# PowerXL DG1 to XV HMI by Modbus TCP

## Introduction

The purpose of this application note is to demonstrate how to operate a PowerXL DG1 drive via Modbus TCP and an XV-102 CoDeSys-3 controller. The PowerXL drive has an onboard Ethernet port that supports both Ethernet/IP and Modbus TCP. The drive needs to be configured with an IP address for Ethernet communications, but it auto senses the protocol. In other words, as long as the Modbus TCP master polls the PowerXL DG1 drive reading and writing valid Modbus data addresses, the drive will respond.

The IP address for the drive can be set via the display on the front of the drive. This will be described later in this document.

While this application example uses an Eaton XV-102 HMI/PLC to control and monitor the PowerXL drive over Modbus TCP, any Modbus TCP master may be used for this purpose. Eaton's XSoft-CoDeSys, version 3.5.4 programming software is used to create the XV-102 project. The XV-102 project is used to configure the controller to poll the PowerXL DG1 drive for control and monitoring purposes.

Modbus data addresses are published for the DG1 drive in the Modbus TCP section of the PowerXL DG1 drive's Communication Manual, publication MN040010EN.

## Configuring the PowerXL DG1 Drive

The IP addresses for the devices used in this example will be as follows:

- PowerXL Drive: 192.168.1.2
- XV102 HMI/PLC: 192.168.1.8
- Computer: 192.168.1.51

Subnet mask: 255.255.255.0

Connect your computer, PLC and the PowerXL drive to an Ethernet switch.

To configure the IP address of the PowerXL DG1 drive via its keypad/display, power the drive and navigate to the Active IP Address to set the IP address for the drive. As noted above, the IP address for the drive in this example is 192.168.1.2.

To access the Ethernet parameters from the drives keypad/display:

1. Start by selecting Parameters with the up/down arrow keys.
2. Then press the right arrow key which will display Basic Parameters
3. Press the up arrow key twice until Communications is displayed
4. Press the right arrow key once, then the down arrow key twice so Ethernet/IP/Modbus TCP is displayed
5. Press the right arrow key to access the following Ethernet parameters:
   a. Address Mode, select Static IP

b. Active IP Address, 192.168.1.2 is entered for this example
c. Active Subnet Mask, 255.255.255.0 is entered for this example

Power cycle the drive to activate these new Ethernet parameters.

Refer to the Communication manual for this drive for additional details. Publication MN040010EN.

The drive must also be configured to accept remote control and speed via the network.

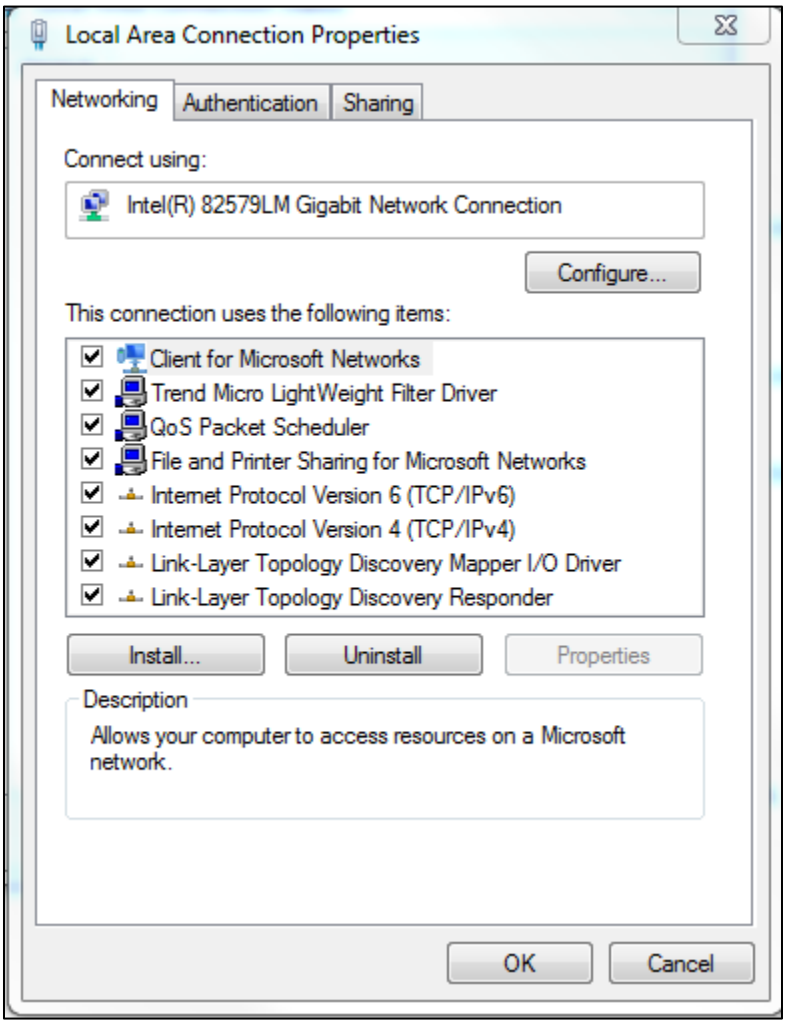
- Set Remote1 Control Place to: Fieldbus
- Set Remote1 Reference to: Fieldbus Ref

Both of these parameters can be found under: Parameters / Basic Parameters from the drive's key-pad/display.

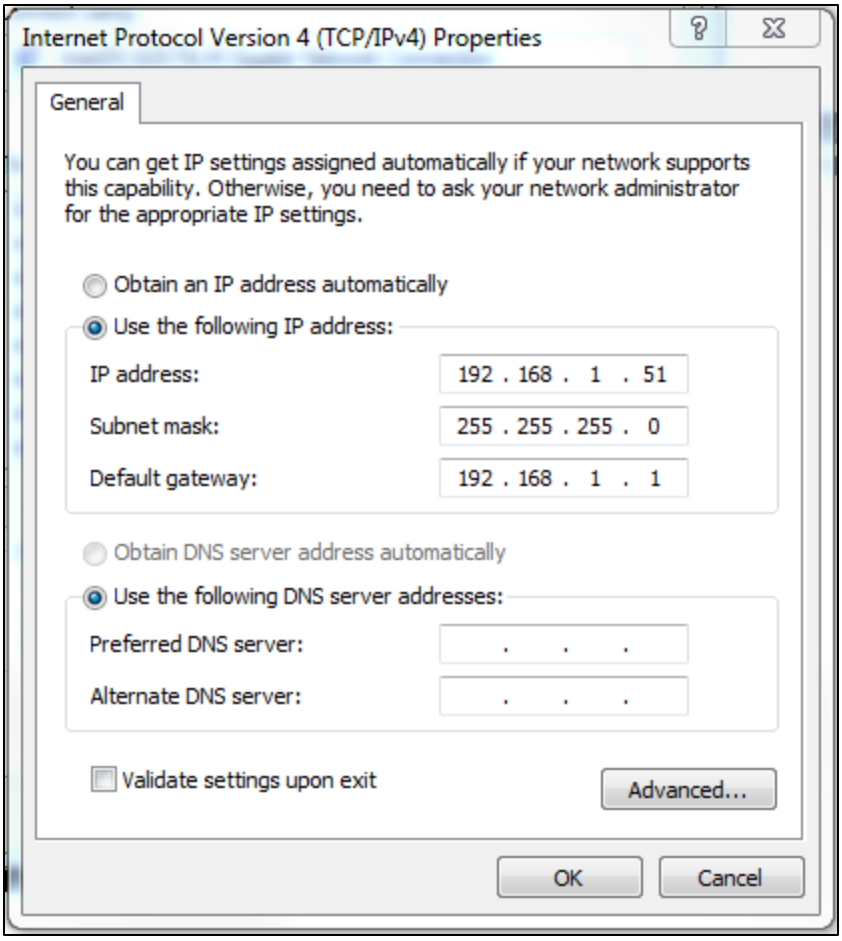
## Change the IP Address of your computer

To change the IP address for a computer running Windows 7, follow the procedure below:

1. From the Start menu, choose Control Panel. From the Control Panel, choose Network and Sharing Center.
2. With the computer connected to an Ethernet network, select the Local Area Connection. Unless the computer is connected to a network, this Local Area Connection will not be present.
3. The Local Area Connection Status window will be displayed. Select Properties.
4. From the window shown below, select Internet Protocol Version 4 (TCP/IPv4) to highlight it, then select Properties.



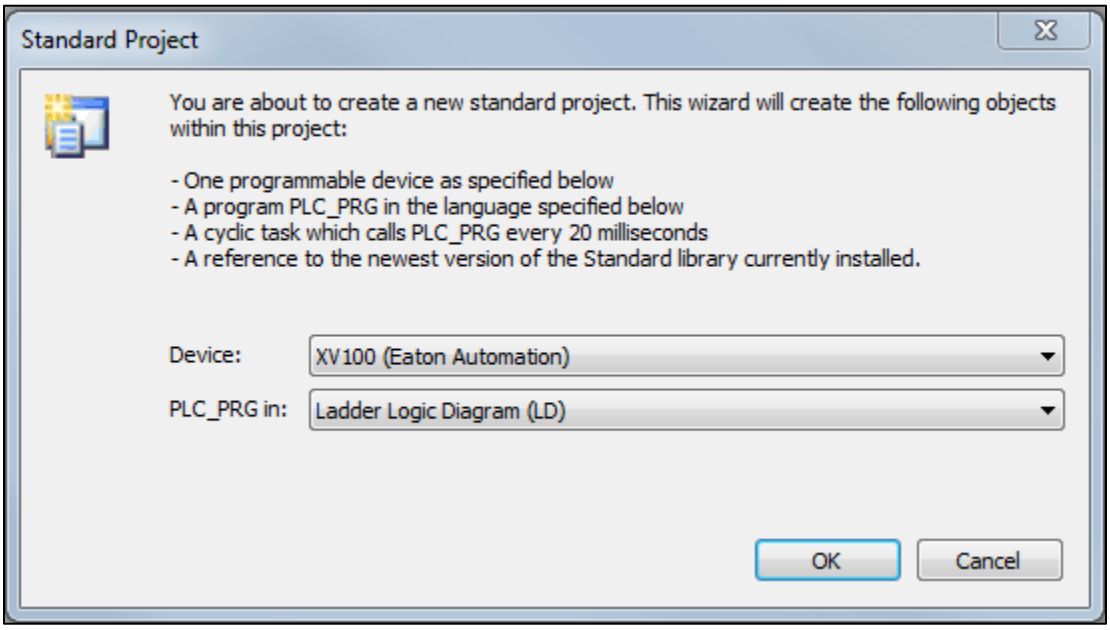
5. Per the following window, select Use the following IP Address, then enter an IP address, Subnet mask and a Default gateway if it applies.




6. When finished, select OK and close all the windows used along the way. Your computer's Ethernet port will now be actively using the IP address and Subnet mask you just entered.

## Creating a Project in XSoft-CoDeSy 3.5.4

Create a project in XSoft-CoDeSys 3.5.4. Give the project a name and select the controller type and programming language per the following:



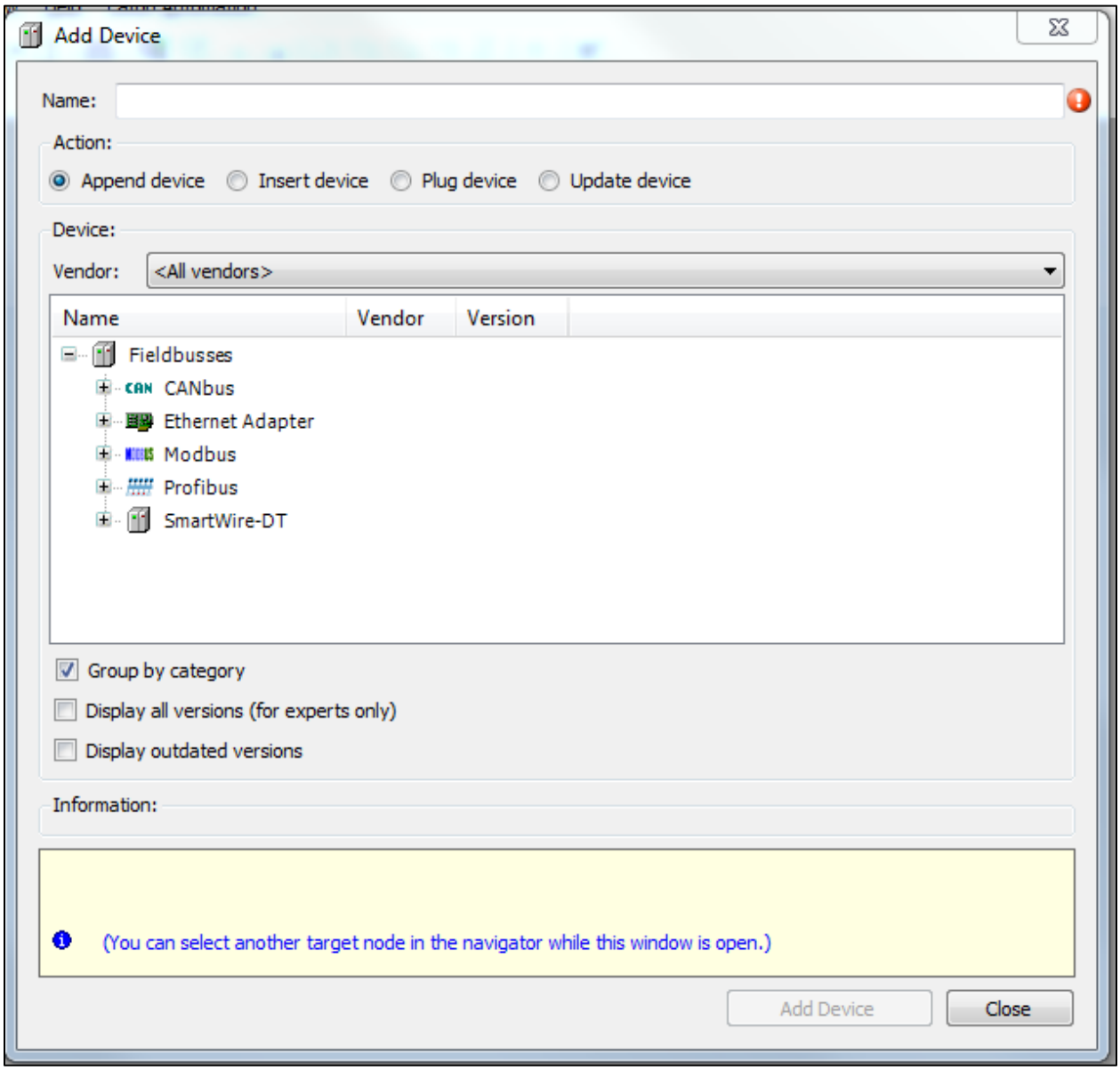

Select OK to create the project.

Note that an XC-152 or XC-202 PLC can also be used. Both of these CoDeSys controllers also have an Ethernet port that supports Modbus TCP and Ethernet/IP. The same project can be used by simply changing the controller type. An XV-102 (XV100) controller was used for this example.

Note also that CoDeSys supports 6 different programming languages and any of them can be used, including: CFC, FBD, IL, LD, SFC and ST. Ladder Logic (LD) is used for this example.

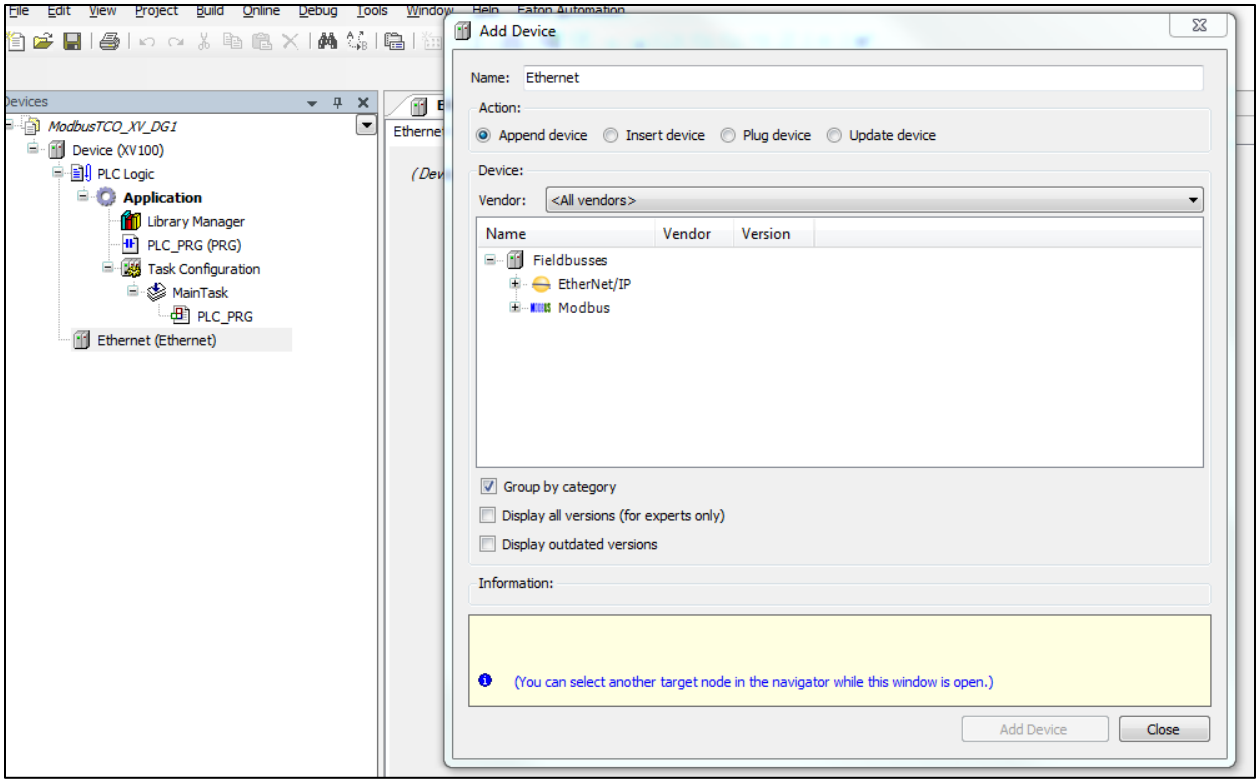
## Creating a Modbus TCP Network in XSoft-CoDeSys 3.5.4

On the left portion of the project screen in XSoft-CoDeSys, right click on “Device (XV100)” and select Add Device. The following screen will open:

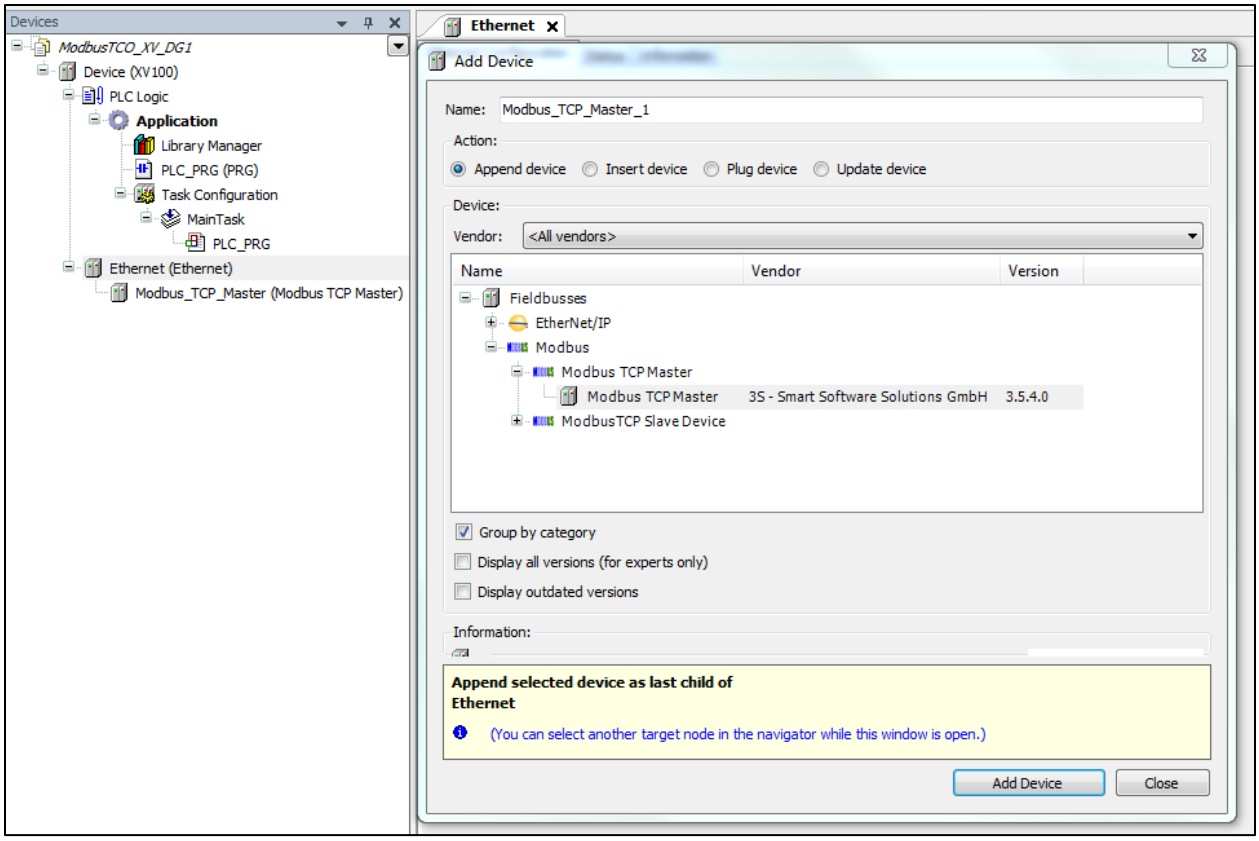



Select the plus sign to the left of Ethernet Adapter. Then select Ethernet below it and select the Add Device button. Don’t close the Add Device Window yet.

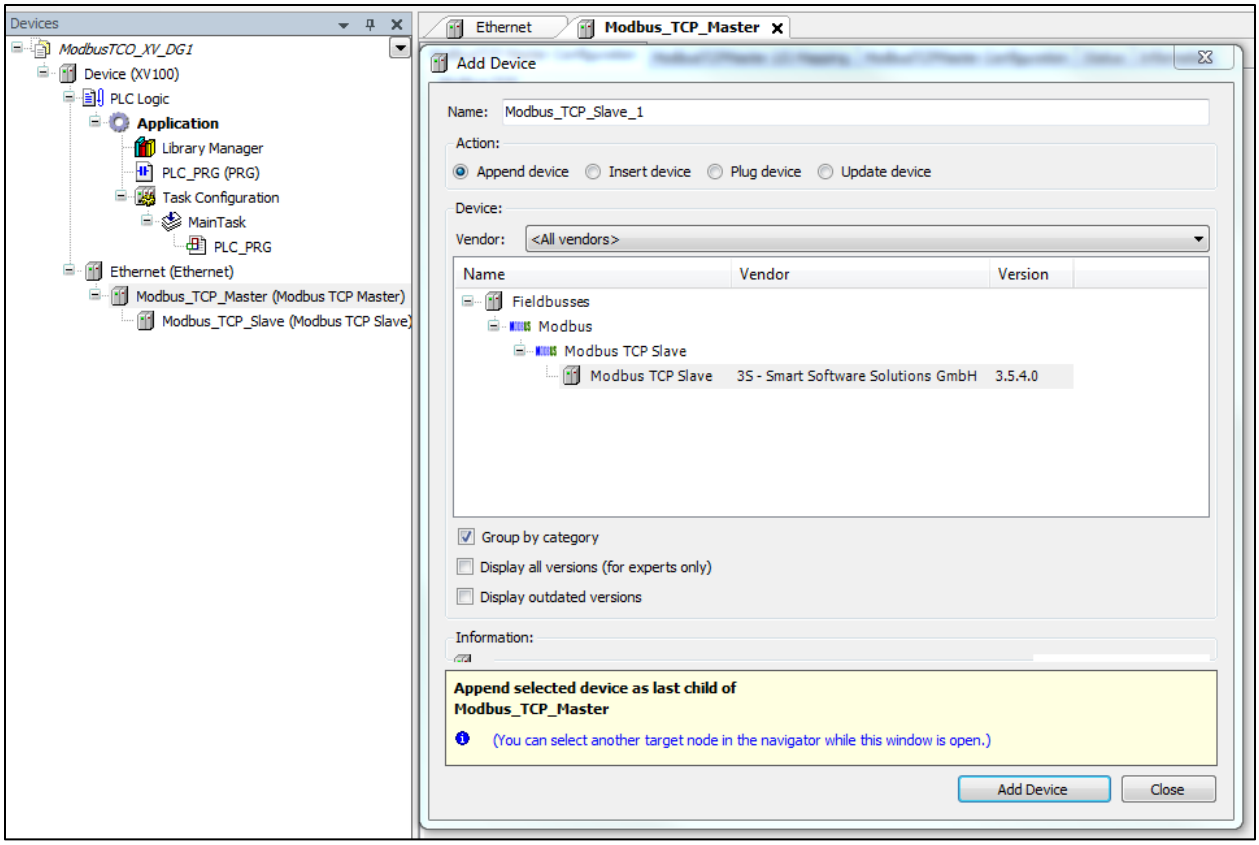
Double click Ethernet that is now displayed at the bottom of the tree on the left and the Add Device Window will change as follows:




Select the plus sign to the left of Modbus in the Add Device Window then the plus sign in front of Modbus TCP Master as well. Then select Modbus TCP Master followed by the Add Device button to add it to the tree on the left as shown below. Do not close the Add Device Window yet.

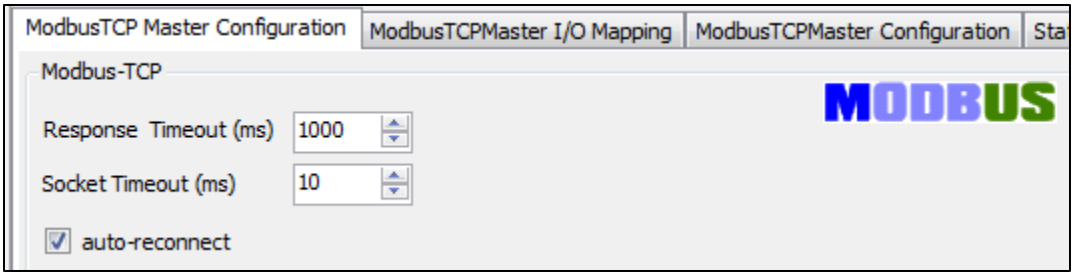



Double click the Modbus TCP Master on the left and select the Modbus TCP Slave under Modbus/Modbus TCP Slave on the Add Device Window as follows to add the Modbus slave to the tree on the left:

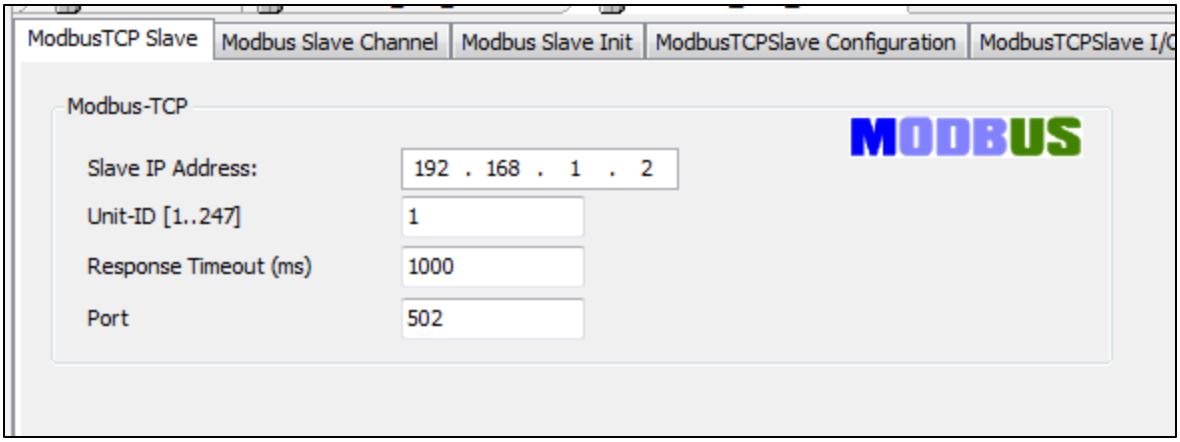



Now close the Add Device Window. The Ethernet port on the XV HMI/PLC will be the master and the PowerXL DG1 Drive will be the Modbus slave.

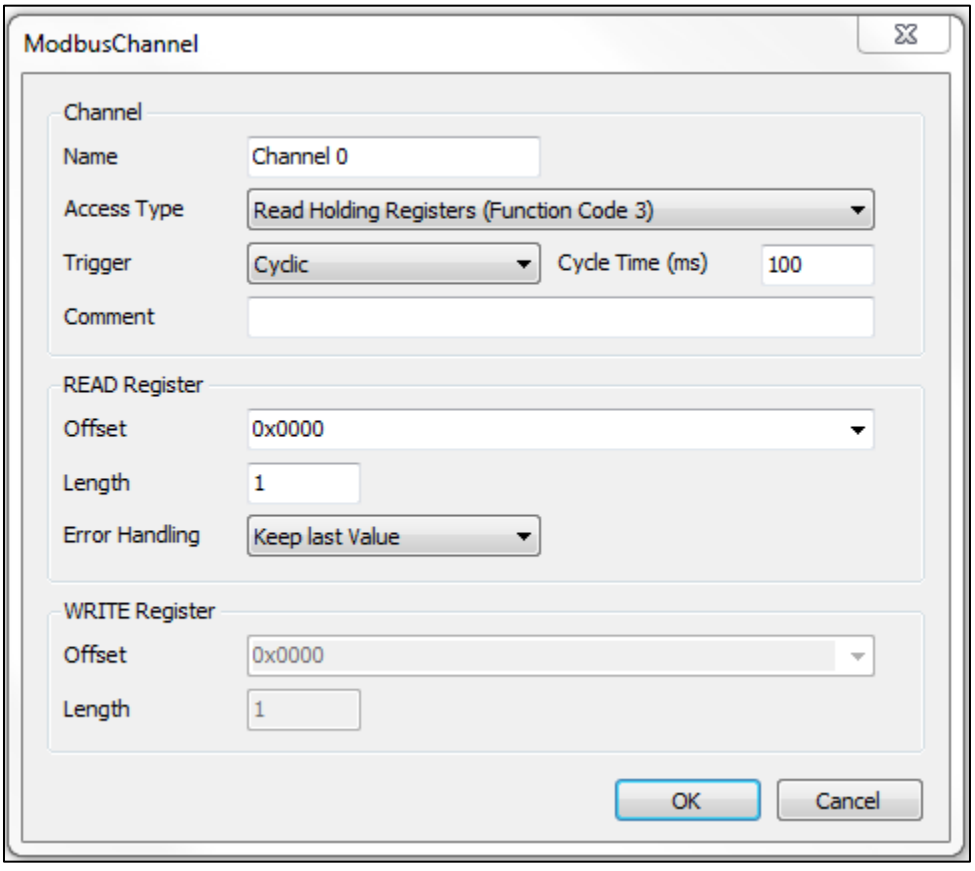
Double click the Modbus TCP Master in the tree to open its configuration pages on the right. Select the ModbusTCP Master Configuration tab. Select “auto-reconnect”. This will allow the Modbus TCP Master to re-connect automatically following a loss of communications, once the communication issue has been resolved.



Next double click the Modbus TCP Slave on the tree to open its configuration pages. Select the ModbusTCP Slave tab and configure the following:

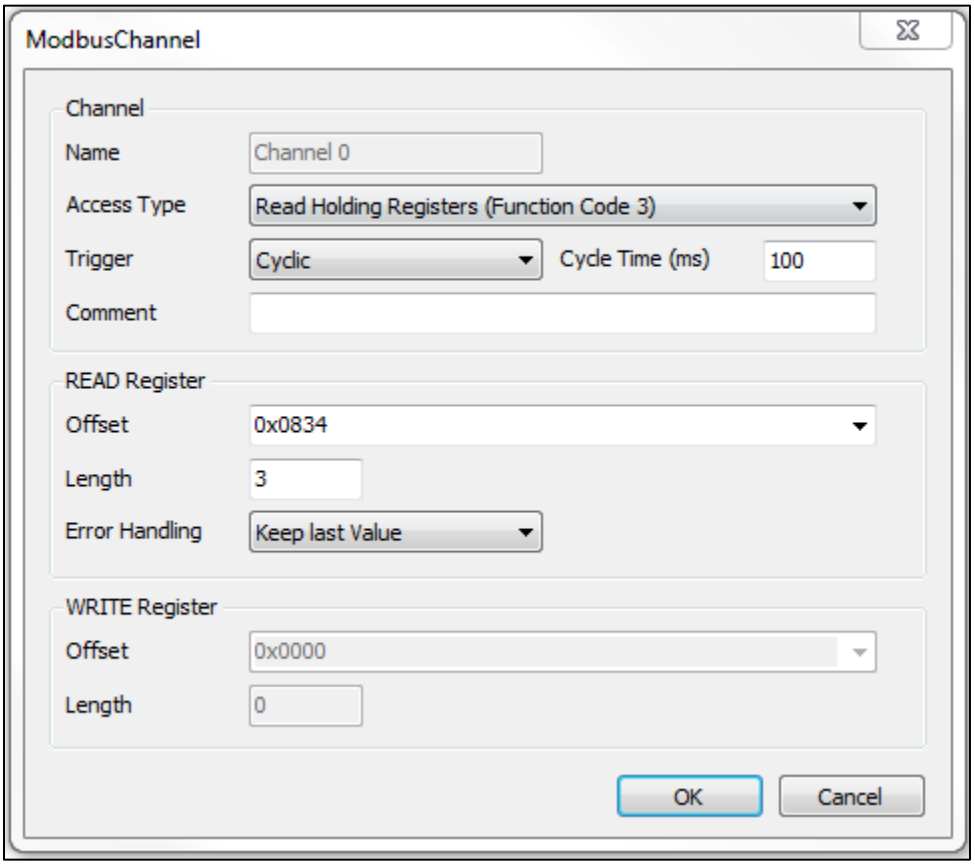



Then select the Modbus Slave Channel tab. Select the Add Channel button at the bottom of that screen and the following window will be displayed:



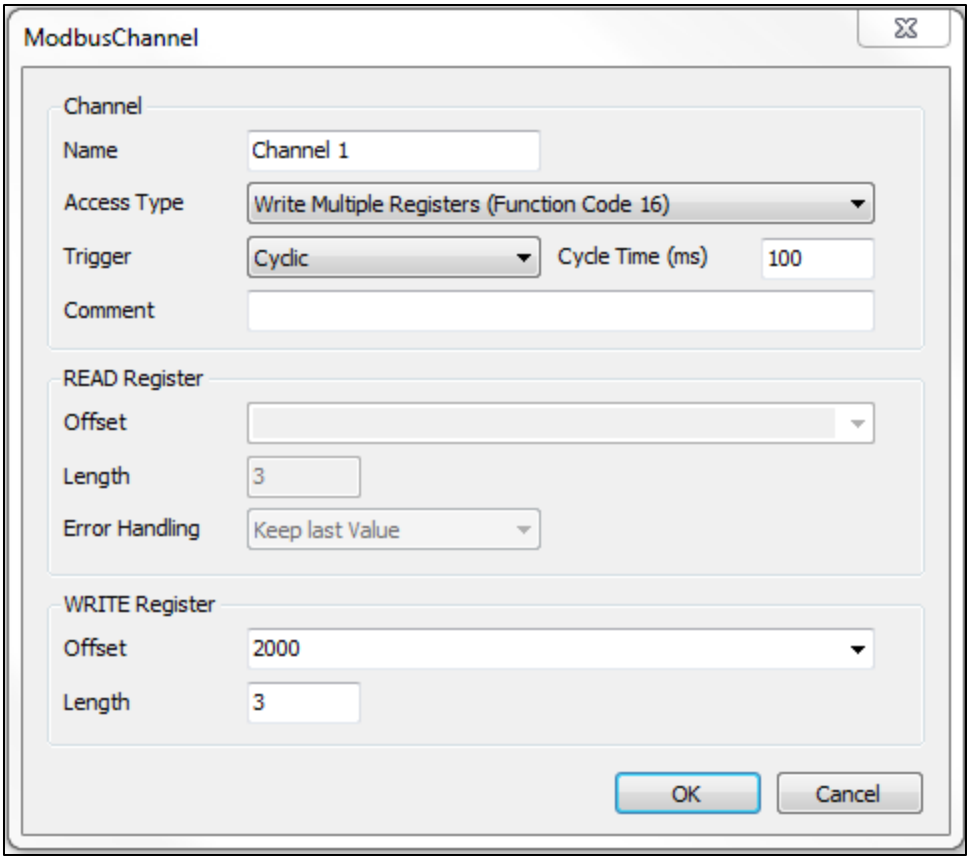

Two Modbus Channels will be added to this project allowing it to control the On/Off state of the drive as well as speed and monitor the Running status and the Actual Speed.

First configure the Read message. Per the PowerXL DG1 Communication Manual, the Status Word, General Status Word and Actual Speed begin at Modbus data address 2100 decimal or 0834 Hexadecimal. The Read Modbus Channel should look like the following:

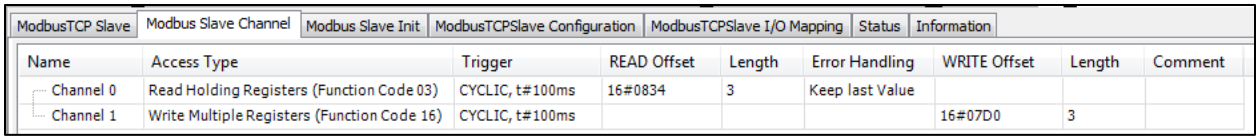



Select the OK button to add it.

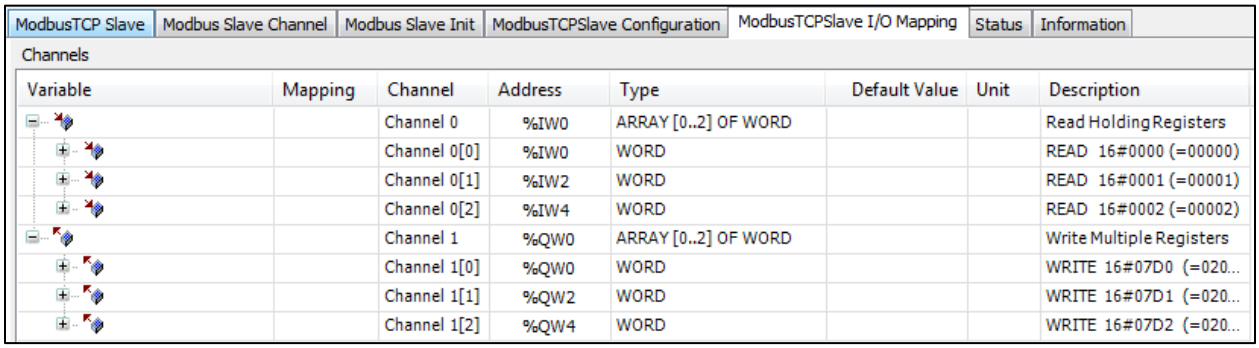
Select the Add Channel button again and fill in the following for the Write message of 3 registers in length that includes the Control Word, the General Control Word and the Speed Reference. The starting Modbus data address is 2000 decimal or 07D0 hexadecimal.




The Modbus Slave Channel tab should now look like the following:




Select the ModbusTCPSlave I/O Mapping tab. This is where the 3 input and 3 output registers are shown per the following:



Descriptive variable names can now be added to these generic I/O tags. These descriptive variables can then be accessed in the program and from visualization screens. The descriptive variable names can be applied to entire words or individual bits as follows:

ModbusTCP Slave | Modbus Slave Channel | Modbus Slave Init | ModbusTCPSlave Configuration | ModbusTCPSlave I/O Mapping | Status | Information

Channels

| Variable | Mapping | Channel | Address | Type | Default Value | Unit | Description |
|---|---|---|---|---|---|---|---|
| | | Channel 0 | %IW0 | ARRAY [0..2] OF WORD | | | Read Holding Registers |
| | | Channel 0[0] | %IW0 | WORD | | | READ 16#0834 (=02100) |
| Ready | | Bit0 | %IX0.0 | BOOL | FALSE | | |
| Running | | Bit1 | %IX0.1 | BOOL | FALSE | | |
| FWD_REV | | Bit2 | %IX0.2 | BOOL | FALSE | | |
| Fault | | Bit3 | %IX0.3 | BOOL | FALSE | | |
| Warning | | Bit4 | %IX0.4 | BOOL | FALSE | | |
| AtRef | | Bit5 | %IX0.5 | BOOL | FALSE | | |
| | | Bit6 | %IX0.6 | BOOL | FALSE | | |
| | | Bit7 | %IX0.7 | BOOL | FALSE | | |
| | | Bit8 | %IX1.0 | BOOL | FALSE | | |
| | | Bit9 | %IX1.1 | BOOL | FALSE | | |
| | | Bit10 | %IX1.2 | BOOL | FALSE | | |
| | | Bit11 | %IX1.3 | BOOL | FALSE | | |
| | | Bit12 | %IX1.4 | BOOL | FALSE | | |
| | | Bit13 | %IX1.5 | BOOL | FALSE | | |
| | | Bit14 | %IX1.6 | BOOL | FALSE | | |
| | | Bit15 | %IX1.7 | BOOL | FALSE | | |
| General_Drive_Status | | Channel 0[1] | %IW2 | WORD | | | READ 16#0835 (=02101) |
| DG1_Actual_Speed | | Channel 0[2] | %IW4 | WORD | | | READ 16#0836 (=02102) |

Variable names have been added to the 3 input status registers above. The first register contains individual status bits, so variable names have been applied at the bit level. The third input status register contains a 16-bit decimal value for the scaled speed in Hz x 100. So, a variable name has been applied at the word level for the Actual Speed.

Below are descriptive variable names added to the 3 output registers in a similar manner.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | Channel 1 | %QW0 | ARRAY [0..2] OF WORD | | | Write Multiple Registers |
| | | Channel 1[0] | %QW0 | WORD | | | WRITE 16#07D0 (=02000) |
| Run | | Bit0 | %QX0.0 | BOOL | FALSE | | |
| Direction | | Bit1 | %QX0.1 | BOOL | FALSE | | |
| Fault_Reset | | Bit2 | %QX0.2 | BOOL | FALSE | | |
| | | Bit3 | %QX0.3 | BOOL | FALSE | | |
| | | Bit4 | %QX0.4 | BOOL | FALSE | | |
| | | Bit5 | %QX0.5 | BOOL | FALSE | | |
| | | Bit6 | %QX0.6 | BOOL | FALSE | | |
| | | Bit7 | %QX0.7 | BOOL | FALSE | | |
| Net_Ctrl | | Bit8 | %QX1.0 | BOOL | FALSE | | |
| Net_Ref | | Bit9 | %QX1.1 | BOOL | FALSE | | |
| | | Bit10 | %QX1.2 | BOOL | FALSE | | |
| | | Bit11 | %QX1.3 | BOOL | FALSE | | |
| | | Bit12 | %QX1.4 | BOOL | FALSE | | |
| | | Bit13 | %QX1.5 | BOOL | FALSE | | |
| | | Bit14 | %QX1.6 | BOOL | FALSE | | |
| | | Bit15 | %QX1.7 | BOOL | FALSE | | |
| General_Drive_Control | | Channel 1[1] | %QW2 | WORD | | | WRITE 16#07D1 (=02001) |
| DG1_Speed_Reference | | Channel 1[2] | %QW4 | WORD | | | WRITE 16#07D2 (=02002) |

The descriptions for these Status and Control registers can be found in the PowerXL DG1 Drive Communications manual, in the Modbus TCP chapter.

These variables can now be used when creating a program to control and monitor the PowerXL DG1 drive. These same variables can also be used to develop visualization screens to control and monitor the drive.

A few things to note:

1. Set the Net_Ctrl (Network Control) and Net_Ref (Network Reference) output bits prior to operating the drive through the network.
2. The Run bit instructs the drive to Run the motor and the Direction bit toggles between FWD (0) and REV (1).
3. The DG1_Speed_Reference value sent to the drive is in percent x 100. So, the range is 0-10000 decimal (0-100.00%)
4. The DG1_Actual_Speed value returned from the drive is in frequency x 100. So, the range is 0-6000 decimal (0-60.00Hz).

## References

- PowerXL DG1 Series VFD Installation Manual, Publication MN040002EN
- PowerXL DG1 Series VFD Application Manual, Publication MN040004EN
- PowerXL DG1 Series Communication User Manual, Publication MN040010EN

## Additional Help

In the US or Canada: please contact the Technical Resource Center at 1-877-ETN-CARE or 1-877-326-2273 option 2, option 6.

All other supporting documentation is located on the Eaton web site at www.eaton.com/Drives

**Eaton**
1000 Eaton Boulevard
Cleveland, OH 44122 USA
Eaton.com


Printed in USA
Publication No. AP040086EN
September 2014