DeviceMaster UP 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 masters or slaves?

The DeviceMaster UP, 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.

Call or email for more information: 1.800.926.6876 | 763.957.6000 | IADSales@comtrol.com

1. Modbus Master-to-Master Connectivity

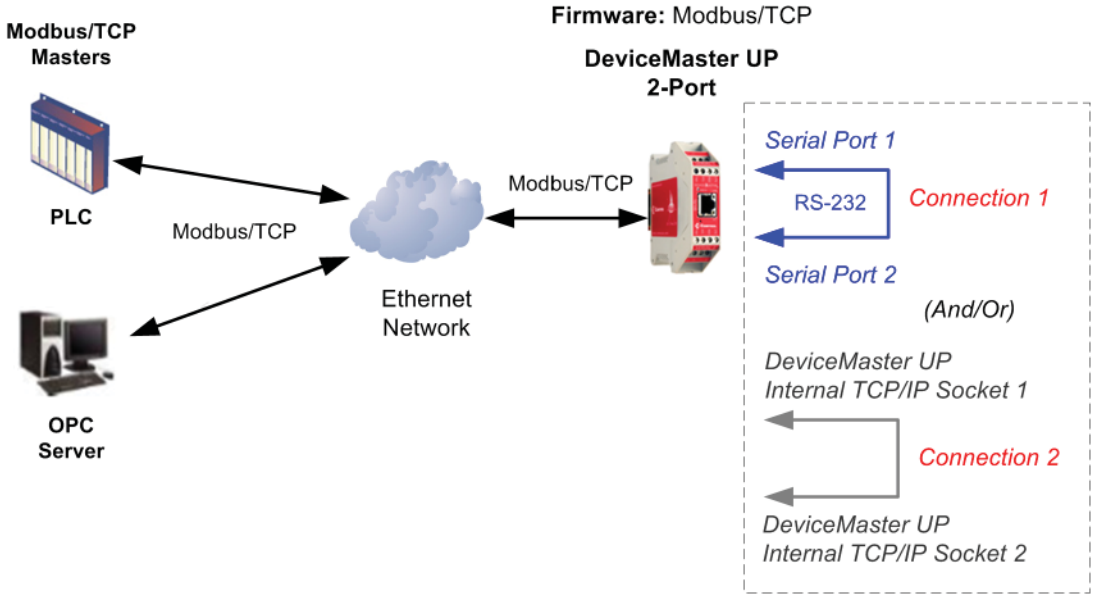
Multiple Modbus masters can communicate to each other through the DeviceMaster UP. Possible communication options include:

- 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.

1.1 Communicating Between Two Modbus Masters

1.1.1 Two Modbus/TCP Masters

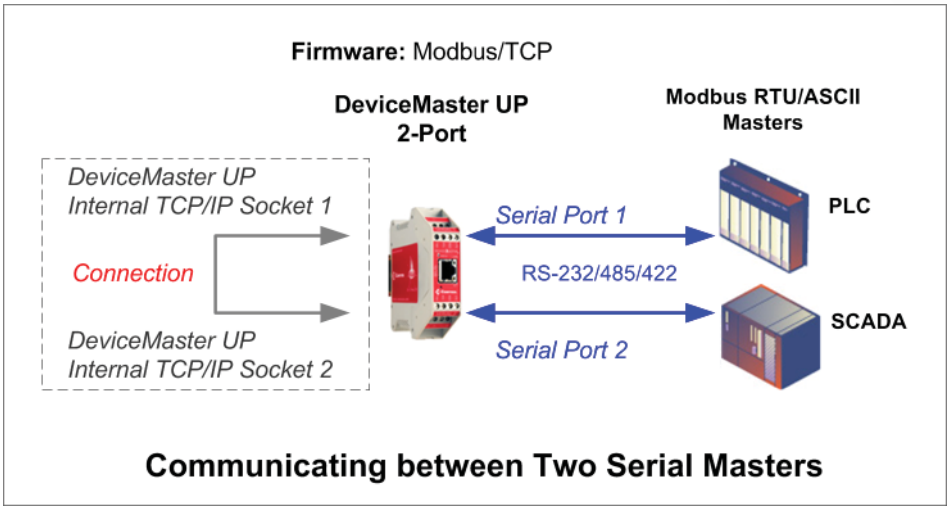
As shown in the following diagram, two Modbus/TCP masters can communicate to each other using either internal TCP/IP socket connections or connecting two serial ports.



Communicating between Two Modbus/TCP Masters

1.1.2 Two Serial Modbus Masters

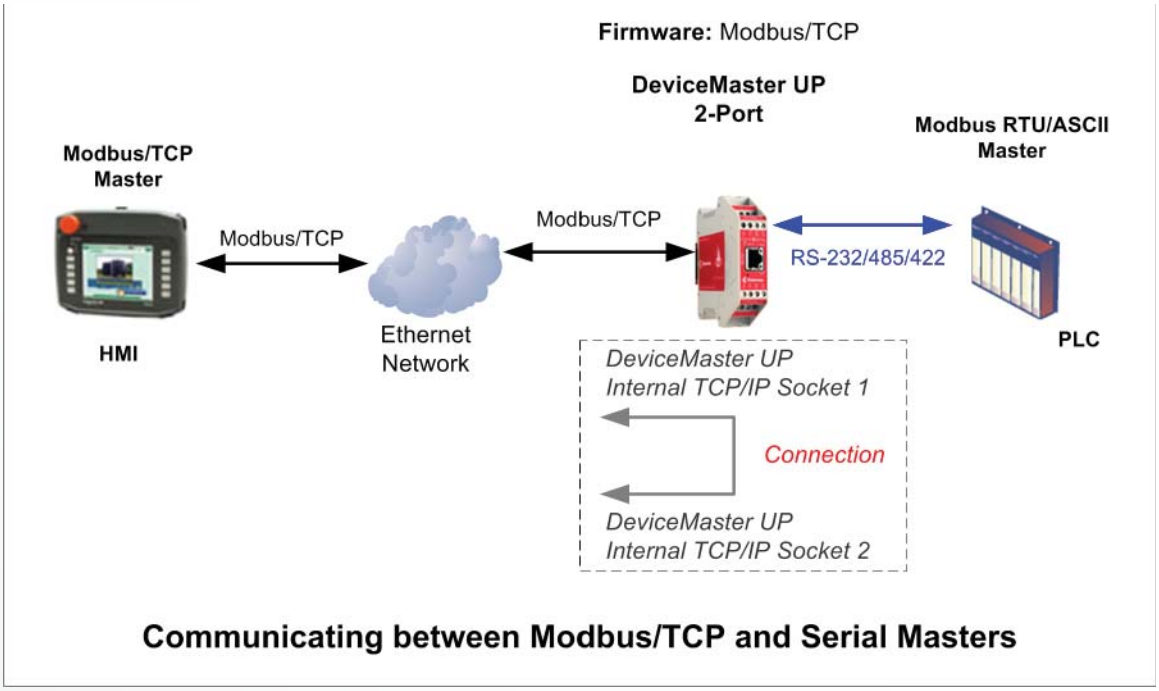
As shown in the following diagram, two serial Modbus masters can communicate to each other using an internal TCP/IP socket connection.



Note: Serial connections could be made with a 4-Port DeviceMaster UP.

1.1.3 Modbus/TCP and Serial Modbus Masters

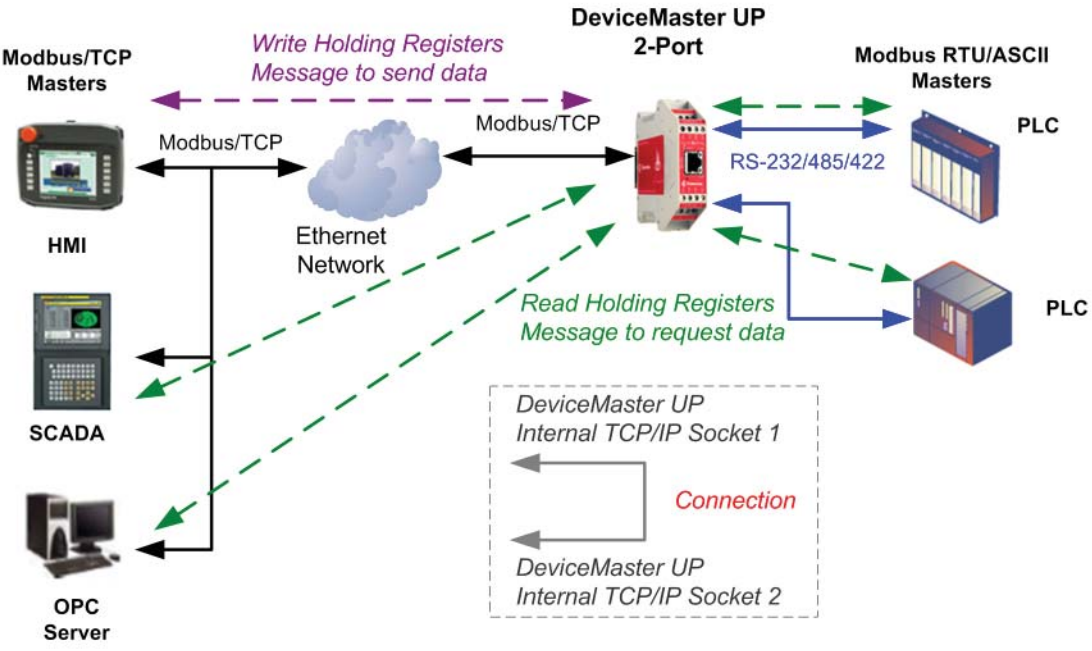
As shown in the following diagram, Modbus/TCP and serial Modbus masters can communicate to each other using an internal TCP/IP socket connection.



Note: Serial connections could be made with a 4-Port DeviceMaster UP.

1.1.4 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.



Method of Operation (Example):
1. HMI sends data via a Write Holding Registers message.
2. All other Modbus masters send Read Holding Registers message to request the data sent by the HMI.

Communicating from One Modbus Master to Multiple Masters

2. Modbus Slave-to-Slave Connectivity

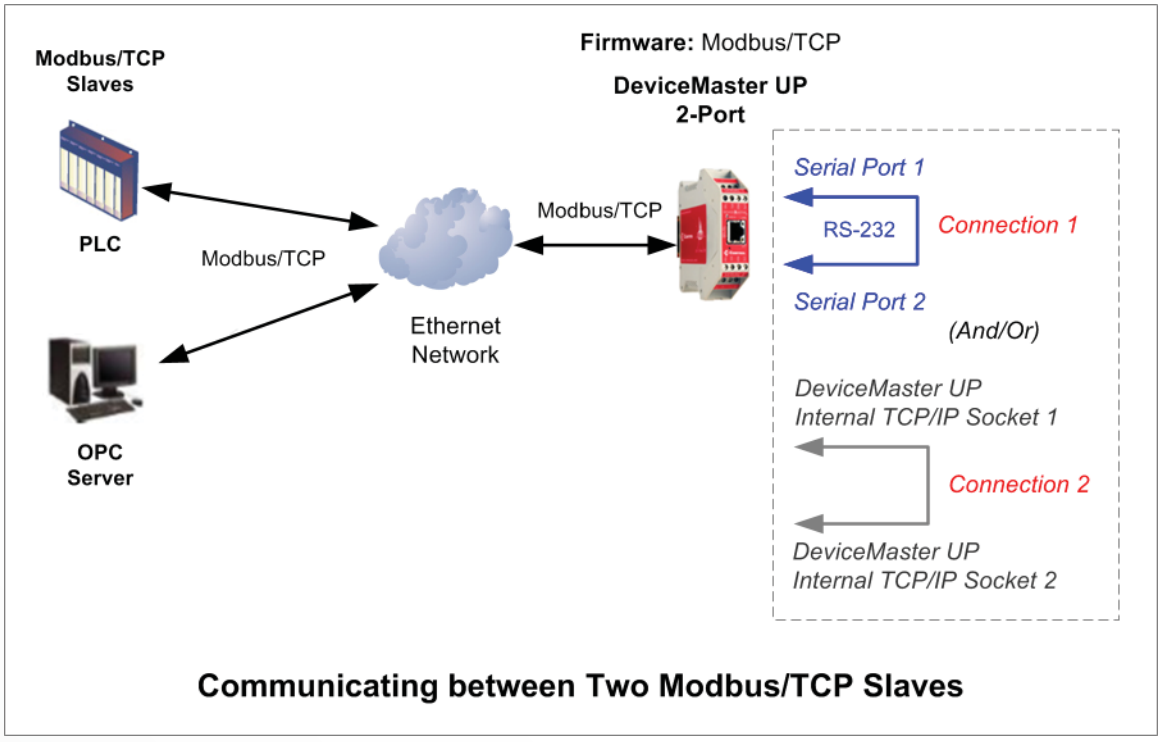
Modbus slaves can communicate to each other through the DeviceMaster UP. Possible communication options include:

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

2.1 Communicating Between Two Modbus Slaves

2.1.1 Two Modbus/TCP Slaves

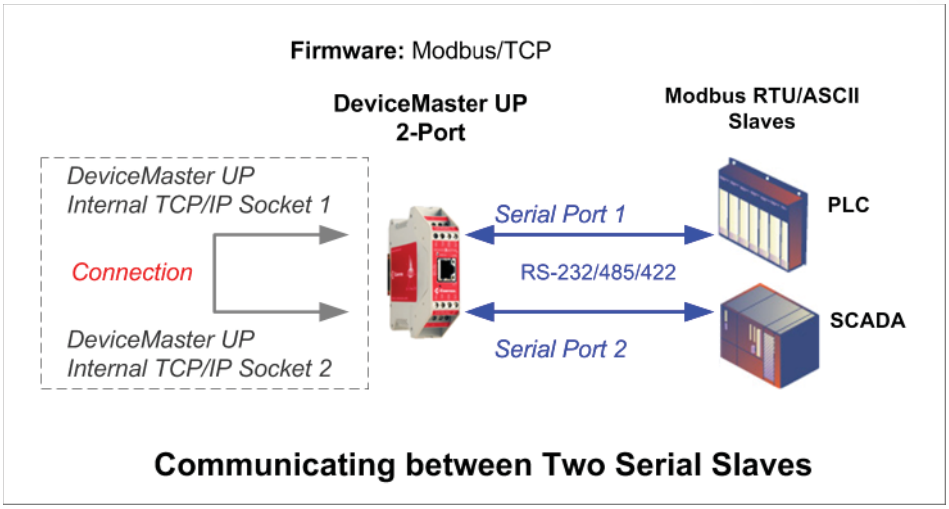
As shown in the following diagram, two Modbus/TCP slaves can communicate to each other using either internal TCP/IP socket connections or connecting two serial ports.



Communicating between Two Modbus/TCP Slaves

2.1.2 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.

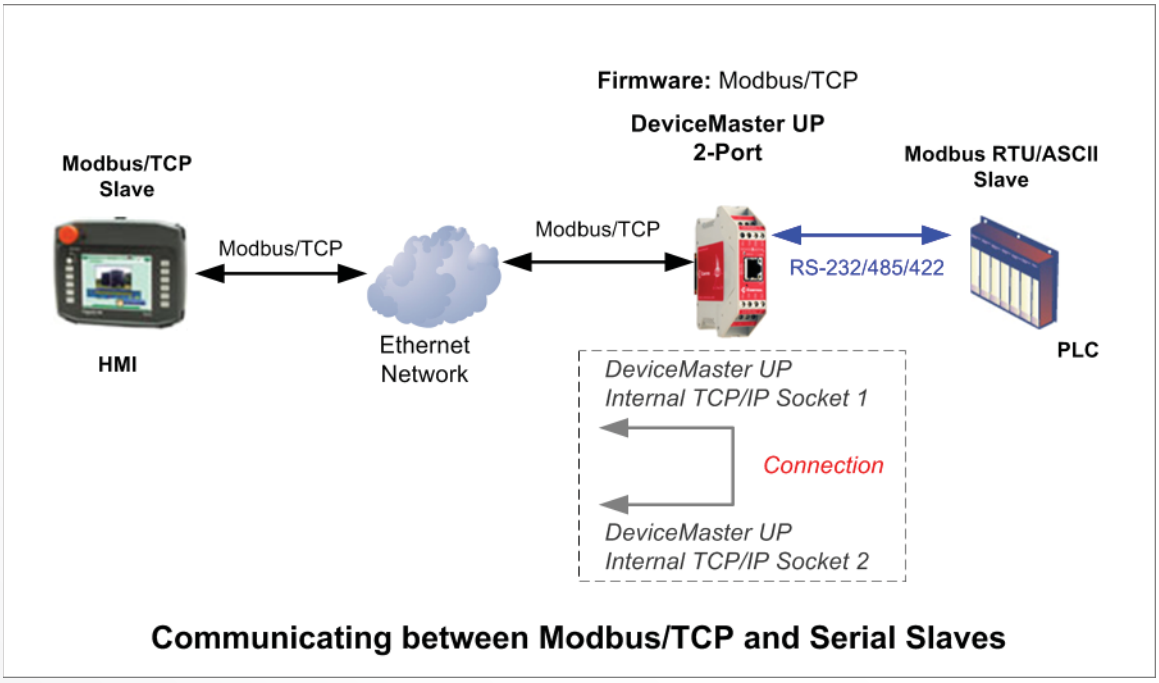


Communicating between Two Serial Slaves

Note: Serial connections could be made with a 4-Port DeviceMaster UP.

2.1.3 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.



Communicating between Modbus/TCP and Serial Slaves

Note: Serial connections could be made with a 4-Port DeviceMaster UP.

3. Creating Connections

Creating the serial and internal TCP/IP connections is performed via the embedded configuration web pages on the DeviceMaster UP.

3.1 Creating Serial Connections

Serial connections required a two step process:

- Configure the two serial ports with the same serial settings (baud rate, stop bits, etc).
- Physically connect the two serial ports with the appropriate serial cable or wiring.

The following settings display a serial port configuration used to connect two serial ports together:

	Port 1	Port 2
Serial Interface Name:	Controller Interface 1	Controller Interface 2
Serial Port Settings		
Mode:	RS-232	RS-232
Baud:	115200	115200
Parity:	none	none
Data Bits:	8	8
Stop Bits:	1	1
Flow:	none	none
DTR:	off	off
Rx Timeout Between Packets:	10	10
General Protocol Settings		
Serial Port Protocol:	Raw-Data	Raw-Data
Discard Rx Pkts With Errors:	yes	yes
Modbus Slave and Raw-Data Device Settings		
Response Timeout (ms):	N/A	N/A
Modbus Slaves Only		
Lost Device Search Enable:	N/A	N/A
Display Devices (all)		
Raw-Data Only		
Raw-Data Message Transfer Mode:	Data-Stream	Data-Stream
Cmd/Resp Age Time, Discard Responses After (sec):	N/A	N/A
Cmd/Resp Expected Responses Per Command:	N/A	N/A
Cmd/Resp Response To Modbus/TCP Based On:	N/A	N/A
Serial Packet ID Settings (Raw-Data Only)		
STX Rx Detect:	none	none
STX Rx:		
ETX Rx Detect:	none	none
ETX Rx:		
PLC Specific Settings		
STX Tx Append:	none	none
STX Tx:		
ETX Tx Append:	none	none
ETX Tx:		
Strip Rx STX/ETX Chars:	no	no

3.2 Creating Internal TCP/IP Socket Connections

Internal TCP/IP connections require only a configuration that connects the two sockets.

- Configure the first TCP/IP socket to “Listen” mode.
- Configure the second TCP/IP socket to “Connect-Always” mode and to connect to the listen port of the first socket on this gateway. (The IP Address of the gateway in this example is 10.0.0.102.)

The following settings display a TCP/IP socket configuration used to connect two sockets together:

	Socket 1	Socket 2
Ethernet Interface Name:	Controller Interface 1	Controller Interface 2
Device TCP Connection Configuration		
Enabled:	Yes	Yes
Listen:	Yes	No
Listen Port:	8000	8001
Connect To Mode:	Never	Connect-Always
Connect Port:	8010	8000
Connect IP Address:	0.0.0.0	10.0.0.102
Disconnect Mode:	Never	Never
Idle Timeout:	0	0
Message Transfer Settings		
Message Transfer Mode:	Data-Stream	Data-Stream
Cmd/Resp Response Timeout (ms):	N/A	N/A
Cmd/Resp Age Time, Discard Responses After (sec):	N/A	N/A
Cmd/Resp Expected Responses Per Command:	N/A	N/A
Cmd/Resp Response To Modbus/TCP Based On:	N/A	N/A
Device TCP Connection Status		
Remote Connection:	10.0.0.102:1024	10.0.0.102:8000
Socket Packet ID Settings		
Rx Timeout Between Packets:	10	10
STX Rx Detect:	none	none
STX Rx:		
ETX Rx Detect:	none	none
ETX Rx:		
PLC Specific Settings		
STX Tx Append:	none	none
STX Tx:		
ETX Tx Append:	none	none
ETX Tx:		
Strip Rx STX/ETX Chars:	no	no

4. Configuring the Modbus Interface

The DeviceMaster UP ports or sockets must be configured to Slave or Master mode depending on the type of controllers being connected.

4.1 Connecting Two Modbus Master Controllers

Both the Receive and Transmit channels on the DeviceMaster UP must be set to “Slave” mode to connect two Modbus masters.

The following settings display a “Slave” configuration:

Serial Modbus Master and Modbus/TCP Interface Settings		
Rx (To PLC) Transfer Mode:	Slave	Slave
Tx (From PLC) Transfer Mode:	Slave	Slave
Maximum Rx Data Packet Size:	246	246
Oversized Rx Packet Handling:	Truncate	Truncate
Rx MS Byte First:	no	no
Tx MS Byte First:	no	no
Disable Non-Filtered To PLC Rx Queue:	no	no
Disable Tx Sequence Number Check:	no	no
Modbus/TCP Master Rx/Tx Settings		
PLC IP Address:	N/A	N/A
PLC Device ID:	N/A	N/A
Master Rx Mode Only		
PLC Rx Data Address (Base 1):	N/A	N/A
Maximum PLC Update Rate (msec):	N/A	N/A
Use Maximum Sized Modbus Messages:	N/A	N/A
Master Tx Mode Only		
PLC Tx Data Address (Base 1):	N/A	N/A
PLC Tx Poll Rate (msec):	N/A	N/A
PLC Tx Poll Message Length (bytes):	N/A	N/A
Tx Sequence Number Syncing Enable:	N/A	N/A
PLC Tx Consumed Seq Nbr Address (Base 1):	N/A	N/A

Please note: *The Modbus/TCP Master settings do not apply to Slave mode.*

4.2 Connecting Two Modbus Slave Controllers

Both the Receive and Transmit channels on the DeviceMaster UP must be set to “Master” mode to connect two Modbus slaves. Compared to Slave mode, more settings must be configured to enable the DeviceMaster UP to operate as a master to both controllers. Please consult the Modbus/TCP User Guide for more details.

The following settings display a “Master” configuration:

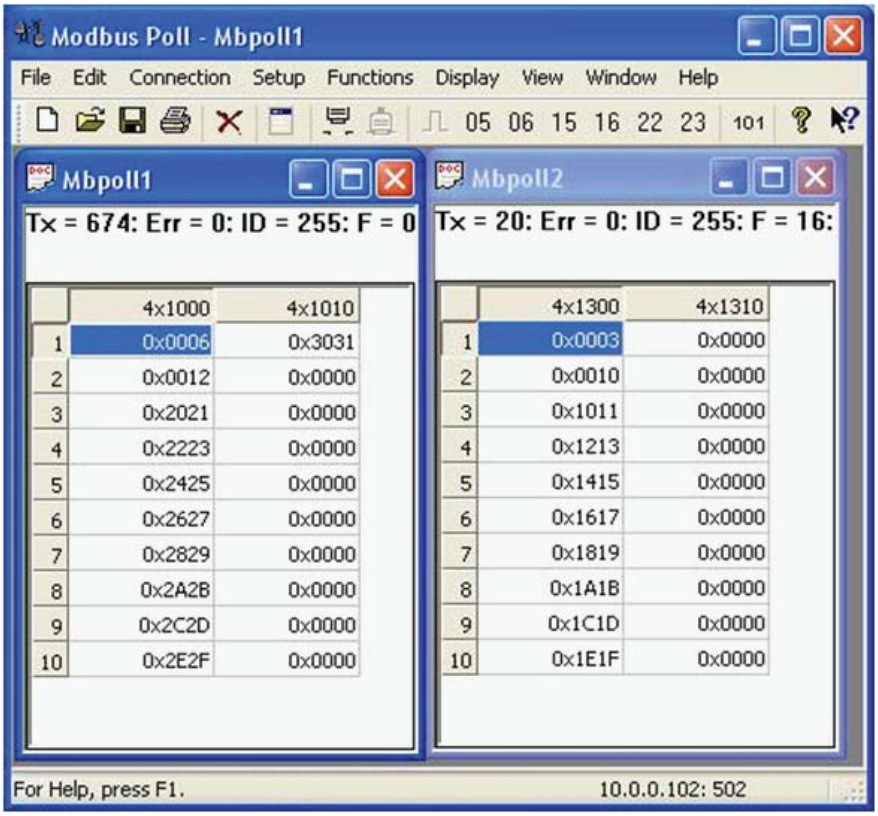
Serial Modbus Master and Modbus/TCP Interface Settings		
Rx (To PLC) Transfer Mode:	Master	Master
Tx (From PLC) Transfer Mode:	Master	Master
Maximum Rx Data Packet Size:	246	246
Oversized Rx Packet Handling:	Truncate	Truncate
Rx MS Byte First:	no	no
Tx MS Byte First:	no	no
Disable Non-Filtered To PLC Rx Queue:	no	no
Disable Tx Sequence Number Check:	N/A	N/A
Modbus/TCP Master Rx/Tx Settings		
PLC IP Address:	10.0.0.10	10.0.0.102
PLC Device ID:	1	2
Master Rx Mode Only		
PLC Rx Data Address (Base 1):	1	2001
Maximum PLC Update Rate (msec):	40	40
Use Maximum Sized Modbus Messages:	no	no
Master Tx Mode Only		
PLC Tx Data Address (Base 1):	21	2021
PLC Tx Poll Rate (msec):	200	200
PLC Tx Poll Message Length (bytes):	32	32
Tx Sequence Number Syncing Enable:	no	no
PLC Tx Consumed Seq Nbr Address (Base 1):	N/A	N/A

Please note: *Serial slave controllers connected to this gateway can be accessed by setting the “PLC IP Address” to that of the gateway.*

5. Controller-to Controller Examples

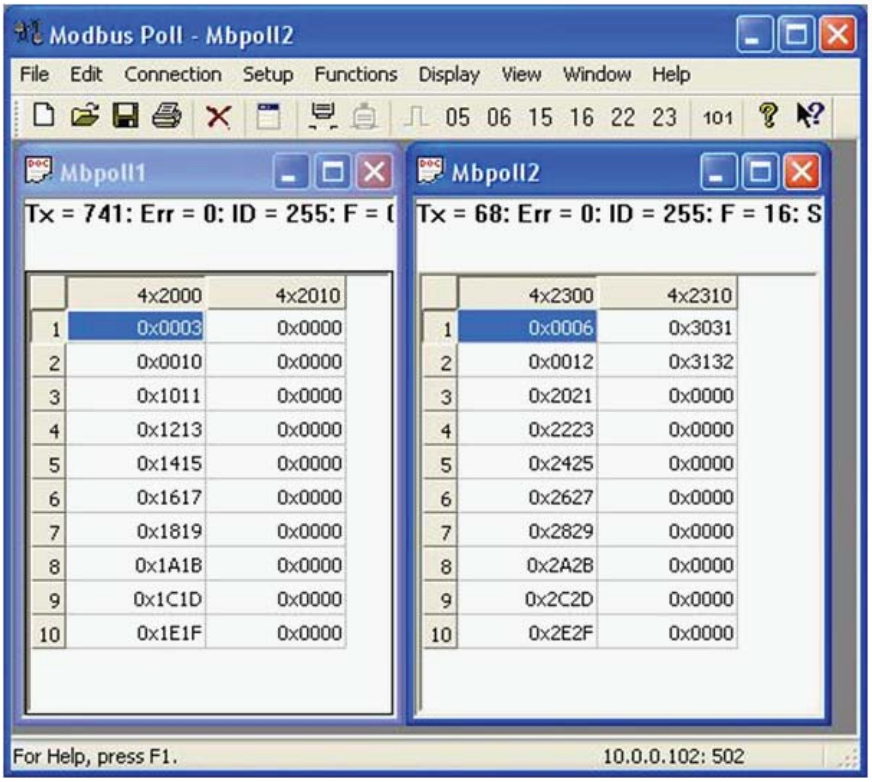
5.1 Two Modbus Masters

The following diagrams demonstrate two Modbus Poll instances communicating to each other through a serial port connection.



The above image displays Modbus Master 1 where:

- Device ID:
  - The device ID of 255 is used to access all serial Raw/ASCII ports.
  - The device ID of 254 is used to access TCP/IP socket ports.
  - If desired, these connections can be accessed by alternate device IDs via the Device ID Alias functionality.
- Address 1001 references serial port 1 received data holding registers. The Modbus master polls this location for received data from the other controller.
- Address 1301 references serial port 1 transmit data holding registers. The Modbus master writes to this location to send data to the other controller.
- The receive and transmit data format:
  - The first 16 bit word is the sequence number. This is incremented when new data is received and can optionally be incremented to indicate when to send data to the other controller.
  - The second word is the length in bytes.
  - The following words are the data. The data can be received and/or transmitted in either least significant (default) or most significant byte order.

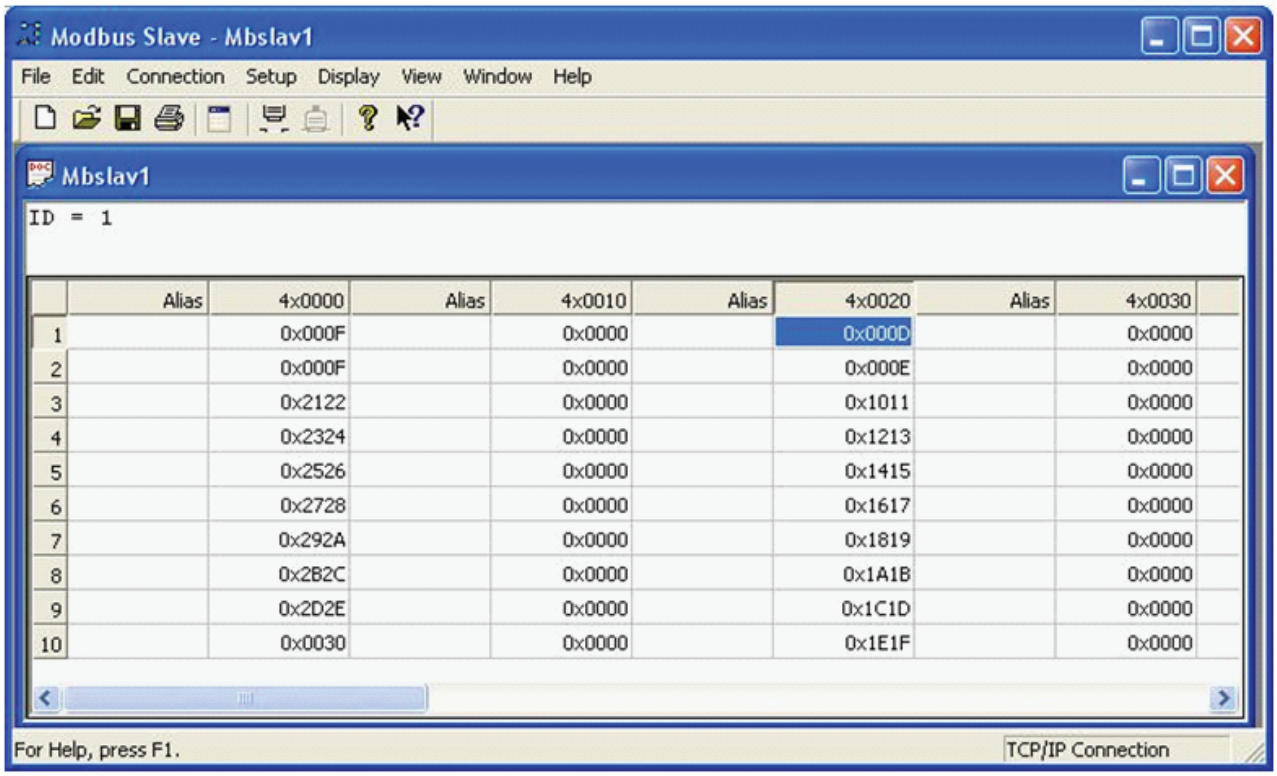


The above image displays Modbus Master 2 where:

- Device ID:
  - The device ID of 255 is used to access all serial Raw/ASCII ports.
  - The device ID of 254 is used to access TCP/IP socket ports.
  - If desired, these connections can be accessed by alternate device IDs via the Device ID Alias functionality.
- Address 2001 references serial port 2 received data holding registers. The Modbus master polls this location for received data from the other controller.
- Address 2301 references serial port 2 transmit data holding registers. The Modbus master writes to this location to send data to the other controller.
- The receive and transmit data format:
  - The first 16 bit word is the sequence number. This is incremented when new data is received and can optionally be incremented to indicate when to send data to the other controller.
  - The second word is the length in bytes.
  - The following words are the data. The data can be received and/or transmitted in either least significant (default) or most significant byte order.

5.2 Two Modbus Slaves

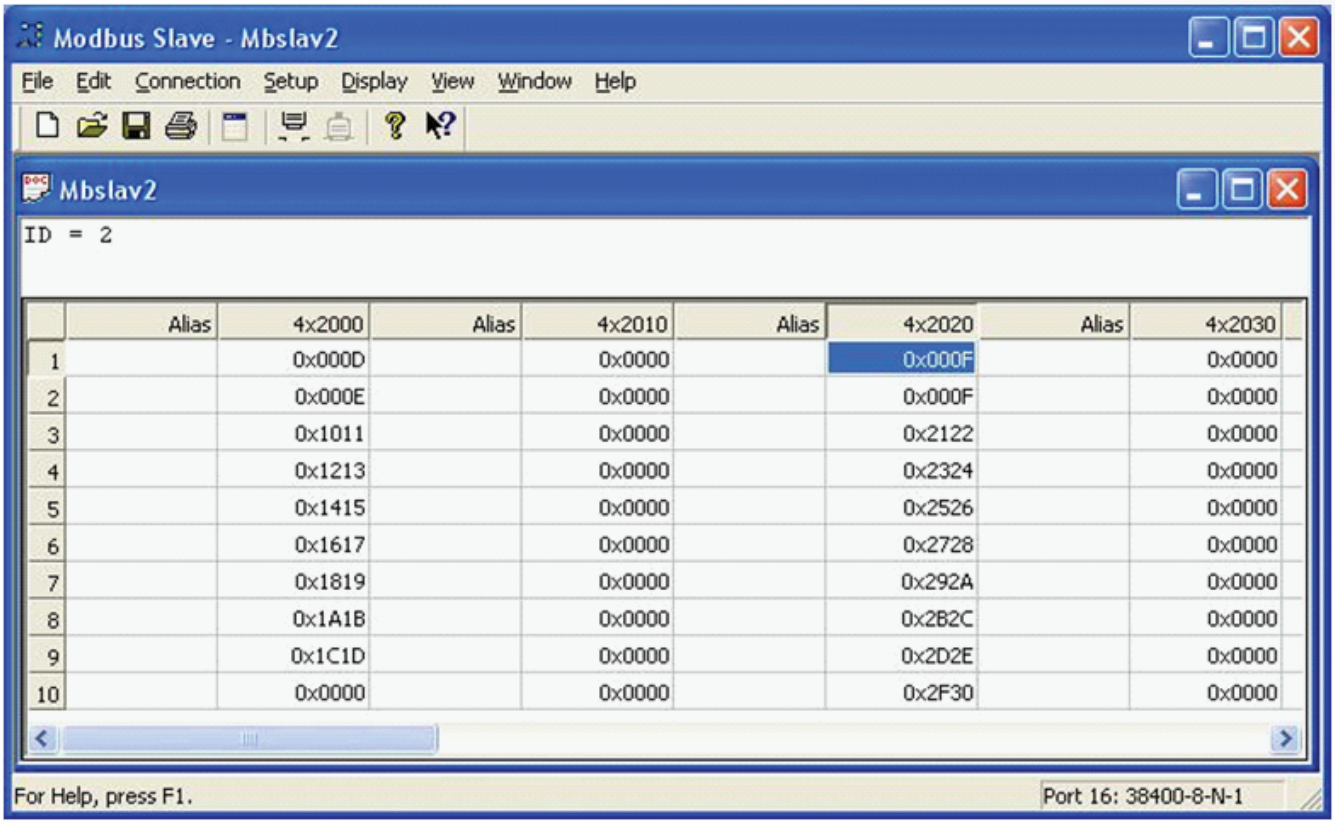
The following diagrams demonstrate two Modbus Slave instances communicating to each other through a TCP/IP socket port connection.



	Alias	4x0000	Alias	4x0010	Alias	4x0020	Alias	4x0030
1		0x000F		0x0000		0x000D		0x0000
2		0x000F		0x0000		0x000E		0x0000
3		0x2122		0x0000		0x1011		0x0000
4		0x2324		0x0000		0x1213		0x0000
5		0x2526		0x0000		0x1415		0x0000
6		0x2728		0x0000		0x1617		0x0000
7		0x292A		0x0000		0x1819		0x0000
8		0x2B2C		0x0000		0x1A1B		0x0000
9		0x2D2E		0x0000		0x1C1D		0x0000
10		0x0030		0x0000		0x1E1F		0x0000

The above image displays Modbus Slave 1 where:

- Device ID of 1 is the configured device ID.
- The receive address of 1 is the configured receive holding register address. The DeviceMaster UP will place the formatted received data here.
- The transmit address of 21 is the configured transmit holding register address. The DeviceMaster UP will look for the formatted transmit data here.
- The receive and transmit data format:
    - The first 16 bit word is the sequence number. The sequence number is incremented when new data is received and must be incremented to indicate when to transmit data to the other controller.
    - The second word is the data length in bytes.
    - The following words contain the data. The data can be received and/or transmitted in either least significant (default) or most significant byte order.



	Alias	4x2000	Alias	4x2010	Alias	4x2020	Alias	4x2030
1		0x000D		0x0000		0x000F		0x0000
2		0x000E		0x0000		0x000F		0x0000
3		0x1011		0x0000		0x2122		0x0000
4		0x1213		0x0000		0x2324		0x0000
5		0x1415		0x0000		0x2526		0x0000
6		0x1617		0x0000		0x2728		0x0000
7		0x1819		0x0000		0x292A		0x0000
8		0x1A1B		0x0000		0x2B2C		0x0000
9		0x1C1D		0x0000		0x2D2E		0x0000
10		0x0000		0x0000		0x2F30		0x0000

The above image displays Modbus Slave 2 where:

- Device ID of 2 is the configured device ID.
- The receive address of 2001 is the configured receive holding register address. The DeviceMaster UP will place the formatted received data here.
- The transmit address of 2021 if the configured transmit holding register address. The DeviceMaster UP will look for the formatted transmit data here.
- The receive and transmit data format:
    - o The first 16 bit word is the sequence number. The sequence number is incremented when new data is received and must be incremented to indicate when to transmit data to the other controller.
    - o The second word is the data length in bytes.
    - o The following words contain the data. The data can be received and/or transmitted in either least significant (default) or most significant byte order.