



GarrettCom™

Industrial Networking at Its Best™

MNS-DX Modbus/TCP

Technical Bulletin

Revision A

Overview

Major features of the MNS-DX Modbus/TCP functionality include:

- Simultaneous client and server operation, including support for multiple serial master and slave Modbus devices.
- Support for both the ASCII and RTU serial protocol variants.
- Modbus Plus gateway exceptions provide feedback to master devices when network configuration or communication errors occur.
- There are configurable timeouts on both the client and server side. This provides maximum flexibility for connecting to a wide variety of serial Modbus devices

1. Theory of Operation

1.1 Modbus

Magnum MNS-DX software supports client (master) and server (slave) modes of operation for the Modbus/TCP protocol as per the March 29, 1999 (Release 1.0) Open Modbus/TCP Specification.

1.1.1 Network Topologies

Figure 2-1-1 depicts an example of a Modbus/TCP network. Modbus devices (masters and slaves) are connected via serial lines to Magnum DX Serial Device Routers at the edge of the network. In addition, Modbus/TCP clients and servers may be connected directly to the IP network over an Ethernet link. The Modbus serial devices are connected to the DX units via RS-232 and/or RS-485 single or multi-drop interfaces.

The serial Modbus masters initiate requests to the slaves. These requests are encapsulated and forwarded by the Modbus/TCP client software to the appropriate Modbus/TCP server. At the server, the request is de-encapsulated, analyzed, and sent over the appropriate serial port to the serial Modbus slave. When the slave device responds, the response is encapsulated and sent back to the Modbus/TCP client, which in turn de-encapsulates and forwards the response to the Modbus master. Device tables are kept on each DX that describe the locally connected Modbus serial devices as well as how to reach each remote device.

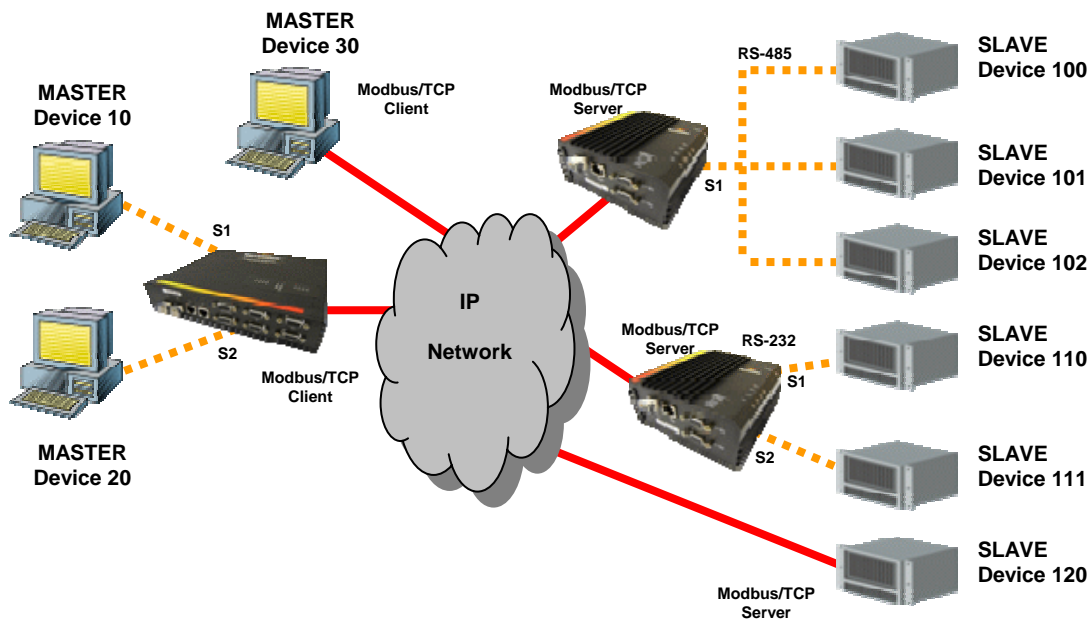


Figure 2-1-1. Example of a MODBUS/TCP Network

1.1.2 Serial Protocol Variants

For serial data, both the Modbus ASCII and the Modbus RTU protocol variants are supported.

Modbus ASCII (depicted in Figure 2-1-2) uses ASCII message encoding with a longitudinal redundancy check (LRC). Each message begins with a ':' character and ends with a CRLF character sequence.



Figure 2-1-2. Format of a Modbus ASCII Packet

Modbus RTU (depicted in Figure 2-1-3) uses binary message encoding with a cyclic redundancy check (CRC). Each message begins with a silent interval of at least 3.5 characters and ends with a similar silent interval.

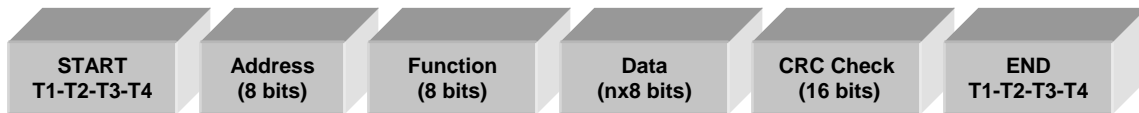


Figure 2-1-3. Format of a Modbus RTU Packet

1.1.3 Network Protocol

The Modbus/TCP format (depicted in Figure 2-1-5) strips the message framing and LRC/CRC from the normal Modbus packet and adds a Modbus/TCP header consisting of a 2-byte Transaction ID (set by the client and echoed by the server), a 2-byte Protocol ID (always 0-0), and a 2-byte length. The device address byte (now referred to as the unit identifier) and the function byte are preserved and are followed by a variable amount of data. This information is then delivered as the payload of a TCP/IP packet. The Modbus LRC/CRC is not included because it is redundant with the CRC provided by the link layer (i.e. Ethernet).

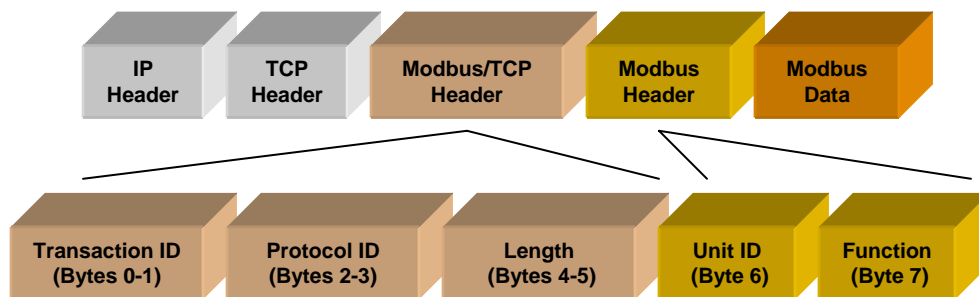


Figure 2-1-5. Format of a Modbus/TCP Packet

1.1.4 Exception Handling

The Modbus/TCP client and server on a Magnum DX Serial Device Router can optionally generate and forward Modbus exception codes when certain communication or configuration errors occur. Specifically, the client will generate a GATEWAY PATH UNAVAILABLE exception message (exception code 0x0A) and pass it back to the master device if a remote address has not been configured for the destination device. The server will generate a similar message if a local device entry has not been configured for the destination device address. The message is sent to the client, which then forwards the exception to the Modbus master device.

The server will also generate a GATEWAY TARGET DEVICE FAILED TO RESPOND exception message (exception code 0x0B) when the destination device does not respond to a request within a user-configured interval. This message is sent to the client, which then forwards the exception to the Modbus master device.

1.1.5 TCP Connection Handling

TCP connection handling performed by a Magnum DX complies with the implementation guidelines spelled out in Appendix A of the Open Modbus/TCP Specification.

When the Modbus/TCP client software receives a request from an attached serial Modbus master, it analyzes the packet and determines the destination device address. It checks to see if it already has an open TCP connection for the destination. If not, the client attempts to open a new TCP connection to the appropriate Modbus/TCP server. Once a connection is established, the request message is sent and the client waits for a response. After the response is received, it is forwarded back to the master.

After the transaction is complete, the TCP connection remains open in anticipation of a subsequent request. If another request is not made within the user-configured idle time, the TCP connection is closed and will be re-opened when a new request is received. The client may also be configured so that it immediately makes a connection for a configured device and keeps that connection open indefinitely. This mode eliminates the latency associated with the TCP handshake required when making the TCP connection for the initial request.

If a response is not received, the Modbus/TCP client will time-out after a user-configured interval. After a time-out, the TCP connection is closed to eliminate the possibility of receiving an unexpected late response. In addition, the GATEWAY TARGET DEVICE FAILED TO RESPOND (exception code 0x0B) exception message is sent to the Modbus Master that can then make the decision on whether or not to retry. If the client is configured to hold connections open indefinitely, a new connection will be established with the remote server immediately following the timeout. Otherwise, the client waits for the next Modbus request before re-opening the connection.

The Modbus/TCP server process always listens for connections on TCP port 502.

2. User Interface

This section describes the various user interfaces provided by the MNS-DX software managing the Modbus/TCP functionality.

2.1 Web Interface

This subsection describes the structure of the web pages used to manage the Modbus/TCP functionality.

2.1.1 Serial : Modbus : Global Settings

This page allows the user to configure global Modbus/TCP parameters.

2.1.1.1 Web Interface Mockup

Serial : Modbus : Global Settings

Management Address:	<i>100</i>
----------------------------	------------

[Reset Settings] [Apply Settings]

2.1.1.2 Description

Management Address The Modbus/TCP unit identifier used to communicate with the DX's management memory map.

2.1.2 Serial : Modbus : Local Masters

This page allows the user to configure local serial Modbus Masters that will act as Modbus/TCP clients.

2.1.2.1 Web Interface Mock-up

Serial : Modbus : Local Masters

Add Device

Port ID	Protocol Variant	Priority (DiffServ)	Supports Exceptions
S1 V	RTU V	Default V	Yes V

[Reset Settings] [Apply Settings]

Existing Devices

Port ID	Protocol Variant	Priority (DiffServ)	Exception Support	Delete
S1	RTU V	Default V	Yes V	<input type="checkbox"/>
S2	ASCII V	Expedited V	Yes V	<input type="checkbox"/>

[Reset Settings] [Apply Settings]

2.1.2.2 Description

This Form is used to define the directly-connected Modbus Master devices. The parameters are defined as follows:

- Port ID** identifier of the serial port to which the device is connected
- Protocol Variant** specifies the protocol variant spoken by the device
- RTU: messages are binary encoded with CRC and begin with a silent interval of 3.5 characters (Default)
 - ASCII: messages are ASCII encoded with LRC and begin with a ‘:’ character and end with a CRLF sequence
- Priority (DiffServ)** each IP packet generated by this device will be assigned a DiffServ marking based on the priority set by the user. The priorities are:
- Default: Best Effort Service (Code Point 0) (Default)
 - Expedited: Expedited Forwarding (Code Point 0x2E) (RFC2598)
- Exception Support** specifies whether or not the attached master understands Modbus exception messages. In some cases, Modbus devices do not support the exception function codes and will be confused by them if received. This option allows you to disable exception forwarding to the master device.

2.1.3 Serial : Modbus : Local Slaves

This page allows the user to configure local serial Modbus Slaves that will be accessible via the Modbus/TCP server.

2.1.3.1 Web Interface Mockup

Serial : Modbus : Local Slaves

Add Device

Port ID	Device Address	Protocol Variant	Priority (DiffServ)	Response Timer (msec)
S1 V		RTU V	Default V	

[Reset Settings] [Apply Settings]

Existing Devices

Port ID	Device Address	Protocol Variant	Priority (DiffServ)	Response Timer (msec)	Delete
S1	10	RTU V	Default V	500	<input type="checkbox"/>
S2	24	ASCII V	Expedited V	500	<input type="checkbox"/>

[Reset Settings] [Apply Settings]

2.1.3.2 Description

This Form is used to define the directly connected Modbus devices. The parameters are defined as follows:

- Port ID** identifier of the serial port to which the device is connected
- Device Address** Modbus/TCP unit identifier (1-247) assigned to the device
- Protocol Variant** specifies the protocol variant spoken by the device
- RTU: messages are binary encoded with CRC and begin with a silent interval of 3.5 characters (Default)
 - ASCII: messages are ASCII encoded with LRC and begin with a ‘:’ character and end with a CRLF sequence
- Priority (DiffServ)** each IP packet generated by this device will be assigned a DiffServ marking based on the priority set by the user. The priorities are:
- Default: Best Effort Service (Code Point 0) (Default)
 - Expedited: Expedited Forwarding (Code Point 0x2E) (RFC2598)
- Response Timer** the amount of time to wait for a response from this device before giving up and sending back a Modbus exception message

2.1.4 Serial : Modbus : Remote Slaves

This page allows the user to configure the forwarding table used to map Modbus slave device addresses to remote IP addresses.

2.1.4.1 Web Interface Mockup

Serial : Modbus : Remote Slaves

Add Device

Device Address	Remote Address	Idle Timer (sec)	Response Timer (msecs)
		30	1000

[Reset Settings] [Apply Settings]

Existing Devices

Device Address	Remote Address	Idle Timer (sec)	Response Timer (msecs)	Delete
16	192.168.1.2	30	1000	<input type="checkbox"/>
24	192.168.3.5	30	1000	<input type="checkbox"/>

[Reset Settings] [Apply Settings]

2.1.4.2 Description *msecs*

This Form is used to add a mapping between a Modbus device address and the IP address of a remote Modbus/TCP server. The parameters are as follows:

- Device Address** Modbus/TCP unit identifier (0-255) assigned to the remote device
- Remote Address** the IP address of the remote Modbus/TCP server
- Idle Timer** the TCP connection for this device is torn down if the idle time (time between messages) exceeds the value specified here. This parameter allows multiple successive requests to the same remote device to re-use a single TCP connection, thereby reducing latency. As a special case, if this value is set to 0, a TCP connection is immediately made to the remote (i.e. the client does not wait for a request) and it is always kept open. This special mode eliminates the connection latency associated with the initial Modbus request.
- Response Timer** the client will wait this amount of time before giving up on a request. If the client times out, it closes down the current TCP connection for the remote device

2.1.5 Serial : Modbus : Connections

This page shows the user the status of all active Modbus/TCP connections.

2.1.5.1 Web Interface Mockup

Serial : Modbus : Connections

Connection Mode	Local Address	Local Port	Remote Address	Remote Port	Requests	Responses	Tx Octets	Rx Octets	Delete
Client	192.168.1.2	1024	192.168.1.2	502	10	10	160	320	<input type="checkbox"/>
Server	192.168.3.5	502	192.168.3.5	1026	15	14	140	90	<input type="checkbox"/>

[Refresh] [Apply Settings]

2.1.5.2 Description

This Table contains all of the active Modbus/TCP connections in the system and the traffic statistics associated with each connection. The user may also manually disconnect any TCP connection by selecting the appropriate Delete checkbox and pressing the “Apply Settings” button. The parameters for each connection are as follows:

Connection Mode indicates whether this connection was established in client or server mode

Local Address the IP address of the local Modbus/TCP client/server

Local Port the TCP port of the local Modbus/TCP client/server

Remote Address the IP address of the remote Modbus/TCP client/server

Remote Port the TCP port of the remote Modbus/TCP client/server

Requests number of requests generated (if client) or number of requests received (if server)

Responses number of responses received (if client) or number of responses generated (if server)

Tx Octets total number of octets transmitted on this connection

Rx Octets total number of octets received on this connection

2.2 XML Configuration

The following is a “template” for the XML configuration data used to specify Modbus.

```
<Serial>
  <Modbus>
    <ManagementAddress/>
    <MasterDeviceTable>
      <MasterDeviceEntry>
        <PortID/>
        <Protocol/>
        <Priority/>
        <Exceptions/>
      </MasterDeviceEntry>
    </MasterDeviceTable>
    <SlaveDeviceTable>
      <SlaveDeviceEntry>
        <PortID/>
        <DeviceAddress/>
        <Protocol/>
        <Priority/>
        <ResponseTimer/>
      </SlaveDeviceEntry>
    </SlaveDeviceTable>
    <RemoteDeviceTable>
      <RemoteDeviceEntry>
        <DeviceAddress/>
        <RemoteAddress/>
        <IdleTimer/>
        <ResponseTimer/>
      </RemoteDeviceEntry>
    </RemoteDeviceTable>
  </Modbus>
</Serial>
```

2.3 Modbus Management Memory Map

The Modbus management memory map is a set of Modbus holding registers associated with the DX device itself. The first section of the map is identical to the map used by the Magnum 6K products while the second section adds registers for monitoring the system’s serial ports. The memory map is defined in a separate document.

End