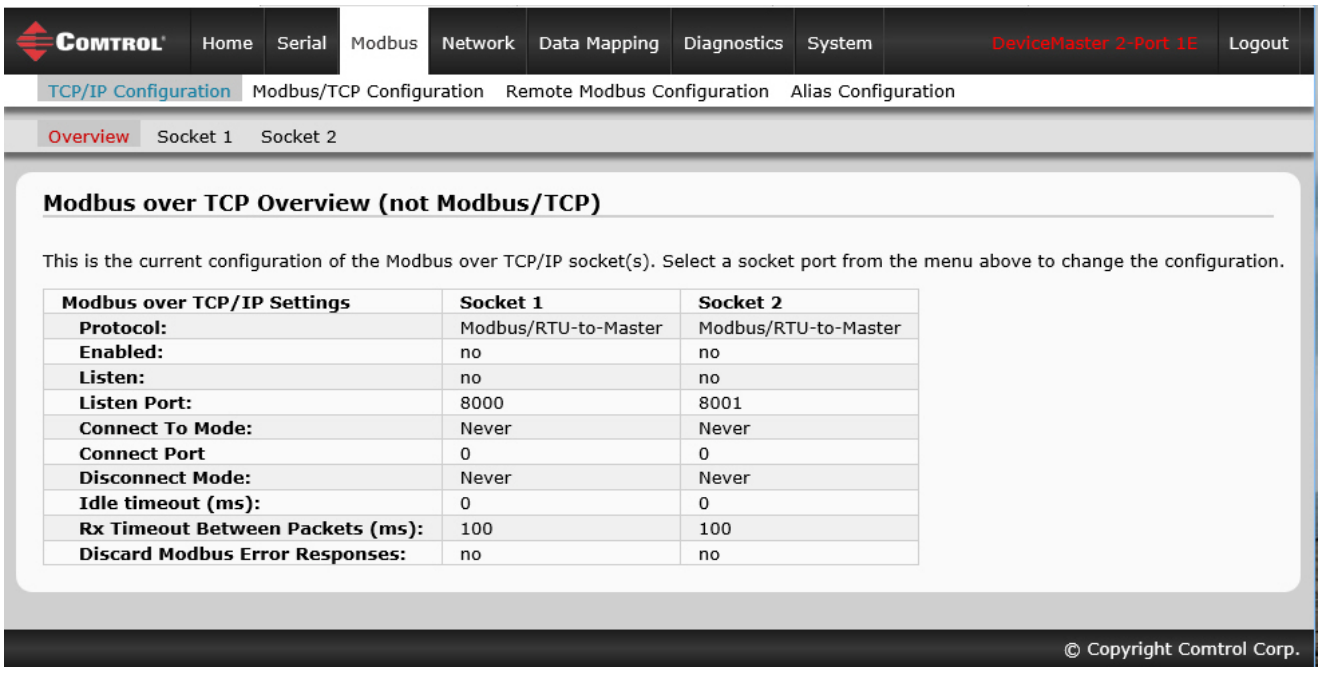
Serial Configuration Page (Continued)	
Device ID Offset	• 0 = disables Device ID Offset functionality. • 1-254 = dependent on the **Device ID Offset Mode**, is added to or subtracted from the message device ID before the Modbus message is transmitted out the serial port.
Modbus To-Master Settings	
Discard Modbus Errors (Default = Off)	If selected, all broadcast messages from the serial master will be forwarded to the Modbus network through the DeviceMaster EIP-MOD.
Modbus Master/Slave Settings	
Forward Broadcasts from Master (Default = Off)	If selected, all broadcast messages from the serial master will be forwarded to the Modbus network through the DeviceMaster EIP-MOD.
Private Slave Device ID Range (Default: Min = 1, Max = 1)	This range (1-255) defines the expected slave device ID range on the serial bus. Modbus request messages received on this port within this device ID range will not be forwarded to the Modbus network. ***Note:*** *The DeviceMaster EIP-MOD has a built-in auto-detect algorithm for detecting private slave device(s) with ID(s) not defined within the private device ID range.* ***Note:*** *The minimum value must be less than or equal to the maximum value.*

Chapter 5. Modbus Menus

This chapter discusses the following Modbus web pages:

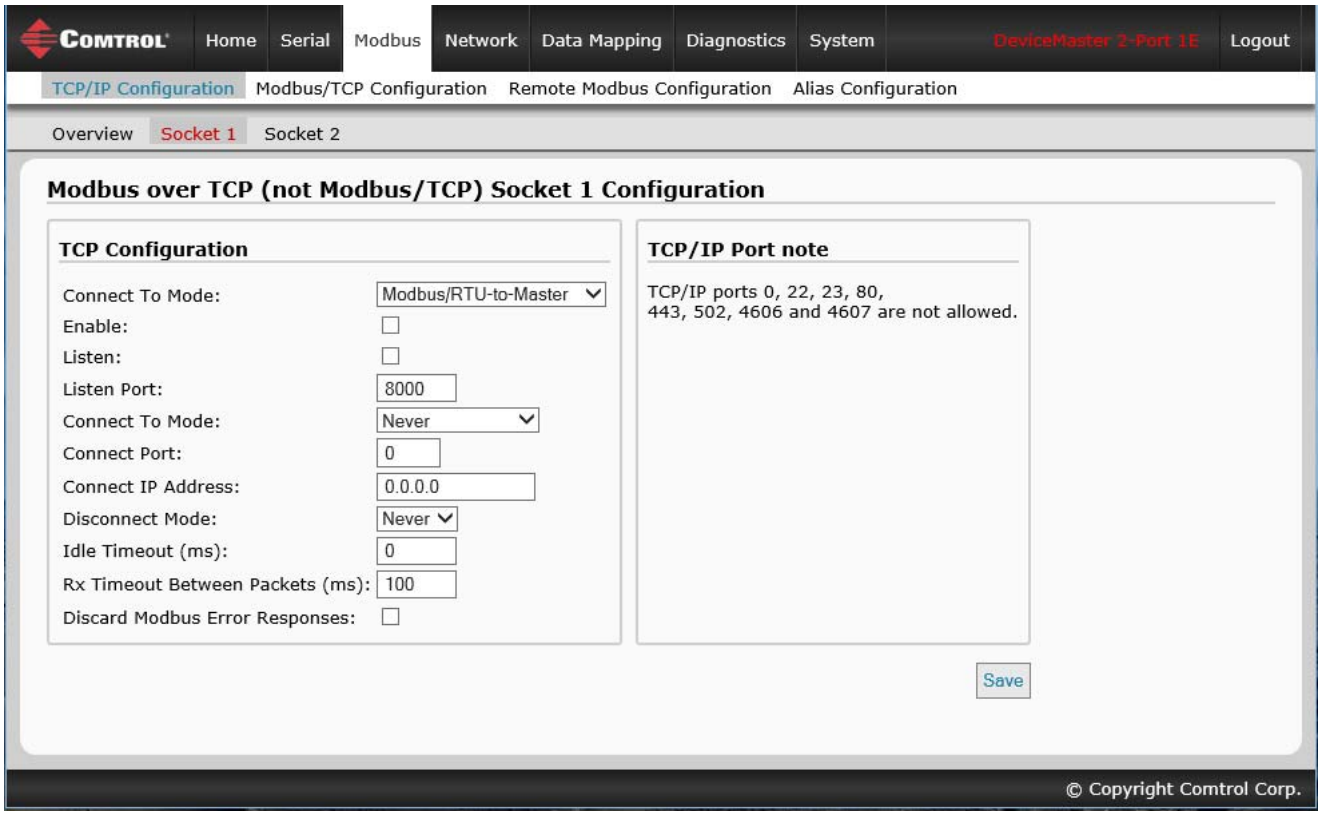
5.1. Modbus over TCP (not Modbus/TCP) Overview

The *Modbus over TCP (not Modbus/TCP) Overview* page provides an overview of the current settings for the ports.



5.2. Modbus over TCP (not Modbus/TCP) Socket Configuration Page

Access the *Modbus over TCP (not Modbus/TCP Socket Configuration* page by clicking **Modbus | TCP/IP Configuration | Socket x**. The following table provides information about each configuration setting.

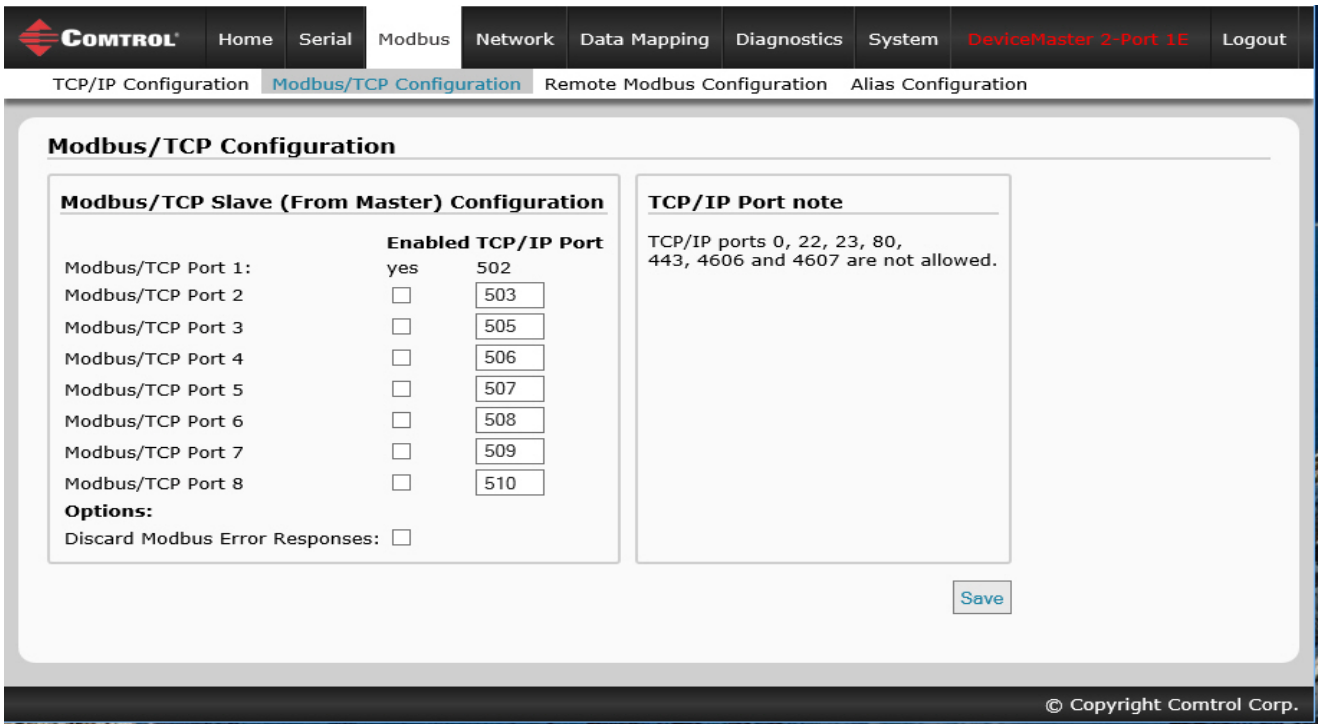


Modbus over TCP (not Modbus/TCP) Socket Configuration Page	
TCP Configuration	
Connect To Mode	• **Modbus/RTU-to-Master** – Configures the TCP/IP connection(s) to communicate to Modbus/RTU master(s). • **Modbus/ASCII-to-Master** – Configures the TCP/IP connection(s) to communicate to Modbus/ASCII master(s).
Enable (Default = Off)	If selected, this TCP/IP socket interface will be enabled.
Listen (Default = Off)	If selected, the TCP/IP socket interface will listen for a connection at the specified **Listen Port.**

Modbus over TCP (not Modbus/TCP) Socket Configuration Page (Continued)	
Listen Port (Defaults: Port 1=8000 Port 2=8001 Port 3=8002 Port 4=8003)	The **Listen Port** values are 1-65535. If **Enable** and **Listen** are both selected, allows acceptance of: • Up to six connections from external applications if there is no active **Connect-to** connection. • Up to five connections if there is an active **Connect-to** connection. The following TCP/IP ports are not allowed: 0, 22, 23, 80, 443, 502, 4606, and 4607.
Connect to Mode (Default = Never)	If **Enable** is selected, this setting determines how to connect to an application. • If **Never**: Do not attempt to make a connection. • If **Connect-Always**: Always attempt to maintain a connection to the application at **Connect IP Address** and **Connect Port**.
Connect Port (Default = 0)	Socket port to connect to (1 to 65535). Used in conjunction with **Connect to Mode** and **Connect IP Address.**
Connect IP Address (Default = 0)	IP Address of application to create a connection. Used in conjunction with **Connect to Mode** and **Connect Port**. ***Note:*** *The IP Address of this DeviceMaster EIP-MOD will not be accepted as valid configuration data.*
Disconnect Mode (Default = Never)	Mode on which to disconnect from the application. • **Never** – Will not disconnect when connection(s) are idle. • **Idle** – Utilizes the Idle Timer to determine when to close the connection.
Idle Timer (Default = 0)	If the **Disconnect Mode** is set to **Idle**, the idle or inactivity time (1 to 65535 ms) when the connection(s) will be closed.
Rx Timeout Between Packets (Default = 100)	Receive timeout (0-65565) between packets in msec. This is the maximum spacing between received bytes allowed before the received Modbus message is expected to be complete.
Discard Modbus Error Responses	

5.3. Modbus/TCP Configuration Page

Access the *Modbus/TCP Interface Configuration* page by clicking **Modbus | Modbus/TCP Configuration**. The following table provides information about each configuration setting.

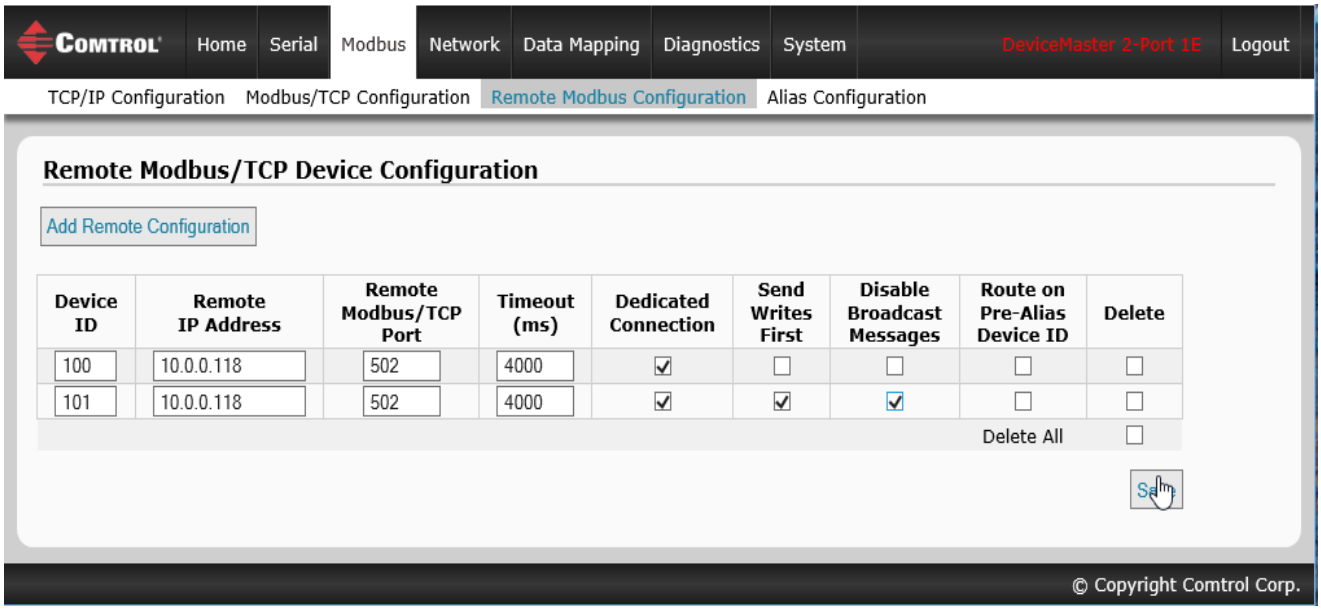


Modbus/TCP Configuration Page	
Modbus TCP/IP Port 1	
Enabled	Always enabled. Cannot be disabled.
TCP/IP Port	The standard Modbus TCP/IP port of 502. This port is always enabled.
Modbus TCP/IP Ports 2 to 8	
Enabled (Default: No)	If selected, the DeviceMaster EIP-MOD will listen for Modbus/TCP requests on the configured TCP/IP port.
TCP/IP Port Default Port 2 = 503 Default Port 3 = 505 Default Port 4 = 506 Default Port 5 = 507 Default Port 6 = 508 Default Port 7 = 509 Default Port 8 = 510	The specified TCP/IP port(1-65535) that the DeviceMaster EIP-MOD will listen for Modbus/TCP requests on. Default ports are the first seven unassigned ports, as determined by the Internet Assigned Numbers Authority after the standard Modbus/TCP port of 502. ***Note:*** *TCP/IP ports 0, 22, 23, 80, 443, 4606 and 4607 are not allowed. Enabling TCP/IP ports other than the defaults may cause disruptions on your network. Please verify any configuration changes with your IT department.*

5.4. Remote Modbus/TCP Device Configuration

All Modbus devices not configured remotely are assumed to be local, or connected to this gateway. The location of all local Modbus devices is determined automatically.

1. Access the *Remote Modbus/TCP Device Configuration* page by clicking **Modbus | Remote Modbus Configuration**.
2. Click the **Add Remote Configuration** button to add additional remote devices.



The following table provides details about configuration options.

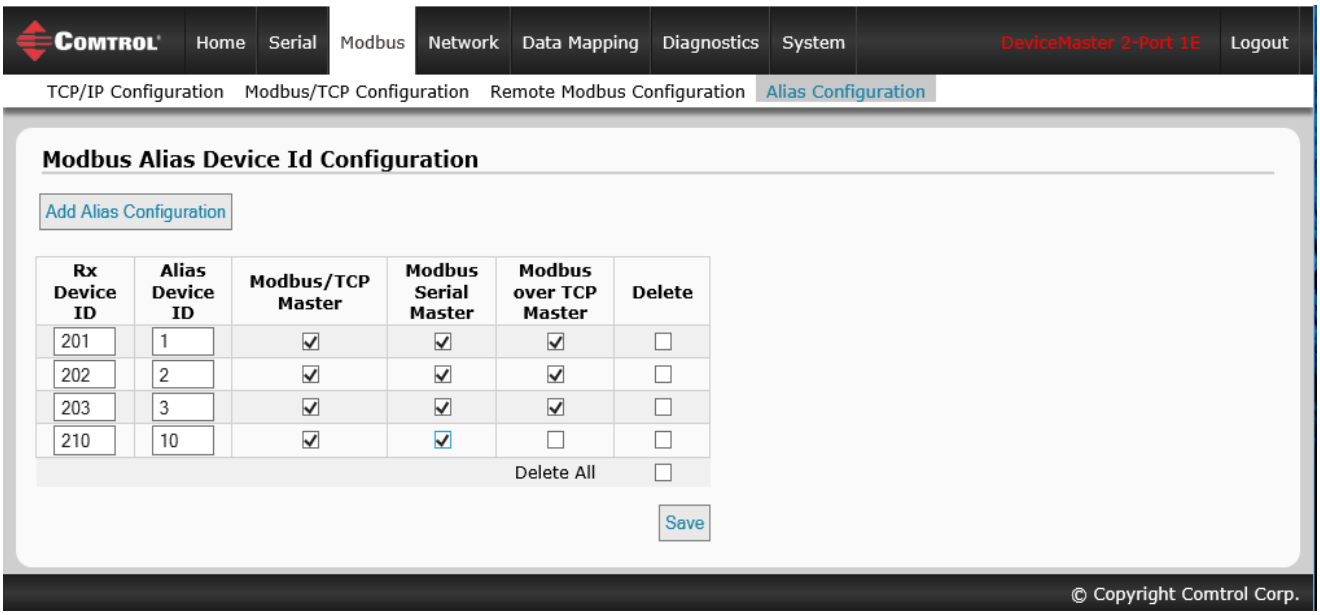
Remote Modbus/TCP Device Configuration Page	
Device ID #	The Device ID (also often called the unit ID) of the remote device must be unique. The Device ID range is 1-255. 0 means that it is not configured.
Remote IP Address	IP address of the Modbus/TCP device. All 0s means that it is not configured. ***Note:*** *The IP address of the DeviceMaster EIP-MOD will not be accepted as valid configuration data.*
Remote Modbus/TCP Port (Default = 502)	The TCP/IP port (1-65535) to connect to on the remote device.
Timeout (ms) (Default = 1000 msec)	The maximum allowable time (0 to 65535 msec) for a slave device to respond to a message before the message is considered timed out.
Dedicated Connection (Default = Off)	If selected, a dedicated Modbus/TCP connection will be used to connect to this remote device. ***Note:*** *This is most commonly used when connecting to another gateway, multiple devices are being accessed, and maximum bandwidth is desired.*
Send Writes First (Default = Off)	If selected, will forward write messages before forwarding any pending read messages. ***Note:*** *This is most commonly used when multiple messages may be outstanding for the remote device(s) and low latency for write messages is desired.*

Remote Modbus/TCP Device Configuration Page (Continued)	
Disable Broadcast Messages (Default = Off)	If selected, will disable broadcasts to this remote device. ***Note:*** *If multiple remote devices are accessed through another gateway, then this option must be selected for all remote devices configured to that gateway to prevent broadcast messages from being sent to those devices.*
Route on Pre-Alias Device ID (Default = Off)	This setting only applies to a Modbus message if the following two statements are true: • The Modbus message device ID has been aliased, or changed, as a result of a corresponding Alias Device ID configuration via the Modbus Alias Id Configuration page. • A Remote Modbus/TCP Device configuration exists for the pre-aliased, or original, device ID. If selected and all requirements are true, then the Remote Device ID configuration for the pre-aliased device ID will be applied to the Modbus message. This includes the IP address/port, timeout and control flags.
Delete	If enabled, that Device ID or IDs are deleted when you click the **Save** button.

5.5. Modbus Alias Configuration

You can configure up to four alias device IDs at one time.

1. Access the *Modbus Alias Device ID Configuration* page by clicking **Modbus | Alias Configuration**.
2. Click the **Add Alias Configuration** button.



The following configuration options apply:

Modbus Alias Device ID Configuration Page	
Rx Device ID	The device ID (also often called the unit ID) of the received message from a master. Device IDs range from 1 to 255.
Alias Device ID	The alias device ID to convert the received device ID to. Alias Device IDs range from 1 to 255.
Modbus/TCP Master (Default = Off)	If selected, this applies the alias device ID configuration to messages received from Modbus/TCP masters.
Modbus Serial Master (Default = Off)	If selected, this applies the alias device ID configuration to messages received from serial Modbus masters.
Modbus over TCP Master (Default = Off)	If selected, this applies the alias device ID configuration to messages received from Modbus RTU/ASCII over Ethernet TCP/IP masters.

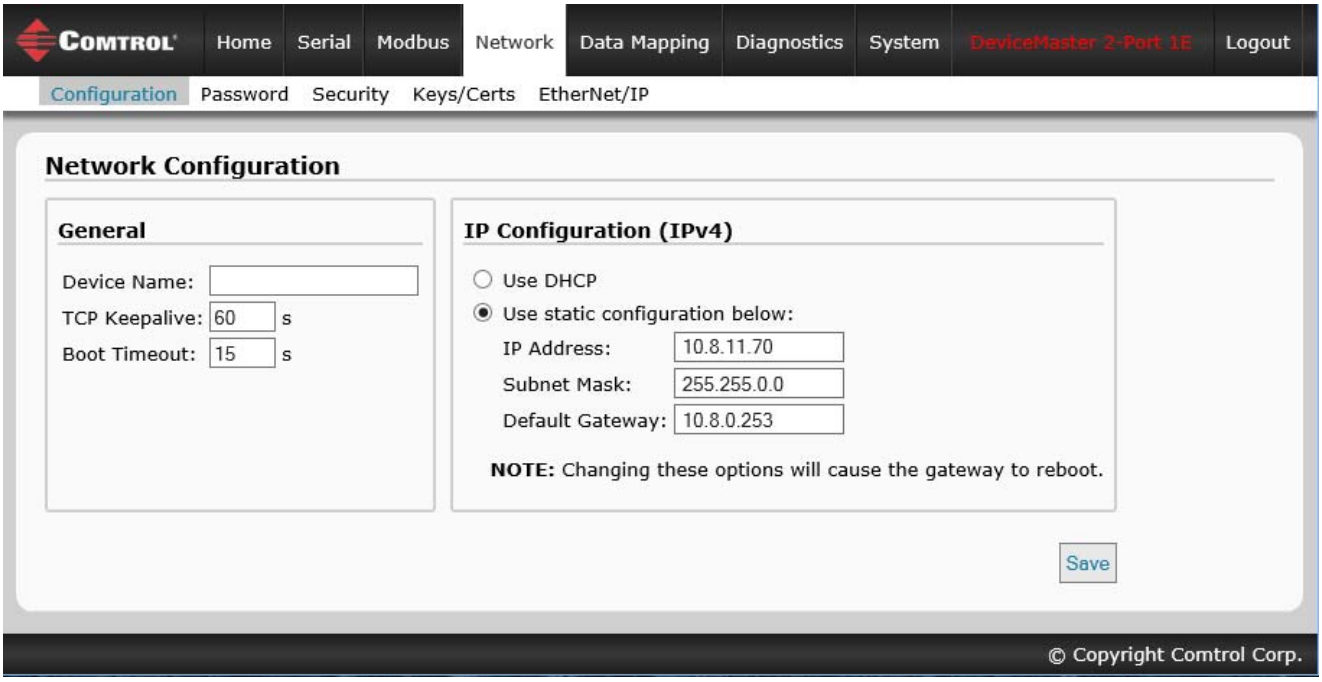
Chapter 6. Network Menus

This section discusses the pages under the *Network* menu, which includes:

6.1. Network Configuration Page

You can use the *Network Configuration* page to change the DeviceMaster EIP-MOD network configuration after using PortVision DX for initial network configuration.

Click the **Network tab** to access this page.

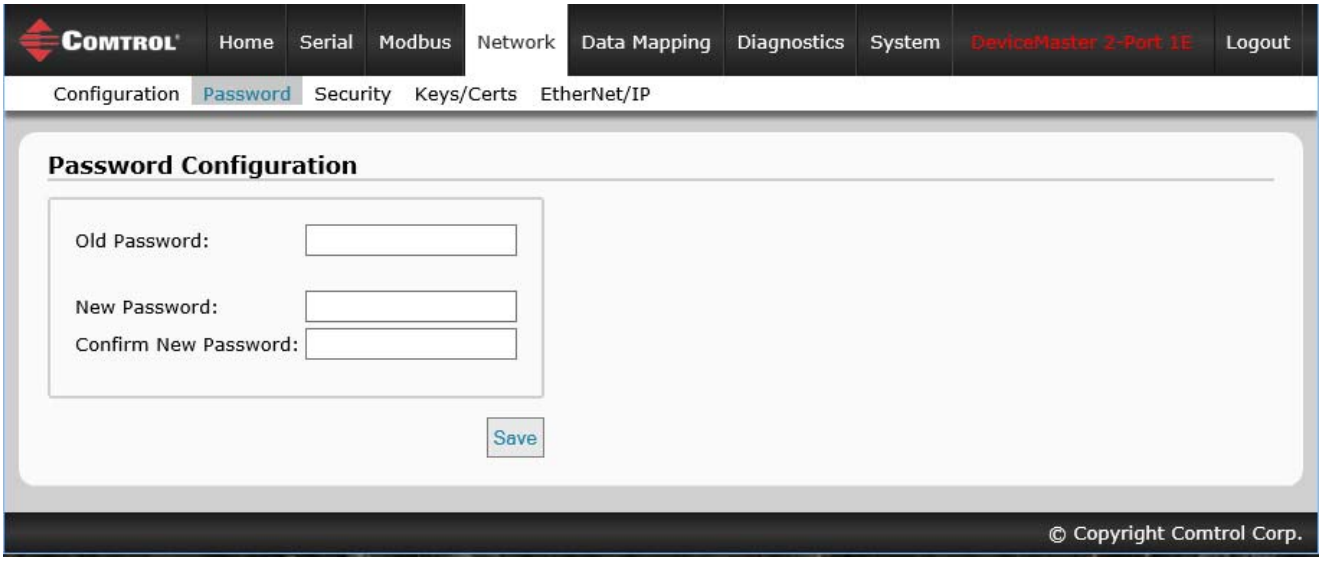


The following table provides information about the **Network Configuration** fields.

Network Configuration Page	
General	
Device Name	You can enter a 16-character **Device Name** to identify this DeviceMaster EIP-MOD on the *Home* page.
TCP Keepalive Default = 60	The TCP protocol has a keepalive feature where the two network stacks periodically ping each other to make sure the connection is still up. Upon loss of a TCP/IP connection, the network stack starts a timer. If the TCP/IP connection is still lost after the number of seconds set by the TCP Keepalive value, then the DeviceMaster EIP-MOD closes the connection and frees all the ports associated with the connection. If the DeviceMaster EIP-MOD was the originator of the first connection, it will then try to re-connect the TCP/IP connection. This allows the DeviceMaster EIP-MOD to be connected and ready to send/receive data even after a network disturbance. For most networks the default value should not need to be changed.
Boot Timeout Default = 15	Allows you to change the bootloader time-out value before the default application, EtherNet/IP-Modbus loads. You may need to increase this time-out value to 45 for compatibility with spanning tree devices (normally switches). If you change the time-out value to 0, this prevents EtherNet/IP-Modbus from loading.
IP Configuration (IPv4)	
Use DHCP	Configures the DeviceMaster EIP-MOD to use DHCPv4 mode. If you select **Use DHCP**, the IPv4 Address field below is disabled and set to 0.0.0.0. See your System Administrator to acquire a unique reserved IPv4 address if you are using DHCP. They will need the MAC address of the unit to provide you with an IPv4 address.
Use static configuration below	Configures the DeviceMaster EIP-MOD with the static IPv4 address information that you provide in the IPv4 Address, IPv4 Netmask, and IPv4 Gateway fields below. The DeviceMaster EIP-MOD is shipped from the factory with the following default IPv4 network settings: • IPv4 address = 192.168.250.250 • IPv4 Netmask = 255.255.0.0 • IPv4 Gateway address = 192.168.250.1

6.2. Password Page

You can easily set up a password to secure the DeviceMaster EIP-MOD.
There is no password set from the factory.



Use the following information to configure a password for this DeviceMaster EIP-MOD.

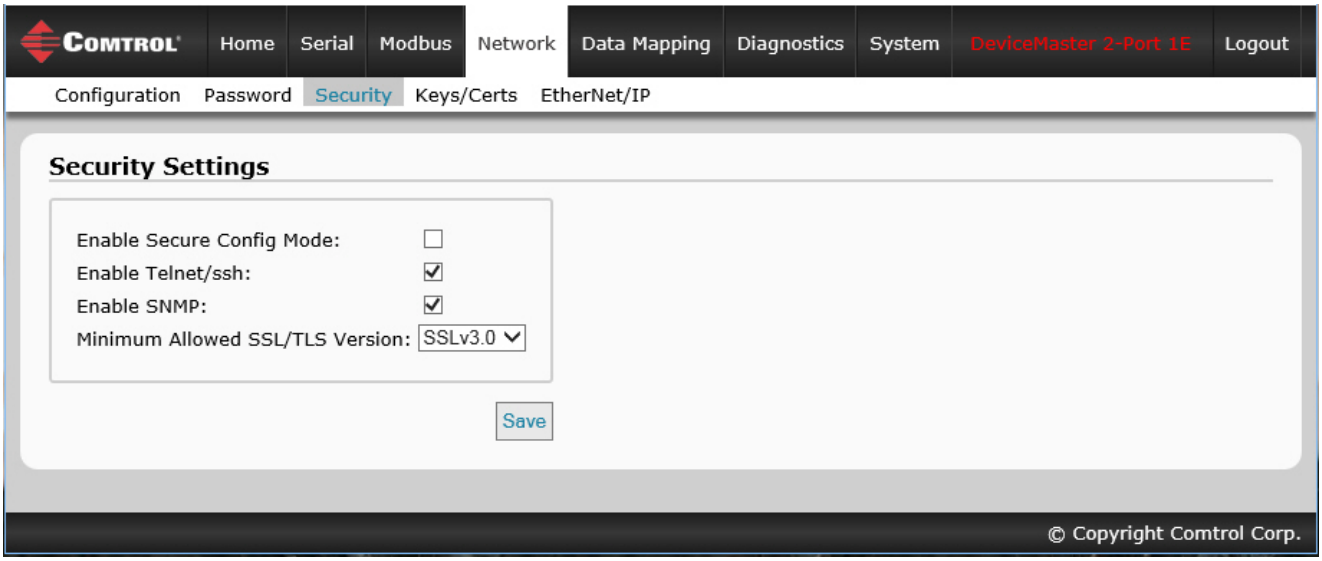
1. If necessary, click **Network | Password**.
2. If changing an existing password, enter that password in the **Old Password** field.
3. Enter a new password.
4. Enter the password in the **Confirm New Password** field.
5. Click the **Save** button.

To log into the DeviceMaster EIP-MOD, you must enter the following:

- **admin** for the username
- The configured password for the password

6.3. Security Settings Page

This table discusses *Security Settings* page options.



Security Settings Page	
Enable Secure Config Mode	If **Secure Config** mode is enabled, unencrypted access to administrative and diagnostic functions is disabled. **Secure Config** mode changes DeviceMaster EIP-MOD behavior as follows: • Telnet access to administrative and diagnostic functions is disabled. SSH access is still allowed. • Unencrypted access to the web server via port 80 (http:// URLs) is disabled. Encrypted access to the web server via port 443 (https:// URLs) is still allowed. • Administrative commands that change configuration or operating state which are received using the Pepperl+Fuchs Comtrol, Inc. proprietary TCP driver protocol on TCP port 4606 are ignored. • Administrative commands that change configuration or operating state that are received using the Pepperl+Fuchs Comtrol, Inc. MAC mode proprietary Ethernet protocol number 0x11FE are ignored.
Enable Telnet/ssh	This option enables or disables the telnet security feature after you click Save and the DeviceMaster EIP-MOD has been rebooted. This option is enabled by default.
Enable SNMP	This option enables or disables the SNMP security feature after you click **Save** and the DeviceMaster EIP-MOD has been rebooted. This option is enabled by default.
Minimum Allowed SSL/TLS Version	You can select the appropriate version for your environment. • SSLv3.0 • TLSv1.0 (default) • TLSv1.1 • TLSv1.2

Use the following steps to change security settings in the DeviceMaster EIP-MOD.

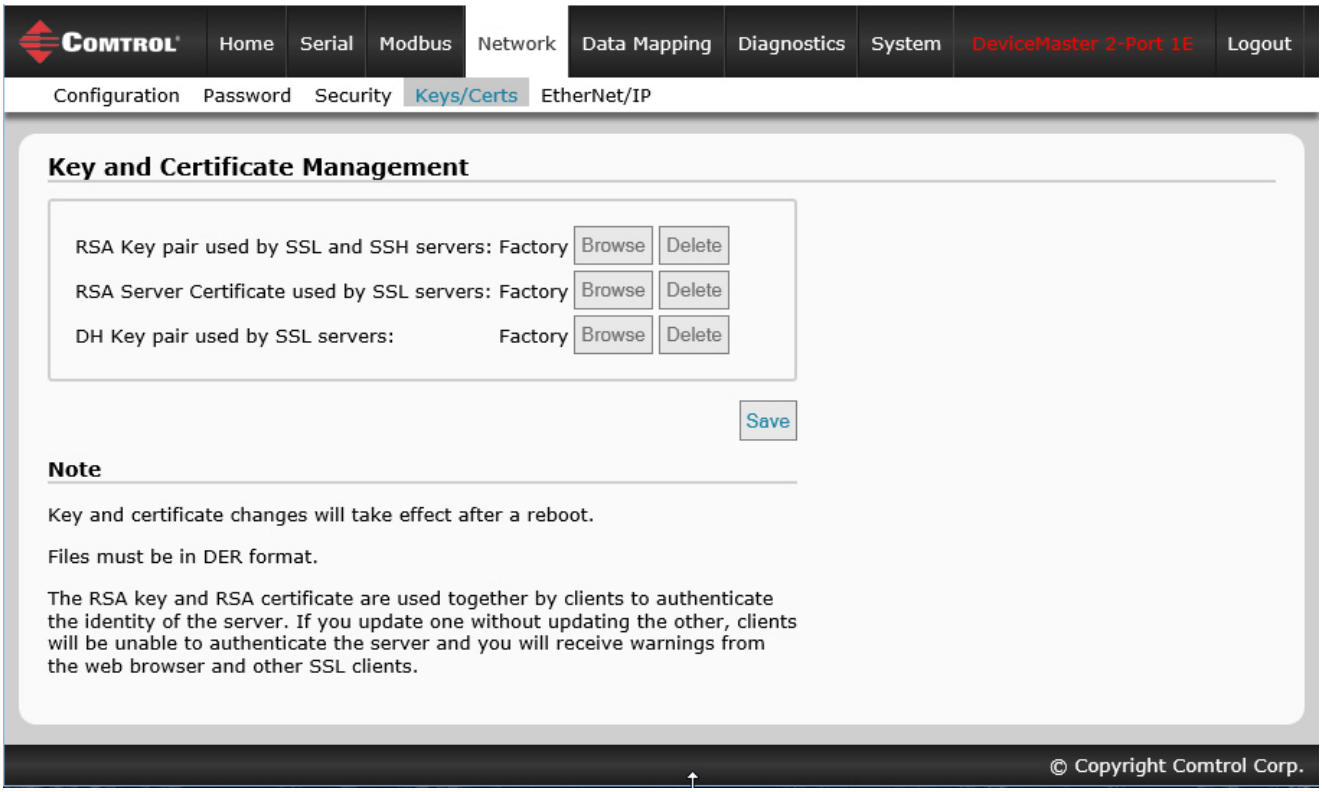
1. Click the **Network | Security**.
2. Click the appropriate check boxes in the *Security Settings* page to enable or disable security accordingly.
3. After making changes, you must click **Save**.
4. You may need to configure security keys or certificates depending on your choices using the next subsection.

6.4. Key and Certificate Management Page

For secure operation, the DeviceMaster EIP-MOD uses a set of three keys and certificates. These keys and certificates are user configurable.

Note: *All* DeviceMaster EIP-MOD *units are shipped from the factory with identical configurations. They all have the identical self-signed, Pepperl+Fuchs Comtrol, Inc. Server RSA Certificates, Server RSA Keys, and Server DH Keys.*

For maximum data and access security, you should configure all DeviceMaster EIP-MOD units with custom certificates and keys.



Key and Certificate Management Page	
RSA Key pair used by SSL and SSH servers	This is a private/public key pair that is used for two purposes: • It is used by some cipher suites to encrypt the SSL/TLS handshaking messages. Possession of the private portion of this key pair allows an eavesdropper to both decrypt traffic on SSL/TLS connections that use RSA encryption during handshaking. • It is used to sign the Server RSA Certificate in order to verify that the DeviceMaster EIP-MOD is authorized to use the server RSA identity certificate. ***Note:*** *Possession of the private portion of this key pair allows somebody to pose as the DeviceMaster EIP-MOD.* If the Server RSA Key is to be replaced, a corresponding RSA identity certificate must also be generated and uploaded or clients are not able to verify the identity certificate.
RSA Server Certificate used by SSL servers	This is the RSA identity certificate that the DeviceMaster EIP-MOD uses during SSL/TLS handshaking to identify itself. It is used most frequently by SSL server code in the DeviceMaster EIP-MOD when clients open connections to the DeviceMaster EIP-MOD's secure web server or other secure TCP ports. If a DeviceMaster EIP-MOD serial port configuration is set up to open (as a client) a TCP connection to another server device, the DeviceMaster EIP-MOD also uses this certificate to identify itself as an SSL client if requested by the server. In order to function properly, this certificate must be signed using the Server RSA Key. This means that the server RSA certificate and server RSA key must be replaced as a pair.
DH Key pair used by SSL servers	This is a private/public key pair that is used by some cipher suites to encrypt the SSL/TLS handshaking messages. ***Note:*** *Possession of the private portion of the key pair allows an eavesdropper to decrypt traffic on SSL/TLS connections that use DH encryption during handshaking.*

Use the following steps to update security keys and certificates in the DeviceMaster EIP-MOD.

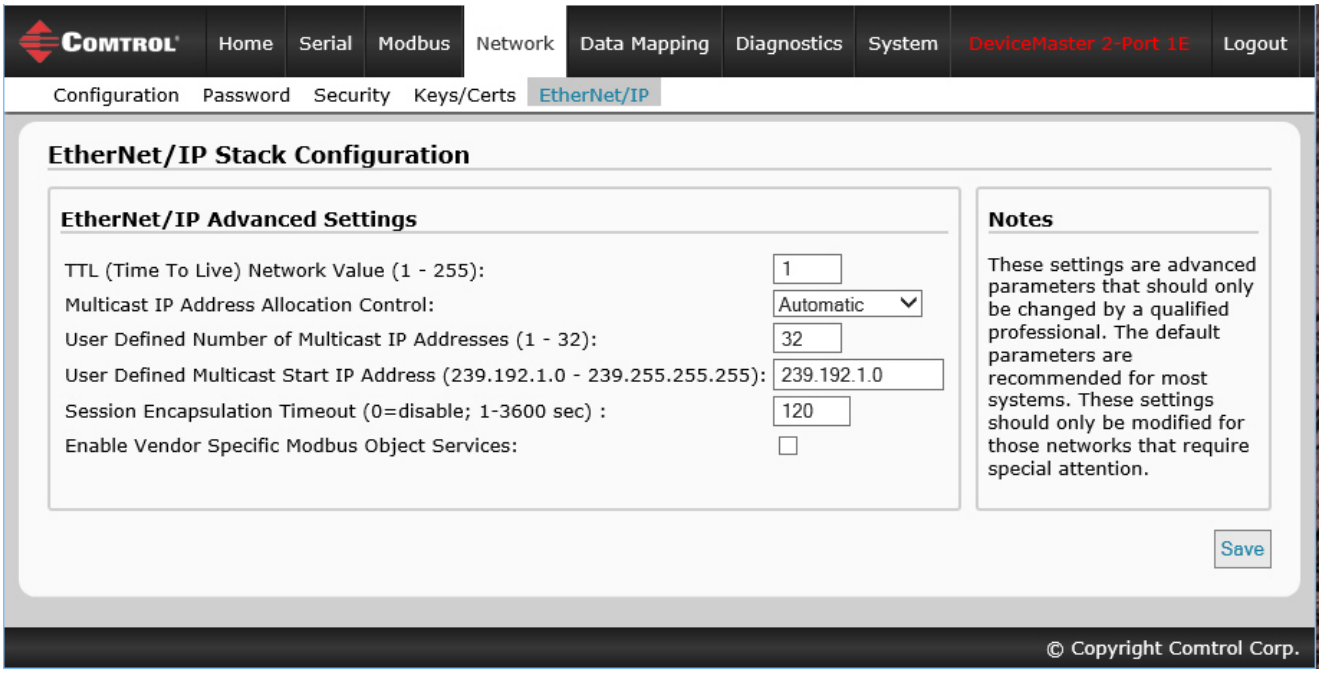
1. Click the **Network | Keys/Cert** menu.
2. Click **Browse** to locate the key or certificate file, highlight the file, and click **Open**.
3. Click **Upload** when you return to the *Key and Certificate Management* page.

   The key or certificate notation changes from **factory** or **none** to **User** when the DeviceMaster EIP-MOD is secure.

Note: *You do not need to click* **Save***, but changes will not take effect until the DeviceMaster EIP-MOD is rebooted.*

6.5. EtherNet/IP Stack Configuration

The default settings on the *EtherNet/IP Stack Configuration* page are recommended for most systems. Access the the *EtherNet/IP Stack Configuration* page by clicking **Network | EtherNet/IP**.



EtherNet/IP Stack Configuration Page	
TTL(Time To Live) Network Value (1 - 255) Default = 1	This is the network routing parameter that allows EtherNet/IP messages to be routed between different subnets. A TTL of 1, the default, allows the messages to be routed over one network “hop”. A TTL allows two hops and so on. This setting generally allows messages to be routed on the same subnet.
Multicast IP Address Allocation Control Default = Automatic	This defines how the multi-cast addresses, which are used in Class1 communications, will be allocated. • Automatic – This instructs the DeviceMaster EtherNet/IP-Modbus gateway to use the standard automatic multicast address mechanism. • User Defined – This instructs the DeviceMaster EtherNet/IP-Modbus gateway to allocate the multicast addresses based on the user settings.
User Defined Number of Multicast IP Addresses (1 - 32) Default = 32	If Multicast IP address allocation control is set to User Defined, this setting instructs the DeviceMaster EtherNet/IP-Modbus gateway to use the configured number of multicast IP addresses.
User Defined Multicast Start IP Address (239.192.1.0 - 239.255.255.255) Default = 239.192.1.0	If Multicast IP address allocation control is set to User Defined, this setting instructs the DeviceMaster EtherNet/IP-Modbus gateway where to start its multicast IP address range.

EtherNet/IP Stack Configuration Page (Continued)	
Session Encapsulation Timeout (0=disable; 1-3600 sec: Default = 120	The session encapsulation timeout instructs the DeviceMaster EtherNet/IP-Modbus gateway how long to wait with no activity before timing out a session.
Enable Vendor Specific Modbus Object Services Default = Disabled	If selected, The DeviceMaster EtherNet/IP-Modbus gateway provides support for the Vendor Specific Modbus Object Services. The services include the following: • 70 hex – Write Single Coil • 71 hex – Write Single Register • 72 hex – Mask Write Register • 73 hex – Read/Write Holding Registers

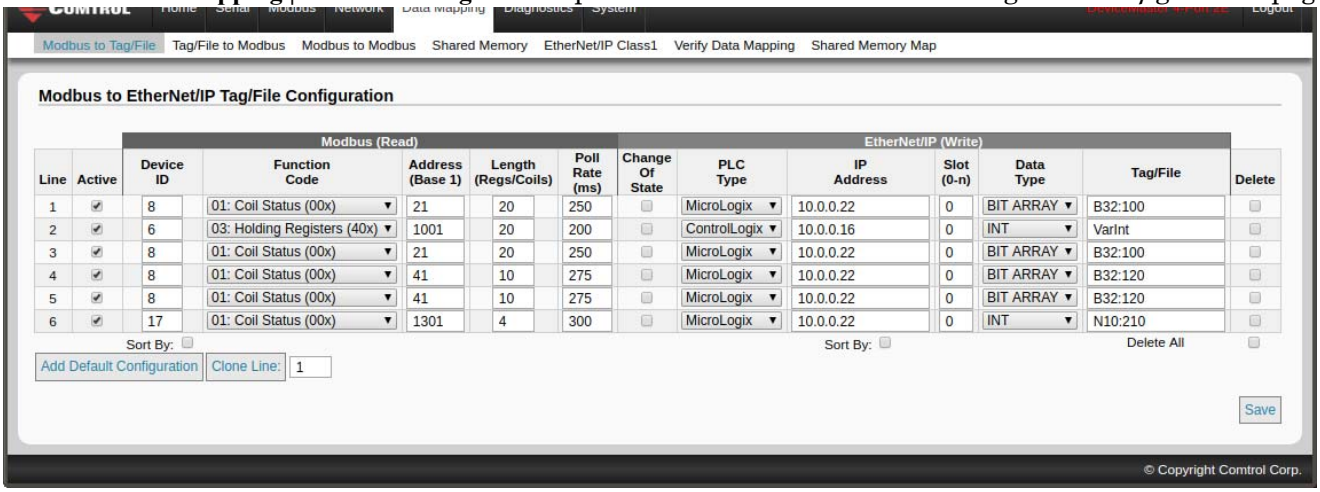
Chapter 7. Data Mapping Menus

This chapter discusses the following web pages:

7.1. Modbus to EtherNet/IP Tag/File Configuration Page

Use the following information to configure Modbus to EtherNet/IP tag/file.

1. Click **Data Mapping | Modbus to Tag/File** to open the *Modbus to EtherNet/IP Tag/File Configuration* page.



2. Click the **Add Default Configuration** button.

The following table provides information about the options on this page.

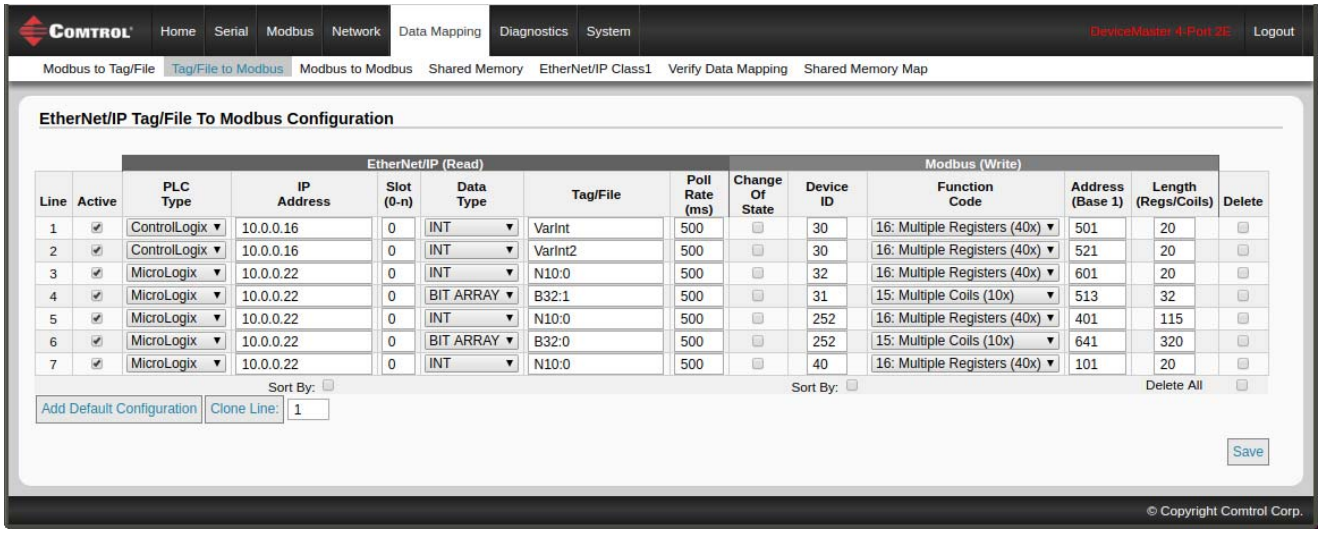
Modbus to EtherNet/IP Tag/File Configuration Page	
Line	This is the configuration line number. • If the configuration entry has been saved, the line number is not directly modifiable. • If the configuration entry is in the process of being added and has not been saved, then the line number can be set to anywhere in the configuration list. - The placement of the entry in the saved list is in relation to the current list of saved entries. For example, if you wish to place an entry before the current 4th entry, then enter 4 for the line number. - If more than one entry is added at one time, the order is preserved but final line numbers may or may not be the same as the selected numbers.
Active (Default = Enabled)	If selected, the configuration becomes active when the **Save** button is clicked on. Data Mapping immediately begins to perform the configured operations. If not selected, the configuration becomes inactive when the **Save** button is clicked. Data Mapping then ignores the configured operations.
Modbus (Read)	
Device ID (Default = 1)	The Modbus Device ID to be read. • If access to the Shared Memory is desired: - The Shared Memory must be enabled. - The Shared Memory device ID must be entered. • If a Device Id for a Modbus slave is entered, the DeviceMaster EtherNet/IP-Modbus gateway routes the message to the appropriate location.
Function Code (Default = 03: Holding Registers (40x))	Select the Modbus Read function code: • 01: Coil Status (00x) - read one or more coils • 02: Input Status (10x) - read one or more discrete inputs • 03: Holding Registers (40x) - read one or more holding registers • 04: Input Registers (30x) - read one or more input registers
Address (Base 1) (Default = 1)	Enter the Modbus address in Base 1 format. (The address of the tables starts at 1, instead of 0 for Base 0). Enter only the lower 16 bits of the address (1-65536).
Length (Regs/Coils) (Default = 1)	Enter the number of registers or coils to be read.
Poll Rate (ms) (Default = 2000 ms)	Enter the rate at which the Data Mapping process should read the configured Modbus device or Shared Memory.
EtherNet/IP (Write)	
Change Of State (Default = Disabled)	If selected, the Data Mapping process only writes the received data to the EtherNet/IP location if: • The data is being received for the first time. • The received data has changed. • The previous write attempt to the EtherNet/IP location was unsuccessful.

Modbus to EtherNet/IP Tag/File Configuration Page (Continued)	
PLC Type (Default = ControlLogix)	Specifies the PLC type to write to: • ControlLogix – This selection enables the full CIP supported interfaces that provide CIP table read/write capabilities. The ControlLogix PLC family of PLCs includes the Rockwell ControlLogix, FlexLogix, SoftLogix and other advanced PLCs. This selection also supports numerous PLCs manufactured by other manufacturers. • MicroLogix – This selection provides support for Rockwell MicroLogix, SLC and PLC-5 PLCs. These PLCs only support a limited EtherNet/IP interface that does not include CIP table read/write support. Only Write-to-File and Read-from-File communication that use PCCC MSG instructions is provided.
IP Address (Default = 0.0.0.0)	Enter the IP address of the EtherNet/IP PLC in xxx.xxx.xxx.xxx format.
Slot (0-n) (Default = 0)	Enter the slot number of the PLC controller board. This is only required for chassis type PLCs such as ControlLogix. For all other PLCs, set this value to 0.
Data Type (Default = INT)	The data type of the Tag or File to write to. • SINT – otherwise called 8 bit, or byte sized data • DINT – otherwise called 16 bit, or WORD format • BOOL – a single Boolean value • BIT ARRAY – an array of Boolean values • FLOAT – floating point, otherwise called REAL
Tag/File (Default = Blank)	Specifies the EtherNet/IP PLC tag or file name. It indicates where to write the data read from the Modbus device or Shared Memory. This entry supports ASCII strings containing up to 40 characters. File names for the PLC-5/ SLC PLCs must be in a SLC/PLC-5 file format (i.e. **N10:0**, **B20:0/6**, etc).
Save	When this button is clicked: • The settings are verified. • Any changes are saved. • The Data Mapping process is immediately reconfigured and begins performing the configured operations.
Delete	Enable the line or lines that you want to delete and click the **Save** button.
Delete All	Enable the **Delete All** option and click the **Save** button to remove all entries.
Add Default Configuration	Adds a default entry to the end of the list. The entry will not become active until it is saved by clicking **Save**.
Clone Line	Adds a new configuration entry to the end of the list that is identical to the entered line number. The entry will not become active until it is saved by clicking **Save**. ***Note:*** *No action is taken if the entered line number is invalid.*
Sort By (Read Device ID) (Default = Disabled)	Reorders the list based on the Read Device ID when **Save** is selected.
Sort By (IP Address) (Default = Disabled)	Reorders the list based on the EtherNet/IP PLC IP address when **Save** is selected.

7.2. EtherNet/IP Tag/File to Modbus Configuration Page

Need supporting information.

1. Click **Data Mapping | Tag/File to Modbus** to open the *EtherNet/IP Tag/File To Modbus Configuration* page.
2. Click the **Add Default Configuration** button.



The following table provides information about the options on this page.

EtherNet/IP Tag/File To Modbus Configuration Page	
Line	This is the configuration line number. • If the configuration entry has been saved, the line number is not directly modifiable. • If the configuration entry is in the process of being added and has not been saved, then the line number can be set to anywhere in the configuration list. - The placement of the entry in the saved list is in relation to the current list of saved entries. For example, if you wish to place an entry before the current 4th entry, then enter 4 for the line number. - If more than one entry is added at one time, the order is preserved but final line numbers may or may not be the same as the selected numbers.
Active (Default = Enabled)	If selected, the configuration becomes active when the **Save** button is clicked. The Data Mapping process immediately begins to perform the configured operations. If not selected, the configuration becomes inactive when the **Save** button is clicked. The Data Mapping process then ignores the configured operations.

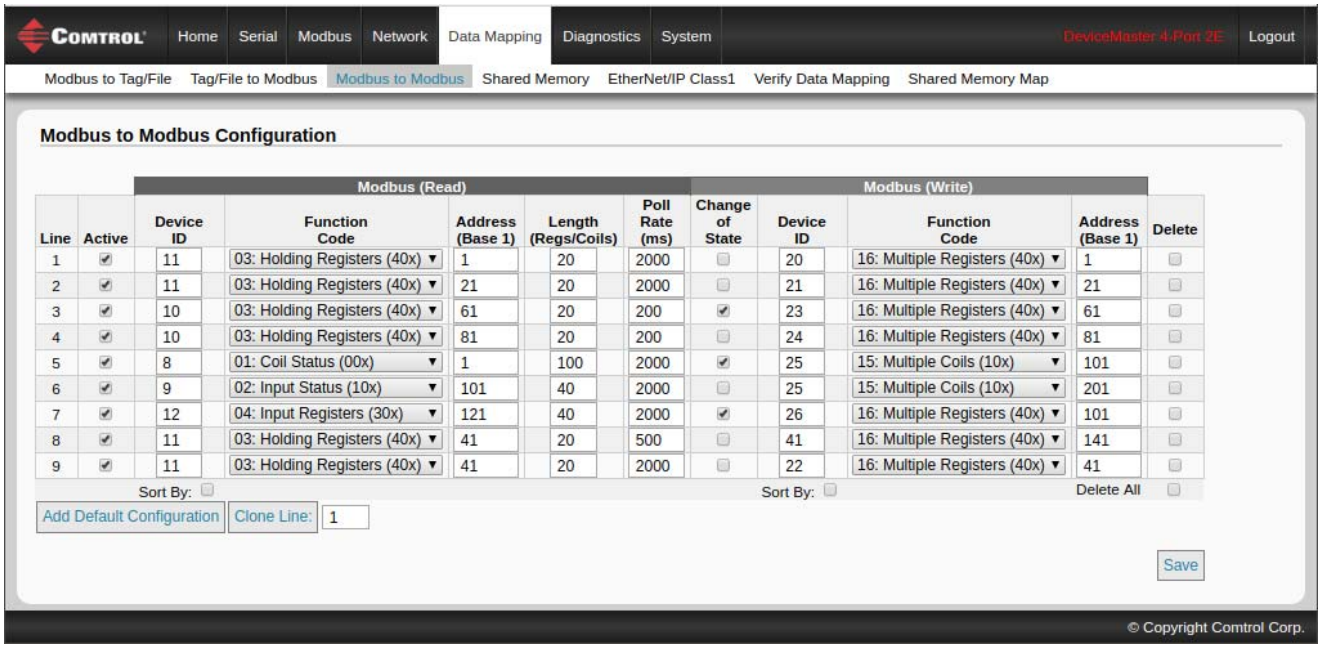
EtherNet/IP Tag/File To Modbus Configuration Page (Continued)	
EtherNet/IP (Read)	
PLC Type (Default = ControlLogix)	Specifies the type of PLC to read from: • ControlLogix – This selection enables the full CIP supported interfaces that provide CIP table read/write capabilities. The ControlLogix PLC family of PLCs includes the Rockwell ControlLogix, FlexLogix, SoftLogix and other advanced PLCs. This selection also supports numerous PLCs manufactured by other manufacturers. • MicroLogix This selection provides support for Rockwell MicroLogix, SLC and PLC-5 PLCs. These PLCs only support a limited EtherNet/IP interface that does not include CIP table read/write support. Only Write-to-File and Read-from-File communication that use PCCC MSG instructions is provided.
IP Address (Default = 0.0.0.0)	Enter the IP address of the EtherNet/IP PLC in xxx.xxx.xxx.xxx format.
Slot (0-n) (Default = 0)	Enter the slot number of the PLC controller board. This is only required for chassis type PLCs such as ControlLogix. For all other PLCs, set this value to 0.
Data Type (Default = INT)	The data type of the Tag or File to read from. • SINT – otherwise called 8 bit, or byte sized data • INT – otherwise called 16 bit, or WORD format • DINT – otherwise called 32 bit or double word format • BOOL – a single Boolean value • BIT_ARRAY – an array of Boolean values • FLOAT – floating point, otherwise called REAL
Tag/File (Default = Blank)	Specifies the EtherNet/IP PLC tag or file name. It indicates where to read the PLC. This entry supports ASCII strings containing up to 40 characters. File names for the PLC-5/ SLC PLCs must be in a SLC/PLC-5 file format (i.e. **N10:0**, **B20:0/6**, etc.).
Modbus (Write)	
Poll Rate (ms) (Default = 2000 ms)	Enter the rate at which the Data Mapping process should read the configured EtherNet/IP PLC tag or file.
Change Of State (Default = Disabled)	If selected, the Data Mapping process will only write the received data to the Modbus Device if: • The data is being received for the first time. • The received data has changed. • The previous write attempt to the Modbus Device was unsuccessful.
Device ID (Default = 1)	The Modbus Device ID to be written to. • If access to the Shared Memory is desired: - The Shared Memory must be enabled. - The Shared Memory device ID must be entered. • If a Device Id for a Modbus slave is entered, the DeviceMaster EtherNet/IP-Modbus gateway routes the message to the appropriate location.
Function Code (Default = 03: Holding Registers (40x))	Select the Modbus Write function code: • 05: Single Coil (10x) - write one coil • 06: Single Register (40x) – write one holding register • 15: Multiple Coils (10x) – write one or more coils • 16: Multiple Registers (40x) – write one or more holding registers

EtherNet/IP Tag/File To Modbus Configuration Page (Continued)	
Address (Base 1) (Default = 1)	Enter the Modbus address in Base 1 format. (The address of the tables starts at 1, instead of 0 for Base 0). Enter only the lower 16 bits of the address (1-65536).
Length (Regs/Coils) (Default = 1)	Enter the number of registers or coils to be read.
Save	When this button is clicked: • The settings are verified. • Any changes are saved. • The Data Mapping process is immediately reconfigured and begins performing the configured operations.
Delete	Enable the line or lines that you want to delete and click the **Save** button.
Delete All	Enable the **Delete All** option and click the **Save** button to remove all entries.
Add Default Configuration	Adds a default entry to the end of the list. The entry will not become active until it is saved by clicking **Save**.
Clone Line	Adds a new configuration entry to the end of the list that is identical to the entered line number. The entry will not become active until it is saved by clicking **Save**. ***Note:*** *No action is taken if the entered line number is invalid.*
Sort By (IP Address) (Default = Disabled)	Reorders the list based on the EtherNet/IP PLC IP address when **Save** is selected. Sort By (Write Device ID) Will reorder the list based on the Write Device ID when **Save** is selected.
Sort By (Write Device ID) (Default = Disabled)	Reorders the list based on the Write Device ID when **Save** is selected.

7.3. Modbus to Modbus Configuration Page

Need supporting information.

1. Click **Data Mapping | Modbus to Modbus** to open the *Modbus to Modbus Configuration* page.
2. Click the **Add Default Configuration** button.



The following table provides information about the options on this page.

Modbus to Modbus Configuration Page	
Line	This is the configuration line number. • If the configuration entry has been saved, the line number is not directly modifiable. • If the configuration entry is in the process of being added and has not been saved, then the line number can be set to anywhere in the configuration list. - The placement of the entry in the saved list is in relation to the current list of saved entries. For example, if you wish to place an entry before the current 4th entry, then enter 4 for the line number. - If more than one entry is added at one time, the order is preserved but final line numbers may or may not be the same as the selected numbers.
Active	If selected, the configuration becomes active when the **Save** button is clicked. The Data Mapping process immediately begins to perform the configured operations. If not selected, the configuration becomes inactive when the **Save** button is clicked. The Data Mapping process then ignores the configured operations.

Modbus to Modbus Configuration Page (Continued)	
Modbus (Read)	
Device ID	The Modbus Device ID to be read. • If access to the Shared Memory is desired: - The Shared Memory must be enabled. - The Shared Memory device ID must be entered. • If a Device Id for a Modbus slave is entered, the DeviceMaster EtherNet/IP-Modbus gateway routes the message to the appropriate location.
Function Code	Select the Modbus Read function code: • 01: Coil Status (00x) - read one or more coils • 02: Input Status (10x) – read one or more discrete inputs • 03: Holding Registers (40x) - read one or more holding registers • 04: Input Registers (30x) – read one or more input registers
Address (Base 1)	Enter the Modbus address in Base 1 format. (The address of the tables starts at 1, instead of 0 for Base 0). Enter only the lower 16 bits of the address (1-65536).
Length (Regs/Coils)	Enter the number of registers or coils to be read.
Poll Rate (ms)	Enter the rate at which the Data Mapping process should read the configured Modbus device or Shared Memory.
Modbus (Write)	
Change of State	If selected, the Data Mapping process only writes the received data to the write Modbus Device if: • The data is being received for the first time. • The received data has changed. • The previous write attempt to the write Modbus Device was unsuccessful.
Device ID	The Modbus Device ID to be written to. • If access to the Shared Memory is desired: - The Shared Memory must be enabled. - The Shared Memory device ID must be entered. • If a Device ID for a Modbus slave is entered, the DeviceMaster EtherNet/IP-Modbus gateway routes the message to the appropriate location.
Function Code	Select the Modbus Write function code: • 05: Single Coil (10x) - write one coil • 06: Single Register (40x) – write one holding register • 15: Multiple Coils (30x) – write one or more coils • 16: Multiple Registers (40x) – write one or more holding registers
Address (Base 1)	Enter the Modbus address in Base 1 format. (The address of the tables starts at 1, instead of 0 for Base 0). Enter only the lower 16 bits of the address (1-65536).
Length (Regs/Coils)	Enter the number of registers or coils to be read.

Modbus to Modbus Configuration Page (Continued)	
Save	When this button is clicked: • The settings are verified. • Any changes are saved. • The Data Mapping process immediately is reconfigured and begins performing the configured operations.
Delete	If selected, the entry is deleted from the list when **Save** is selected.
Delete All	If selected, the entire list will be deleted when **Save** is selected.
Add Default Configuration	Adds a default entry to the end of the list. The entry will not become active until it is saved by clicking **Save**.
Clone Line	Add a new configuration entry to the end of the list that is identical to the entered line number. The entry will not become active until it is saved by clicking **Save**. ***Note:*** *No action will be taken if the entered line number is invalid.*
Sort By (Read Device ID)	Reorders the list based on the Read Device ID when **Save** is selected.
Sort By (Write Device ID)	Reorders the list based on the Write Device ID when **Save** is selected.

7.4. Shared Memory Configuration Page

The Shared Memory functionality enables multiple methods for communicating between a wide variety of EtherNet/IP controllers, Modbus masters and Modbus slave devices.

The Shared Memory interface contains eight 200 Holding Register blocks and eight 320 Coil blocks.

- EtherNet/IP controllers can access the Shared Memory through Class1 connections, the Modbus Object, and Write-to-Tag/File and Read-from-Tag/File interfaces.
- All Modbus masters, (Modbus/TCP, serial Modbus RTU/ASCII, and Modbus RTU/ASCII over Ethernet TCP/IP), can read the contents of the Shared Memory blocks.
- Data Mapping configurations can read Shared Memory blocks.
- Write access can be controlled to each Holding Register and Coil block. Each block can be configured to:
  - Provide all masters, expect Class1, write access
  - Can be restricted to:
    - A port-specific serial master
    - A Modbus/TCP master
    - An Ethernet TCP/IP master
    - EtherNet/IP Class1
    - Modbus Object message(s)
    - Tag/File to Modbus Data Mapping configuration(s).
    - Modbus to Modbus Data Mapping configuration(s)
- The Shared Memory contents can be displayed and cleared via the embedded web pages.
- Diagnostics for each block include read, write and blocked write message counts.
- Blocked write messages are recorded in the Write Violation Log.
- Special EtherNet/IP Class1 handling:
  - In order to read from a Shared Memory block via a Class1 connection, the shared memory block must be Class1 Read Enabled. The Class1 connection receive data contents and size are determined by which Shared Memory blocks are read enabled.
  - In order to write to a Shared Memory block via a Class1 connection, the shared memory block Write Master(s) setting must be set to EIP Class1. Only then may an EtherNet/IP Class1 connection write to the Shared Memory. That is because the Class1 connection writes continually to the block and over write any data written by another controller.

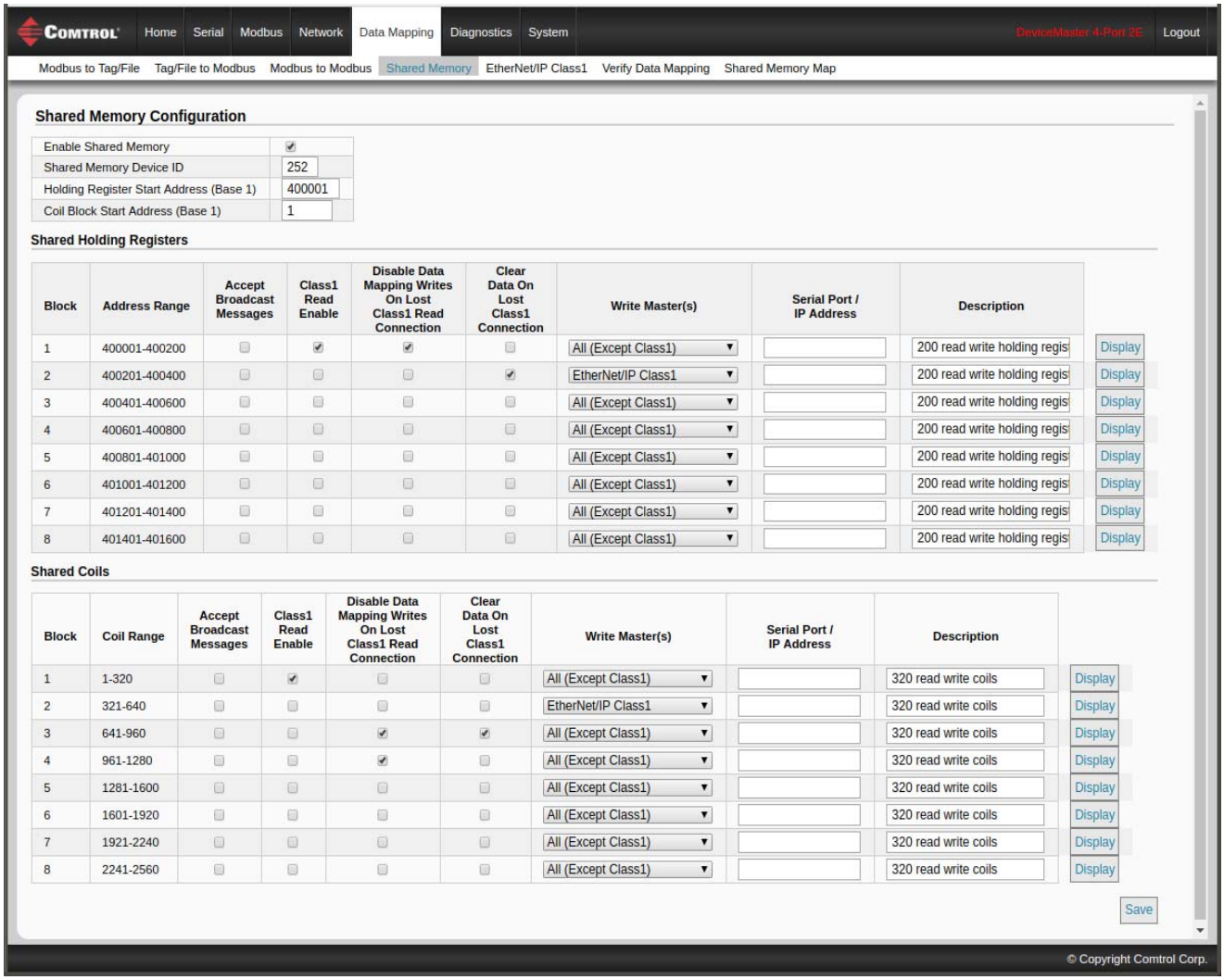
This table displays the supported Holding Register Block Function Codes:

Function Code	Description
3	Read Holding Registers
4	Read Input Registers
6	Write Single Register
16	Write Multiple Registers
22	Write Mask Register
23	Read Write Registers

This table shows the supported Coil Block Function Codes:

Function Code	Description
1	Read Coils
2	Read Discrete Inputs
5	Write Single Coil
15	Write Multiple Coils

Click **Data Mapping | Shared Memory** to open the *Shared Memory Configuration* page.



The following table provides details about the configuration options. Click the **Display** button to view detailed information about a specific block or coil.

Shared Memory Configuration Page	
Enable Shared Memory (Default = Off)	If selected, enables the Shared Memory functionality.
Shared Memory Device ID (Default = 252)	The Device ID (also often called the unit ID) of the Shared Memory must be unique within the public Modbus network. The Device ID range is 1 to 255.
Holding Register Start Address (Base 1) (Default = 400001)	Specifies the starting address of the Shared Memory Holding Register blocks. The range is 400001 to 463935.
Coil Block Start Address (Base 1) (Default = 1)	Specifies the starting address of the Shared Memory Coil blocks. The range is 1 to 64255.
Shared Holding Registers	
Block	Specifies the block number.
Address Range	Specifies the block address range.
Accept Broadcast Messages (Default = Disabled)	If selected, the Shared Memory block(s) will accept broadcast messages addressed to their memory block(s).
Class1 Read Enable Default = Block 1 enabled and Blocks 2 to 8 are disabled	If selected, the Shared Memory block is added to the Class1 to the Active Rx Class 1 Configuration.
Disable Data Mapping Writes On Lost Class1 Read Connection (Default = Disabled)	If selected, the Data Mapping process performs writes to this Shared Memory block only when there is an active Class1 read connection to this block.
Clear Data On Lost Class1 Connection (Default = Disabled)	If selected, the Shared Memory block's data are cleared when all Class1 connections to this block are lost.

Shared Memory Configuration Page (Continued)	
Write Master(s) Defaults: **Block 2: EIP Class1** Other blocks: **All (Except Class1)**	Indicates which master(s) have write access to the Shared Memory block. • **All (Except Class1)** – Except for Class1 connections, all Modbus masters, Modbus Object and Data Mapping functions have write access to the block. • Port specific serial master: - **Port-1** - **Port-2** (2-port and 4-port models only) - **Port-3** (4-port models only) - **Port-4** (4-port models only) • **Modbus/TCP** - Modbus/TCP master(s) at a specified IP address • **Ethernet TCP/IP** – Ethernet TCP/IP master(s) at a specified IP address • **EIP Class1** – EtherNet/IP Class1 connection. Only one EtherNet/IP Class connection can access the block at one time. • **EIP TAG/FILE** – Tag/File to Modbus configurations at a specified IP address • **EIP Modbus Object** – Messages via the Modbus Object from an EtherNet/IP controller at a specified IP address • **Modbus to Modbus** – Modbus to Modbus configuration(s)
Serial Port / IP Address (Default = blank)	IP address of the Modbus/TCP master, Ethernet TCP/IP master or EtherNet/IP controller. Zeros indicate that there is no configuration. ***Note:*** *Does not apply to All (Except Class1), port-specific serial masters, Class1, or Modbus to Modbus configuration(s)..*
Description (Default = 200 read write holding registers)	User-defined description of the Shared Memory block. ASCII string with a maximum of 80 characters in length.
Shared Coils	
Block	Specifies the block number.
Coil Range	Specifies the coil range.
Accept Broadcast Messages (Default = Disabled)	If selected, the Shared Memory block(s) accept broadcast messages from Modbus masters addressed to their memory block(s).
Class1 Read Enable Default = Block 1 enabled Blocks 2 to 8 disabled	If selected, the Shared Memory block is added to the Class1 to the Active Rx Class 1 Configuration.
Disable Data Mapping Writes On Lost Class1 Read Connection (Default = Disabled)	If selected, the Data Mapping process performs writes to this Shared Memory block only when there is an active Class1 read connection to this block.
Clear Data On Lost Class1 Connection (Default = Disabled)	If selected, the Shared Memory block's data is cleared when all Class1 connections to this block are lost.

Shared Memory Configuration Page (Continued)	
Write Master(s) Defaults: **Block 2: EIP Class 1** Other blocks: **All (Except Class1)**	Indicates which master(s) have write access to the Shared Memory block. • **All (Except Class1)** – Except for Class1 connections, all Modbus masters, Modbus Object and Data Mapping functions have write access to the block. • Port specific serial master: - **Port-1** - **Port-2** (2-port and 4-port models only) - **Port-3** (4-port models only) - **Port-4** (4-port models only) • **Modbus/TCP** - Modbus/TCP master(s) at a specified IP address • Ethernet TCP/IP – Ethernet TCP/IP master(s) at a specified IP address • **EIP Class1** – EtherNet/IP Class1 connection. Only one EtherNet/IP Class connection can access the block at one time. • **EIP TAG/FILE** – Tag/File to Modbus configurations at a specified IP address • **EIP Modbus Object** – Messages via the Modbus Object from an EtherNet/IP controller at a specified IP address • **Modbus to Modbus** – Modbus to Modbus configuration(s)
Serial Port / IP Address (Default = Blank)	IP address of the Modbus/TCP master, Ethernet TCP/IP master or EtherNet/IP controller. Zeros indicate that there is no configuration. ***Note:*** *Does not apply to All (Except Class1), port-specific serial masters, Class1, or Modbus to Modbus configuration(s).*
Description (Default = 200 read write holding registers)	User-defined description of the Shared Memory block. ASCII string with a maximum of 80 characters in length.

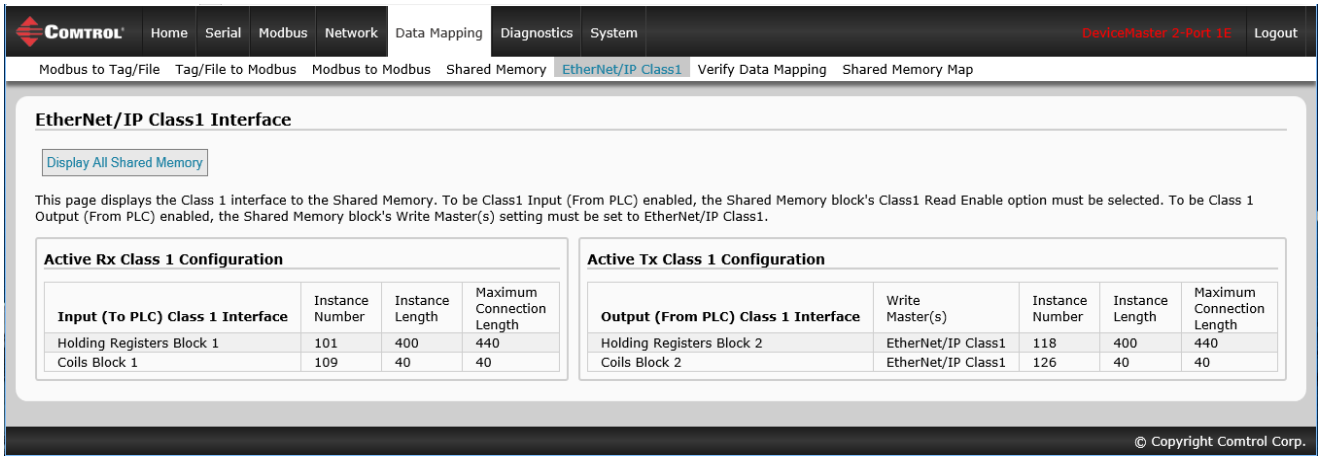
7.5. EtherNet/IP Class 1 Page

The DeviceMaster EIP-MOD provides a highly informative Class1 interface web page designed to provide easily understand information to aid the PLC programmer:

- Instance numbers
- Data offsets for each instance
- Instance and connection lengths

7.5.1. Active Class1 Interface Page

This page displays the current active Class1 configuration. This configuration is determined by the Shared Memory Configuration. Assembly instances are enabled only if they are enabled for Class1 in the Shared Memory Configuration.

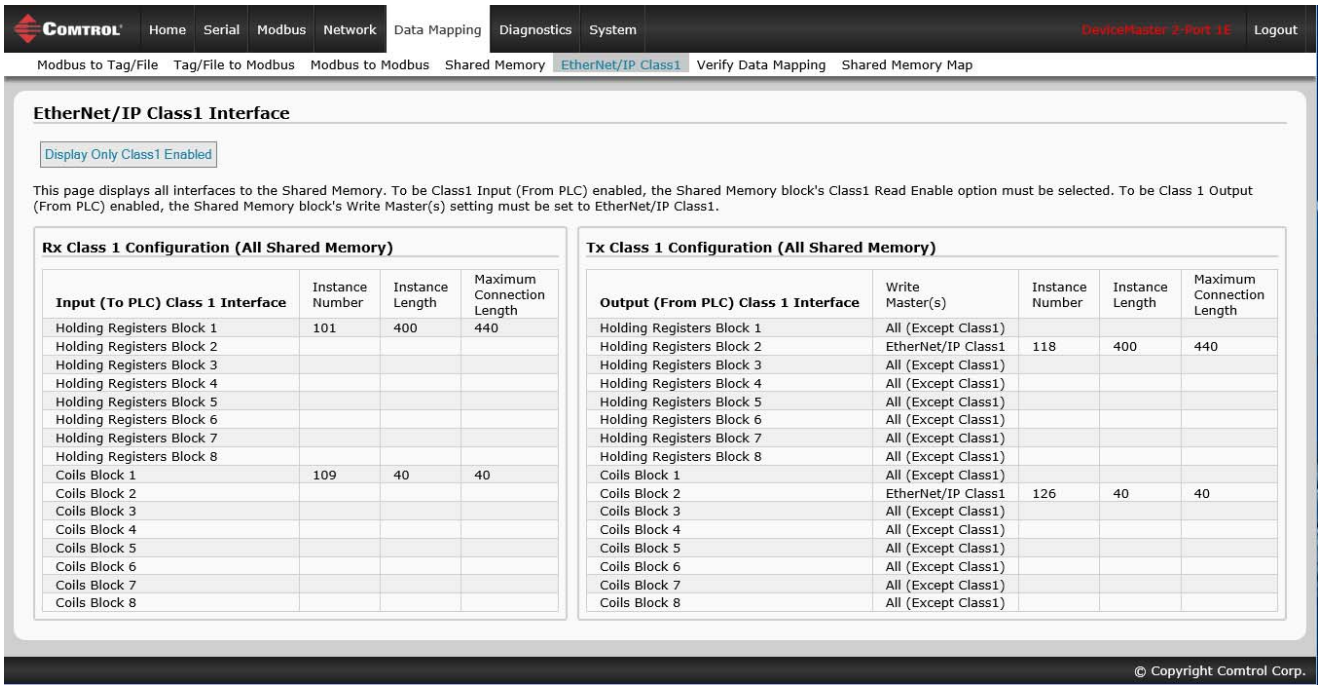


The following table provides information about this page.

EtherNet/IP Class 1 Interface Page	
Active Rx Class 1 Configuration - Input (To PLC) Class 1 Interface	
Holding Registers Block 1 - 8	Displays only Read Enabled memory blocks. The rows display the Instance Number, Instance Length, and Maximum Connection length for the corresponding Rx Holding Register Block.
Coils Block 1 - 8	Displays only Read Enabled memory blocks. The rows display the Instance Number, Instance Length, and Maximum Connection length for the corresponding Rx Coil Block.
Active Tx Class 1 Configuration - Output (From PLC) Class 1 Interface	
Holding Registers Block 1 - 8	Displays only Write Enabled memory blocks. The rows display the Instance Number, Instance Length, and Maximum Connection length for the corresponding Tx Holding Register Block.
Coils Block 1 - 8	Displays only Write Enabled memory blocks. The rows display the Instance Number, Instance Length, and Maximum Connection length for the corresponding Tx Coil Block.

7.5.2. All Shared Memory Class1 Interface Page

If the **Display All Shared Memory** button is clicked, the following page displays. This page is provided to show the relationships of the active Class1 interface with the rest of the Shared Memory blocks.



The following table provides information about this page.

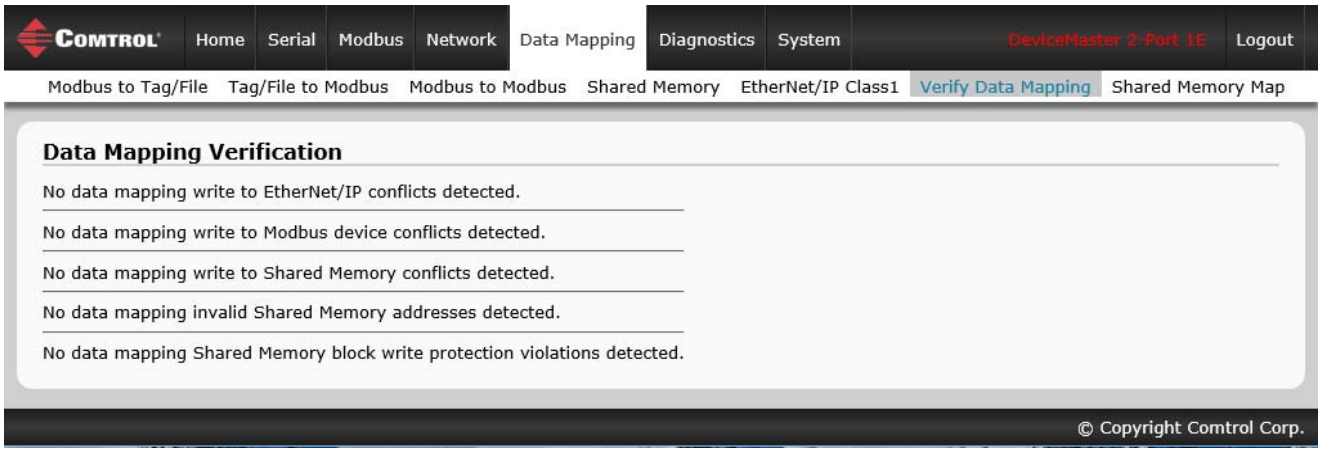
EtherNet/IP Class 1 Interface Page	
Active Rx Class 1 Configuration - Input (To PLC) Class 1 Interface	
Holding Registers Block 1 - 8	Displays all shared memory blocks. The rows display the Instance Number, Instance Length, and Maximum Connection length for the corresponding Rx Holding Register Block.
Coils Block 1 - 8	Displays all shared memory blocks. The rows display the Instance Number, Instance Length, and Maximum Connection length for the corresponding Rx Coil Block.
Active Tx Class 1 Configuration - Output (From PLC) Class 1 Interface	
Holding Registers Block 1 - 8	Displays all shared memory blocks. The rows display the Instance Number, Instance Length, and Maximum Connection length for the corresponding Tx Holding Register Block.
Coils Block 1 - 8	Displays all shared memory blocks. The rows display the Instance Number, Instance Length, and Maximum Connection length for the corresponding Tx Coil Block.

7.6. Data Mapping Verification Page

The *Verify Data Mapping* page is used to check for the following configuration problems.

Configuration Issue	Description
Write conflicts to EtherNet/IP PLCs	This occurs when two or more Data Mapping configurations can write to the same PLC tag or File locations.
Write conflicts to Modus devices	This occurs when two or more Data Mapping configurations can write to the same Modbus device address.
Write to Shared Memory conflicts	This occurs when two or more Data Mapping configurations can write to the same Shared Memory address.
Invalid Shared Memory Addresses	This occurs when Data Mapping configuration can write to an invalid Shared Memory address.
Shared Memory block write protection violations	This occurs when a Data Mapping configuration can write to a Shared Memory address that is write-protected and there is no write access allowed.

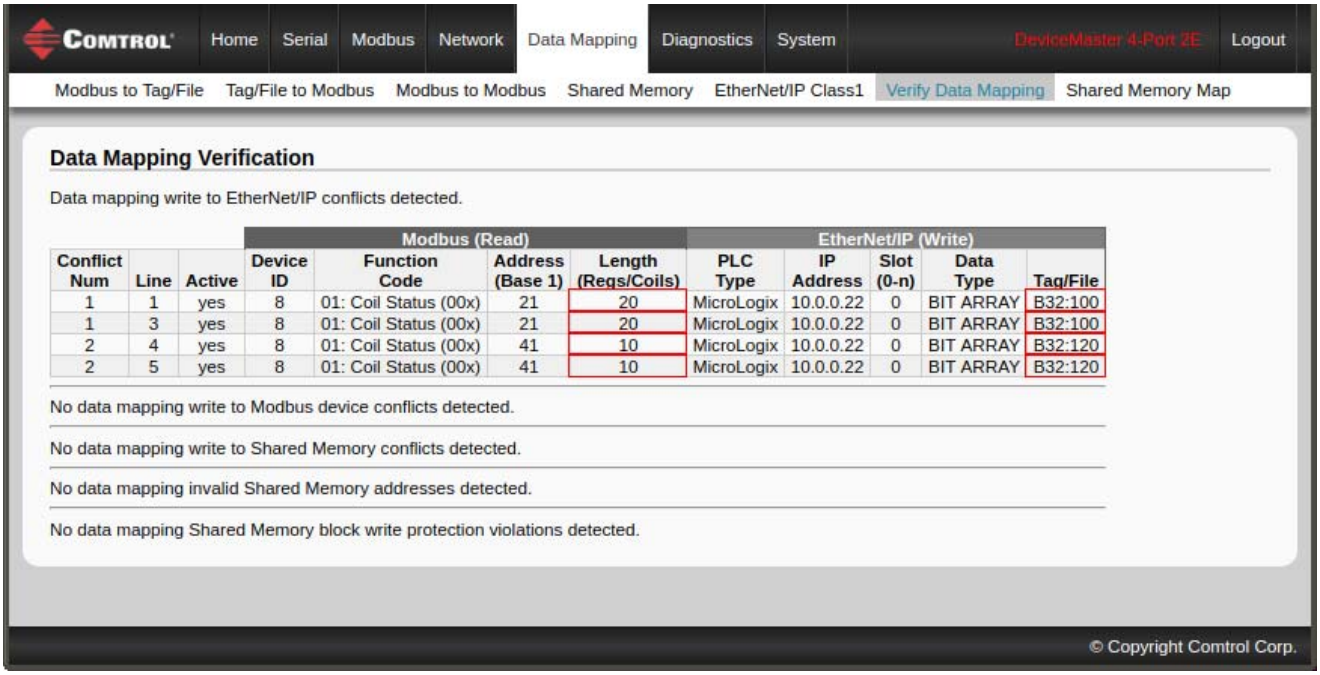
Access the *Data Mapping Verification* page by clicking **Data Mapping | Verify Data Mapping**. The following page indicates that the are no conflicts or violations detected.



The following page demonstrates what is displayed when there are multiple conflicts with the Data Mapping configurations:

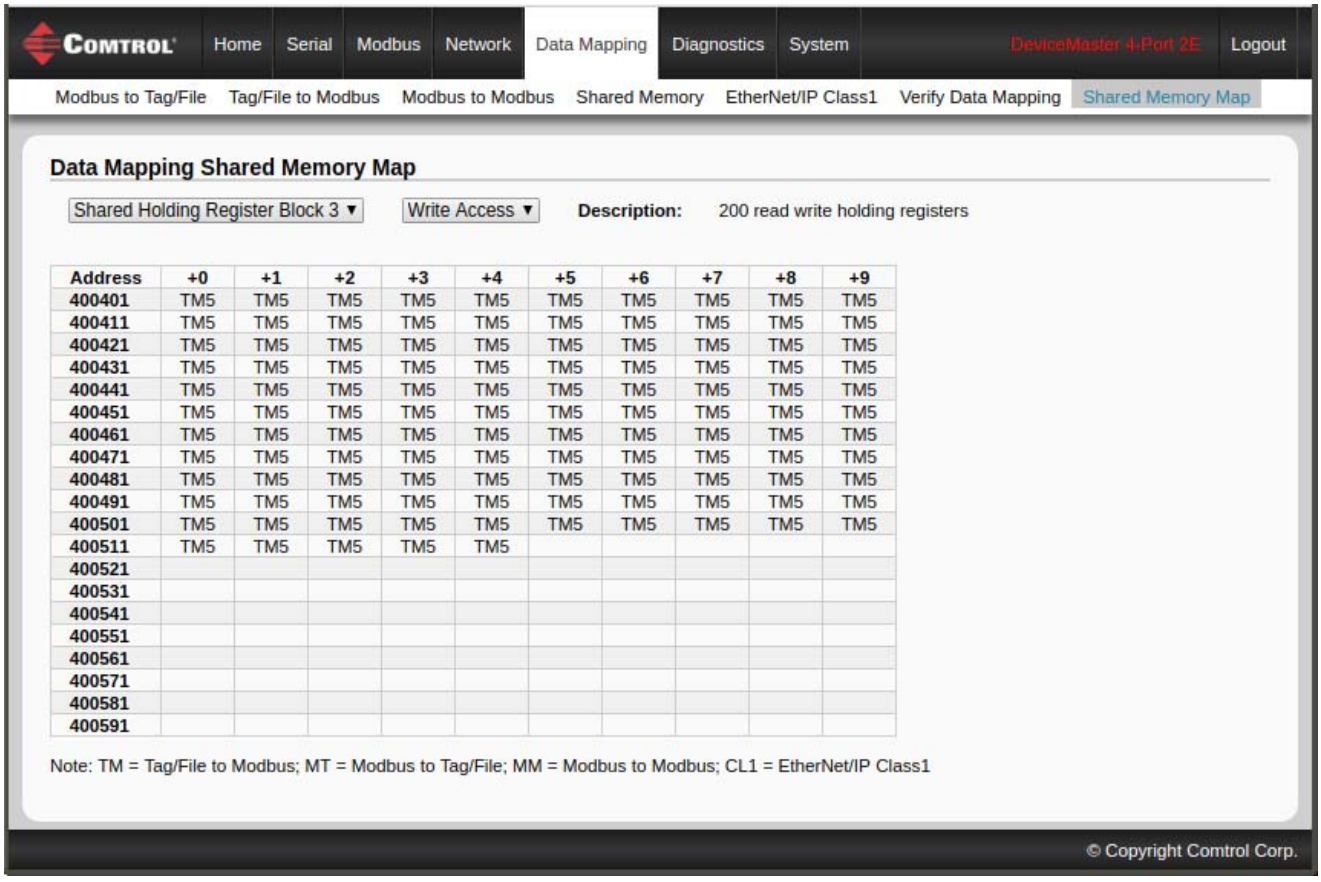
- A write to EtherNet/IP PLC conflict. Two configurations are writing to the same tag on a ControlLogix type PLC. Conflicting settings are highlighted in red.
- Two configuration are writing to the same address on a Modbus device.

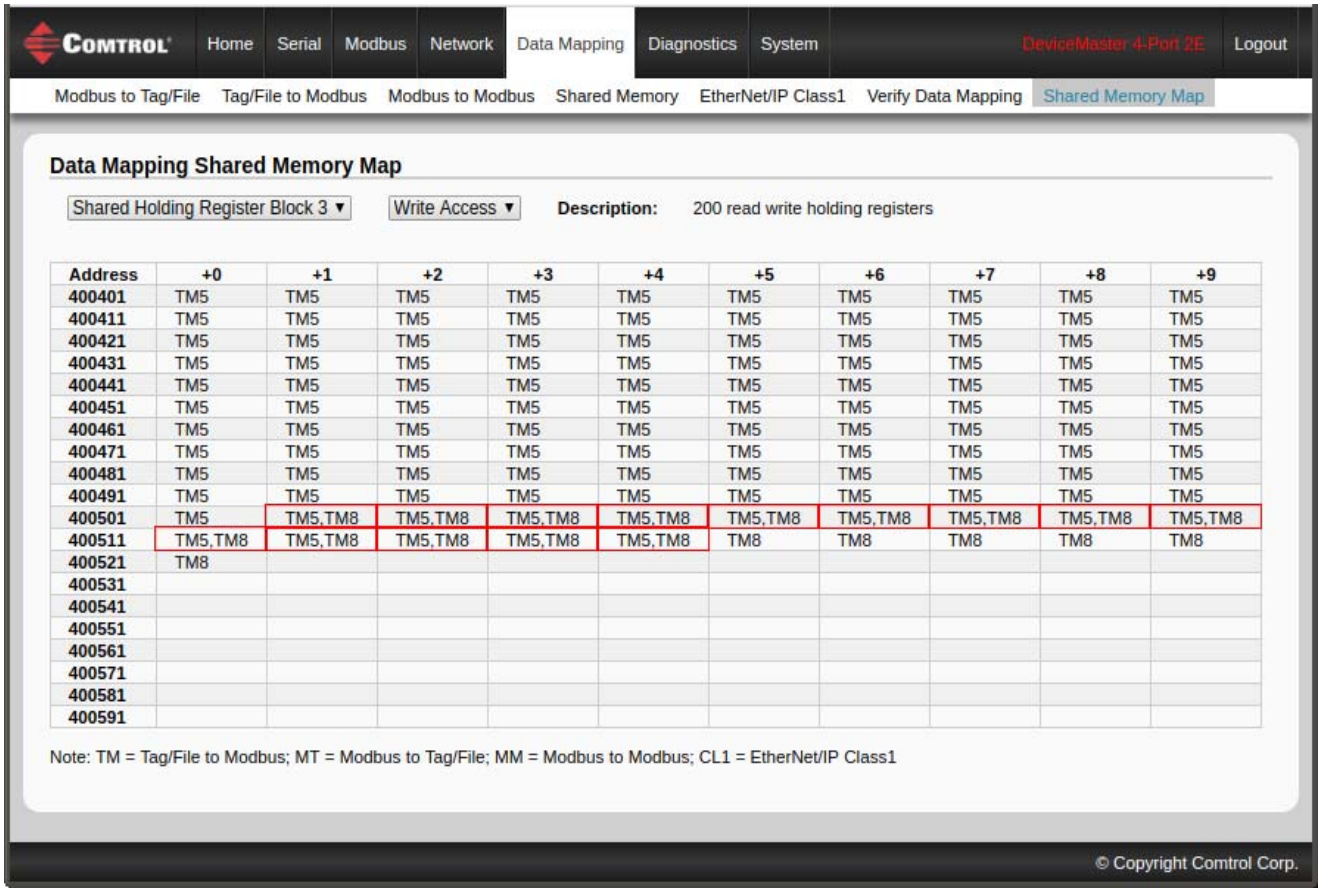
- There are two separate conflicts writing to the same Shared Memory addresses. Conflicting settings are highlighted in red.



7.7. Shared Memory Map Page

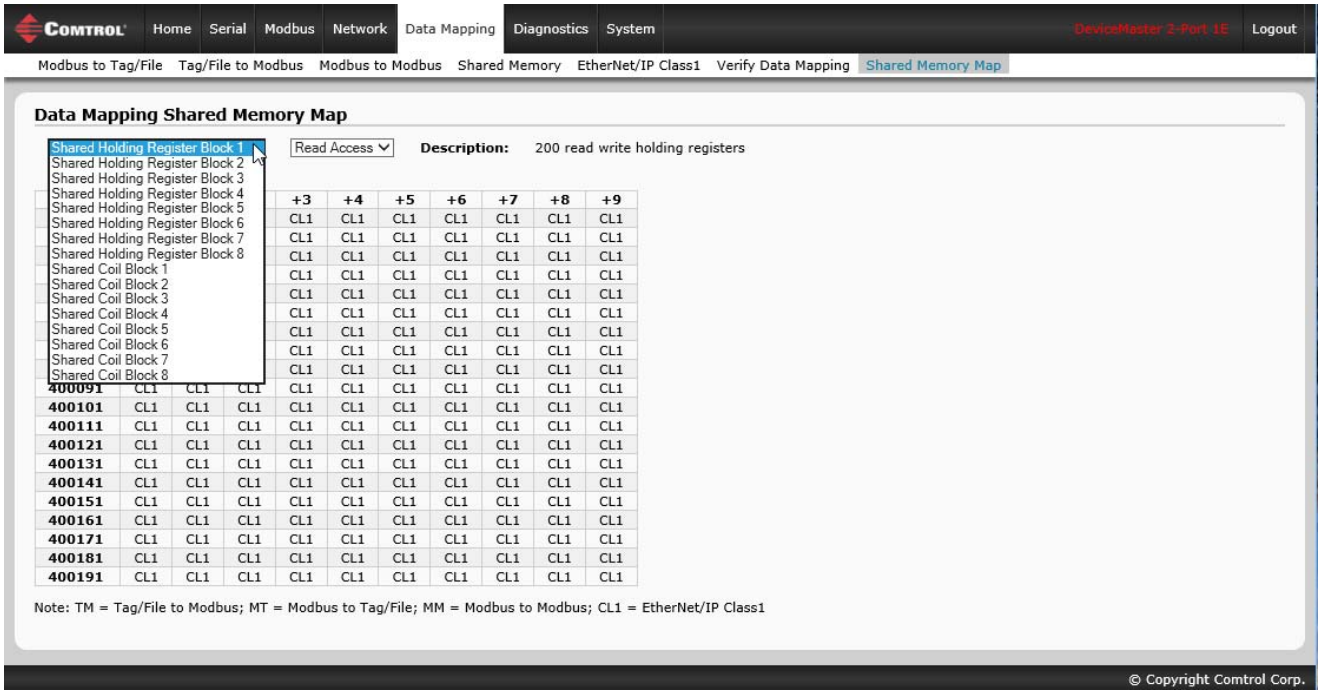
The *Shared Memory Map* page is provided to show the Built-in Configuration and Class1 access to each Shared Memory block. Write conflicts are shown highlighted in red as shown in the second image.




COMTROL
Home
Serial
Modbus
Network
Data Mapping
Diagnostics
System
DeviceMaster 4-Port 2E
Logout
Modbus to Tag/File
Tag/File to Modbus
Modbus to Modbus
Shared Memory
EtherNet/IP Class1
Verify Data Mapping
Shared Memory Map
Data Mapping Shared Memory Map
Shared Holding Register Block 3
Write Access
Description:
200 read write holding registers
Address +0 +1 +2 +3 +4 +5 +6 +7 +8 +9
400401 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5
400411 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5
400421 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5
400431 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5
400441 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5
400451 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5
400461 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5
400471 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5
400481 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5
400491 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5 TM5
400501 TM5 TM5,TM8 TM5,TM8 TM5,TM8 TM5,TM8 TM5,TM8 TM5,TM8 TM5,TM8 TM5,TM8 TM5,TM8
400511 TM5,TM8 TM5,TM8 TM5,TM8 TM5,TM8 TM5,TM8 TM8 TM8 TM8 TM8 TM8
400521 TM8
400531
400541
400551
400561
400571
400581
400591
Note: TM = Tag/File to Modbus; MT = Modbus to Tag/File; MM = Modbus to Modbus; CL1 = EtherNet/IP Class1
© Copyright Comtrol Corp.

Access the *Data Mapping Shared Memory Map* page by clicking **Data Mapping | Shared Memory Map**.

1. Select the shared holding register or coil block that you want to review.
2. Select **Write Access** or **Read Access**.

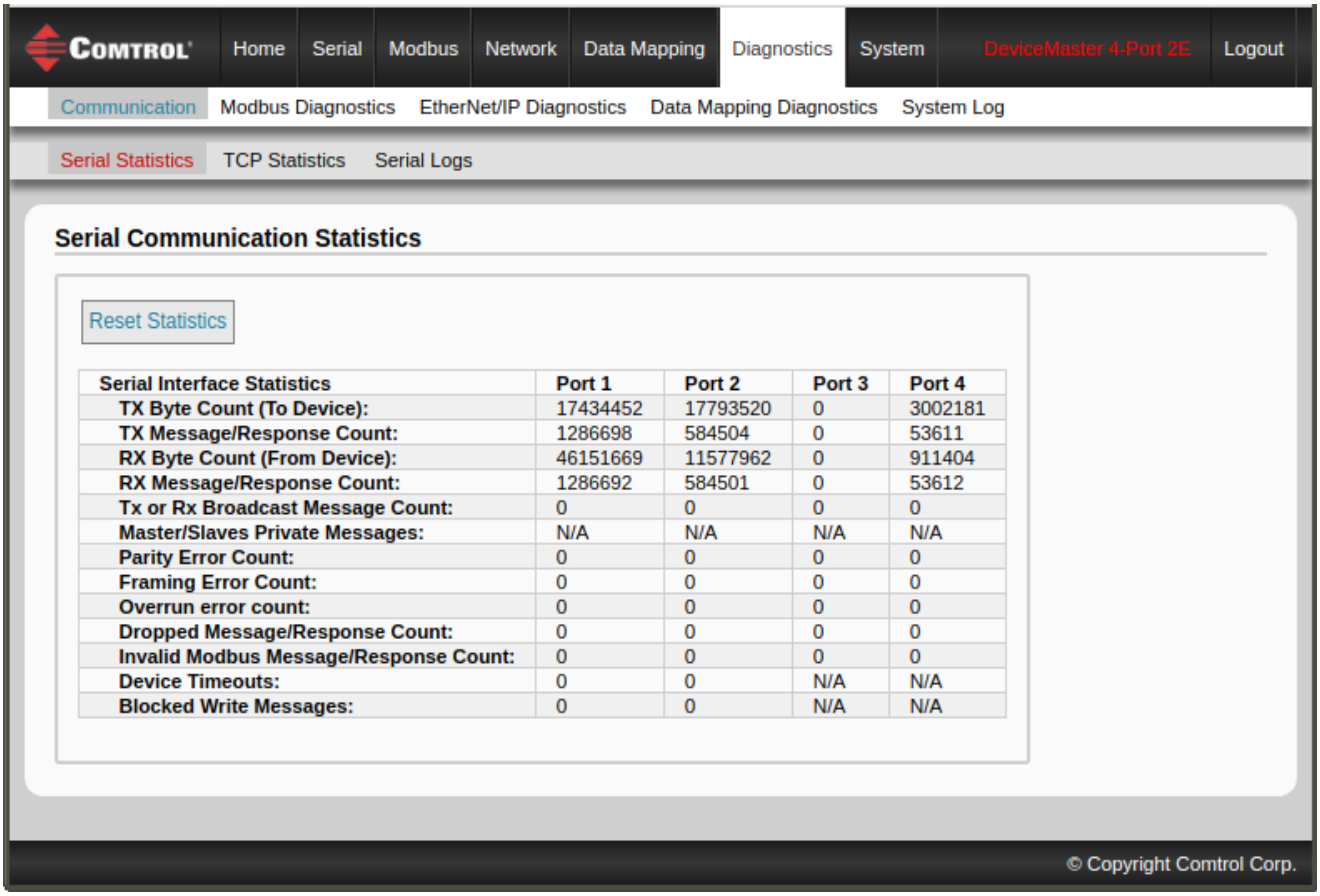


Chapter 8. Diagnostic Menus

You can access the following diagnostic and statistics pages from the *Diagnostics* menu:
This section discusses the following pages:

8.1. Serial Communication Statistics Page

The default *Diagnostics* menu page is the *Serial Communication Statistics* page.



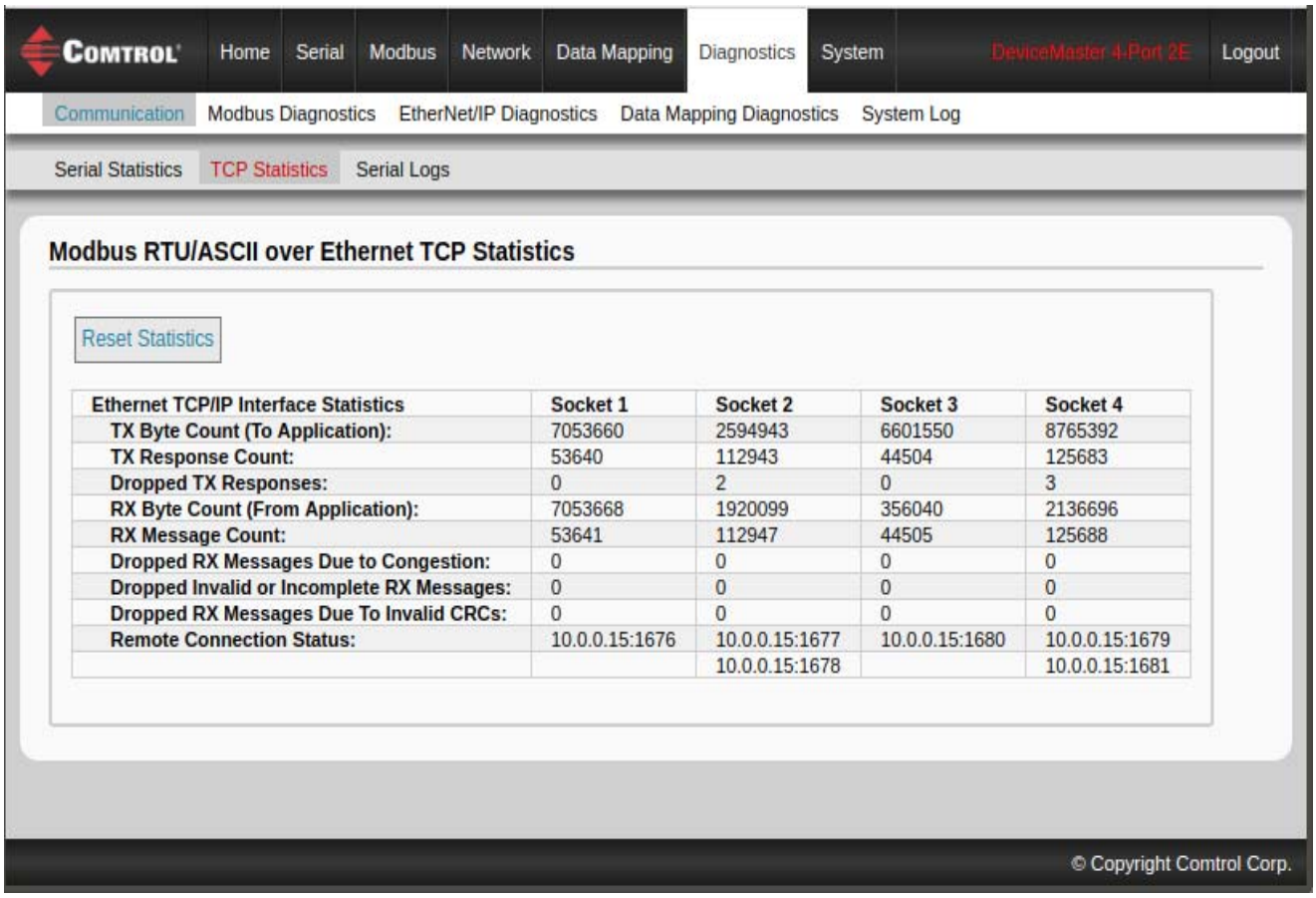
The following table provides detailed information about the *Serial Communications Statistics* page.

Serial Communication Statistics Page	
TX Byte Count (To Device)	Displays the number of bytes transmitted out the serial port.
TX Message/Response Count	Displays the number of messages or responses transmitted out of the serial port.
RX Byte Count (From Device)	Displays the number of bytes received on the serial port.
RX Message/Response Count	Displays the number of messages or responses received on the serial port.
Tx or Rx Broadcast Message Count	Displays the number of broadcast messages transmitted out the serial port.
Master/Slaves Private Messages	Displays the number of private messages detected, those between a master and private slave(s), on a serial port configured in Master/Slaves mode.
Parity Error Count	Displays the number of parity errors received on the serial port. Typically occurs due to an incorrect parity setting.

Serial Communication Statistics Page (Continued)	
Framing Error Count	Displays the number of framing errors received on the serial port. Typically occurs due to an incorrect baud rate or stop bit setting.
Overrun Error Count	Displays the number of overrun errors received on the serial port. This typically occurs to one of the following events: incorrect flow control, incorrect baud rate, incorrect data size, or incorrect stop bit setting.
Dropped Message/Response Count	Displays the number of messages or responses dropped to any of the following: • Incomplete message or response. • Did not receive valid start and/or end characters (Modbus/ASCII only).
Invalid Modbus Message/ Response Count	Displays the number of invalid messages or responses received to any of the following events: • Message received after the timeout period. This may require increasing the **Device Response Timeout**. • Incorrect device ID in response message. • Incorrect function code in response message.
Device Timeouts	Displays the number of device timeouts that occurred when there was no response for a Modbus message.
Blocked Write Messages	Displays the number of Modbus write messages that were not transmitted as a result of the **Disable Writes (Read Only)** option being set.

8.2. Modbus RTU/ASCII over Ethernet TCP Statistics Page

Click **Diagnostics | Communication | TCP Statistics** to access the *Modbus RTU/ASCII over Ethernet TCP Statistics* page.



The following table provides information about Ethernet TCP/IP interface statistics on the *Modbus RTU/ASCII over Ethernet TCP Statistics* page.

Modbus RTU/ASCII over Ethernet TCP Statistics Page	
TX Byte Count (To Application)	The number of bytes transmitted out of the TCP/IP connection(s).
TX Response Count	The number of responses transmitted out of the TCP/IP connection(s).
Dropped TX Responses	The number of responses that were intended to be transmitted out the TCP/IP connection(s) but could not be and were dropped. This typically occurs when one or more connections close unexpectedly.
RX Byte Count (From Application)	The number of bytes received on the TCP/IP connection(s).
RX Message Count	The number of messages received on the TCP/IP connection(s).
Dropped RX Messages Due to Congestion	The number of messages that were dropped to the gateway being overly congested. This typically occurs when the application(s) send messages faster than the slave device(s) can respond.

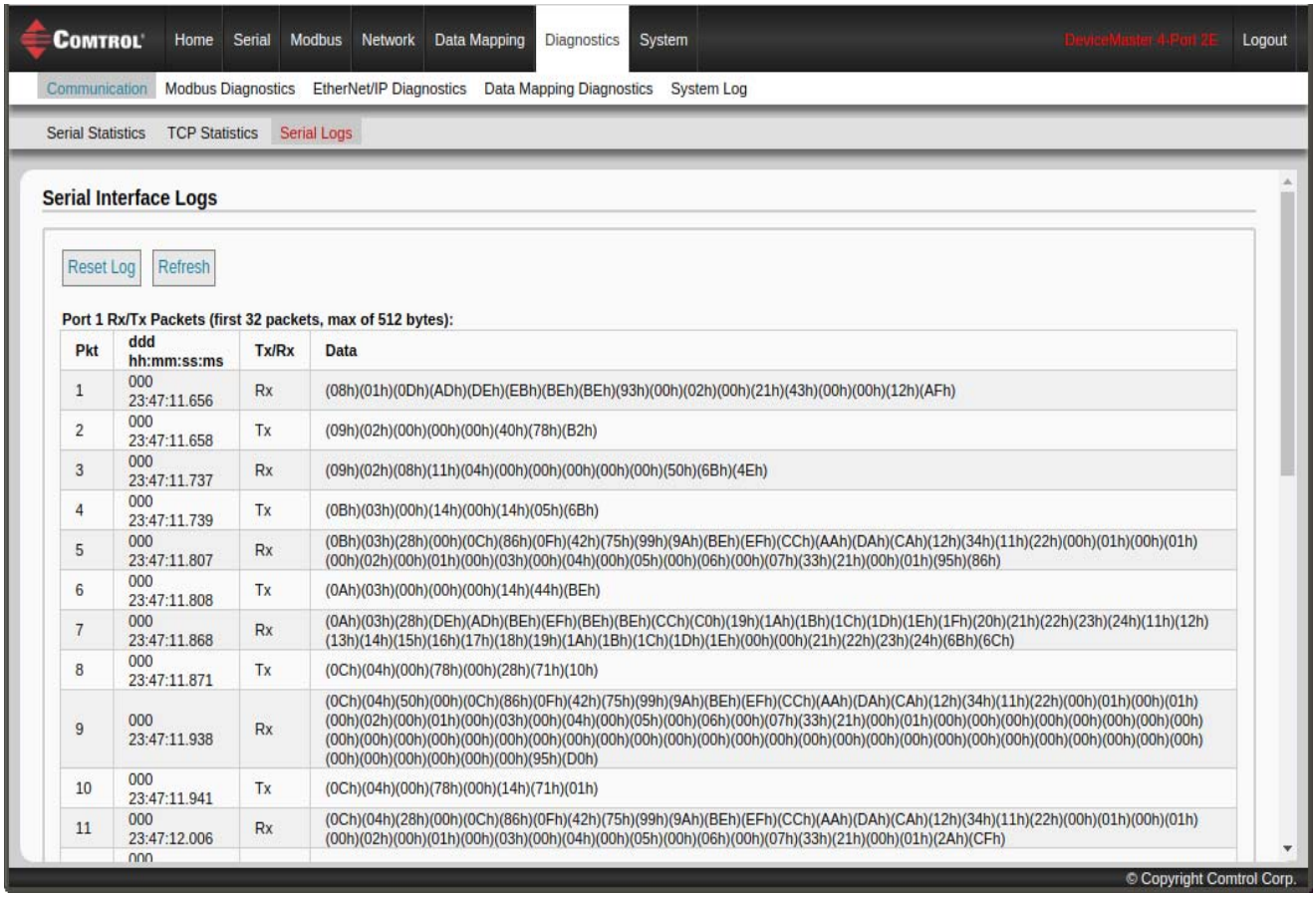
Modbus RTU/ASCII over Ethernet TCP Statistics Page (Continued)	
Dropped Invalid or Incomplete RX Messages	The number of messages from the application(s) that were dropped to: • Containing an invalid Modbus message format. • Containing an incomplete Modbus message.
Dropped RX Messages Due to Invalid CRCs	The number of messages from the application(s) that were dropped due to an invalid Modbus/RTU CRC or Modbus/ASCII LRC.
Remote Connection Status	Displays remote TCP/IP connections.

8.3. Serial Interface Logs

Access the *Serial Interface Logs* page by clicking **Diagnostics | Communication | Serial Logs**.

The *Serial Interface Logs* page provides a log of received and transmitted serial port messages. Up to 512 bytes per message and up to 32 messages are logged. It is intended to help with debugging serial connectivity problems, determining the proper start and end of transmission bytes, and diagnosing device problems.

The format is as follows:




```
Pkt(n): ddd:hh:mm:ss.ms Tx/Rx:<Data>
```

Where:

ddd – days since last system restart

hh – hours since last system restart

ms – minutes since last system restart

ss – seconds since last system restart

mm – milliseconds since last system restart

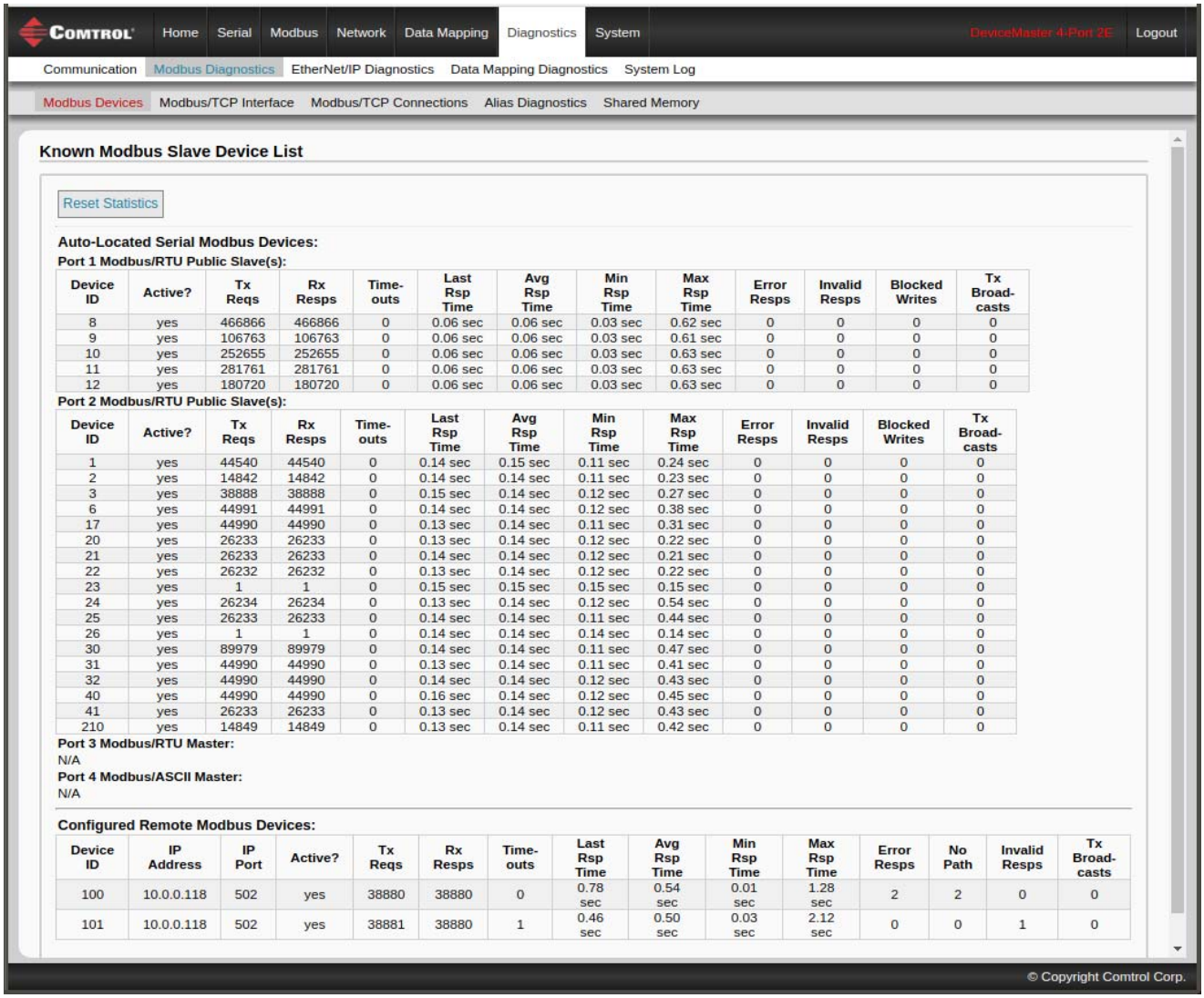
<Data> - Data packet received.

- For Modbus slave data, all data bytes shown in hex (xxh) format.
- For Raw/ASCII and Modbus/ASCII data
- ASCII characters displayed as characters
- Non-ASCII displayed in hex (xxh) format

8.4. Known Modbus Slave Device List

The *Known Modbus Slave Device List* page provides device specific status and statistics for each device connected locally to one or more of the serial ports or remotely through a remote Modbus/TCP device configuration.

Access the *Known Modbus Slave Device List* page by clicking **Diagnostics | Modbus Diagnostics | Modbus Devices**.



The following table provides detailed information about this page.

Known Modbus Slave Device List Page - Public Devices	
Device ID	Displays the device ID associated with this device. ***Note:*** *If* **Device ID Offset Mode** *is enabled, the actual device ID transmitted out the serial port displays as (SP=xxx).*

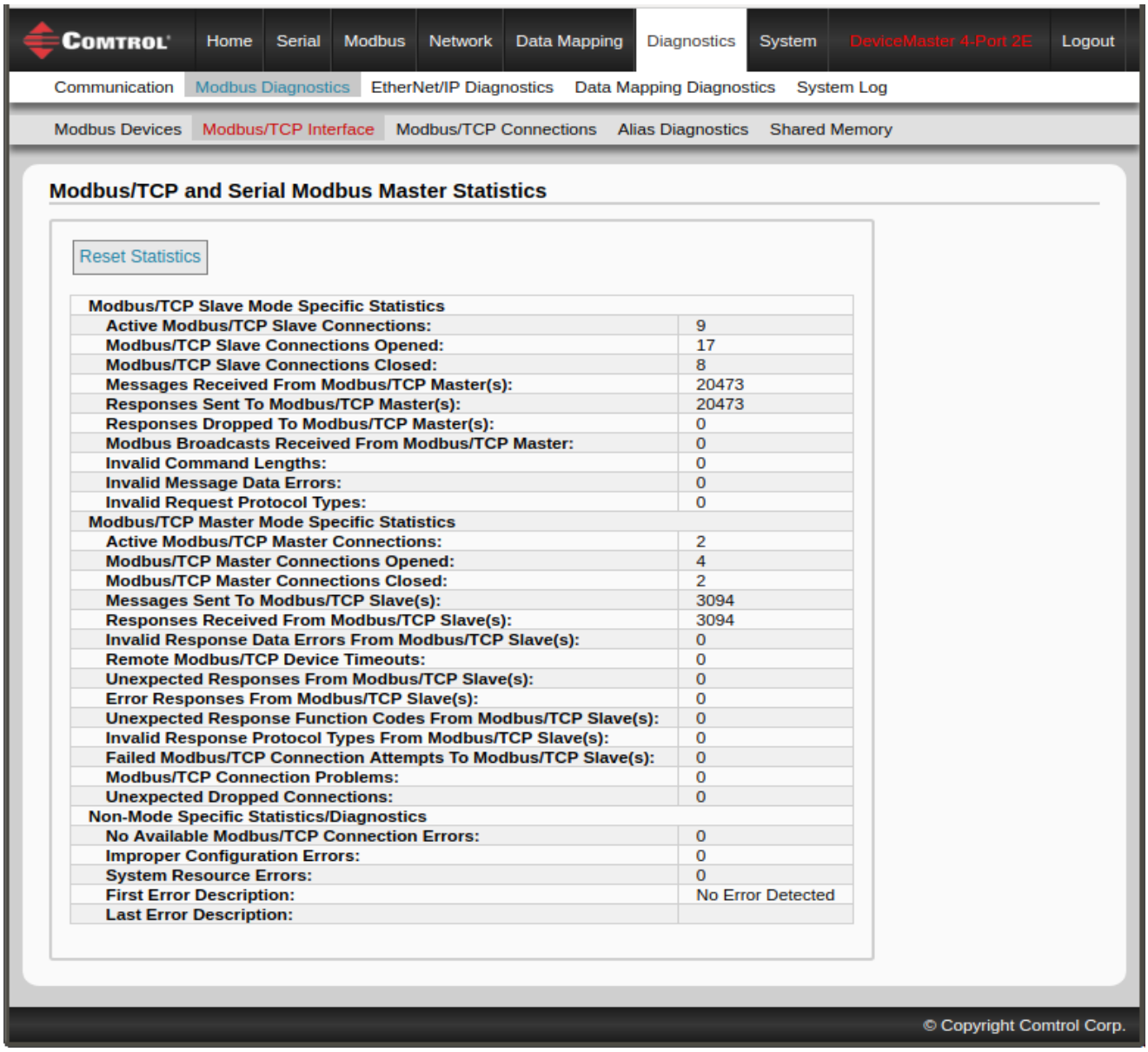
Known Modbus Slave Device List Page - Public Devices (Continued)	
Active?	Displays the status of device: • **Yes** means that the last request received a valid response and did not time out. • **No** means that the last request timed out or the device has not yet received a message.
IP Address	Displays the IP address associated with the local device under **Configured Remote Modbus Devices**.
IP Port	Displays the TCP/IP port associated with the remote device under **Configured Remote Modbus Devices**.
Tx Req	Displays the number of Modbus messages transmitted to this device.
Rx Rsp	Displays the number of Modbus responses received from this device.
Timeouts	Displays the number of response timeouts associated with this device.
Last Rsp Time	Displays the last response time from the Modbus device.
Avg Rsp Time	Displays the average response time from the Modbus device.
Min Rsp Time	Displays the minimum response time from the Modbus device.
Max Rsp Time	Displays the maximum response time from the Modbus device.
Error Rsp	Displays the number of responses with Modbus error indications.
No Path	This displays under **Configured Remote Modbus Devices** the number of times the network path could not be connected. This could be a result of: • Out of Modbus/TCP connections. • Modbus/TCP device not responding. • Incorrect IP address.
Invalid Responses	Displays the number of invalid messages or responses received to any of the following: • Message received after the timeout period. This may require increasing the **Device Response Timeout**. • Incorrect device ID in response message. • Incorrect function code in response message.
Blocked Writes	Displays the number of Modbus write messages that were not transmitted for this device. This only occurs when the **Disable Writes (Read Only)** serial port option is selected.
Tx Broadcasts	Displays the number of Modbus broadcast messages transmitted to this device.

Known Modbus Slave Device List Page - Private Devices	
Device ID	Displays the device ID associated with this device. ***Note:*** *If* **Device ID Offset Mode** *is enabled, the actual device ID transmitted out the serial port displays as (SP=xxx).*

Known Modbus Slave Device List Page - Private Devices (Continued)	
Requests	Displays the number of private requests addressed to this Modbus device.
Responses	Displays the number of private responses from this Modbus device.
Req Or Resp?	Displays the number of private requests/responses addressed to/from this Modbus device that could not be identified specifically as either a request or a response.
No Responses	Displays the number of requests that this Modbus device did not respond to.
Last Rsp Time	Displays the last response time from the Modbus device.
Avg Rsp Time	Displays the average response time from the Modbus device.
Min Rsp Time	Displays the minimum response time from the Modbus device.
Max Rsp Time	Displays the maximum response time from the Modbus device.
Error Rsp	Displays the number of responses with Modbus error indications.

8.5. Modbus/TCP and Serial Modbus Master Statistics Page

Click **Diagnostics | Modbus Diagnostics | Modbus/TCP Interface** to access the *Modbus/TCP and Serial Modbus Master Statistics* page, which is explained in the following table.



Note: *The Modbus/TCP interface uses the standard socket port of 502.*

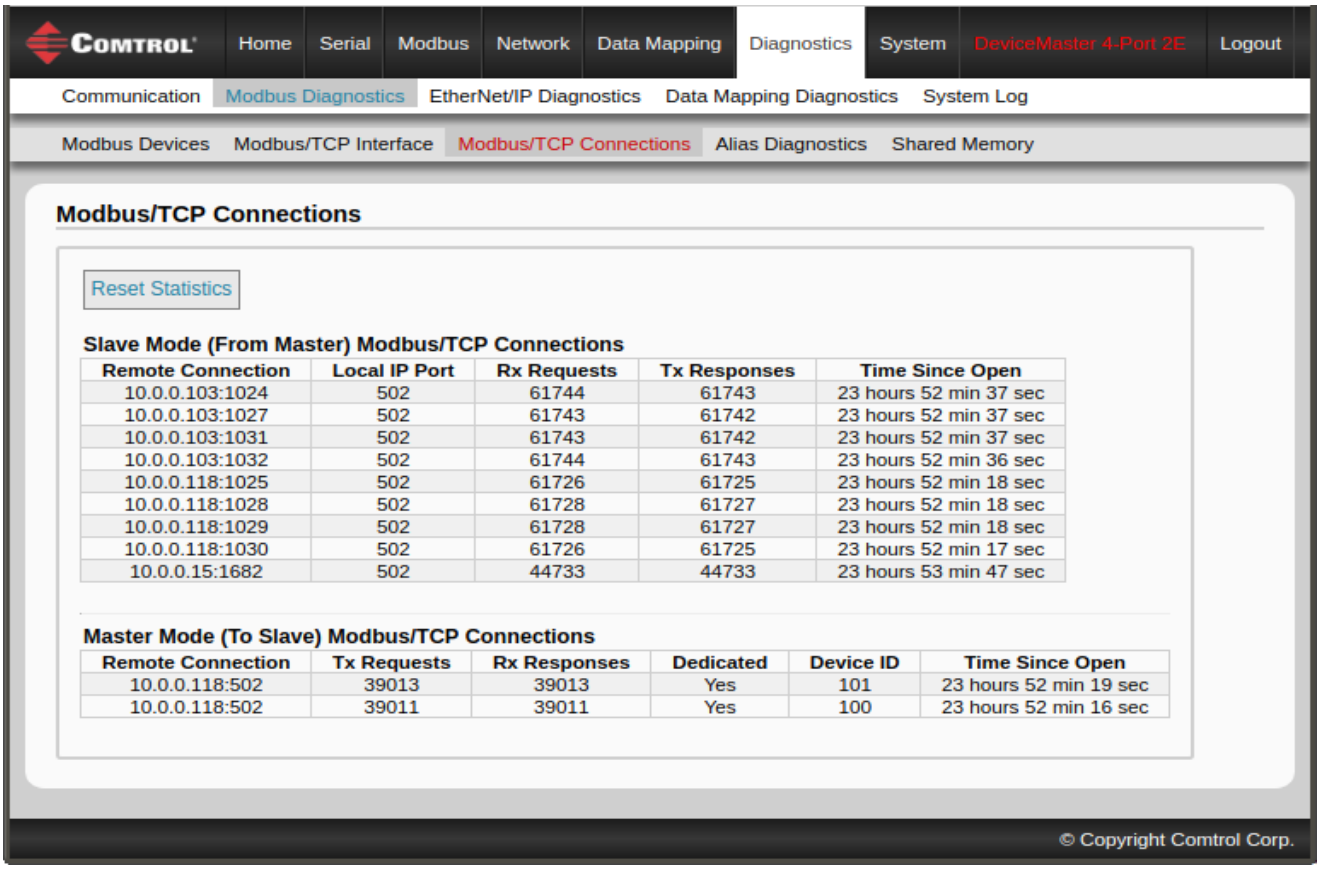
Modbus/TCP and Serial Modbus Master Statistics Page	
Modbus/TCP Slave Mode Specific Statistics	
Active Modbus/TCP Slave Connections	Displays the current number of active Modbus/TCP slave connections. These connections were initiated by a controller to the DeviceMaster.
Modbus/TCP Slave Connections Opened	Displays the total number of Modbus/TCP slave connections that have been opened.
Modbus/TCP Slave Connections Closed	Displays the total number of Modbus/TCP slave connections that have been closed.
Messages Received From Modbus/TCP Master(s)	Displays the total number of Modbus/TCP messages received from Modbus/TCP master(s).
Responses Sent to Modbus/TCP Master(s)	Displays the total number of Modbus/TCP responses sent to Modbus/TCP master(s).
Responses Dropped to Modbus/TCP Master	
Modbus Broadcasts Received From Modbus/TCP Master	Displays the number of Modbus broadcast messages received from Modbus/TCP Masters.
Invalid Command Lengths	Displays the number of messages received with invalid command lengths.
Invalid Message Data Errors	Displays the number of messages received with invalid message data errors. These errors occur when the DeviceMaster EIP-MOD receives a message that cannot be processed due to improper message data.
Invalid Request Protocol Types	Displays the number of messages received with invalid protocol errors. This occurs when a message is received with a protocol other than the Modbus/TCP protocol value of zero.
Modbus/TCP Master Mode Specific Statistics	
Active Modbus/TCP Master Connections	Displays the current number of active Modbus/TCP master connections. These connections were initiated by the DeviceMaster EIP-MOD to a Modbus/TCP slave.
Modbus/TCP Master Connections Opened	Displays the total number of Modbus/TCP master connections that have been opened.
Modbus/TCP Master Connections Closed	Displays the total number of Modbus/TCP master connections that have been closed.
Messages Sent To Modbus/TCP Slave(s)	Displays the total number Modbus messages sent to remote Modbus/TCP slaves.
Responses Received From Modbus/TCP Slave(s)	Displays the total number of Modbus responses received from the Modbus/TCP Slave(s).
Invalid Response Data Errors From Modbus/TCP Slave(s)	Displays the number of response data errors to polling requests returned from the Modbus/TCP Slave(s). Possible causes include: • Incorrect transaction ID incorrect. • Message command length to large. • Incorrect device Id in response.

Modbus/TCP and Serial Modbus Master Statistics Page (Continued)	
Remote Modbus/TCP Device Timeouts	Displays the number of messages to remote devices that were determined to have timed out by this gateway.
Unexpected Responses From Modbus/TCP Slave(s)	Displays the number of responses received when no response was expected.
Error Responses from Modbus/TCP Slave(s)	Displays the number of responses received from Modbus/TCP slaves with errors indicated. This may be caused by such things as: • Device timeouts detected by slave Modbus/TCP device, such a gateway. • Invalid device address. • Invalid device ID. • Invalid message data.
Unexpected Response Function Codes From Modbus/TCP Slave(s)	Displays the number of unexpected response function codes from Modbus/TCP slaves. This occurs when a response was received with a different function code than what was sent.
Invalid Response Protocol Types From Modbus/TCP Slave(s)	Displays the number of responses with invalid protocol errors. This occurs when a response is returned with a protocol other than the Modbus/TCP protocol value of zero.
Failed Modbus/TCP Connection Attempts to Modbus/TCP Slave(s)	Displays the number of failed Modbus/TCP connection attempts to the specified PLC IP address.
Modbus/TCP Connection Problems	Displays the number of Modbus/TCP connection attempt problems. This occurs when the device responds and the connection is made, but there are problems setting up the connection options. The possible option problems include: • Setting the TCP connection to **TCP_NODELAY**. • Setting the socket connection to **SO_OOBINLINE**. • Setting the socket connection to **SO_KEEPALIVE**.
Unexpected Dropped Connections	Displays the number of Modbus/TCP connections that were unexpectedly dropped.

Modbus/TCP and Serial Modbus Master Statistics Page (Continued)	
Non-Mode Specific Statistics/Diagnostics	
No Available Modbus/TCP Connection Errors	Displays the number of connections aborted when there are no available Modbus/TCP connections. This error occurs when the maximum number of Modbus/TCP connections has been reached and the DeviceMaster EIP-MOD is attempting to form another Modbus/TCP connection.
Improper Configuration Errors	Displays the number of errors that were caused by improper configuration errors.
System Resource Errors	Displays the numbers of system resource errors. These errors are typically caused by congestion and/or non-responding devices.
First Error Description	Displays the first error detected.
Last Error Description	Displays the last or most recent error detected.

8.6. Modbus/TCP Connections Page

Access the *Modbus / TCP Connections* page by clicking **Diagnostics | Modbus Diagnostics | Modbus/TCP Connections**.

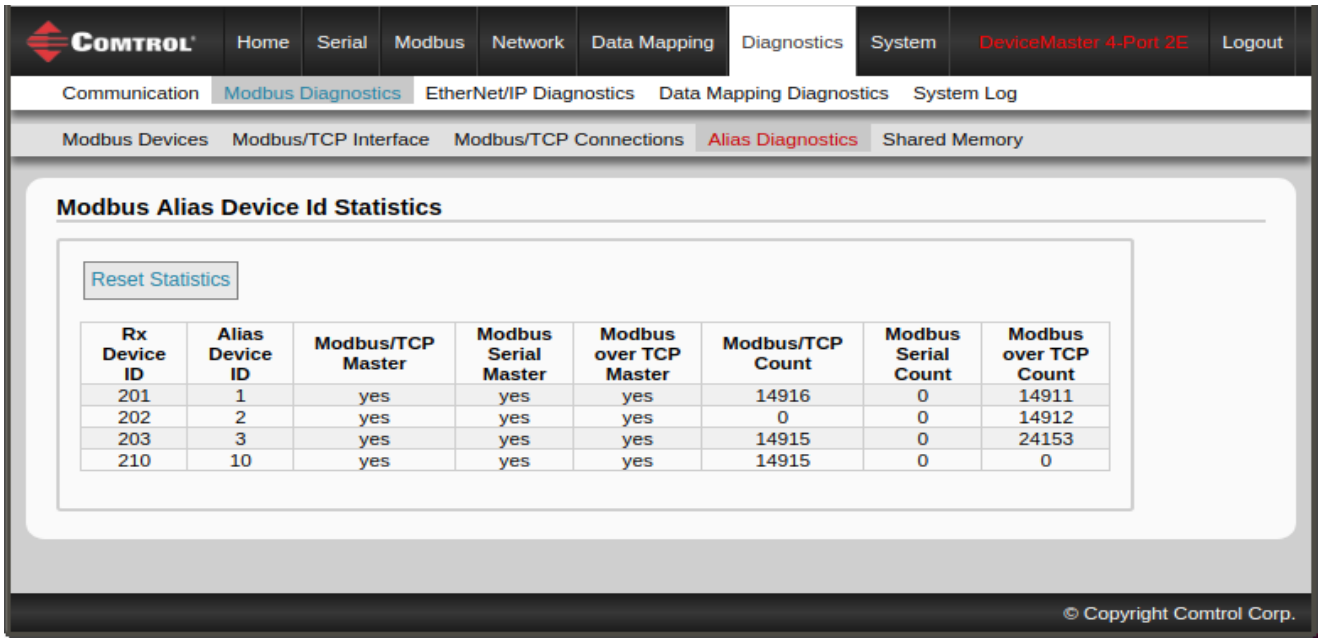


Modbus/TCP Connections Page	
Slave Mode (From Master) Modbus/TCP Connections ***Note:*** *Only displayed if active connections.*	
Remote Connection	The Modbus/TCP master connection in *IP Address:Port Number* format (xxx.xxx.xxx.xxx:pppp).
Local IP Port	The local TCP/IP port on the DeviceMaster EIP-MOD. The standard Modbus/TCP port of 502 is always enabled. Optionally, up to seven additional Modbus/TCP ports may also be enabled.
Rx Requests	Displays the number of Modbus requests that have been received since the connection was opened.
Tx Responses	Displays the number of Modbus responses that have been transmitted since the connection was opened.
Time Since Open	The time that has elapsed since the connection was opened.

Modbus/TCP Connections Page (Continued)	
Master Mode (To Slave) Modbus/TCP Connections ***Note:*** *Only displayed if active connections.*	
Remote Connection	The Modbus/TCP master connection in IP *Address:Port* Number format (xxx.xxx.xxx.xxx:pppp).
Tx Requests	Displays the number of Modbus requests that have been transmitted since the connection was opened.
Rx Responses	Displays the number of Modbus responses that have been received since the connection was opened.
Dedicated	Indicates if the connection is dedicated for a specified Modbus device ID.
Device ID	If the connection is dedicated, the corresponding device ID.
Time Since Open	The time that has elapsed since the connection was opened.

8.7. Modbus Alias Device ID Statistics Page

Access the *Modbus Alias Device ID Statistics* page by clicking **Diagnostics | Modbus Diagnostics | Alias Diagnostics**.



Modbus Alias Device ID Statistics Page	
Rx Device ID	Displays the device ID (also often called the unit ID) of the received message from a master.
Alias Device ID	The alias device ID to convert the received device ID to.
Modbus/TCP Master	If selected, this applies the alias device ID configuration to messages received from Modbus/TCP masters.
Modbus Serial Master	If selected, this applies the alias device ID configuration to messages received from serial Modbus masters.
Modbus over TCP Master	If selected, this applies the alias device ID configuration to messages received from Modbus RTU/ASCII over Ethernet TCP/IP masters.
Modbus/TCP Count	Number of Aliased Modbus messages received from Modbus/TCP masters.
Modbus Serial Count	Number of Aliased Modbus messages received from Modbus Serial masters.
Modbus over TCP Count	Number of Aliased Modbus messages received from Modbus over TCP masters.

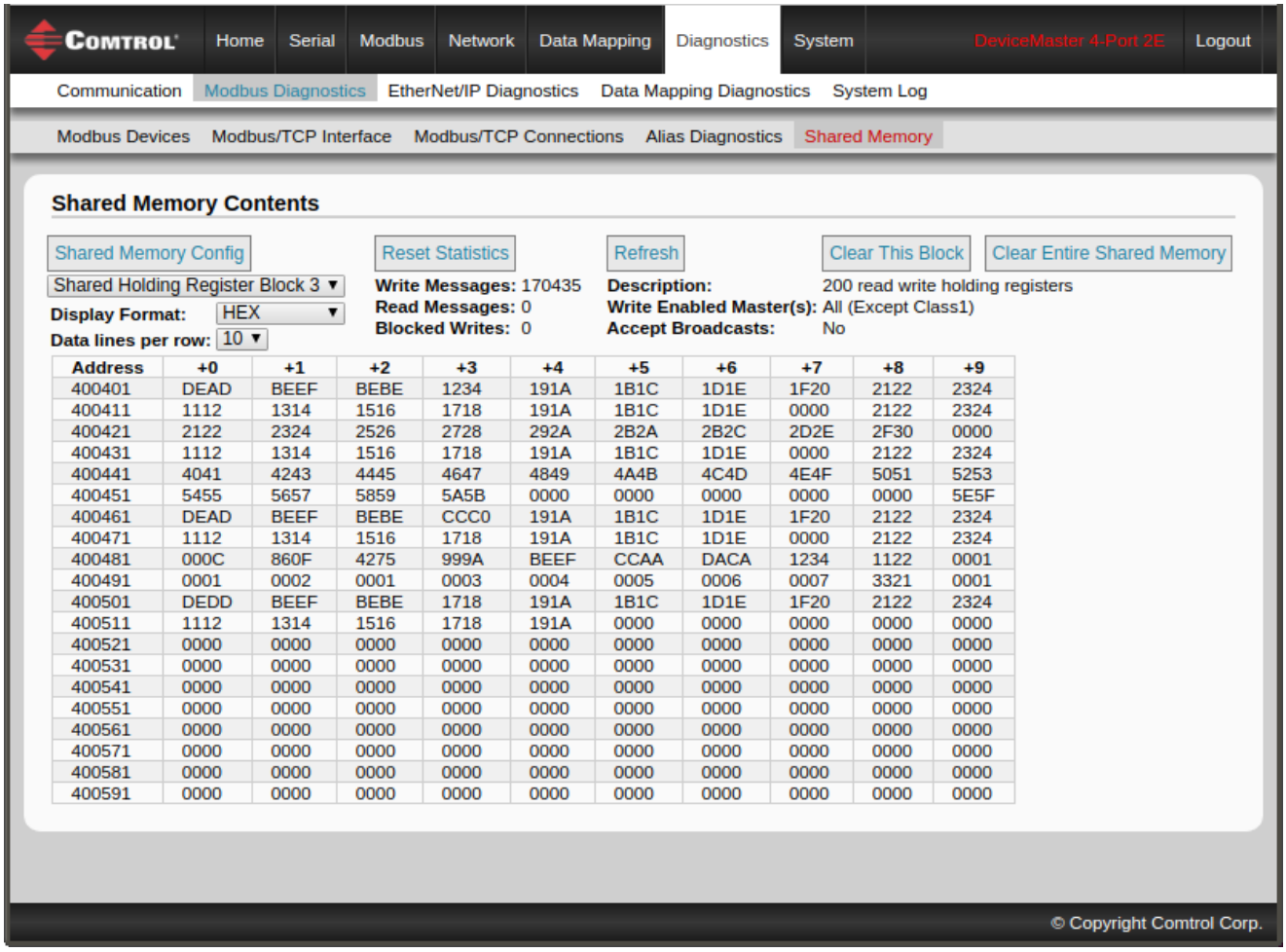
8.8. Shared Memory Contents

This page displays the contents of a Shared Holding Register and Shared Coil blocks.

Access this page by clicking **Diagnostics | Modbus Diagnostics | Shared Memory** and select the Holding Register Block or Shared Coil Block that you want to display in the drop list.

8.8.1. Shared Holding Block Register Blocks

The following image shows the Shared Holding Register Block 1.

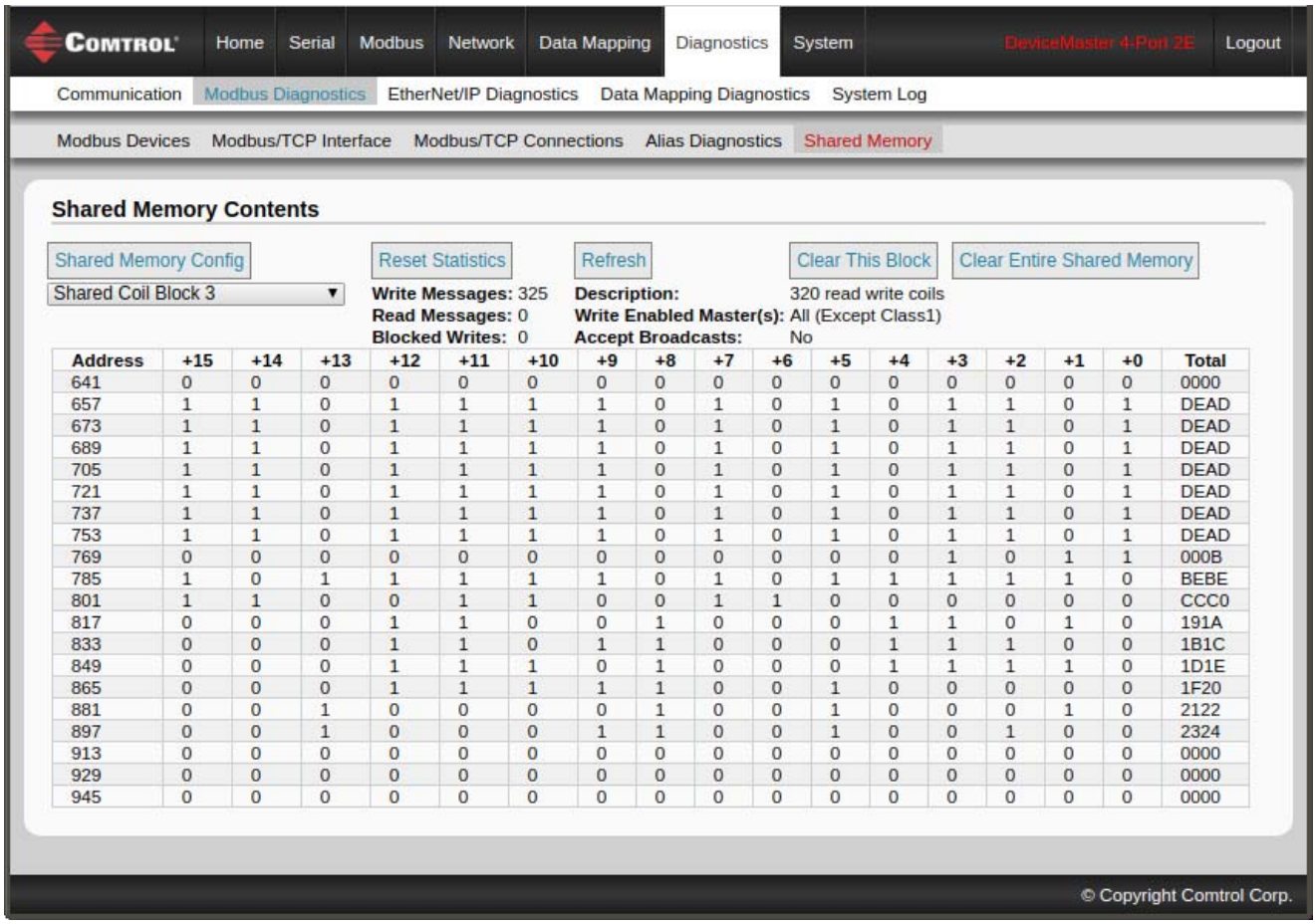


Shared Memory Contents Page	
Shared Memory Config button	Click this button to jump to the *Shared Memory Configuration* page.
Reset Statistics button	Click this button to reset the following: • Write messages • Read messages • Blocked writes
Refresh button	Click this button to refresh this page.
Clear This Block button	Click this button to clear the diagnostics and data contents of this Holding Register block.
Clear Entire Shared Memory button	Click this button to clear the diagnostics and data contents of all Holding Register and Coil blocks.
Shared Holding Register Block drop list	Use this drop list to select the Shared Holding Register Block or Shared Coil Block that you want to display.
Rx Rsp	Displays the number of Modbus responses received from this device.
Display Format	Selectable data format to display the contents of the Holding Register block data. • Hex - 16 bit word (default) • word-16 - unsigned 16 bit decimal • word-32 - unsigned 32 bit decimal • string - ASCII character string
Data line per row	Selectable data entries per row to display the contents of the Holding Register block data. • 10-per-row – ten entries display per row • 20-per-row – twenty entries displayed per row (default)
Write Messages	Displays the number of successful write messages to this Shared Memory block.
Read Messages	Displays the number of read messages addressed to this Shared Memory block.
Blocked Writes	Displays the number of write messages that were blocked, or prevented, from writing to this Shared Memory block. ***Note:*** *Blocked writes are treated as write violations and are added to the Write Violation Log.*
Description	This is the description from the *Shared Memory Configuration* page.
Write Enabled Master(s)	Displays how this is configured in the *Shared Memory Configuration* page.
Accept Broadcasts	Displays if the **Accept Broadcast Messages** option is enabled in the *Shared Memory Configuration* page.

8.8.2. Shared Coil Blocks

This page displays the contents of the selected Shared Coil block.

Access this page from the *Shared Memory Configuration* page and click **Display** next to the Coil block that you want to display.

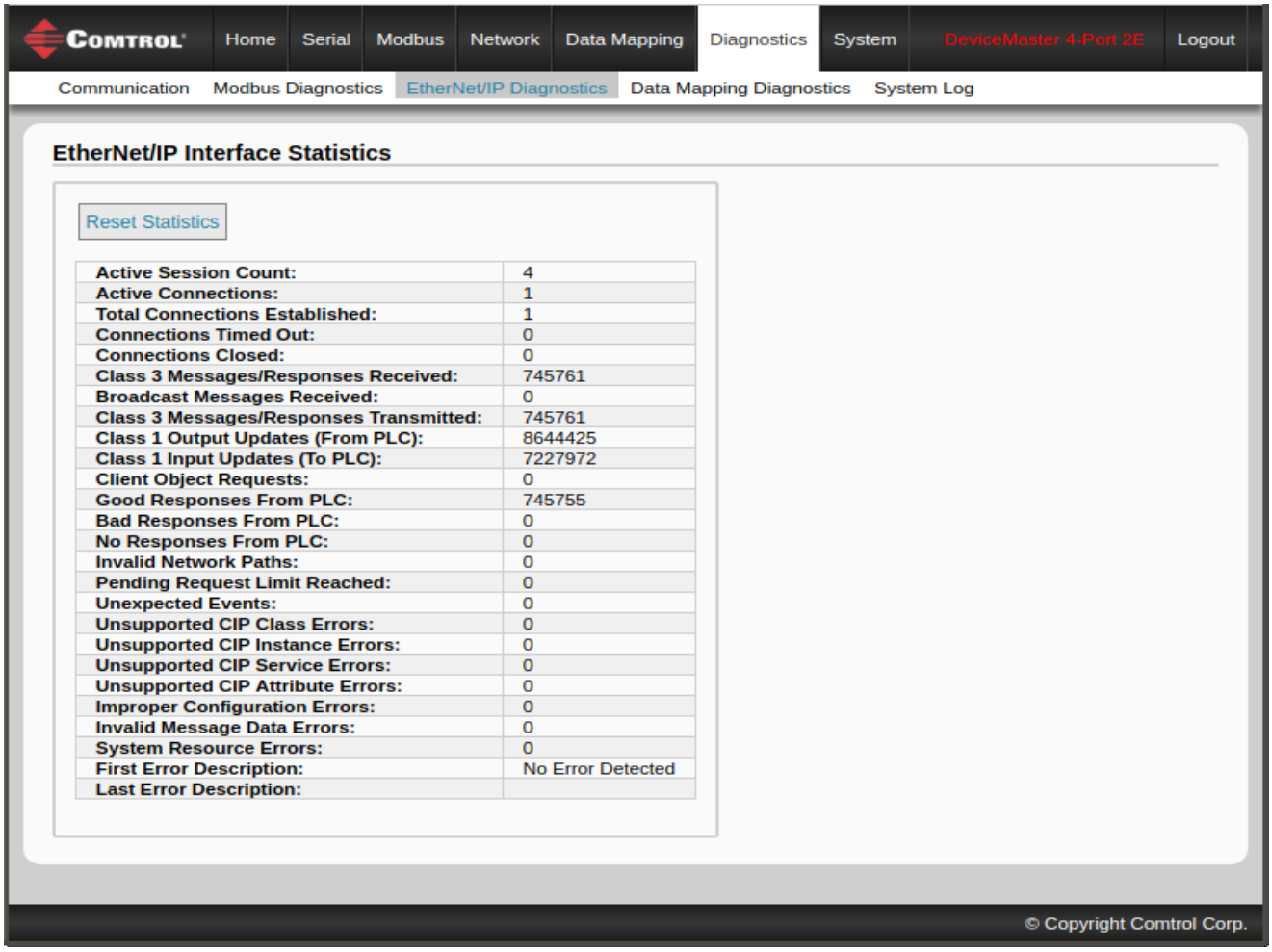


Shared Memory Contents Page - Shared Coil Blocks	
Buttons	
Shared Memory Config	Selecting this button returns you to the *Shared Memory Configuration* page.
Reset Statistics	Click this button to clear the diagnostics for this Holding Register block.
Refresh	Click this button to refresh the data on the page.
Clear This Block	Selecting this button clears the diagnostics and data contents of the Holding Register block in the drop list.
Clear Entire Shared Memory	Click button to clear the diagnostics and data contents of all Holding Register and Coil blocks.

Shared Memory Contents Page - Shared Coil Blocks (Continued)	
Drop Lists and Fields	
Shared Coil Block Drop List	Use this drop list to display data for a specific Shared Holding Register Block or Shared Coil Block.
Write Messages	Displays the number of successful write messages to this Shared Memory block.
Read Messages	Displays the number of read messages addressed to this Shared Memory block.
Blocked Writes	Displays the number of write messages that were blocked, or prevented, from writing to this Shared Memory block. ***Note:*** *Blocked writes are treated as write violations and are added to the Write Violation Log.*
Description	This is the description from the *Shared Memory Configuration* page.
Write Enabled Master(s)	Displays how this is configured in the *Shared Memory Configuration* page.
Accept Broadcasts	Displays if the **Accept Broadcast Messages** option is enabled in the *Shared Memory Configuration* page.

8.9. Ethernet/IP Interface Statistics Page

Access the *EtherNet/IP Interface Statistics* page by clicking **Diagnostics | EtherNet/IP Diagnostics**.



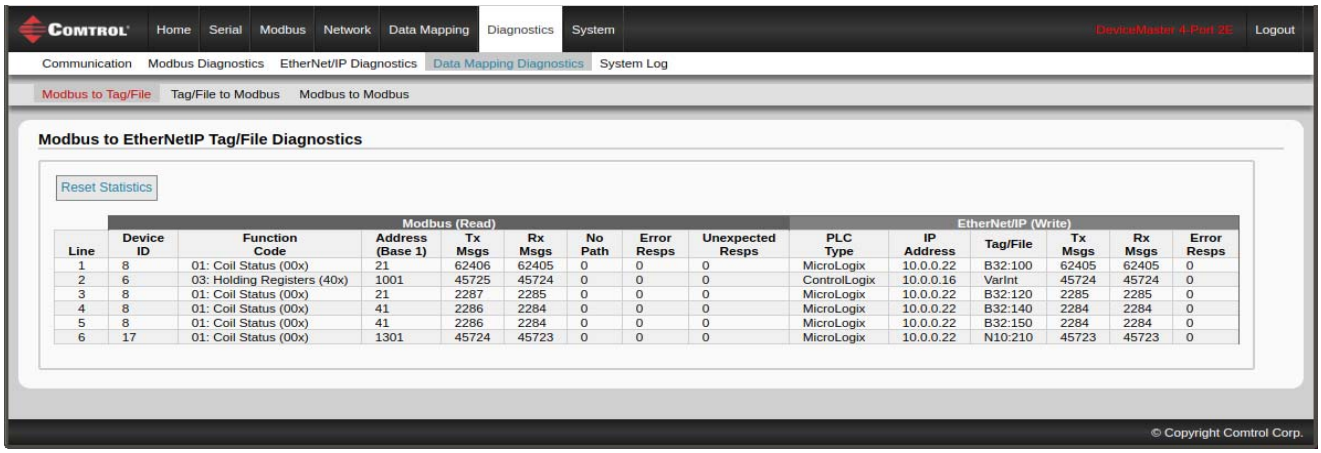
EtherNet/IP Interface Statistics Page	
Active Session Count	The number of active Ethernet/IP sessions. A session can: • Support both Class 1 I/O and Class 3 Messages • Can be initiated by either the PLC or the GW EIP/ASCII • Can be terminated by either the PLC or the GW EIP/ASCII
Active Connections	The current number of active connections (both Class 1 and 3).
Total Connections Established	The total number of connections that have been established.
Connection Timed Out	The number of connections that have closed due to timing out.
Connections Closed	The number connections that have closed due to a standard processes.
Class3 Messages/ Responses Received	Displays the number of Class3 messages and responses received from the PLC(s).

EtherNet/IP Interface Statistics Page (Continued)	
Broadcasts Messages Received	Displays the number of broadcast messages received from the PLC(s).
Class 3 Messages/ Responses Transmitted	Displays the number of messages and responses sent to the PLC(s).
Class 1 Output Updates (From PLC)	The number of Class 1 output data updates received from the PLC or PLCs.
Class 1 Input Updates (To PLC)	The number of Class 1 input data updates sent to the PLC or PLCs.
Client Objects Requests	Displays the number of Class3 request messages received from the PLC(s).
Good Responses From PLC	The number of good responses received from the PLC.
Bad Responses From PLC	Displays the number of bad responses from messages sent to the PLC(s). Bad responses are typically returned for such errors as: • Incorrect tag or file names • Incorrect tag or file data types • Incorrect tag or file data sizes • PLC is overloaded and cannot handle the amount of Ethernet traffic • PLC malfunction
No Responses From PLC	Displays the number of no responses from messages sent to the PLC(s). No responses are typically returned for such errors as: • Incorrect IP address • Incorrect PLC configuration • PLC malfunction • PLC is overloaded and cannot handle the amount of Ethernet traffic
Invalid Network Paths	Displays the number of network path errors on messages sent to the PLC(s). These are typically caused by incorrect IP address settings.
Pending Request Limit Reached	Displays the number of pending request limit errors. These errors occur when the PLC is sending a continuous stream of messages to the DeviceMaster EIP-MOD faster than the DeviceMaster EIP-MOD can process them.
Unexpected Events	Displays the number of unexpected event errors. Unexpected event errors occur when the DeviceMaster EIP-MOD receives an unexpected message from the PLC such as an unexpected response or unknown message.
Unsupported CIP Class Errors	Displays the number of unsupported CIP request instance errors. These errors occur when a message with an invalid class is sent to the DeviceMaster EIP-MOD.
Unsupported CIP Instance Errors	Displays the number of unsupported CIP request instance errors. These errors occur when a message with an invalid instance is sent to the DeviceMaster EIP-MOD.
Unsupported CIP Service Errors	Displays the number of unsupported CIP request instance errors. These errors occur when a message with an invalid service is sent to the DeviceMaster EIP-MOD.
Unsupported CIP Attribute Errors	Displays the number of unsupported CIP request instance errors. These errors occur when a message with an invalid attribute is sent to the DeviceMaster EIP-MOD.
Improper Configuration Errors	Displays the number of improper configuration errors. These errors occur when the DeviceMaster EIP-MOD receives a message that cannot be performed due to an invalid configuration.

EtherNet/IP Interface Statistics Page (Continued)	
Invalid Message Data Errors	Displays the number of invalid message data errors. These errors occur when the DeviceMaster EIP-MOD receives a message that cannot be performed due to invalid data.
System Resource Errors	Displays the number of system resource errors. These errors indicate a system error on the DeviceMaster EIP-MOD such as an inoperable serial port or a full transmit queue. These errors typically occur when the PLC(s) are sending data to the DeviceMaster EIP-MOD faster than the DeviceMaster EIP-MOD can process it.
First Error Description	Text description of the first error that occurred.
Last Error Description	Text description of the last or most recent error that occurred.

8.10. Modbus to EtherNet/IP Tag/File Diagnostics Page

Access the *Modbus to EtherNet/IP Tag/File Diagnostics* page by clicking **Diagnostics | Data Mapping Diagnostics | Modbus to Tag/File**.

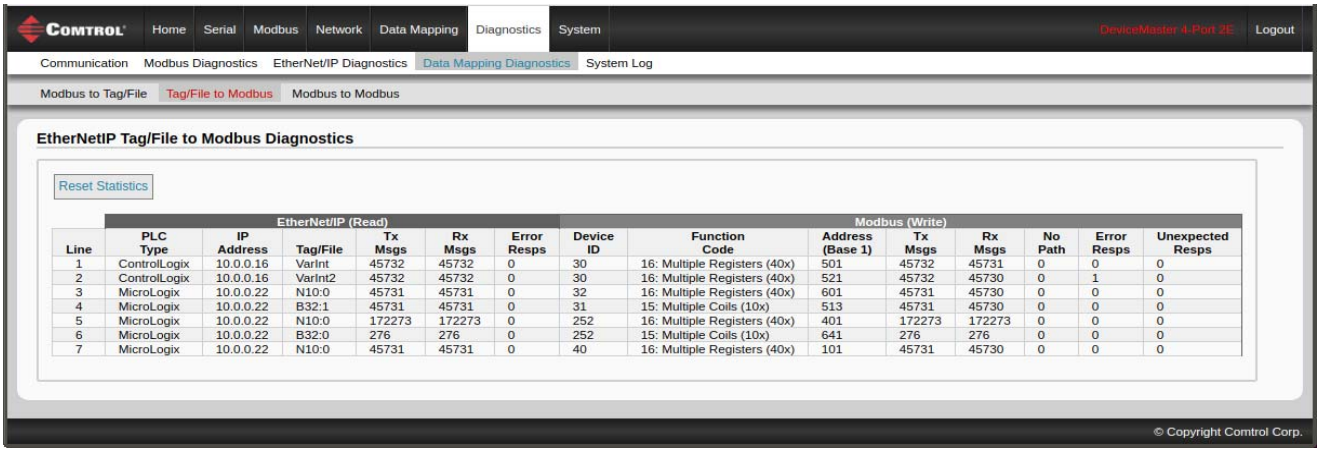


Modbus to EtherNet/IP Tag/File Diagnostics Page	
Line	Displays the configuration line number.
Device ID	Displays the Modbus Device ID that is being read. This may be the device ID of either the Shared Memory or a Modbus slave device.
Function code	Displays the function code being used to perform the read operation.
Address (base 1)	Displays the Modbus address in Base 1 format.
Tx Messages	Displays the number of read messages transmitted to the Shared Memory or Modbus device.
Rx Messages	Displays the number of valid read responses received from the Shared Memory or Modbus device.

Modbus to EtherNet/IP Tag/File Diagnostics Page (Continued)	
No Path	Displays the number of no path conditions. No path conditions occur when any of the following occur: • There is no valid destination to send the Modbus message. - There are no serial ports configured as slave ports. - There is no remote Modbus configuration for the device ID. • If there is a remote Modbus configuration, the gateway is unable to connect to the Modbus/TCP IP address.
Unexpected Responses	Displays the number of unexpected read responses received from the Shared Memory or Modbus device. These occur when an incorrect function code is received in the response, or a response message is received when it is not expected.
PLC Type	Displays the EtherNet/IP PLC type that is being written to.
IP Address	Displays the IP address of the EtherNet/IP PLC in xxx.xxx.xxx.xxx format.
Tag/File	Displays the EtherNet/IP PLC tag or file name.
Tx Messages	Displays the number of write messages transmitted to the EtherNet/IP PLC.
Rx Messages	Displays the number of valid write responses received from the EtherNet/IP PLC.
Error Responses	Displays the number of write response errors received from the EtherNet/IP PLC.
Reset Statistics	If click it resets the statistics to all zeros.
Error Responses	Displays the number of read response errors received from the Shared Memory or Modbus device.
Unexpected Responses	Displays the number of unexpected read responses received from the Shared Memory or Modbus device. These occur when an incorrect function code is received in the response, or a response message is received when it is not expected.
PLC Type	Displays the EtherNet/IP PLC type that is being written to.
IP Address	Displays the IP address of the EtherNet/IP PLC in xxx.xxx.xxx.xxx format.
Tag/File	Displays the EtherNet/IP PLC tag or file name.
Tx Messages	Displays the number of write messages transmitted to the EtherNet/IP PLC.
Rx Messages	Displays the number of valid write responses received from the EtherNet/IP PLC.
Error Responses	Displays the number of write response errors received from the EtherNet/IP PLC.
Reset Statistics	If click it resets the statistics to all zeros.

8.11. EtherNet/IP Tag/File to Modbus Diagnostics Page

Access the *EtherNet/IP Tag/File to Modbus Diagnostics* page by clicking **Diagnostics | Data Mapping Diagnostics | Tag/File to Modbus**.

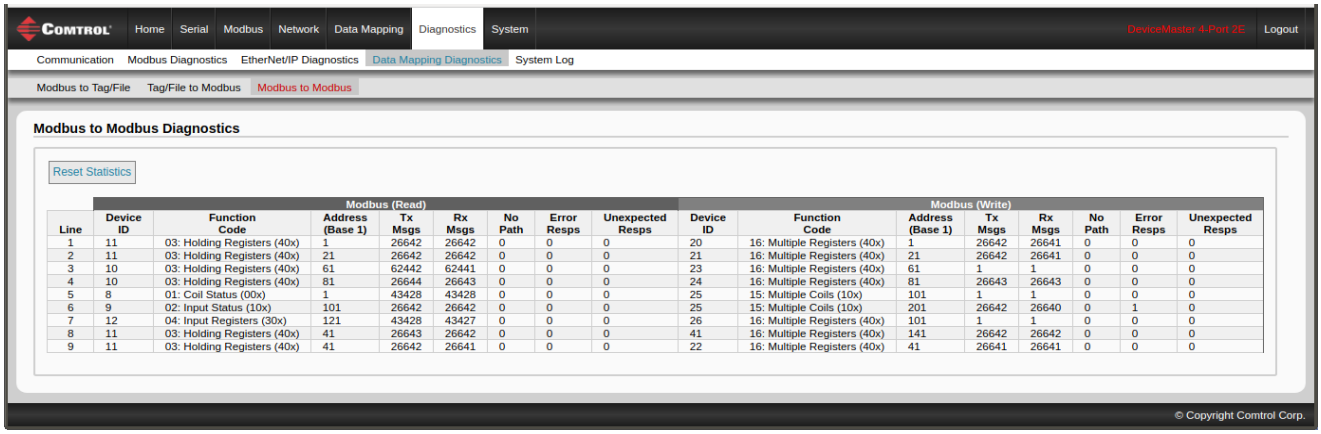


EtherNet/IP Tag/File to Modbus Diagnostics Page	
Line	Displays the configuration line number.
PLC Type	Displays the EtherNet/IP PLC type that is being read from.
IP Address	Displays the IP address of the EtherNet/IP PLC in xxx.xxx.xxx.xxx format.
Tag/File	Displays the EtherNet/IP PLC tag or file name.
Tx Messages	Displays the number of read messages transmitted to the EtherNet/IP PLC.
Rx Messages	Displays the number of valid read responses received from the EtherNet/IP PLC.
Error Responses	Displays the number of read response errors received from the EtherNet/IP PLC.
Device ID	Displays the Modbus Device ID that is being written to. This may be the device ID of either the Shared Memory or a Modbus slave device.
Function code	Displays the function being used to perform the write operation.
Address (base 1)	Displays the Modbus address in Base 1 format.
Tx Messages	Displays the number of write messages transmitted to the Shared Memory or Modbus device.
Rx Messages	Displays the number of valid write responses received from the Shared Memory or Modbus device.
No Path	Displays the number of no path conditions. No path conditions occur when any of the following occur: • There is no valid destination to send the Modbus message. - There are no serial ports configured as slave ports. - There is no remote Modbus configuration for the device ID. • If there is a remote Modbus configuration, the gateway is unable to connect to the Modbus/TCP IP address.
Error Responses	Displays the number of write response errors received from the Shared Memory or Modbus device.
Unexpected Responses	Displays the number of unexpected write responses received from the Shared Memory or Modbus device. These occur when an incorrect function code is received in the response, or a response message is received when it is not expected.

EtherNet/IP Tag/File to Modbus Diagnostics Page (Continued)	
Reset Statistics	Resets the statistics to all zero.

8.12. Modbus to Modbus Diagnostics Page

Access the *Modbus to Modbus Diagnostics* page by clicking **Diagnostics | Data Mapping Diagnostics | Modbus to Modbus**.



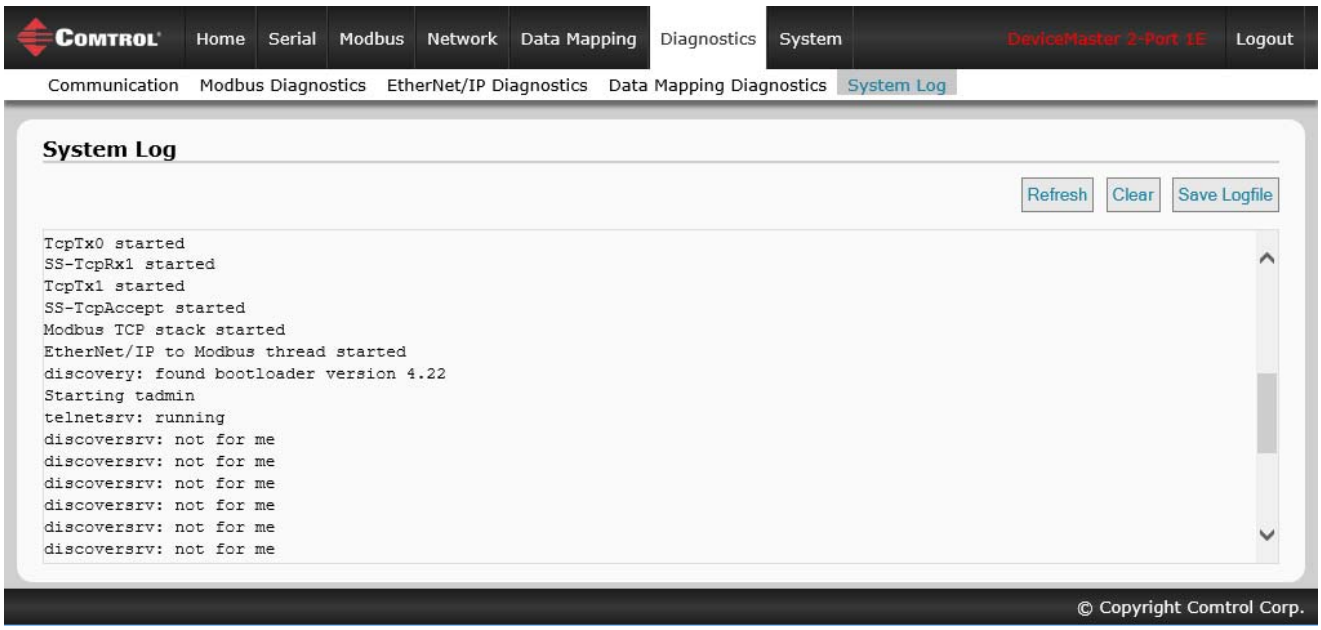
Modbus to Tag/File Diagnostics Page	
Line	Displays the configuration line number.
Device ID (Read)	Displays the Modbus Device ID that is being read. This may be the device ID of either the Shared Memory or a Modbus slave device.
Function code	Displays the function being used to perform the read operation.
Address (base 1)	Displays the Modbus address in Base 1 format.
Tx Messages	Displays the number of read messages transmitted to the Shared Memory or Modbus device.
Rx Messages	Displays the number of valid read responses received from the Shared Memory or Modbus device.
Error Responses	Displays the number of read response errors received from the Shared Memory or Modbus device.
No Path	Displays the number of no path conditions. No path conditions occur when any of the following occur: • There is no valid destination to send the Modbus message. - There are no serial ports configured as slave ports. - There is no remote Modbus configuration for the device ID. • If there is a remote Modbus configuration, the gateway is unable to connect to the Modbus/TCP IP address.
Unexpected Responses	Displays the number of unexpected read responses received from the Shared Memory or Modbus device. These occur when an incorrect function code is received in the response, or a response message is received when it is not expected.
Device ID (Write)	Displays the Modbus Device ID that is being written to. This may be the device ID of either the Shared Memory or a Modbus slave device.

Modbus to Tag/File Diagnostics Page	
Function code	Displays the function being used to perform the write operation.
Address (base 1)	Displays the Modbus address in Base 1 format.
Tx Messages	Displays the number of write messages transmitted to the Shared Memory or Modbus device.
Rx Messages	Displays the number of valid write responses received from the Shared Memory or Modbus device.
No Path	Displays the number of no path conditions. No path conditions occur when any of the following occur: • There is no valid destination to send the Modbus message. - There are no serial ports configured as slave ports. - There is no remote Modbus configuration for the device ID. • If there is a remote Modbus configuration, the gateway is unable to connect to the Modbus/TCP IP address.
Error Responses	Displays the number of write response errors received from the Shared Memory or Modbus device.
Invalid Responses	Displays the number of invalid write responses received from the Shared Memory or Modbus device. These occur when an incorrect function code is received in the response, or a response message is received when it is not expected.
Reset Statistics	Resets the statistics to all zeros.

8.13. System Log

The *System Log* page provides system level information, which is updated every 10 seconds.

- Click the **Refresh** button to view the latest system log information.
- Click the **Clear** button to restart a new system log file.
- Click the **Save Logfile** button to save the system log file, if requested by Technical Support.



Chapter 9. System Menus

This section discusses the web pages under the **System** menu:

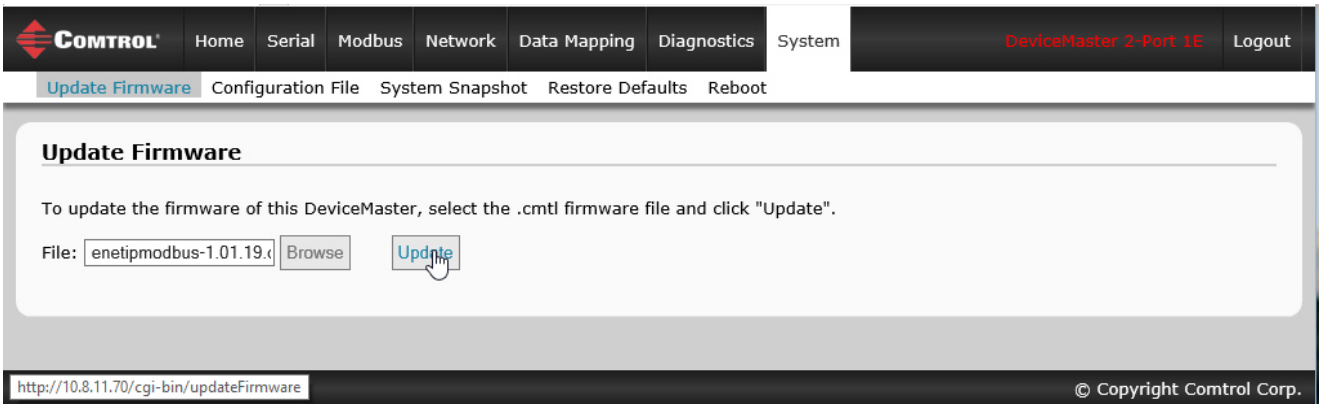
9.1. Update Firmware

You can upload firmware (EtherNet/IP-Modbus or Bootloader) using the **System | Update Firmware** page. You must first unpackage the firmware from the **.msi** file.

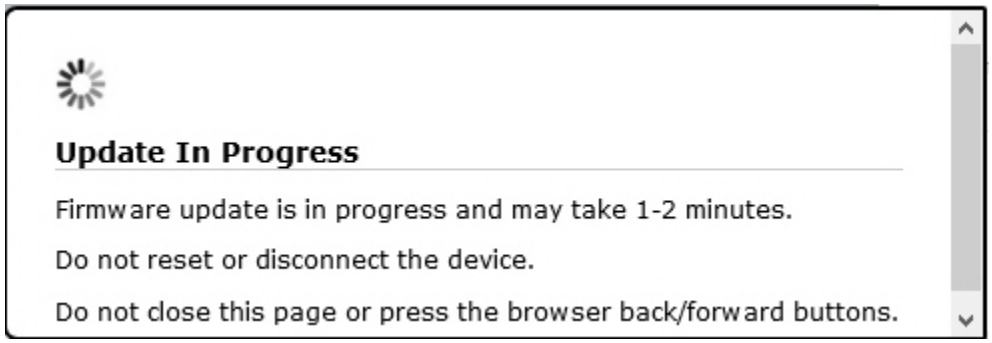
Note: *Optionally, you can use PortVision DX to upload firmware after unpackaging the* **.msi** *file.*

Use the following procedure to upload the latest firmware onto the DeviceMaster EIP-MOD.

1. If necessary, download the firmware from the *Comtrol download site*.
2. Execute the **enetipmodbus-x.x.msi** file.
3. Click the **Next** button.
4. After reviewing the license, click **I accept the terms in the License Agreement** and the **Next** button.
5. Click the **Next** button or browse to the location you want the files stored.
6. Click the **Install** button.
7. Click **Yes** to the **Do you want to allow this app to make changes to your device** pop up message.
8. Click the **Finish** button.
9. Open your web browser and enter the IP address of the DeviceMaster EIP-MOD.
10. Click the **System** menu, which opens the *Update Firmware* page.
11. Click the **Browse** button, navigate to the file, select it and click the **Open** button.
12. Click the **Update** button.



An *Update In Progress* pop up notifies you with the upload duration, not to reset or disconnect the device or to close the page.



9.2. Configuration File Page

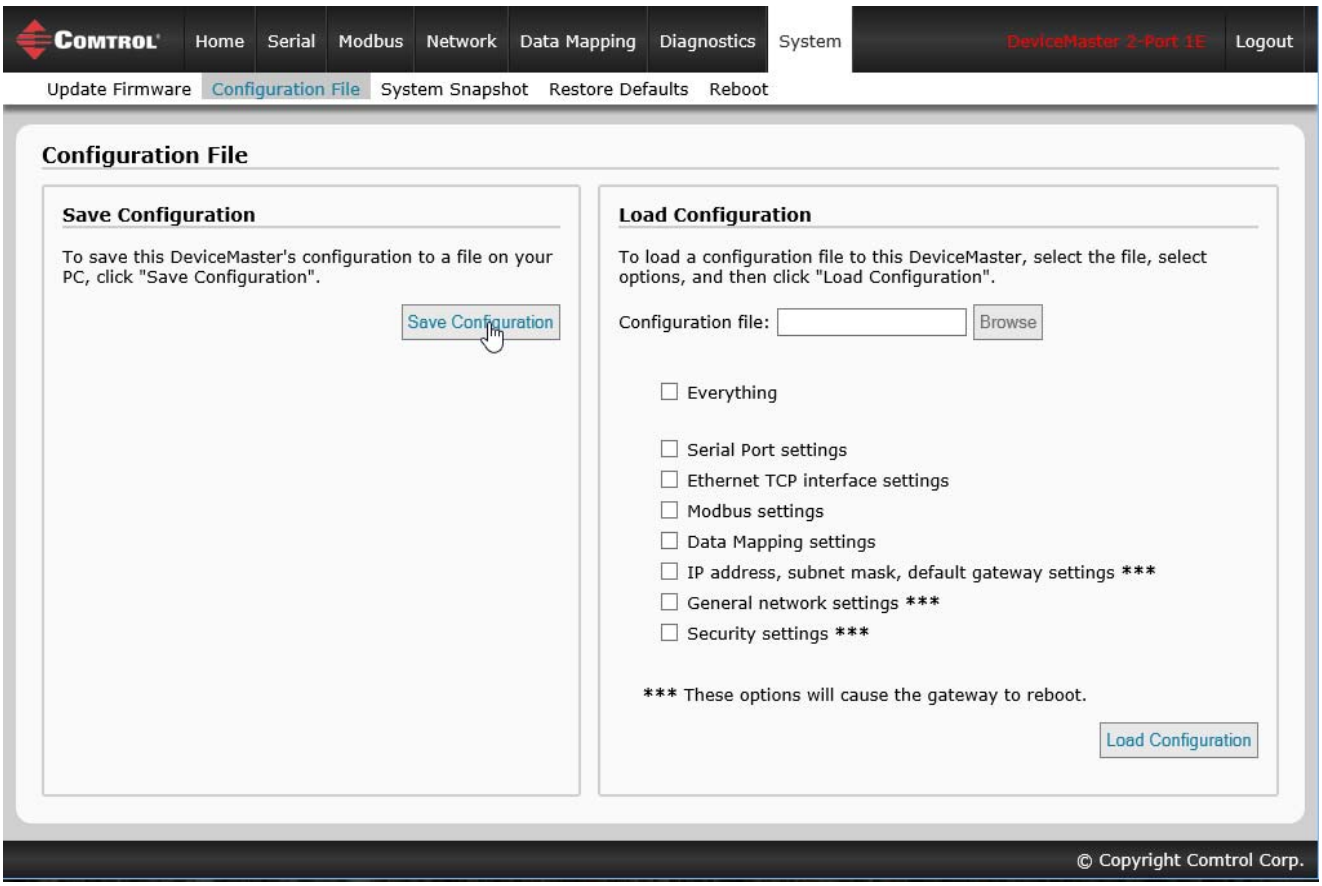
You can use the **Save Configuration** option to save a DeviceMaster EIP-MOD configuration file for recovery purposes or to quickly configure other DeviceMaster EIP-MODs that require the same configuration using the **Load Configuration** option.

Note: *Optionally, you can use PortVision DX to save and load configuration files.*

9.2.1. Saving a Configuration File

You can use this procedure to save a DeviceMaster EIP-MOD configuration file.

1. Click **System | Configuration File**.
2. Click the **Save Configuration** button.

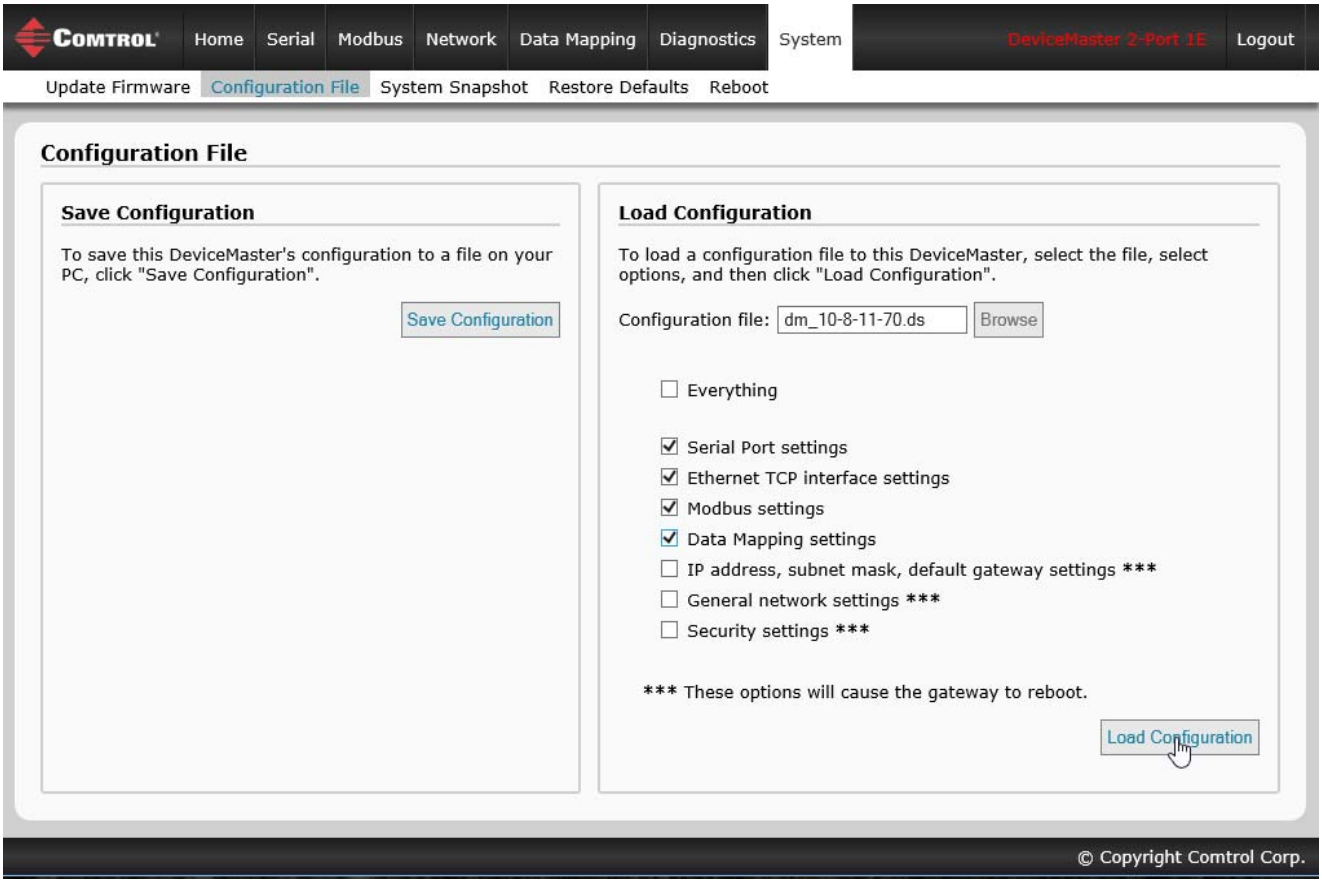


3. Save the configuration file following your browser prompts.

9.2.2. Loading a Configuration File

You can use this procedure to load a previously saved DeviceMaster EIP-MOD configuration file.

1. Click **System | Configuration File**.
2. Click the **Browse** button, highlight the configuration file you want to load, and click the **Open** button.
3. Select **Everything** or the items you want to load.

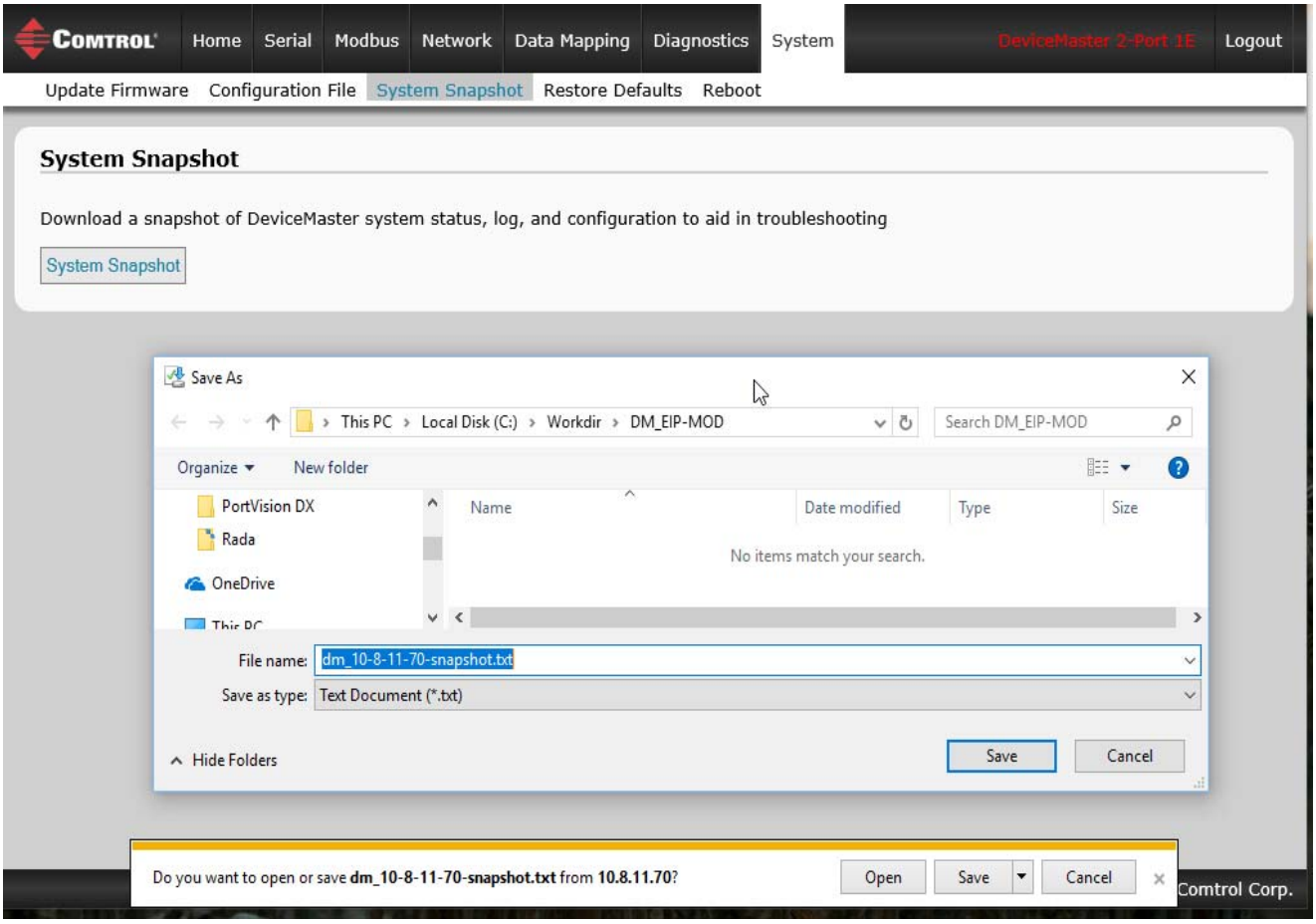


4. Click the **Load Configuration** button.

9.3. System Snapshot Page

You can use the *System Snapshot* page to download a snapshot of the device status, log, and configuration. You may find the information can help you diagnose a problem with the DeviceMaster EIP-MOD. In addition, this information may be requested by technical support if you have called for assistance.

1. Open your browser and enter the IP address of the DeviceMaster EIP-MOD.
2. Click **System | System Snapshot**.
3. Click the **Device Snapshot** button.

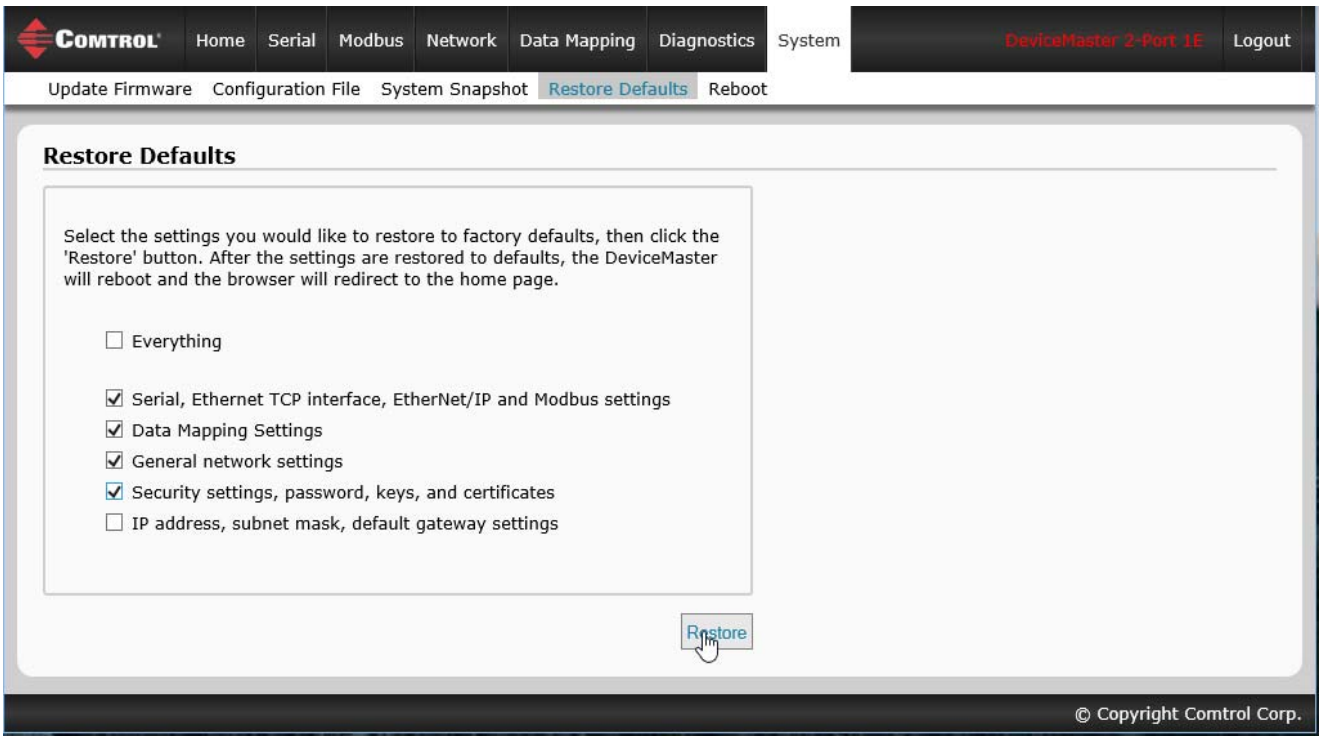


4. Save the file using the method for your browser.

9.4. Restore Defaults Page

You can easily some or all of your settings to factory defaults by using the procedure below.

1. Open your browser and enter the IP address of the DeviceMaster EIP-MOD.
2. Click **System | Restore Defaults**.
3. Select **Everything** or the specific setting or settings that you want to restore.

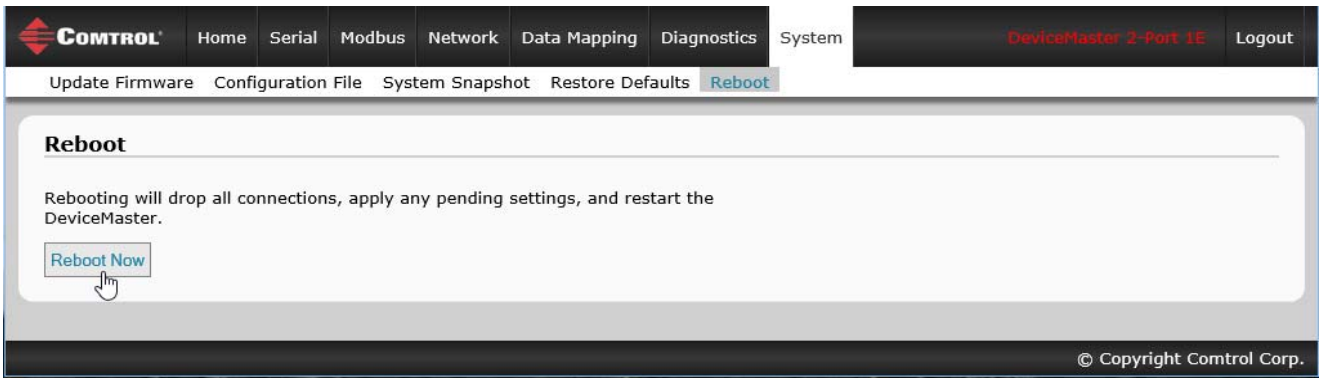


4. Click the **Restore** button.
5. The DeviceMaster EIP-MOD reboots and re-opens the web interface.

9.5. Reboot

You can reboot the DeviceMaster EIP-MOD remotely using the *Reboot* web page.

1. Click **System | Reboot**.
2. Click the **Reboot Now** button or wait the 10 seconds for it automatically reboot.

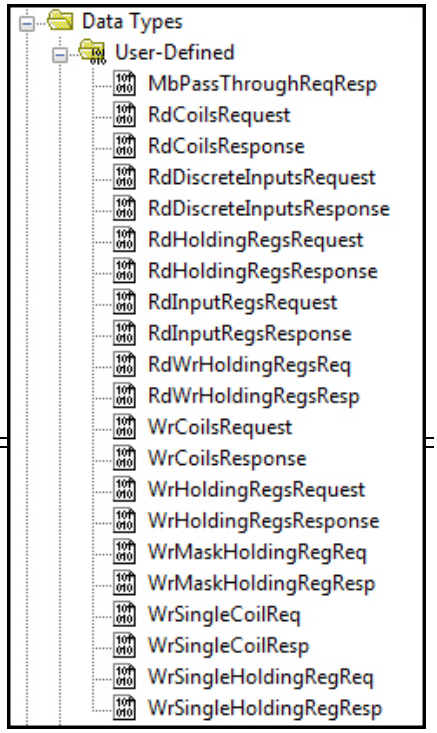


Chapter 10. PLC Programming for Modbus Object (44 Hex)

The CIP Modbus Object is supported on the DeviceMaster EtherNet/IP-Modbus gateway. This functionality provides the ability to send read and write messages directly to the Shared Memory or a Modbus slave device. Each EtherNet/IP MSG instruction requires some setup. The following is a programming guide to how to set up MSG instructions and message data for each Modbus function.

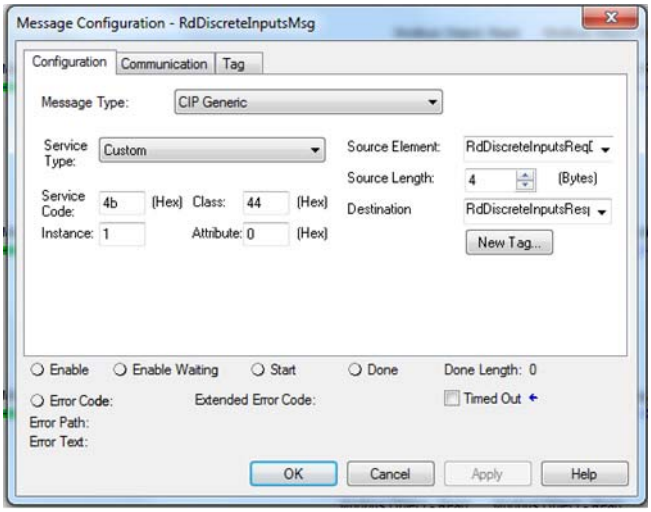
The provided PLC example program utilizes standard MSG instructions and User Defined data structures to format the Modbus request and response parameters.

The list (right) of the User Defined data structures from the example PLC program.



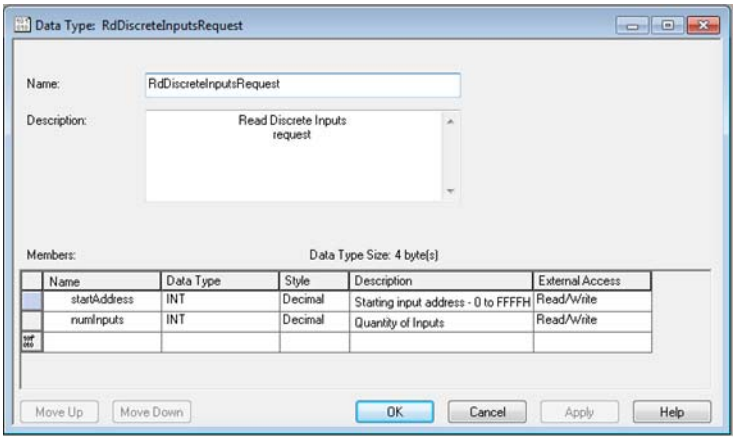
10.1. Read Discrete Inputs Service (4B Hex)

This service reads one or more discrete inputs from the Modbus Discrete Inputs table. This service results in the DeviceMaster EIP-MOD issuing a Read Discrete Inputs function (function code 0x02) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3* and/or *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

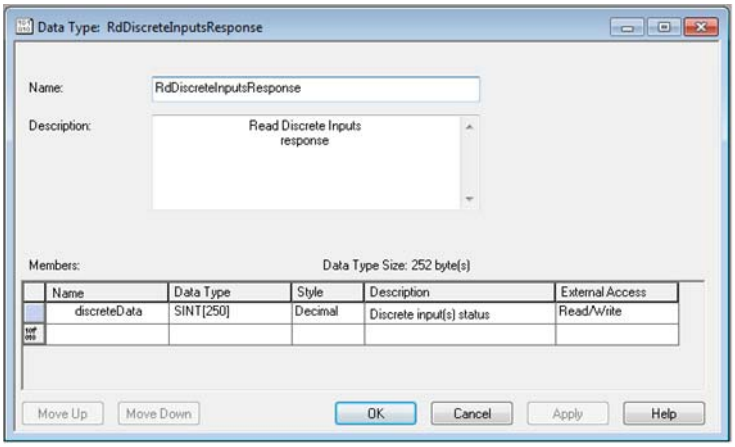


Parameter	Value	Description
Service Code	4b Hex	Directly corresponds to Modbus function code: Read Discrete Inputs (function code 2)
Instance	1	Modbus Device ID of 1 (Valid Range = 1 to 255)

Parameter	Value	Description
Class	44 Hex	Modbus Object
Attribute	0	Not used
Source Element	RdDiscreteInputsReqData	Tag of structure type RdDiscreteInputsRequest
Source Length	4	Length in bytes of RdDiscreteInputsReqData
Destination	RdDiscreteInputsRespData	Tag of structure type RdDiscreteInputsResponse



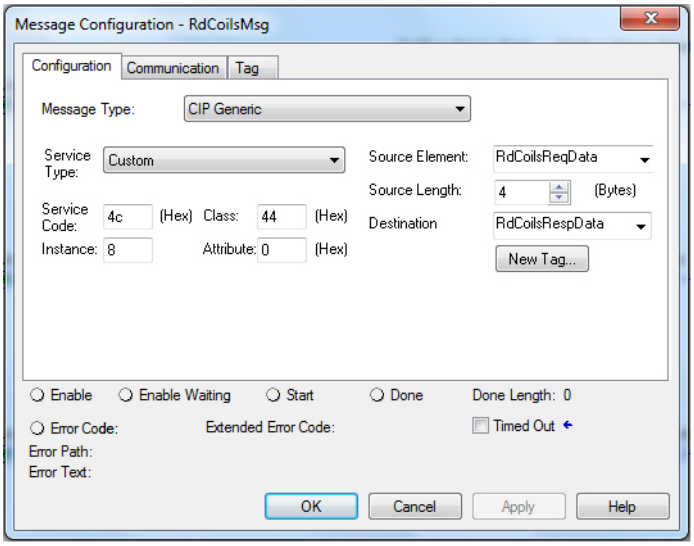
Parameter	Data Type	Range	Description
startAddress	INT (16 bit WORD)	0 to FFFF hex	Starting address base 0
numInputs	INT (16 bit WORD)	1 to 2000 (7D0 hex)	Number of inputs represented by a single bit.



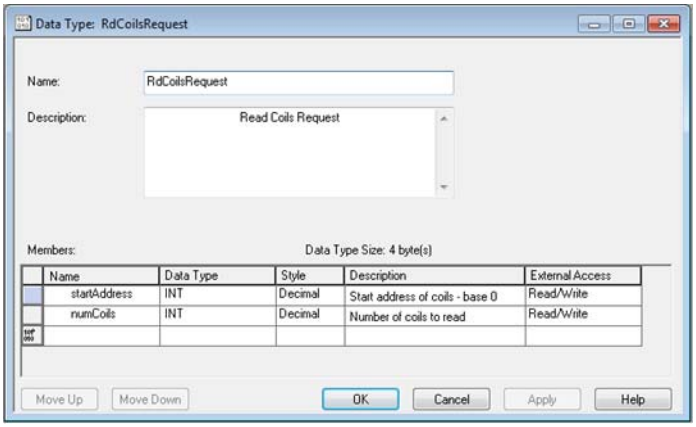
Parameter	Data Type	Range	Description
discreteData	SINT array (8 bit BYTEs)	0 to FF hex	Input data with each input being represented by a single bit.

10.2. Read Coils Service (4C Hex)

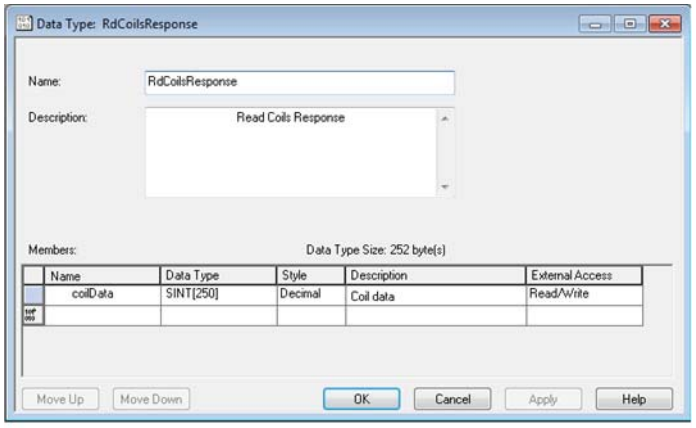
This service reads one or more coils from the Modbus Coils table. This service results in the DeviceMaster EIP-MOD issuing a Read Coils function (function code 0x01) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3* and/or *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.



Parameter	Value	Description
Service Code	4C hex	Directly corresponds to Modbus function code: Read Coils (function code 1)
Instance	8	Modbus Device ID of 8 (Valid Range = 1 to 255)
Class	44 hex	Modbus Object
Attribute	0	Not used
Source Element	RdCoilsReqData	Tag of structure type RdCoilsRequest
Source Length	4	Length in bytes of RdCoilsReqData
Destination	RdCoilsRespData	Tag of structure type RdCoilsResponse



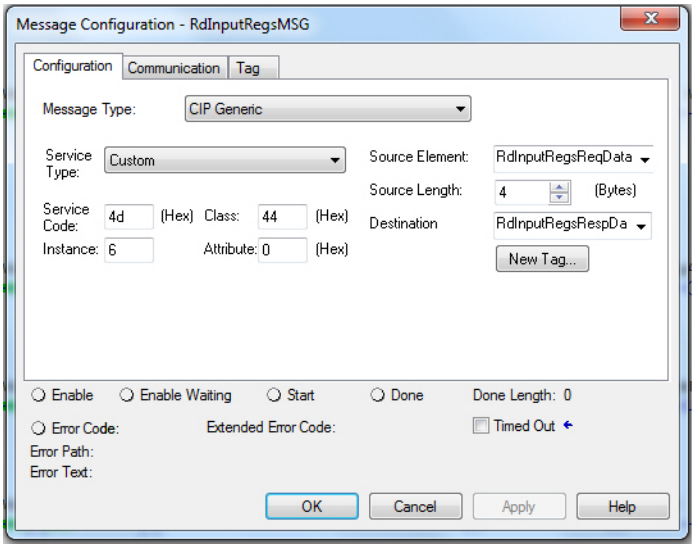
Parameter	Data Type	Range	Description
startAddress	INT (16 bit WORD)	0 to FFFF hex	Starting address base 0
numCoils	INT (16 bit WORD)	1 to 2000 (7D0 hex)	Number of coils represented by a single bit.



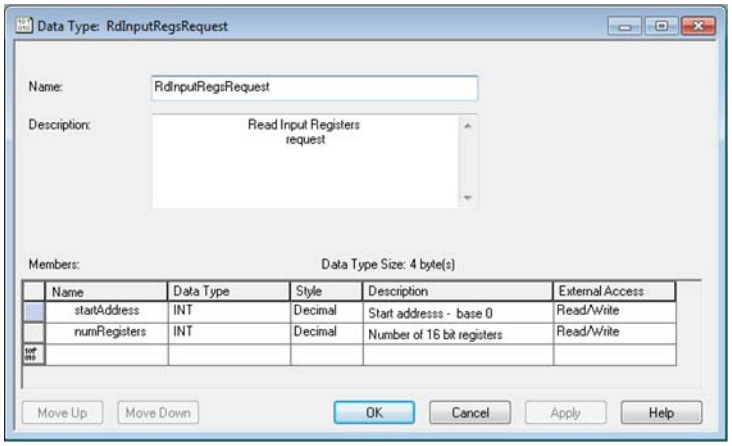
Parameter	Data Type	Range	Description
coilData	SINT array (8 bit BYTEs)	0 to FF hex	Coil data with coil each being represented by a single bit.

10.3. Read Input Registers Service (4D Hex)

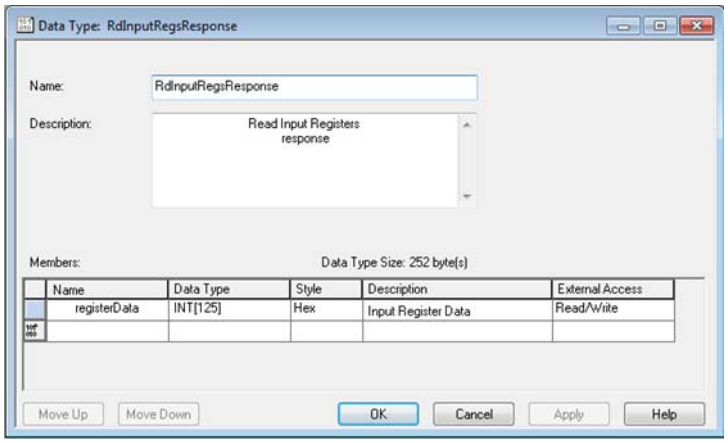
This service reads one or more input registers from the Modbus Input Register table. This service results in the DeviceMaster EIP-MOD issuing a Read Input Registers function (function code 0x04) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3* and/or *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.



Parameter	Value	Description
Service Code	4D hex	Directly corresponds to Modbus function code: Read Input Registers (function code 4)
Instance	6	Modbus Device ID of 6 (Valid Range = 1 to 255)
Class	44 hex	Modbus Object
Attribute	0	Not used
Source Element	RdInputRegsReqData	Tag of structure type RdInputRegsRequest
Source Length	4	Length in bytes of RdInputRegsReqData
Destination	RdInputRegsRespData	Tag of structure type RdInputRegsResponse



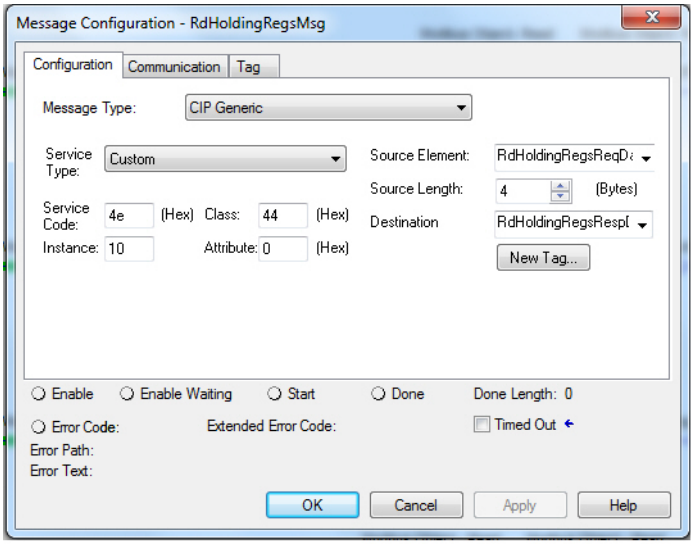
Parameter	Data Type	Range	Description
startAddress	INT (16 bit WORD)	0 to FFFF hex	Starting address base 0
numRegisters	INT (16 bit WORD)	1 to 125 (7D0 hex)	Number input registers to read.



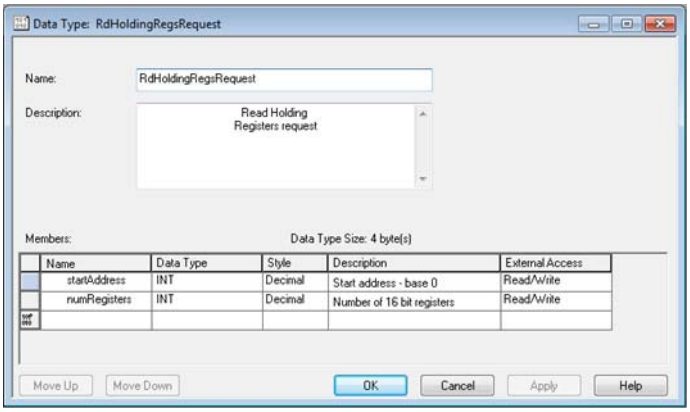
Parameter	Data Type	Range	Description
registerData	INT (16 bit WORD)	0 to FF hex	Input register data.

10.4. Read Holding Registers Service (4E Hex)

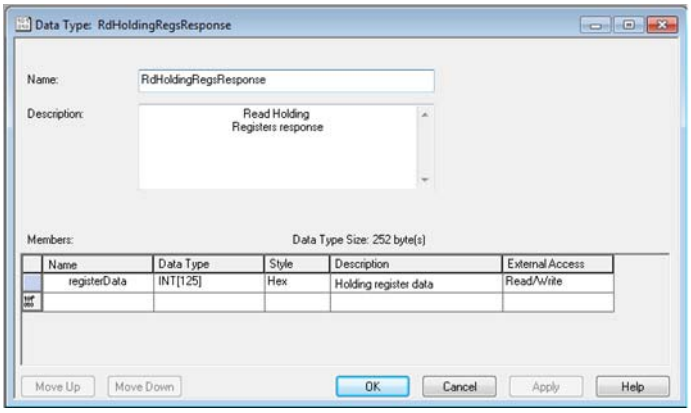
This service reads one or more holding registers from the Modbus Holding Register table. This service results in the DeviceMaster EIP-MOD issuing a Read Holding Registers function (function code 0x03) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3* and/or *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.



Parameter	Value	Description
Service Code	4E hex	Directly corresponds to Modbus function code: Read Input Registers (function code 3)
Instance	10	Modbus Device ID of 10 (Valid Range = 1 to 255)
Class	44 hex	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



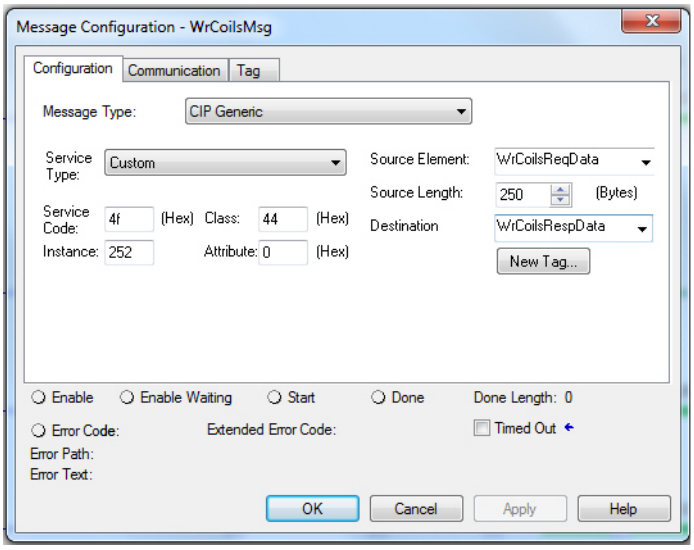
Parameter	Data Type	Range	Description
startAddress	INT (16 bit WORD)	0 to FFFF hex	Starting address base 0
numRegisters	INT (16 bit WORD)	1 to 125 (7D0 hex)	Number holding registers to read.



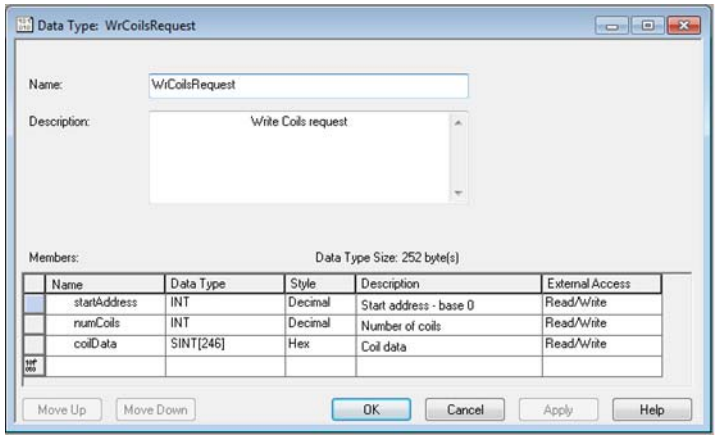
Parameter	Data Type	Range	Description
registerData	INT (16 bit WORD)	0 to FF hex	Holding register data.

10.5. Write Coils Service (4F Hex)

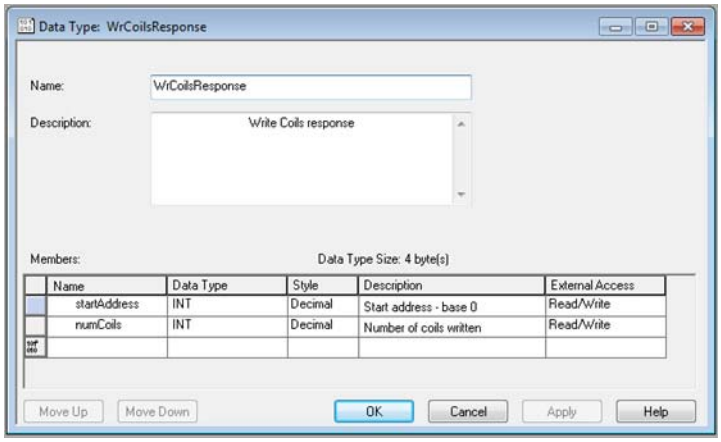
This service writes one or more coils to the Modbus Coils table. This service results in the DeviceMaster EIP-MOD issuing a Write Multiple Coils function (function code 0x0F) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3* and/or *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.



Parameter	Value	Description
Service Code	4f hex	Directly corresponds to Modbus function code: Write Coils (function code 15)
Instance	252	Modbus Device ID of 252 (Valid Range = 1 to 255)
Class	44 hex	Modbus Object
Attribute	0	Not used
Source Element	WrCoilsReqData	Tag of structure type WrCoilsRequest
Source Length	250	Maximum Length in bytes of WrCoilsReqData (Length must be greater or equal to length of request message.)
Destination	WrCoilsRespData	Tag of structure type WrCoilsResponse



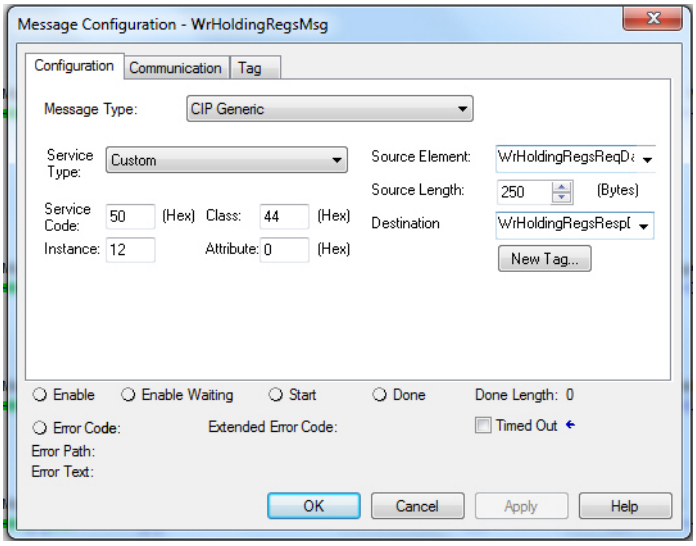
Parameter	Data Type	Range	Description
startAddress	INT (16 bit WORD)	0 to FFFF hex	Starting address base 0
numCoils	INT (16 bit WORD)	1 to 1968 (7b0 hex)	Number of coils represented by a single bit.
coilData	SINT array	00 to FF hex	Coil data to write



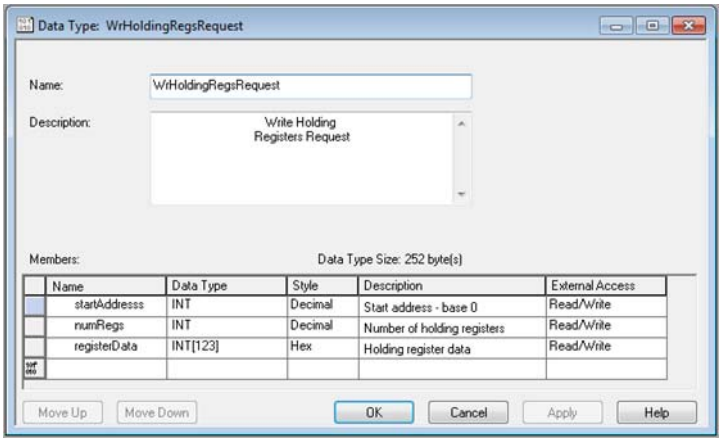
Parameter	Data Type	Range	Description
startAddress	INT (16 bit WORD)	0 to FFFF hex	Starting address base 0
numCoils	INT (16 bit WORD)	1 to 1968 (7b0 hex)	Number of coils written

10.6. Write Holding Registers Service (50 Hex)

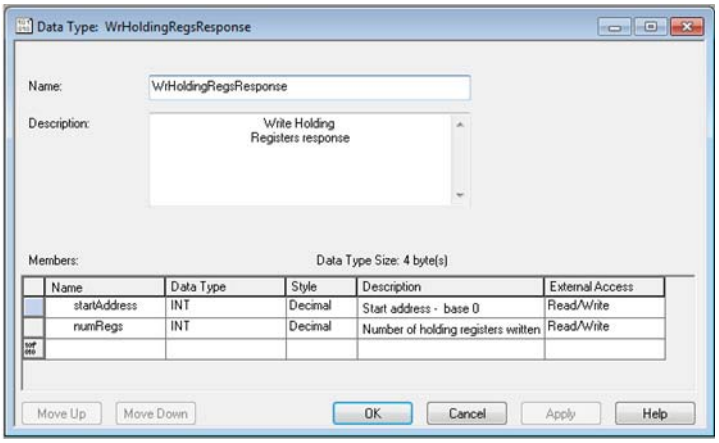
This service writes one or more holding registers to the Modbus Holding Registers table. This service results in the DeviceMaster EIP-MOD issuing a Write Multiple Registers function (function code 0x10) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *CIP Standard Vol1_3.22 Section 5B-3* and/or *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.



Parameter	Value	Description
Service Code	50 hex	Directly corresponds to Modbus function code: Write Holding Registers (function code 16)
Instance	12	Modbus Device ID of 12 (Valid Range = 1 to 255)
Class	44 hex	Modbus Object
Attribute	0	Not used
Source Element	WdHoldingRegsReqData	Tag of structure type WrHoldingRegsRequest
Source Length	250	Maximum length in bytes of WrHoldingRegsReqData (Length must be greater or equal to length of request message.)
Destination	WdHoldingRegsRespData	Tag of structure type WrHoldingRegsResponse

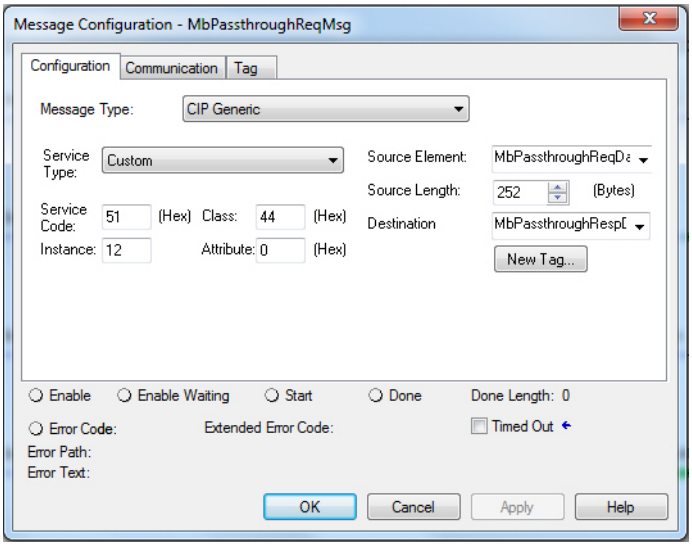


Parameter	Data Type	Range	Description
startAddress	INT (16 bit WORD)	0 to FFFF hex	Starting address base 0
numRegs	INT (16 bit WORD)	1 to 123	Number holding registers to write.
registerData	INT array	0-FFFF hex	Holding register data

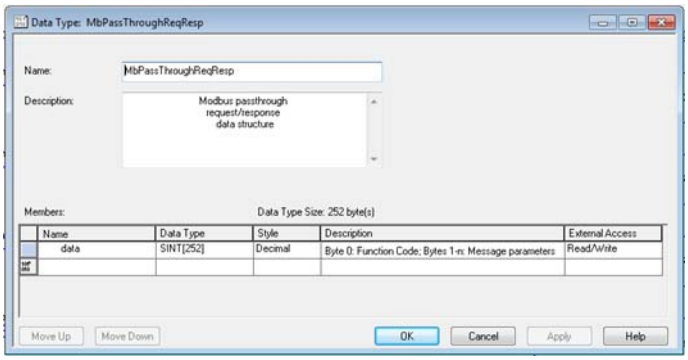


Parameter	Data Type	Range	Description
startAddress	INT (16 bit WORD)	0 to FFFF hex	Starting address base 0
numRegs	INT (16 bit WORD)	1 to 123	Number holding registers written

DeviceMaster EIP-MOD does not attempt to perform any endian conversion on the bytes in the data stream (either request or response). Refer to *CIP Standard Vol1_3.22 Section 5B-3* and/or *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.



Parameter	Value	Description
Service Code	51 hex	Service code for a Modbus passthrough message.
Instance	12	Modbus Device ID of 12 (Valid Range = 1 to 255)
Class	44 hex	Modbus Object
Attribute	0	Not used
Source Element	MbPassthroughReqData	Tag of structure type MbPassthroughReqResp
Source Length	252	Maximum length in bytes of MbPassthroughReqData (Length must be greater or equal to length of request message.)
Destination	MbPassthroughReqData	Tag of structure type MbPassthroughReqResp

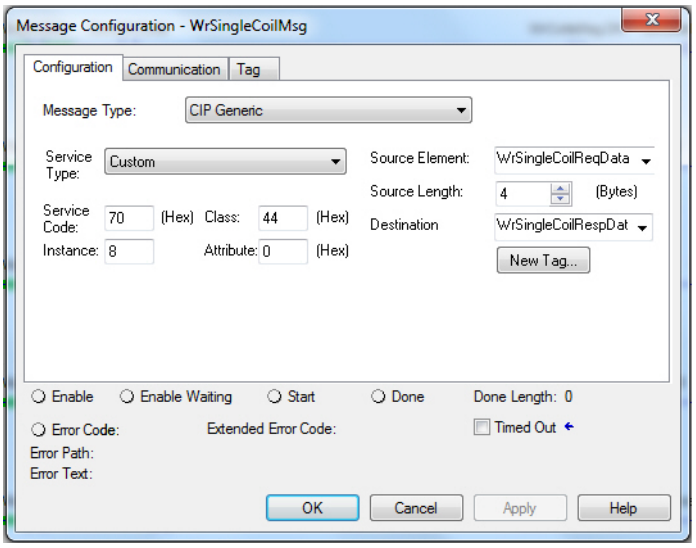


Parameter	Data Type	Range	Description
data	SINT array	0 to FF hex	Modbus Message to be passed through. Byte 0: Function Code Bytes 1-n: Message parameters ***Note:*** • PLC operates in little endian format. • Message request parameters must be byte swapped to big endian format. • Message response parameters will be received in big endian format and may require byte swapping back to PLC format.

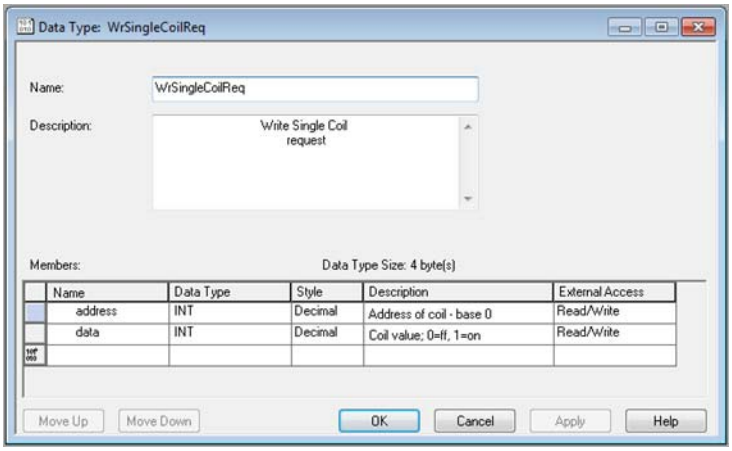
10.7. Vendor Specific Write Single Coil Service (70 Hex)

This service writes one coil to the Modbus Coils table. This service results in the DeviceMaster EIP-MOD issuing a Write Single Coil function (function code 0x05) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

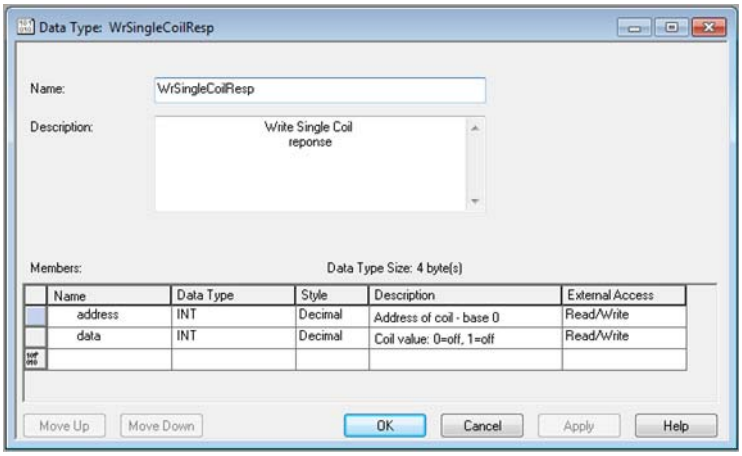
Note: *In order to use this service, the Enable Vendor Specific Modbus Object Services option on the EtherNet / IP Stack Configuration web page must be selected.*



Parameter	Value	Description
Service Code	70 hex	Directly corresponds to Modbus function code: Write Single Coil (function code 5)
Instance	8	Modbus Device ID of 8 (Valid Range = 1 to 255)
Class	44 hex	Modbus Object
Attribute	0	Not used
Source Element	WrSingleCoilReqData	Tag of structure type WrSingleCoilRequest
Source Length	4	Length in bytes of WrSingleCoilReqData
Destination	WrSingleCoilRespData	Tag of structure type WrSingleCoilResponse



Parameter	Data Type	Range	Description
address	INT (16 bit WORD)	0 to FFFF hex	Coil address base 0
data	INT (16 bit WORD)	0 = Off 1 = On	Coil data to write

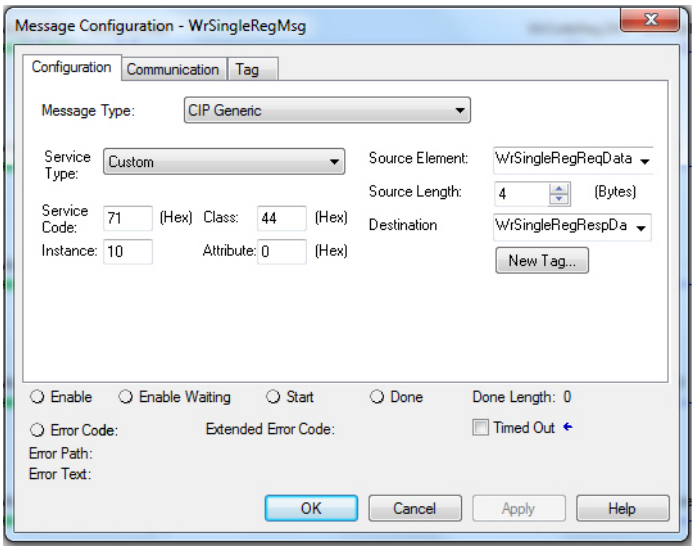


Parameter	Data Type	Range	Description
address	INT (16 bit WORD)	0 to FFFF hex	Coil address base 0
data	INT (16 bit WORD)	0 = Off 1 = On	Coil data written

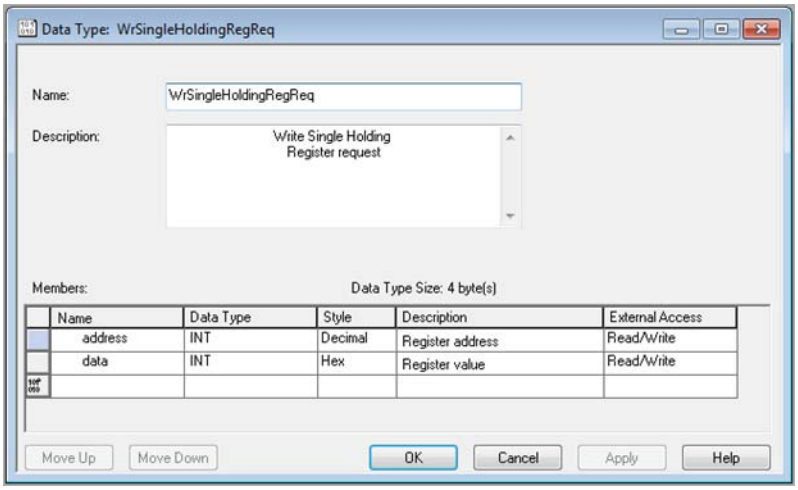
10.8. Vendor Specific Write Single Register Service (71 Hex)

This service writes one holding register to the Modbus Holding Registers table. This service results in the DeviceMaster EIP-MOD issuing a Write Single Registers function (function code 0x06) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

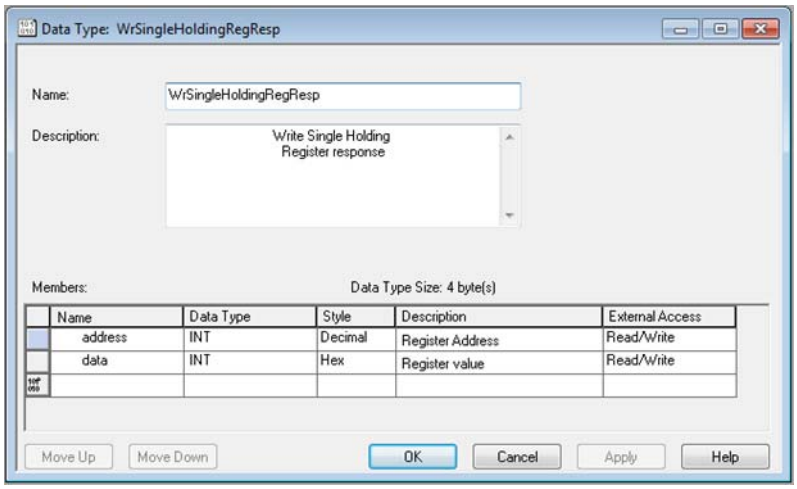
Note: *In order to use this service, the Enable Vendor Specific Modbus Object Services option on the EtherNet / IP Stack Configuration web page must be selected.*



Parameter	Value	Description
Service Code	71 hex	Directly corresponds to Modbus function code: Write Single Holding Register (function code 6)
Instance	10	Modbus Device ID of 10 (Valid Range = 1 to 255)
Class	44 hex	Modbus Object
Attribute	0	Not used
Source Element	WrSingleRegReqData	Tag of structure type WrSingleHoldingRegRequest
Source Length	4	Length in bytes of WrSingleRegReqData
Destination	WrSingleRegRespData	Tag of structure type WrSingleHoldingRegResponse



Parameter	Data Type	Range	Description
address	INT (16 bit WORD)	0 to FFFF hex	Holding register address base 0
data	INT (16 bit WORD)	0 to FFFF hex	Register data to write

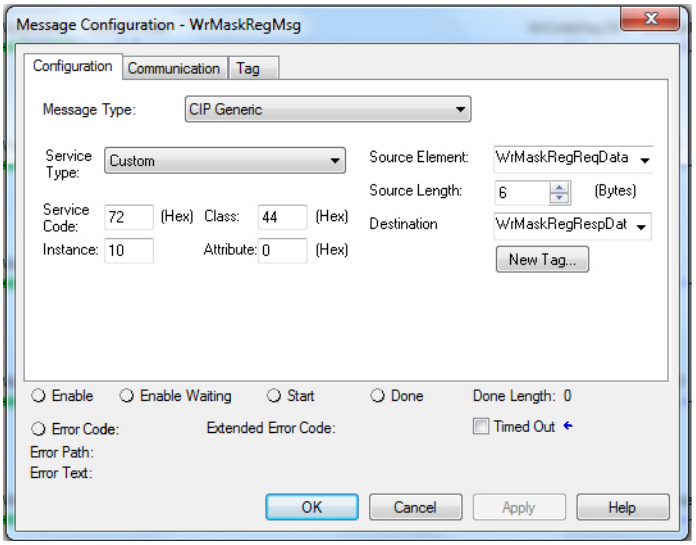


Parameter	Data Type	Range	Description
address	INT (16 bit WORD)	0 to FFFF hex	Holding register address base 0
data	INT (16 bit WORD)	0 to FFFF hex	Register data to write

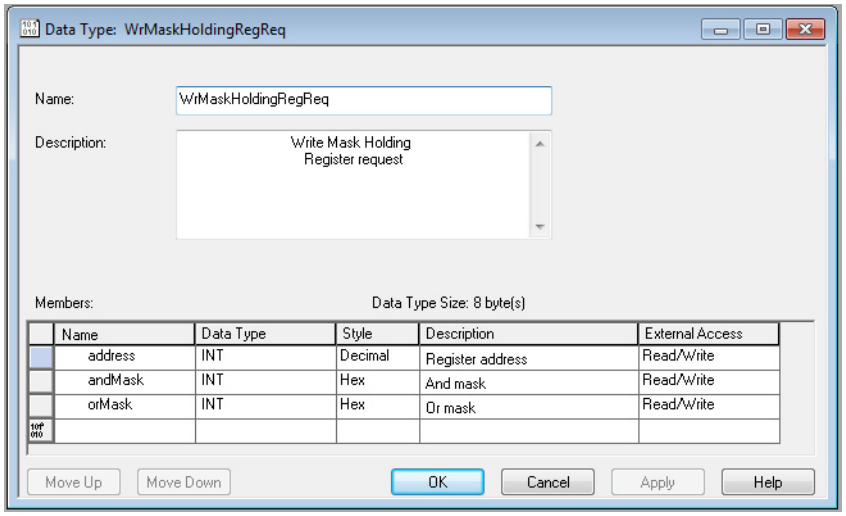
10.9. Vendor Specific Write Mask Register (72 Hex)

This service performs a write mask operation on one holding register in the Modbus Holding Registers table. This service results in the DeviceMaster EIP-MOD gateway issuing a Write Mask Register function (function code 0x16) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

Note: *In order to use this service, the Enable Vendor Specific Modbus Object Services option on the EtherNet / IP Stack Configuration web page must be selected.*



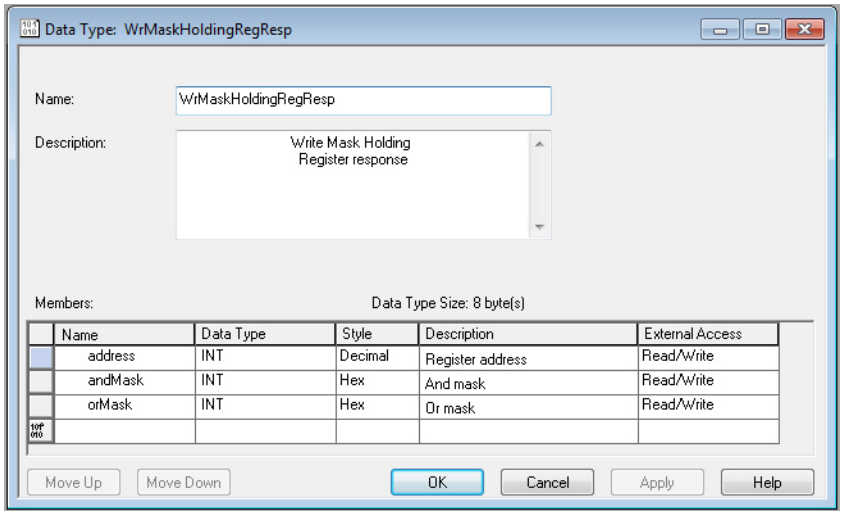
Parameter	Value	Description
Service Code	72 hex	Directly corresponds to Modbus function code: Write Single Holding Register (function code 22)
Instance	10	Modbus Device ID of 10 (Valid Range = 1 to 255)
Class	44 hex	Modbus Object
Attribute	0	Not used
Source Element	WrMaskRegReqData	Tag of structure type WrMaskHoldingRegReq
Source Length	6	Length in bytes of WrMaskRegReqData
Destination	WrMaskRegRespData	Tag of structure type WrMaskHoldingRegResp



Parameter	Data Type	Range	Description
address	INT (16 bit WORD)	0 to FFFF hex	Holding register address base 0
andMask	INT (16 bit WORD)	0 to FFFF hex	And mask
orMask	INT (16 bit WORD)	0 to FFFF hex	Or mask

1. Contents of register is ANDed with andMask.
2. The orMask is ANDed with ones compliment of andMask.
3. Results of step 1 is ORed with results of step 2.

In “C” language coding syntax:
Result = (contents & andMask) | (orMask & (~andMask);

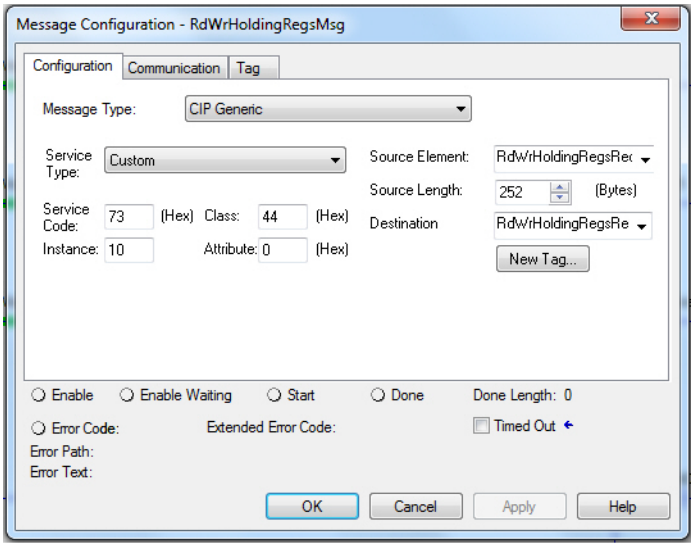


Parameter	Data Type	Range	Description
address	INT (16 bit WORD)	0 to FFFF hex	Holding register address base 0
andMask	INT (16 bit WORD)	0 to FFFF hex	And mask
orMask	INT (16 bit WORD)	0 to FFFF hex	Or mask

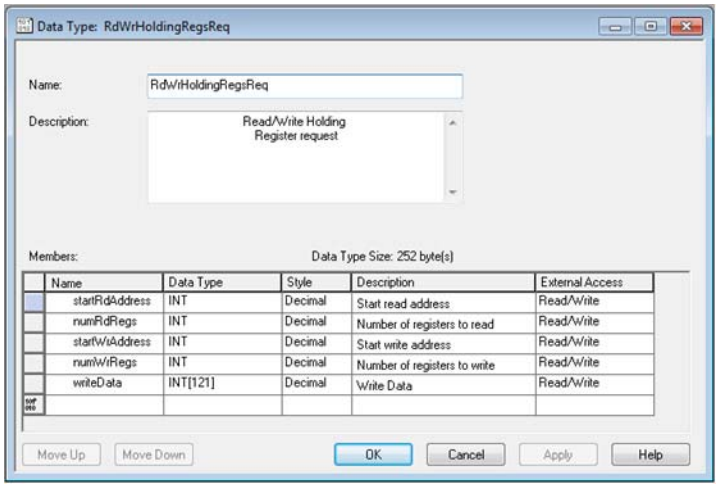
10.10. Vendor Specific Read/Write Holding Registers (73 Hex)

This service performs first write and then read operations on multiple holding registers in the Modbus Holding Registers table. This service results in the DeviceMaster EIP-MOD gateway issuing a Read/Write Holding Registers function (function code 0x17) to either the Shared Memory or the Modbus Message Routing subsystem. Refer to *MODBUS Application Protocol Specification V1.1b3.pdf* at www.modbus.org for more details.

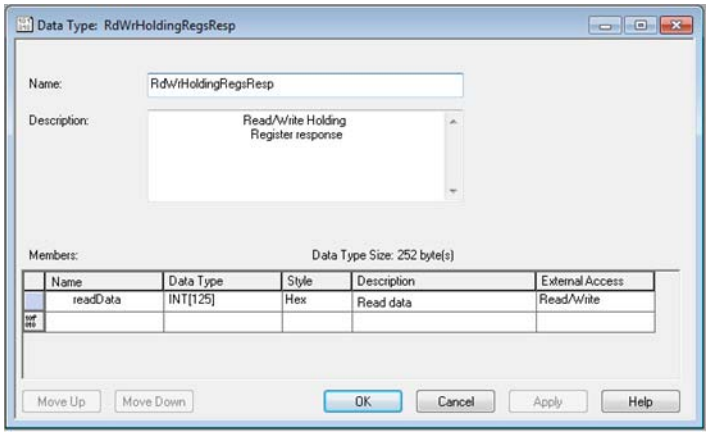
Note: *In order to use this service, the Enable Vendor Specific Modbus Object Services option on the EtherNet/IP Stack Configuration web page must be selected.*



Parameter	Value	Description
Service Code	73 hex	Directly corresponds to Modbus function code: Read/Write Holding Registers (function code 23)
Instance	10	Modbus Device ID of 10 (Valid Range = 1 to 255)
Class	44 hex	Modbus Object
Attribute	0	Not used
Source Element	RdWrHoldingRegsReqData	Tag of structure type RdWrHoldingRegsReq
Source Length	252	Maximum length in bytes of RdWrHoldingRegsReqData (Length must be greater or equal to length of request message.)
Destination	RdWrHoldingRegsRespData	Tag of structure type RdWrHoldingRegsResp



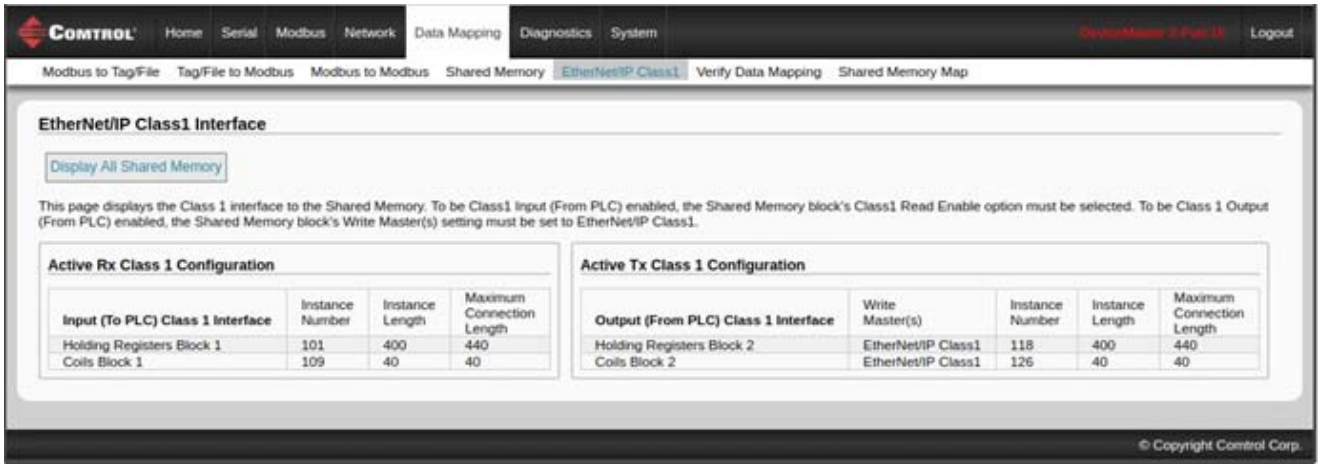
Parameter	Data Type	Range	Description
startRdAddress	INT (16 bit WORD)	0 to FFFF hex	Start read holding register address base 0
numRdRegs	INT (16 bit WORD)	0 to 125	Number of registers to read
startWrAddress	INT (16 bit WORD)	0 to FFFF hex	Start write holding register address base 0
numWrRegs	INT (16 bit WORD)	0 to 121	Number of registers to write
writeData	INT array	0 to FFFF hex	Data to write



Parameter	Data Type	Range	Description
readData	INT array	0 to FFFF hex	Data that was read

Chapter 11. Class 1 Interface Definition

The EtherNet/IP Class 1 interface web page displays the Class 1 parameters.



Where the following Class1 interfaces are supported:

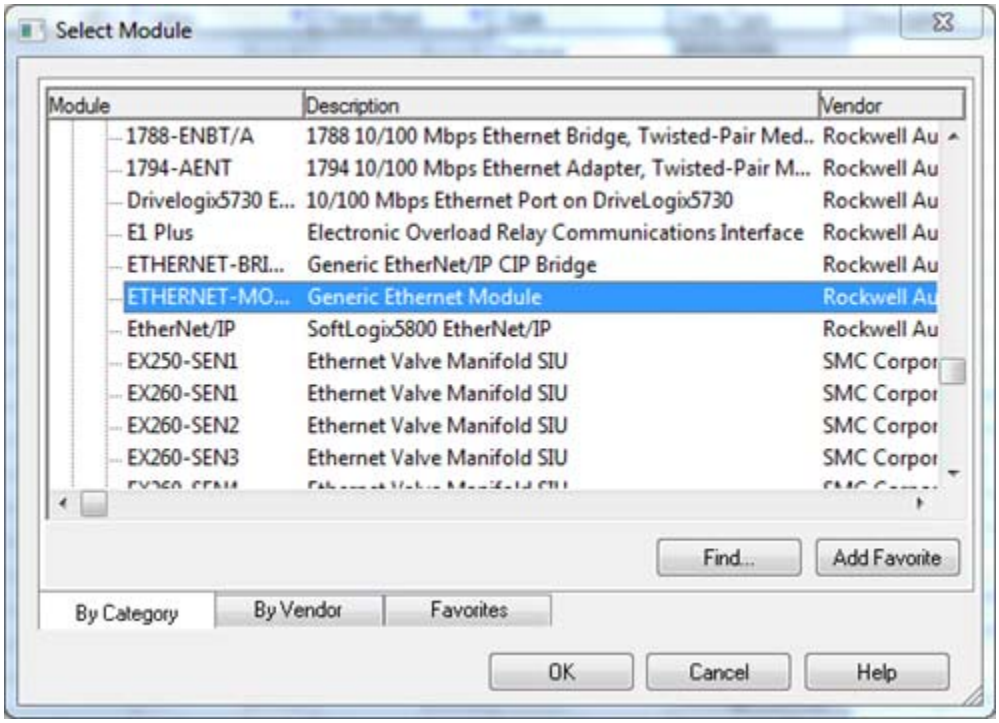
	Instance	Length (in bytes)
Input (To PLC)		
Holding Register Block 1 Only	101	1 - 400
Coil Block 1 Only	109	1 - 40
Both Holding Register Block 1 and Coil Block 1	101	1 - 440
Output (From PLC)		
Holding Register Block 2 Only	118	400
Coil Block 2 Only	126	40
Both Holding Register Block 2 and Coil Block 2	118	440

11.1. Configuring an I/O Ethernet Module on a ControlLogix PLC

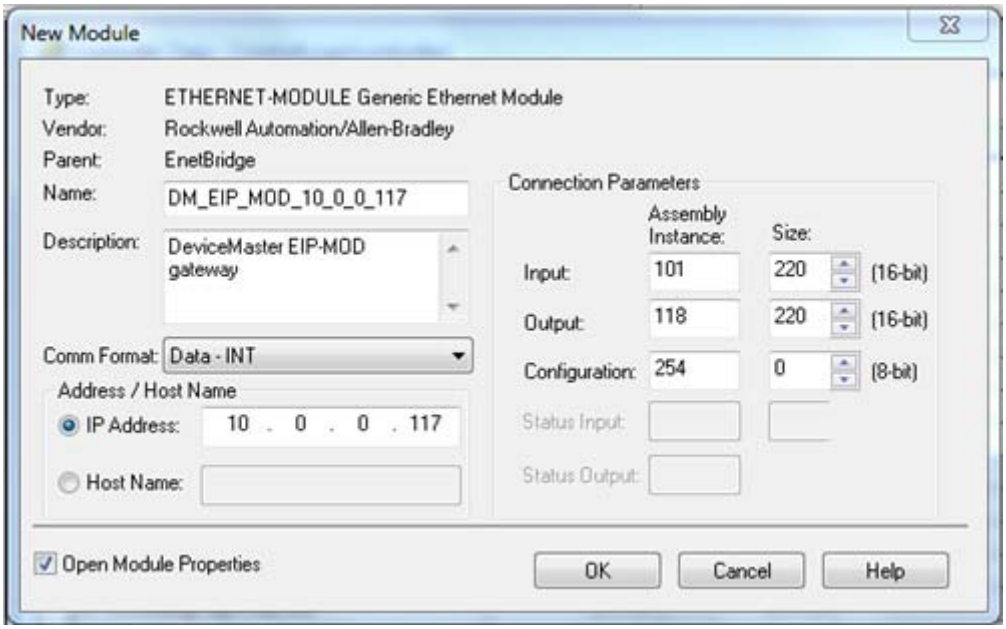
In RsLogix, each DeviceMaster EIP-MOD gateway must be set up as a Generic Ethernet Module to interface to the PLC via a Class 1 connection. The class 1 interface, as shown above, is displayed on the *EtherNet/IP Class1 Interface* page.

1. First, right click the Ethernet module on RSLogix5000 and select **New Module…**.
2. Click **Communications**.

3. Scroll down and select **Generic Ethernet Module**.

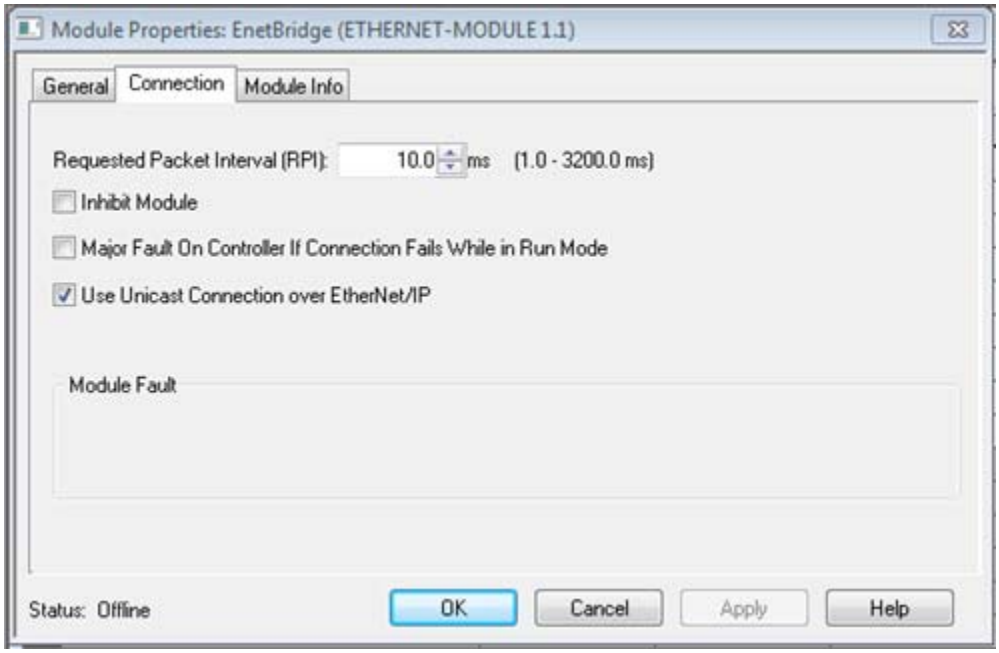


4. Click **OK**. The following pane appears.



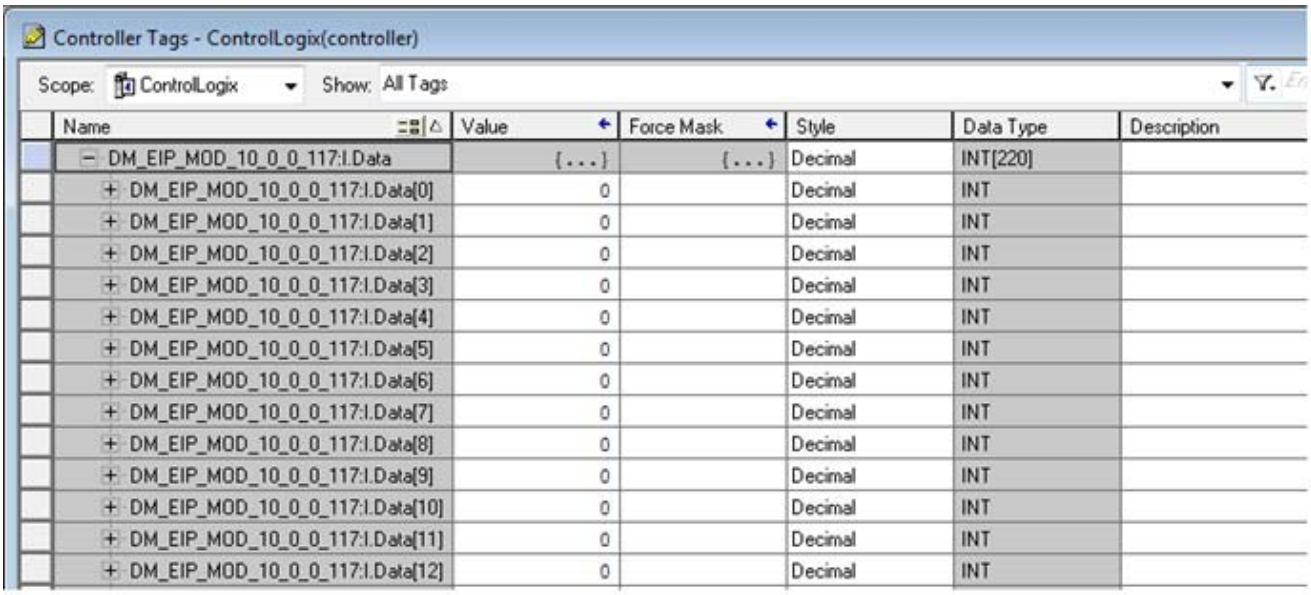
a. Enter a Name.
b. Select the **Comm Format** as **Data-INT**.
c. Enter the IP Address of the gateway.
d. Using the data displayed on the *Class 1 Overview* page, enter the Connection Parameters.

5. Click **OK**. The following pane will appear.



   a. The fastest allowable RPI is 10 ms.
   b. Both Unicast, (point-to-point), and Multicast, (one-to-many), connections are supported.
6. Click **OK**. The module will be added.
7. View the corresponding Input and Output data tags created when the gateway module was added. Once the PLC has connected to the DeviceMaster EIP-MOD gateway, these tags will communicate directly to the configured Shared Memory blocks.

   Input tag:



Name	Value	Force Mask	Style	Data Type	Description
DM_EIP_MOD_10_0_0_117:I.Data	{...}	{...}	Decimal	INT[220]	
DM_EIP_MOD_10_0_0_117:I.Data[0]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[1]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[2]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[3]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[4]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[5]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[6]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[7]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[8]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[9]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[10]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[11]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:I.Data[12]	0		Decimal	INT	

Output tag:

Name	Value	Force Mask	Style	Data Type	Description
DM_EIP_MOD_10_0_0_117:O	{...}	{...}		AB:ETHERNET_...	
DM_EIP_MOD_10_0_0_117:O.Data	{...}	{...}	Decimal	INT[220]	
DM_EIP_MOD_10_0_0_117:O.Data[0]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:O.Data[1]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:O.Data[2]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:O.Data[3]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:O.Data[4]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:O.Data[5]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:O.Data[6]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:O.Data[7]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:O.Data[8]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:O.Data[9]	0		Decimal	INT	
DM_EIP_MOD_10_0_0_117:O.Data[...	0		Decimal	INT	

Chapter 12. Troubleshooting and Technical Support

This section contains troubleshooting information for your DeviceMaster EIP-MOD. You should review the following subsections before calling Technical Support because they will request that you perform many of the procedures or verifications before they will be able to help you diagnose a problem.

If you cannot diagnose the problem, you can contact *12.4. Technical Support* on Page 158.

12.1. Troubleshooting Checklist

The following checklist may help you diagnose your problem:

- Verify that you are using the correct types of cables on the correct connectors and that all cables are connected securely.
- Isolate the DeviceMaster EIP-MOD from the network by connecting the device directly to a NIC in a host system.

Model	Connected to	Ethernet Cable	Connector Name
EIP-MOD-2101 EIP-MOD-2201 EIP-MOD-2102	Ethernet hub or NIC	Standard	10/100
EIP-MOD-2302 EIP-MOD-2304	Ethernet hub or NIC	Standard	10/100 - E1/E2

- Verify that the Ethernet hub and any other network devices between the system and the DeviceMaster EIP-MOD are powered up and operating.
- Reset the power on the DeviceMaster EIP-MOD and watch the **Status** light activity.

DeviceMaster EIP-MOD Status LED Activity	
5 sec. off, 3 flashes, 5 sec. off, 3 flashes ...	Redboot™ checksum failure.
5 sec. off, 4 flashes, 5 sec. off, 4 flashes ...	SREC load failure.
5 quick flashes	The default application is starting up.
10 sec. on, .1 sec. off, 10 sec. on .1 sec. off ...	The default application is running.

- Verify that the network IP address, subnet mask, and gateway is correct and appropriate for the network. If IP addressing is being used, the system should be able to ping the DeviceMaster EIP-MOD.
- Verify that the IP address programmed into the DeviceMaster EIP-MOD matches the unique reserved IP configured address assigned by the system administrator.
- If using DHCP, the host system needs to provide the subnet mask and gateway.

- Reboot the system and the DeviceMaster EIP-MOD.
- If you have a spare DeviceMaster EIP-MOD, try replacing the device.

12.2. General Troubleshooting

This table illustrates some general troubleshooting tips.

Note: *Make sure that you have reviewed the Troubleshooting Checklist (Page 155).*

General Condition	Explanation/Action
Status LED flashing	Indicates that boot program has not downloaded to the unit. 1. Reboot the system. 2. Make sure that you have downloaded the most current firmware for *EtherNet/IP-Modbus*. ***Note:*** *If the Status LED is still flashing, contact Technical Support.*
Status LED not lit	Indicates that power has not been applied or there is a hardware failure. Contact Technical Support.
Cannot ping the device through Ethernet hub	Isolate the DeviceMaster EIP-MOD from the network. Connect the device directly to the NIC in the host system (see Page 155).
Cannot ping or connect to the DeviceMaster EIP-MOD	The default IP address is often not accessible due to the subnet masking from another network unless **192.168** is used in the network. In most cases, it will be necessary to program in an address that conforms to your network.
DeviceMaster EIP-MOD continuously reboots when connected to some Ethernet switches or routers	Invalid IP information may also cause the switch or router to check for a gateway address. Lack of a gateway address is a common cause.

12.3. Daisy-Chaining DeviceMaster EIP-MOD Units With Two Ethernet Ports

DeviceMaster EIP-MOD models with two Ethernet ports follow the IEEE specifications for standard Ethernet topologies.

When using the E1/E2 ports, the DeviceMaster EIP-MOD is classified as a switch. When using the UP port only, it is a simple end node device.

The maximum number of daisy-chained DeviceMaster EIP-MOD units, and the maximum distance between units is based on the Ethernet standards and will be determined by your own environment and the conformity of your network to these standards.

Pepperl+Fuchs Comtrol, Inc. has tested with seven DeviceMaster EIP-MOD units daisy-chained together using 10 foot CAT5 cables, but this is not the theoretical limit. You may experience a performance hit on the devices at the end of the chain, so it is recommended that you overload and test for performance in your environment. The OS and the application may also limit the total number of ports that may be installed.

Following are some quick guidelines and URLs of additional information. Please note that standards and URLs do change.

- Ethernet 10BASE-T Rules
  - The maximum number of repeater hops is four.
  - You can use Category 3 or 5 twisted-pair 10BASE-T cables.
  - The maximum length of each cable is 100m (328ft).

    Note: *Category 3 or 5 twisted pair cables look the same as telephone cables but they are not the same. The network will not work if telephone cables are used to connect the equipment.*
- Fast Ethernet 100BASE-TX rules
  - The maximum number of repeater hops is two (for a Class II hub). A Class II hub can be connected directly to one other Class II Fast Ethernet hub. A Class I hub cannot be connected directly to another Fast Ethernet hub.
  - You must use Category 5 twisted-pair 100BASE-TX cables.
  - The maximum length of each twisted-pair cable is 100m (328ft).
  - The total length of twisted-pair cabling (across directly connected hubs) must not exceed 205m (672ft).

    Note: *Category 5 twisted pair cables look the same as telephone cables but they are not the same. The network will not work if telephone cables are used to connect the equipment.*
- IEEE 802.3 specification: A network using repeaters between communicating stations (PCs) is subject to the “5-4-3” rule of repeater placement on the network:
  - Five segments connected on the network.
  - Four repeaters.
  - Three segments of the 5 segments can have stations connected. The other two segments must be inter-repeater link segments with no stations connected.

    See *http://www.optronics.gr/Tutorials/ethernet.htm* for more specific information.

    Additional information may be found at *http://compnetworking.about.com/cs/ethernet1/* or by searching the web.

12.4. Technical Support

It contains troubleshooting procedures that you should perform before contacting Technical Support since they will request that you perform, some or all of the procedures before they will be able to help you diagnose your problem. If you need technical support, use one of the following methods.

Pepperl+Fuchs Comtrol, Inc. Contact Information	
Downloads	http://downloads.comtrol.com/html/DM_EIP_MOD_main.htm
Web site	http://www.comtrol.com
Phone	763.957.6000 Monday - Friday 8AM - 5PM CST