ICDM-RX/MOD
Modbus Solution Examples

Table of Contents

1 Introduction

This document includes a number of Modbus connectivity examples which are intended to help the system integrator or plant engineer determine how to set up installations. These examples include information such as the recommended chassis and Modbus firmware for each configuration. Information on how to configure the functionality for each particular ICDM-RX/MOD Modbus application is left to the User Manuals.

Due to flexibility of the ICDM-RX/MOD Modbus firmware applications, it is not possible to show every possible Modbus solution. The Modbus Router and Modbus/TCP firmware applications provide the ability to create either very simple or very complex Modbus installations. For instance, you can combine more than one of these connectivity solutions onto one gateway or use several gateways to build a Modbus network. (A Modbus network typically involves multiple gateways providing communication between Modbus controller(s) and device(s) on a network.)

When setting up your Modbus installation, please keep the following in mind:

- When using either Modbus Router or Modbus/TCP firmware, all Modbus masters can communicate to all public slave devices. For example, you can have multiple serial and Modbus/TCP masters communicating to the same or different slave(s) at one time.
- When using Modbus/TCP firmware, both Modbus controllers and applications can communicate to a single raw/ASCII device at the same time.
- Modbus messages are automatically converted from one format to another. For instance, you can connect Modbus/RTU masters to Modbus/TCP or Modbus/ASCII slaves and the gateway automatically performs all conversions and message verification.
- Each serial port or Ethernet TCP/IP device interface is individually configurable. That allows:
    - For Modbus/TCP and Modbus Router firmware, multiple serial master or slave devices of either the same or different Modbus type can be attached to the same gateway.
    - For Modbus/TCP firmware, each raw/ASCII serial port or Ethernet TCP/IP device interface can operate in different receive and transmit modes. (that is, one can operate in master mode while others operate slave or master/slave mode.)
- An ICDM-RX/MOD gateway is required to run both the Modbus Router and Modbus/TCP firmware applications.

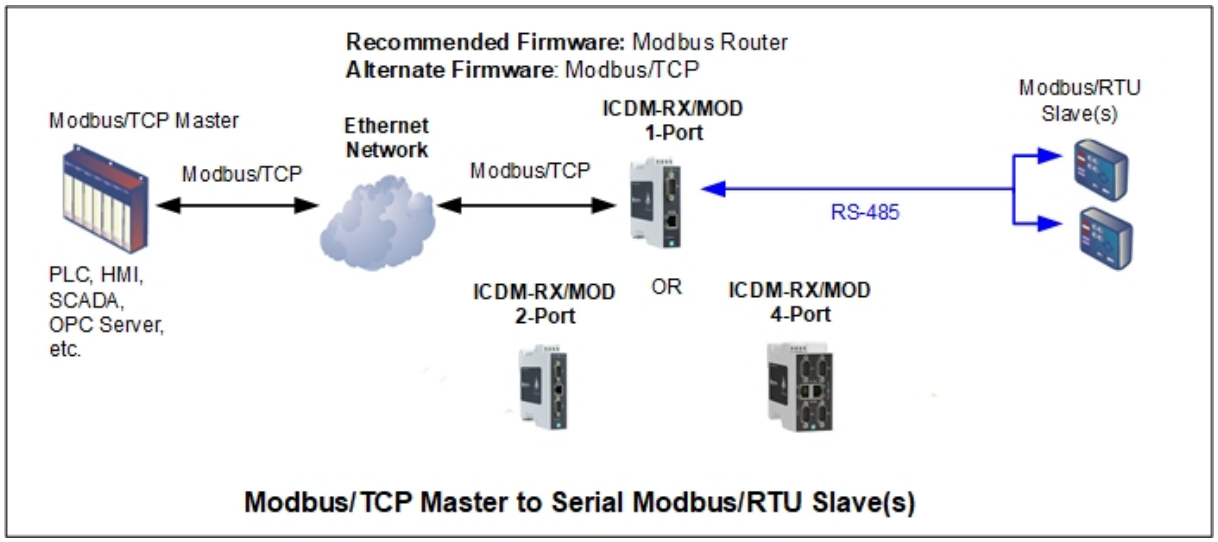
To find the Modbus Solution for your installation, please follow the solution outline.

2 Modbus Controller to Raw/ASCII and Modbus RTU/ASCII Device Environments

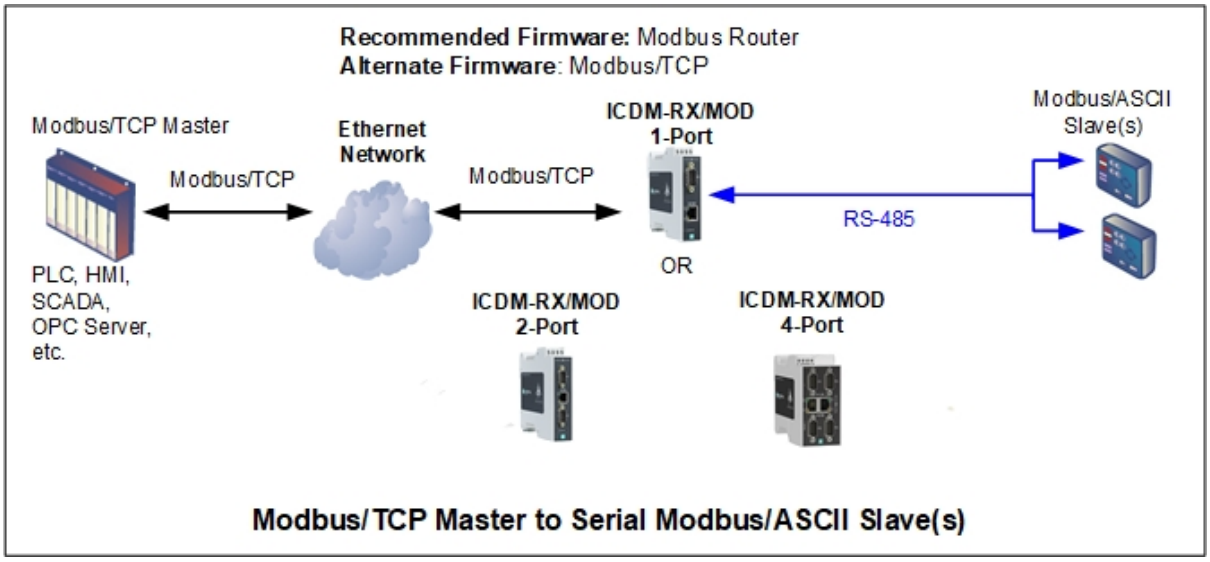
2.1 Modbus/TCP Master Controller

This typically includes such controllers as PLCs, OPC Servers, SCADA Systems, HMIs and applications that communicate as a master via Modbus/TCP.

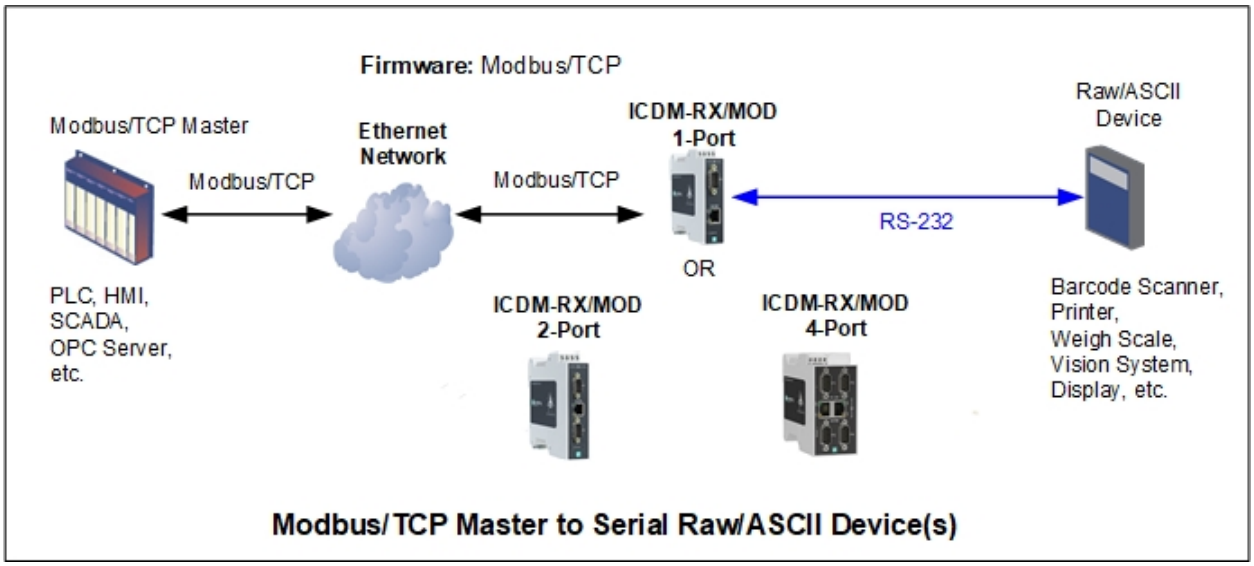
2.1.1 Serial Modbus/RTU Slave(s)



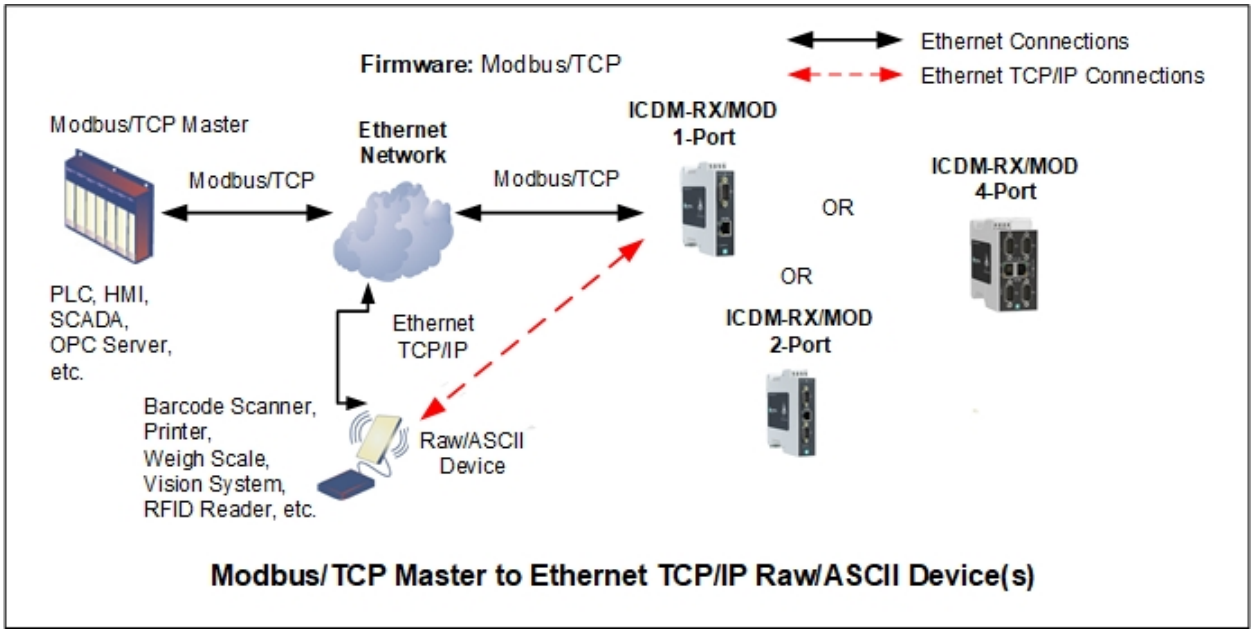
2.1.2 Serial Modbus/ASCII Slave(s)



2.1.3 Serial Raw/ASCII Device(s)



2.1.4 Ethernet Raw/ASCII Device(s)



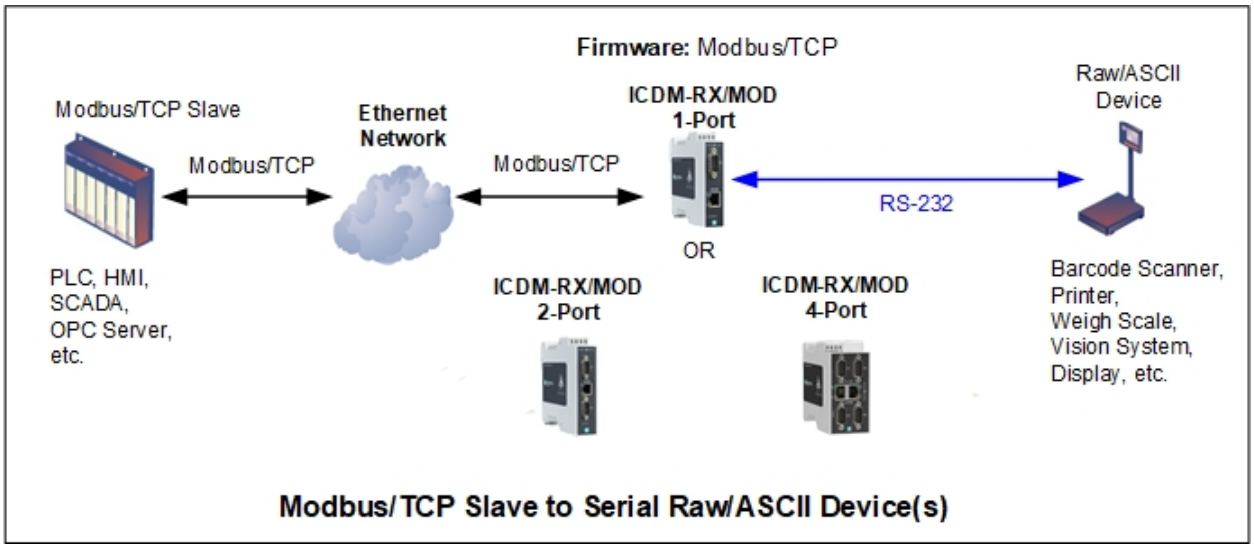
Modbus/TCP Master to Ethernet TCP/IP Raw/ASCII Device(s)

2.2 Modbus/TCP Slave Controller

This typically includes such controllers as PLCs, OPC Servers, SCADA Systems, HMIs and applications that communicate as a slave via Modbus/TCP.

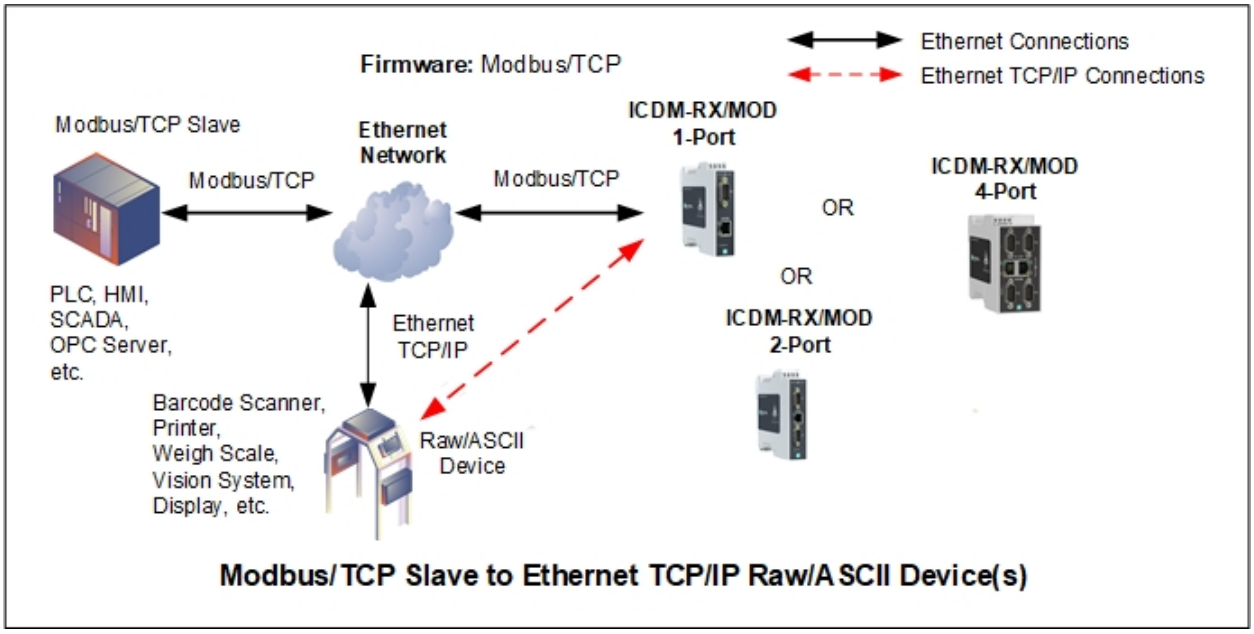
Due to the nature of master/slave connectivity, the functionality for Modbus/TCP slave controllers is limited to communicating to raw/ASCII devices.

2.2.1 Raw/ASCII Serial Device(s)



Modbus/TCP Slave to Serial Raw/ASCII Device(s)

2.2.2 Raw/ASCII Ethernet Device(s)

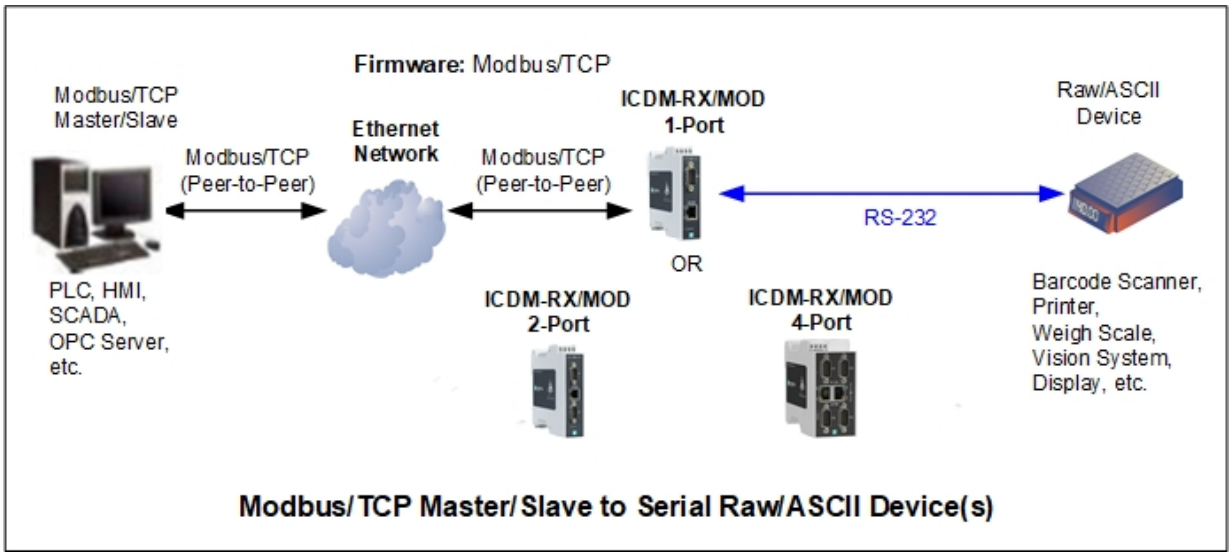


2.3 Modbus/TCP Master/Slave Controller

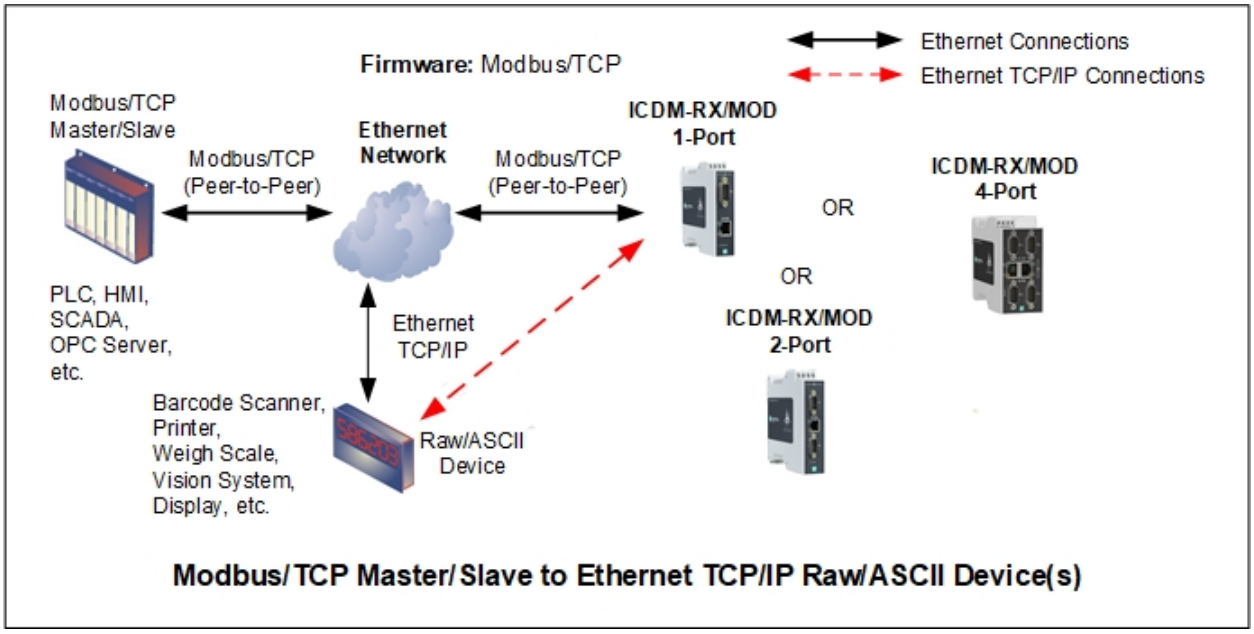
This typically includes full-featured controllers such as PLCs, OPC Servers, SCADA Systems, HMIs and applications that can communicate simultaneously as a master and a slave via Modbus/TCP. Utilizing Master/Slave functionality allows the controller and the ICDM-RX to operate as peers.

Due to the nature of master/slave connectivity, the functionality for Modbus/TCP master/slave controllers is limited to communicating to raw/ASCII devices.

2.3.1 Raw/ASCII Serial Device(s)



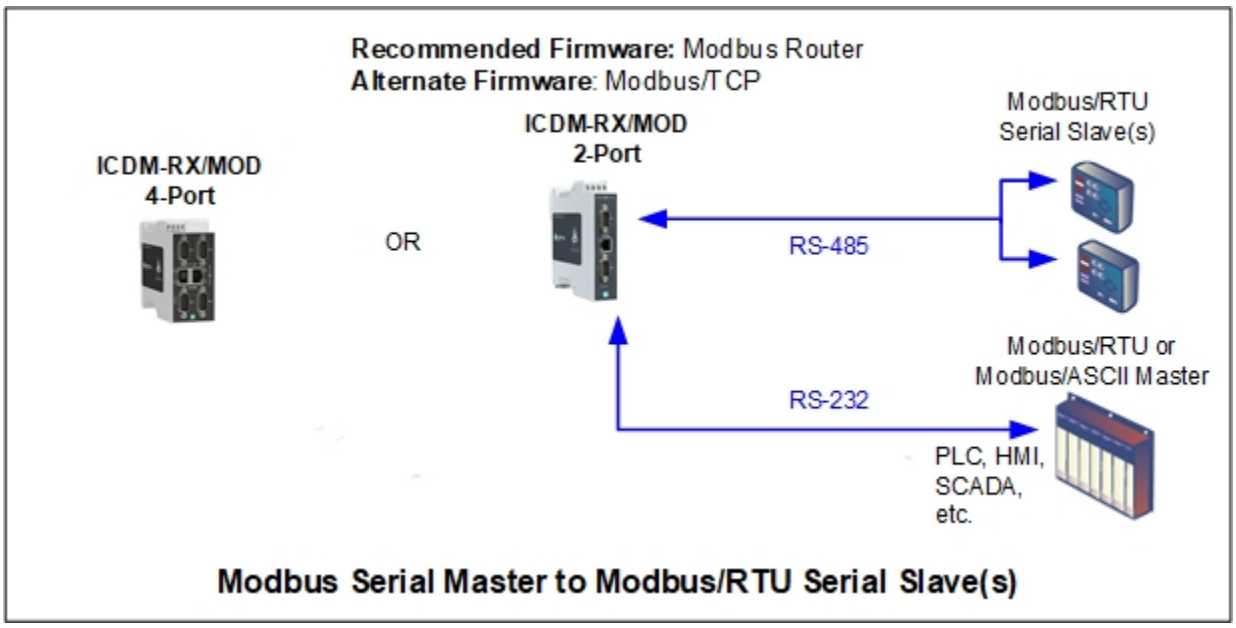
2.3.2 Raw/ASCII Ethernet Device(s)



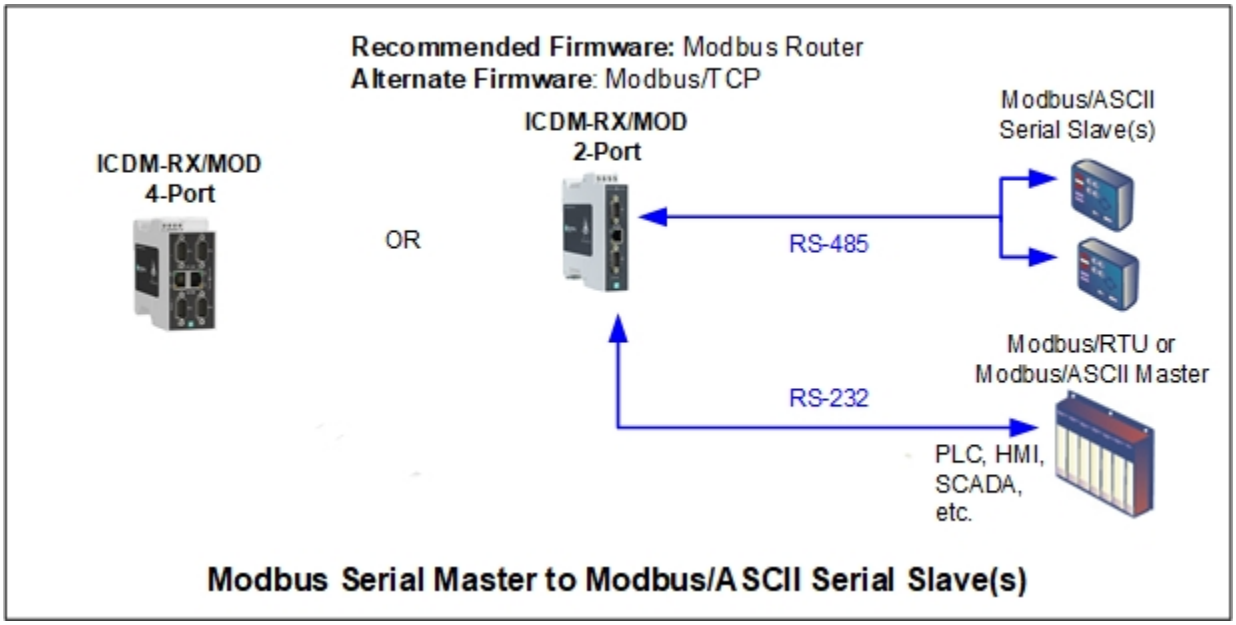
2.4 Modbus/RTU or Modbus/ASCII Serial Master Controller

This typically includes such controllers as PLCs, SCADA Systems, HMIs and applications that communicate as a master via Modbus/RTU or Modbus/ASCII over a serial or COM port.

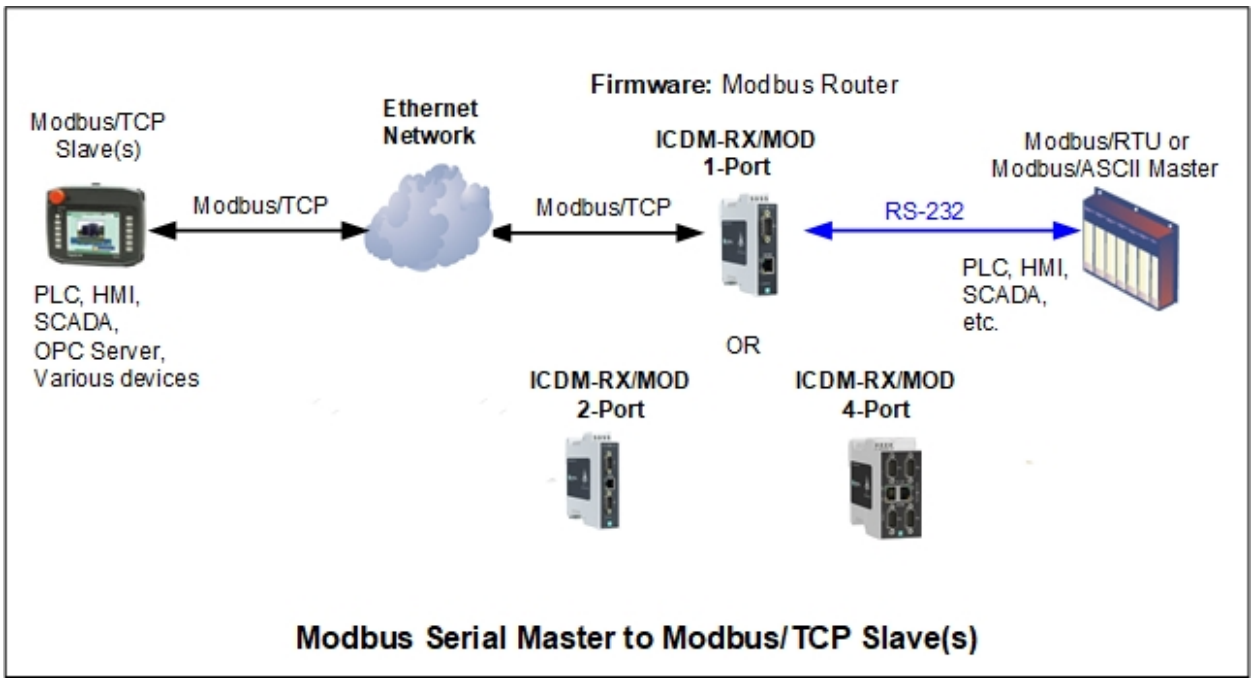
2.4.1 Serial Modbus/RTU Slave(s)



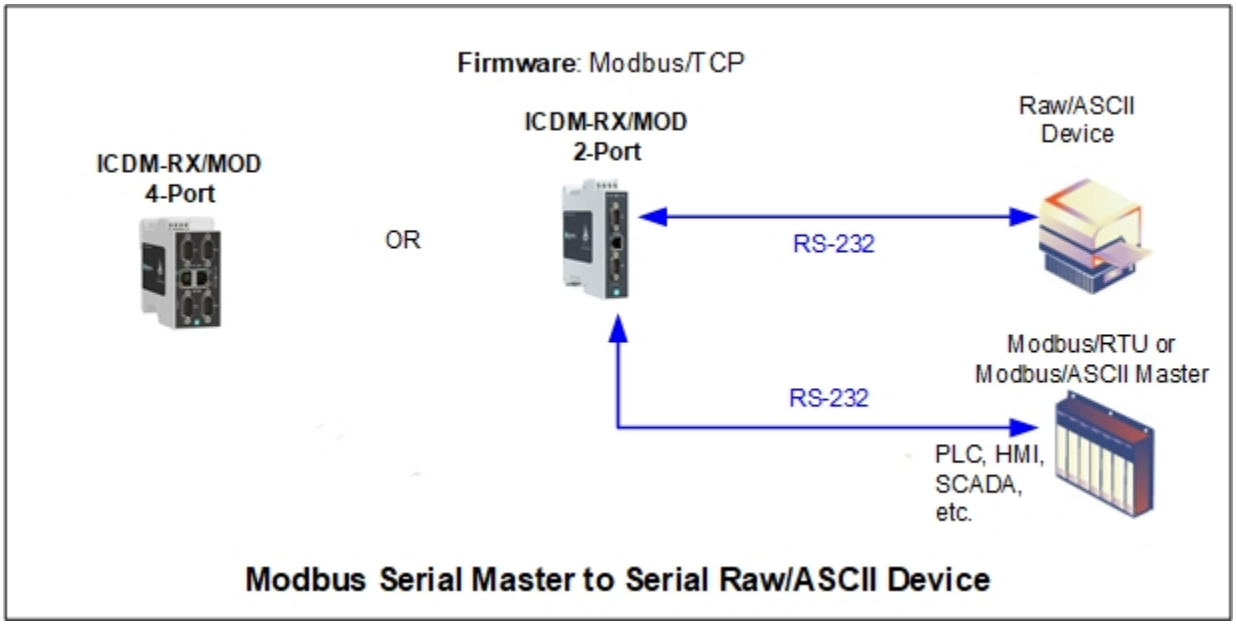
2.4.2 Serial Modbus/ASCII Slave(s)



2.4.3 Modbus/TCP Slaves

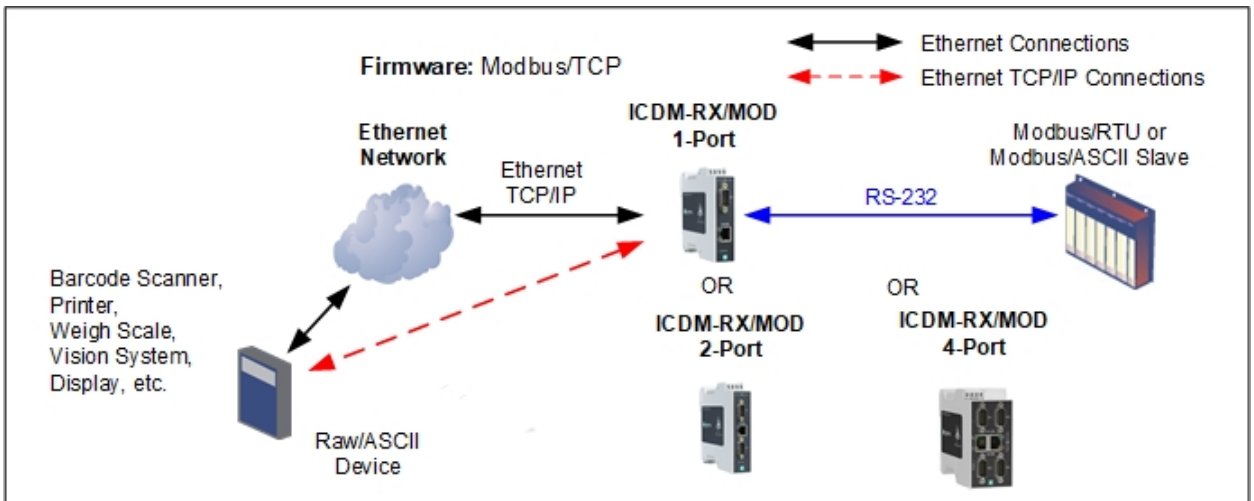


2.4.4 Serial Raw/ASCII Device(s)



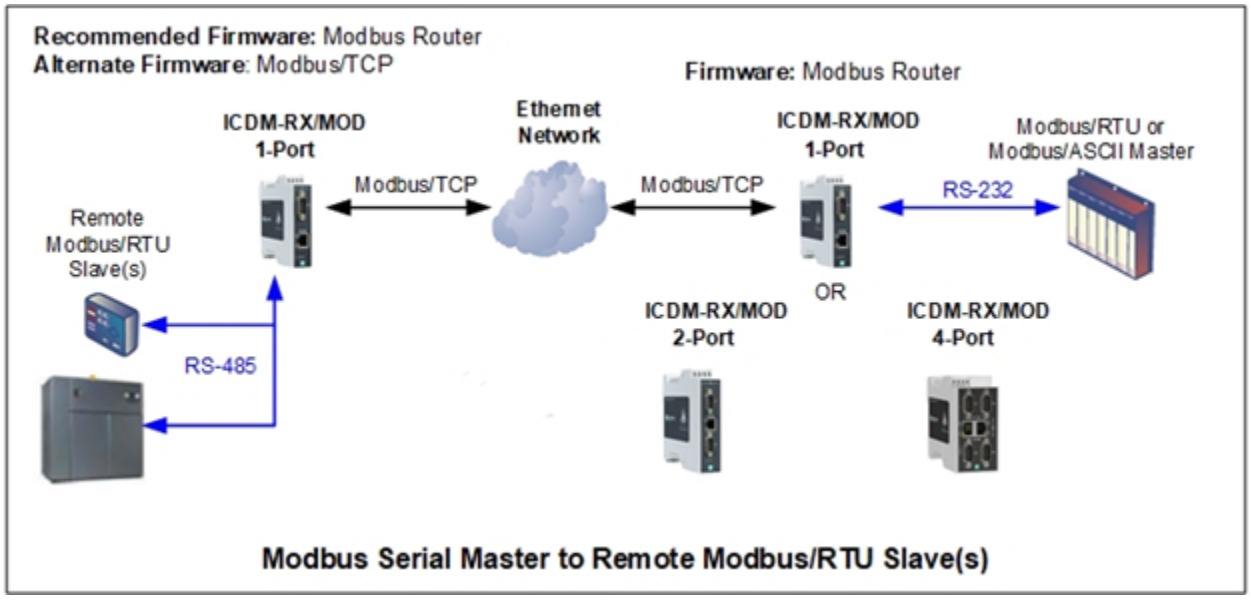
Modbus Serial Master to Serial Raw/ASCII Device

2.4.5 Raw/ASCII Ethernet Device(s)



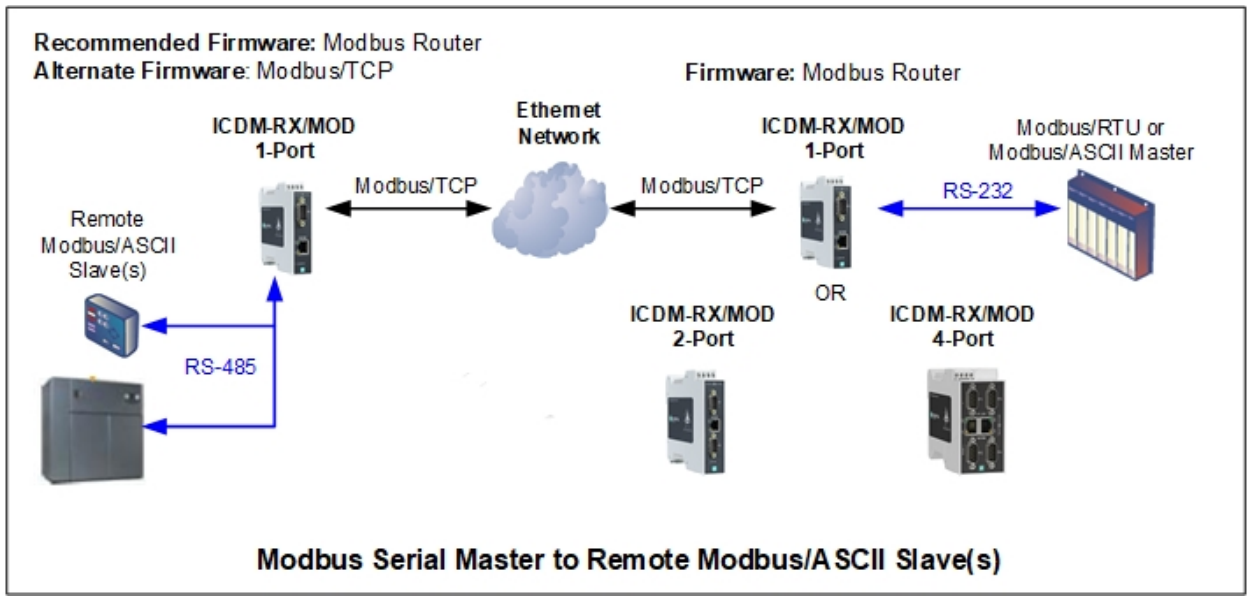
Modbus Serial Slave to Ethernet TCP/IP Raw/ASCII Device(s)

2.4.6 Remote Serial Modbus/RTU Slave(s)



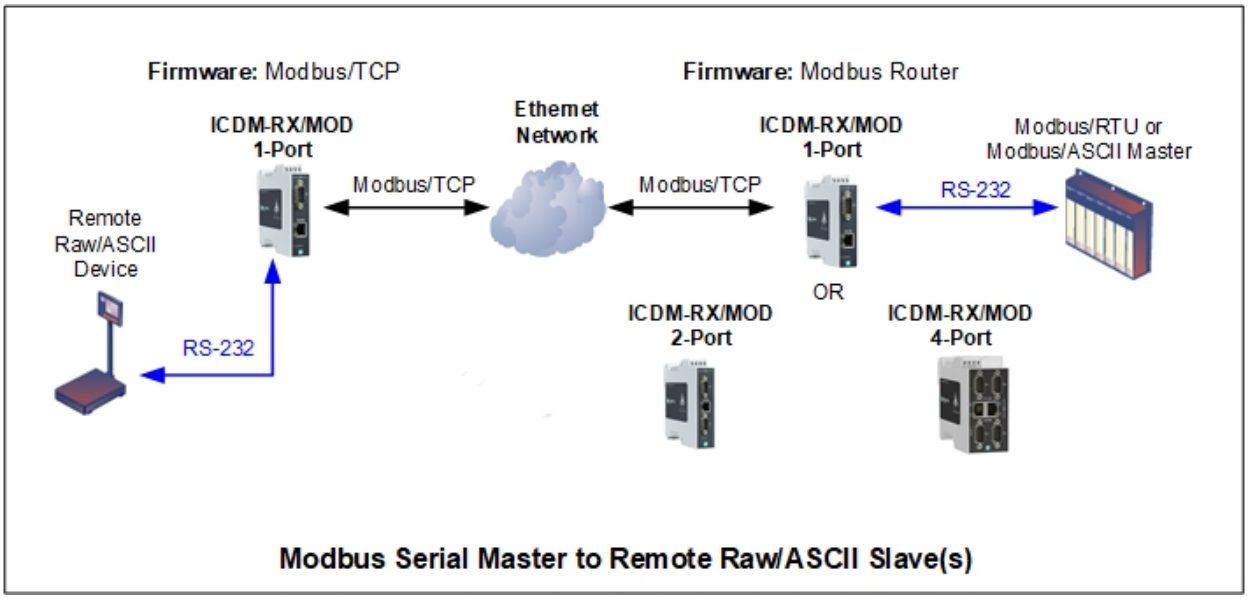
Modbus Serial Master to Remote Modbus/RTU Slave(s)

2.4.7 Remote Modbus/ASCII Slave(s)



Modbus Serial Master to Remote Modbus/ASCII Slave(s)

2.4.8 Remote Raw/ASCII Serial Device(s)

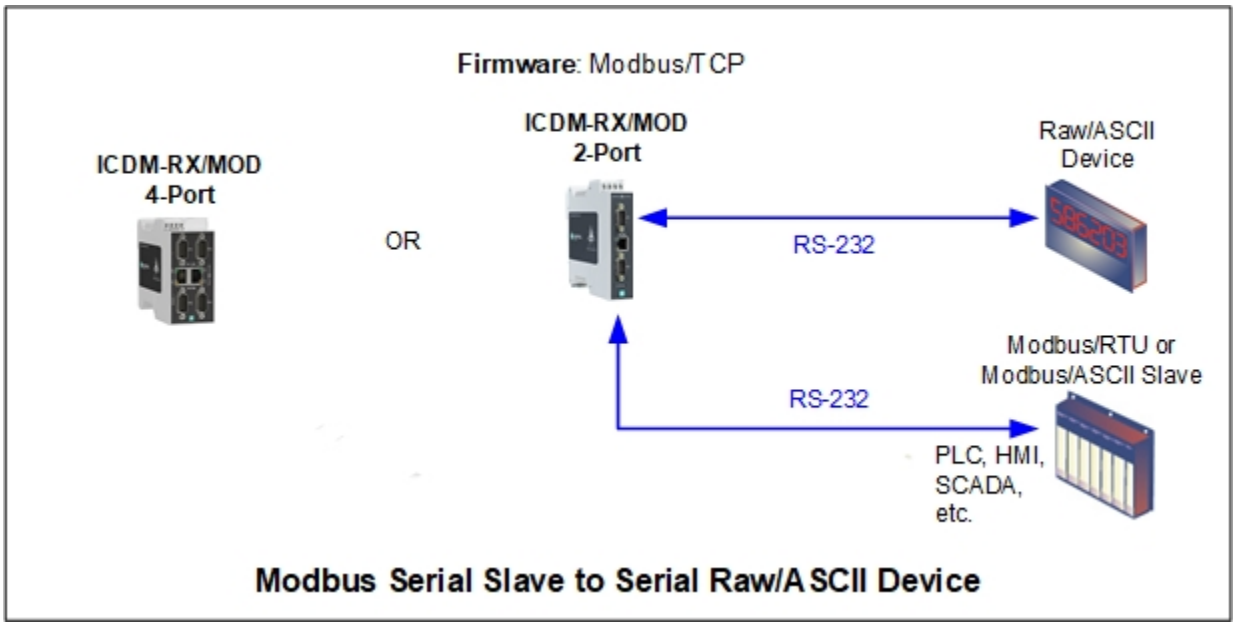


2.5 Modbus/RTU or Modbus/ASCII Serial Slave Controller

This typically includes such controllers as PLCs, SCADA Systems, HMIs and applications that communicate only as a slave via Modbus/RTU or Modbus/ASCII over a serial or COM port.

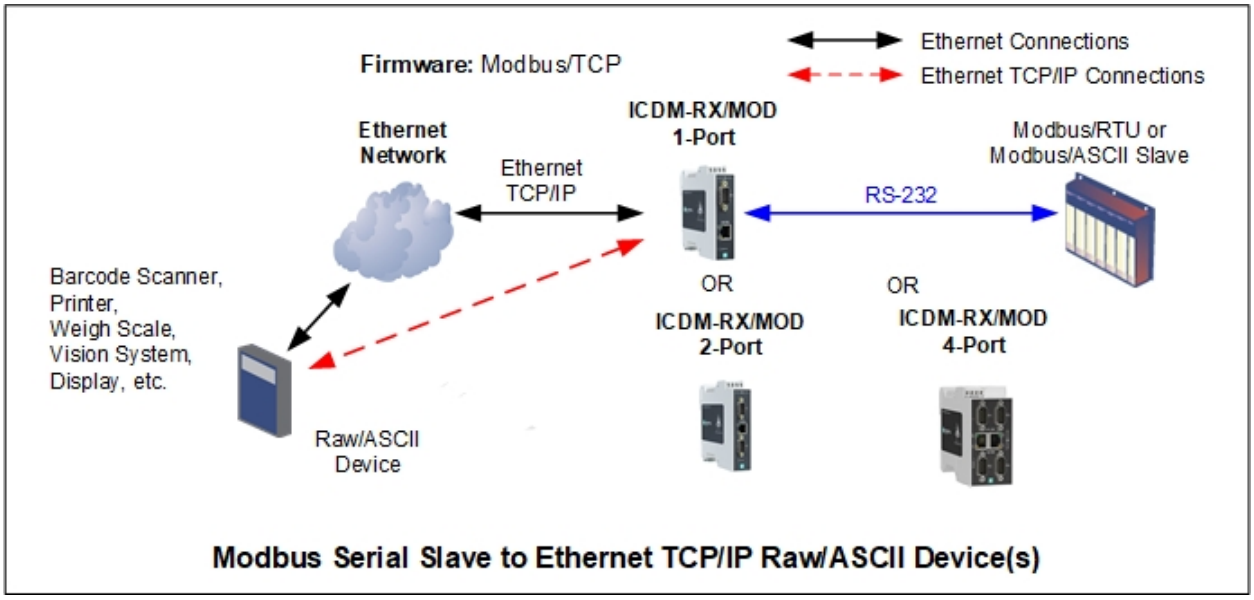
Due to the nature of master/slave connectivity, the functionality for Modbus serial slave controllers is limited to communicating to raw/ASCII devices.

2.5.1 Raw/ASCII Serial Device(s)



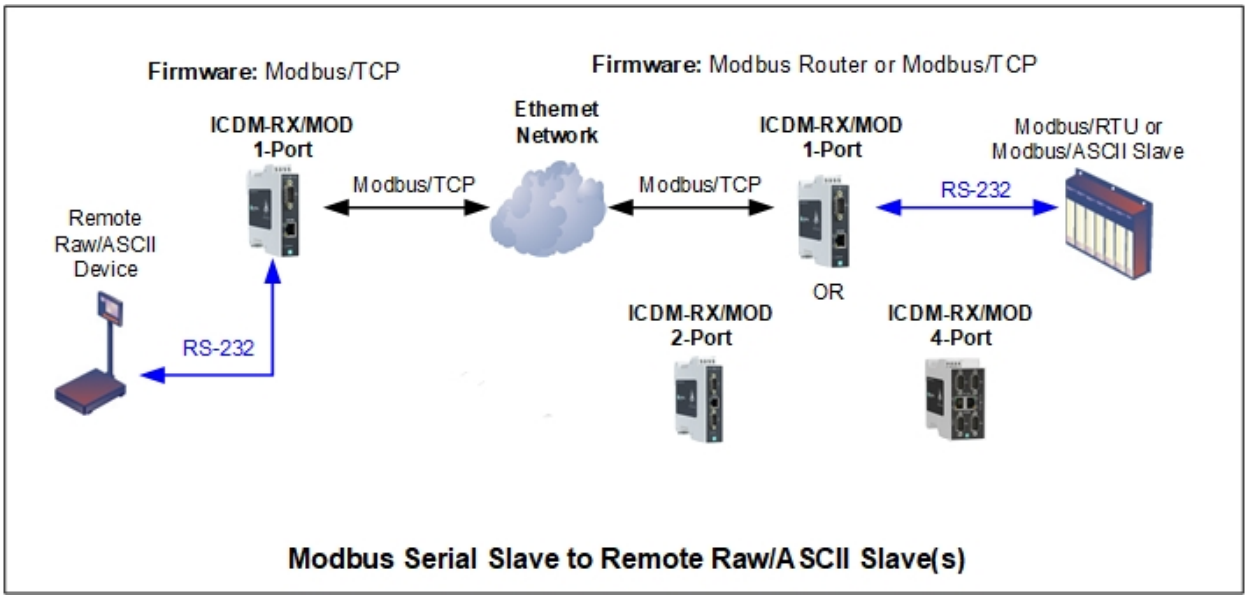
Modbus Serial Slave to Serial Raw/ASCII Device

2.5.2 Raw/ASCII Ethernet Device(s)



Modbus Serial Slave to Ethernet TCP/IP Raw/ASCII Device(s)

2.5.3 Remote Raw/ASCII Serial Device(s)

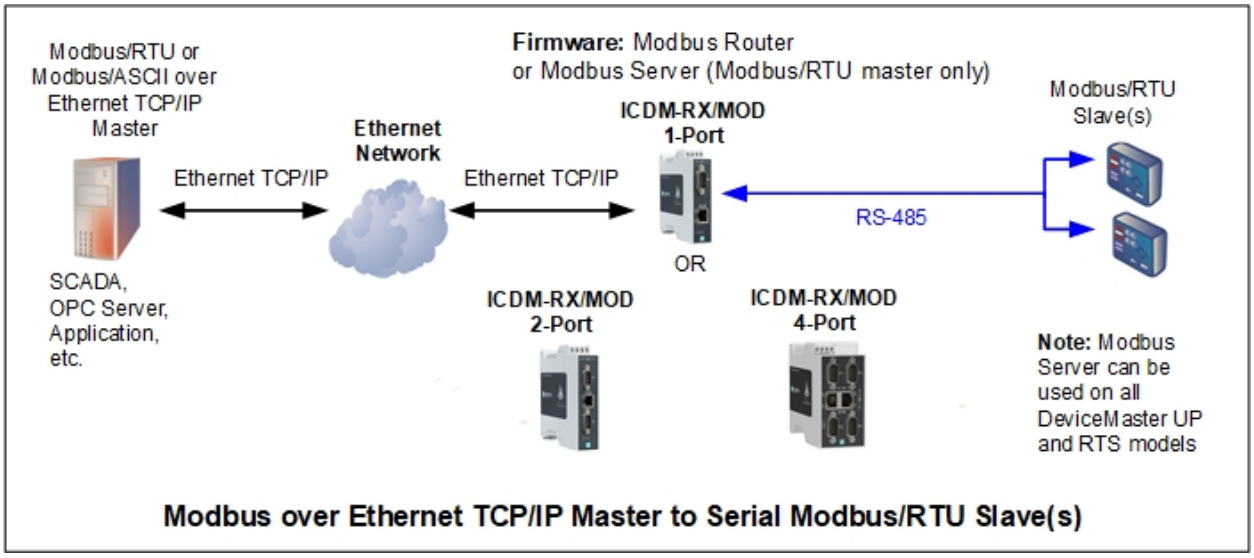


Modbus Serial Slave to Remote Raw/ASCII Slave(s)

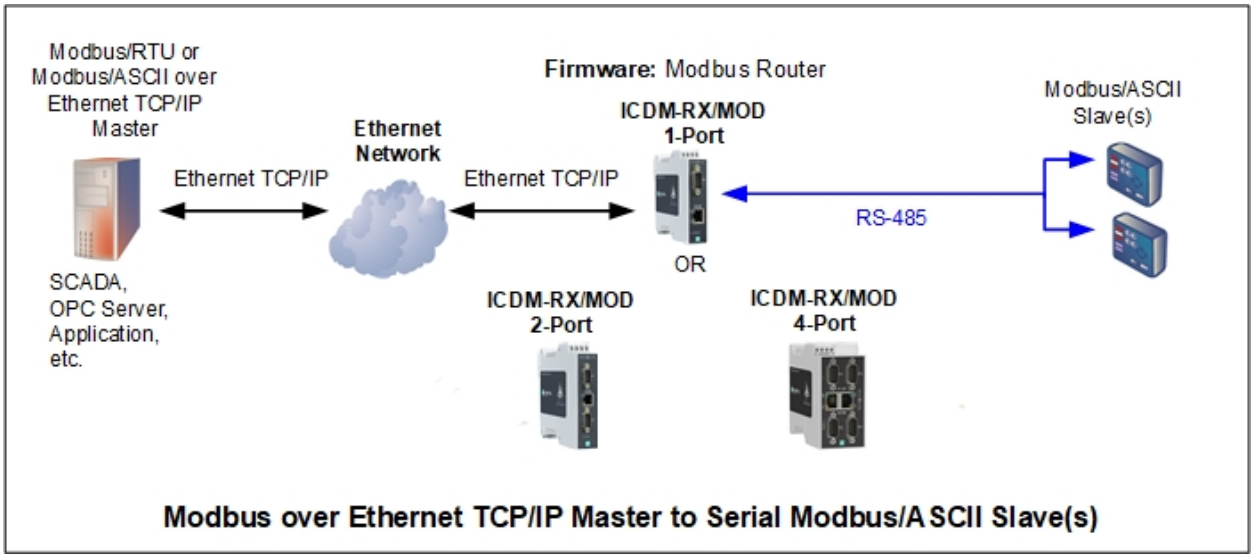
2.6 Modbus/RTU or Modbus/ASCII over Ethernet TCP/IP Master

This typically includes such controllers as some OPC Servers, SCADA Systems, and applications that communicate as a master via an Ethernet TCP/IP connection or, with the use of a serial port redirector, a COM port.

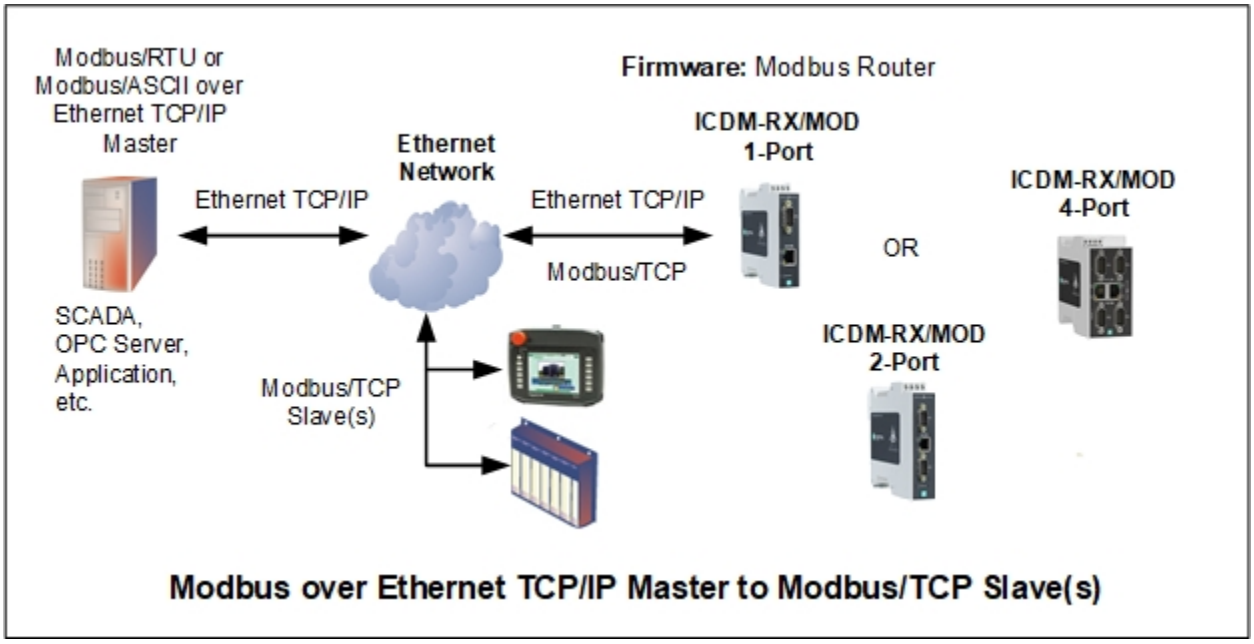
2.6.1 Modbus/RTU Serial Slave(s)



2.6.2 Modbus/ASCII Serial Slave(s)



2.6.3 Modbus/TCP Slaves

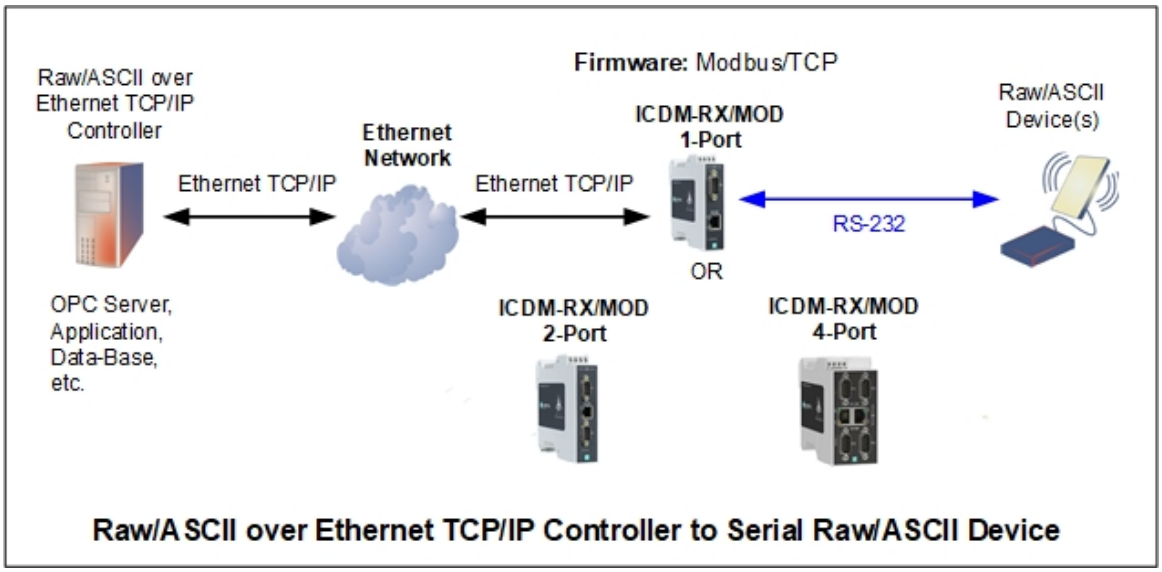


Modbus over Ethernet TCP/IP Master to Modbus/TCP Slave(s)

2.7 Application that Communicates via Raw/ASCII over Ethernet TCP/IP Connection(s)

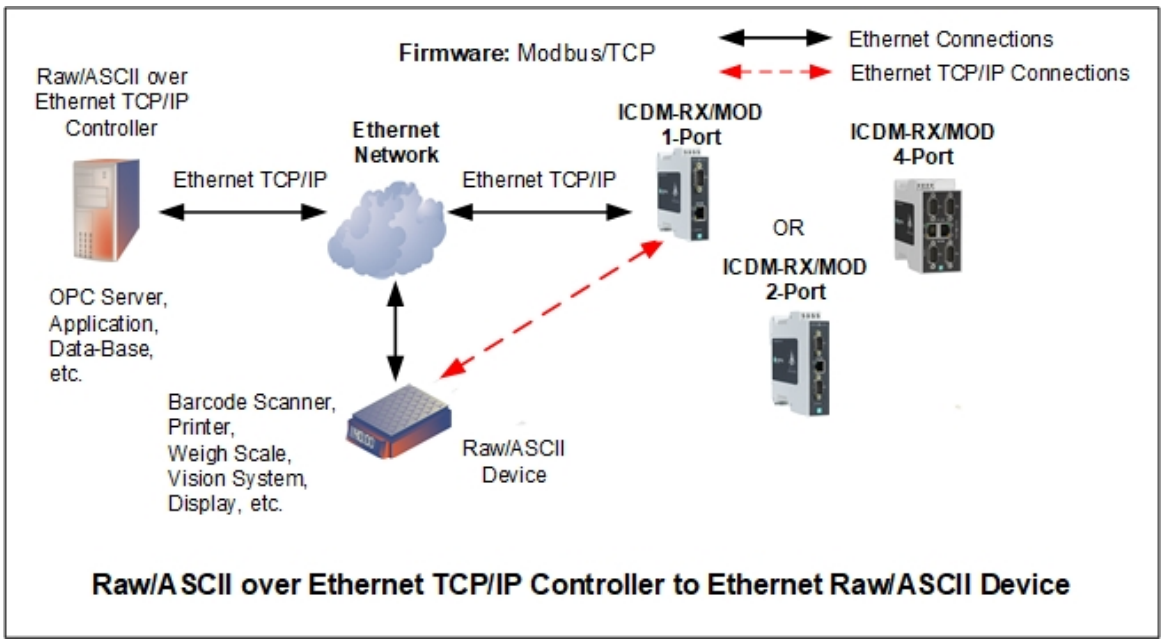
This typically includes control or data-base management applications that receive and/or transmit raw/ASCII data over an Ethernet TCP/IP connection or, with the use of a serial port redirector, a COM port.

2.7.1 Raw/ASCII Serial Device(s)



Raw/ASCII over Ethernet TCP/IP Controller to Serial Raw/ASCII Device

2.7.2 Raw/ASCII Ethernet Device(s)



3 Modbus Controller to Controller Communication

Today's Modbus installations are becoming increasingly complex. More and more installations are requiring the use of multiple Modbus controllers and the need to share information between the controllers is becoming increasingly important.

Sharing information between Modbus controllers can be relatively easy if one controller can communicate as a master (or client) and the other as a slave (or server). The master controller simply sends a message to the slave controller and the slave responds. But what do you do when both controllers can only be configured as a master?

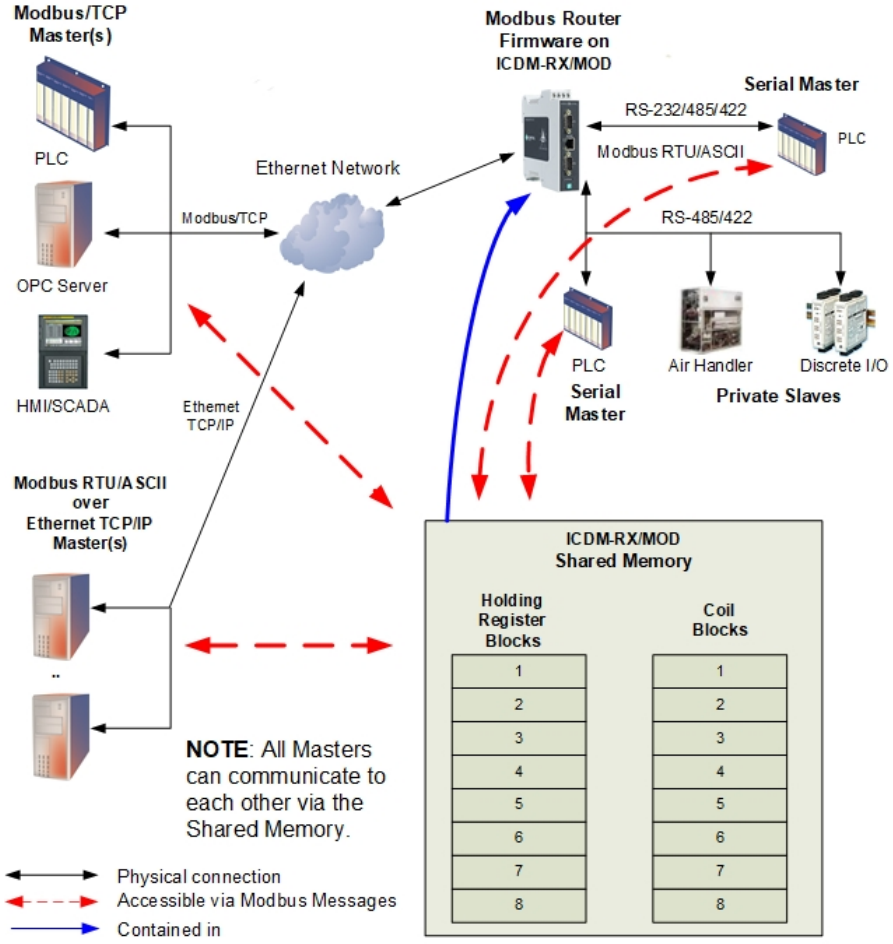
The ICDM-RX/MOD gateways provide two different methods of communication between Modbus masters:

- **Communication via Shared Memory** - The Modbus Router firmware contains a Shared Memory functionality that provides ***master-to-master*** controller communication.
- **Communication via Queued Messages** – The Modbus/TCP firmware provides a queued message method that enables both ***master-to-master*** and ***slave-to-slave*** controller communication.

3.1 Modbus Router Firmware – Master-to-Master via Shared Memory

The ICDM-RX/MOD gateways, running with the Modbus Router firmware, provide master-to-master connectivity using a configurable Shared Memory sub-system. **ICDM-RX/MOD gateways are loaded with Modbus Router at the factory**.

- The Shared Memory interface contains eight 200 Holding Register blocks and eight 160 Coil blocks.
- 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.
- Write access can be controlled to each Holding Register and Coil block. Each block can be configured to provide all masters write access or be restricted to a port-specific serial master, a Modbus/TCP master or an Ethernet TCP/IP master.
- 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.



Modbus Router Shared Memory Functionality

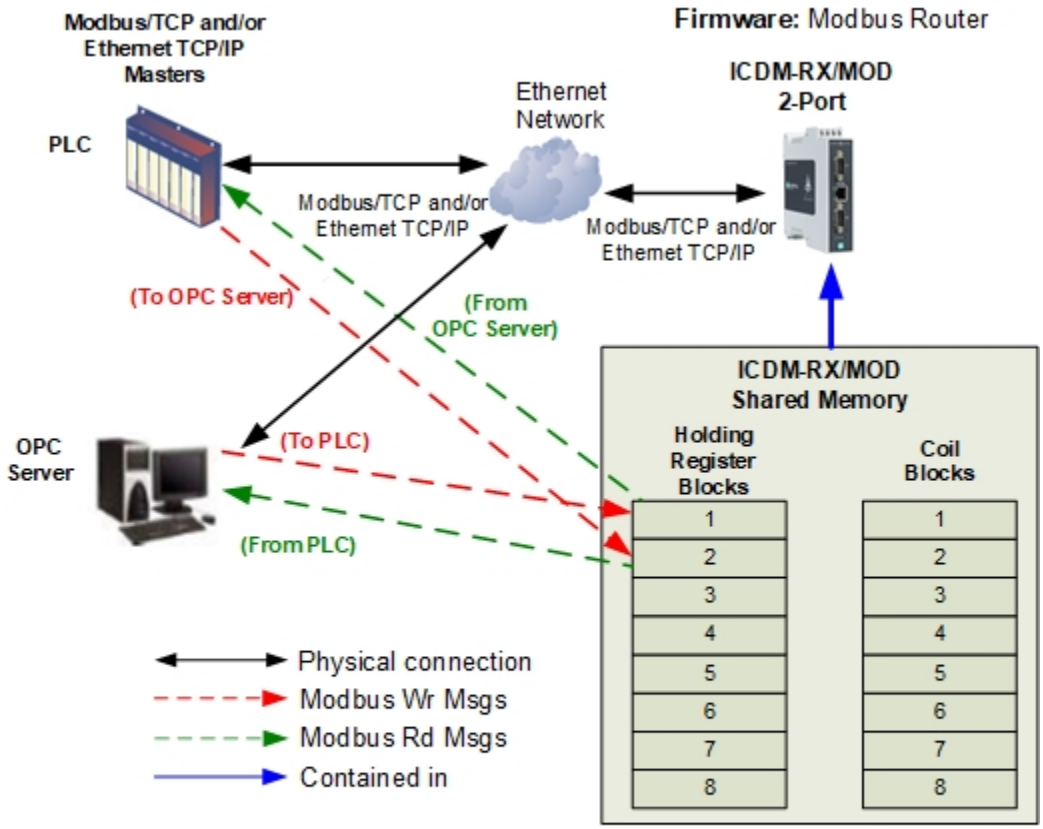
Multiple Modbus masters can communicate to each other through the Modbus Router Shared Memory. Possible communication options include:

- Two Modbus masters communicating directly to each other through two separate Shared Memory blocks.
- One Modbus master writing data to be read by one or more Modbus masters.
- Modbus/TCP, Modbus over Ethernet TCP/IP, and serial Modbus master communication.
- Communication from master(s) to a serial master with slave(s) on the same serial bus.

3.1.1 Using Shared Memory to Communicate Between Two Modbus Masters

3.1.1.1 Two Modbus/TCP and/or Ethernet TCP/IP Masters

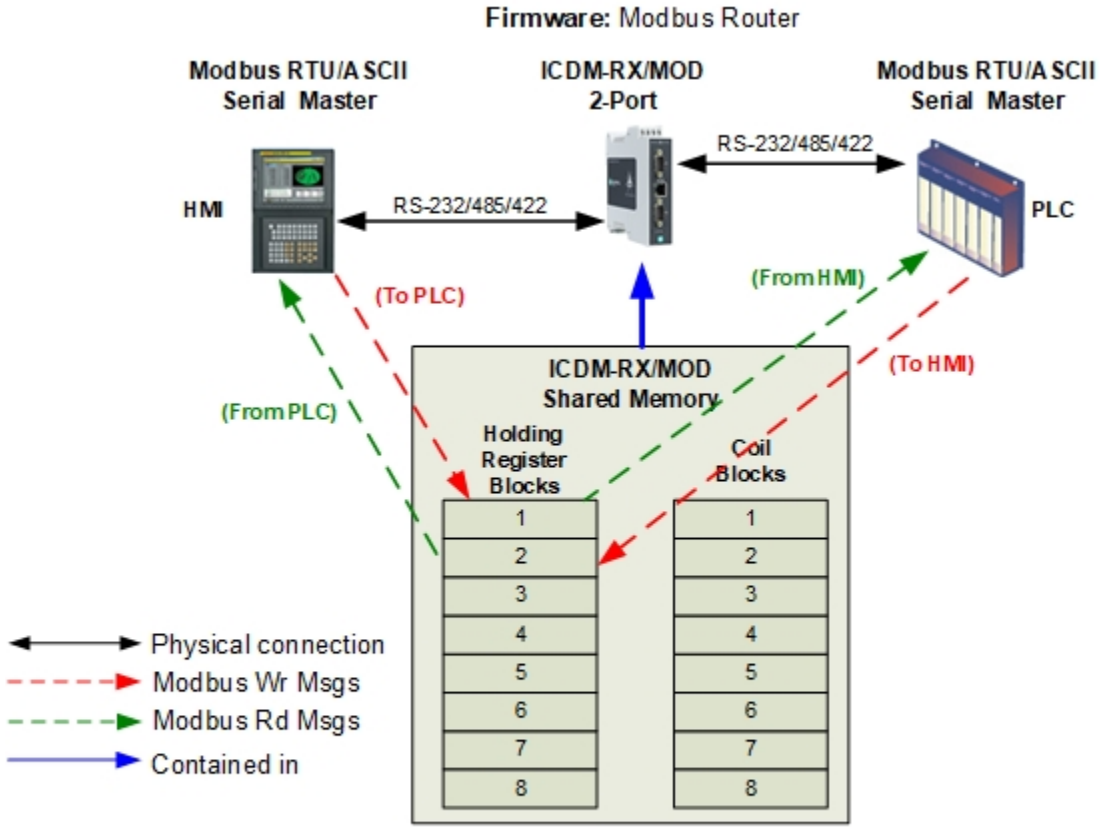
As shown in the following diagram, two Modbus/TCP and/or Ethernet TCP/IP masters can communicate to each other using the Shared Memory blocks.



Communicating Between Two Modbus/TCP and/or Ethernet TCP/IP Masters via Shared Memory

3.1.1.2 Two Serial Modbus Masters

As shown in the following diagram, two serial Modbus masters can communicate to each other using the Shared Memory blocks.

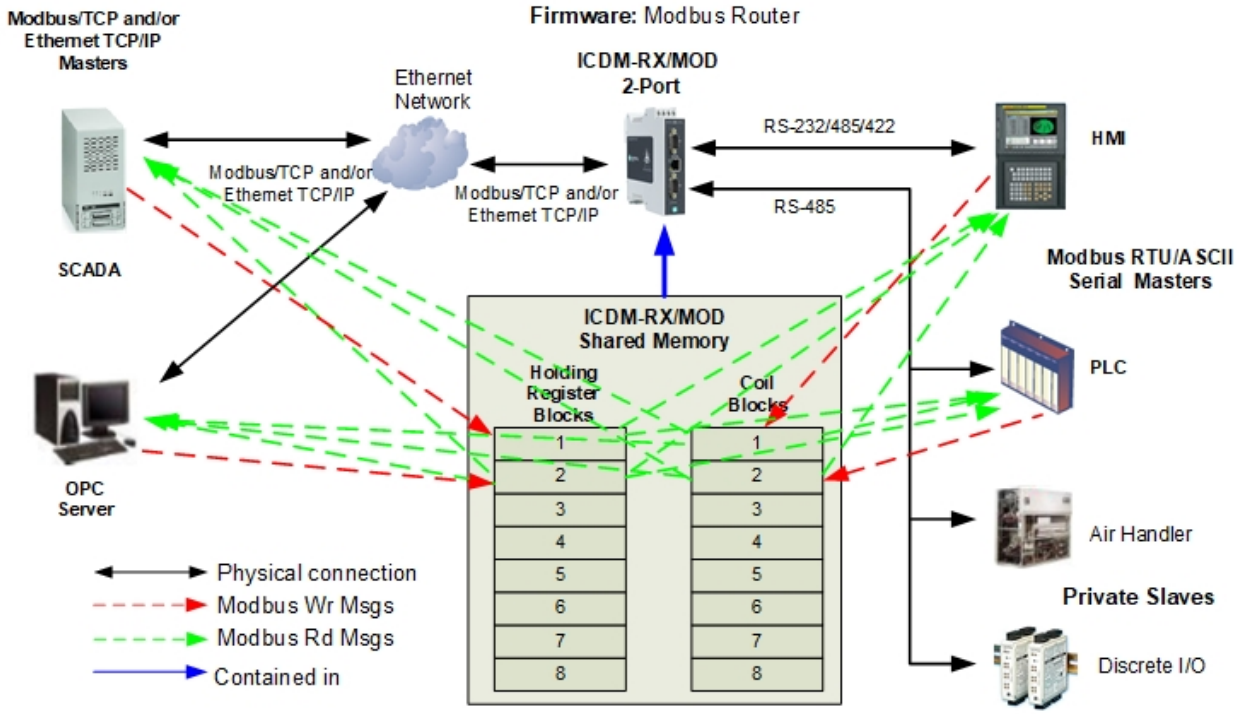


Communicating Between Two Serial Masters via Shared Memory

Note: Serial connections can also be made with a 4-Port ICDM-RX/MOD.

3.1.2 Modbus/TCP, Ethernet TCP/IP and Serial Modbus Masters

As shown in the following diagram, multiple Modbus/TCP, Ethernet TCP/IP and serial Modbus masters can communicate to each other using the Shared Memory Blocks. Please note that a serial bus connecting both a Modbus master and Modbus slaves can also be connected to an ICDM-RX/MOD.



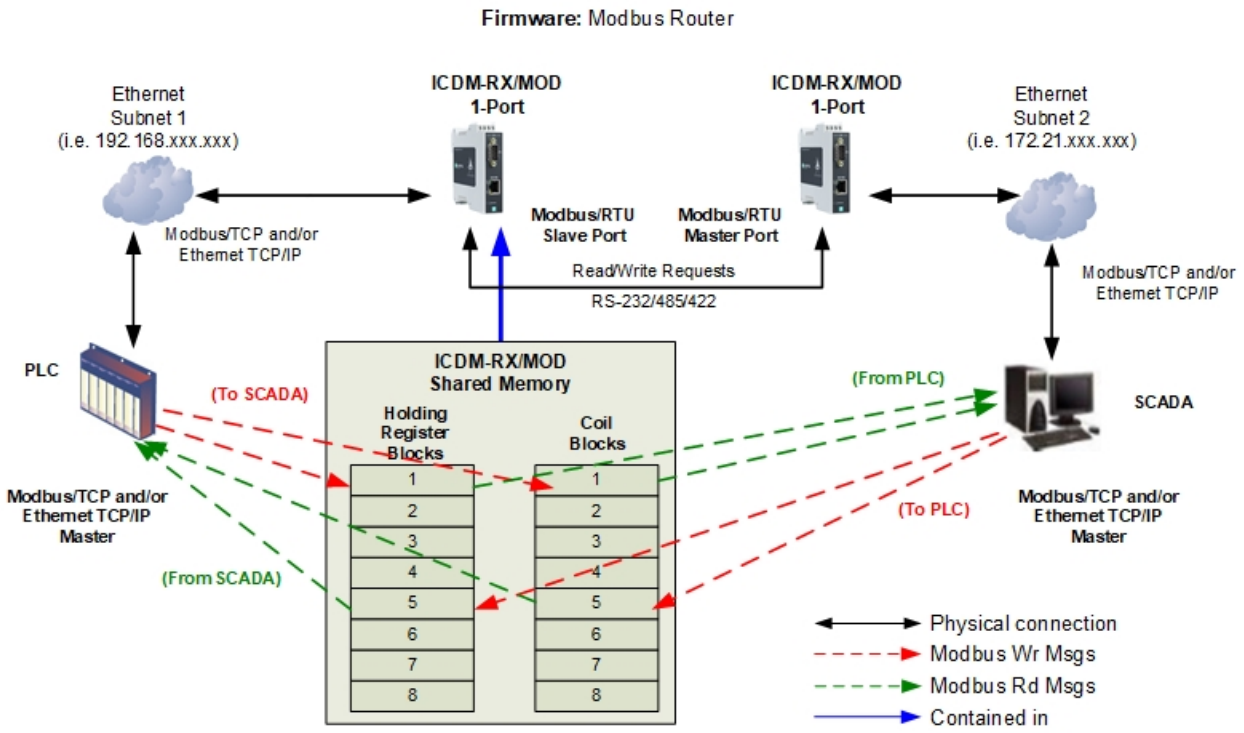
Communicating Between Modbus/TCP, Ethernet TCP/IP and Serial Masters via Shared Memory

Note: Serial connections can also be made with a 4-Port ICDM-RX/MOD.

3.1.3 Communicating Between Masters on Different Ethernet Subnets

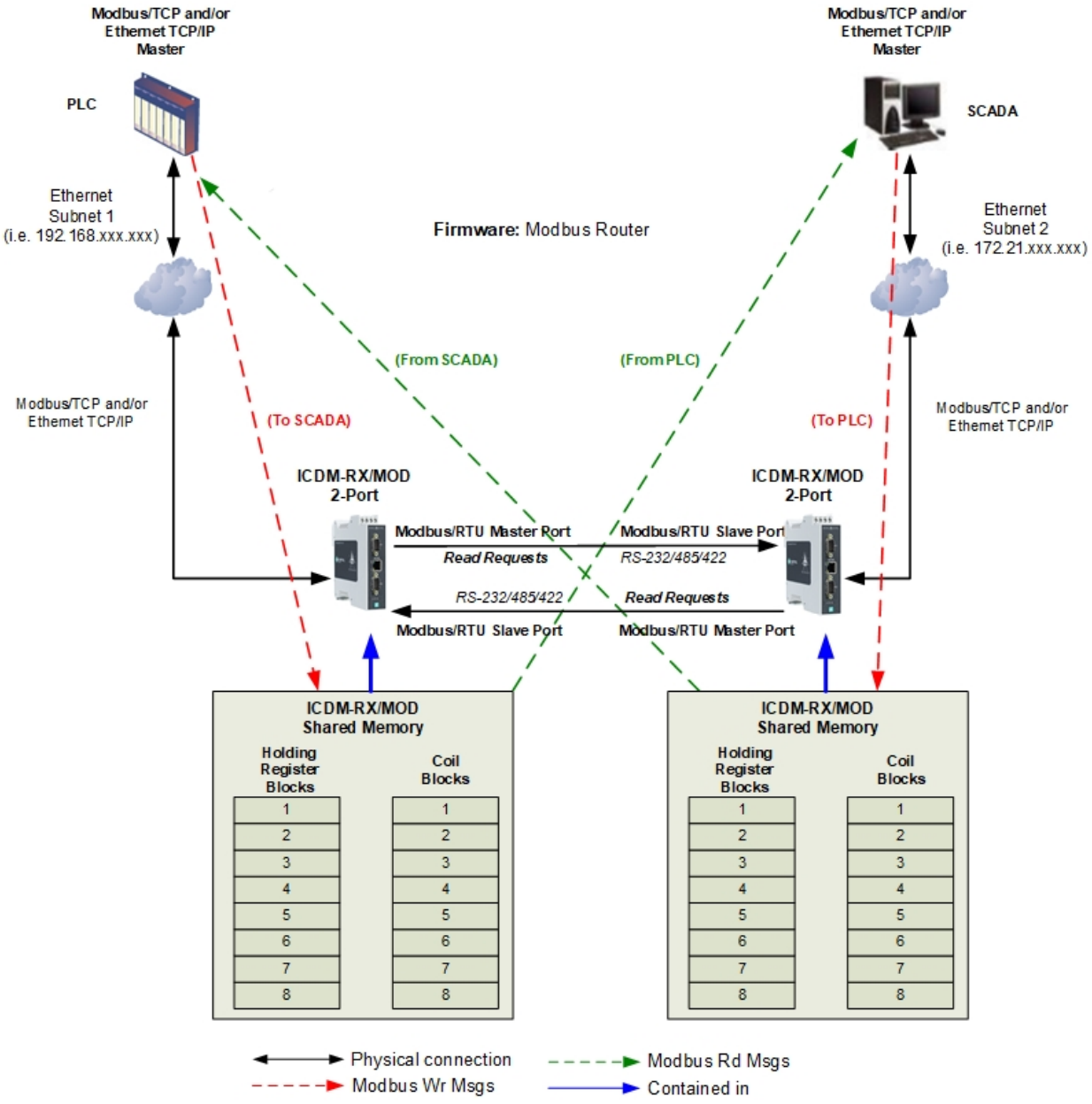
As shown in the following diagrams, Modbus/TCP and Ethernet TCP/IP masters on different Ethernet subnets can communicate to each other using the Shared Memory Blocks.

3.1.3.1 Using1 Port ICDM-RX/MOD Gateways



Communication Between Masters via Shared Memory on Different Ethernet Subnets with 1 Port Gateways

3.1.3.2 Using 2 Port ICDM-RX/MOD Gateways



Communication Between Masters via Shared Memory on Different Ethernet Subnets with 2 Port Gateways

3.2 Modbus/TCP Firmware – Master-to-Master and Slave-to-Slave via Queued Messages

The ICDM-RX/MOD gateways, running with the Modbus/TCP firmware, can provide both master-to-master and slave-to-slave controller connectivity. Bi-directional data paths can be created by connecting serial ports and/or internal TCP/IP sockets together. The end result is an asynchronous, queued holding register interface which can allow multiple Modbus controllers to communicate to each other.

You must upload Modbus/TCP firmware to the ICDM-RX/MOD as by default it is loaded with Modbus Router. See the Pepperl+Fuchs web site or contact our Technical Support team for the firmware file.

3.2.1 Modbus Master-to-Master Connectivity

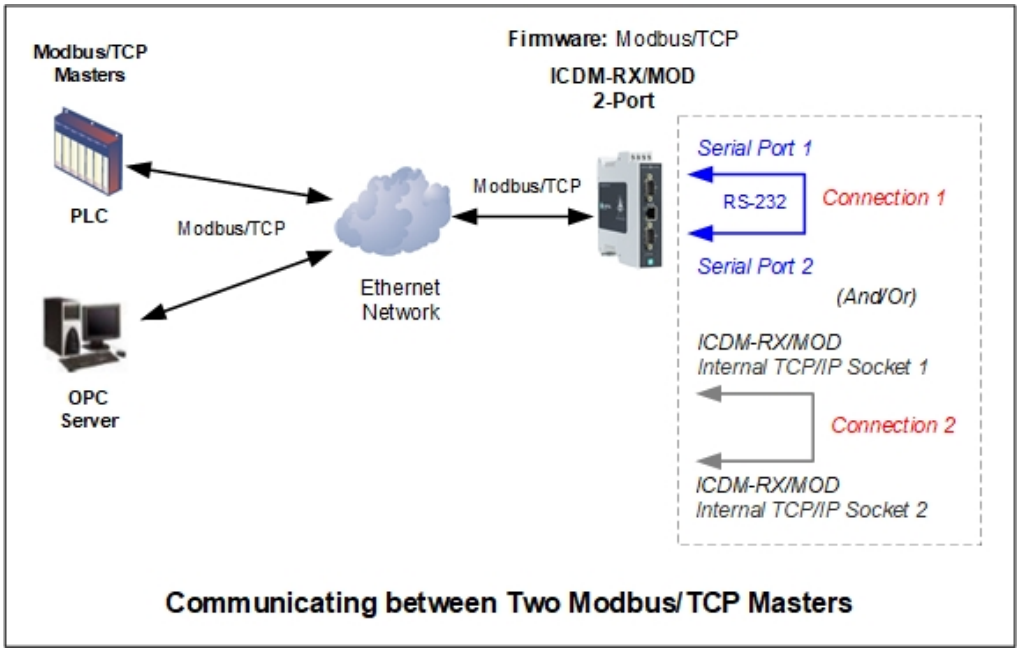
Multiple Modbus masters can communicate to each other through the ICDM-RX/MOD. Possible communication options include:

- Two Modbus masters communicating directly to each other.
- One Modbus master sending messages to be received by multiple Modbus masters.
- Both Modbus/TCP and serial Modbus master communication.

3.2.1.1 Communicating Between Two Modbus Masters

3.2.1.1.1 Two Modbus/TCP Masters

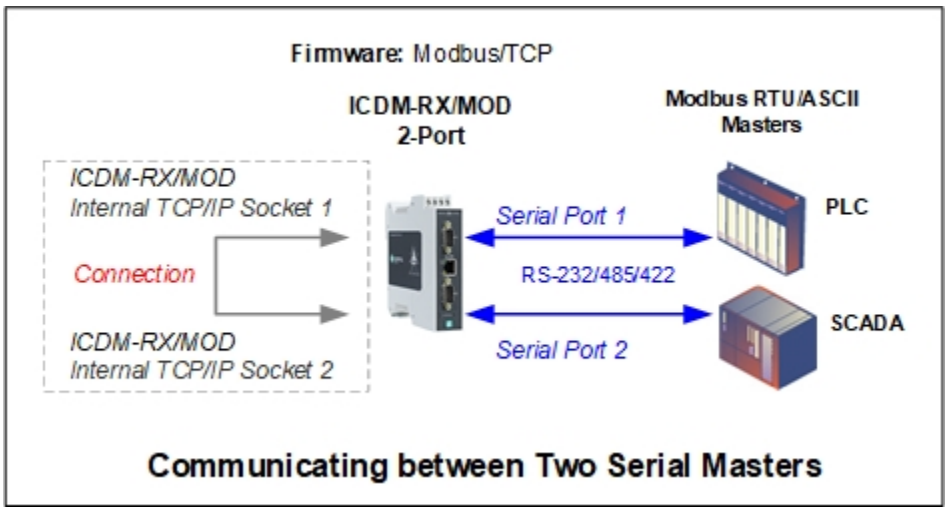
As shown in the following diagram, two Modbus/TCP masters can communicate to each other using either internal TCP/IP socket connections or connecting two serial ports.



Communicating between Two Modbus/TCP Masters

3.2.1.1.2 Two Serial Modbus Masters

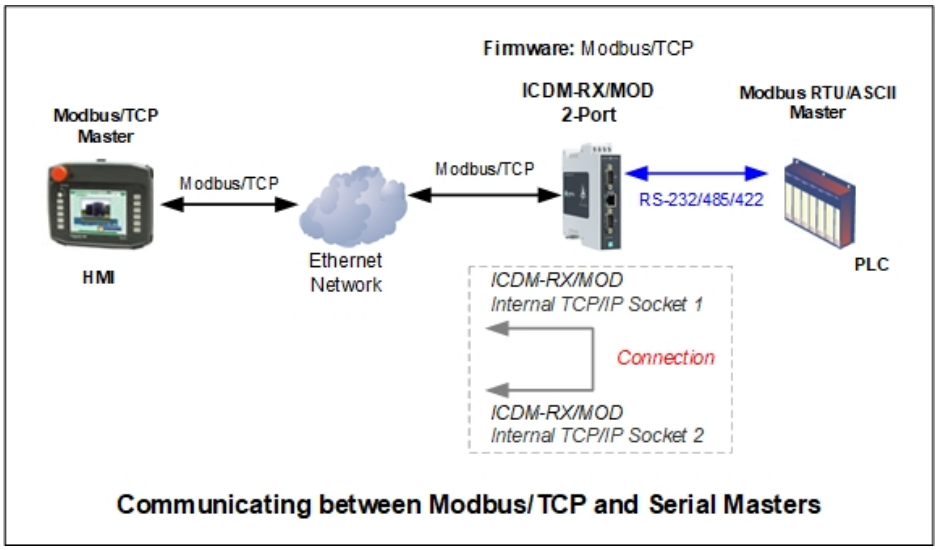
As shown in the following diagram, two serial Modbus masters can communicate to each other using an internal TCP/IP socket connection.



Note: Serial connections could be made with a 4-Port ICDM-RX/MOD.

3.2.1.1.3 Modbus/TCP and Serial Modbus Masters

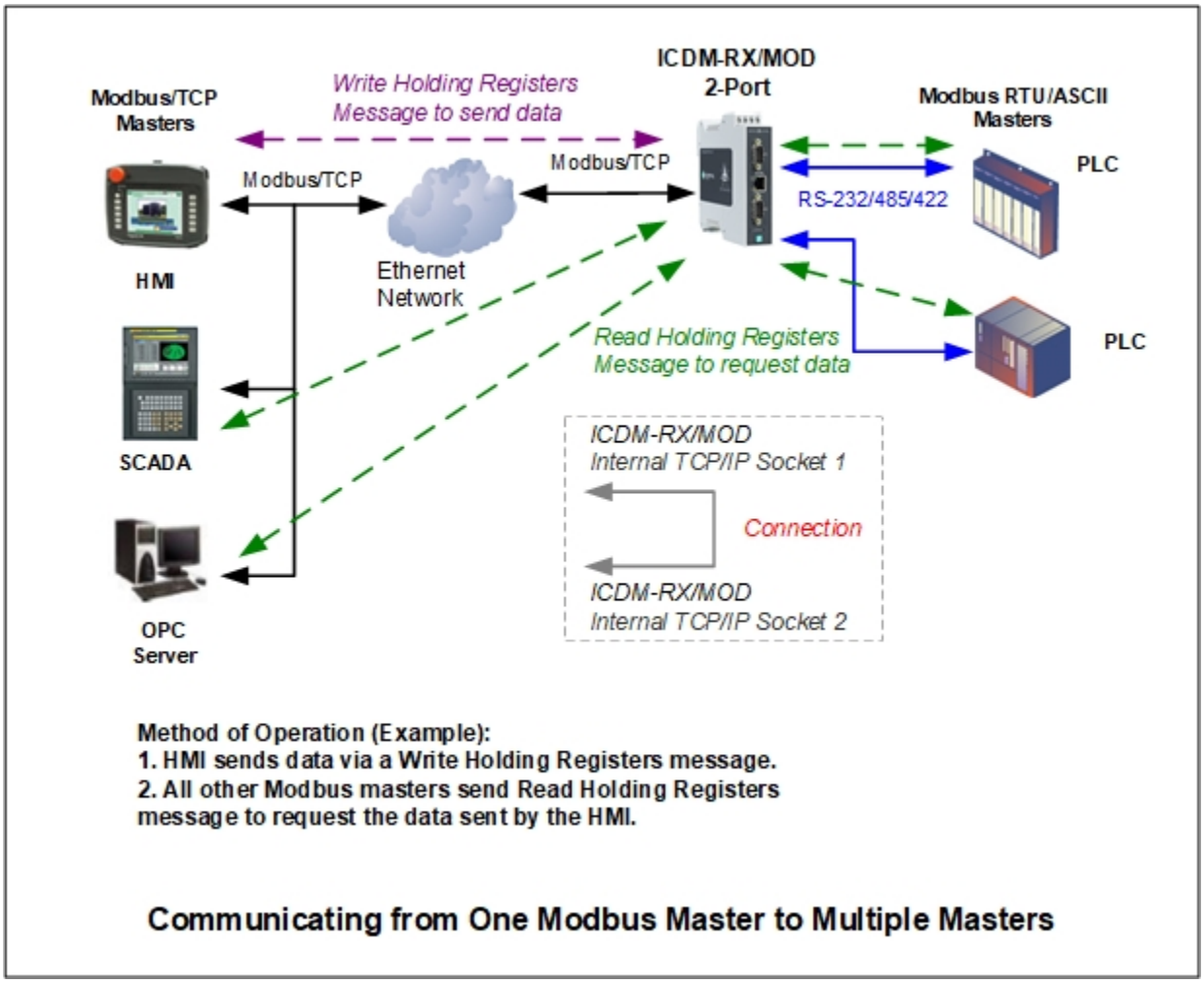
As shown in the following diagram, Modbus/TCP and serial Modbus masters can communicate to each other using an internal TCP/IP socket connection.



Note: Serial connections could be made with a 4-Port ICDM-RX/MOD.

3.2.1.2 One Modbus Master Communicating to Multiple Modbus Masters

As shown in the following diagram, one Modbus master can send messages to more than one other Modbus master. The following diagram displays an example of communication from one master to several other masters.



Communicating from One Modbus Master to Multiple Masters

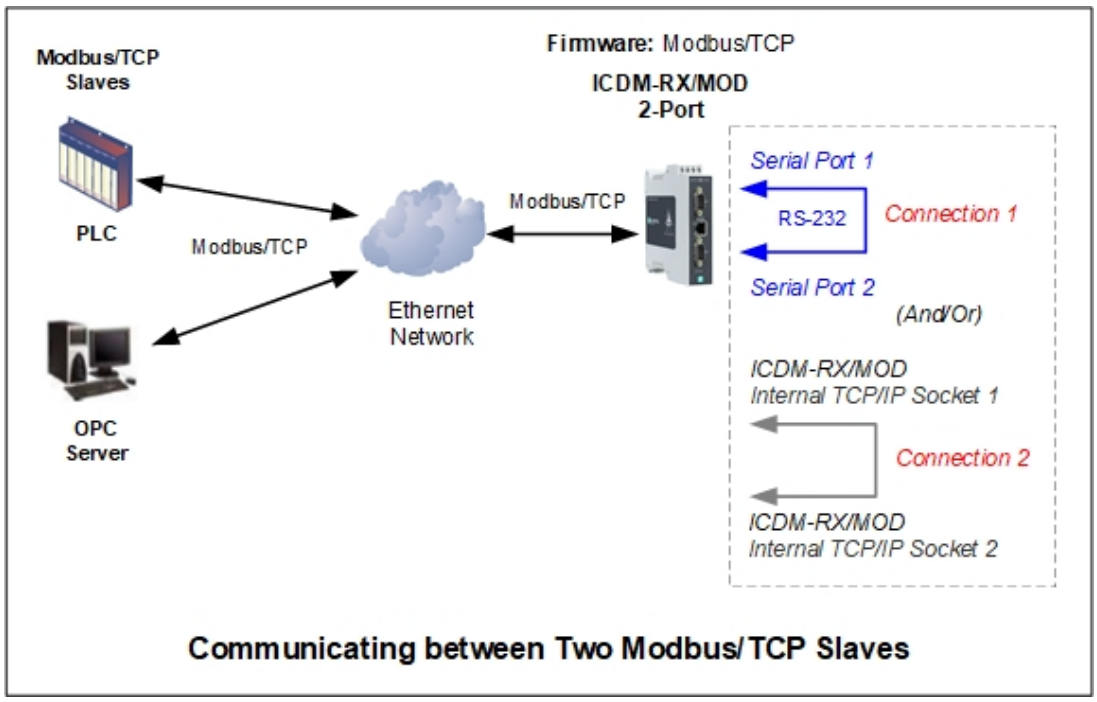
3.2.2 Modbus Slave-to-Slave Controller Connectivity

Modbus slave controllers can communicate to each other through the ICDM-RX/MOD. Possible communication options include:

- Two Modbus/TCP slaves communicating directly to each other.
- Two serial Modbus slaves communicating directly to each other
- Both Modbus/TCP and serial Modbus slave communication.

3.2.2.1 Two Modbus/TCP Slaves

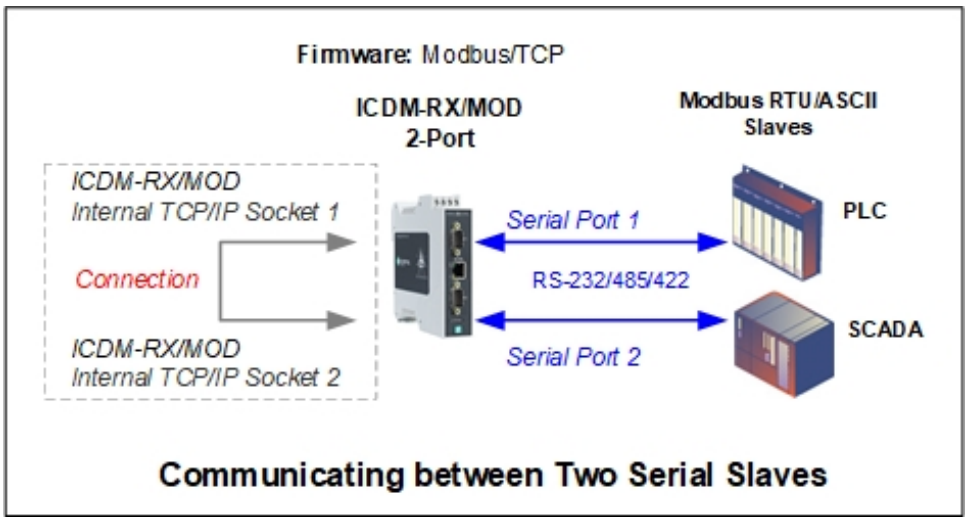
As shown in the following diagram, two Modbus/TCP slaves can communicate to each other using either internal TCP/IP socket connections or connecting two serial ports.



Communicating between Two Modbus/TCP Slaves

3.2.2.1.1 Two Serial Modbus Slaves

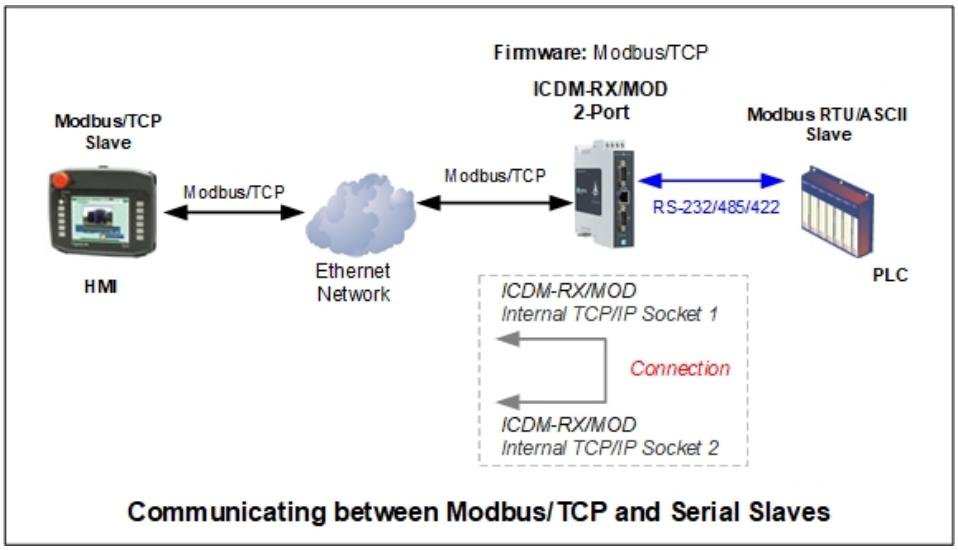
As shown in the following diagram, two serial Modbus slaves can communicate to each other using an internal TCP/IP socket connection.



Note: Serial connections could be made with a 4-Port ICDM-RX/MOD.

3.2.2.2 Modbus/TCP and Serial Modbus Slaves

As shown in the following diagram, Modbus/TCP and serial Modbus slaves can communicate to each other using an internal TCP/IP socket connection.



Note: Serial connections could be made with a 4-Port ICDM-RX/MOD.

4 Private Modbus Serial Buses

Security, visibility and extendibility are becoming increasingly important in today's Modbus installations. Engineers are being asked to provide more visibility to Modbus installations and, at the same time, are also being asked to keep devices secure from unauthorized access and errant configuration changes. To make things more challenging, they are also being asked to do more with existing equipment. And let us not forget the all-too-common limited budgets and tight schedules.

So, how does one add a SCADA or HMI system to an existing Modbus network without compromising the security of existing Modbus devices?

How does one design a Modbus installation that limits control of Modbus slave devices to a single Modbus master and yet can provide status information from those same device(s)?

How does one add additional equipment quickly, easily and economically?

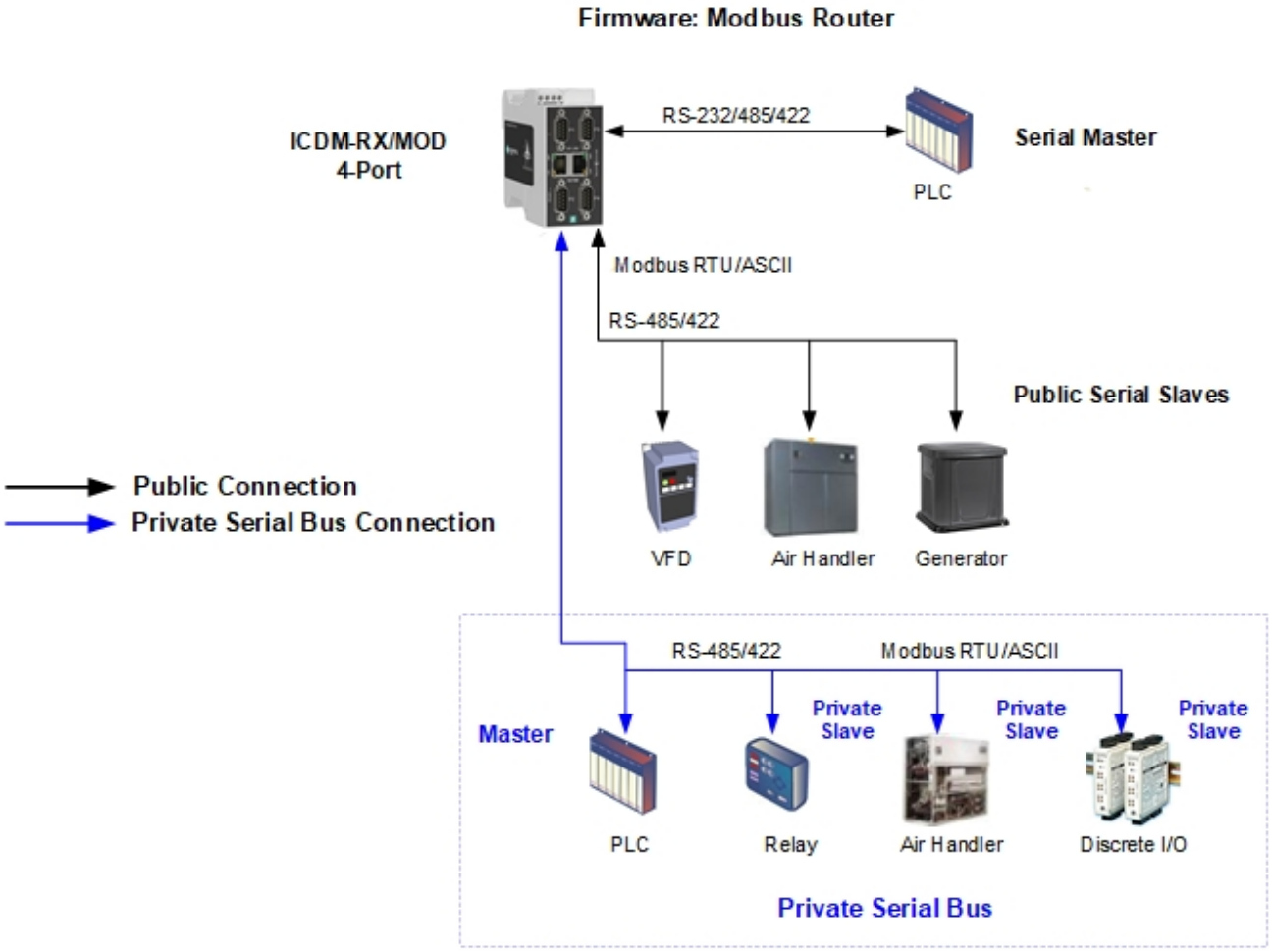
And how does one maintain such an installation once it is operational?

That is not easy – unless if ICDM-RX/MOD running Modbus Router.

4.1 Private Modbus Serial Bus Definition

A private Modbus serial bus is one that:

1. Contains both a Modbus master and one or more Modbus slaves on the same serial bus.
2. Allows only the Modbus master the ability to communicate directly to the Modbus slaves.
3. May provide access from the Modbus master to public Modbus network(s) via an advanced gateway, such as the ICDM-RX/MOD running Modbus Router firmware.



Private Serial Bus Communication

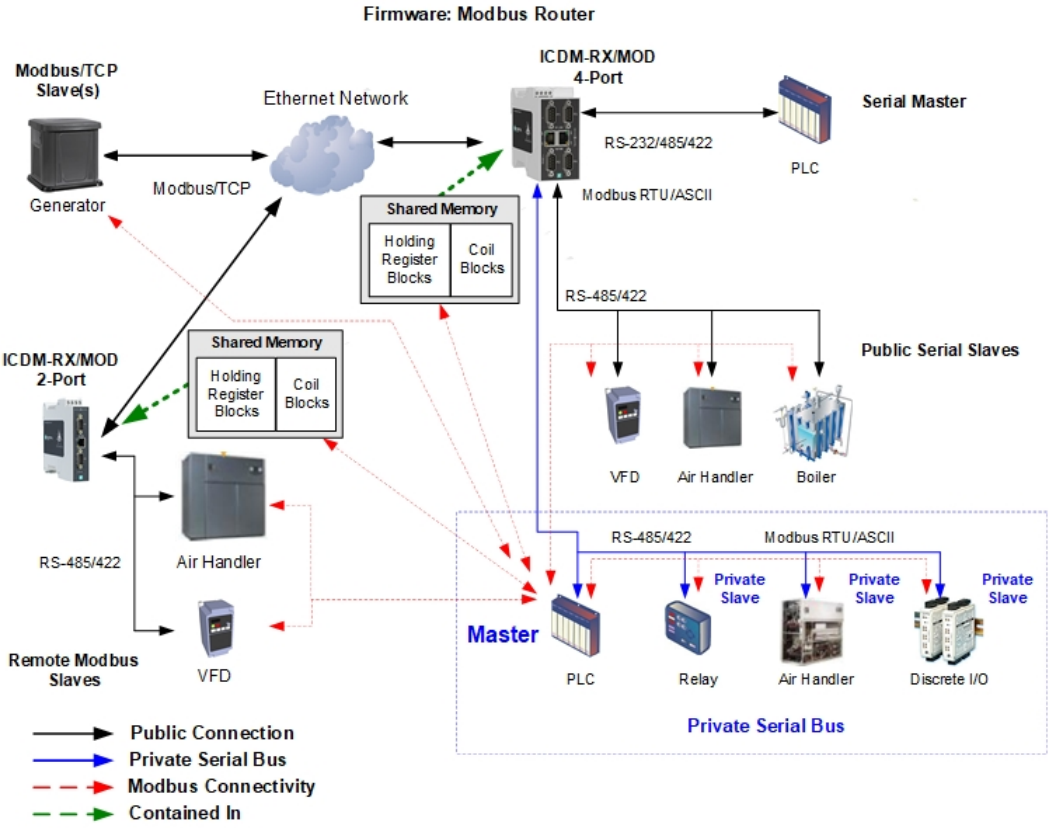
4.2 Private Serial Bus Capabilities

The Private Modbus serial bus functionality, combined with the many other Modbus Router features, provides powerful capabilities to aid plant engineers and system integrators.

4.2.1 Provides Modbus Network Connectivity to Private Serial Bus Masters

A serial Modbus master can communicate to slaves on its' own private serial bus as well as public slaves and other Modbus masters on a Modbus network. The connectivity includes the following:

- Modbus RTU/ASCII slave(s) on its own serial bus.
- Public Modbus RTU/ASCII serial slave(s) connected to the same ICDM-RX/MOD.
- Modbus/TCP slaves.
- Remote Modbus RTU/ASCII serial slave(s) via an Ethernet attached Modbus gateway.
- All other Modbus master(s) on the Modbus network via the Shared Memory functionality provided on every ICDM-RX/MOD running Modbus Router.
- The private master can provide data to/from its private slave(s) to the Modbus network, and other Modbus masters, via the Shared Memory functionality.

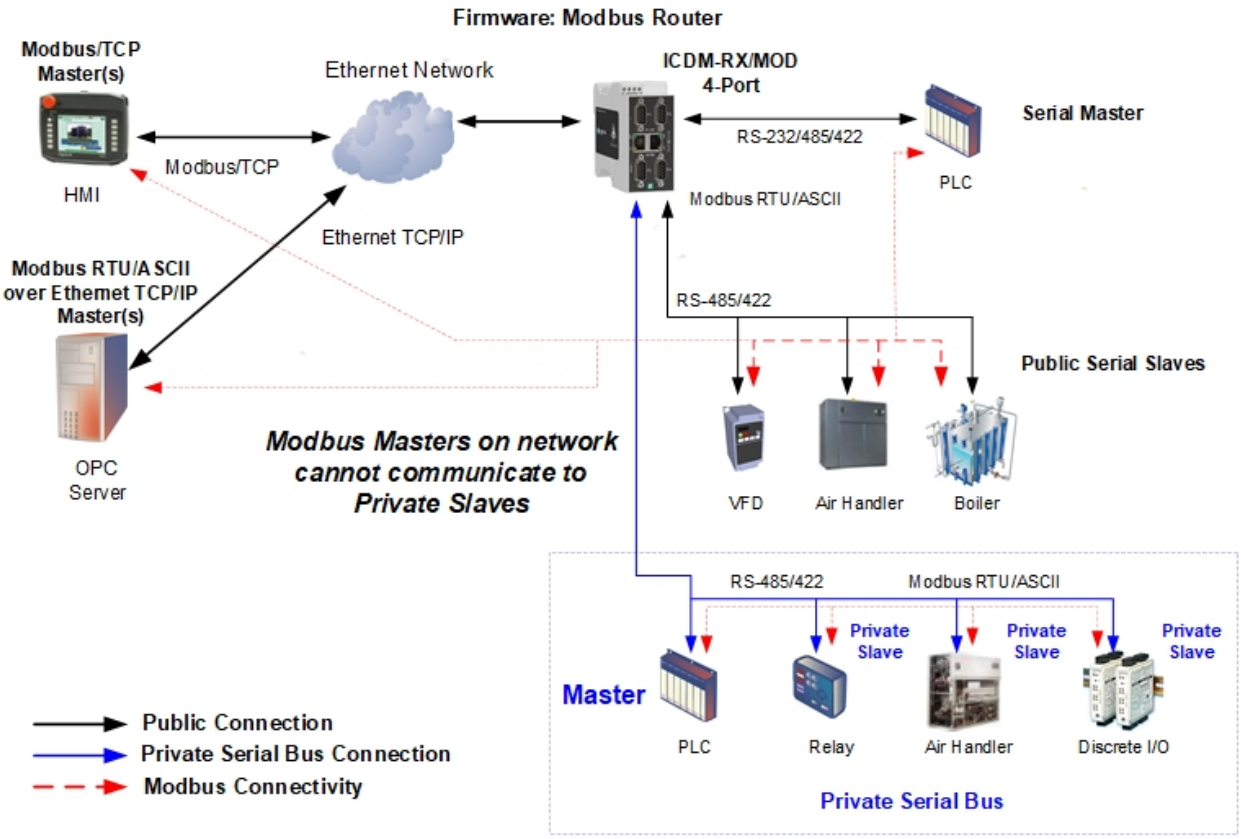


Private Serial Bus Master Connectivity

4.2.2 Provides Security for Private Modbus Slaves

The Modbus slaves on the serial bus are “private” to the master on that serial bus.

- The private slave device(s) are protected from all other Modbus masters on the Modbus network.
- The private master has total control of communication to the slave(s) on its private serial bus.

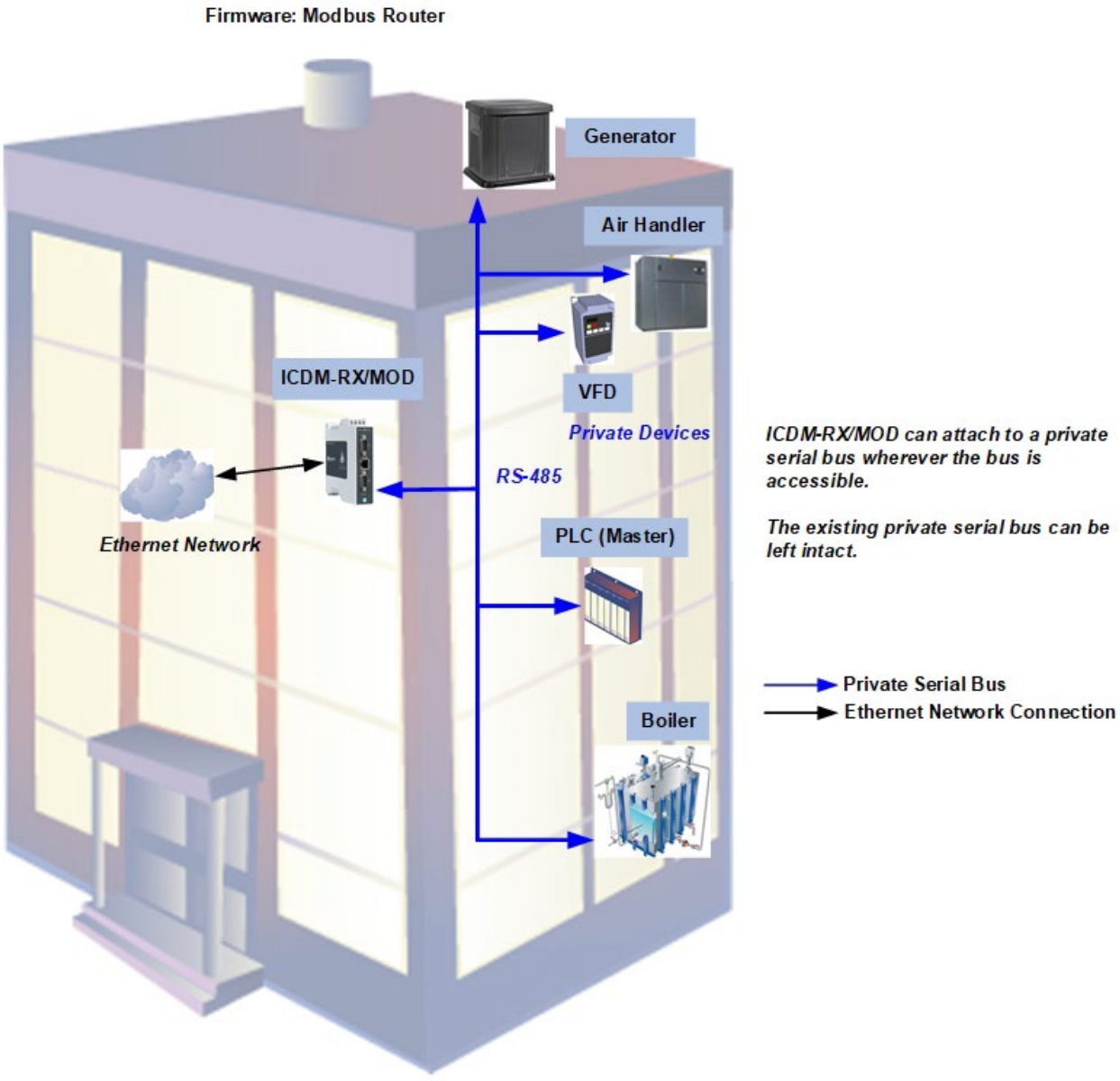


Private Serial Bus Security

4.2.3 Simplifies Deployment

Deployment can be greatly simplified.

- An existing serial bus can be left intact, thusly reducing the rewiring effort.
- The only required wiring change is to attach the ICDM-RX/MOD to the serial bus anywhere there is access.



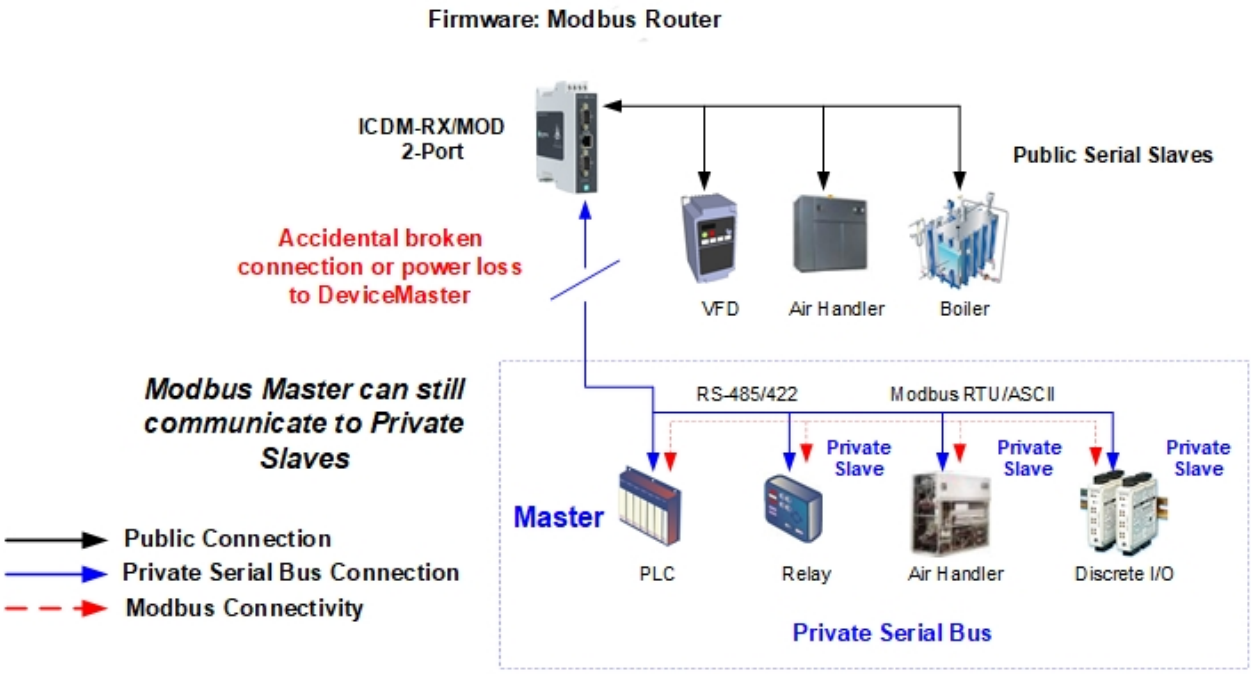
Connecting to a Private Modbus Serial Bus

4.2.4 Increased Fault Tolerance

Implementing Private Serial bus(s) can increase a communication systems' fault-tolerance.

4.2.4.1 Private Serial Bus Loses Connection to ICDM-RX/MOD

In the event the private serial bus is accidentally disconnected from the ICDM-RX/MOD is accidentally powered off, the master and slaves on the private serial bus can still communicate to each other.



Private Serial Bus with Lost Connection to DeviceMaster

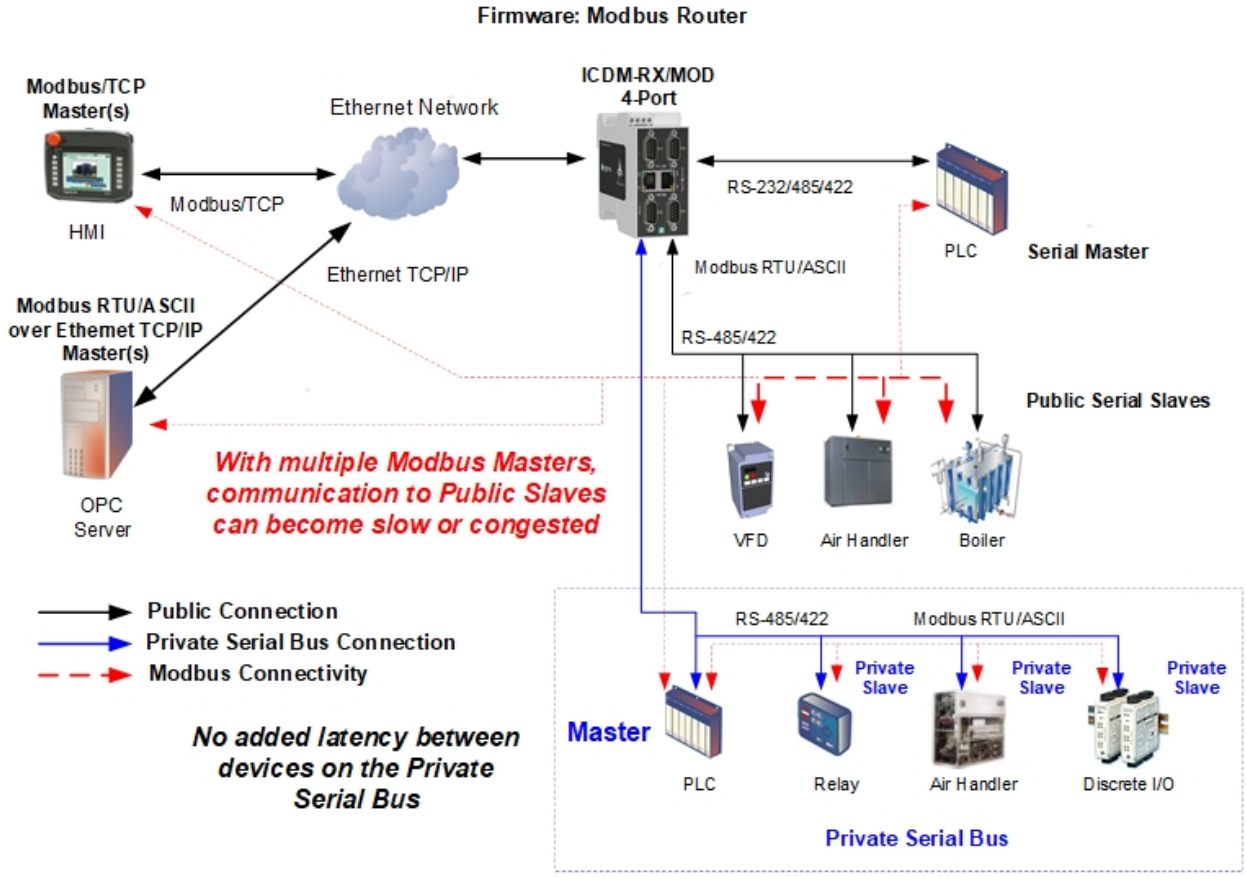
Note: *If the Modbus master and all Modbus slaves were attached to a Modbus gateway on separate serial ports, accidental disconnection or loss of power to the gateway would result in the master losing communication to all slaves. A private serial bus can eliminate such situations.*

4.2.4.2 Reduce Multiple Modbus Master Latency and Congestion Problems

In standard Modbus gateways, multiple Modbus masters communicating to Modbus slave device(s) can cause increased response latency and, in extreme cases, can cause the gateway to become congested. The main causes are:

1. **Latency:** By eliminating routing through the gateway, there is no added message latency between the master and slave device(s). While routing Modbus messages through the ICDM-RX/MOD adds minimal latency, that latency can become more significant if other Modbus masters are attempting to read from the device(s) at the same time. This is especially true when communicating to slow responding device(s).

2. **Congestion**: In worst-case scenarios, multiple Modbus masters sending requests to public slave device(s) faster than they can respond may cause the gateway to become congested. This, in turn, can force the gateway to reject some Modbus requests. By preventing other masters from communicating to the slave devices on the serial bus, a private serial bus can eliminate possible communication disruptions between the serial master and slaves caused by gateway congestion.



A Private Serial Bus Can Eliminate Added Latency and Congestion

4.3 Examples of Common Installations Using Private Serial Bus(s)

Private Serial bus connectivity can be used to solve many challenging installation requirements. The following examples display a few of the many ways that Private Serial Bus technology can be used to enhance and simplify installations.

4.3.1 Connecting Public Modbus Slaves to an Existing Modbus Serial Bus

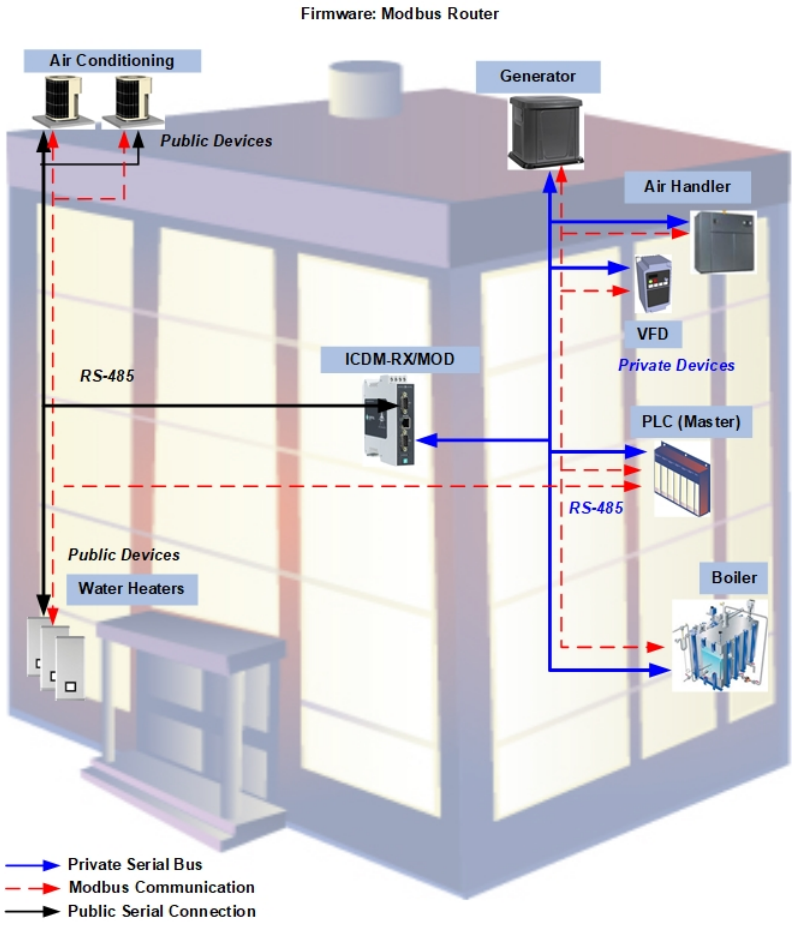
4.3.1.1 Connect to Local Serial Slaves

Requirements:

- A serial Modbus master on an existing serial bus requires communication to serial Modbus devices that can be connected to the same gateway.

Solution:

- Use a 2-Port or 4-Port ICDM-RX/MOD to provide both the public and private serial connections. Messages from the serial master will be forwarded to the public devices.



Connecting Serial Modbus Master to Public Modbus Slaves

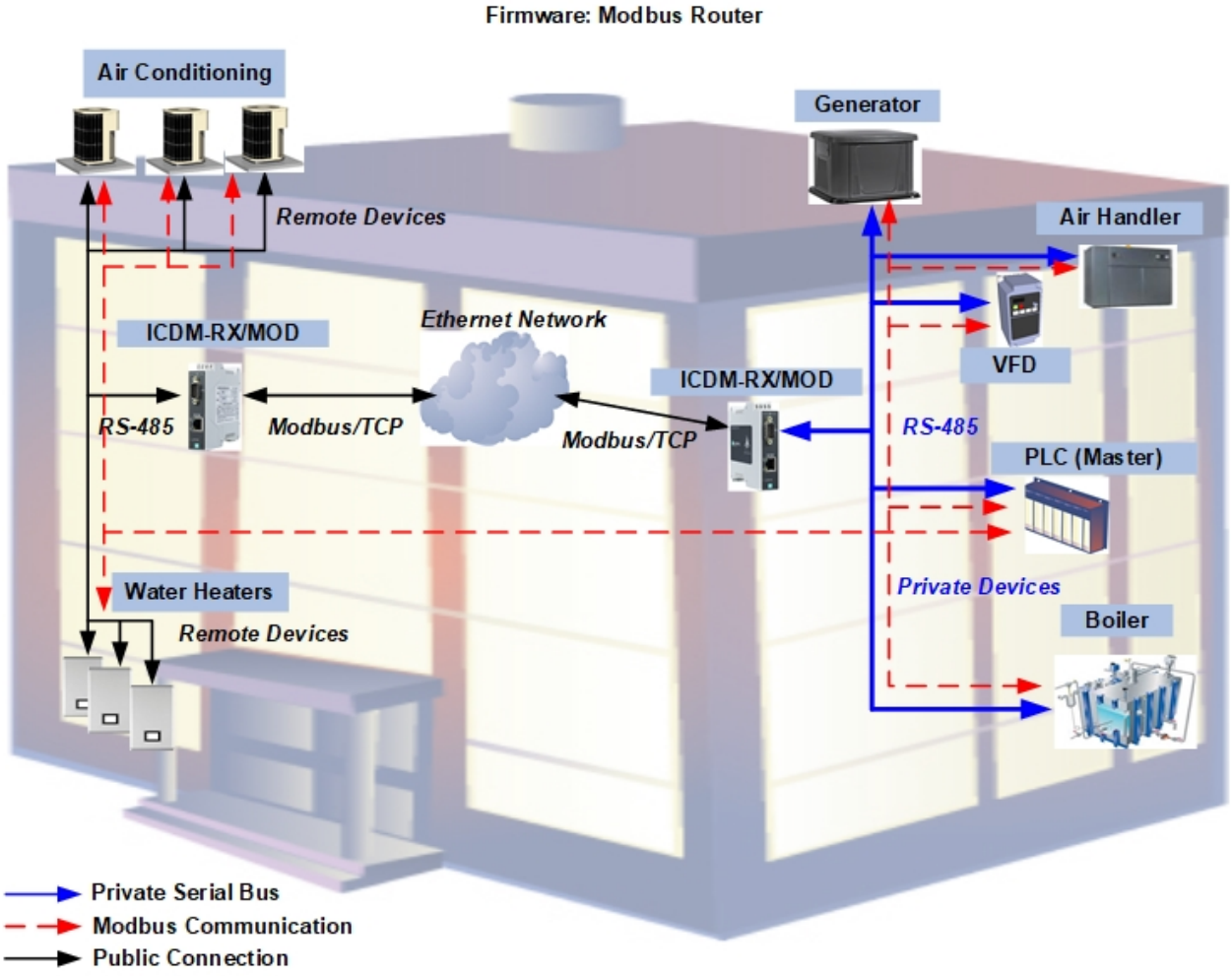
4.3.1.2 Connect to Serial Slaves over Ethernet Network

Requirements:

- A serial Modbus master on an existing serial bus requires communication to Modbus devices over an Ethernet network.
- The Modbus slave devices on its own serial bus must be secure from the rest of the Modbus network.

Solution:

- Use the Ethernet network and two ICDM-RX/MOD gateways to connect the Modbus devices to the serial master.



Connecting Serial Modbus Master to Remote Modbus Slaves

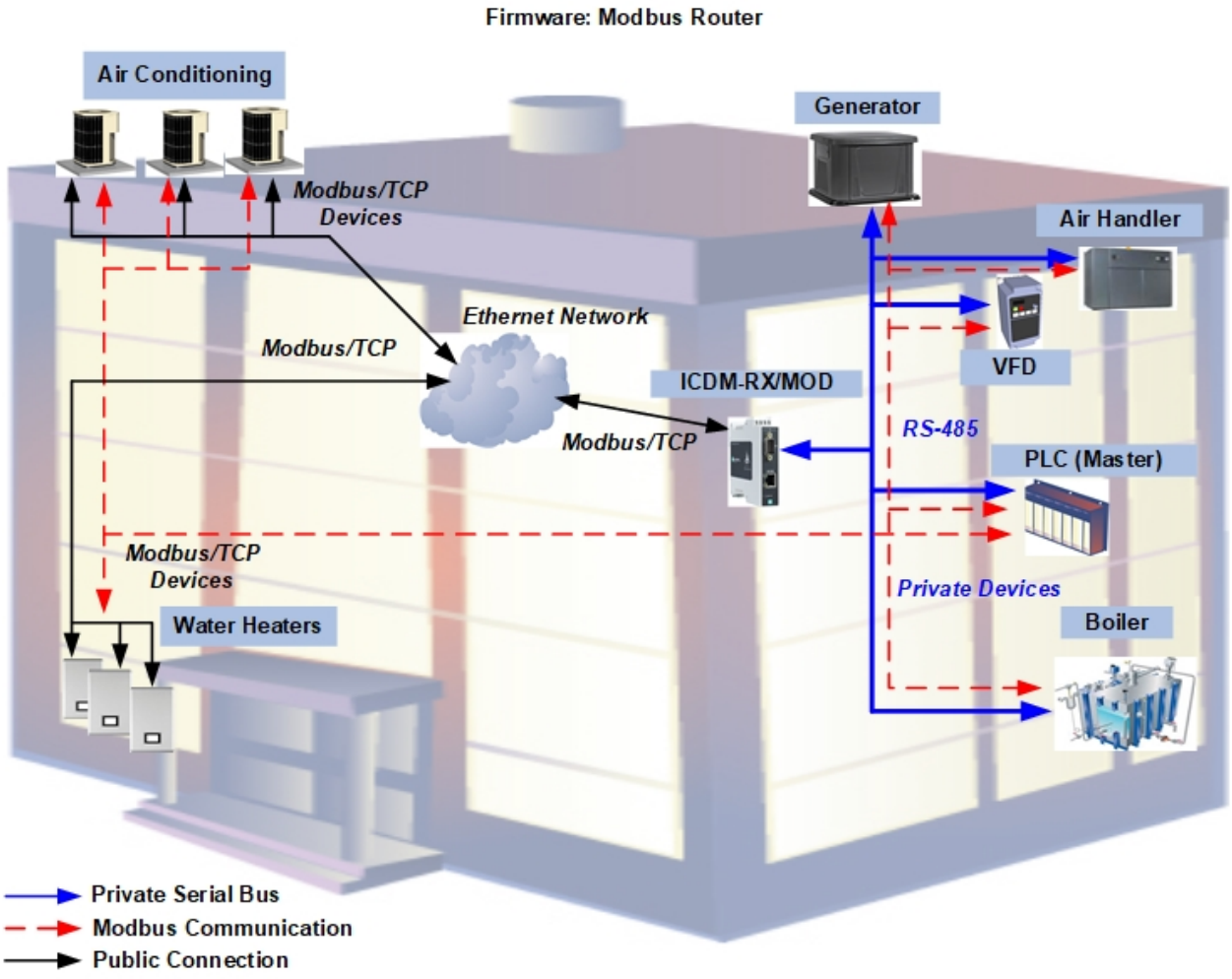
4.3.1.3 Connect to Modbus/TCP Slaves

Requirements:

- A serial Modbus master on an existing serial bus requires communication to Modbus/TCP devices.
- The Modbus slave devices on its own bus need to be secure from the rest of the Modbus network.

Solution:

- Use an ICDM-RX/MOD to provide the communication from the private serial bus to the Modbus/TCP devices.



Connecting Serial Modbus Master to Modbus/TCP Slaves

4.3.2 Providing Master-to-Master Communication

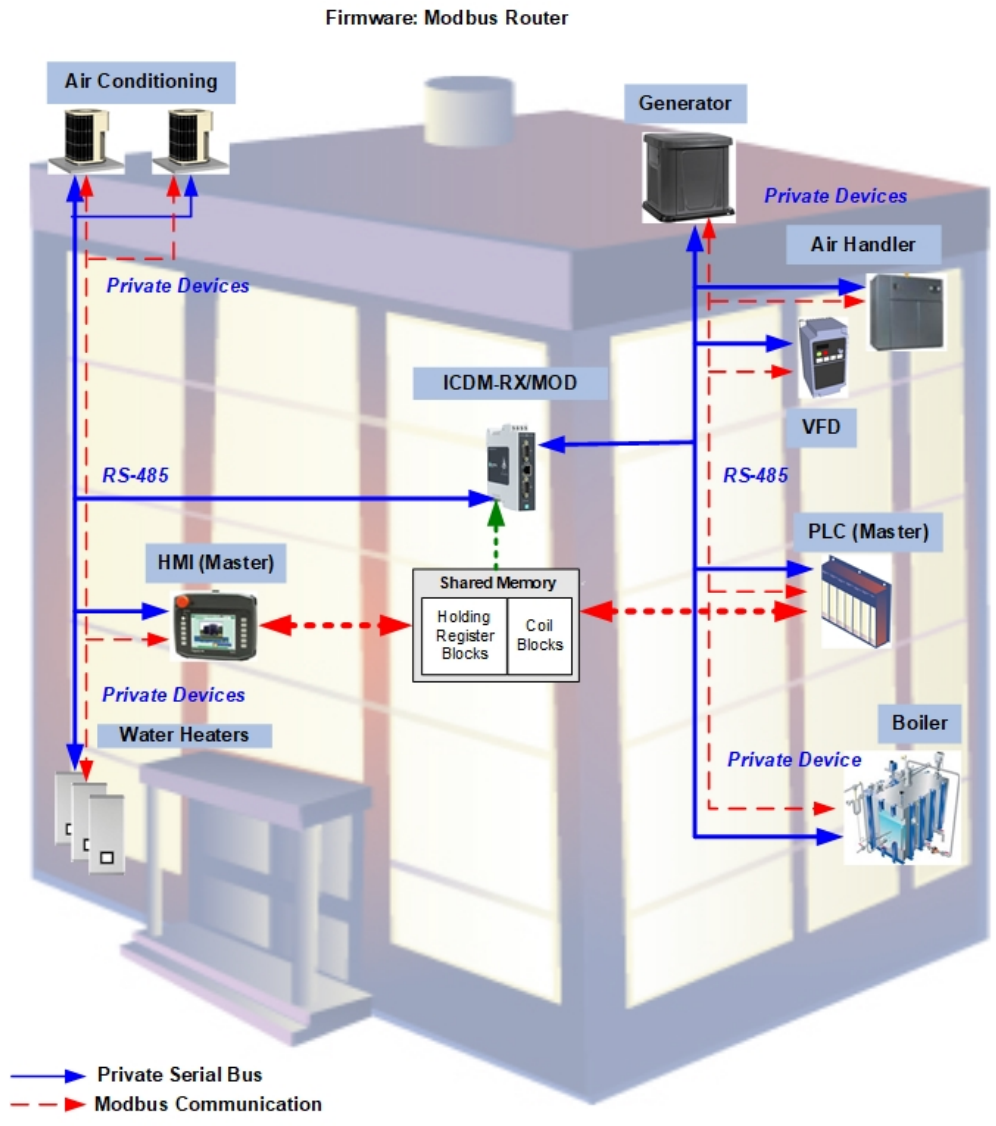
4.3.2.1 Communicate Between Serial Modbus Masters

Requirements:

- Communication is required between two serial Modbus masters residing on existing serial buses.
- Existing serial Modbus slaves are to remain protected from the Modbus network.

Solution:

- Use a 2-Port or 4-Port ICDM-RX/MOD running Modbus Router:
  - 2 serial ports configured to Private Loop mode (To-Master/Slaves)
  - Shared Memory configured to allow the two masters to communicate to each other



Connecting Two Serial Masters via Shared Memory

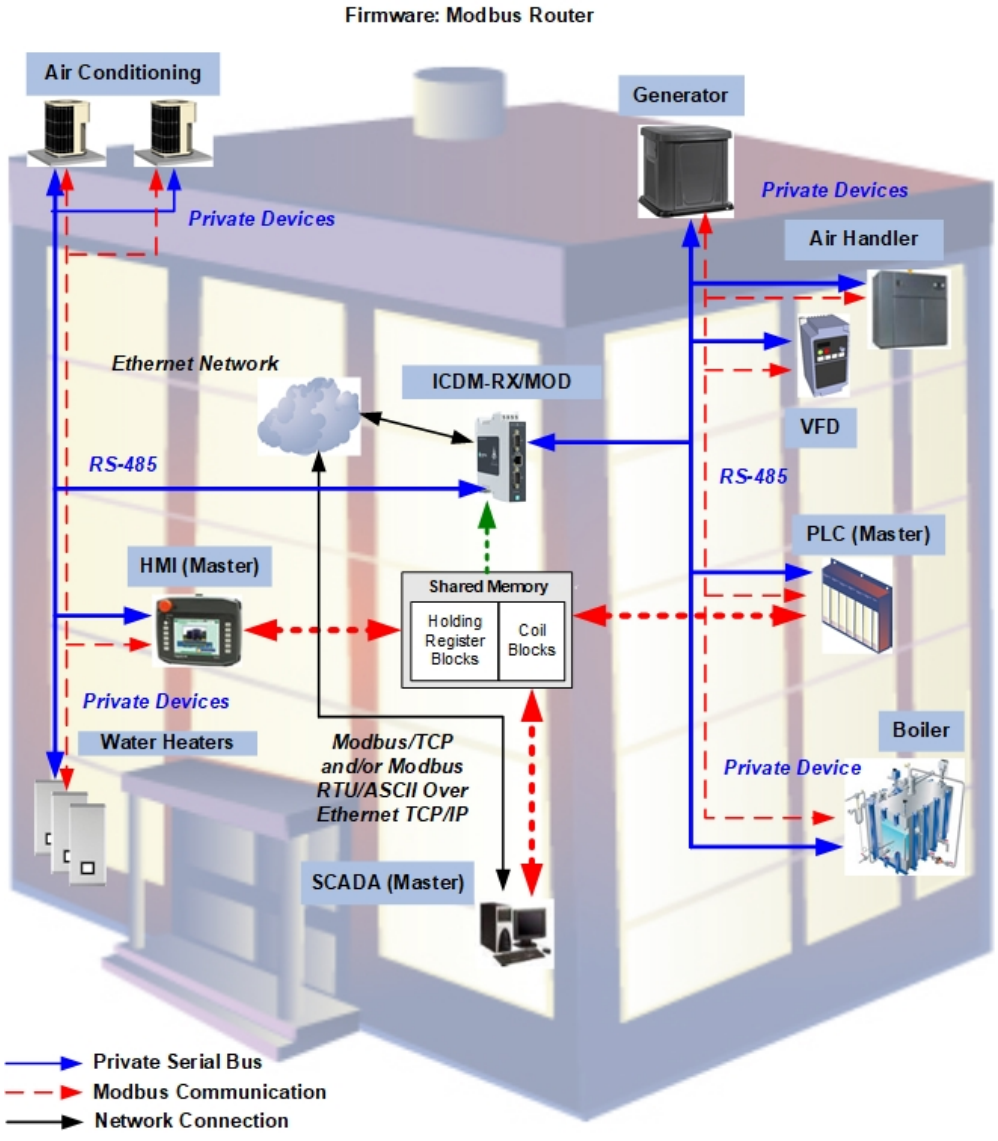
4.3.2.2 Ethernet Based Modbus Master(s) to Serial Modbus Masters

Requirements:

- Communication is required between Modbus/TCP and / or Modbus RTU/ASCII over Ethernet master(s) to serial Modbus master(s) on existing serial buses.
- Existing serial Modbus slaves are to remain protected from the Modbus network.

Solution:

- Use an ICDM-RX/MOD running Modbus Router:
  - Serial port(s) configured to Private Loop mode (To-Master/Slaves)
  - Shared Memory configured to allow the Ethernet master(s) and serial master(s) to communicate to each other



Connecting Ethernet Master to Serial Masters via Shared Memory

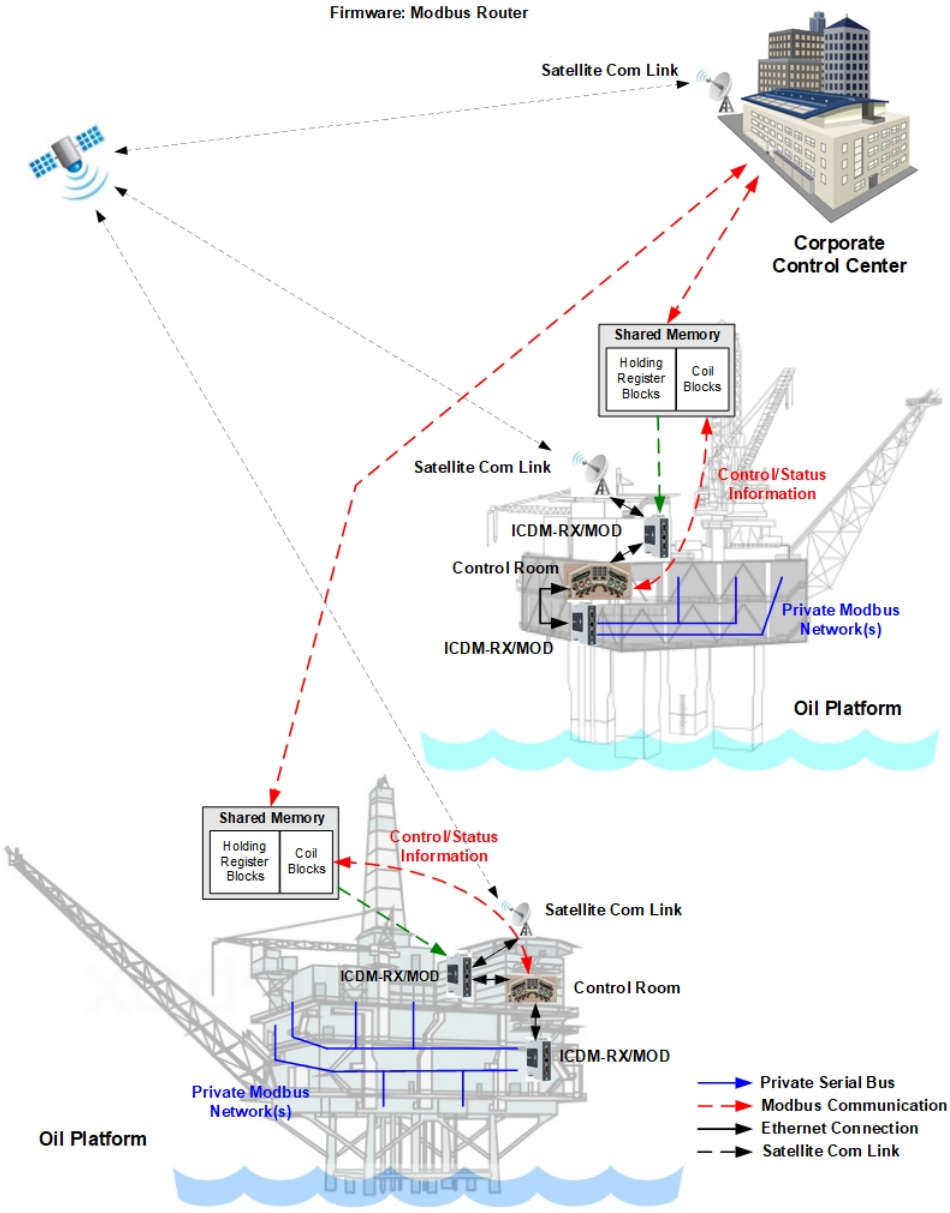
4.3.3 Access to Remote Installations

Requirements:

- Communication is required between multiple installations. This may include Ethernet and/or serial based masters combined with private serial buses.
- Modbus slave devices on each installation must be protected from other installation(s).

Solution:

- Use a series of ICDM-RX/MOD running Modbus Router:
  - Serial port(s) configured to Private Loop mode (To-Master/Slaves)
  - Shared Memory configured to allow the Ethernet master(s) and serial master(s) to communicate to each other



5 Read Only Modbus Protection

Two of the greatest challenges are providing increased access to device level data and the ever increasing need for security. These two challenges are compounded because as access to data increases, so do the security risks.

So, how does one solve these two conflicting challenges? Secure networks are a good solution for tightly controlled installations. But what if your installation can no longer be tightly controlled? What if your IT department now requires data for their new SCADA system? What if a government agency wants to monitor the installation? And what if you have no direct control over those monitoring systems and cannot prevent those systems from attempting to change device level configurations or set-points? Then what?

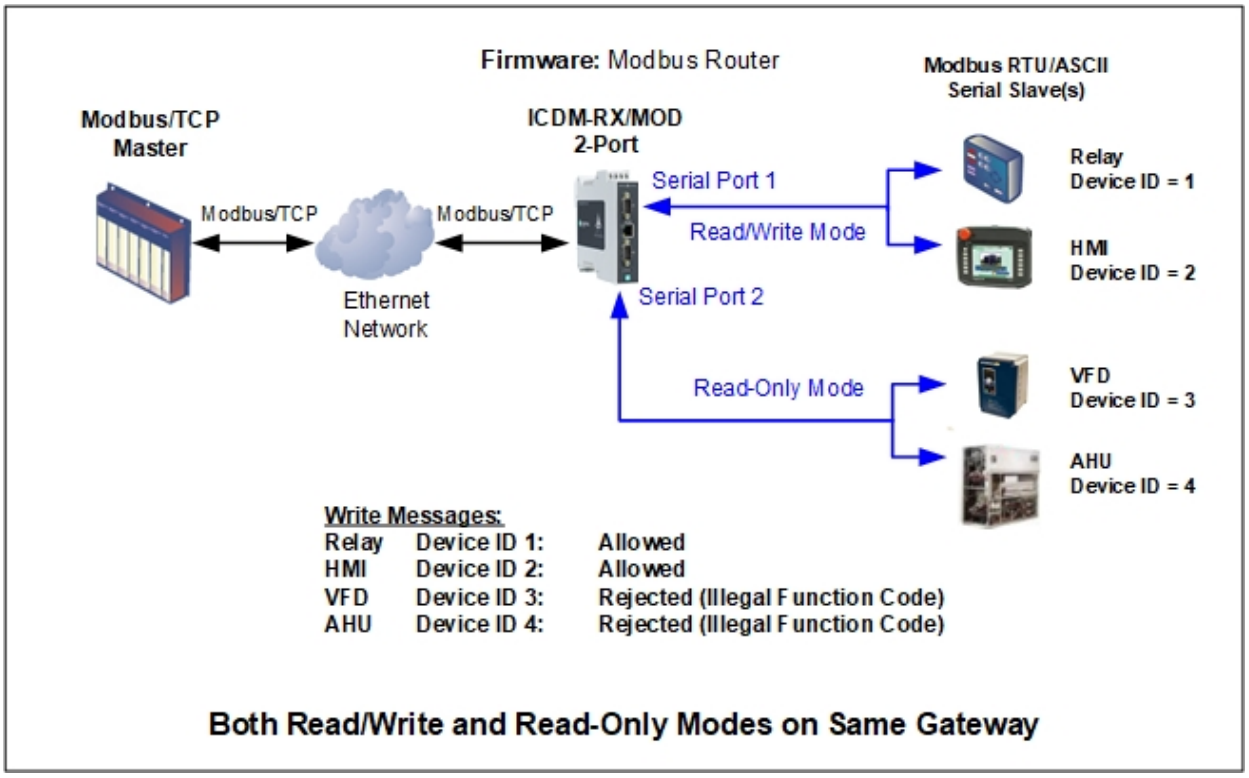
In order to provide the required data and prevent unauthorized changes, the Read-Only serial ports option was added to the Modbus Router firmware. This *Disable Writes (Read Only)* option affectively creates Read-Only devices by rejecting all standard Modbus write messages. This allows monitoring systems to retrieve the required data and prevents changes by blocking all write messages from being transmitted to the serial devices. Write violations are also logged to help locate the source of the write messages.

With an extensive set of connectivity options and the Read-Only Modbus message option, the Modbus Router firmware application provides the connectivity and security options required for today's challenging installations.

5.1 Implementing the Disable Writes (Read Only) Option

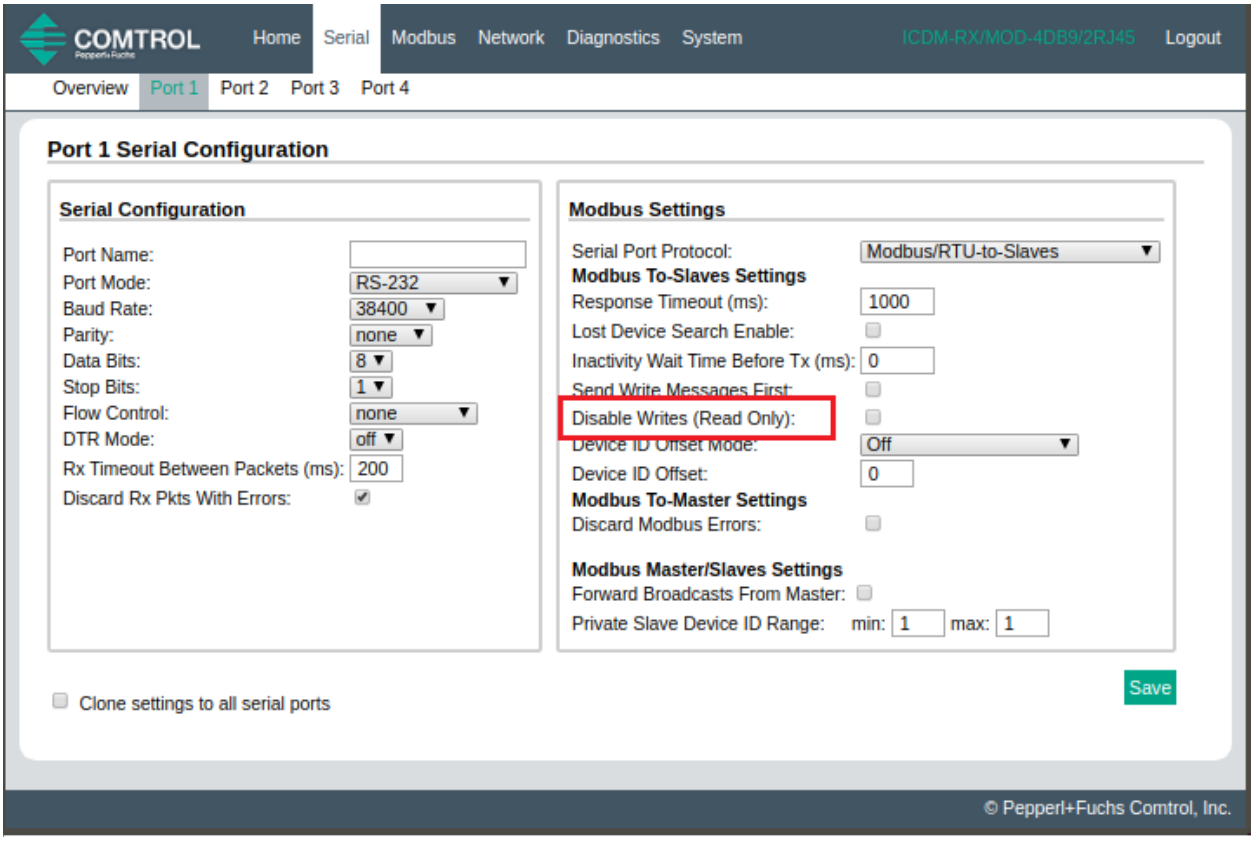
The *Disable Writes (Read Only)* option has been developed to block all standard Modbus write messages from being transmitted out a serial port. An entire gateway can be configured to be Read-Only by selecting this option on all of its serial ports.

The Read-Only mode as compared to standard Read/Write mode is demonstrated in the following diagram:



5.1.1 Web Page Configuration

The ***Disable Writes (Read Only)*** option is enabled using the serial port configuration page:



5.2 Solutions for Read-Only Modbus Devices

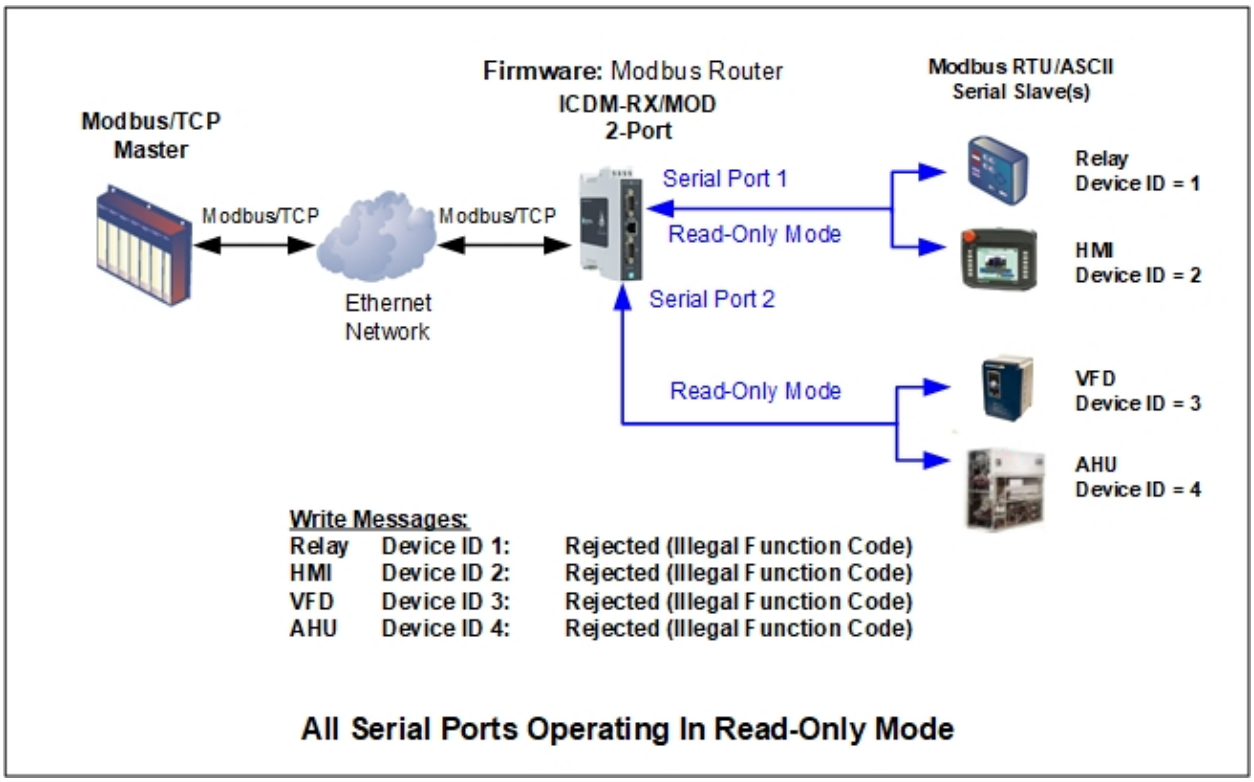
5.2.1 Providing Access to Read-Only Modbus Devices

Requirements:

- A Modbus master needs to communicate to read-only devices and it is desired to block all write messages.

Solution:

- Enable *the Disable Write (Read Only)* option for all serial ports on the ICDM-RX/MOD.



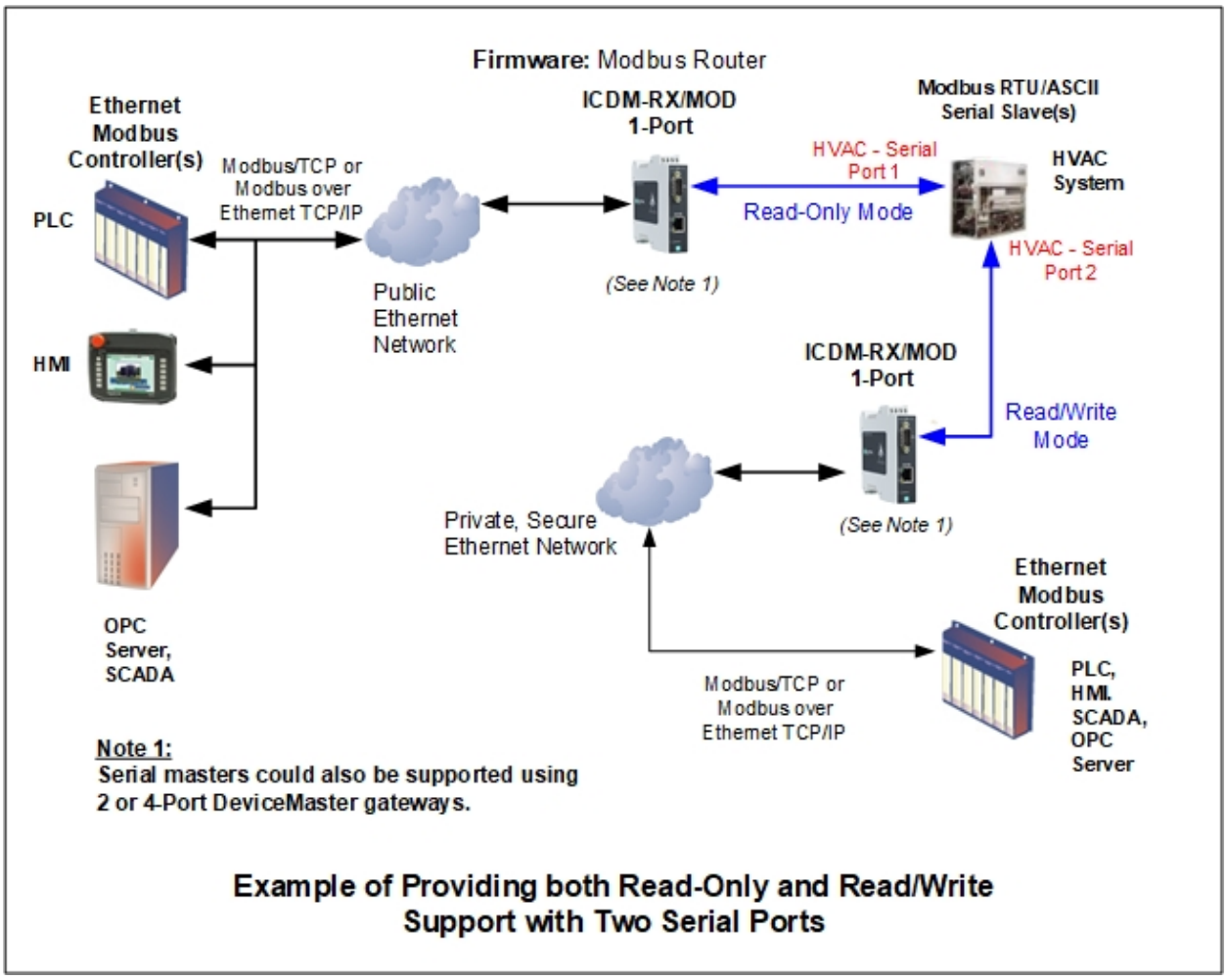
5.2.2 Accessing Read-Only and Read/Write Devices that have Two Serial Ports

Requirements:

- Read-Only access is required for a public network and Read/Write is required for a private, secured network.

Solution:

- Use two ICDM-RX/MOD gateways, one connected to each device serial port, to provide the desired connectivity.
- The Read-Only gateway is connected to the public network and the read/write gateway is connected to the private, secure network.



Example of Providing both Read-Only and Read/Write Support with Two Serial Ports

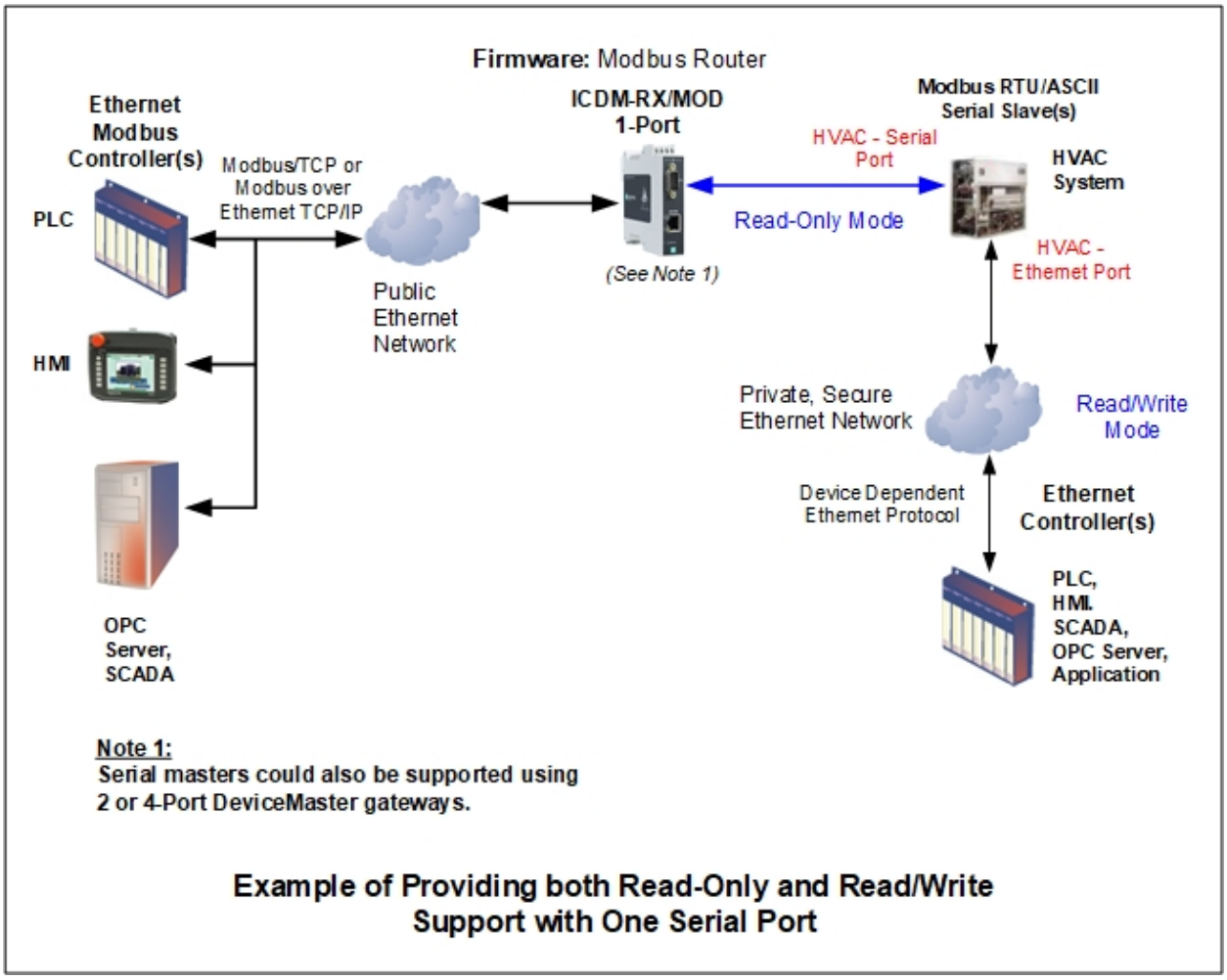
5.2.3 Accessing Read-Only and Read/Write Devices that have One Serial Port and One Ethernet Port

Requirements:

- Read-Only access is required for a public network and Read/Write is required for a private, secured network.

Solution:

- Connect the Ethernet port to the private, secured network. Connect an ICDM-RX/MOD gateway to the device's serial Modbus port.
- The gateway is then connected to the public Ethernet network.



Example of Providing both Read-Only and Read/Write Support with One Serial Port

6 Resolving Modbus Device ID Conflicts

Many of the most common challenges faced when configuring Modbus installations involve conflicts arising from device ID assignments. Device ID reassignment is often required when deploying, updating or reconfiguring installations and that is not always easy or possible.

In an ideal situation, device ID reassignment can be accomplished rather easily with a simple configuration setting or setting a toggle switch. But all too often device IDs can be difficult or, in some cases, impossible to reassign. Devices may be inaccessible or a system is already in operation and changes to the existing installation are not allowed. Furthermore, reassigning device IDs almost always requires changes to the controller logic and there could be problems or barriers to changing that as well. Even when controller logic can be modified, doing so is often time consuming and expensive. Compounding this are everyday problems such as limited project funding, engineering resource availability, aggressive deployment schedules, system testing, and approval processes.

With the goal to solve these problems, Pepperl+Fuchs Comtrol has developed the Alias Device ID and Device ID Offset functionality. This functionality has been designed to resolve even the most challenging device ID conflicts. Modbus device IDs can be effectively reassigned without modifications to controller logic or slave devices. All it takes is a few entries through logical and easy-to-use web configuration pages. With this advanced functionality, system integrators and plant engineers can now solve device ID conflicts in minutes instead of days, weeks, or months.

6.1 Common Causes of Modbus Device ID Conflicts

6.1.1 Modbus Specification Limitations

Many device ID conflicts arise simply due to the limitations of the Modbus Specification. The Modbus specification has the following limitations:

- Requires all devices attached to gateway to be addressed by a device ID.
- Allows only 256 device IDs with a range of 0 to 255.
- Not all device IDs can be used for addressing devices.
    - Device ID 0 is reserved for broadcast messages
    - 1-247 are for device addressing
    - 248 to 255 are reserved for such things as gateway functions. Depending on your environment, these device IDs may or may not be available for assignment to devices.

6.1.2 Common Implementation Constraints

The following implementation constraints can also cause device ID conflicts:

- It is not always possible or practical to change the device ID of serial Modbus slave devices.
- It is not always possible or practical to modify the device IDs on existing Modbus master programs. For instance, this is often true when adding a SCADA system to an existing PLC controlled system.
- A gateway must route Modbus messages based on the device ID. Therefore, it cannot route to multiple Modbus devices with the same device ID.
- Modbus masters with only one available connection may need to access multiple devices with the same device ID. Furthermore, these devices may be located locally or remotely.

The Alias Modbus Device ID and Device ID Offset functionality have been developed to resolve these device ID conflicts.

6.2 Alias Modbus Device ID Functionality

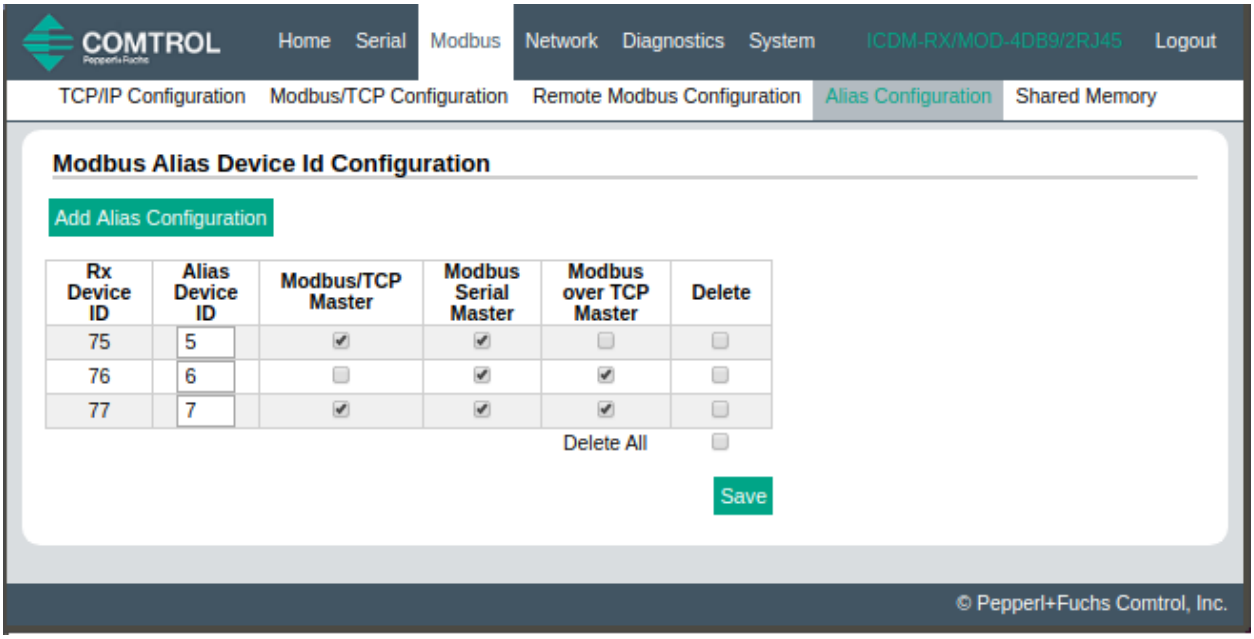
Available on both the Modbus Router and Modbus/TCP firmware applications, the Alias Device ID functionality has been developed primarily to help resolve device ID conflicts involving Modbus masters. These conflicts arise from situations such as:

- A controller requiring connectivity to multiple devices with the same device ID located locally and remotely
- Both controllers and slave devices that cannot be modified, but yet require connectivity
- Multiple controllers requiring access to the same device, but must use different device IDs to access the device
- Single connection controllers, such as serial or Ethernet TCP/IP, that require full usage of the Modbus device ID range

Device ID Aliasing involves:

- Modifying the received device ID to an “aliased” device ID immediately after the message is received from a Modbus master
- The Modbus message is then routed throughout the Modbus gateway or network with the aliased device ID.
- The response message is returned to the Modbus master with the original device ID.
- Modbus masters can communicate to slave devices through either the actual device ID or through an Alias Device ID conversion.

The following web page displays Alias Modbus Device ID Configurations.



6.3 Device ID Offset Functionality

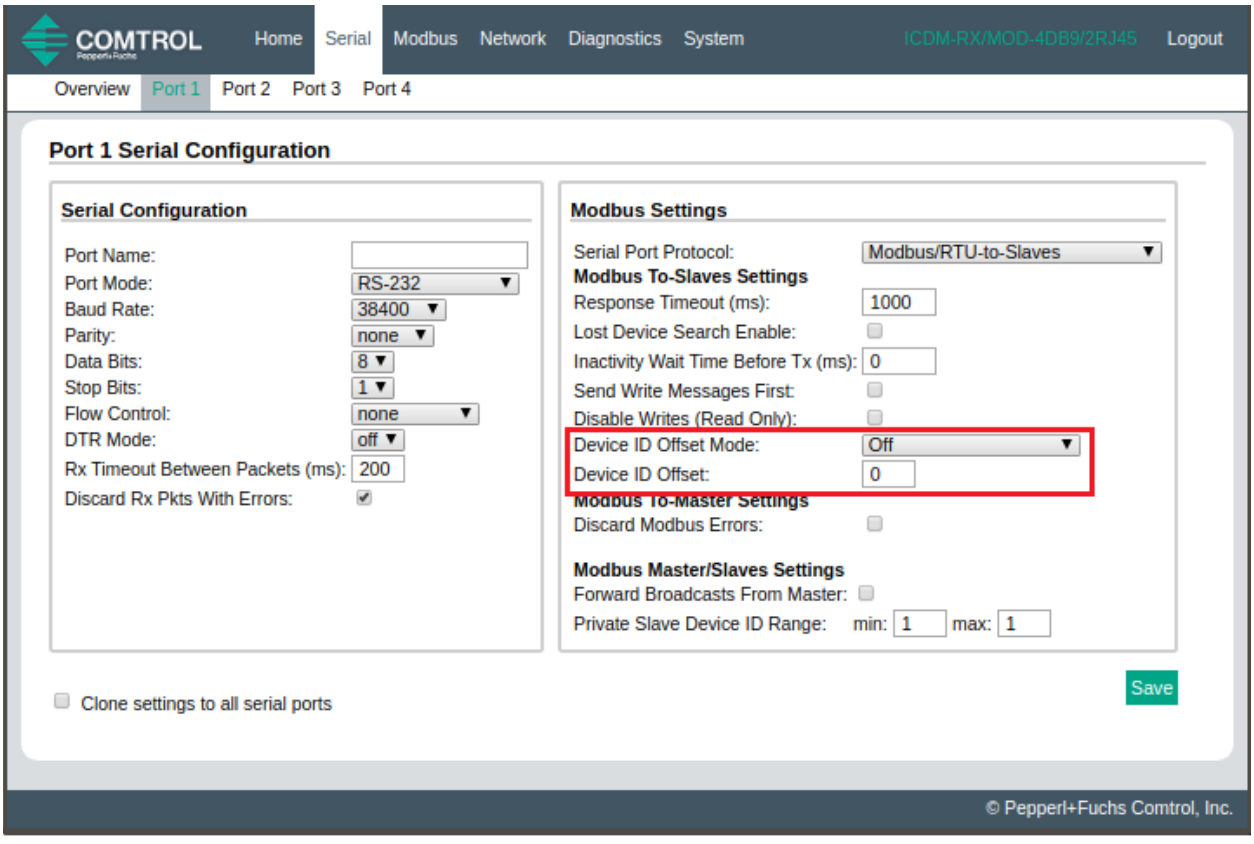
Available on the Modbus Router firmware application, the Device ID Offset functionality has been designed primarily to resolve device ID conflicts involving Modbus slave devices. These conflicts typically arise when slave device IDs cannot be changed and it is desired that two or more slave devices with the same device ID be attached to the same gateway.

The Device ID Offset functionality involves:

- Adding or subtracting an offset to the message device ID immediately before the messages are transmitted out the serial port.
- The device ID, as seen by the Modbus gateway and/or network, is effectively reassigned without any configuration changes to the slave devices.
- The responses, when received from the serial port interface, are immediately converted back to the message device ID and routed back to the Modbus master on the Modbus network.
- The device ID offset is applied to all devices connected to the same serial port.

The Device ID Offset and Alias Device ID functionality can be used together to solve device ID conflicts.

Device ID Offset web page configuration options.

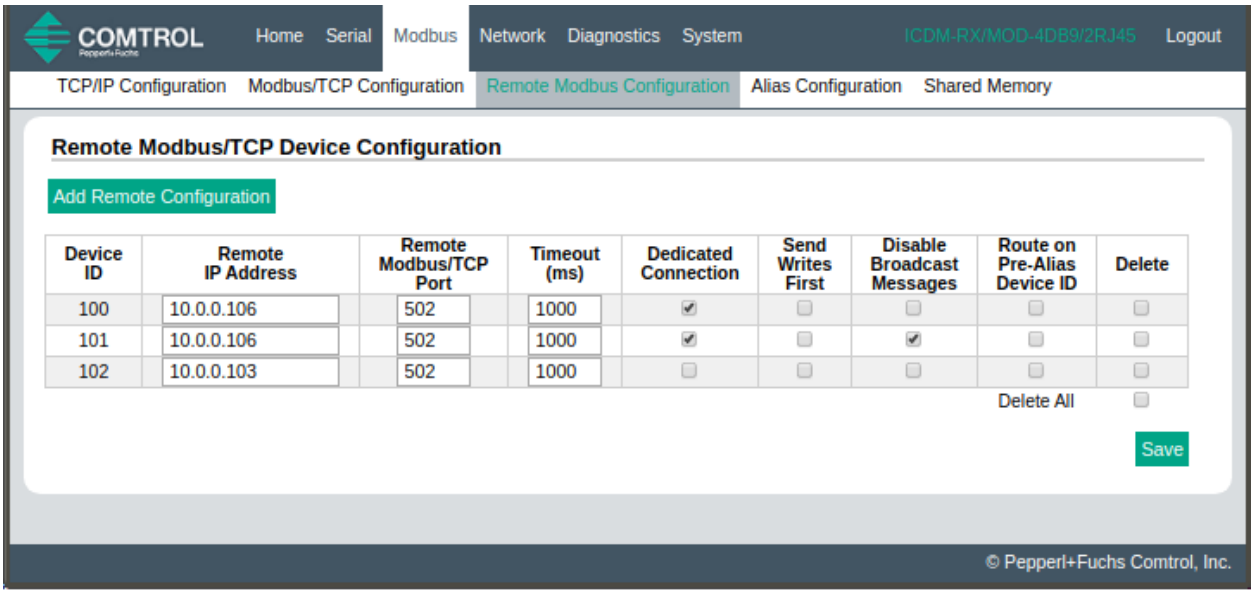


Device ID Offset Mode options include: ***Off***, ***Add-To-Msg-ID***, ***Subtract-From-Msg-ID***.

6.4 Remote Modbus/TCP Device Connectivity

Available on the Modbus Router firmware application, the Remote Modbus/TCP Device ID functionality provides connectivity to either Modbus/TCP slaves or serial Modbus slave devices connected to other Modbus gateways. The Remote Device ID functionality can be used in conjunction with the Alias Device ID and Device ID Offset functionality to provide network-wide Modbus connectivity solutions.

The following web page capture displays Remote Modbus/TCP Device Configurations.



6.5 Solutions to Common Device ID Conflicts

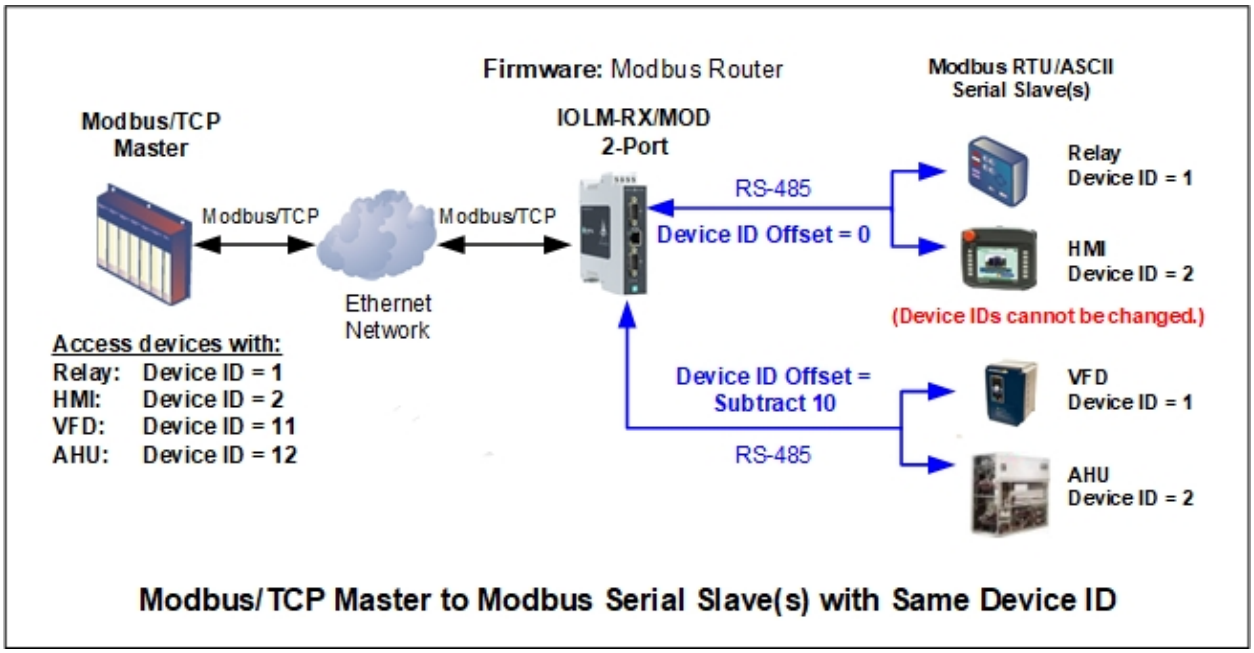
6.5.1 Modbus/TCP Master Communicating to Local Device(s) with Same Device ID

Requirements:

- A Modbus/TCP master needs to communicate to different devices on the same gateway with the same device IDs. It is desired to connect all devices to the same gateway.

Solution:

- Use Device ID Offset to provide unique device IDs for each device.



Modbus/TCP Master to Modbus Serial Slave(s) with Same Device ID

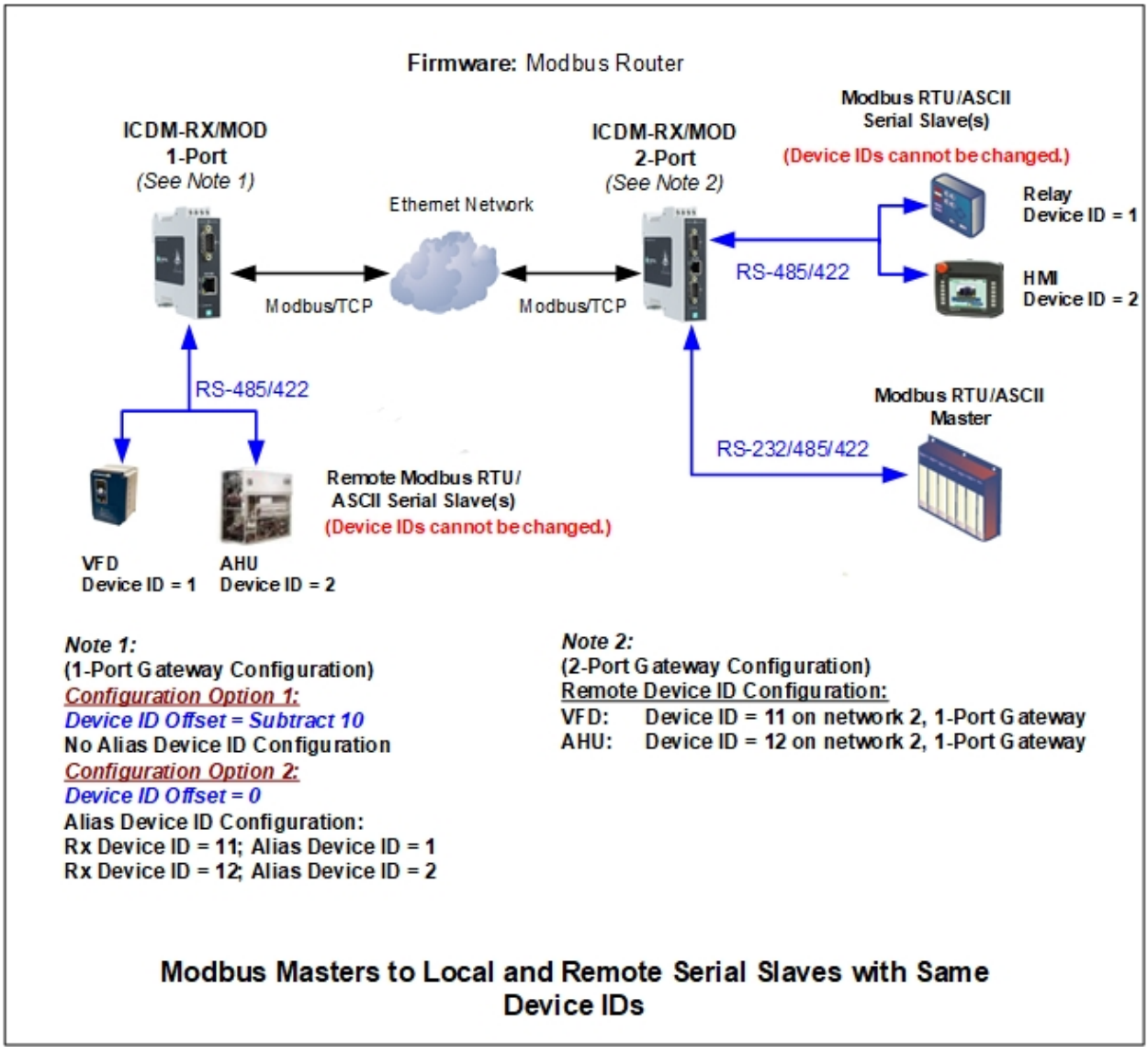
6.5.2 Modbus Serial Master Communicating to Local and Remote Devices with Same Device IDs

Requirements:

- A Modbus Serial Master needs to communicate to different devices with the same device IDs. Two are connected locally and two are connected remotely via another gateway.

Solution:

- Use Remote Device Routing and Alias Device ID functionality to provide unique device IDs for both the remote devices.



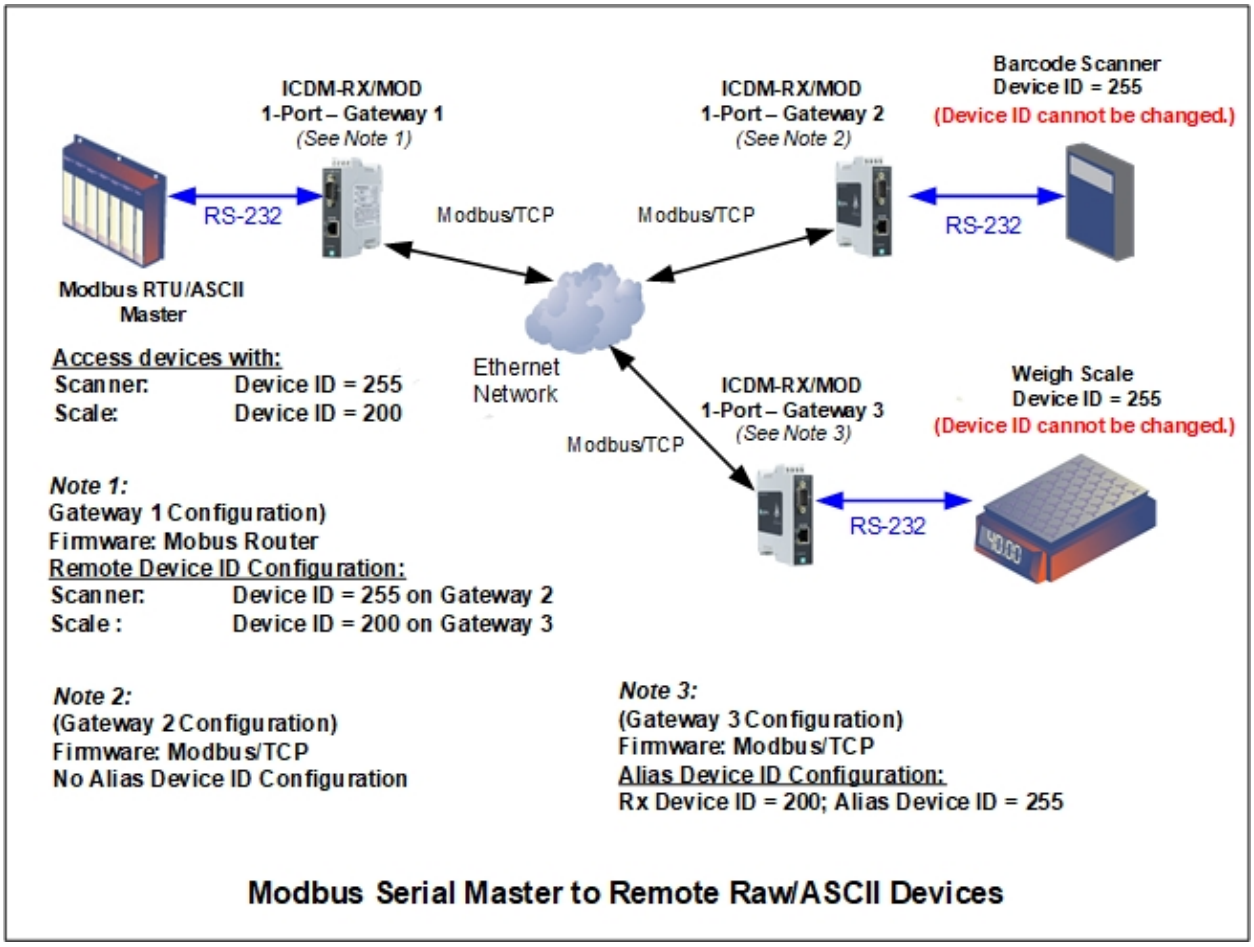
6.5.3 Modbus Serial Master Communicating to Two Remote Serial Raw/ASCII Devices

Requirements:

- A Modbus Serial Master requires connectivity to two remotely located raw/ASCII devices.

Solution:

- Use one gateway to provide Modbus/TCP connectivity from the serial master to the Modbus network.
- Use one gateway to provide communication to each remote raw/ASCII device.
- Use Remote Device Routing and Alias Device ID functionality to provide communication to the serial raw/ASCII devices.



6.5.4 Merging Two Serial Modbus Networks

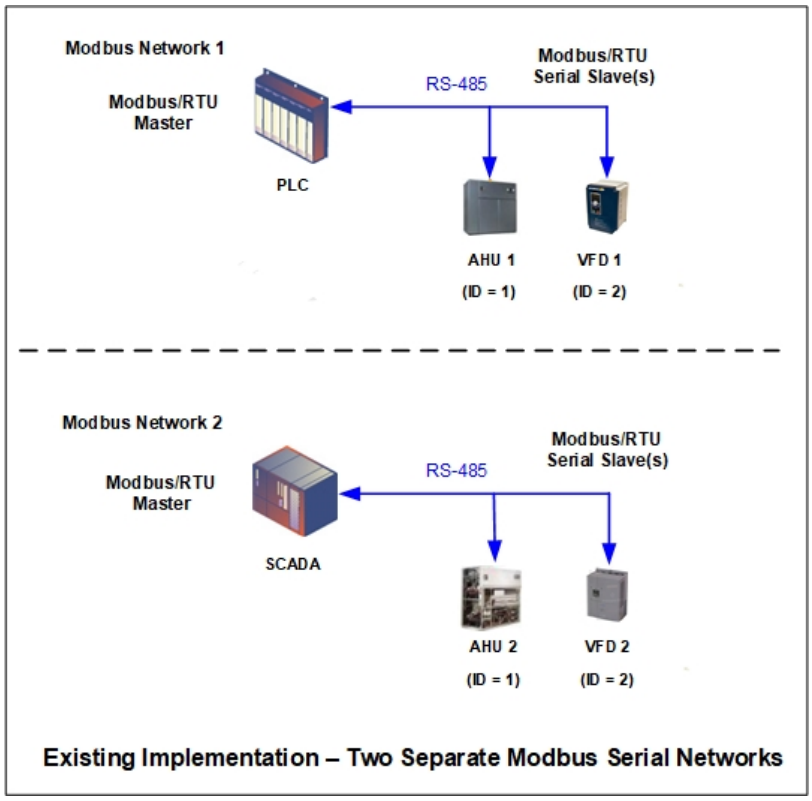
Requirements:

- Two serial Modbus networks, each with a controller and slave devices, need to be merged into one.
- Each controller needs access to all of the slave devices.
- To limit the engineering effort, it is desired to limit the scope of the project by:
  - Retaining the existing Modbus master program logic and slave device IDs.
  - Limiting the project to only adding the additional connectivity and corresponding control logic.

Solution:

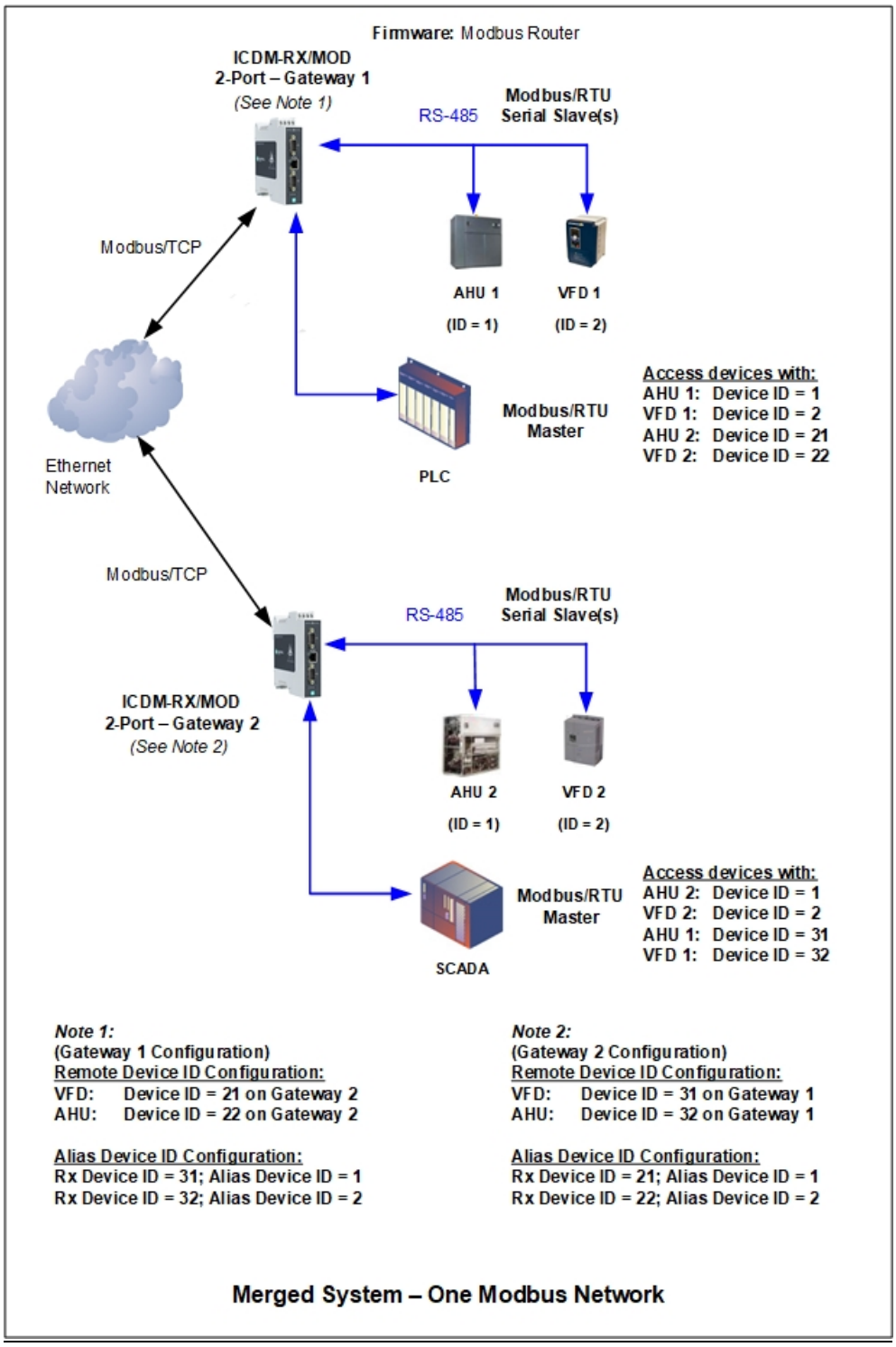
- Place a 2-Port ICDM-RX/MOD with Modbus Router between each controller and its corresponding slave devices.
- On each gateway, use Remote Device Routing and Alias Device ID functionality to provide access to the devices attached to the other gateway.
- Each slave device will then be accessed locally using its assigned device ID and from the remote controller using its aliased device ID.

Existing Systems:



Existing Implementation – Two Separate Modbus Serial Networks

Merged System:



6.5.5 Providing Modbus Connectivity between Separate Ethernet Networks

Requirements:

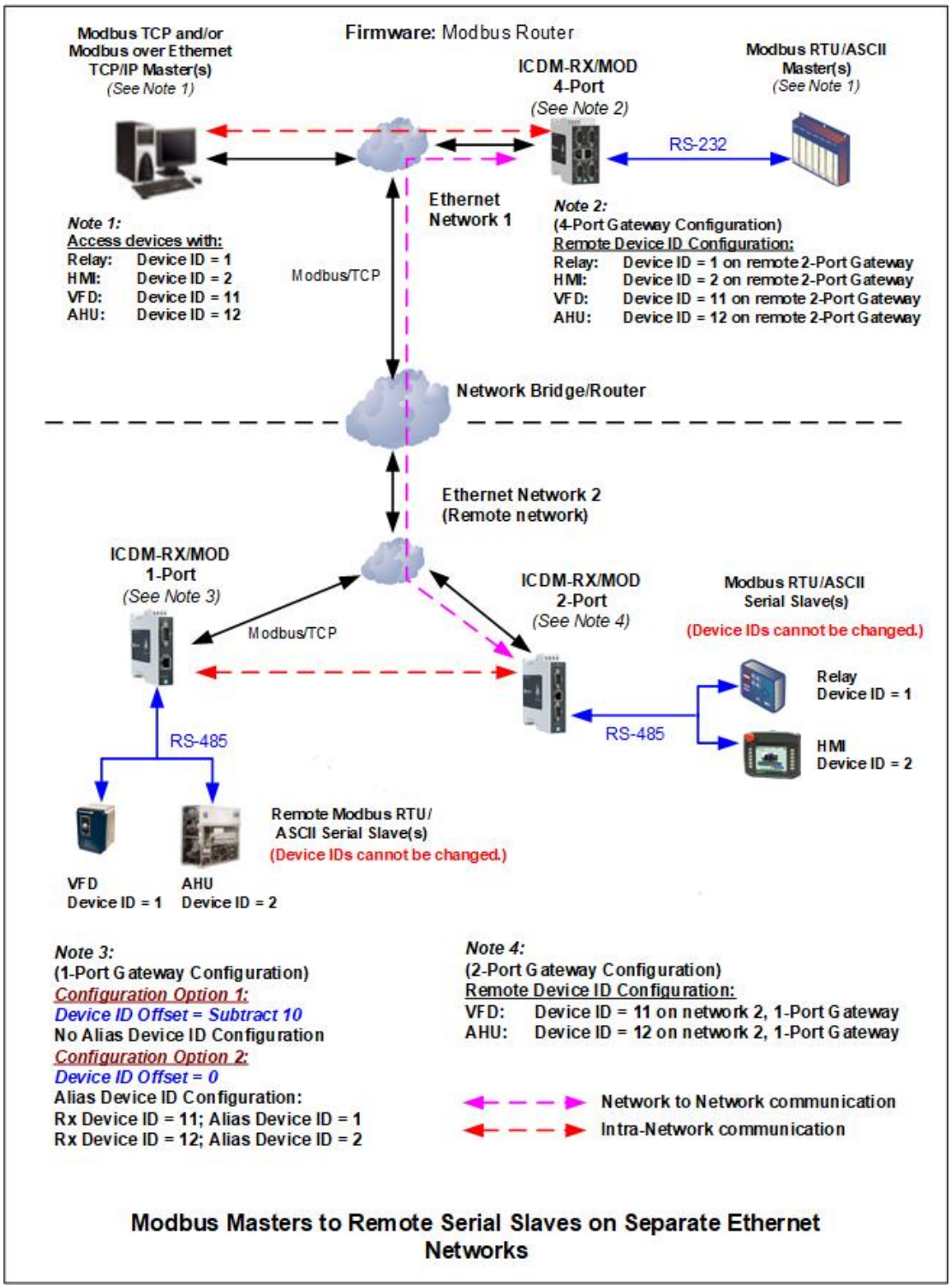
- Modbus Controllers on one Ethernet network need to connect to remote Modbus slave devices on a separate Ethernet network through an Ethernet Router.

Solution:

Using ICDM-RX/MOD gateways running Modbus Router firmware:

- Connect one or more ICDM-RX/MOD gateways to the Ethernet Network containing Modbus controllers. Connect any serial Modbus controllers to the ICDM-RX/MOD serial ports.
- Use one or more ICDM-RX/MOD gateways connected to the second Ethernet network to provide connectivity to the remote Modbus slaves.
- Use the Remote Device Routing and, if needed, the Alias Device ID functionality to provide connectivity from the Modbus masters to the remote Modbus slaves.

The solution is demonstrated in the following diagram on the next page.



Modbus Masters to Remote Serial Slaves on Separate Ethernet Networks