4B Ethernet Node AB PLC Software

This is a first draft of the document and the software and although it has been tested as far as possible, it may contain errors not yet identified.

Author	AM
Document Reference	-
Revision	1
Date	11/07/2018

Rev	Details	Modified	Checked	Date
1	First Release	AM	RG	30/11/2017
2	Changed device name & figures for the ENode data structure	AM	RG	21/06/2018
3	Added information related to Latching / Unlatching and Communication Update Counters	AM	RG	10/06/2018

Table of Contents

Information

The software supplied includes the following:

1) The complete PLC Software **– EthernetNode.ACD.**
2) The individual function ladder logic files mentioned below:
   - **MainProgram.L5X –** This routine calls the other routines listed below:
     - **MainRoutine.L5X –** This is main routine which calls the sub routine **(ENodeCommunication.L5X)** for the data decoding purpose.
     - **ENodeCommunication.L5X –** The sub-routine to call the Add-On-Instructions for the ENode Modules to decode the input data. Also, this routine checks the configuration flag of the module before executing the Add-On-Instructions.
     - **ENodeCommCounterUpdate.L5X** – The sub-routine to check the ENode communication for Digi and PIC based on counters.
     - **ENODE_AOI.L5X –** AOI routine to decode the Ethernet Module Data.
     - **ENode_DecToHex.L5X –** AOI routine to convert the data from decimal value to HEX.

This document is divided into two main sections:

- First section deals with establishing communication with up to two Ethernet Nodes by using the **EthernetNode.ACD** project.
- The second section deals with creation of a new project using the "**.L5X**" files detailed above.

This document is NOT comprehensive and assumes that the reader does have a familiarity with the AB CompactLogix PLC series if not the exact model in use.
The routines provided should work equally well with a ControlLogix PLC.

Section 1

This section describes the steps required to establish communication with up to two Ethernet Nodes by using the **EthernetNode.ACD** project.

1. Import EDS file

1.1. Open the EDS Hardware Installation Tool and Select **Add**.

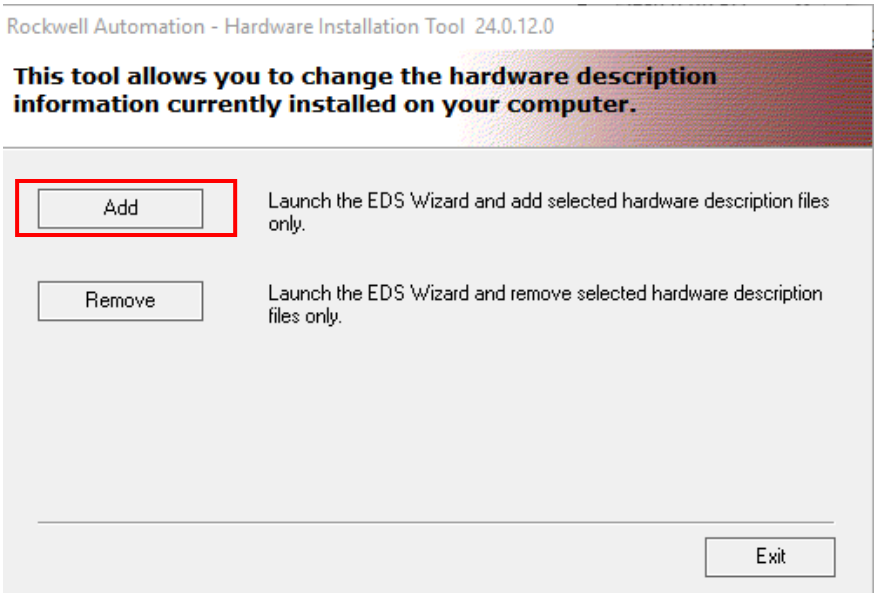

Figure 1 - Add EDS file in the PLC Module Database

1.2. Browse the EDS file form the saved location and press **Next**.

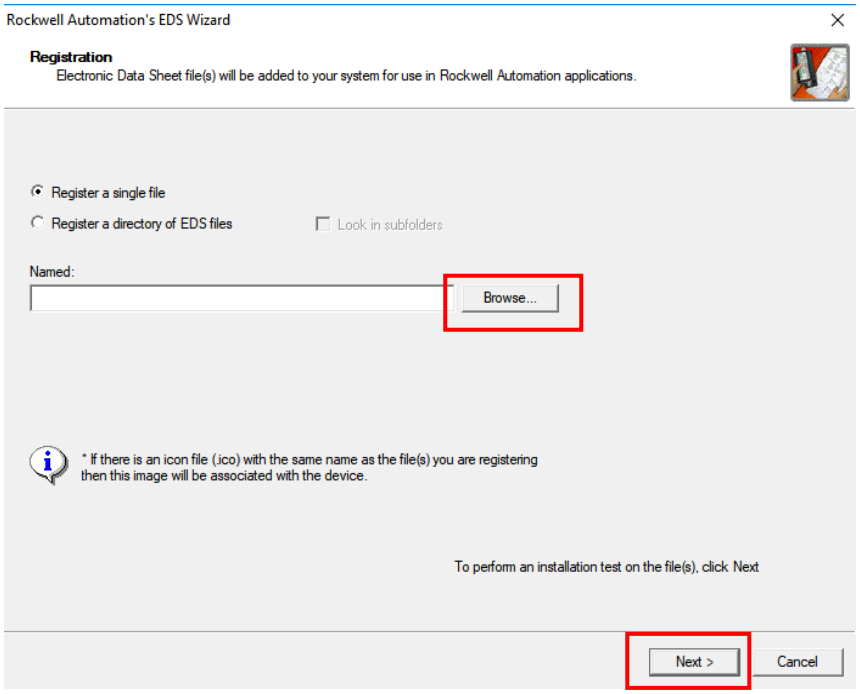

Figure 2 - Browse the EDS file and press Next

1.3. Follow the process through to add the EDS file in the PLC database for the Ethernet Node Controller.

2. Load the Existing Project

2.1. The simplest way to load the existing project is to double click the **EthernetNode.ACD** file. This will automatically open the project and the related routines in the Logix Designer.

2.2. The second way is to open the RSLogix 5000 studio and click **Existing Project** under the **Open** menu.

Figure 3 - Open Existing Project using Studio 5000

2.3. Please follow the process to browse and open the project into the Logix Designer.

3. Download Module into PLC and Run

3.1. Go into the relevant module properties (right click) and change the module IP address.

3.2. Go to the PLC Remote window and select **Offline → Go Online**.

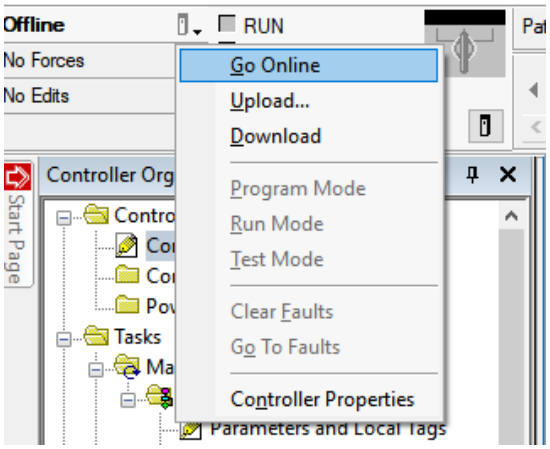

Figure 4 - Download Module in the PLC

3.3. On the next two screens press **Download.**

3.4. Go to the PLC Remote window and select **Run Mode.**

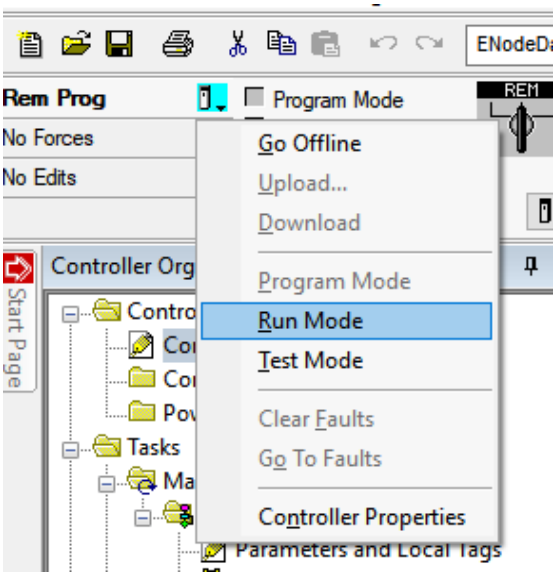

Figure 5 - Change PLC status to Run Mode

3.5. Press **Yes** on the warning Screen.

3.6. You can now look at the ENode Data in the relevant array location (under **Monitor Tags** Tab).

Section 2

This section details creation of a new project using the "**.L5X**" files provided with the **EthernetNode.ACD** project and adding a new Module in the PLC Logic Rung. The example shows the addition of ENode_1 but the same steps can be applied for addition of any module name.

4. Import the EDS File

Refer to the **"Import EDS File"** description in Section 1.

5. Create a New Project

If you have an older version of Logix Designer or are working on RSLogix you may need to create a new project and import the files separately. Please follow the steps given below to create a new project:

5.1. Open the Studio 5000 and click the **New Project** under the **Create** menu.

Figure 6 - Create a new Project

5.2. Select the PLC type you are using (in this project **1796-L19ERBB1B**) and enter the project name (PTO).

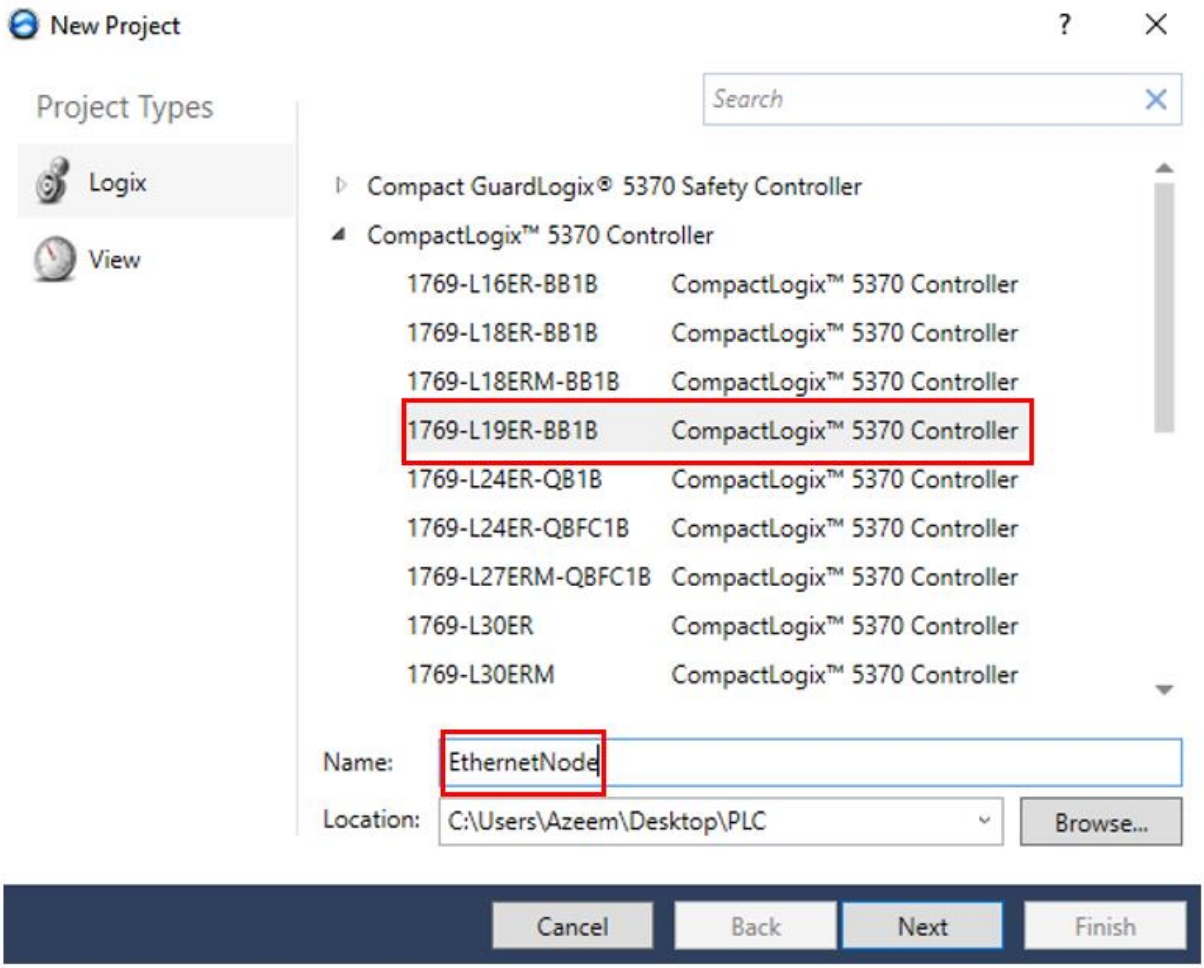

Figure 7 - Selecting PLC type and Project Name

5.3. The image below has the default settings that we used in creating the project, but you may have to choose the different settings depending upon the PLC in use. When you are ready click **Finish** (PTO).

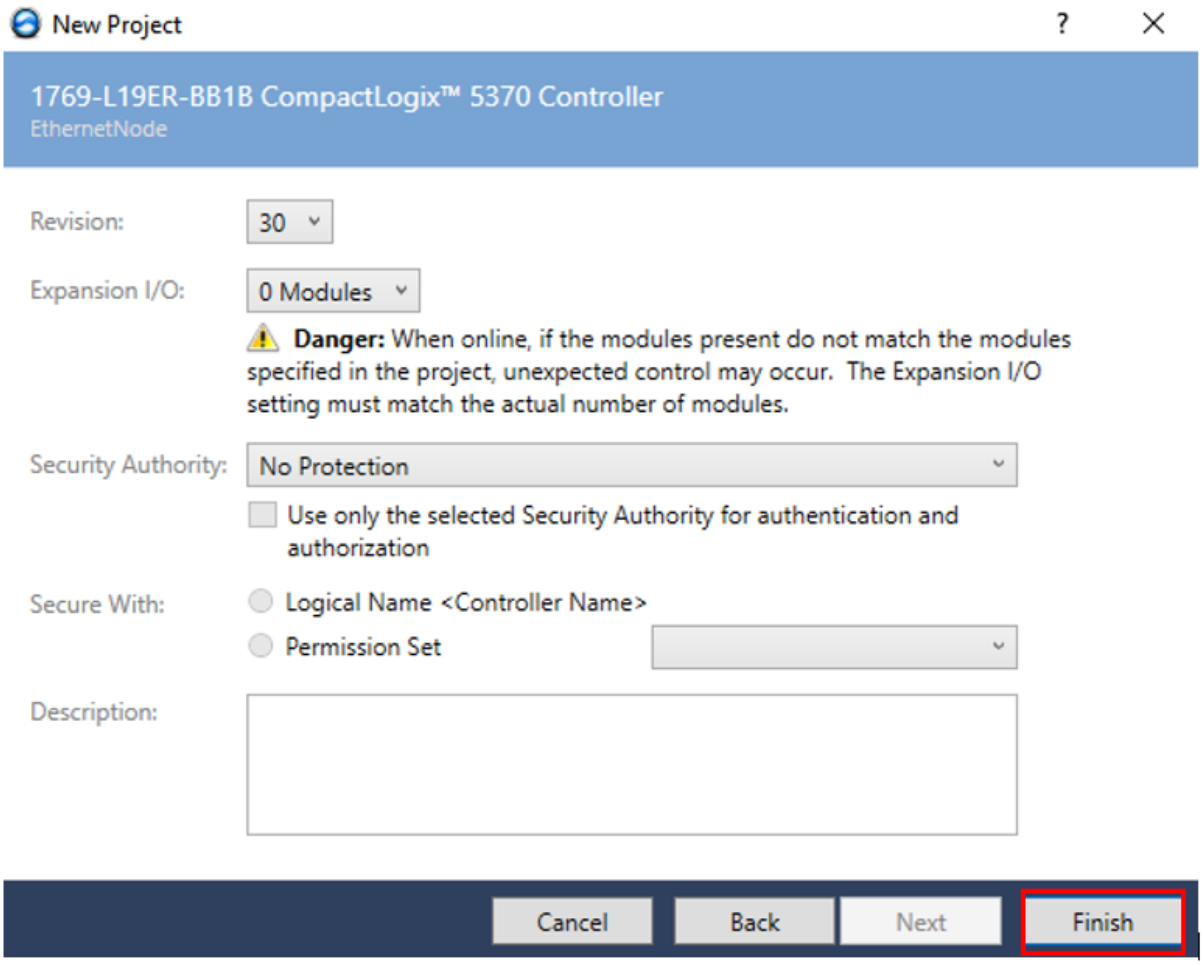

Figure 8 - Default settings for PLC project

The new project will look something like this.

Figure 9 - New Project Start Window

6. Add a New Module

By default, the EthernetNode.ACD project has two ENode modules already included in the Ethernet setup, a user can add more modules by using the information given in this section.

6.1. Right click on **Ethernet** (under I/O configuration) and select **New Module**.

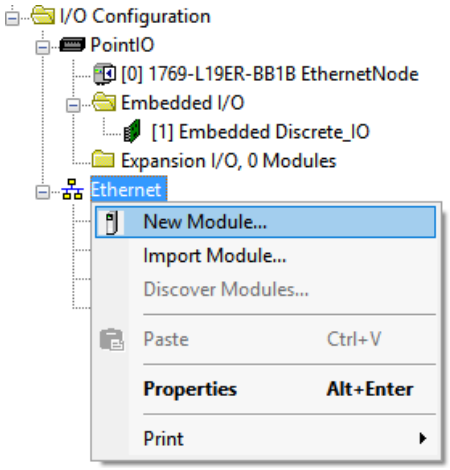

Figure 10 - Creating a new Module in Logix Designer

6.2. Enter 4B in the filter bar, this will bring up a list of modules that contain the characters 4B. Click on the entry **4B ETH Nodes** under the Catalog Number column and click **Create**.

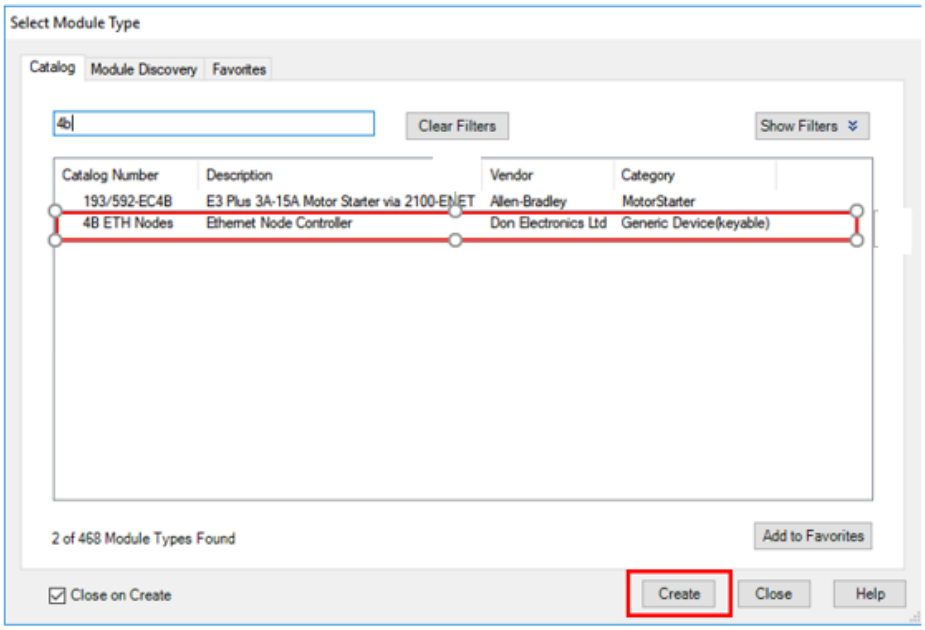

Figure 11 - Search Ethernet Node Controller using Catalog id

6.3. In this step please give the module a **Name** (ENode_1, ENode_2 etc) and assign an **IP address**. The IP address must match that assigned during the setup of the Ethernet Node.

Note: User can also add a module by simple Copy / Paste commands. Just Right on the Module and click Copy and to paste the module, Right Click on the Ethernet Icon and click Paste. Please change the IP address afterward as shown in the image below:

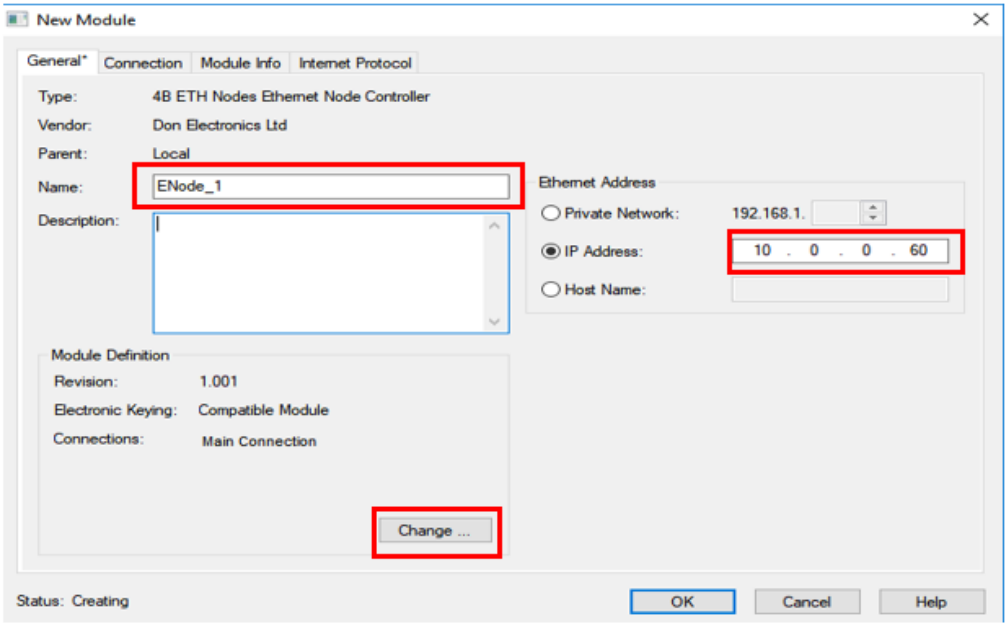

Figure 12 - Assign Name and IP Address

6.4. After assigning the Name and IP address please click on the **Change** and change the data type of **Incoming Data** from SINT to INT under the **Size** column. Then click **OK** (ignore the warning press **Yes**) and when you returned to the configuration window click **OK** to complete the configuration process.

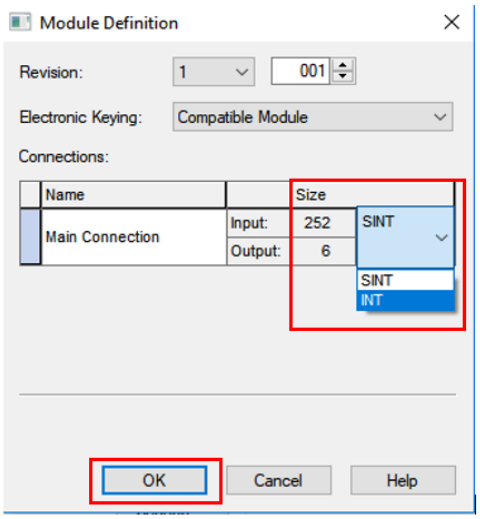

Figure 13 - Select Data type for incoming data

7. Import the Required files

7.1. Under the **Controller Organizer** window**,** right click on **MainTask** and select **Add→Import Program**.

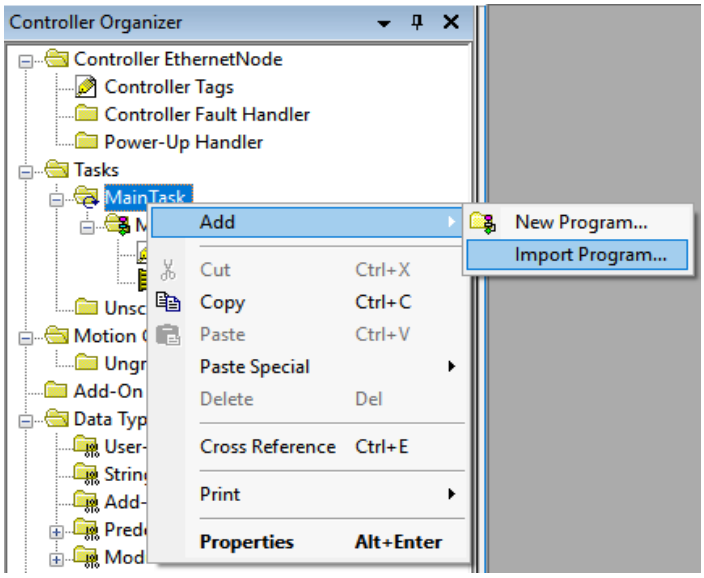

Figure 14 - Add Ethernet Project Main Task

7.2. Browse and find the **.L5X** files. Select **MainProgram.L5X** and click Open. This will import the file. Please make sure the **"Operation"** is set to **"Overwrite"** and Press **OK**.

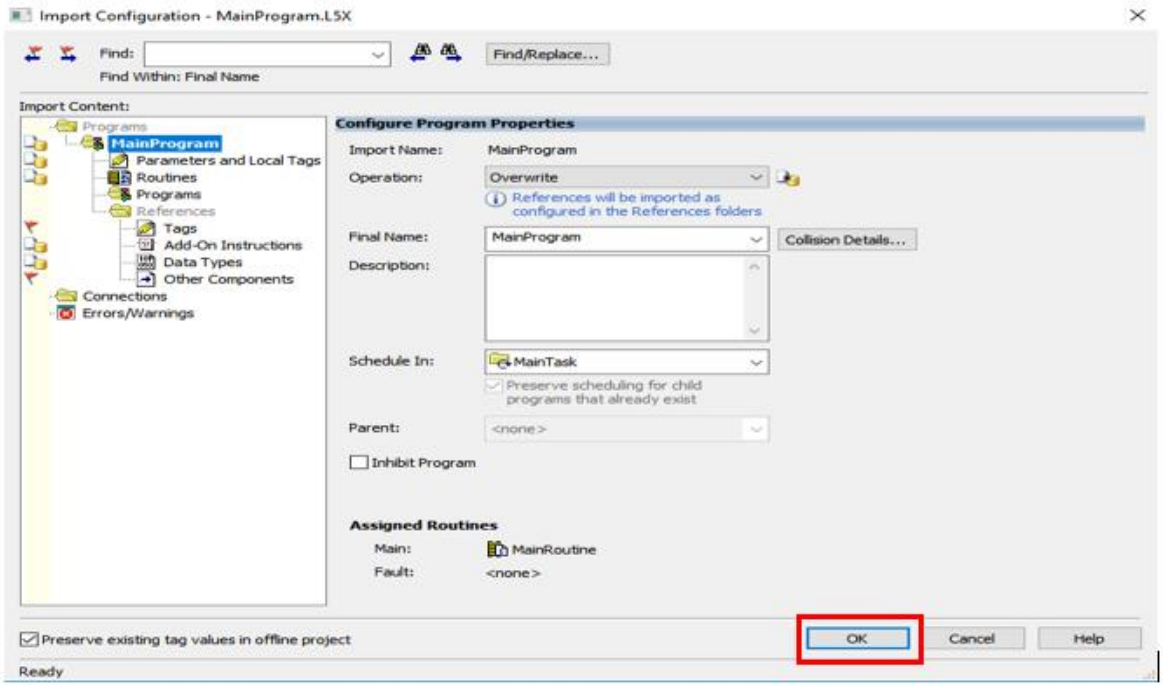

Figure 15 - Finish Import Process

During the importing of the **MainProgram.L5X**, the program will import the following routine and Add-On-Instructions:

- **ENodeCommunication.L5X**
- **ENODE_AOI.L5X**
- **ENode_DecToHex.L5X**

The Controller Organizer should look something like the image given below:

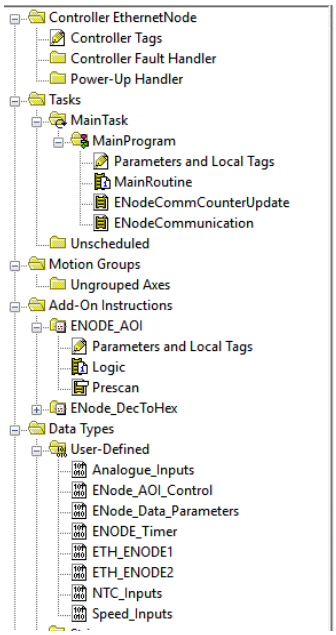

Figure 16 - Controller Organiser after importing MainProgram

8. Add a Module into the MainProgram Routine

After adding a new module into the Logix Designer, the user must add the module into the MainProgram routine for decoding the input data. Please follow the steps given below to add the module into the PLC routine:

8.1. Double click the **ENodeCommunication** routine under the **MainTask→MainProgram** menu.

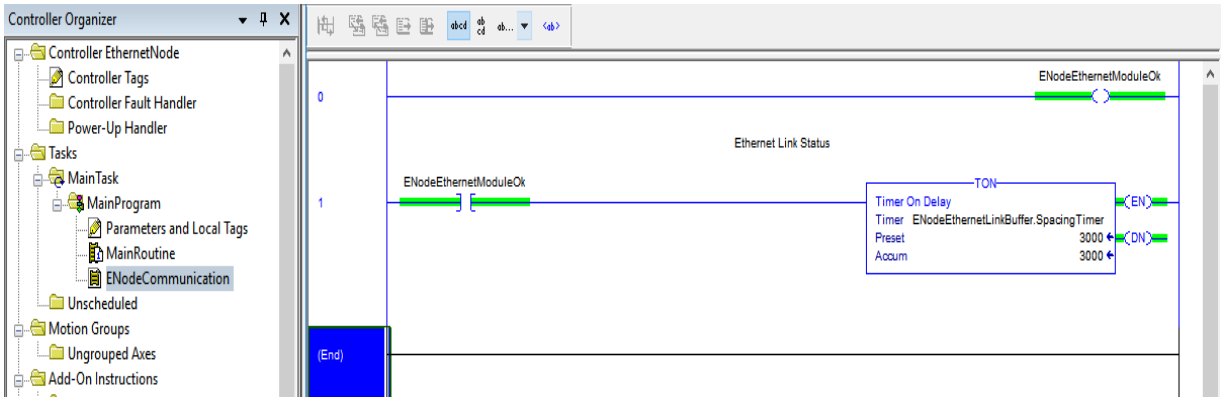

Figure 17 - ENodeCommunication window

8.2. Click on the **Rung** icon as highlighted in the image below or press **CTRL+R** keys to add a new Rung into the routine.

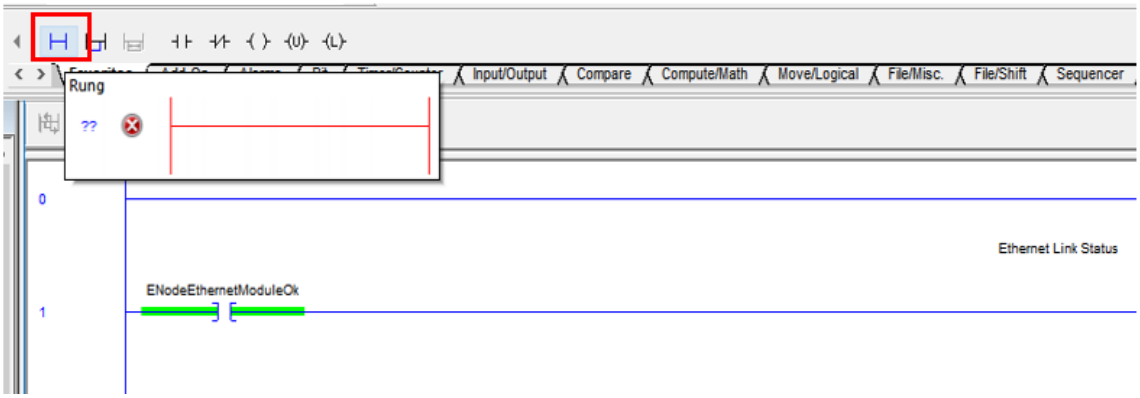

Figure 18 - Add a Rung by clicking on the Rung icon

8.3. Click the **Examine On** icon three times to add the Examine blocks to configure the Ethernet Module. Double click on the block and add the Configuration parameters (**ENodeEthernetLinkBuffer.SpacingTimer.DN**, **ENodeEthernetModuleOk**, **ENodeControl[index].Enable**) from the controller tags. The window should look something like the image given below:

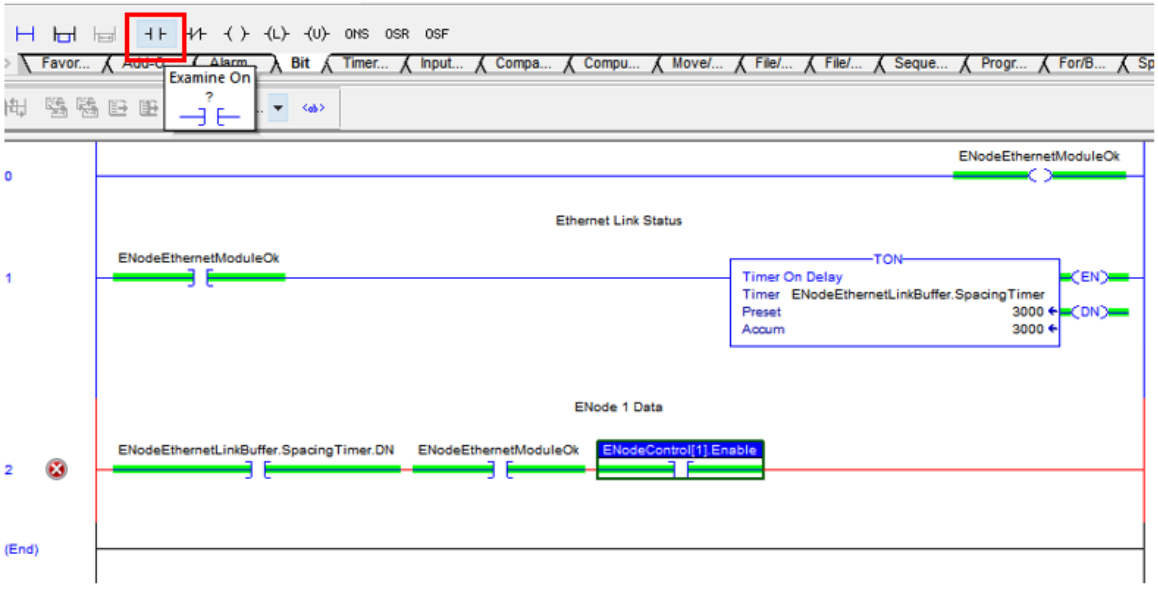

Figure 19 - Adding the Examine On gate to configure the Module

8.4. Click on the Add-On-Instruction Block and add the **ENODE_AOI** block in the Rung. Double click on each variable and add the required parameters into the AOI block. The Rung should look like this (PTO).

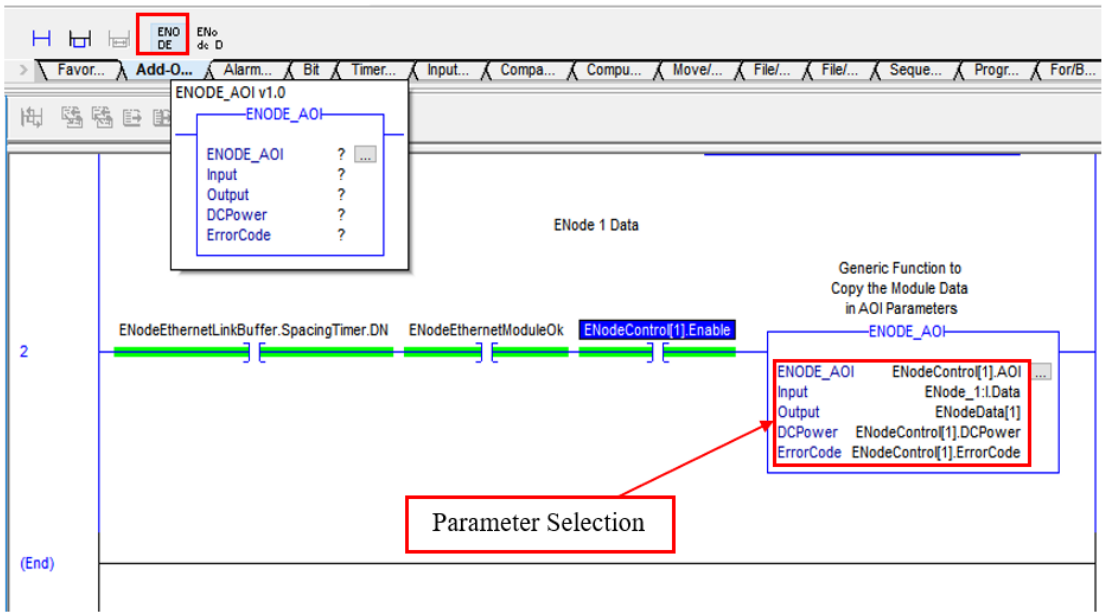

Figure 20 - Adding AOI for ENode_1 Module

8.5. The details of the **ENODE_AOI** block is given below:

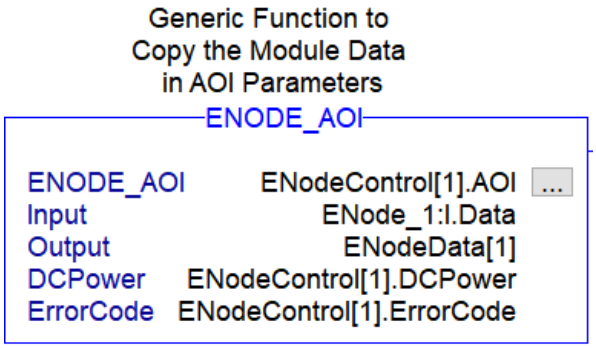

Figure 21 - ENODE_AOI Parameters

8.5.0. **ENODE_AOI - [ENodeControl[1].AOI]** → Calls the User-Defined parameters and assign the Input data in to the ENodeControl array index 1.

8.5.1. **Input – [ENode_1:I.Data]** → Input parameter gets the data from the Ethernet Node (ENode_1) Input Data Array.

8.5.1.1. **ENode_1** is the Module Name.

8.5.1.2. **I.Data** is the Input array of the specific Module.

8.5.2. **Output – ENodeData[1]** → The AOI decodes the raw data and adds the decoded data into the ENodeData array index 1.

8.5.3. **DCPower – ENodeControl[1].DCPower** → The parameter gets the decoded DC Power value of the module from the AOI and copies it into the **ENodeControl[1].DCPower** variable of ENodeControl array index 1.

8.5.4. **ErrorCode – ENodeControl[1].ErrorCode →** The parameter gets the decoded ErrorCode value of the module from the AOI and copies it into the **ENodeControl[1]. ErrorCode** variable of ENodeControl array index 1.

8.6. To pass the data into the AOI please make sure you have enabled the **ENodeControl[index].Enabled** flag for each module from the Controller Tags:

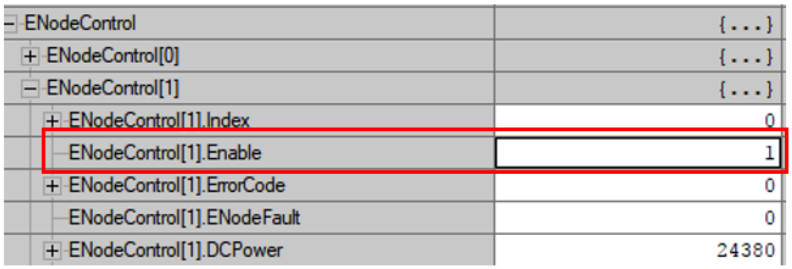

Figure 22 - Enable Module from the Control Array

Note:

- This Rung example will only assign the **ENode_1** Module Input data to the **ENode_AOI**. For additional ENodes user must add the Rungs and assign the parameters of the specific ENodes.
- The user can also copy and paste the Rung. Please make sure to change the required parameters of the Rung after adding/Copy-Paste the new Rung into the routine.
- After replicating the Rung in the ENodeCommunication routine for other modules please make sure to change the indices in the Rung code.

These images below explain how the rungs would look like for the two ENodes:

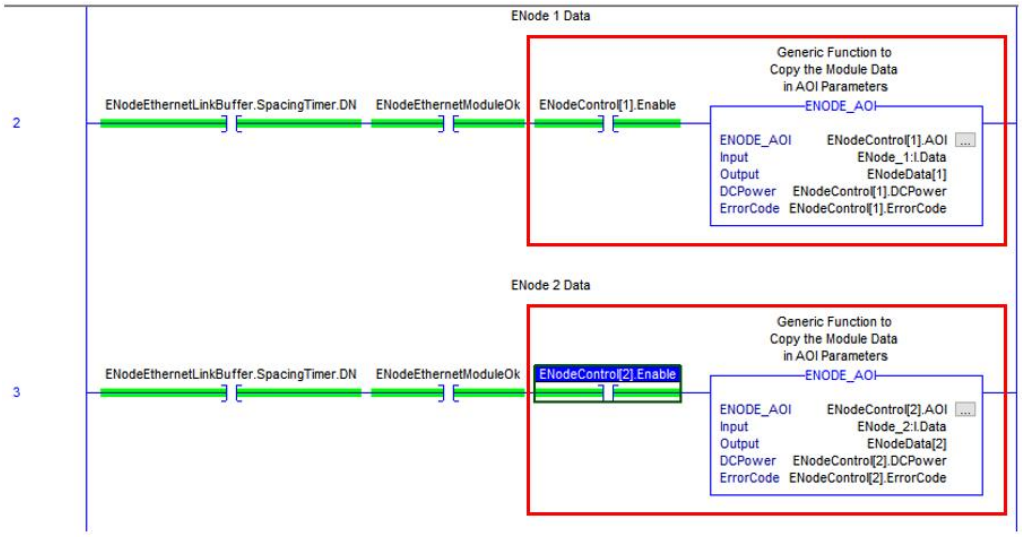

Figure 23 - Rungs Example for ENodes

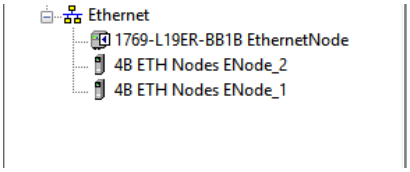

Figure 24 - Two ENodes in the PLC software

9. Download Module into PLC and Run

Refer to the "**Download Module into PLC and Run**" information in Section 1.

Note: The following sections are for additional information purpose.

10. Unregister EDS File

10.1. Please open the EDS Hardware Installation Tool and Select Remove.

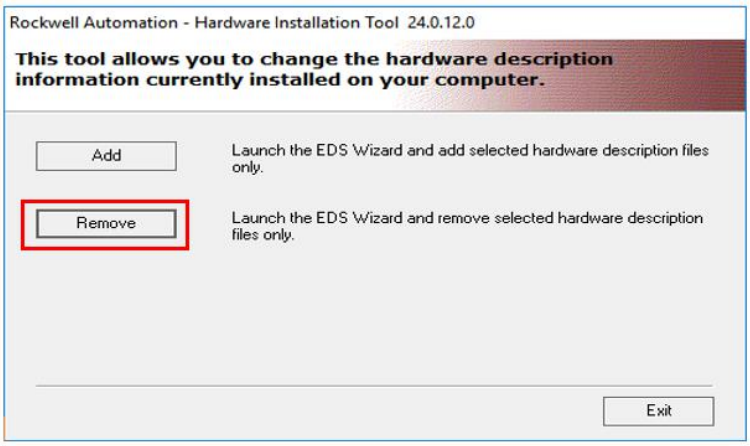

Figure 25 - Remove EDS file

10.2. Search for the **Ethernet Node Controller** and select the module.

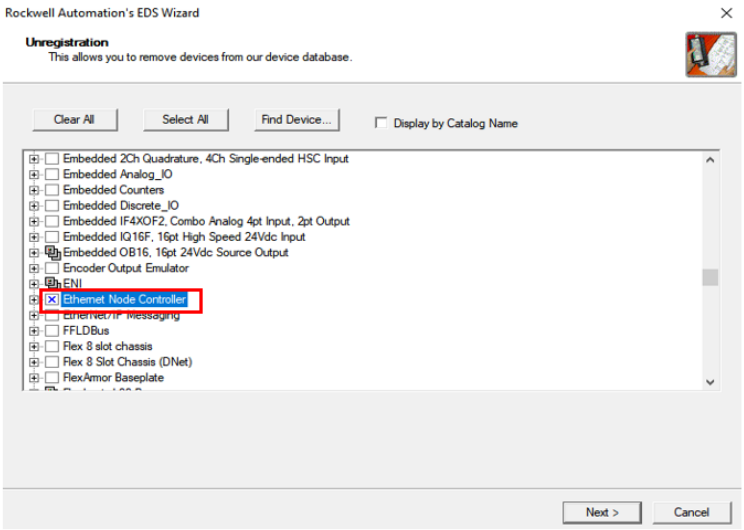

Figure 26 - Ethernet Node Module

10.3. Please select **Next** to unregister the current EDS installation.

11. Delete a Module

11.1. Please open the RSLogix Designer software to remove a module.

11.2. Select the specific module that needs to be removed and right click on it.

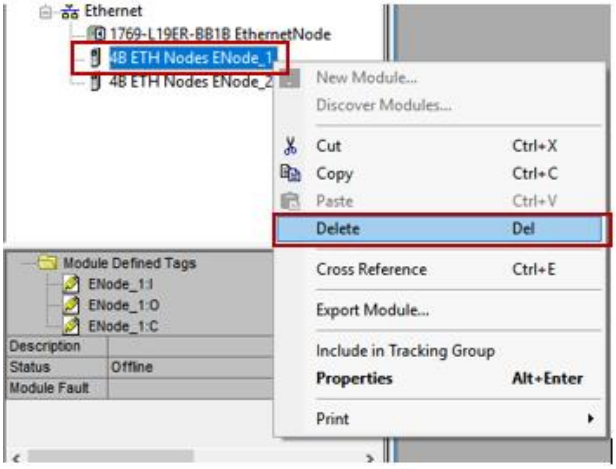

Figure 27 - Delete Module from PLC

11.3. Select **Delete** and press **Yes** to remove the module from the list.

12. ENode Data Structure

The images below are to describe the data structure for ENodes (ETH-NODEX):

ENodeData[1]	{...}
ENodeData[1].ENode_Name	''
ENodeData[1].ENode_Type	'ETH-NODE1'
ENodeData[1].ENode_Device	2
ENodeData[1].ETH_NODE_1	1
ENodeData[1].ETH_NODE_2	0
ENodeData[1].Node_ID	30
ENodeData[1].ProtocolVersion	1
ENodeData[1].ProtocolVersionUnknown	0
ENodeData[1].SerialNumber	98765432
ENodeData[1].SerialNumberString	'98765432'
ENodeData[1].PICVersion	2105
ENodeData[1].PICVersionString	'02.01.05'
ENodeData[1].DIGIVersion	201
ENodeData[1].DIGIVersionString	'00.02.01'
ENodeData[1].ExpansionVersion	0
ENodeData[1].ExpansionVersionString	'00.00.00'
ENodeData[1].RotarySwitch	3
ENodeData[1].DIPSwitch	129
ENodeData[1].DIPSwitch_EthernetIP	1
ENodeData[1].DIPSwitch_Profinet	0
ENodeData[1].DIPSwitch_Fahrenhiet	0
ENodeData[1].DIPSwitch_Bootloader	1
ENodeData[1].DIPSwitch_Celsius	1
ENodeData[1].DIPSwitch_DeviceID	1
ENodeData[1].ETH_ENODE_1	{...}
ENodeData[1].ETH_ENODE_2	{...}
ENodeData[1].ExpansionType	0
ENodeData[1].AnalogueBoard	0
ENodeData[1].NTCBoard	0
ENodeData[1].AnalogueBoardInputs	{...}
ENodeData[1].NTCBoardInputs	{...}

Figure 28 - ENode General Data Structure

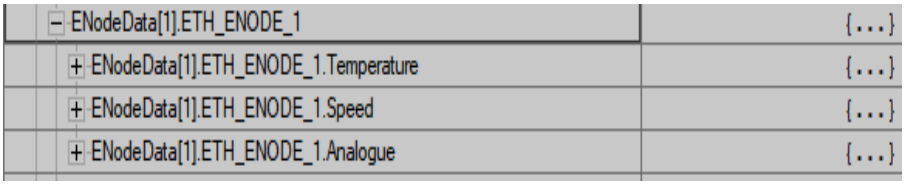

Figure 29 – ETH-NODE1 Data Structure

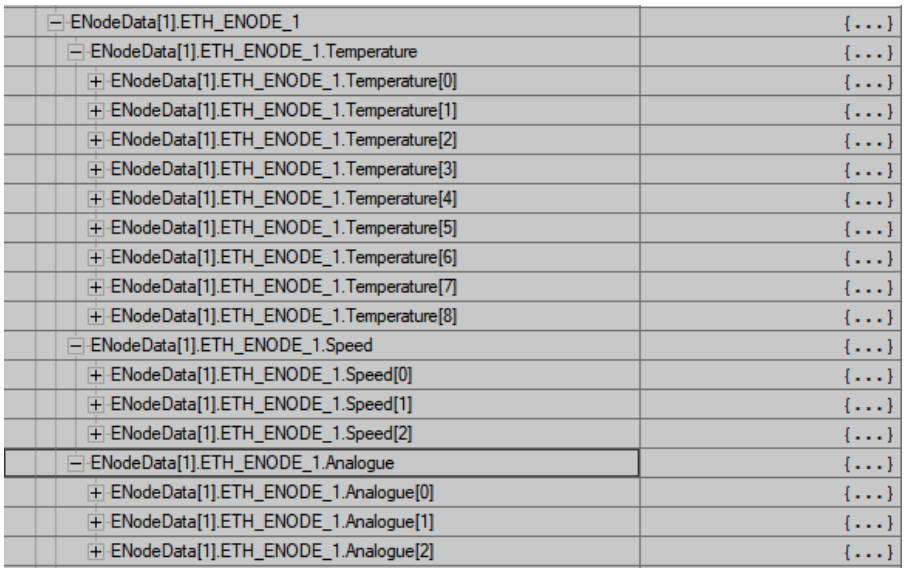

Figure 30 – ETH_NODE1 Inputs

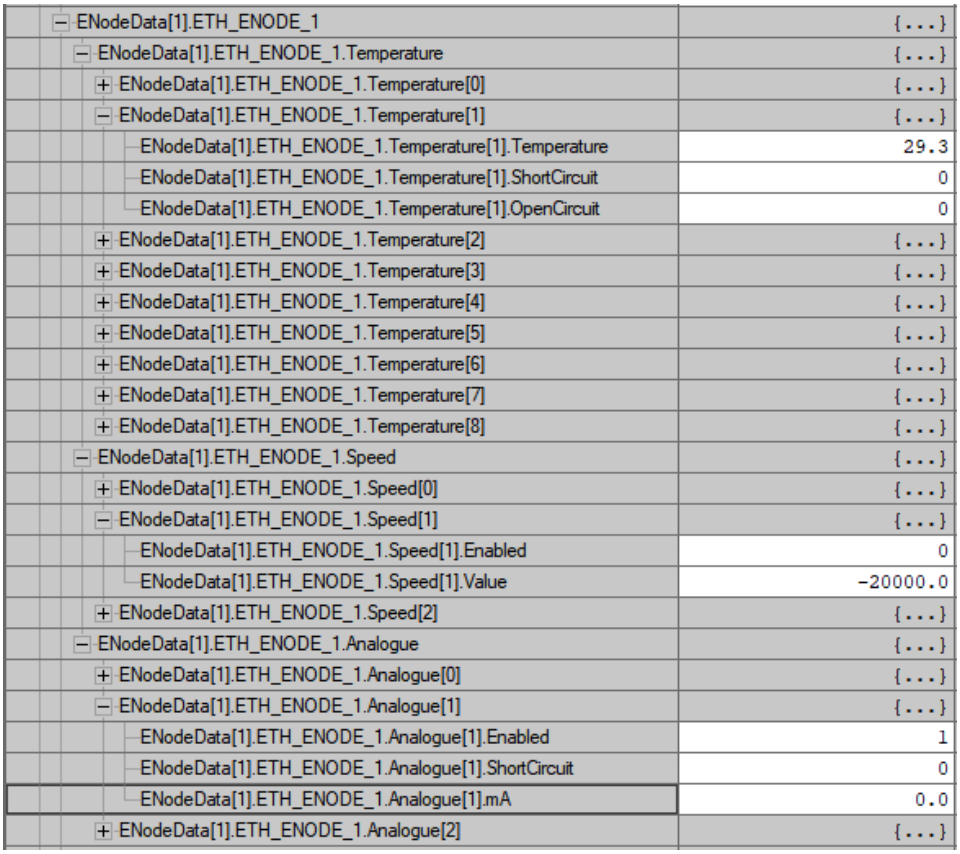

Figure 31 – ETH-NODE1 Parameters

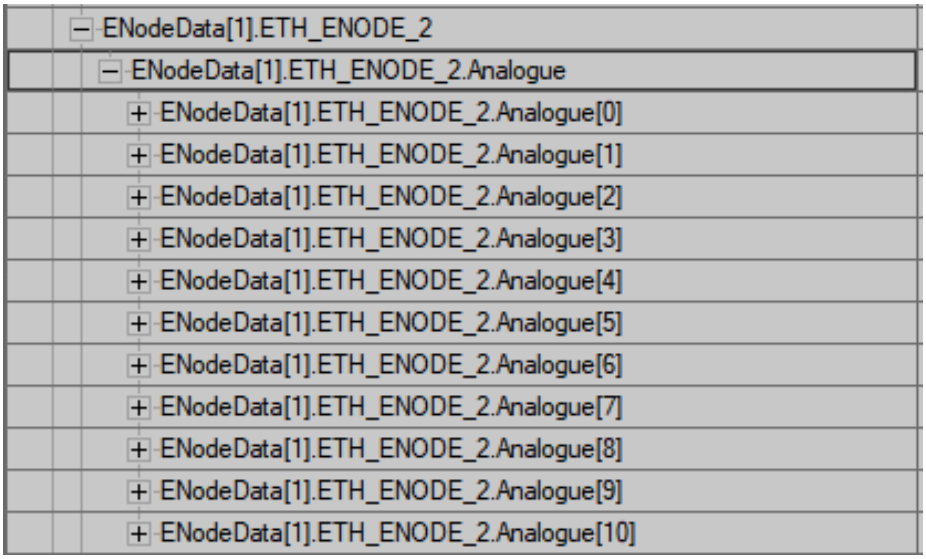

Figure 32 - ETH-NODE2 Inputs

ENodeData[1].ETH_ENODE_2	{...}
ENodeData[1].ETH_ENODE_2.Analogue	{...}
ENodeData[1].ETH_ENODE_2.Analogue[0]	{...}
ENodeData[1].ETH_ENODE_2.Analogue[1]	{...}
ENodeData[1].ETH_ENODE_2.Analogue[1].Enabled	0
ENodeData[1].ETH_ENODE_2.Analogue[1].ShortCircuit	0
ENodeData[1].ETH_ENODE_2.Analogue[1].mA	0.0
ENodeData[1].ETH_ENODE_2.Analogue[2]	{...}
ENodeData[1].ETH_ENODE_2.Analogue[3]	{...}
ENodeData[1].ETH_ENODE_2.Analogue[4]	{...}
ENodeData[1].ETH_ENODE_2.Analogue[5]	{...}
ENodeData[1].ETH_ENODE_2.Analogue[6]	{...}
ENodeData[1].ETH_ENODE_2.Analogue[7]	{...}
ENodeData[1].ETH_ENODE_2.Analogue[8]	{...}
ENodeData[1].ETH_ENODE_2.Analogue[9]	{...}
ENodeData[1].ETH_ENODE_2.Analogue[10]	{...}

Figure 33 - ENode 10CLI Parameters

ENodeData[1].AuxiliaryType	1
ENodeData[1].AuxiliaryEnabled	1
ENodeData[1].AnalogueBoard	0
ENodeData[1].NTCBoard	1
ENodeData[1].AnalogueBoardInputs	{...}
ENodeData[1].NTCBoardInputs	{...}

Figure 34 – ETH-NODEX Auxiliary Type

ENodeData[1].AnalogueBoardInputs	{...}
ENodeData[1].AnalogueBoardInputs[0]	{...}
ENodeData[1].AnalogueBoardInputs[1]	{...}
ENodeData[1].AnalogueBoardInputs[1].Enabled	1
ENodeData[1].AnalogueBoardInputs[1].ShortCircuit	0
ENodeData[1].AnalogueBoardInputs[1].mA	0.0
ENodeData[1].AnalogueBoardInputs[2]	{...}
ENodeData[1].AnalogueBoardInputs[3]	{...}
ENodeData[1].AnalogueBoardInputs[4]	{...}
ENodeData[1].AnalogueBoardInputs[5]	{...}
ENodeData[1].AnalogueBoardInputs[6]	{...}
ENodeData[1].NTCBoardInputs	{...}
ENodeData[1].NTCBoardInputs[0]	{...}
ENodeData[1].NTCBoardInputs[1]	{...}
ENodeData[1].NTCBoardInputs[1].Temperature	0.0
ENodeData[1].NTCBoardInputs[1].ShortCircuit	0
ENodeData[1].NTCBoardInputs[1].OpenCircuit	0
ENodeData[1].NTCBoardInputs[2]	{...}
ENodeData[1].NTCBoardInputs[3]	{...}
ENodeData[1].NTCBoardInputs[4]	{...}
ENodeData[1].NTCBoardInputs[5]	{...}
ENodeData[1].NTCBoardInputs[6]	{...}

Figure 35 - Expansion Inputs (NTC & Analogue)

13. ENode Controller Alarm Parameters

In the **MainProgram** a subroutine **(ENodeCommCounterUpdate.L5X)** has been set to make sure the System or Auxiliary is not in Alarm State and working as expected. This subroutine checks the heartbeat counters and set the flag to true if there is no communication. These flags are in two arrays and can be found in the **Controller Tags** (**ENodeTroubleShootingOutput_System & ENodeTroubleShootingOutput_Aux**). These flags can also be used for Latching / Unlatching the PLC Outputs as they are bit arrays. There are also two different tags into the ENodeControl[X] array to set the alarm tags for System and Auxiliary.

In the **ENodeCommCounterUpdate.L5X** routine you can see that the troubleshooting flags have been set for the maximum of 8 ENodes.

ENodeControl[1].SystemAlarm	0
ENodeControl[1].AuxAlarm	0

Figure 36 - ENodeControl Tags for System and Auxiliary

⊞ ENodeTroubleShootingOutput_System	{...}	{...}
⊞ ENodeTroubleShootingOutput_Aux	{...}	{...}

Figure 37 - Troubleshooting Boolean Tags for Latching / Unlatching Outputs

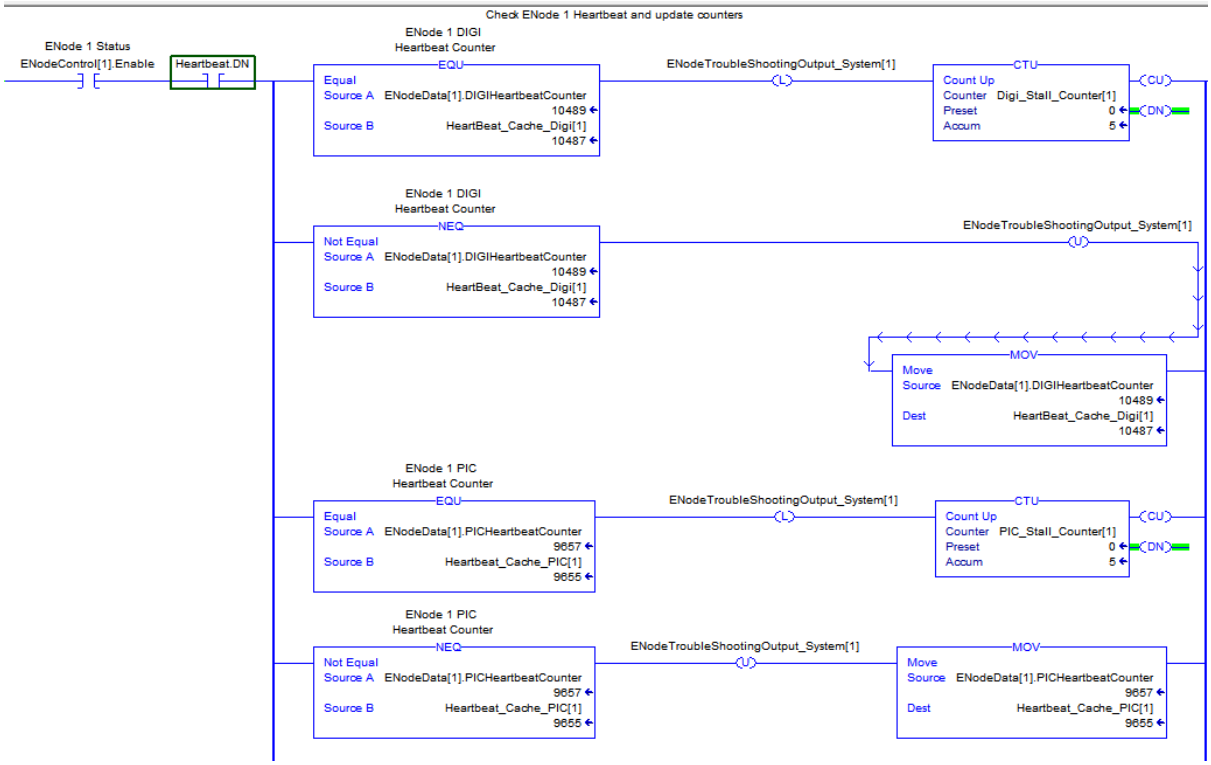

Figure 38 - Troubleshooting rung for ENode1 - Part 1

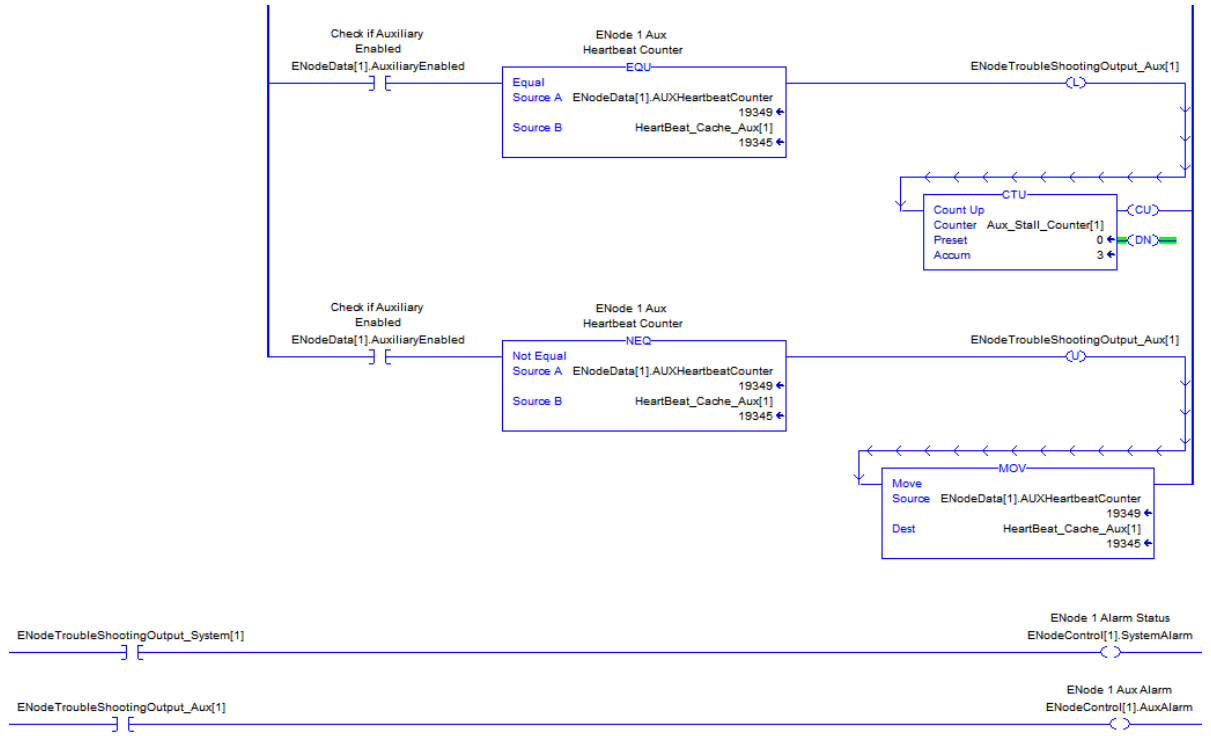

Figure 39 - Troubleshooting rung for ENode1 - Part 2

14. Conclusion

This application note has shown how to integrate the ENode Controller into an existing Allen Bradley CompactLogix series project or how to use the example project provided with this application note. The end user will have found all the information necessary to establish an Ethernet/IP connection and read the data in as well as basic data decoding example has been given.

In an event of any problems arising in connecting the ENode Controller to the PLC or for more information please contact your system provider or your 4B Group local sales support.

15. Project Breakdown

MainProgram → Contains MainRoutine, ENodeCommunication and ENodeCommCounterUpdate

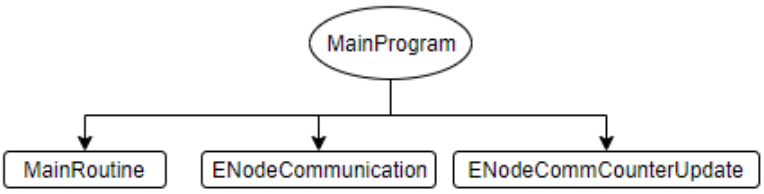

Figure 40 - Main Program Routines

Add-On-Instructions →Contains ENODE_AOI and ENode_DecToHex

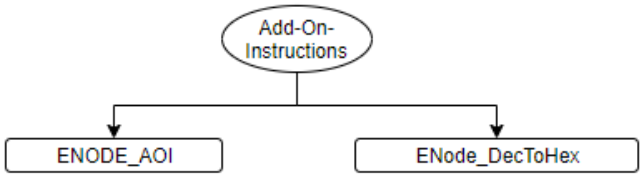

Figure 41 - Add-On-Instructions

User-Defined Parameters → Analogue_Inputs, ENode_AOI_Control, ENode_Data_Parameters, ENODE_Timer, ETH_ENODE1, ETH_ENODE2, NTC_Inputs, Speed_Inputs

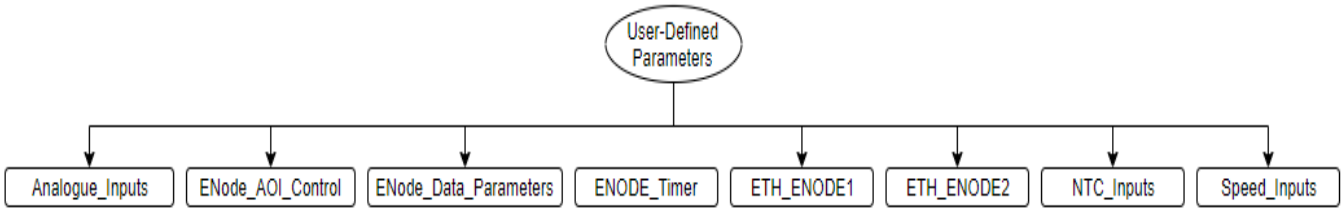

Figure 42 - User-Defined Parameters