PULSEROLLER

CONVEYLINX

Connecting ConveyLinx-Ai2 to Rockwell PLCs

Version 1.0

February 2019

For ConveyLinx Ai2 Firmware version 4.19 and higher

Publication *ERSC*-1521

SYMBOL CONVENTIONS

This symbol indicates that special attention should be paid in order to ensure correct use as well as to avoid danger, incorrect application of product, or potential for unexpected results

This symbol indicates important directions, notes, or other useful information for the proper use of the products and software described herein.

IMPORTANT USER INFORMATION

ConveyLinx Ai2 modules contain ESD (Electrostatic Discharge) sensitive parts and components. Static control precautions are required when installing, testing, servicing or replacing these modules. Component damage may result if ESD control procedures are not followed. If you are not familiar with static control procedures, reference any applicable ESD protection handbook. Basic guidelines are:

- Touch a grounded object to discharge potential static
- Wear an approved grounding wrist strap
- Do not touch connectors or pins on component boards
- Do not touch circuit components inside the equipment
- Use a static-safe workstation, if available
- Store the equipment in appropriate static-safe packaging when not in use

Because of the variety of uses for the products described in this publication, those responsible for the application and use of this control equipment must satisfy themselves that all necessary steps have been taken to assure that each application and use meets all performance and safety requirements, including any applicable laws, regulations, codes, and standards

The illustrations, charts, sample programs and layout examples shown in this guide are intended solely for purposes of example. Since there are many variables and requirements associated with any particular installation, Insight Automation Inc. does not assume responsibility or liability (to include intellectual property liability) for actual use based on the examples shown in this publication

SUMMARY OF CHANGES

The following table summarizes the changes and updates made to this document since the last revision

Revision	Date	Change / Update
1.0	February 2019	Initial Release

GLOBAL CONTACT INFORMATION

PULSE ROLLER

WWW.PULSEROLLER.COM
SALES@PULSEROLLER.COM
SUPPORT@PULSEROLLER.COM

TABLE OF CONTENTS

PREFACE

WHO SHOULD USE THIS MANUAL?

This manual is intended for users who need to utilize a Rockwell PLC equipped with Ethernet I/P capability to connect to a *ConveyLinx* Ethernet network to access module status and control conveyor operation.

PREREQUISITES

You should have reviewed and understood publication *ConveyLinx-Ai2 PLC Developer's Guide* (Insight Automation publication ERSC-1500) before utilizing this manual's instructions to physically connect your Rockwell PLC to a ConveyLinx network.

This manual also assumes you have a solid working knowledge of both Rockwell PLC's and the RSLogix 5000 / RSLogix Designer development environments.

NOT INCLUDED IN THIS MANUAL

Because system applications vary; this manual assumes users and application engineers have properly sized their PLC's Ethernet port capacity to accommodate the quantity of ConveyLinx module connections desired. Please refer to you particular PLC's specifications.

This manual is for ConveyLinx-Ai2 modules only. For information on how to connect ConveyLinx ERSC modules please see publication *Connecting ConveyLinx-Ai2 to Rockwell PLCs* (Insight Automation publication ERSC-1520)

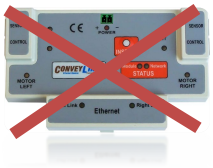
INTRODUCTION

This manual will provide instructions on how to connect your Rockwell Ethernet I/P capable PLC to a network of ConveyLinx modules. There are three basic methods for connecting ConveyLinx to Rockwell PLCs:

- **Use Generic Ethernet Device**
- **Import EDS and optionally import and use AOIs**
- **Use MSG Instruction**

All three methods can be used for ConveyLinx modules in ZPA mode and in PLC I/O mode. However, the MSG Instruction method does not maintain a constant connection to a ConveyLinx module and should not be used for "time critical" operations.

This manual is for ConveyLinx-Ai2 and NOT for ConveyLinx-ERSC. For information on how to work with ConveyLinx-ERSC, see publication ERSC-1520 Conecting ConveyLinx-ERSC to Rockwell PLCs



ETHERNET I/P GUIDELINES

Each Allen-Bradley PLC has 2 metrics for limiting maintained Ethernet I/P communications to remote devices:

- Fixed quantity of TCP connections available on its Ethernet Port
- Fixed quantity of I/O data table memory available for connected devices

If the limit of either of these quantities is reached, the PLC processor will indicate I/O communications fault on one or more instances of device declaration. For *ConveyLinx* device declarations utilizing either ZPA or PLC I/O Mode instances, in general the PLC limitation on TCP connections will be reached before I/O data table memory limit is realized.

For example, for a CompactLogix L3x series processor, the documented quantity of TCP connections available on its Ethernet Port is 32. The processor always keeps one TCP connection in reserve for programming terminal access, etc. An L3x series processor can accept 31 full-time *ConveyLinx C*onnections as generic I/O modules utilizing any combination of ZPA mode and PLC I/O Mode instances.

When a *ConveyLinx* module is attached as a "full-time generic I/O module" to the PLC, the connection is continually maintained and data is exchanged at a minimum RPI value (referred to as an implicit connection). If the PLC cannot communicate with the *ConveyLinx* module for any reason, the PLC's I/O tree will register a fault. It is possible for the PLC to communicate via Ethernet I/P with any *ConveyLinx* module it can physically reach over its Ethernet port without the module being "full-time connected as a generic I/O module". This is accomplished with a Logix5000 MSG instruction (referred to as explicit connection).

Reserve Ethernet I/P TCP connections for *ConveyLinx* modules in PLC I/O Mode and for key ZPA Mode modules where permanent accumulate/query/release functionality is required.
Use MSG Instruction to gather less time-critical data for things such as status and diagnostics.
For more information on determining the design and capacity of your Ethernet I/P network; please refer to Allen-Bradley document *EtherNet/IP Performance Application Solution* (Rockwell publication ENET-AP001D-EN-P).

SELECTING YOUR CONNECTION METHOD BASED UPON ASSEMBLY

As described in the *ConveyLinx-Ai2 PLC Developer's Guide* (publication ERSC 1520), the data that you exchange with your PLC and a given ConveyLinx module depends on the mode of the module and how you want to use it. The I/O data to be exchanged are arranged in register ***Assemblies*** and depending on the assembly, will dictate whether you can connect using the EDS file method or the Generic Ethernet Module method.

All available assemblies can be connected utilizing the Generic Ethernet Module method and only selected assemblies are available from the EDS file installation

All available assemblies can be connected utilizing the Generic Ethernet Module method. Only a selected few assemblies are available from the EDS file installation.

Assembly Pair	Available from EDS File Installation	Available as Generic Ethernet Module
ZPA Mode Assembly Inputs ZPA Mode Assembly Outputs	✓	✓
ZPA Mode Assembly Inputs with Reset Protection ZPA Mode Assembly Outputs with Reset Protection	✓	✓
Reduced Size ZPA Mode Assembly Inputs Reduced Size ZPA Mode Assembly Outputs		✓
Reduced Size ZPA Mode Assembly Inputs with Reset Protection Reduced Size ZPA Mode Assembly Outputs with Reset Protection		✓
PLC I/O Mode Assembly Inputs PLC I/O Mode Assembly Outputs	✓	✓
PLC I/O Mode Assembly Inputs with Reset Protection PLC I/O Mode Assembly Outputs with Reset Protection	✓	✓
Reduced Size PLC I/O Mode Assembly Inputs Reduced Size PLC I/O Mode Assembly Outputs		✓
Reduced Size PLC I/O Mode Assembly Inputs with Reset Protection Reduced Size PLC I/O Mode Assembly Outputs with Reset Protection		✓
ConveyLogix Assembly Inputs ConveyLogix Assembly Outputs	✓	✓

USING GENERIC ETHENRET MODULE METHOD

When using the Generic Ethernet Module construct in RSLogix 5000, you must supply configuration information about the device you are trying to connect. The following sections show the step by step procedure to connect a module for each set of Input and Output Assemblies described in publication ERSC-1506 *ConveyLinx-Ai2 PLC Developer's Guide.*

Each Assembly defined for ConveyLinx-Ai2 has a corresponding Assembly Instance value used when configuring your Generic Ethernet Module in your RSLogix 5000 environment. The following chart is a reference showing all the available Assemblies and their respective Instance Values used when connecting as a Generic Ethernet Device.

Assembly	Instance Values
ZPA Mode Assembly Inputs	105
ZPA Mode Assembly Outputs	106
ZPA Mode Assembly Inputs with Reset Protection	305
ZPA Mode Assembly Outputs with Reset Protection	306
Reduced Size ZPA Mode Assembly Inputs	119
Reduced Size ZPA Mode Assembly Outputs	120
Reduced Size ZPA Mode Assembly Inputs with Reset Protection	319
Reduced Size ZPA Mode Assembly Outputs with Reset Protection	320
PLC I/O Mode Assembly Inputs	107
PLC I/O Mode Assembly Outputs	108
PLC I/O Mode Assembly Inputs with Reset Protection	307
PLC I/O Mode Assembly Outputs with Reset Protection	308
Reduced Size PLC I/O Mode Assembly Inputs	117
Reduced Size PLC I/O Mode Assembly Outputs	118
Reduced Size PLC I/O Mode Assembly Inputs with Reset Protection	317
Reduced Size PLC I/O Mode Assembly Outputs with Reset Protection	318
ConveyLogix Assembly Inputs	121
ConveyLogix Assembly Outputs	122

General Procedure for Connecting using Generic Ethernet Module

All assembly pairs can be connected to a single ERSC using the same procedure within RSLogix 5000 environment:

1. Create a New Module in your Ethernet Tree
2. Select Generic Ethernet Module from the list of devices
3. Enter name and I.P. Address
4. Select the correct Comm Data type
5. Enter Input Assembly Instance Value and Size
6. Enter Output Assembly Instance Value and Size
7. Enter desired RPI value

For example, if you need to attach to 5 Ai2 modules that are in ZPA Mode, each module will have to have a unique name and I.P. address (step 3) and steps 4, 5, 6, and 7 will use the same values for each ERSC.

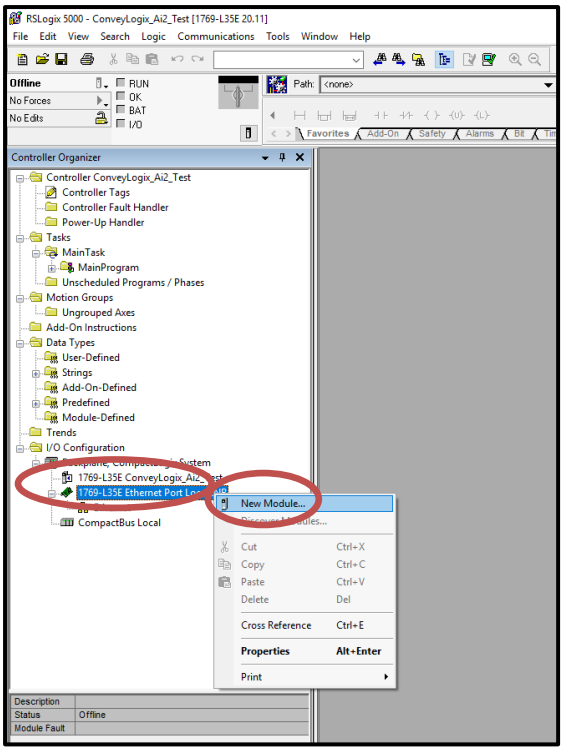
Example for Ai2 Module in ZPA Mode

This section will provide the set-by-step procedure for creating an instance of an *Ai2* module into the I/O configuration for an Allen-Bradley CompactLogix processor in RSLogix 5000 software.

Step #1

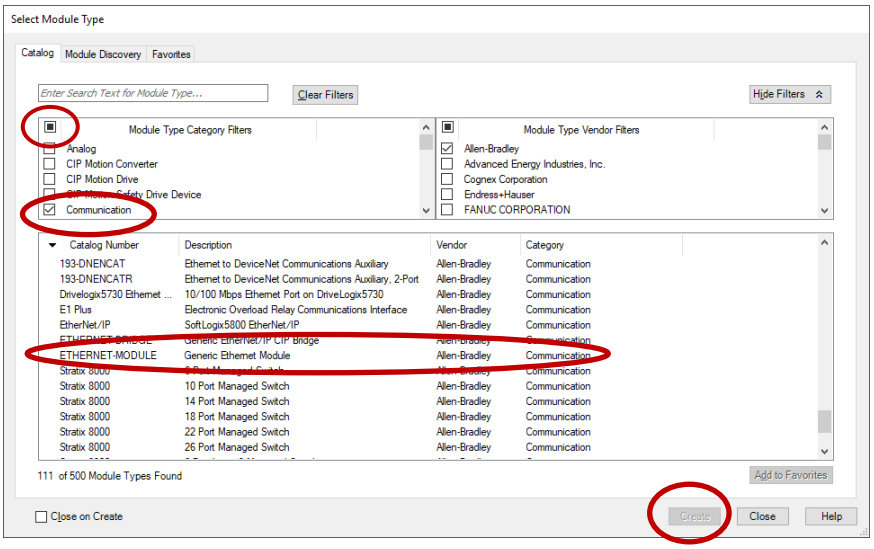
Add a New Module to the processor's I/O configuration by highlighting the processor's local Ethernet port in the I/O configuration tree.

Right-clicking will show the context menu. Select "New Module…"



Step #2

From the Select Module pop-up window, clear the filters, select the *Communication* check box, scroll to and select "Generic Ethernet Module" and click *Create*, which will open up the *New Module* window

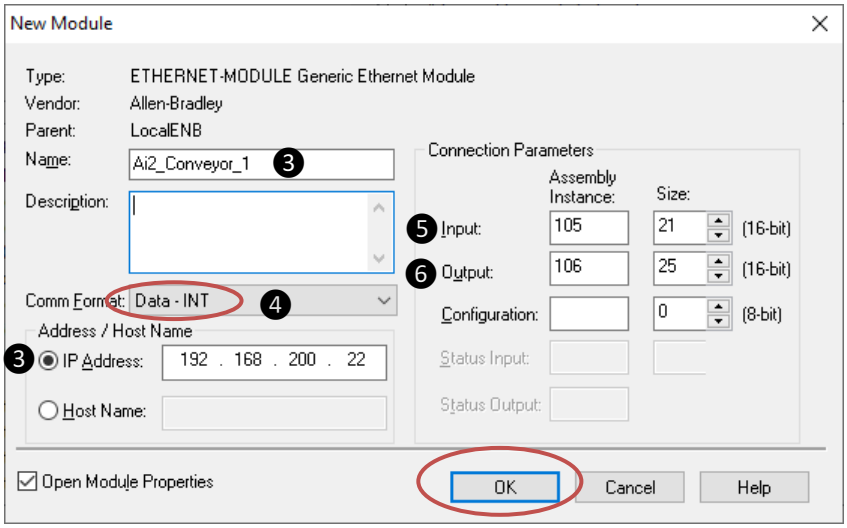


Step #3 thru #6

Fill in the *Name* and IP Address fields. The *Name* will be the *ModuleName* that will appear in your program Tag Database for any addressing.

Select Comm Format to be "Data – INT"

Fill in the Connection Parameters as shown. Configuration parameter is always Instance 1 and Size 0

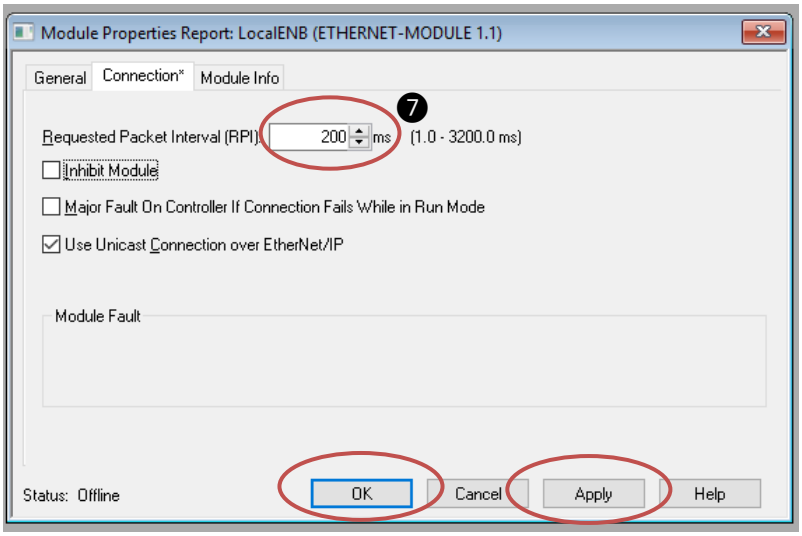


It is very important to select *Comm Format* data type to be INT or interface to Ai2 module will not operate correctly!

Step #7

Set RPI to a value no lower than 10ms. 200 ms is typical for ZPA Interface. You may also optionally select Unicast Connection.

Click “Apply” to update the value and then “OK” to exit the window.



Once you have completed the configuration of your Ai2 module, you can see the input and output registers in your Controller Tags screen. The register format and order within their respective Input/Output arrays match up exactly with the Assembly descriptions provided in the *ConveyLinx-Ai2 PLC Developer’s Guide* publication.

For our example, we created an Ai2 module named “Ai2_Conveyor_1”. Figure 1 and Figure 2 show the Input and Output register arrays for this module. You can access the data registers directly in your user program.

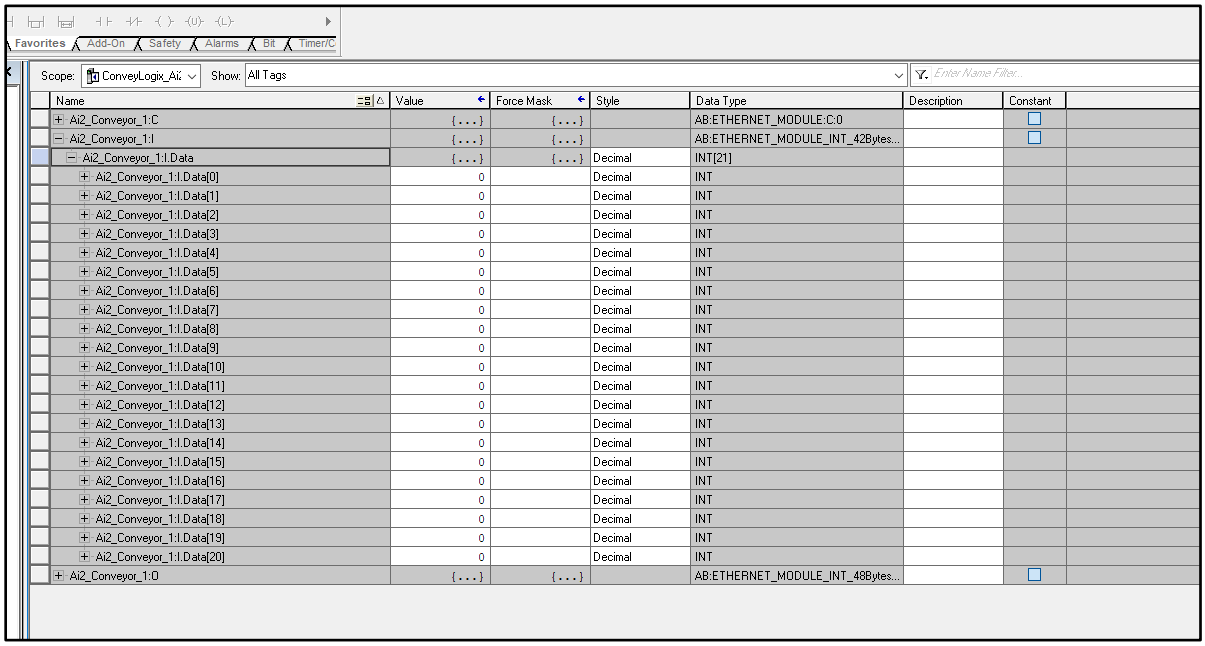


FIGURE 1 - GENERIC MODULE ERSC INPUT DATA ARRAY

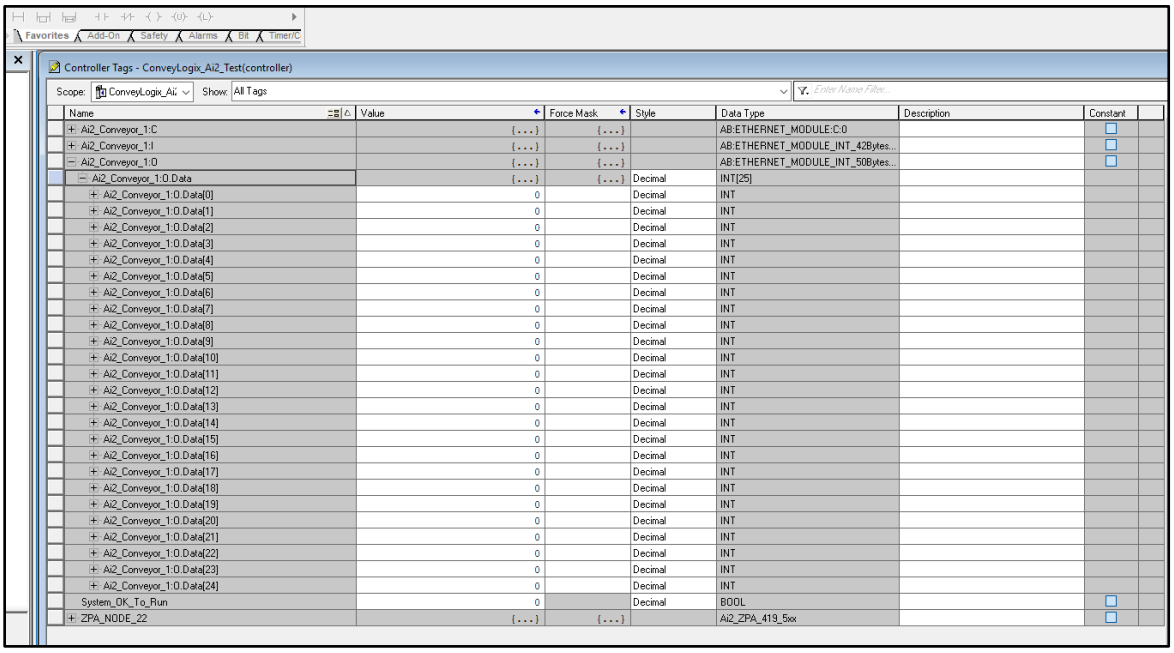


FIGURE 2 - GENERIC MODULE ERSC OUTPUT DATA ARRAY

PARAMETERS FOR EACH ASSEMBLY

Please note that for all Assemblies and all versions of firmware the Instance value for the "Configuration" parameter is always "1" and its size is always "0".

The following chart shows each available Assembly and its corresponding Input and Output Instance values and data sizes:

Assembly	Type	Instance Value	Size Value
ZPA Mode Assembly	Input	105	21
	Output	106	25
ZPA Mode Assembly with Reset Protection	Input	305	21
	Output	306	25
Reduced Size ZPA Mode Assembly	Input	119	12
	Output	120	15
Reduced Size ZPA Mode Assembly with Reset Protection	Input	319	12
	Output	320	15
PLC I/O Mode Assembly	Input	107	23
	Output	108	27
PLC I/O Mode Assembly with Reset Protection	Input	307	23
	Output	308	27
Reduced Size PLC I/O Mode Assembly	Input	117	9
	Output	118	9
Reduced Size PLC I/O Mode Assembly with Reset Protection	Input	317	9
	Output	318	9
ConveyLogix Assembly	Input	121	16
	Output	122	16

USING EDS FILE METHOD

SELECTING THE PROPER EDS FILE

The first step is to select the proper EDS file based upon the firmware version of your ConveyLinx-Ai2 modules. Our Pulseroller.com website contains all EDS files for download including older versions. The following chart lists firmware version, operation mode, and EDS file cross-reference information:

Ai2 Firmware	Operational Mode	EDS File
4.19 and later	**ZPA & PLC I/O Mode**	**ConveyLinxAi_V5_1.eds**

Please note that there may be updates since publication of this document. Please go to pulseroller.com to download the latest versions of EDS files when available.

For best results, you should remove any previous Ai2 EDS file(s) you may have installed in your RSLogix 5000 environment before installing the version described in this section.

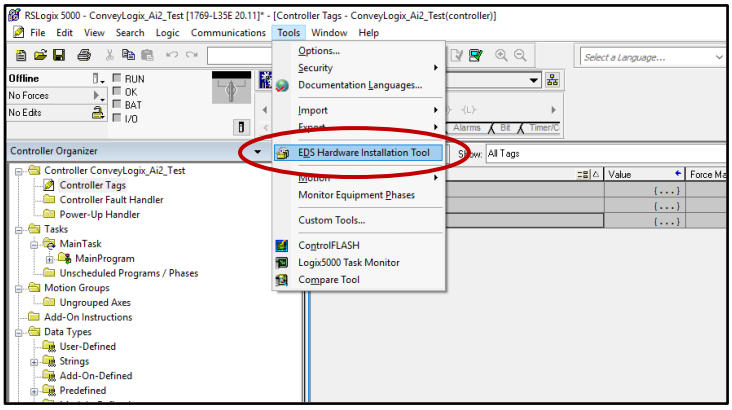
Also, delete all unused module data types from your program especially if you are modifying or starting with an existing program

Installing the EDS file provided by Insight Automation into your RSLogix 5000 environment will allow you to select the Ai2 module from your list of known devices without having to use the Generic Ethernet Module method. The EDS file contains the Instance and size parameters so you do not have to fill in this information. When you connect to an Ai2 module, the data is arranged in assembled registers as described in the *ConveyLinx-Ai2 PLC Developer's Guide* publication with the data appearing in your Controller Tags similarly to how the data appears when you connect to an Ai2 module as a Generic Ethernet Module.

INSTALLING CONVEYLINX EDS FILE INTO RSLOGIX5000

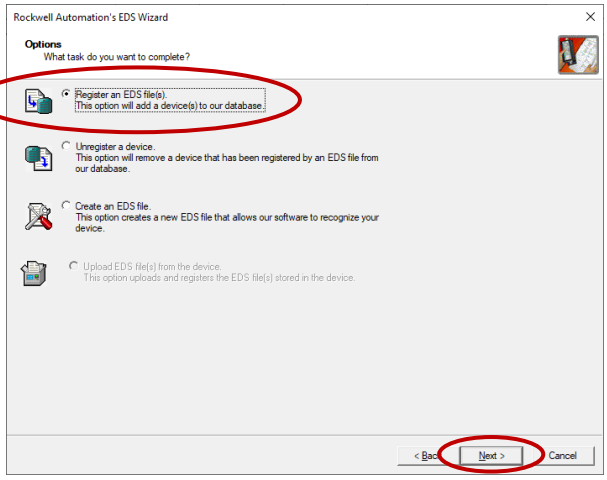
Step 1

With RSLogix5000 open, select *Tools* from the menu and *EDS Hardware Installation Tool*



Step 2

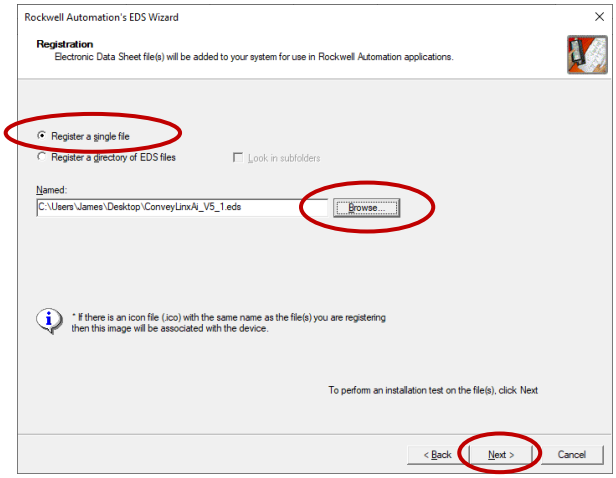
Select the Register an EDS file(s) radio button and click next



Step 3

Select the *Register a single file* radio button and click *Browse* and then browse to the location on your PC where you downloaded the EDS file. Click *Next* to continue.

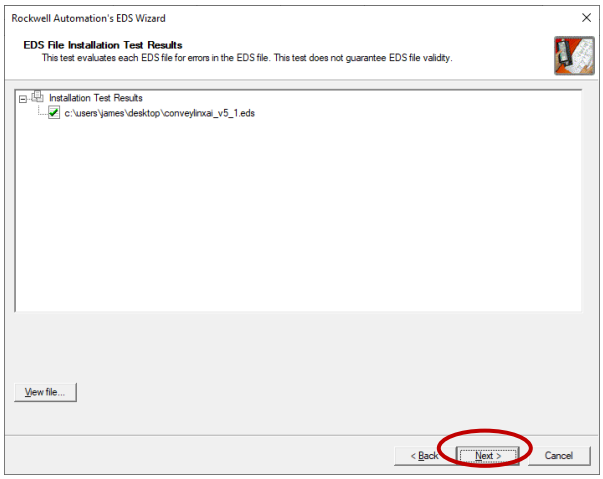
Note: Filename shown is for example only. The filename you select will be based upon the filename table shown at the beginning of this section.



Step 4

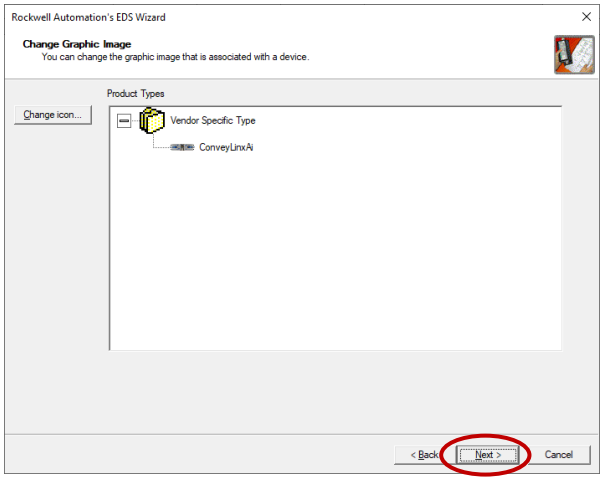
This window should appear with the green check indicating there were no errors. Click *Next* to continue.

Note: Filename shown is for example only. The filename you select will be based upon the filename table shown at the beginning of this section



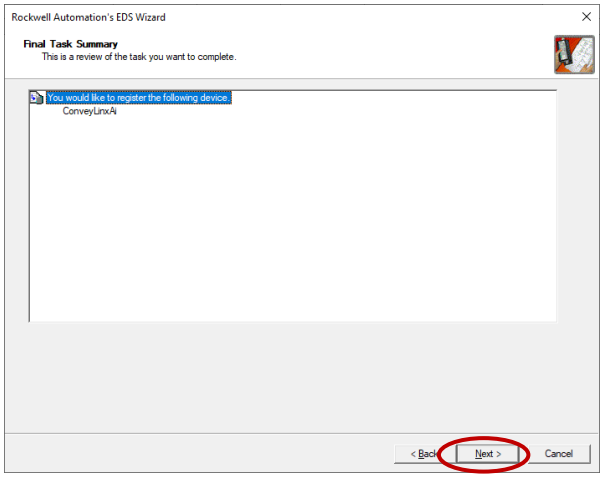
Step 5

A window appears indicating the graphic image included in the EDS file. This image will be used if you want to show network topology in RSNetworx. You can change to your own icon if you wish. Click *Next* to Continue



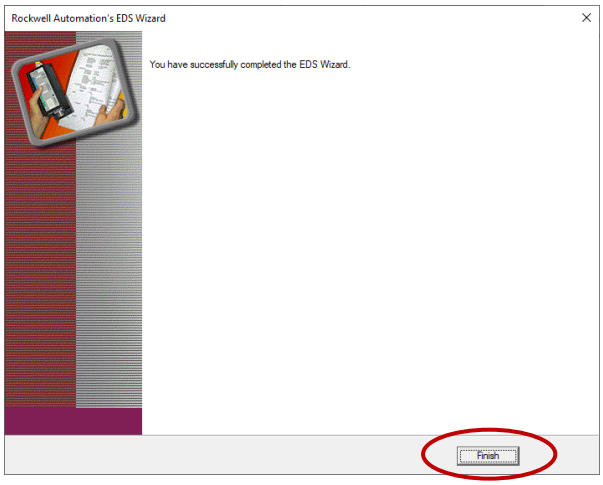
Step 6

RSLogix5000 asks if you want to complete the installation. Click *Next* to proceed.



Step 7

RSLogix5000 lets you know when it is done by showing this window. Click *Finish*.



Please refer to applicable Rockwell Software documentation for further details and information for removing and installing EDS files.

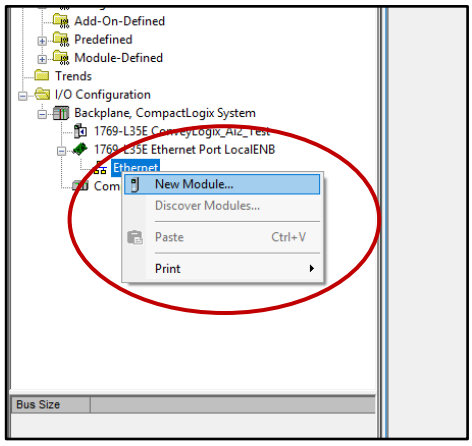
CREATING A ZPA MODE AI2 MODULE IN THE ETHERNET TREE

Once you have installed the EDS file into your RSLogix 5000 environment, you can now add specific instances of Ai2 modules into your project. You follow a similar procedure as described for the Generic Ethernet Module method.

We are going to show adding a ZPA mode Ai2 module to your program as an example.

Step 1

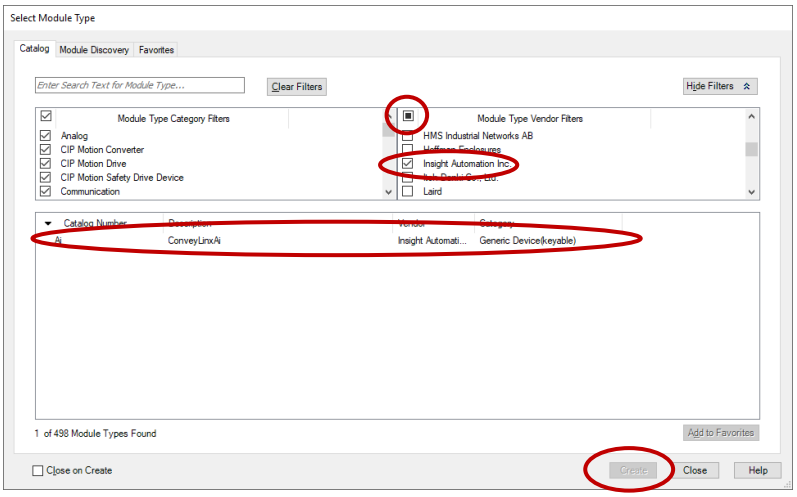
Right click on your Ethernet Tree and select *New Module* to open the *Select Module Type* window.



Step 2

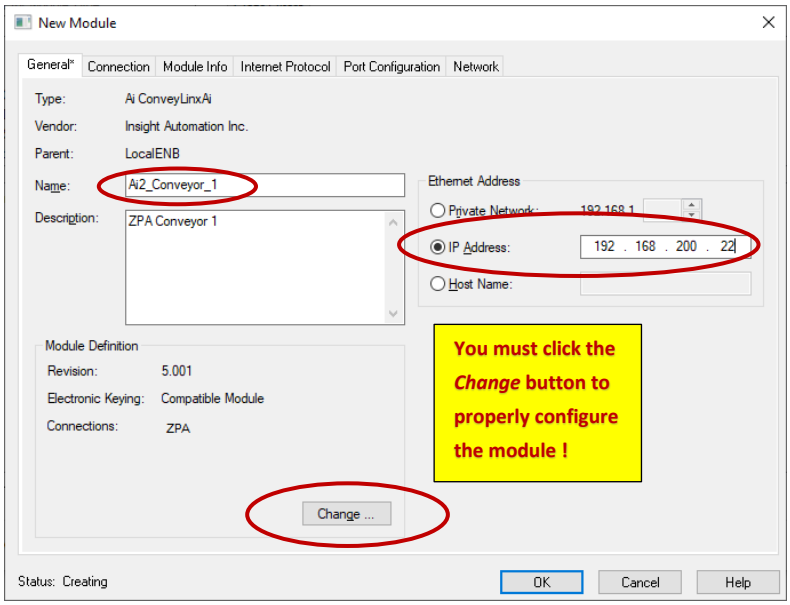
For this example we cleared the *Module Type Vendor Filters* and scrolled to find *Insight Automation* and checked the box thus showing only Insight Automation items. Then you select the *Ai* item, click the *Create* button to open the *New Module* window.

Note: Your list may look different depending on what devices / vendors you have already installed in your RSLogix5000 environment.



Step 3

For our example, we entered the Name and IP address information as shown. You can choose whatever name you desire and enter the proper IP address for your application. **Then you must click the *Change* button to open the *Module Definition* window.**

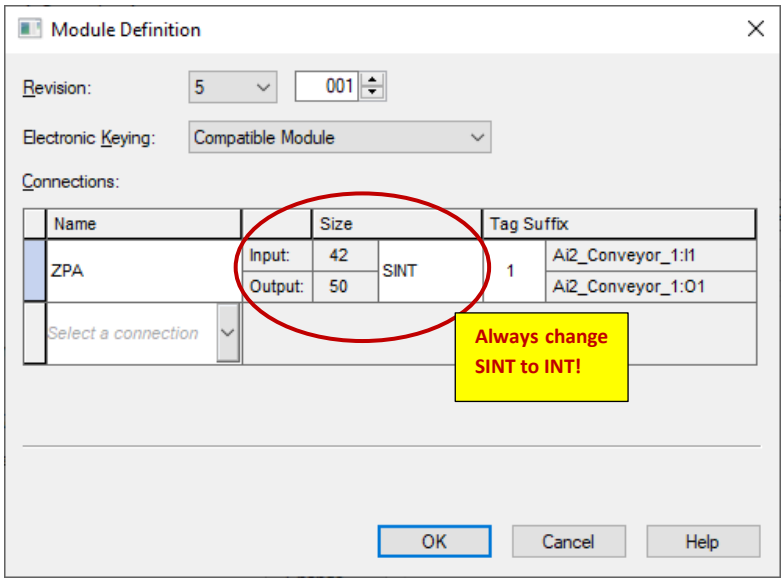


Step 4

In the Module Definition window you will see the default settings from the EDS file. The EDS file only allows SINT data type to be default. This needs to be changed to INT.

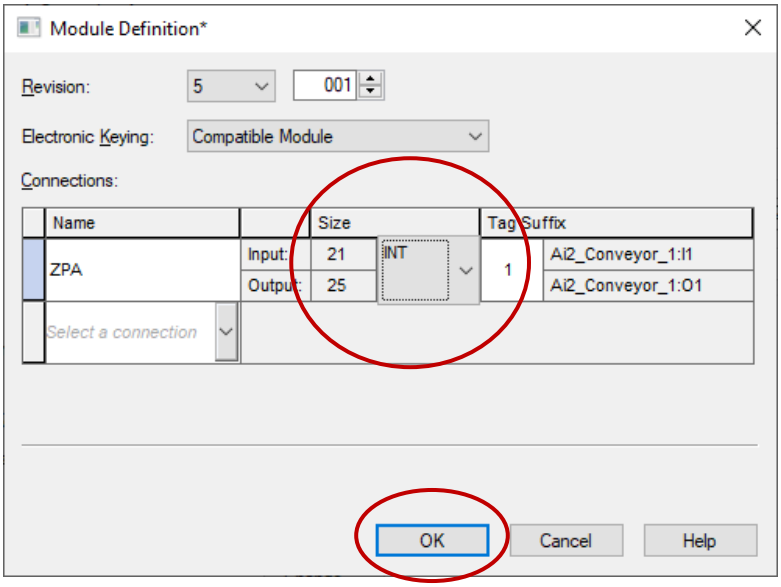
See Note ① below.

Note: Your Revision information shown may show a newer value than shown in this example based upon the EDS file you downloaded from the web site.



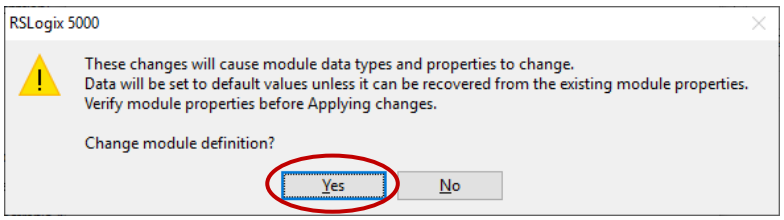
Click the right of the *Size* box to show a drop down box and then select INT

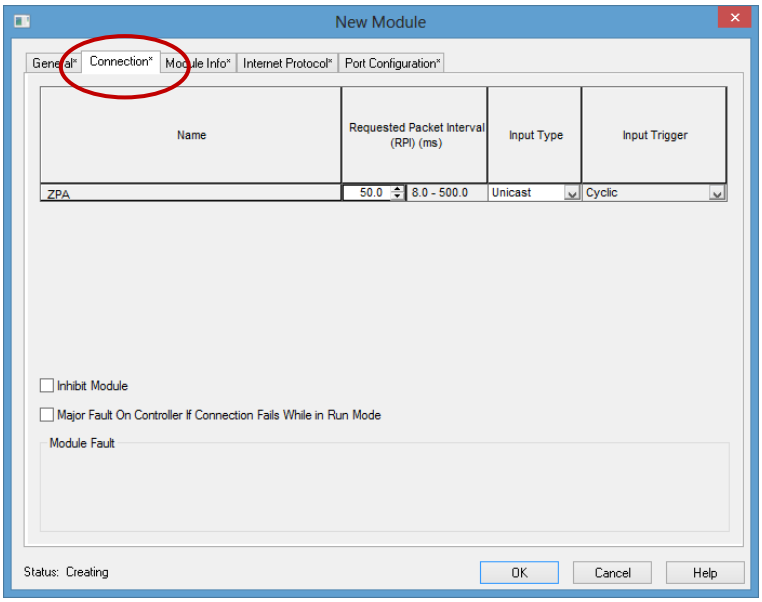
Once you have selected *INT* as the data size, you will notice the input and output sizes now reflect the register quantities as described in the *ConveyLinx-Ai2 PLC Developer's Guide* for ZPA mode. Click *OK*.



Step 5

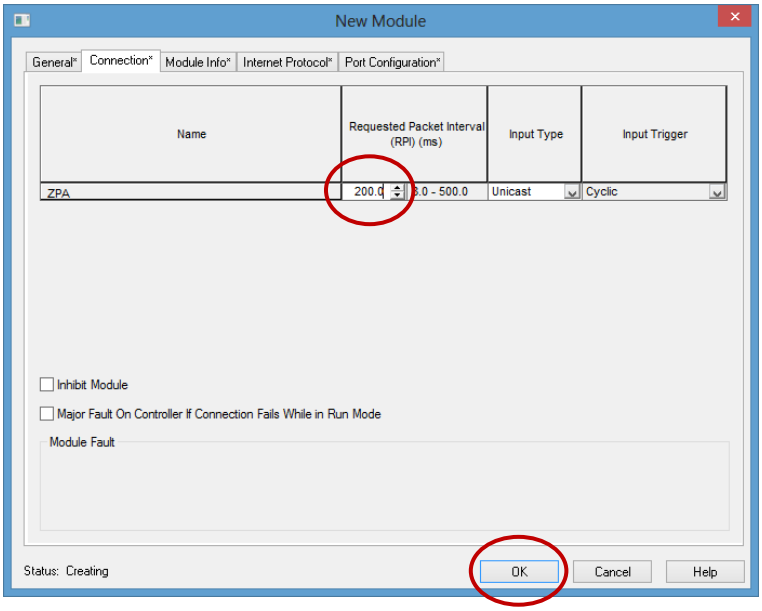
When you click *OK*, a warning will appear to tell you that you are changing the default parameters. Click *Yes*.





Step 6

You should be back to the New Module window. You can change the RPI of the connection to your Ai2 module by clicking the Connection tab.



In our example, we changed the RPI to 200 ms because we are in ZPA mode. Note that the EDS file limits your RPI range to between 8 and 500 ms.

Click OK and your Ai2 module is ready to use in your program

See Note ② - RPI Settings for more details.

For our example, Figure 3 and Figure 4 show the PLC's Controller tags generated when the Ai2 module was created. You can see that the quantities of INT registers correspond with the registers defined in the *ConveyLinx-Ai2 PLC Developer's Guide* for the ZPA Mode Assemblies.

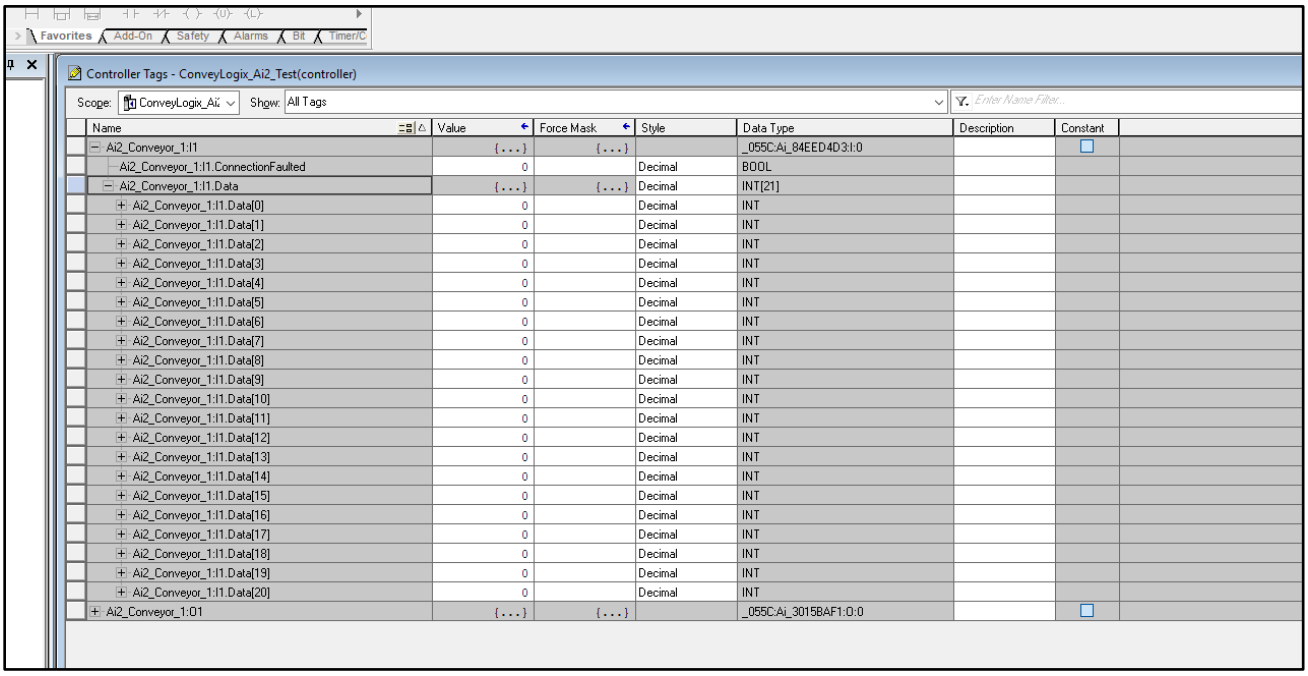


FIGURE 3 - ZPA MODE CONTROLLER TAGS FOR INPUTS

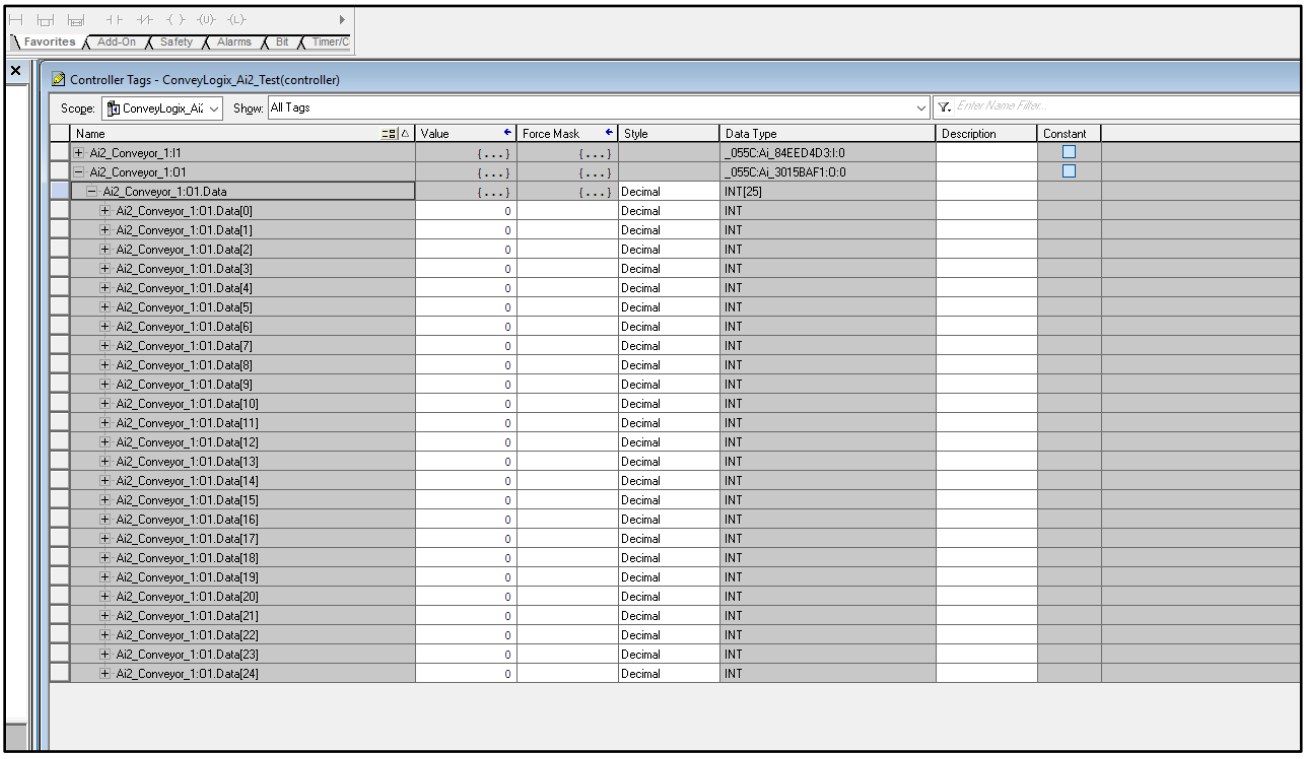


FIGURE 4 - ZPA MODE CONTROLLER TAGS FOR OUTPUTS

NOTE ① - DATA TYPE SIZE

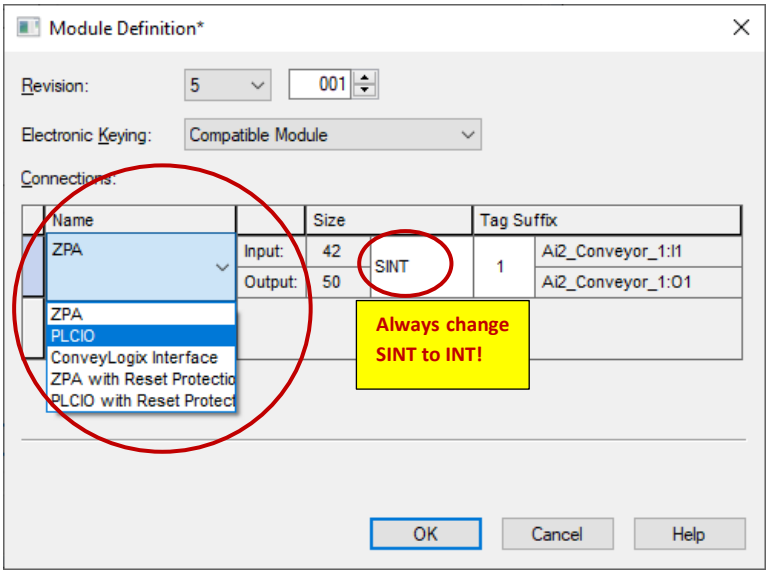
As noted, the EDS specification only allows for SINT data type as default. You can leave SINT as the default data type if this fits your particular programming preferences. However, keep in mind the documented register types in the *ConveyLinx-Ai2 PLC Developer's Guide* are described as 16 bit INT and this could lead to cross-referencing confusion. Furthermore, if you also wish to use Insight Automation's **Add On Instructions** (AOIs – described later in this document), you **must change the data type to INT** because these items are written expecting INT data type.

NOTE ② - RPI SETTINGS

Please note that RPI settings do not affect the Ai2 module nearly as much as it affects the PLC Ethernet port's throughput. A combination of the quantity of module connections (of all types not just Ai2 modules) along with small RPI values can create a bottleneck at the PLC's Ethernet port. A higher quantity of device connections coupled with a small RPI for each can result in dropped connections to devices (Ai2 and other connected Ethernet devices). This is not an issue with the Ai2 module (or other device); it is an issue with the PLC. It is always recommended to use the largest RPI value you can for a given connection while maintaining reasonable device response. For example, 10 msec RPI for a module in ZPA mode will not necessarily produce noticeable operation difference when compared to 100 msec. The rule of thumb is to reserve your 10 msec RPI settings for PLC I/O modules only.

Creating Other Connection Types

The steps are basically the same as for adding a ZPA mode module, with the exception of changing the default connection type of ZPA in *Step 4* to the connection you need for the particular Ai2 module you are connecting.

In ***Step 4***, click on the right side of the *Name* area to show a drop down box of the available connection types and select. You still need to change the data type from *SINT* to *INT* regardless of which connection type you select.	

You need to verify that the particular Ai2 modules you are connecting to be set to the proper corresponding mode. If your connection is PLC I/O mode, the Ai2 module must be placed in PLC I/O mode using EasyRoll. Similarly, for ConveyLogix Interface connection, the Ai2 module must both be in PLC I/O mode and have a ConveyLogix program installed.

Connection type mismatch (using the PLC I/O connection to an Ai2 module that happens to be in ZPA mode for example) will not indicate any specific errors but it will produce unexpected results.

USING ERSC ADD ON INSTRUCTIONS (AOI) WITH RSLOGIX 5000

Insight Automation has authored and made available Add On Instructions (AOI) in order to make your programming easier to follow. In this document up until this section, when connecting to an Ai2 module regardless of mode; your PLC program needs to directly access the register data array tags created when you created the Ai2 module's instance. The AOIs attach to created Ai2 module's register data arrays and maps the data into user tags and functions with meaningful names. There are two separate AOIs for use depending on the operational mode of the module you want to connect: a ZPA mode AOI and a PLC I/O AOI.

Please note that the use of AOI(s) is purely optional. However, you must install the EDS file and set the Data Type to INT as previously described before you can use any AOI.

SELECTING THE PROPER AOI INSTRUCTION

AOI(s) are imported to your specific PLC program file and not into the RSLogix 5000 environment like an EDS file. The following chart provides a cross-reference for selecting the proper AOI file based upon the Operational Mode:

Operational Mode	EDS File	AOI File
ZPA Mode	**ConveyLinxAi_V5_1.eds**	**Ai2_ZPA_419_5xx.L5X**
PLC I/O Mode	**ConveyLinxAi_V5_1.eds**	**Ai2_PLCIO_419_5xx.L5X**

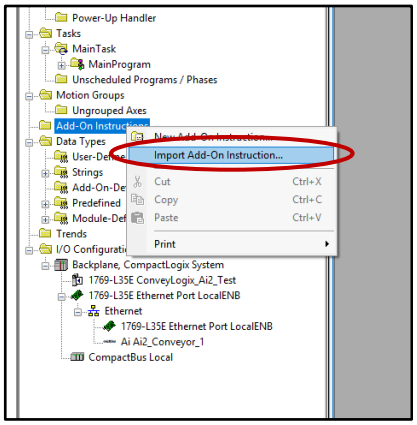
Please note that there may be updates since publication of this document. We recommend that you please go to pulseroller.com to download the latest versions of AOI files when available.

INSTALLING THE AOIS INTO RSLOGIX 5000

After your EDS files have been installed; the next procedure is to install the Add On Instruction (AOI) files that you downloaded.

Step 1

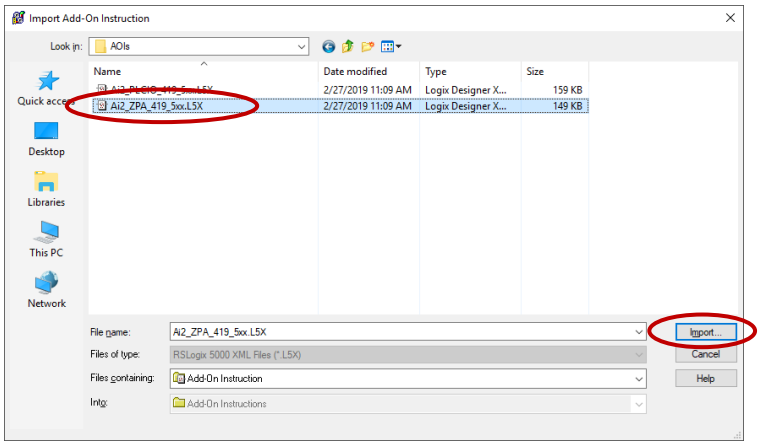
Right click on the *Add On Instruction* folder in the explorer tree. From the pop-up menu select Import *Add On Instruction.*



Step 2

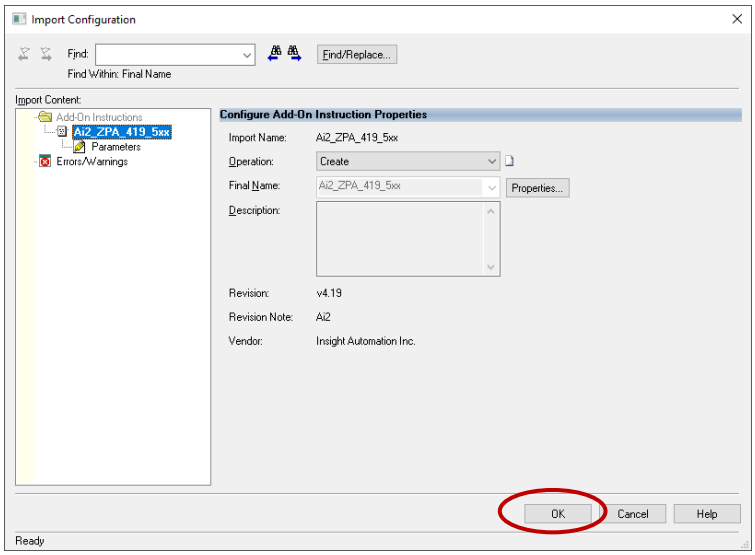
Navigate to the folder location where you downloaded your AOIs, select the file then click import. In this example we are importing the AOI for ZPA mode.

Note: The filenames shown are for example only. The filenames you download from the web site may be different and should be the latest release.



Step 3

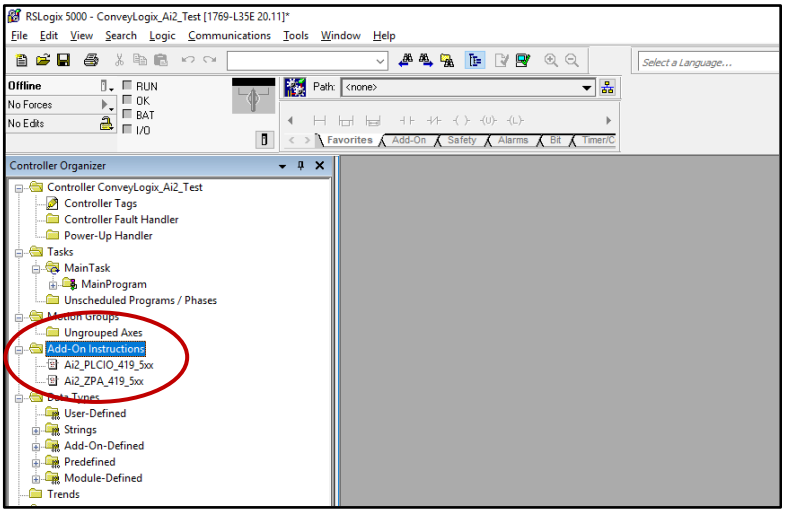
A window will appear indicating the details about the AOI you are about to import. There should be no errors or warnings. Click OK to proceed with the import.



If desired, simply repeat these 3 steps again to import the other AOI for PLC I/O mode.

When you are importing both the ZPA and PLC I/O mode AOIs, they should appear in the explorer tree as shown.

Note: You do not have to import both AOI instructions. If you only need an AOI for a particular mode, you only have to import the one you need.

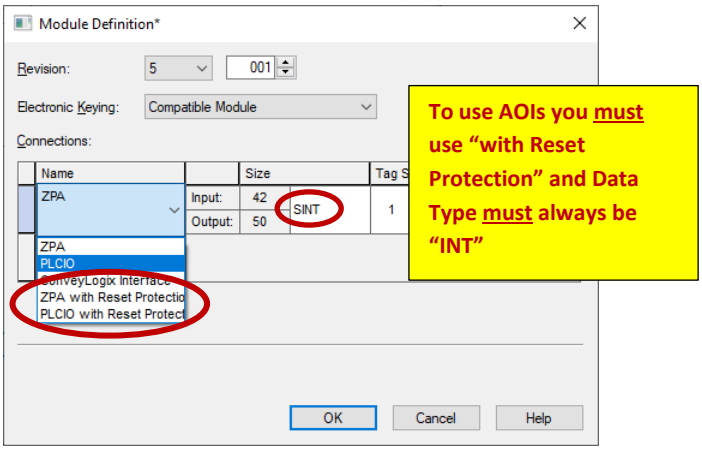


EXAMPLE FOR ASSIGNING AOI TO AI2 MODULE IN YOUR PLC PROGRAM

For our example, we are going to add one ZPA mode Ai2 module to our current program. This module was added following the steps outlined in section *Creating a ZPA Mode Ai2 Module in the Ethernet Tree* beginning on page 20. For our example we are assuming this module has been configured with I.P. address 192.168.200.22 and was

attached using the *ZPA with Reset Protection* connection as outlined in section *Creating Other Connection Types* on page 26.

Insight Automation provided AOI's require connections to be "with Reset Protection". Please refer to *Creating Other Connection Types* on page 26 for details on creating these connections. Also refer to the *ConveyLinx-Ai2 PLC Developer's Guide* for details on Reset Protection assemblies.

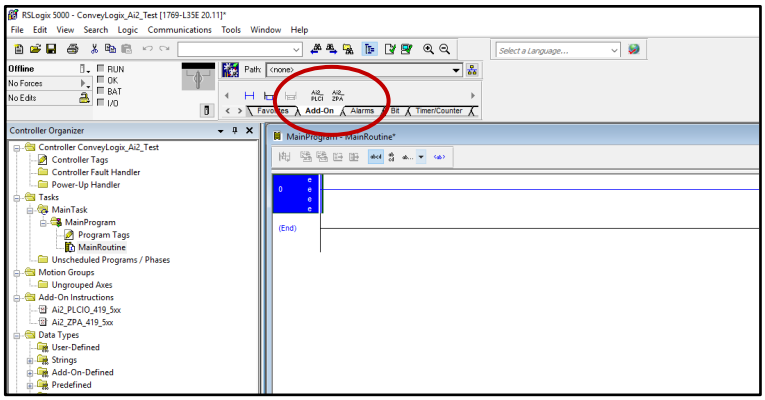


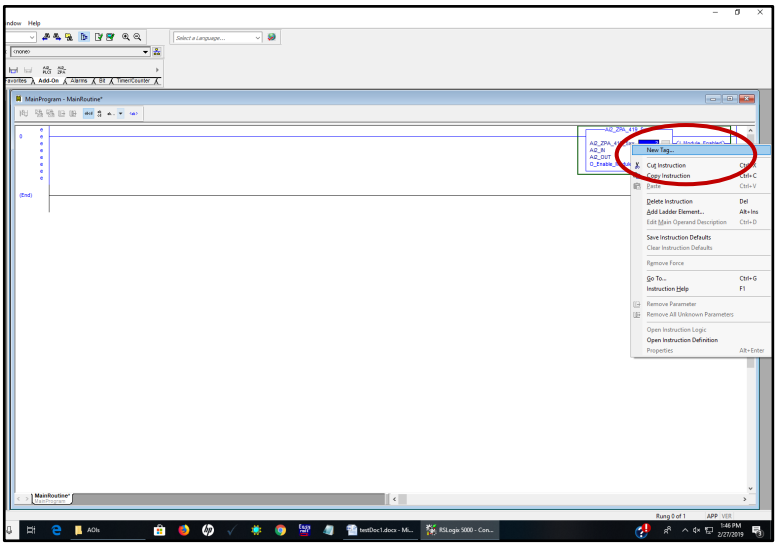
ASSIGNING NEW MODULES TO AOI

Now that we have our Ai2 module defined with their correct connection type and the AOI instruction imported into our RSLogix5000 program; the next step is to create an instance of the appropriate AOI for the Ai2 module.

Step 1

Locate the AOIs and place in your ladder diagram. For our example we are selecting the *Ai2_ZPA_419_5xx* instruction

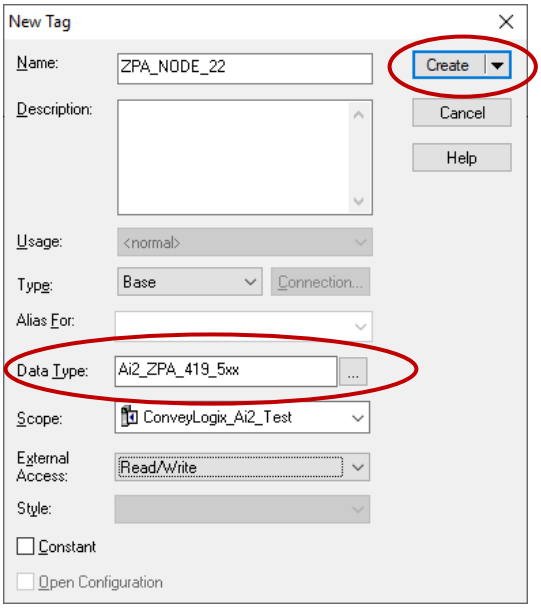




Step 2

Once the instruction has been added to the ladder, we need to create a tag that will be how you access the modules data.

For our example we will call this tag "ZPA_NODE_22"

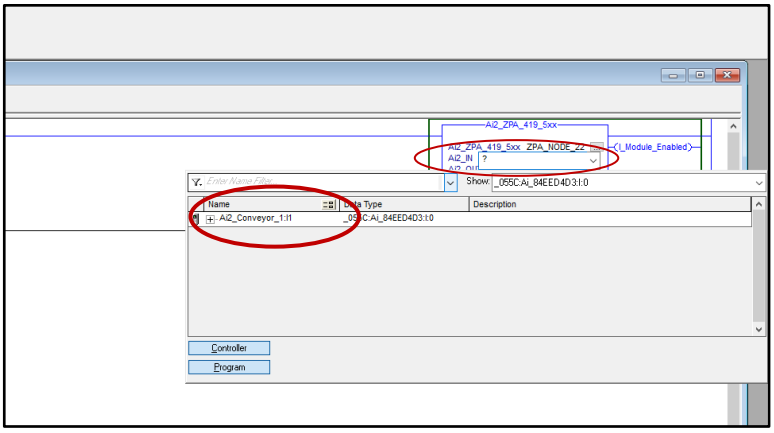


Step 3

This is the typical New Tag window you invoke from the ladder diagram screen. We entered "ZPA_NODE_22" as the name. Note that the DataType automatically defaults to the AOI's data type. Click Create to create the new tag

Step 4

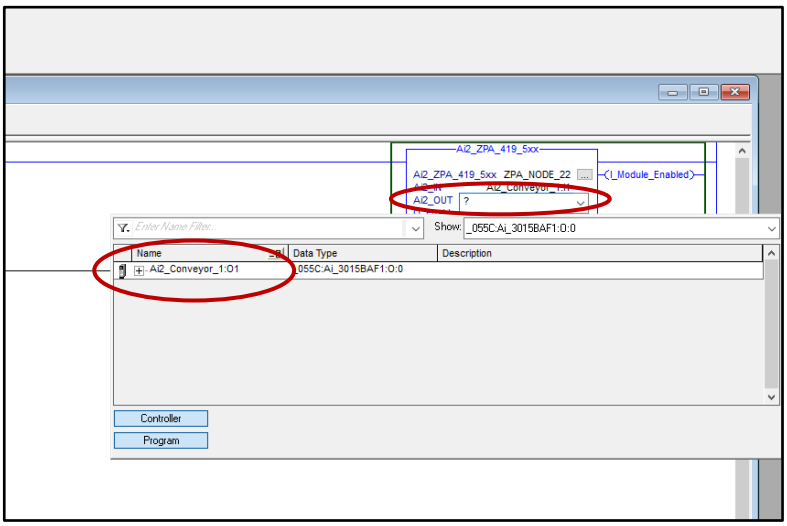
The AOI requires two other parameters; *Ai2_IN* for the data coming from the module and *Ai2_OUT* for data from the PLC to the module. These will be assigned to the I/O arrays created by the EDS file when we previously added the module. Here we will assign the *Ai2_IN* parameter by clicking the drop-down box arrow to automatically show all tags that match the data type for the *Ai2_IN* parameter. In this case, *Ai2_Conveyor_1* is the only ZPA module we created, so it is the only selection. Double click this and it will be assigned to the *Ai2_IN* parameter of our ZPA AOI.



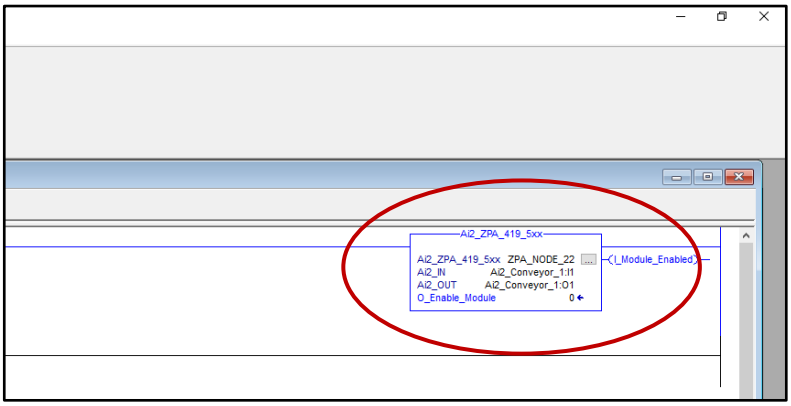
Step 5

Similar to ***Step 4***, we need to assign the *Ai2_OUT* parameter by clicking the drop-down box arrow to show all physical modules that have the matching data type for the *Ai2_OUT* parameter.

Double click this and it will be assigned to our ZPA AOI's *Ai2_OUT* parameter.



Now the ZPA AOI has been set up to use in your logic program. All of the tags associated with using the module *Ai2_Conveyor_1* that we added using the EDS file are mapped to the structured tag *ZPA_Node_22*.



You simply follow this same 5 step procedure for creating a new instance for any other modules you create using the EDS file. For Ai2 modules created in PLC I/O mode, use the *Ai2_PLCIO_419_5xx* AOI. The drop down for the data types for the *Ai2_IN* and *Ai2_OUT* parameters will automatically display only the Ai2 modules you have installed with a PLC I/O connection.

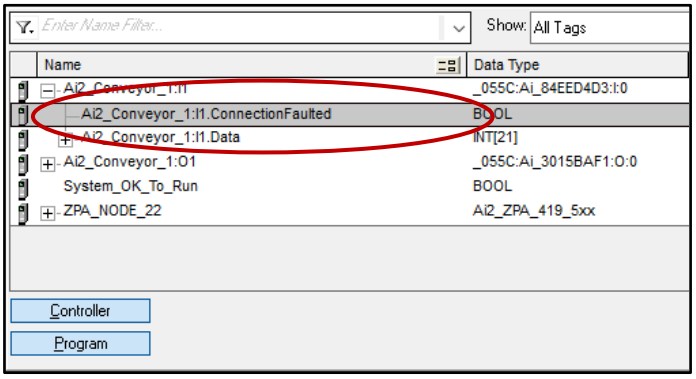
Enabling the Module for Operation

Before using the AOI in your program, you need to add some logic to enable the outputs on the physical module. Both the ZPA and PLC I/O connections defined in the EDS file use the "with reset Protection" assemblies that require the PLC to instruct the Ai2 module to process output data coming from the PLC.

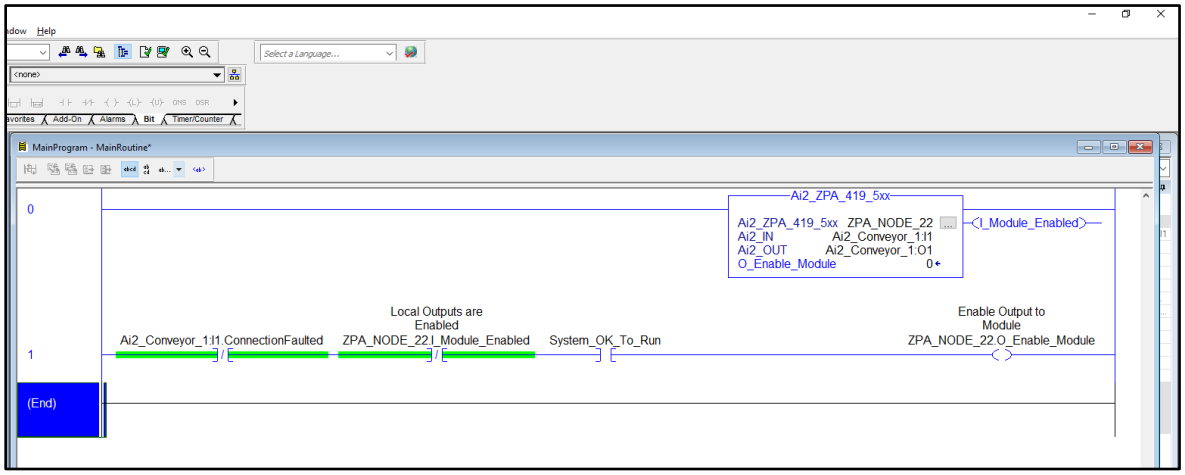
The Reset Protection topic is covered in publication ERSC-1500 ConveyLinx-Ai2 PLC Developer's Guide

Another function that is built-in when you created the module is indication of whether the PLC is communicating with the module. For example, for the ZPA module we created (Ai2_Conveyor_1); if you look in the Controller tags for the input data coming from the module, there is a Boolean value that indicates "Connection Faulted".

From our example, when you expand the *Ai2_Conveyor_1* structure, there is a BOOL that indicates *Connection Faulted*. This tag can be used in your logic to assure connection is OK prior to enabling the module.



We recommend a simple rung of logic for each module that will allow the module to be enabled when its connection is OK and when "the system is OK" to allow the Ai2 module to respond to data being written to it. This "system OK" state is wholly up to you as the programmer to determine or omit as desired.



For our example, we added a N.C. contact for the module's "ConnectionFaulted" tag, a N.C. contact for the AOI's tag that indicates that the module is not enabled, and a N.O. contact for the programmer's "System_OK_to_Run" condition as previously described. When this logic becomes true, then AOI's input to "EnableModule" is energized. When the module becomes enabled, the "ModuleEnabled" contact becomes true so that the "EnableModule" input to the AOI does not need to be held ON to keep the module enabled.

Simply repeat this rung of logic for each instance of the AOI. Also, these same tags are utilized for both the ZPA and PLC I/O AOIs, so this same type of rung is needed to enable both module modes of operation.

If you do not include some logic to set the "O_EnableModule" bit in your program, the Ai2 module will NOT respond to any data written to it by the PLC.

Please note that even if the Ai2 module is not enabled, your PLC should still receive input data from the module

ZPA AOI TAG DESCRIPTIONS

The following chart lists each tag made available in the *Ai2_ZPA_419_5xx* AOI along with the register reference from the *ConveyLinx-Ai2 PLC Developer's Guide* version 1.0.

Tag Name	Data Type	Developers Guide Register	Bit	Description
I_Arrival_DnZone	BOOL	4:0196	AOI Logic	Accumulated Arrival Downstream
I_Arrival_UpZone	BOOL	4:0116	AOI Logic	Accumulated Arrival Upstream
I_ConveyStopByLostConnection	BOOL	4:0020	6	ConveyStop Activated Because Of Lost Connection
I_ConveyStopByPLCCmd	BOOL	4:0020	7	ConveyStop Activated Because Of PLC Command
I_ConveyStopByPLCDisconnect	BOOL	4:0020	10	ConveyStop Activated Because Of Lost PLC Connection
I_ConveyStopByRemoteModule	BOOL	4:0020	5	ConveyStop Activated By Another Module in ConveyStop Group
I_Get_Tracking_FWD	DINT	4:0201 (MSW) 4:0202 (LSW)	-	Get Forward Tracking Value at Induct to Local Upstream Zone
I_Get_Tracking_REV	DINT	4:0121 (MSW) 4:0122 (LSW)	-	Get Reverse Tracking Value at Induct to Local Downstream Zone
I_Heartbeat	BOOL	4:0035	15	Module Heartbeat, 2s On / 2s Off
I_Jam_at_DnZone	BOOL	4:0089	5	Sensor Jam Local Downstream Zone
I_Jam_at_UpZone	BOOL	4:0088	5	Sensor Jam Local Upstream Zone
I_Module_Enabled	BOOL	-	-	Local Outputs are Enabled
I_Module_Fault_Left	BOOL	4:0088	3	Left Motor Error Active
I_Module_Fault_Right	BOOL	4:0089	3	Right Motor Error Active
I_Module_Sts_Left	INT	4:0088		Left Module Status
I_Module_Sts_Right	INT	4:0089		Right Module Status
I_Sensor_Port_Left_Pin2	BOOL	4:0035	0	Left Sensor Port Pin 2 Energized (if used as input)
I_Sensor_Port_Left_Pin4	BOOL	4:0035	4	Left Sensor Port Pin 4 Energized
I_Sensor_Port_Right_Pin2	BOOL	4:0035	2	Right Sensor Port Pin 2 Energized (if used as input)
I_Sensor_Port_Right_Pin4	BOOL	4:0035	6	Right Sensor Port Pin 4 Energized
I_Tracking_DnZn	DINT	4:0199 (MSW) 4:0200 (LSW)	-	For Load Accumulated in Downstream Zone
I_Tracking_UpZn	DINT	4:0119 (MSW) 4:0120 (LSW)	-	For Load Accumulated in Upstream Zone
I_Zone_Status_DnZn_FWD	SINT	4:0196 (Lo Byte)	AOI Logic	Local Downstream Status Forward Direction
I_Zone_Status_DnZn_REV	SINT	4:0196 (Hi Byte)	AOI Logic	Local Downstream Status Reverse Direction
I_Zone_Status_UpZn_FWD	SINT	4:0116 (Lo Byte)	AOI Logic	Local Upstream Status Forward Direction
I_Zone_Status_UpZn_REV	SINT	4:0116 (Hi Byte)	AOI Logic	Local Upstream Status Reverse Direction
O_Acc_Adj_UpStr_to_DnZn	BOOL	4:0184	8	Accumulate Adjacent Upstream to Local Downstream Zone
O_Acc_Adj_UpStr_to_UpZn	BOOL	4:0104	8	Accumulate Adjacent Upstream to Local Upstream Zone
O_Acc_Arr_DnZn	BOOL	4:184	0	Set Local Downstream Zone to Accumulate
O_Acc_Arr_UpZn	BOOL	4:0104	0	Set Local Upstream Zone to Accumulate
O_Clear_Jam_DnZn	BOOL	4:0189	0	Clear Jam at local Downstream Zone
O_Clear_Jam_UpZn	BOOL	4:0109	0	Clear Jam at local Upstream Zone
O_Clear_Motor_Error	BOOL	4:0022	0	Clear Motor Error Left & Right
O_ConfArrivalAdjDownstreamToDnZn	BOOL	4:0184	9	Confirm Downstream Arrival for Local Downstream Zone
O_ConfArrivalAdjDownstreamToUpZn	BOOL	4:0104	9	Confirm Downstream Arrival for Local Upstream Zone
O_ConveyMerge_Disable_Center	BOOL	4:0387	4	ConveyMerge: Disable Center Line from Releasing
O_ConveyMerge_Disable_Left	BOOL	4:0387	5	ConveyMerge: Disable Left Line from Releasing
O_ConveyMerge_Disable_Right	BOOL	4:0387	6	ConveyMerge: Disable Right Line from Releasing

Tag Name	Data Type	Developers Guide Register	Bit	Description
O_ConveyMerge_Enable_PLC_Cntrl	BOOL	4:0387	15	Enable PLC to Control ConveyMerge Priority & Release
O_ConveyMerge_Priority	SINT	4:0387 (Lo Byte)	-	Numerical Value (0-3) to Set Merge Priority
O_ConveyStop_Command	INT	4:0020	-	Set Local ConveyStop Command
O_DA_Mode_Cmd_DnZn	SINT	4:0375 (Lo Byte)	-	Direction & Accumulation Mode Command Byte for Downstream Zone
O_DA_Mode_Cmd_UpZn	SINT	4:0365 (Lo Byte)	-	Direction & Accumulation Mode Command Byte for Upstream Zone
O_DA_Mode_Value_DnZn	SINT	4:0375 (Hi Byte)	-	Direction & Accumulation Mode Number of Zones
O_DA_Mode_Value_UpZn	SINT	4:0365 (Hi Byte)	-	Direction & Accumulation Mode Number of Zones
O_Enable_Module	BOOL	-	AOI Logic	Enable Output to Module
O_Jog_FWD_DnZn	BOOL	4:0184	10	Jog Forward for Local Downstream Zone
O_Jog_FWD_UpZn	BOOL	4:0104	10	Jog Forward for Local Upstream Zone
O_Jog_REV_DnZn	BOOL	4:0184	11	Jog Reverse for Local Downstream Zone
O_Jog_REV_UpZn	BOOL	4:0104	11	Jog Reverse for Local Upstream Zone
O_Maintenance_Stop_DnZn	BOOL	4:0184	13	Force Stop Local Downstream Zone
O_Maintenance_Stop_UpZn	BOOL	4:0104	13	Force Stop Local Upstream Zone
O_Release_DnZn	BOOL	4:0185	AOI Logic	Release and Accumulate on Next at Downstream Zone
O_Release_UpZn	BOOL	4:0105	AOI Logic	Release and Accumulate on Next at Upstream Zone
O_Sensor_Prt_P2_Left_Output_Enable	BOOL	4:0063	8	Enable Sensor Port Pin 2 Left to be an Output
O_Sensor_Prt_P2_Left_Output_Energize	BOOL	4:0063	12	Energize Sensor Port Pin 2 Left, Output Enable must be Energized
O_Sensor_Prt_P2_Right_Output_Enable	BOOL	4:0063	9	Enable Sensor Port Pin 2 Right to be an Output
O_Sensor_Prt_P2_Right_Output_Energize	BOOL	4:0063	13	Energize Sensor Port Pin 2 Right, Output Enable must be Energized
O_Speed_Motor_Left	INT	4:0040	-	Value in mm/sec for MDR or RPM x 10 for PGD
O_Speed_Motor_Right	INT	4:0064	-	Value in mm/sec for MDR or RPM x 10 for PGD
O_Sts_DnZn_Discharge_FWD	SINT	4:0232 (Lo Byte)	-	Set Downstream Discharge Zone Forward Status Value
O_Sts_DnZn_Discharge_REV	SINT	4:0232 (Hi Byte)	-	Set Downstream Discharge Zone Reverse Status Value
O_Sts_UpZn_Induct_FWD	SINT	4:0134 (Lo Byte)	-	Set Upstream Induct Zone Forward Status Value
O_Sts_UpZn_Induct_REV	SINT	4:0134 (Hi Byte)	-	Set Upstream Induct Zone Reverse Status Value
O_Tracking_DnZn	DINT	4:0212 (MSW) 4:0213 (LSW)	-	Write 32 Bit Tracking Data for Local DnZn When Accumulated.
O_Tracking_Induct_FWD	DINT	4:0139 (MSW) 4:0140 (LSW)	-	Set Forward Induct Tracking Value
O_Tracking_Induct_REV	DINT	4:0237 (MSW) 4:0238 (LSW)	-	Set Reverse Induct Tracking Value
O_Tracking_UpZn	DINT	4:0132 (MSW) 4:0133 (LSW)	-	Write 32 Bit Tracking Data for Local UpZn When Accumulated.
O_WakeUp_DnZn	BOOL	4:0184	12	Wake Up Local Downstream Zone
O_WakeUp_UpZn	BOOL	4:0104	12	Wake Up Local Upstream Zone

PLC I/O AOI Tag Descriptions

The following chart lists each tag made available in the *Ai2_PLCIO_419_5xx* AOI along with the register reference from the *ConveyLinx-Ai2 PLC Developer's Guide* version 1.0.

Tag Name	Data Type	Developers Guide Register	Bit	Description
I_ConveyStopByLostConnection	BOOL	4:0020	6	Stop active due to lost communication connection
I_ConveyStopByPLCCmd	BOOL	4:0020	7	Stop active due to Stop Command from PLC
I_ConveyStopByPLCDisconnect	BOOL	4:0020	10	Stop active due to lost PLC connection
I_ConveyStopByRemoteModule	BOOL	4:0020	5	Stop active on another module in Stop Group
I_Current_Left_MDR	REAL	4:0055	AOI Logic	Current Draw of Left MDR in Amps
I_Current_Right_MDR	REAL	4:0079	AOI Logic	Current Draw of Right MDR in Amps
I_Dig_Mtr_Over_Cur_Left	BOOL	4:0060	14	Left: More than 1A detected on one or more outputs
I_Dig_Mtr_Over_Cur_Right	BOOL	4:0084	14	Right: More than 1A detected on one or more outputs
I_Dig_Mtr_Short_Cir_Left	BOOL	4:0060	12	Left: Short Circuit Error on one or more outputs
I_Dig_Mtr_Short_Cir_Right	BOOL	4:0084	12	Right: Short Circuit Error on one or more outputs
I_Frequency_Left_MDR	INT	4:0056	-	Frequency of Left MDR in Hz
I_Frequency_Right_MDR	INT	4:0080	-	Frequency of Right MDR in Hz
I_Heartbeat	BOOL	4:0035	15	Module Heartbeat, 2s On / 2s Off
I_Module_Enabled	BOOL	-	-	Local Outputs are Enabled
I_Motor_Sts_Left	INT	4:0058		Left Module Status
I_Motor_Sts_Right	INT	4:0082		Right Module Status
I_Sensor_Detect_Left_Port	BOOL	4:0036	1	Sensor Detected on Left Pin 4
I_Sensor_Detect_Right_Port	BOOL	4:0036	0	Sensor Detected on Right Pin 4
I_Sensor_Port_Left_Pin2	BOOL	4:0035	0	Left Sensor Port Pin 2 Energized if used as input
I_Sensor_Port_Left_Pin4	BOOL	4:0035	4	Left Sensor Port Pin 4 Energized
I_Sensor_Port_Right_Pin2	BOOL	4:0035	2	Right Sensor Port Pin 2 Energized if used as input
I_Sensor_Port_Right_Pin4	BOOL	4:0035	6	Right Sensor Port Pin 4 Energized
I_Servo_CmdComplete_Left	BOOL	4:0011	0	1 = Last Servo Run Command Complete
I_Servo_CmdComplete_Right	BOOL	4:0016	0	1 = Last Servo Run Command Complete
I_Servo_CmdStatus_Left	BOOL	4:0011	2	1 = Running to Pulse Command Position
I_Servo_CmdStatus_Right	BOOL	4:0016	2	1 = Running to Pulse Command Position
I_Servo_Position_Left	INT	4:0062	-	Signed Integer Value for Position from "0"
I_Servo_Position_Right	INT	4:0086	-	Signed Integer Value for Position from "0"
I_Speed_Left	INT	4:0507	-	Value in mm/sec for MDR or RPM x 10 for PGD
I_Speed_Left_Max	BOOL	4:0507	14	Left Motor Set Speed is Greater Than Motor Maximum
I_Speed_Left_Min	BOOL	4:0507	15	Left Motor Set Speed is Less Than Motor Minimum
I_Speed_Right	INT	4:0508	-	Value in mm/sec for MDR or RPM x 10 for PGD
I_Speed_Right_Max	BOOL	4:0508	14	Right Motor Set Speed is Greater Than Motor Maximum
I_Speed_Right_Min	BOOL	4:0508	15	Right Motor Set Speed is Less Than Motor Minimum
I_Temp_MDR_Left	INT	4:0057 (Hi Byte)	-	Left Motor Temperature Reading in °C
I_Temp_MDR_Right	INT	4:0081 (Hi Byte)	-	Right Motor Temperature Reading °C
I_Temp_On_Board_Left	INT	4:0057 (Lo Byte)	-	Left On Board Temperature Reading °C
I_Temp_On_Board_Right	INT	4:0081 (Lo Byte)	-	Right On Board Temperature Reading °C
I_Tracking_UpZn	DINT	4:0139 (MSW) 4:0140 (LSW)	-	Tracking Hi Word for Load Discharged from Adjacent Upstream Module

Tag Name	Data Type	Developers Guide Register	Bit	Description
I_Voltage_MDR	REAL	4:0024	-	Voltage Measured on MDR Power Connection
I_Zone_Status_DnModule	INT	4:0232 (Lo Byte)	-	Local Downstream Status
I_Zone_Status_UpModule	INT	4:0134 (Lo Byte)	-	Local Upstream Status
O_Clear_Motor_Error	BOOL	4:0022	0	Clear Motor Error Left & Right
O_ConveyStop_Command	INT	4:0020	-	Set Local ConveyStop Command
O_Digital_Mode_Enable_Left	BOOL	4:0060	15	Enable Digital Output Mode for Left Motor Port
O_Digital_Mode_Enable_Right	BOOL	4:0084	15	Enable Digital Output Mode for Right Motor Port
O_Digital_Out_Left_P3_Energize	BOOL	4:0060	2	Energize Digital Output Pin 3 Left Motor Port
O_Digital_Out_Left_P4_Energize	BOOL	4:0060	1	Energize Digital Output Pin 4 Left Motor Port
O_Digital_Out_Right_P2_Energize	BOOL	4:0084	0	Energize Digital Output Pin 2 Right Motor Port
O_Digital_Out_Right_P3_Energize	BOOL	4:0084	2	Energize Digital Output Pin 3 Right Motor Port
O_Enable_Module	BOOL	-	AOI Logic	Enable Output to Module
O_Mtr_Accel_Ramp_Left	INT	4:0043	-	Value in mm for MDR or Pulses for PGD
O_Mtr_Accel_Ramp_Right	INT	4:0067	-	Value in mm for MDR or Pulses for PGD
O_Mtr_Brake_Method_Left	INT	4:0261	-	Left Motor Control
O_Mtr_Brake_Method_Right	INT	4:0271	-	Right Motor Control
O_Mtr_Decel_Ramp_Left	INT	4:0044	-	Value in mm for MDR or Pulses for PGD
O_Mtr_Decel_Ramp_Right	INT	4:0068	-	Value in mm for MDR or Pulses for PGD
O_Mtr_Dir_Left	BOOL	4:0260	0 & 8	On = Run in Opposite of Config. Dir.
O_Mtr_Dir_Right	BOOL	4:0270	0 & 8	On = Run in Opposite of Config. Dir.
O_Mtr_Run_Left	BOOL	4:0260	0	On = Run
O_Mtr_Run_Right	BOOL	4:0270	0	On = Run
O_Mtr_Slave_Mode_Left	INT	4:0262	-	0 = Ignore 1 = OFF: Left motor independently controlled 2 = ON: Left motor mirrors Right motor control
O_Mtr_Slave_Mode_Right	INT	4:0272	-	0 = Ignore 1 = OFF: Right motor independently controlled 2 = ON: Right motor mirrors Left motor control
O_Mtr_Speed_Left	INT	4:0040	-	Value in mm/sec for MDR or RPM x 10 for PGD
O_Mtr_Speed_Right	INT	4:0064	-	Value in mm/sec for MDR or RPM x 10 for PGD
O_Sensor_Prt_Mask_P2_Left	BOOL	4:0034	0	Invert the Active State for the Pin
O_Sensor_Prt_Mask_P2_Right	BOOL	4:0034	2	Invert the Active State for the Pin
O_Sensor_Prt_Mask_P4_Left	BOOL	4:0034	4	Invert the Active State for the Pin
O_Sensor_Prt_Mask_P4_Right	BOOL	4:0034	6	Invert the Active State for the Pin
O_Sensor_Prt_P2_Left_Output_Enable	BOOL	4:0037	5	Enable Sensor Port Pin 2 Left to be an Output
O_Sensor_Prt_P2_Left_Output_Energize	BOOL	4:0037	0	Energize Sensor Port Pin 2 Left, Output Enable must be Energized
O_Sensor_Prt_P2_Right_Output_Enable	BOOL	4:0037	6	Enable Sensor Port Pin 2 Right to be an Output
O_Sensor_Prt_P2_Right_Output_Energize	BOOL	4:0037	1	Energize Sensor Port Pin 2 Right, Output Enable must be Energized
O_Servo_CmdPosition_Left	INT	4:0008		Value in mm for MDR or Pulses for PGD
O_Servo_CmdPosition_Right	INT	4:0013		Value in mm for MDR or Pulses for PGD
O_Servo_GoCmd_Left	BOOL	4:0009	1	Go to Commanded Position from Position "0"
O_Servo_GoCmd_Right	BOOL	4:0014	1	Go to Commanded Position from Position "0"
O_Servo_Zero_Left	BOOL	4:0009	0	Set Current Pulse Count as “0”
O_Servo_Zero_Right	BOOL	4:0014	0	Set Current Pulse Count as “0”
O_Tracking_Discharge	DINT	4:0201 (MSW) 4:0202 (LSW)	-	Tracking 32 Bit for Load Being Discharged
O_Zone_Sts_DnZn	INT	4:0196 (Lo Byte)	-	Downstream Zone Status
O_Zone_Sts_UpZn	INT	4:0116 (Lo Byte)	-	Upstream Zone Status

USING LOGIX5000 MSG INSTRUCTION

Access to ConveyLinx Ai2 modules is also available utilizing the Logix 5000 *MSG* instruction. The *MSG* instruction utilizes CIP Explicit Messaging. This means that the connection is not maintained as an implicit connection. Generic Ethernet Module and EDS connections are implicit and thus must be maintained at all times or there will be a communication fault. Explicit Messaging opens the connection, reads/writes data, and then closes the connection thus freeing up communications resources for the PLC.

WHEN TO USE MSG INSTRUCTIONS

Because the MSG instruction is executed asynchronous to program scan and is not subject to implicit messaging RPI restrictions; the response time between requesting data and receiving data is not deterministic and can vary between separate requests for the same data from the same device. Therefore, **we recommend that MSG instructions should not be used for dedicated "real time" control of equipment**. For ConveyLinx Ai2 modules, MSG instructions are intended to gather "low priority" status information and/or to send infrequent parameter changes. Please note that this is only a recommendation. Your particular application's specifics, PLC's capacity, available network bandwidth, etc. may allow you to get expected results with "real time" control utilizing MSG instructions to interface with ERSC modules.

REFRESHER ON ASSEMBLIES

The topic of Assemblies is covered in detail in publication ERSC-1506 ConveyL:inx Ai2 PLC Developer's Guide

The parameter that is entered in a MSG instruction's configuration to define the location on the remote device to read/write data corresponds to a Modbus address in the ConveyLinx Ai2 module. The Ai2 has 512 "***Module***" Modbus data registers and these can be thought of as "physical" module address locations. An Assembly is a grouping of some subset of these 512 ***Module*** registers based upon the relevance of the data. For example, the ZPA Input Assembly groups together 21 ***Module*** registers out of the 512 that are relevant for ZPA Inputs. This relevant data from within the ***Module*** 512 registers are not necessarily in consecutive address locations and are scattered throughout the 512 addresses. The Assembly groups them together so they can be read efficiently all at once.

In publication *ERSC-1500 ConveyLinx Ai2 PLC Developer's Guide*, each Assembly chart cross references the ***Module*** Modbus address with its corresponding "***Assembled***" Modbus address. For example, the ZPA mode inputs are shown to use ***Assembled*** Modbus addresses 4:1500 thru 4:1520. The ***Assembled*** Modbus addresses can be thought of as "virtual". ***Assembled*** addresses cannot be accessed individually; they can only be accessed from the initial boundary address (i.e. 4:1500 in the ZPA Input Assembly example).

Module Vs. Assembled Address with MSG Instructions

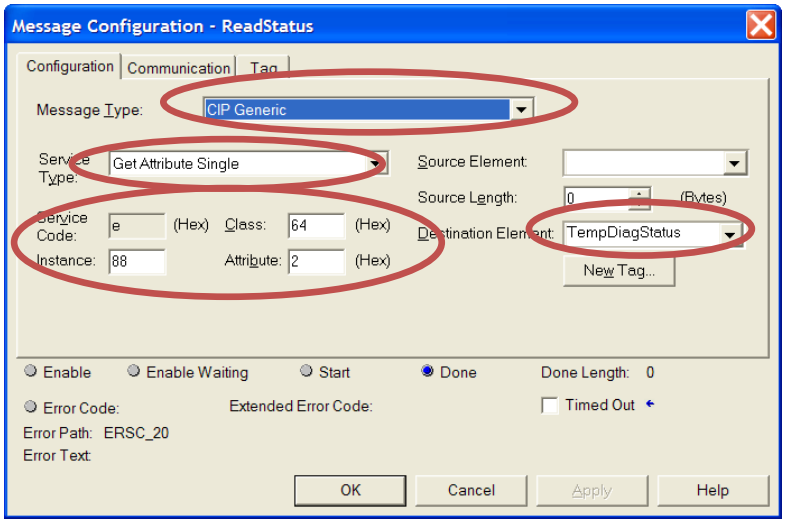
What this means is that there are certain restrictions on what you can do with MSG instructions with an Ai2 module. Here is a list:

- You can us a single MSG instruction to read one and up to 30 consecutive ***Module*** Modbus registers
- You can use a single MSG instruction to write to one (an only one) of the ***Module*** Modbus registers
- You can use a single MSG instruction to read any of the available ***Assembled Input*** registers in their entirety
- You ***CANNOT*** use any MSG instruction to **write** to any ***Assembled Output*** registers.

Message Configuration for Reading from Ai2 Module Registers

Read MSG Setup

- Select "CIP Generic" as the *Message Type*
- Select "Get Attribute Single" as the *Service Type*
- *Class* is always set to 64
- *Instance* is the ***Module*** Modbus register address. In this example the Instance is 88 indicating register 4:0088
- *Attribute* is the number of registers to read. In this example it is set = 2. This means the MSG instruction will read ***Module*** Modbus registers 4:0088 and 4:0089
- *Destination Element* is the user defined tag for the MSG instruction to place the data it reads from the Ai2. In this example, "TempDiagStatus" is the user defined tag.



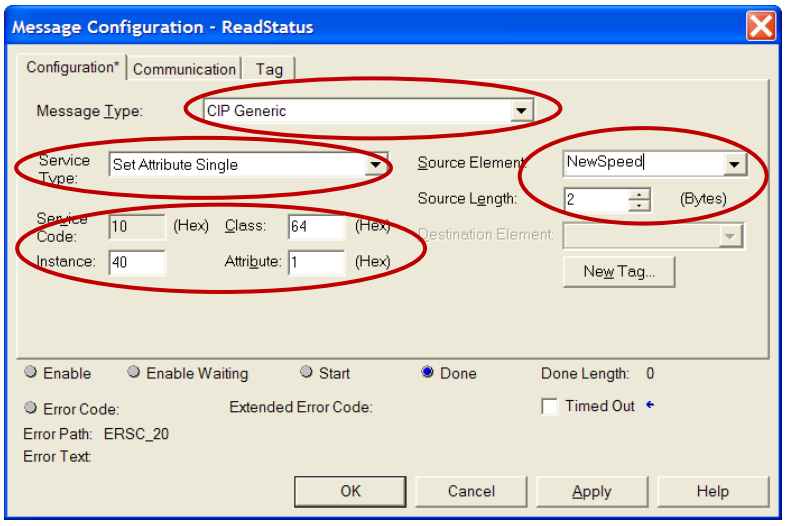
The acceptable values for *Attribute* are from 0x1 to 0x1E which is 1 to 30 contiguous registers. In the above example, the data being read is Module Status #1 and Module Status #2 registers (4:0088 and 4:0089). This same MSG instruction could be duplicated for each *Ai2* in ZPA mode in a given conveyor system and used to populate an array of module status data that could in turn be used for example to feed an HMI diagnostic application.

MESSAGE CONFIGURATION FOR WRITING DATA TO AI2 MODULE REGISTER

Write MSG Setup

- Select "CIP Generic" as the *Message Type*
- Select "Set Attribute Single" as the *Service Type*
- *Class* is always set to 64
- *Instance* is the Modbus register address. In this example the Instance is 40 indicating register 4:0040
- *Attribute* is the number of registers to write. This value is always set to 1
- *Source Element* is the PLC tag that contains the data to be written to the defined Modbus register.

Source Length is always set to 2



The above example illustrates how to set-up a MSG instruction to write a new speed reference to a specific Ai2 module's Left motor (Module register 4:0040). The tag "NewSpeed" contains the value of speed reference that the PLC wants to write.

Please note that the data type of each Modbus register is integer (INT). The user defined controller tag used for "Source Element" must of appropriate data type to accept the MSG instruction data. Please consult Allen-Bradley documentation for full description of MSG instruction usage.

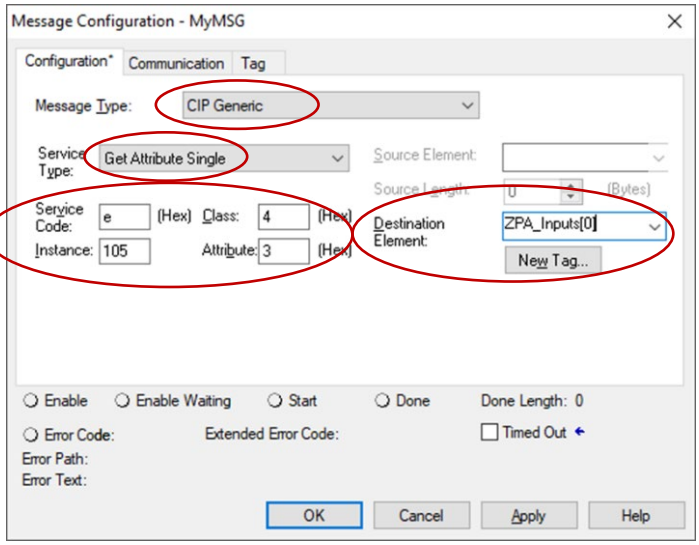
Although a read MSG instruction can be used on a module in PLC I/O mode, it is assumed that any Ai2 module in PLC I/O must already be utilizing a permanent TCP connection and should not ever need to be accessed with a read MSG instruction.

Refer to Allen-Bradley reference documentation for the particular PLC processor being used as to the proper usage and expected performance loading on the processor communication channels due to multiple MSG instructions executing simultaneously.

MESSAGE CONFIGURATION FOR READING AI2 ASSEMBLED REGISTERS

Read Assembly MSG Setup

- Select "CIP Generic" as the Message Type
- Select "Get Attribute Single" as the Service Type
- *Class* is always set to 4
- *Instance* is the Assembly number. In this example the Instance is 105 corresponding to ZPA Inputs Assembly
- *Attribute* is always set to 3
- *Destination Element* is the user defined tag for the MSG instruction to place the data it reads from the Ai2 module. In this example, "ZPA_Inputs" is the user-defined tag that is an array of INT that is equal to the number of registers provided by the Assembly



Please note that the data type of your *Destination Element* tag that you create must be an INT array equal to the size of the Assembly. Please refer to section *Parameters for Each Assembly* beginning on page 16 to determine the register array size required for each Assembly. Please consult Allen-Bradley documentation for full description of MSG instruction usage

Refer to Allen-Bradley reference documentation for the particular PLC processor being used as to the proper usage and expected performance loading on the processor communication channels due to multiple MSG instructions executing simultaneously.

NOTES:

PULSE ROLLER

WWW.PULSEROLLER.COM
SALES@PULSEROLLER.COM
SUPPORT@PULSEROLLER.COM

Publication ERSC-1521 Rev 1.0 – February 2019