This AOI handles the 16-bit unsigned integer to 16-bit signed integer data type conversion issue that can occur when a Banner DXM transfers non-temperature data to an Allen Bradley PLC. Allen Bradley uses the 16-bit signed integer format, where bit 15 is the sign value (0 = positive, 1 = negative). Large numbers sent from the DXM that are not meant to have a sign associated with them can be mis-identified without this sign fix.

Components
DXM_Sign_Fix.L5X

Hardware
DXM R1 or R3 Gateway

UDT's Packaged with AOI
DXM_UDT

DXM Program
194730.xml configuration

1. Installation Process

This section describes how to install the AOI in Logix Designer software.

1. Open a project.
2. In the Controller Organizer window, right-click on the Add-On Instruction folder. Select Import Add-On Instruction option.
3. Navigate to the correct file location and select the AOI to be installed. In this example the "DXM_Sign_Fix.L5X" file will be selected. Click the Open button.
4. The Import Configuration window will pop up. The default selection will create all the necessary items for the AOI. Click the OK button to complete the import process.
5. The AOI is added to the Controller Organizer window.
6. AOI installation into the Logix Designer software complete.

2. DXM EtherNet/IP Connection

Make an EtherNet/IP connection to the DXM.

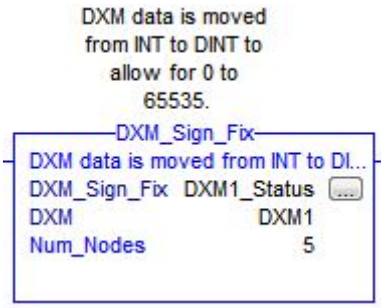
Create an Ethernet communications module for the DXM device. In this example the EDS file was used, and the connection was named "DXM1". The controller tags include Input (I) and Output (O) Assembly Instances. Each Assembly has a corresponding tag array. The AOI assumes that tags are arrays of 16-bit integers. Creating this Class 1 EtherNet/IP implicit IO connection will provide the PLC access to the internal registers of the DXM. See the Appendix A of this document for more information of how to create this connection.

3. Configuring the AOI

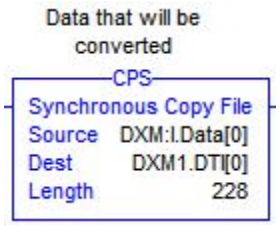
This section describes how to configure the AOI.

1. Add the “DXM_Sign_Fix” AOI to your ladder logic program. For each question mark shown in the instruction we need to create and link a new tag array.

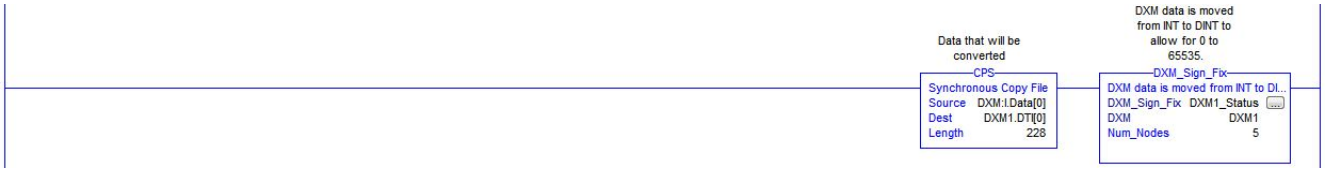
2. In the AOI, right-click on the question mark on the line labeled “DXM_Sign_Fix”. Click New Tag. In this example, we’ll use the name “DXM1_Status”.
3. Now click on the question mark on the line labled “DXM”. Click New Tag. In this example, we’ll use the name “DXM1”.



4. On the “Num_Nodes” line enter the number of nodes that in the system. In the above example, 5 is used. The maximum number of nodes allowed is 27.
5. Next add a CPS (Synchronous Copy File) command before the AOI. This instruction allows the AOI to read from the local registers found in the DXM. For this CPS instruction, the Source is “DXM1:I:Data[0]” and the Destination is “DXM1.DTI[0]”. The length is set as 228.



Here is what the entire rung looks like when completed.



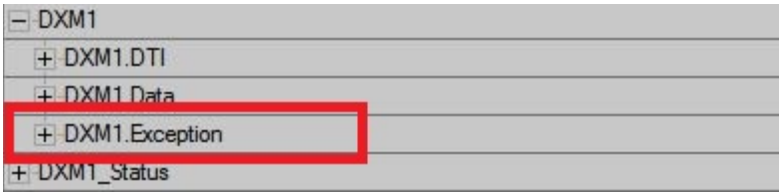
The “Banner_DXM_Sign_Fix” AOI is now ready to use.

4. Using the AOI

The Add-On Instruction has created a group of 32-bit signed integer tags which will show the 16-bit unsigned integer data from the DXM. In the case of a 32-bit signed integer, bit 31 is the sign value (0 = positive, 1 = negative). The DXM will never send a number large enough to mistakenly change this bit, so the sign flipping issue is resolved.

Look in the Controller Tags to find the name used above. This example used the name “DXM1”.

− DXM1	{...}
+ DXM1.DTI	{...}
− DXM1.Data	{...}
+ DXM1.Data[0]	0
+ DXM1.Data[1]	0
+ DXM1.Data[2]	0
+ DXM1.Data[3]	0
+ DXM1.Data[4]	0
+ DXM1.Data[5]	0
+ DXM1.Data[6]	0
+ DXM1.Data[7]	0
+ DXM1.Data[8]	1
+ DXM1.Data[9]	0
+ DXM1.Data[10]	52640
+ DXM1.Data[11]	32785
+ DXM1.Data[12]	0
+ DXM1.Data[13]	0
+ DXM1.Data[14]	0
+ DXM1.Data[15]	0



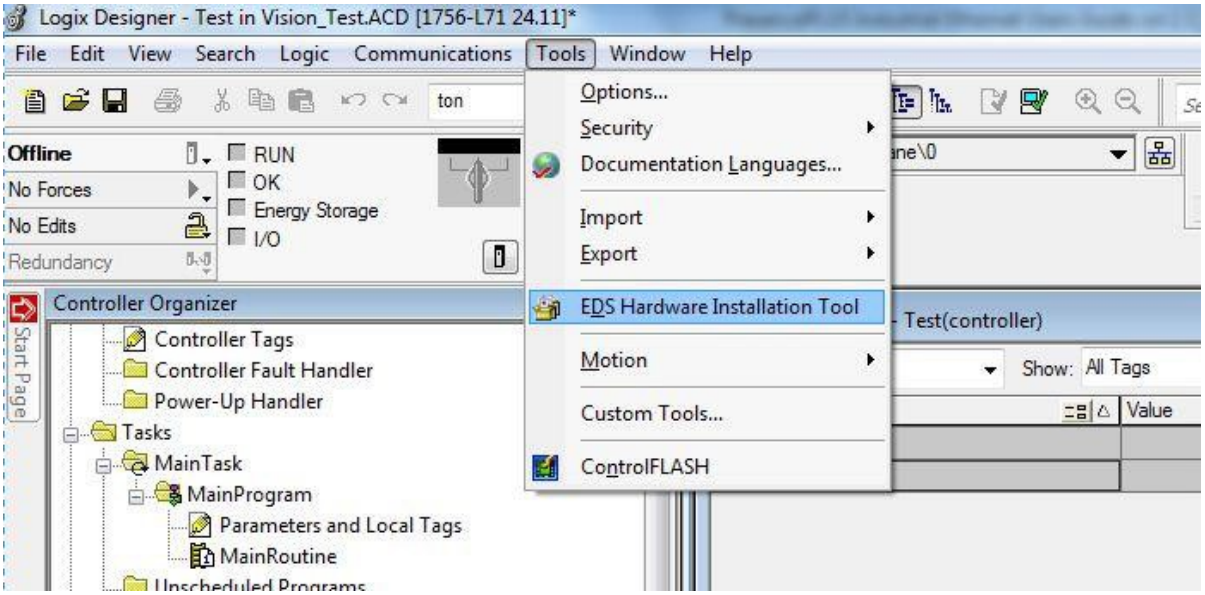
DXM1.Exception: In those cases where a Temperature reading is part of the DXM data (the only case where negative numbers are used in the DXM system), one can make an exception to the sign-fixing behavior of this AOI. Exception is an array of 48 SINT values. Each element in the array corresponds to either the DXM itself, [0], or one of the nodes connected to the system, [1] through [47]. Each bit in a given SINT corresponds to one of that node's 8 possible inputs. Setting one of these bits to "1" preserves the sign information for that DXM register.

For example, if input #3 on node 2 is a temperature probe, setting DXM1.Exception[2].2 to a "1" will preserve the signed nature of the original DXM information. (Recall that DXM1.Exception[0] refers to the DXM itself).

Appendix A, Using DXM EDS File

Here is an example of using the DXM EDS file to create a connection on a ControlLogix PLC.

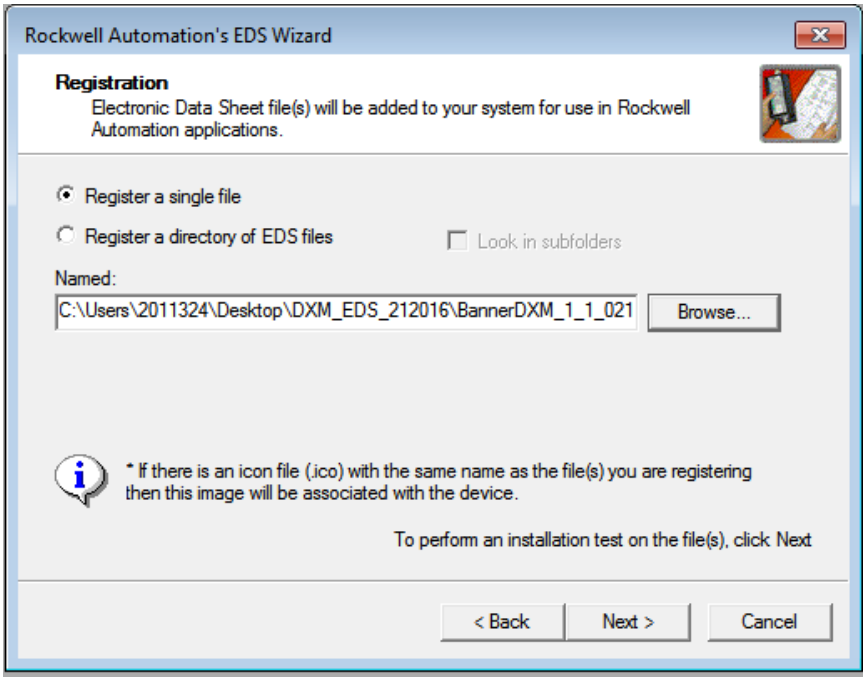
1. First we use the EDS Hardware Installation Tool to register the EDS file.



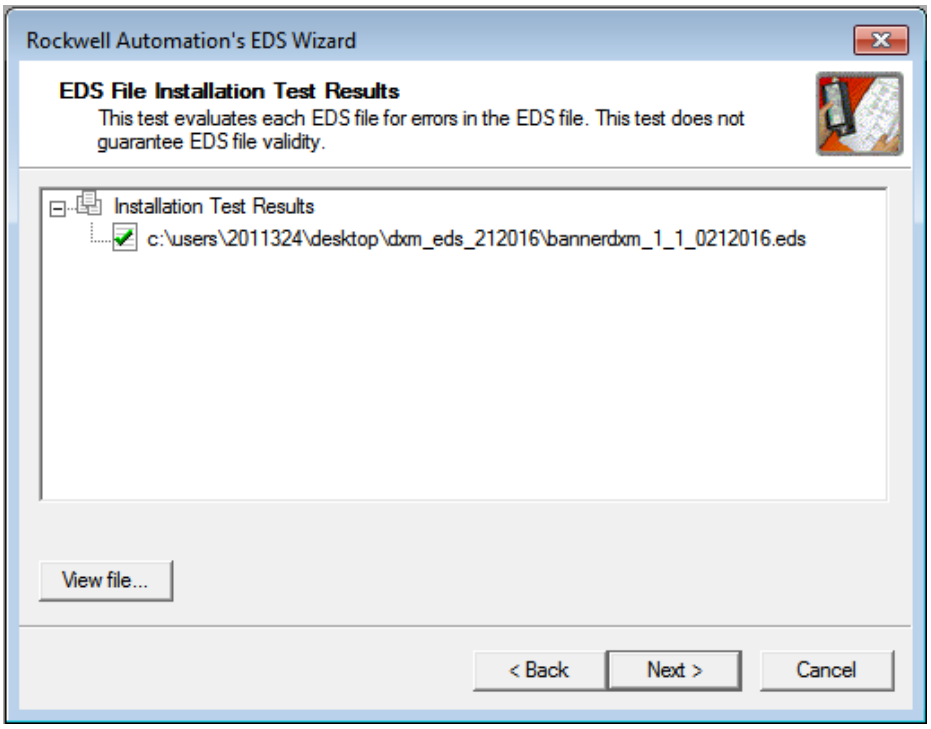
2. Click Next

3. Choose the “Register and EDS file(s)” option

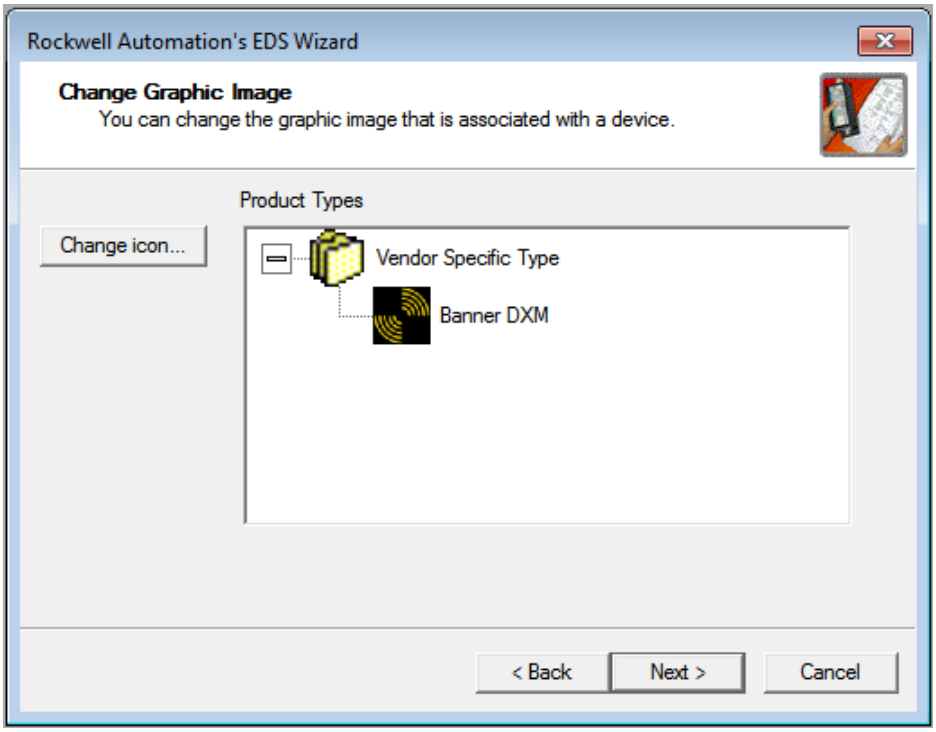
4. Browse to find the EDS file, then click Next.



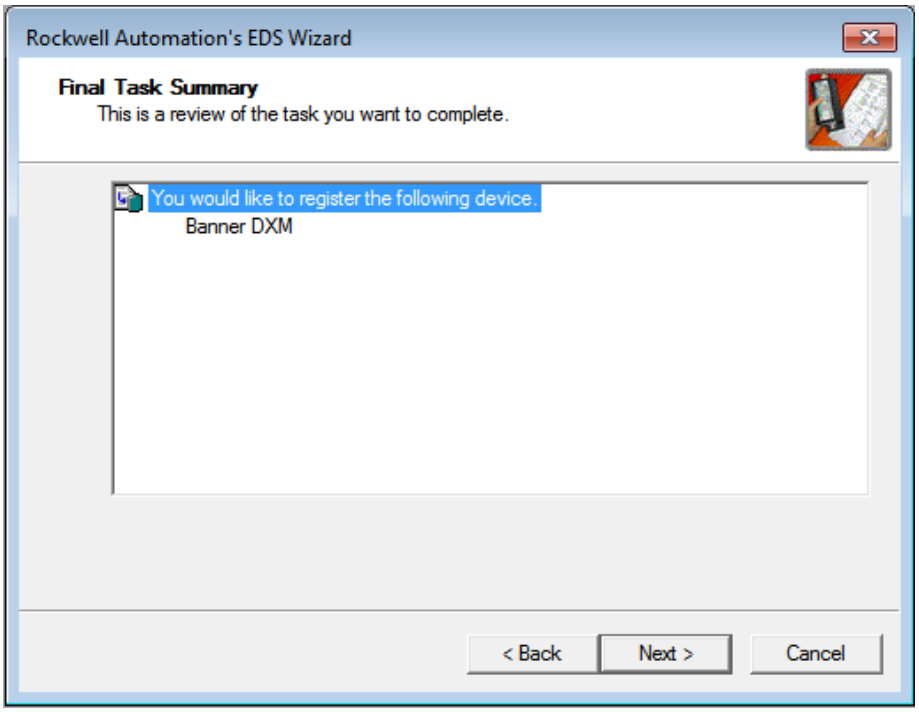
5. The file has been tested and can be registered. Click Next.



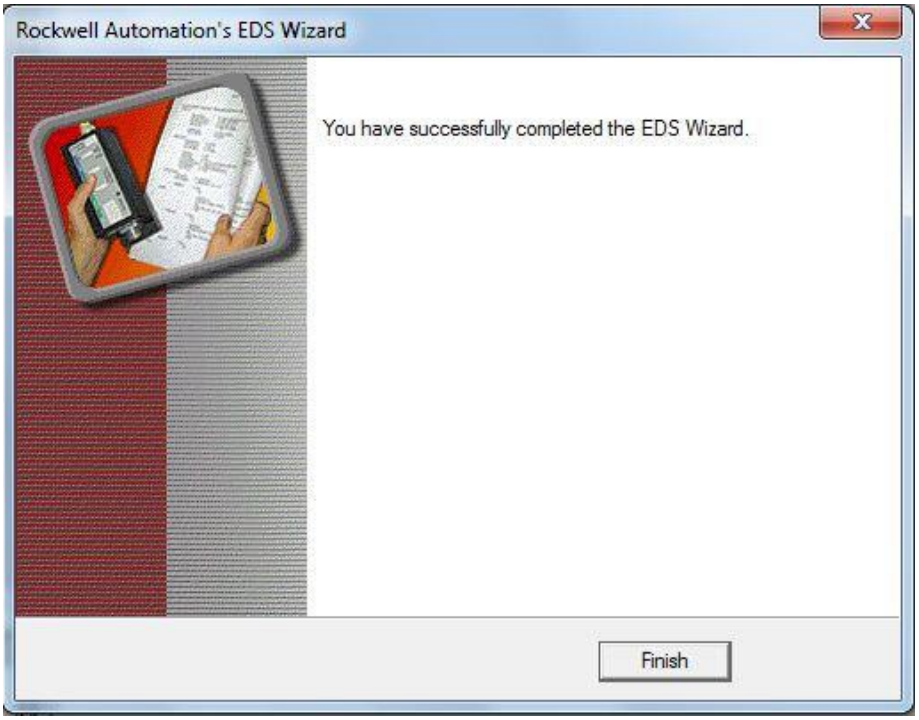
6. Here is the icon associated with the EDS file. Click Next.



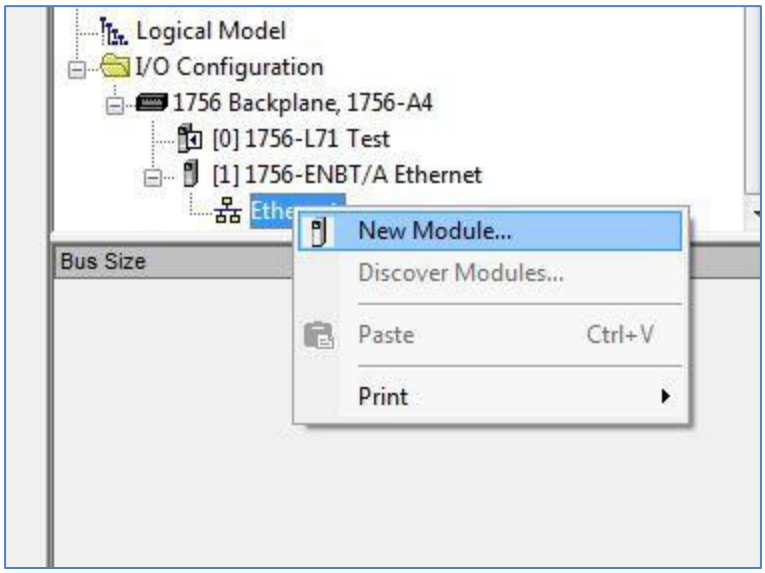
7. Everything looks good. Click Next to register this EDS file.



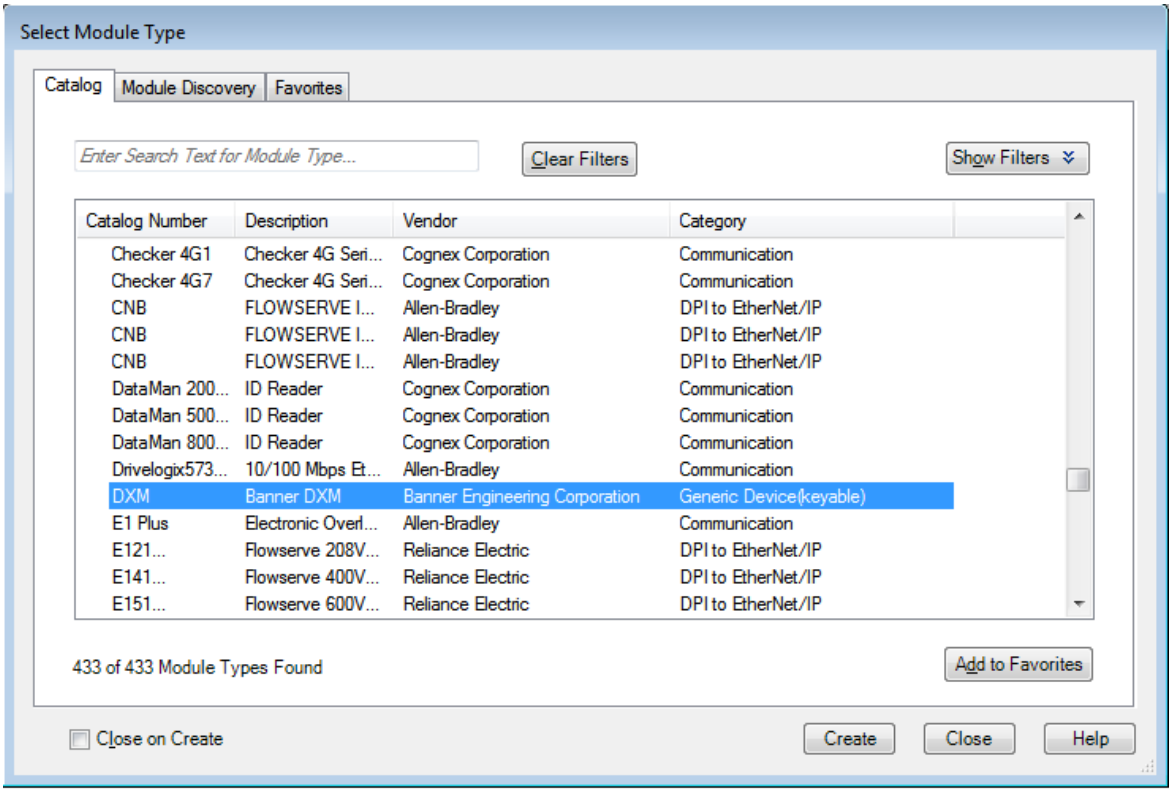
8. Complete! Click Finish. The EDS file is now registered in the Rockwell software.



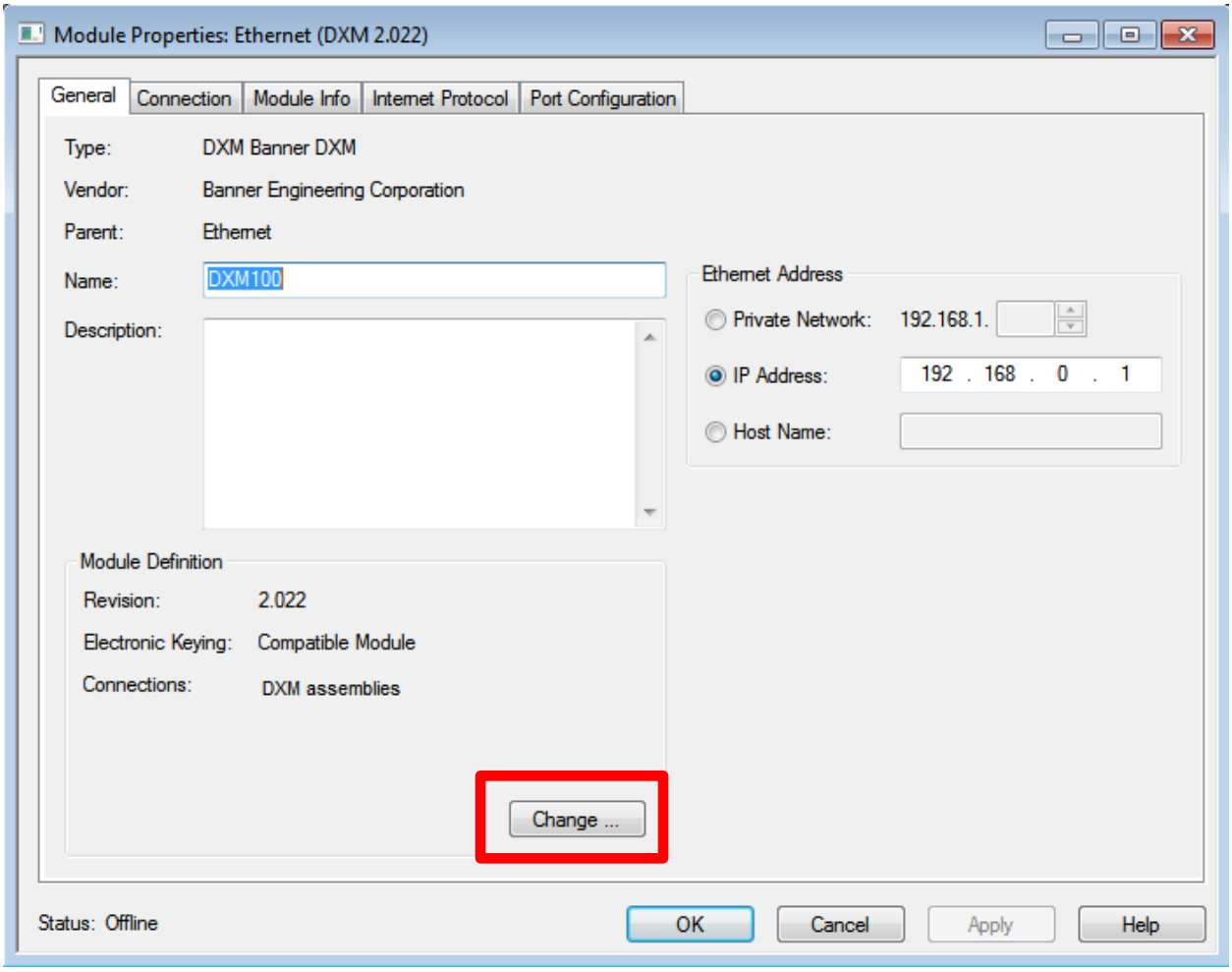
9. Now we will make a new module using the EDS file. Right click on the PLC's Ethernet adapter and choose "New Module".



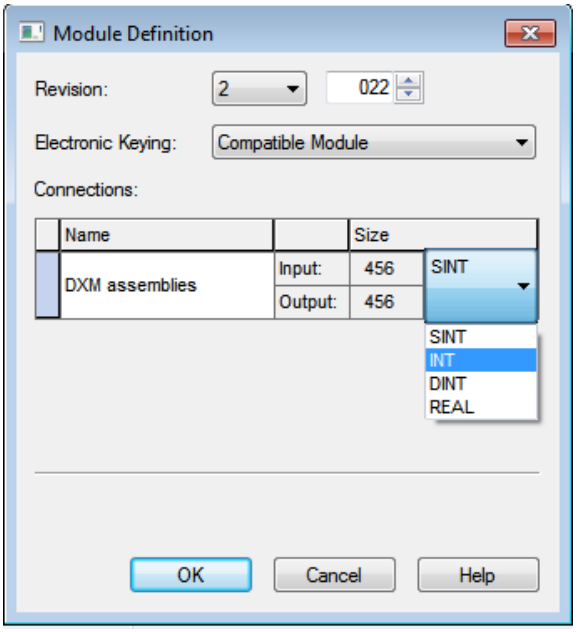
10. From the list, locate "DXM" then click Create.



11. Fill in a name, optional description, and IP address for the DXM. Then click the “Change” button in the Module Definition box.



12. Make sure to select “INT” as the data type.



13. Click OK, then OK again and download the program to the PLC. The connection will look like that seen below.