

# **Modbus ASCII Serial / Modicon Serial Device Driver Guide**

# Table of Contents

## Modbus ASCII Serial / Modicon Serial Device Driver Guide 1-1

<b>1.</b>	<b>Modbus ASCII Serial Device Communications</b>	<b>2</b>
1.1	Introduction to Modbus ASCII Serial .....	2
1.1.1	Modbus ASCII Serial: RS-232, RS-422 and RS485 .....	2
1.1.2	Wiring and Cabling Requirements.....	3
1.1.3	Ease of Use: Parameters .....	4
1.1.4	Redundant Comports .....	5
1.1.5	Modbus Protocols.....	5
1.1.5.1	Modbus ASCII .....	5
1.1.5.2	Modbus RTU .....	5
1.1.5.3	Modbus Ethernet / TCP/IP .....	5
1.2	Configure Modicon / Modbus device .....	6
1.3	Serial Comport Properties .....	7
1.3.1	Comport Number .....	7
1.3.2	Description.....	7
1.3.3	Baud Rate.....	8
1.3.4	Data Bits .....	8
1.3.5	Stop Bits .....	8
1.3.6	Parity.....	8
1.3.7	Scan Time .....	9
1.3.8	Timeout.....	9
1.3.9	Retry Count .....	9
1.3.10	Auto Recover Time.....	10
1.3.11	Hand Shake RTS.....	10
1.3.12	Hand Shake DTR .....	11
1.3.13	Backup Port .....	11
1.4	Device Properties – Modicon ASCII .....	12
1.4.1	Device Name .....	13
1.4.2	Description.....	13
1.4.3	Unit Number .....	13
1.4.4	Device Type.....	14
1.5	Configure a Tag.....	14

July 30, 2005

# 1. Modbus ASCII Serial Device Communications

---

## 1.1 Introduction to Modbus ASCII Serial

The main advantage of the Modbus ASCII protocol is that it allows time intervals of up to one second to occur between characters without causing an error. This is beneficial for communications which have long delays. For example satellites technology routinely sees 2 second delays.

WebAccess SCADA Node provides a Modbus master interface using Modbus ASCII protocol for communicating with Modbus slave devices. Slave devices include AEG, Modicon, GE Fanuc and many others.

The Modicon driver accesses real-time data and control automation equipment with Modbus ASCII protocol.

Modbus is a "De-facto" standard for communications. Modbus is an "open" communications protocol designed for industrial control and monitoring applications. Programmable Logic Controllers (PLC), Single Loop and Multi-Loop Controllers, Remote Terminal Units (RTU, Distributed control Systems (DCS), computers, shop floor operator panels and other devices can communicate throughout plants and substations via Modbus ASCII, RTU or Modbus Ethernet network.

Especially for connection of SCADA and HMI systems to intelligent operator panels, PLCs and controllers, Modbus became a de-facto standard. Many automation devices support the Modbus protocol in Serial Modbus (ASCII and RTU) and Modbus Ethernet.

Modbus ASCII usually is not used to move pure ASCII string files. For example, it is not used to move JAN-16-1994. In Modbus ASCII, each eight-bit byte, in a message, is sent as two ASCII characters.

### 1.1.1 Modbus ASCII Serial: RS-232, RS-422 and RS485

The WebAccess Modicon Modbus Device Driver can communicate with either RS-232, RS-422 or RS-485 communications. This aspect of communications is device-limited.

The computer communication port must be designed for use with the Windows 32-bit operating system. If a RS-232 port is used, then the port must contain a FIFO (First-In, First-Out) buffer. If an RS-422 or 485 port is

used, then the hardware card must contain the appropriate communication chip for Windows use, such as the Intel 16C850.

Modicon's Modbus network is a single master, multi-drop network, which supports up to 247 slave devices. The preferred physical layer for the Modbus network is four- wire RS-422/485 asynchronous serial communications. Three-wire RS-232 communications may be required in some installations. In this case, additional hardware is necessary. Modems can be used for communication over long distances or to provide multi-drop operation using RS-232 devices.

WebAccess can scan every 100 milliseconds over serial connections limited only by the PLC, Controller or RTU and the connection. Although typical scan rates for Modbus ASCII are greater than 1 second, with 5 to 30 seconds being very common.

### **1.1.2 Wiring and Cabling Requirements**

The Modbus protocol allows standard RS-232 and RS-422 communication formats.

The proper cable type for each is device-specific and application-specific. Refer to the hardware manuals accompanying your device for the proper cabling required.

Modicon and Square-D hardware devices can use a standard pre-built cable available from Modicon or Square-D distributors. The part number of this cable is 990NAA26320. It is a RS-232 cable designed for programming use, however it can also be used to communicate with WebAccess.

The standard RS-232 cable configuration is given below.

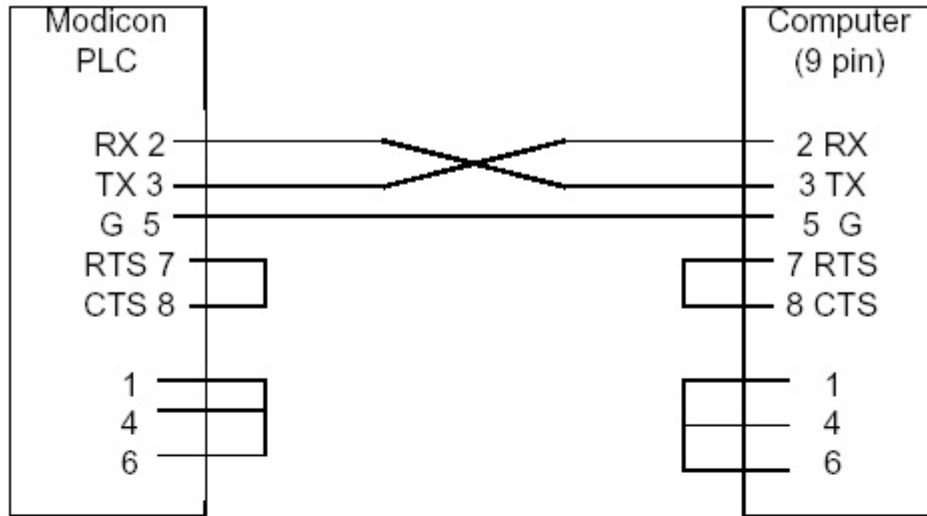


Figure 1 RS232 - Modicon 984 Cabling Diagram (9 pin COM Port)

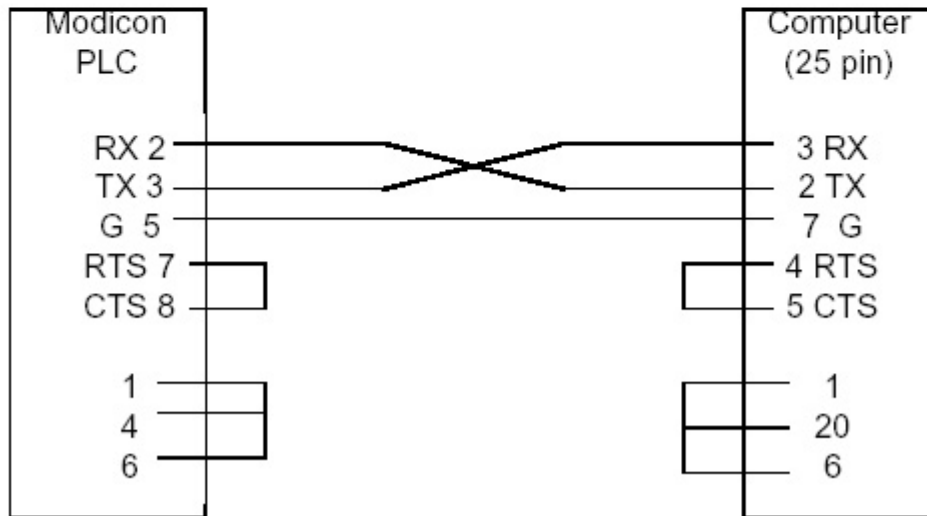


Figure 2 RS232 - Modicon 984 Cabling Diagram (25 pin COM Port)

### 1.1.3 Ease of Use: Parameters

Like all Genuine WebAccess drivers, object-oriented "parameters" guide novice users with pre-built templates containing typical addresses.

AI is an Analog Input (30001 to 39999 range of addresses). These are typically read only numbers from the PLC.

AO is an Analog Output (40001 to 49999 range of addresses). These are typically values written to the PLC by operators and programs. Setpoints, Outputs, alarm limits are examples.

DI is a Digital Input (00001 to 09999 range of addresses). These are typically read only statuses (On, Off, True, False, etc) from the PLC.

DO is a Digital Output (10001 to 19999 range of addresses). These are typically values written to the PLC by operators and programs. On/ OFF, RUN/STOP are examples.

Users can select a parameter type, and then modify the address to the correct register in order to build a tag.

### **1.1.4 Redundant Comports**

WebAccess supports redundant Comports. Two Serial comports can be used, the second acts as a backup to the first.

Modbus Ethernet can also be backed up with a Serial connection (and vice versa).

### **1.1.5 Modbus Protocols**

#### **1.1.5.1 Modbus ASCII**

This is an option to the Modicon protocol of the WebAccess Modicon serial device driver (set on the Device Property page, Use ASCII Protocol).

In Modbus ASCII, each eight-bit byte, in a message, is sent as two ASCII characters. The main advantage is that it allows time intervals of up to one second to occur between characters without causing an error.

#### **1.1.5.2 Modbus RTU**

This is the default protocol of the WebAccess Modicon serial device driver. In Modbus RTU, each eight-bit byte, in a message, contains two four-bit hexadecimal characters. The main advantage of Modbus RTU is the greater character density allows better data throughput than Modbus ASCII for same baud rate.

WebAccess can scan every 100 milliseconds over serial connections limited only by the PLC, Controller or RTU and the connection.

Most modern Modbus serial devices use Modbus RTU.

#### **1.1.5.3 Modbus Ethernet / TCP/IP**

The Modbus Ethernet Driver is described in another User Guide.

Modicon's Modbus Ethernet network is a single master, multi-drop network, which supports up to 247 slave devices. The preferred physical layer for the Modbus Ethernet network TCP/IP over Ethernet, although any TCP/IP network

connection is supported including the Internet, WANs and LANs. A single IP address can support up to 255 devices. Serial communications can be "encapsulated" into TCP/IP packets using Modbus Serial-to-Ethernet gateways.

---

## 1.2 Configure Modicon / Modbus device

The steps, in summary, are:

1. Start Internet Explorer **Web Browser**.
2. Enter IP address of the **Project Node**.
3. Use **WebAccess Configuration**.
4. Open or Create a **Project**.
5. Configure a **SCADA node** (the PC that will connect to the automation hardware).
6. Configure a **Comport** for the SCADA Node that is a **Serial type Comport**.
7. Configure Baud Rate, Data Bit, Stop Bits and Parity to match those in the PLC. All PLCs on this Comport must use the same settings.
8. Configure a **Modicon Device** (determines the communications Protocol or Device Driver) using **Add Device**. This will open the Device Properties page.
9. Select **USE ASCII PROTOCOL** on the Device Properties page.
10. Use **Add Tag** or **Add Block** to create tags.
11. Select a Parameter (AI, AO, DI, DO) to match the type of data to be read (Analog Input, Analog Output, Digital Input, Digital Output). The Address of the data must match the Parameter Type:

AI is an Analog Input (30001 to 39999 range of addresses).  
These are typically read only numbers from the PLC.

AO is an Analog Output (40001 to 49999 range of addresses).  
These are typically values written to the PLC by operators and programs. Setpoints, Outputs, alarm limits are examples.

DI is a Digital Input (00001 to 09999 range of addresses).  
These are typically read only statuses (On, Off, True, False, etc) from the PLC.

DO is a Digital Output (10001 to 19999 range of addresses).  
These are typically values written to the PLC by operators and programs. On/ OFF, RUN/STOP are examples.

12. Modify the Address to match the actual address.
13. Apply a Tag name.
14. Edit Tags in Project Manager to assign **Alarms, Scaling, Engineering Units**, Description and other features.

## 1.3 Serial Comport Properties

The Serial Comport is associated with an RS232, RS422 or RS485 COM port on the SCADA Node PC (usually an RS232 port). This number must match the actual COM1, COM2, etc on your SCADA node.

Update Comport		<a href="#">Cancel</a>	<input type="button" value="Submit"/>
Interface Name	SERIAL <input type="button" value="v"/>		
Comport Number	<input type="text" value="1"/>		
Description	<input type="text" value="Description"/>		
Baud Rate	9600 <input type="button" value="v"/> bps		
Data bit	<input type="radio"/> 7 <input checked="" type="radio"/> 8 bits		
Stop bit	<input checked="" type="radio"/> 1 <input type="radio"/> 2 bits		
Parity	<input type="radio"/> None <input type="radio"/> Odd <input checked="" type="radio"/> Even		
Scan time	<input type="text" value="3"/> <input type="radio"/> MilliSecond <input checked="" type="radio"/> Second <input type="radio"/> Minute <input type="radio"/> Hour		
TimeOut	<input type="text" value="1000"/> MilliSecond		
Retry count	<input type="text" value="3"/>		
Auto Recover Time	<input type="text" value="20"/> Second		
HandShakeRts	<input type="radio"/> Yes <input checked="" type="radio"/> No		
HandShakeDtr	<input type="radio"/> Yes <input checked="" type="radio"/> No		
Backup Port Number	<input type="text" value="0"/>		

Figure 1.1 Serial Comport properties

### 1.3.1 Comport Number

The Serial Comport requires the comport number to match that of the physical interface (e.g. COM1, COM2, COM3, etc) on the SCADA Node.

### 1.3.2 Description

This is an optional field used for user reference.

### 1.3.3 Baud Rate

This must match the baud rate configured in the PLC using the PLC programming package (supplied by the PLC manufacturer) or dipswitches on the PLC COM card. All PLCs connected to this comport must use the same Baud Rate. Values are from 300 to 11200 baud.

### 1.3.4 Data Bits

This must match the number of data bits configured in the PLC using the PLC programming package (supplied by the PLC manufacturer) or dipswitches on the PLC COM card. All PLCs connected to this comport must use the same number of data bits. A typical value is 8 bits.

Once the start bit has been sent, the transmitter sends the actual data bits. There may either be 5, 6, 7, or 8 data bits, depending on the number you have selected. Both receiver and the transmitter must agree on the number of data bits, as well as the baud rate. Almost all devices transmit data using either 7 or 8 data bits.

### 1.3.5 Stop Bits

This must match the number of stop bits configured in the PLC using the PLC programming package (supplied by the PLC manufacturer) or dipswitches on the PLC COM card. All PLCs connected to this comport must use the same number of stop bits. A typical value is 1 bit. WebAccess supports **1** or **2** bits.

After the data has been transmitted, a stop bit is sent. A stop bit has a value of 1 - or a mark state - and it can be detected correctly even if the previous data bit also had a value of 1. This is accomplished by the stop bit's duration. Stop bits can be 1, 1.5, or 2 bit periods in length. WebAccess supports **1** or **2** bits.

### 1.3.6 Parity

This must match the parity configured in the PLC using the PLC programming package (supplied by the PLC manufacturer) or dipswitches on the PLC COM card. All PLCs connected to this comport must use the parity. A typical value is Even.

WebAccess supports **Even**, **Odd** and **None** for parity.

Besides the synchronization provided by the use of start and stop bits, an additional bit called a parity bit is optionally transmitted along with the data. A parity bit affords a small amount of error checking, to help detect data corruption that might occur during transmission. You can choose even parity, odd parity, mark parity, space parity or none at all. When even or odd parity is being used, the number of marks (logical 1 bits) in each data byte are counted, and a single bit is transmitted following the data bits to indicate whether the number of 1 bits just sent is even or odd. Mark parity means that

the parity bit is always set to the mark signal condition and likewise space parity always sends the parity bit in the space signal condition. Since these two parity options serve no useful purpose whatsoever, they are almost never used and are not supported in most Web Access drivers.

### 1.3.7 Scan Time

This is the time in milliseconds to scan the PLC. This must match the ability of the PLC to respond.

If the PLC cannot respond as fast as the SCAN Time entered, WebAccess will scan at a slower rate.

### 1.3.8 Timeout

Timeout is the time waited before re-sending a communications packet that did not have a reply.

Timeout specifies how long the software waits for a response to a data request, specifically to wait for a reply from one packet. A recommended value is 7 to 10 ticks, longer if the communication device is slow. This is protocol dependent: some protocols do not allow changes in time out.

Combined with Retry count, Timeout also determines time to consider a device or port as BAD. Timeout is the time to wait since last communication packet sent without a reply. Time is in milliseconds. The slow or poor quality communications require longer timeout. The faster the communications network or device, the shorter the timeout required. Shorter timeouts notify operators of communications failure more quickly.

---

*A note about ModBus ASCII – a one second timeout is standard for many Modbus ASCII devices. It is rarely smaller and often longer. For many Modbus ASCII devices timeout is fixed at 1 second. Satellite Technology routinely sees 2-second delays. 1 to 5 second timeouts are common for Modbus ASCII*

---

### 1.3.9 Retry Count

Number of times to retry communications if no reply is received from a device. Combined with Timeout, also determines time to consider a device or port as BAD.

In addition, Indicates the number of times after the first attempt has failed that communication should be attempted before indicating a failure. Specifically, how many times to send a single packet after the field device fails to respond to the first packet. After the retry count is exceeded, all the tags in the packet are marked with asterisks and the next packet of requests

is sent. A reasonable value is 3 to 5 times. After this number of tries, the tags in this packet are marked as "fail to respond" (i.e. asterisks) and are disabled. In reality, increasing the number of retries hides failures on the part of the field device to respond to a request. Essentially, increasing the retries gives the field device more chances to reply.

### 1.3.10 Auto Recover Time

Auto Recover Time is the time to wait before attempting to re-establish communications with a BAD device or port.

If communications to the PLC is unusually slow due to hardware, communications or network issues, you might consider increasing this value. If communications to the PLC or RTU fails frequently, you may want to decrease this number in order to have WebAccess try to re-establish communications sooner.

If communications to the PLC, RTU or device Fails (i.e. exceeds Timeout) WebAccess will wait the Auto Recover Time before trying to re-establish communications.

### 1.3.11 Hand Shake RTS

Is RTS (Request To Send) signal raised and lowered on the Serial Communications Port. RTS is determined by settings in the field device. Refer to your device interface manual to determine the value for this field and the type of cable used. Some WebAccess Drivers ignore this and leave RTS always ready. See the specific WebAccess Driver Manual for your device.

A Personal computer is a DTE device, while most other field devices are usually DCE devices. If you have trouble keeping the two straight then replace the term "DTE device" with "SCADA Node" and the term "DCE device" with "field device" in the following description.

RTS stands for Request To Send. This line and the CTS line are used when "hardware flow control" is enabled in both the DTE and DCE devices. The DTE device puts this line in a mark condition to tell the remote device that it is ready and able to receive data. If the DTE device is not able to receive data (typically because its receive buffer is almost full), it will put this line in the space condition as a signal to the DCE to stop sending data. When the DTE device is ready to receive more data (i.e. after data has been removed from its receive buffer), it will place this line back in the mark condition. The complement of the RTS wire is CTS, which stands for Clear To Send. The DCE device puts this line in a mark condition to tell the DTE device that it is ready to receive the data. Likewise, if the DCE device is unable to receive data, it will place this line in the space condition. Together, these two lines make up what is called RTS/CTS or "hardware" flow control. The Software Wedge supports this type of flow control, as well as Xon / XOff or "software" flow control. Software flow control uses special control characters transmitted from one device to another to tell the other device to stop or start sending data. With software flow control the RTS and CTS lines are not used

### 1.3.12 Hand Shake DTR

Is DTR (Data Terminal Ready) signal raised and lowered on the Serial Communications Port. Determined by settings in the field device and the type of cable used. Most WebAccess Drivers ignore this and leave DTR always ready. See the specific WebAccess Driver Manual for your device.

DTR stands for Data Terminal Ready. Its intended function is very similar to the RTS line. DSR (Data Set Ready) is the companion to DTR in the same way that CTS is to RTS. Some serial devices use DTR and DSR as signals to simply confirm that a device is connected and is turned on. The Software Wedge sets DTR to the mark state when the serial port is opened and leaves it in that state until the port is closed. The DTR and DSR lines were originally designed to provide an alternate method of hardware handshaking. It would be pointless to use both RTS/CTS and DTR/DSR for flow control signals at the same time. Because of this, DTR and DSR are rarely used for flow control.

### 1.3.13 Backup Port

This enables a redundant communications path to the Device. If communications cannot be established through this Comport, WebAccess will try a second Comport, specified as the Backup Port. You must configure the backup Port number in WebAccess, but without any devices on it. Usually the device must have two comports also. Not all Device Types support a backup Port.

The Backup Port is usually configured as the same type (SERIAL) for the Modbus ASCII protocol.

## 1.4 Device Properties – Modicon ASCII

Serial Port is an RS232, RS422, RS485 or other port on your SCADA node PC identified as COM1, COM2, COM3, COM4, etc. For More information on Serial Ports, see [Serial Comport Properties](#). Serial type Devices usually have a "native" driver written by WebAccess. It is possible to have a Serial connection to the device, but use the API or OPC Port Types in WebAccess.

Add your device to the Serial Port, by selecting the Serial Port you have configured, then select **Add Device**.

To modify an existing Device, Select **Device Properties**. The Device Properties Page for a Serial Type Device appears.

Delete		Add Tag		Add Block	
<b>Device Property</b> [Cancel] Submit					
Device Name	ModSim				
Description	Modbus ASCII RTU 1				
Unit Number	1				
Device Type	Modicon ▼				
Use ASCII Protocol:	1	Packet Delay (ms) :	0		
Digital block size :	512	Analog block size :	64		
[Cancel] Submit					

Figure 3-12 Modicon - Modbus ASCII device

**Important – Use ASCII Protocol** – set this to 1. To use the Modbus ASCII protocol, this must be set to 1. Otherwise, the Modbus RTU protocol (the default=0) will be used.

**Device Name** is any user-defined name. See [Device Name](#) for more information.

**Description** is a user defined. See [Description](#) for more information.

**Unit Number**, for Modbus, must correspond to the Unit Number used in the protocol addressing. See [Unit Number](#) for more information.

The **Device Type** is Modicon.

**Use ASCII Protocol** 1=Modbus ASCII protocol. 0 = Modbus RTU protocol. This must match the protocol of the field device (the PLC, RTU, etc.) See 1.1.5.1 Modbus RTU and 1.1.5.2 Modbus ASCII for a brief description of the differences.

**Important – Use ASCII Protocol:1** – set this to 1. To use the Modbus ASCII protocol, this must be set to 1. Otherwise, the Modbus RTU protocol (the default=0) will be used.

**Packet Delay (ms):** Some devices cannot receive very fast request after they respond previous packet. A delay may be required for the next request from WebAccess for those slow devices, especially for some old power meters.

**Digital block size:** Some Modbus compatible devices use only a certain part of a Modbus address or only handle a short data range for data request from client. WebAccess allows users to define the maximum data block size for both requested Digital and Analog type data.

**Analog block size:** Some Modbus compatible devices use only a certain part of a Modbus address or only handle a short data range for data request from client. WebAccess allows users to define the maximum data block size for both requested Digital and Analog type data.

### 1.4.1 Device Name

A Device is a PLC, Controller, VAV or other automation hardware or software entity. **Device name** is a User-assigned name that will appear in the Project Manager (Configuration Tool) and in runtime VIEW Displays. Choosing a descriptive Name can help technicians identify the location of your device.

Changing only the Device Name will rename the existing device.

Changing both the **Device Name** and the **Unit Number** will make a copy of the device (e.g. create another device).

### 1.4.2 Description

User assigned description up to 70 characters

### 1.4.3 Unit Number

For Modbus, this must correspond to the Unit Number used in the protocol addressing. This is the address configured in the device or by a dipswitch on the device. The range of Unit Number is 0 to 255 for Modbus.

This Unit Number will appear on the System Status Display, Point Detail, user-built displays and tags to reference the status of this device.

Changing only the Unit Number here will change the existing device.

Changing both the **Device Name** and the **Unit Number** will make a copy of the device (e.g. create another device).

#### 1.4.4 Device Type

This is the communication Driver used to communicate with all devices on this Com Port. Only one communications protocol is supported on the same COM port. Once a Device Type is created on a COM port, the Device Type of additional devices will be limited to this Device Type.

To use another communications device, you must configure another COM port. Multiple TCP/IP type Com Ports can be added which use the same TCP/IP Network Card on your PC.

---

### 1.5 Configure a Tag

This example is to configure two Tags that read an Analog Input (Address 30003) and an Analog Output (Address 40015).

1. Open **Internet Explorer**.
2. Connect to **Project Node**.
3. Start **WebAccess Configuration**.
4. Select **Project**.
5. Select **SCADA Node**.
6. Select the Modicon **Device**.
7. Select **Add Tag**.
8. From **Parameter** Pull Down List Select **AI**. This will configure an Analog Input. Wait for the Page to update.
9. Optionally, select **ALARM** from the ALARM pulldown list. Wait for the Page to update with a PINK highlight around alarm (an additional Alarm Fields at bottom of page).
10. Enter a **Tagname** users can use to identify this Analog Input measurement. For example, if this is a Flow measurement, enter **Flow1**.
11. Edit the **Address** to the actual address. From the example, Enter: **30003**
12. Enter a Description. This will help identify this tag to Users and Operators. For example, enter Boiler #1 Steam Flow.

13. Optionally enter, Scaling, Span Hi, Span Low, Engineering Units, and Alarms; enable data logging, etc.
14. Press **Submit**.
15. From **Parameter** Pull Down List Select **AO**. This will configure an Analog Output. Wait for the Page to update.
16. Optionally, select **ALARM** from the ALARM pulldown list. Wait for the Page to update with a PINK highlight around alarm (and additional Alarm Fields at bottom of page).
17. Enter a **Tagname** users can use to identify this Analog Output measurement. For example, if this is a signal to a Valve, enter **Valve1**.
18. Edit the **Address** to the actual address. From the above example, Enter: **40015**
19. Enter a Description. This will help identify this tag to Users and Operators. For example, enter Boiler #1 Steam Valve.
20. Optionally enter, Scaling, Span Hi, Span Low, Output Limits, Engineering Units, and Alarms; enable data logging, etc.
21. Press **Submit**.

Congratulations! You have just configured a Measurement and Output Tags to Modbus device.