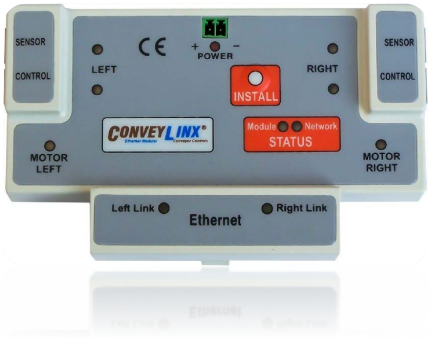
PULSEROLLER

CONVEYLINX

Connecting ConveyLinx-ERSC to Rockwell PLCs

Version 3.0

February 2019



For ConveyLinx ERSC Firmware versions 4.25 and 5.02

ConveyLinx module firmware and functionality is protected by U.S. and international patents. For complete patent information visit www.pulseroller.com/patents

Publication *ERSC*-1520

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 ERSC 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, PULSEROLLER does not assume responsibility or liability (to include intellectual property liability) for actual use based on the examples shown in this publication

Reproduction of the contents of this manual, in whole or in part, without written permission of PULSEROLLER is prohibited.

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	September 2014	Initial Release
2.0	January 2016	Major revision for firmware updates 4.25 and 5.02
3.0	February 2019	Added assembly read MSG instruction

Contact Information

PULSE ROLLER

www.pulseroller.com

North & South America

sales@pulseroller.com

support@pulseroller.com

Global

Global_sales@pulseroller.com

Global_support@pulseroller.com

TABLE OF CONTENTS

Symbol Conventions 3
Important User Information 3
Summary of Changes 4
Contact Information 4
Table of Contents 5
Preface 6
  Who Should Use This Manual? 6
  Prerequisites 6
  Not Included in This Manual 6
Introduction 7
  Ethernet I/P Guidelines 8
  Selecting Your Connection Method based upon Assembly 8
Using Generic Ethenret Module Method 11
  ODVA Compliant Firmware 5.02 for ERSC 11
  General Procedure for Connecting using Generic Ethernet Module 12
  Example for ERSC in ZPA Mode 12
  Parameters for Each Assembly 16
    For Firmware 4.24 and Earlier 16
    For Firmware 4.25 16
    For Firmware 5.02 17
Using EDS File Method 18
  Selecting the Proper EDS File 18
  Installing ConveyLinx EDS File into RSLogix5000 19
  Creating a ZPA Mode ERSC Module in the Ethernet Tree 21
    Note ① - Data Type Size 25
    Note ② - RPI Settings 26
  Creating Other Connection Types 27
Using ERSC Add On Instructions (AOI) with RSLogix 5000 29
  Selecting the Proper AOI Instruction 29
  Installing the AOIs into RSLogix 5000 30
  Example for Assigning AOI to ERSC Modules in Your Project 31
  Assigning New Modules to AOI 32
  Enabling the Module for Operation 35
  ZPA AOI Tag Descriptions 37
  PLC I/O AOI Tag Descriptions 39
Using Logix5000 MSG Instruction 43
  When to Use MSG Instructions 43
  Refresher on Assemblies 43
    Module Vs. Assembled Address with MSG Instructions 44
  Message Configuration for Reading from ERSC Module Registers 44
  Message Configuration for Writing Data To ERSC Module Register 45
  Message Configuration for Reading ERSC Assembled Registers 46
Appendix A – EDS Module Data Type Cross Reference 47
Notes: 49

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 either the *ConveyLinx PLC Developer's Guide* (Pulseroller 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 ERSC only. For information on how to connect ConveyLinx-Ai2 modules please see publication *ERSC-1521 Connecting ConveyLinx-Ai2 to Rockwell PLCs*

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-ERSC and NOT for ConveyLinx-Ai2. For information on how to work with ConveyLinx-Ai2, see publication ERSC-1521 Conecting ConveyLinx-Ai2 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* (publication ENET-AP001D-EN-P).

Selecting Your Connection Method based upon Assembly

As described in our *PLC Developer's Guide* (publication ERSC 1500), 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 the *PLC Developer's Guide.*

ODVA Compliant Firmware 5.02 for ERSC

Pulseroller has been granted a Certificate of Conformity from ODVA for ConveyLinx ERSC firmware version 5.02. The main difference between firmware 5.02 and previous versions (4.xx, 3.xx) as it pertains to connecting to ODVA compliant Ethernet I/P (EIP) PLC devices is that these previous versions utilized Instance Identifiers that were classified as "reserved" by the ODVA specification.

All Firmware 5.xx versions have re-assigned these identifiers into the allowable range for ODVA compliance. The actual data registers and functionality of all EIP assemblies has remained unchanged from the published assemblies in our *PLC Developer's Guide*. The only thing that has changed in version 5.xx is the value used for the Instance Identifiers when connecting to the PLC. Firmware 4.25 recognizes both the previous and ODVA values for backward compatibility if you happen to upgrade ERSC firmware from 4.24 (or earlier) to 4.25. 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	Pre 4.25 Recognized Instance Values	4.25 Recognized Instance Values	5.02 Recognized Instance Values
ZPA Mode Assembly Inputs	5	5 & 105	105
ZPA Mode Assembly Outputs	6	6 & 106	106
ZPA Mode Assembly Inputs with Reset Protection	25	25 & 305	305
ZPA Mode Assembly Outputs with Reset Protection	26	26 & 306	306
Reduced Size ZPA Mode Assembly Inputs	19	19 & 119	119
Reduced Size ZPA Mode Assembly Outputs	20	20 & 120	120
Reduced Size ZPA Mode Assembly Inputs with Reset Protection	39	39 & 319	319
Reduced Size ZPA Mode Assembly Outputs with Reset Protection	40	40 & 320	320
PLC I/O Mode Assembly Inputs	7	7 & 107	107
PLC I/O Mode Assembly Outputs	8	8 & 108	108
PLC I/O Mode Assembly Inputs with Reset Protection	27	27 & 307	307
PLC I/O Mode Assembly Outputs with Reset Protection	28	28 & 308	308
Reduced Size PLC I/O Mode Assembly Inputs	17	17 & 117	117
Reduced Size PLC I/O Mode Assembly Outputs	18	18 & 118	118
Reduced Size PLC I/O Mode Assembly Inputs with Reset Protection	37	37 & 317	317
Reduced Size PLC I/O Mode Assembly Outputs with Reset Protection	38	38 & 318	318
ConveyLogix Assembly Inputs	Not Available	121	121
ConveyLogix Assembly Outputs	Not Available	122	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 ERSC 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.

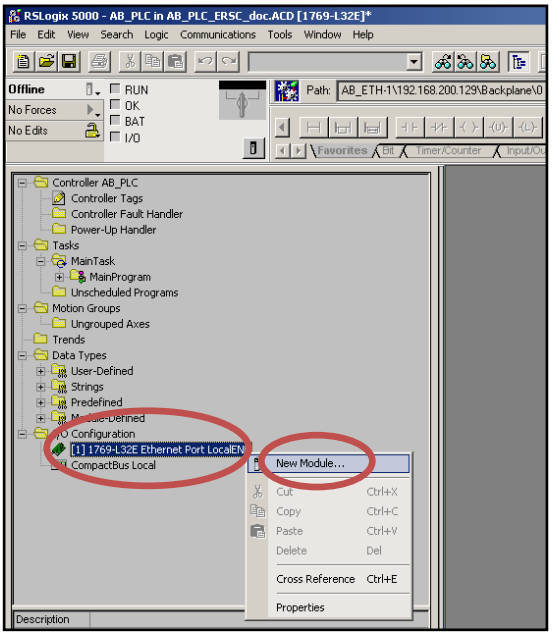
Example for ERSC in ZPA Mode

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

Step #1

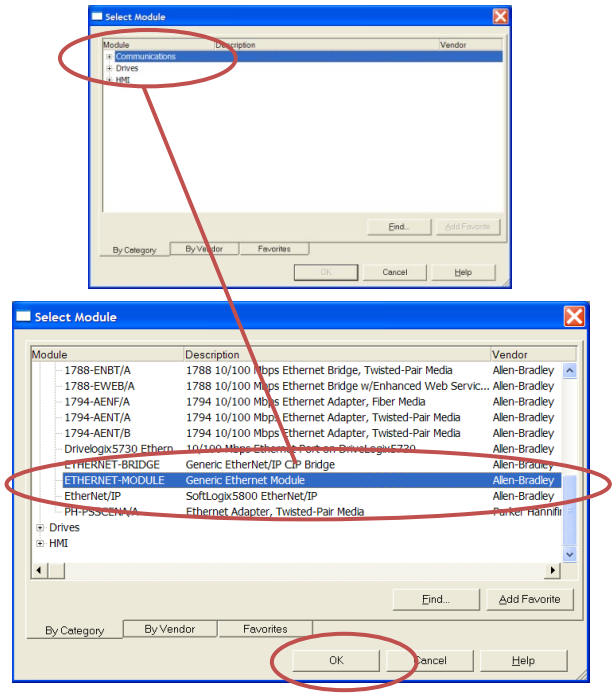
Add a New Module to the processor's I/O configuration by highlighting the processor's local Ethernet port in the I/O configuration tree.

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



Step #2

From the Select Module pop-up window, expand the Communications tree and select "Generic Ethernet Module" and click OK, which will open up the New Module window

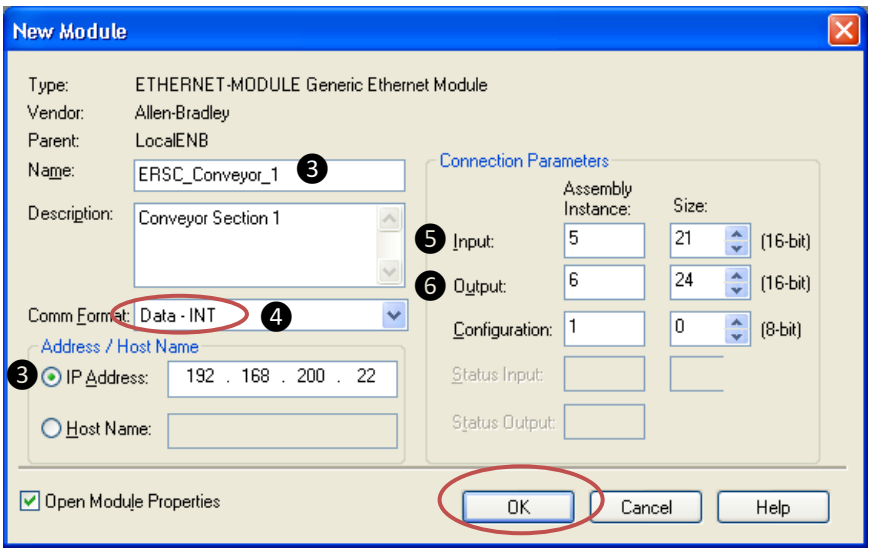


Step #3 thru #6

Fill in the Name field. This will be the *ModuleName* that will appear in your program Tag Database for any addressing.

Select Comm Format to be "Data – INT" and fill in the I.P. address of the *ERSC*.

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



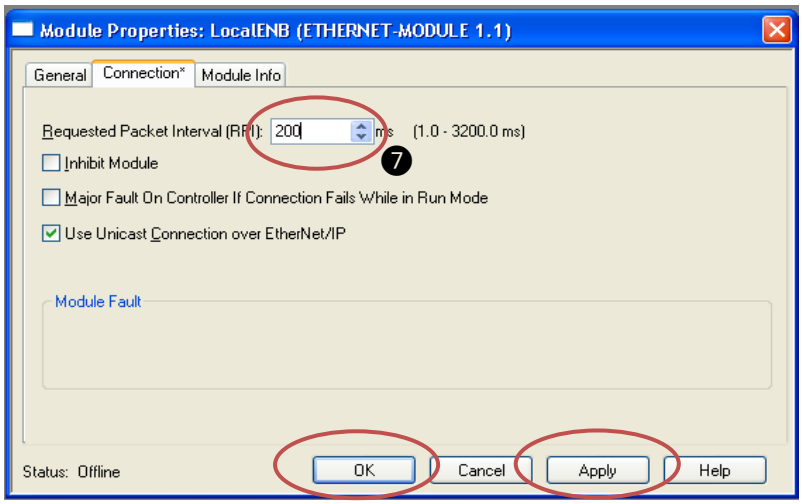
It is <u>very important</u> to select *Comm Format* data type to be INT or interface to *ERSC* will not operate correctly!

Please note that in steps 5 and 6 in the example, Input and Output Assembly Instance values shown (5 and 6 respectively) are valid for FW 4.25. For FW 5.02, these are invalid and the correct values would be 105 and 106 respectively.

Step #7

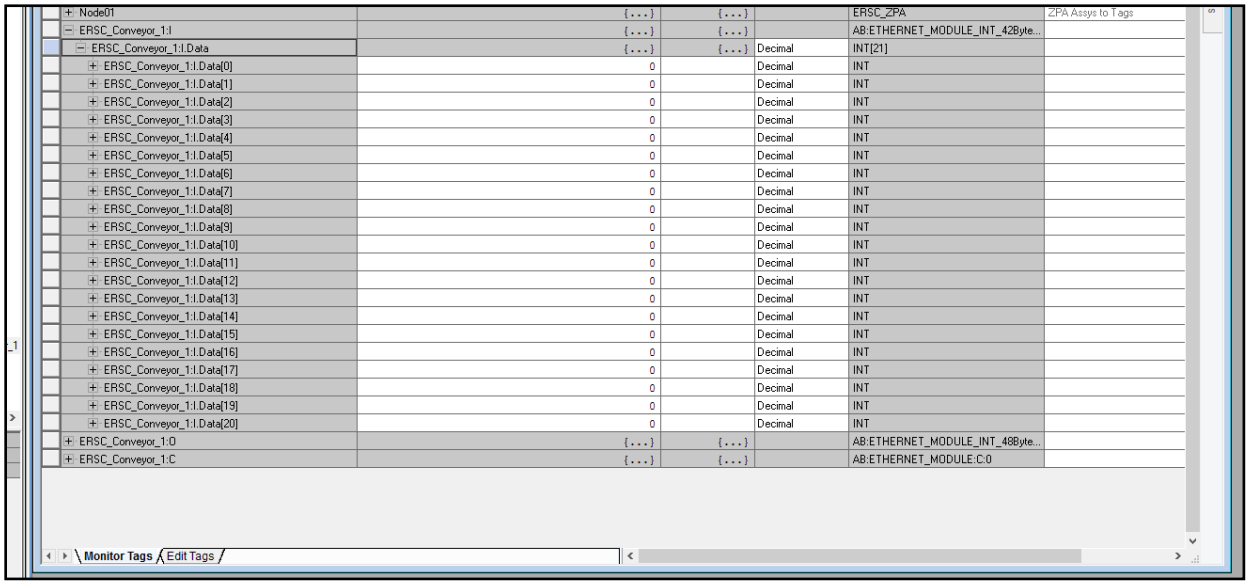
Set RPI to a value no lower than 10ms. 200 ms is typical for ZPA Interface. You may also optionally select Unicast Connection.

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



Once you have completed the configuration of your ERSC, 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 PLC Developer's Guide.

For our example, we created an ERSC named "ERSC_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.



Node01	{...}	{...}		ERSC_ZPA	ZPA Assys to Tags
ERSC_Conveyor_1:I	{...}	{...}		AB:ETHERNET_MODULE_INT_42Byte...	
ERSC_Conveyor_1:I.Data	{...}	{...}	Decimal	INT[21]	
ERSC_Conveyor_1:I.Data[0]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[1]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[2]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[3]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[4]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[5]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[6]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[7]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[8]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[9]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[10]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[11]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[12]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[13]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[14]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[15]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[16]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[17]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[18]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[19]	0		Decimal	INT	
ERSC_Conveyor_1:I.Data[20]	0		Decimal	INT	
ERSC_Conveyor_1:O	{...}	{...}		AB:ETHERNET_MODULE_INT_48Byte...	
ERSC_Conveyor_1:C	{...}	{...}		AB:ETHERNET_MODULE:C:0	

Monitor Tags / Edit Tags

FIGURE 1 - GENERIC MODULE ERSC INPUT DATA ARRAY

Scope: SampleAOI Show: All Tags Enter Name Filter...

Name	Value	Force Mask	Style	Data Type	Description
Arrival_LE	0		Decimal	BOOL	
System_OK_To_Run	1		Decimal	BOOL	
+ Tracking_Data	-2147352831		Decimal	DINT	
+ Node01	{...}	{...}		ERSC_ZPA	ZPA Assys to Tags
+ ERSC_Conveyor_1:I	{...}	{...}		AB:ETHERNET_MODULE_INT_42Byte...	
− ERSC_Conveyor_1:O	{...}	{...}		AB:ETHERNET_MODULE_INT_48Byte...	
− ERSC_Conveyor_1:O.Data	{...}	{...}	Decimal	INT[24]	
+ ERSC_Conveyor_1:O.Data[0]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[1]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[2]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[3]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[4]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[5]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[6]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[7]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[8]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[9]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[10]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[11]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[12]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[13]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[14]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[15]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[16]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[17]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[18]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[19]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[20]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[21]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[22]	0		Decimal	INT	
+ ERSC_Conveyor_1:O.Data[23]	0		Decimal	INT	
+ ERSC_Conveyor_1:C	{...}	{...}		AB:ETHERNET_MODULE:C:0	

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".

For Firmware 4.24 and Earlier

Assembly	Type	Instance Value	Size Value
ZPA Mode Assembly	Input	5	21
	Output	6	24
ZPA Mode Assembly with Reset Protection	Input	25	21
	Output	26	24
PLC I/O Mode Assembly	Input	7	23
	Output	8	27
PLC I/O Mode Assembly with Reset Protection	Input	27	23
	Output	28	27

For Firmware 4.25

Assembly	Type	Instance Value	Size Value
ZPA Mode Assembly	Input	5	21
	Output	6	25
ZPA Mode Assembly with Reset Protection	Input	25	21
	Output	26	25
Reduced Size ZPA Mode Assembly	Input	19	12
	Output	20	15
Reduced Size ZPA Mode Assembly with Reset Protection	Input	39	12
	Output	40	15
PLC I/O Mode Assembly	Input	7	23
	Output	8	27
PLC I/O Mode Assembly with Reset Protection	Input	27	23
	Output	28	27
Reduced Size PLC I/O Mode Assembly	Input	17	9
	Output	18	9
Reduced Size PLC I/O Mode Assembly with Reset Protection	Input	37	9
	Output	38	9
ConveyLogix Assembly	Input	121	16
	Output	122	16

FOR FIRMWARE 5.02

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 ERSC 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:

ERSC Firmware	ERSC Mode	EDS File
4.24 and Earlier	**ZPA Mode Only**	**ConveyLinx_ZPA_Instance_1.eds**
4.24 and Earlier	**PLC I/O Mode Only**	**ConveyLinx_PLC_IO_Instance_1.eds**
4.25	**ZPA & PLC I/O Mode**	**ConveyLinx_V5_6.eds**
5.02 and Later	**ZPA & PLC I/O Mode**	**ConveyLinx_V5_6.eds**

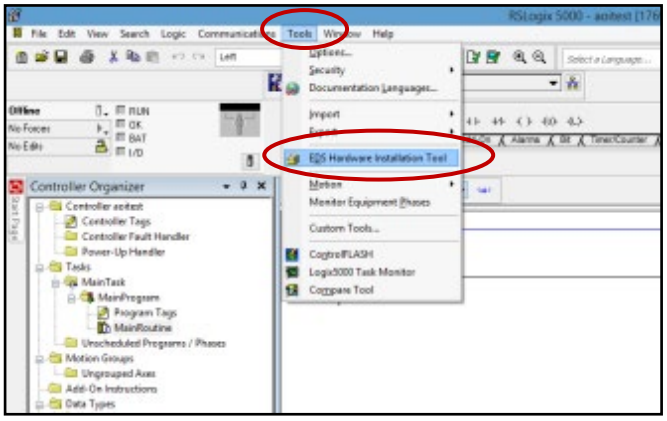
Please note that there may be updates since publication of this document. Please go to pulseroller.com to download the latest versions of EDS files when available.

For best results, you should remove any previous ERSC 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 Pulseroller into your RSLogix 5000 environment will allow you to select the ERSC 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 ERSC, the data is arranged in assembled registers as described in the *PLC Developer's Guide* with the data appearing in your Controller Tags similarly to how the data appears when you connect to an ERSC as a Generic Ethernet Module.

INSTALLING CONVEYLINX EDS FILE INTO RSLOGIX5000

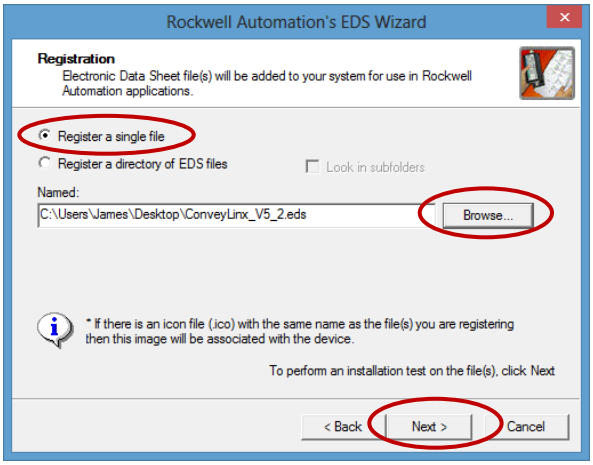


Step 1

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

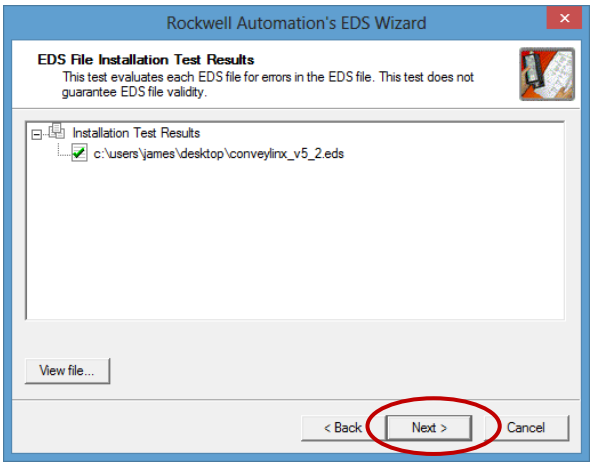
Select the register a single file radio button and click Browse and then browse to the location on your PC where you downloaded the EDS file. In this example we are installing the ZPA version. Click Next to continue.

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

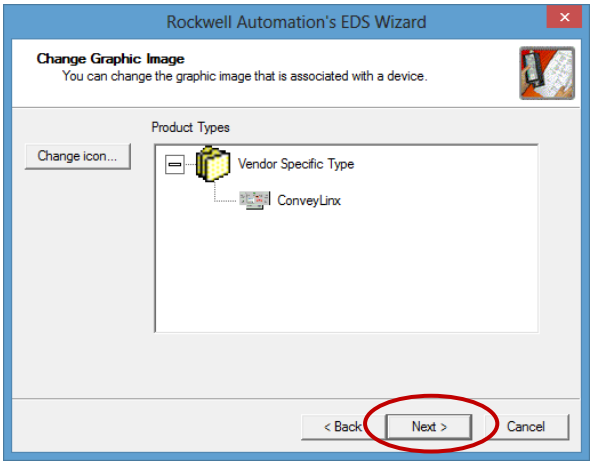
This window should appear with the green check indicating there were no errors. Click Next to continue.

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



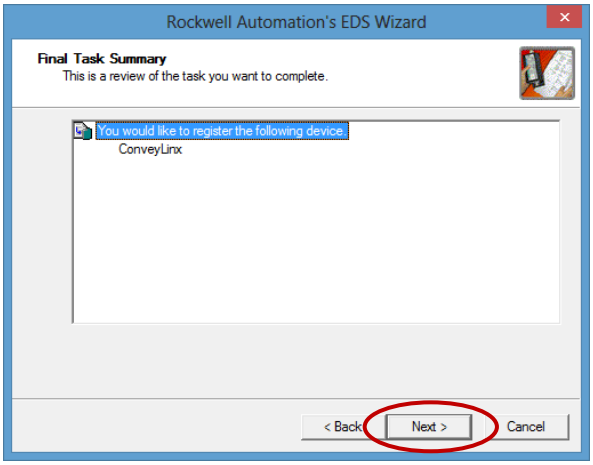
Step 5

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



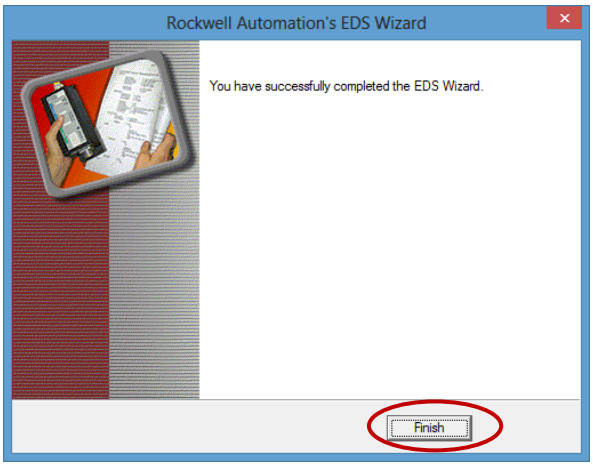
Step 6

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.

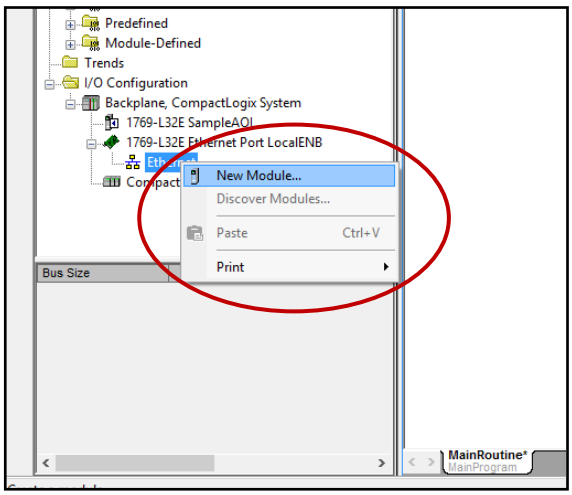
Creating a ZPA Mode ERSC Module in the Ethernet Tree

Once you have installed the EDS file into your RSLogix 5000 environment, you can now add specific instances of ERSC 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 ERSC to your program as an example.

Step 1

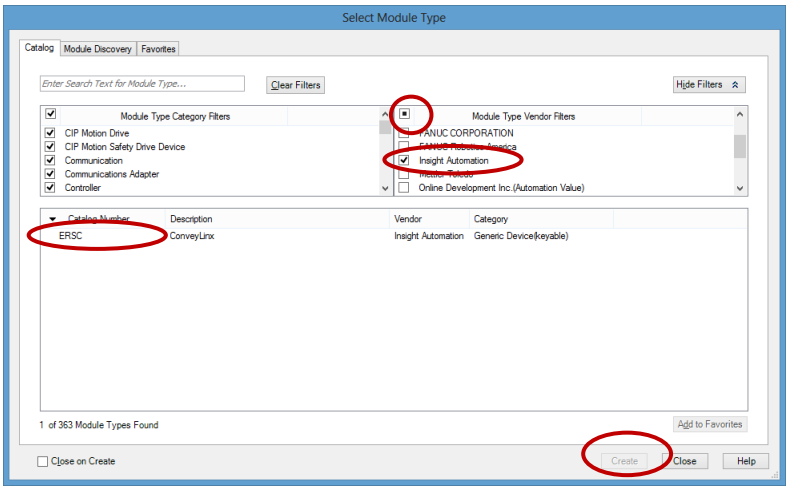
Right click on your Ethernet Tree and select *New Module* to open the *Select Module Type* window.



Step 2

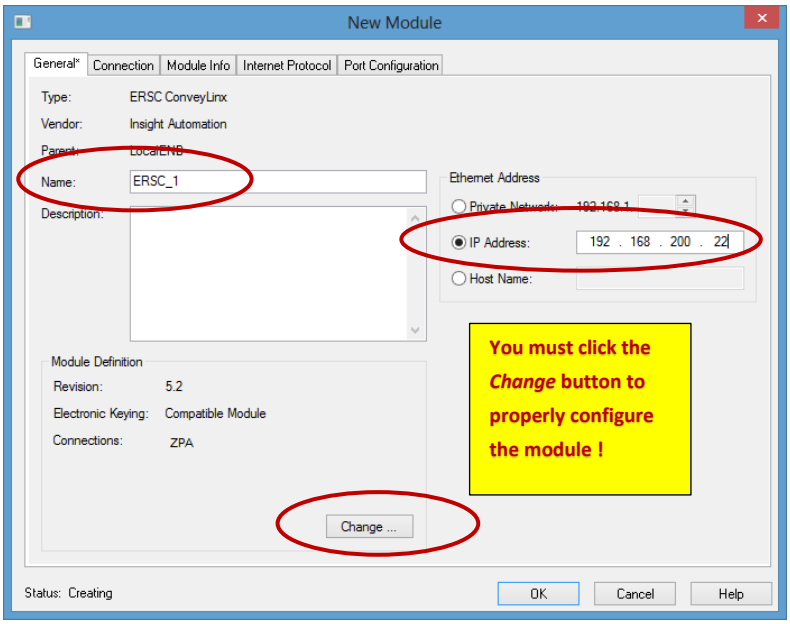
In the *Select Module Type* window, locate the ERSC catalog number. In this example we cleared the vendor filter and checked Insight Automation. Once you select ERSC, click the *Create* button to open the *New Module* window.

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.**

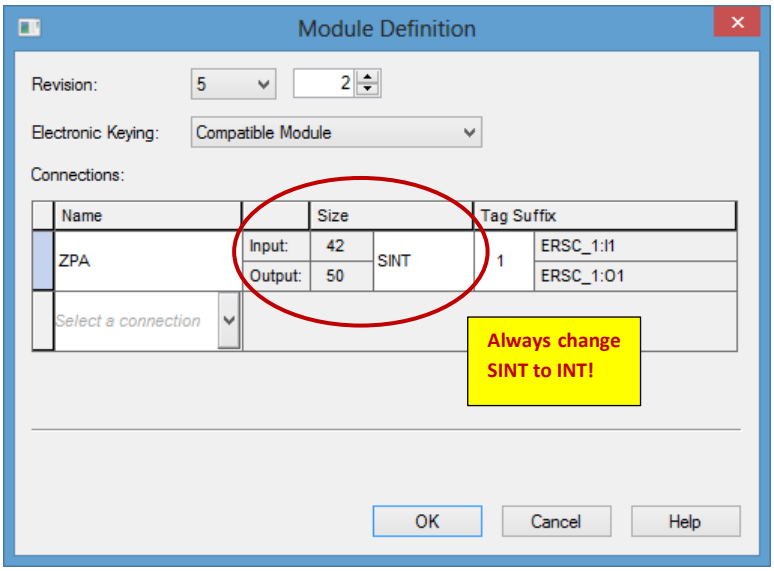


Step 4

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.

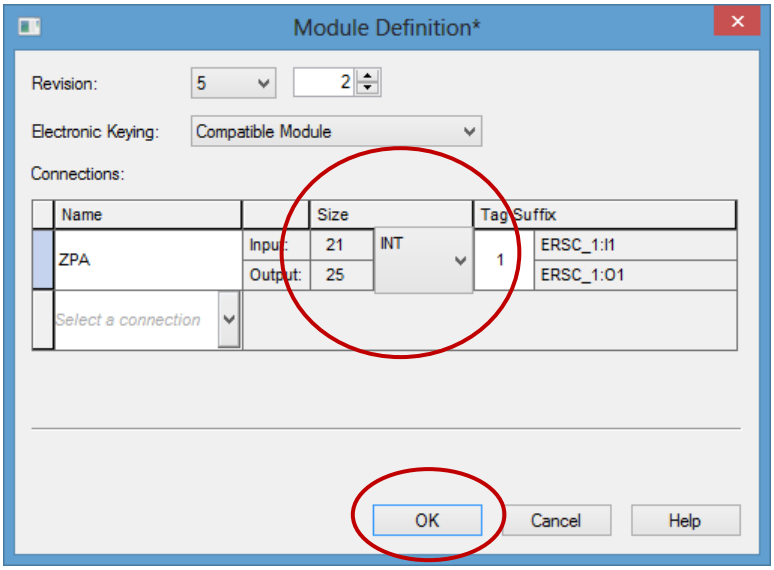
See Note ① below.

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.



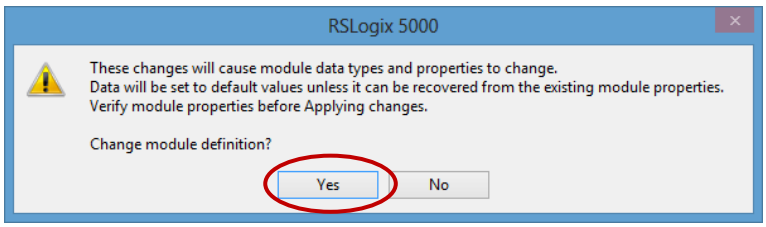
Click the right of the *Size* box to show a drop down box and then select INT

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 *PLC Developer's Guide* for ZPA mode. Click *OK*.



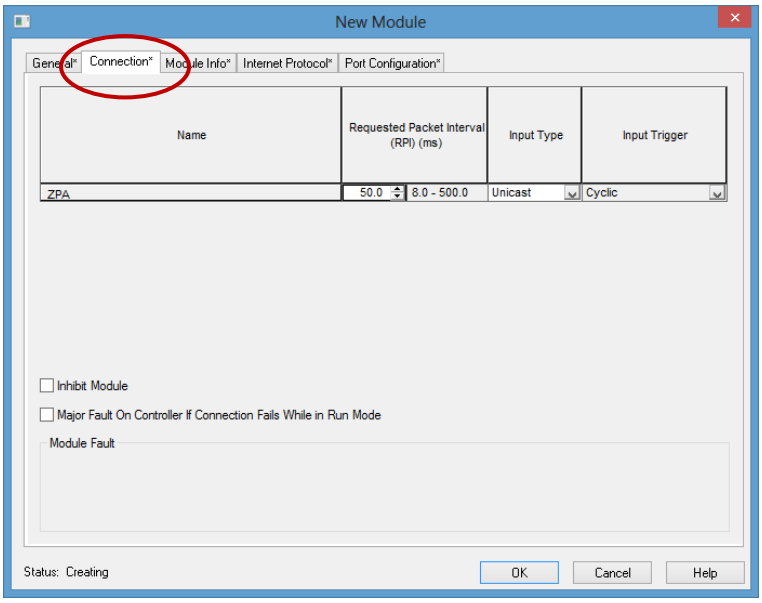
Step 5

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



Step 6

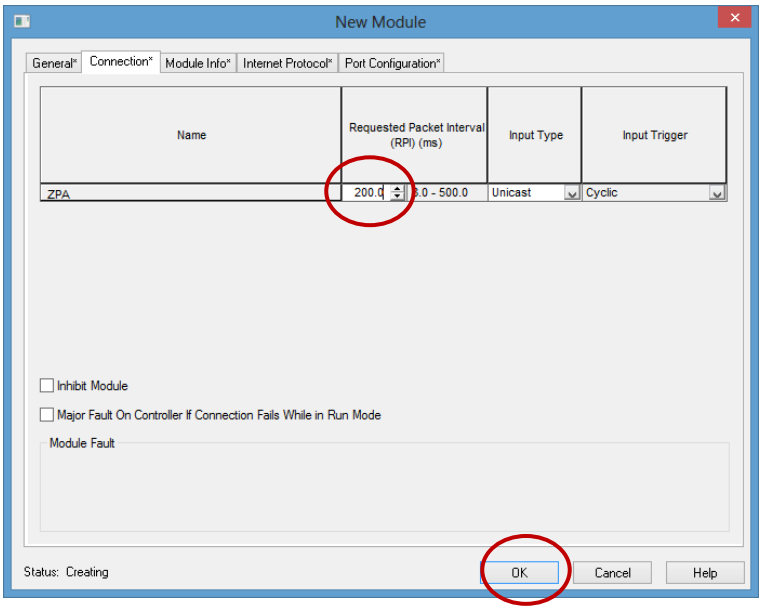
You should be back to the New Module window. You can change the RPI of the connection to your ERSC by clicking the Connection tab.



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 ERSC 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 ERSC was created. You can see that the quantities of INT registers correspond with the registers defined in the *PLC Developer's Guide* for ZPA mode.

Scope: SampleAOI Show: Unused Enter Name Filter...

Name	Value	Force Mask	Style	Data Type	Description
ERSC_1:I1	{...}	{...}		_055C:ERSC_84EED4D3:I:0	
ERSC_1:I1.ConnectionFaulted	0		Decimal	BOOL	
ERSC_1:I1.Data	{...}	{...}	Decimal	INT[21]	
ERSC_1:I1.Data[0]	0		Decimal	INT	
ERSC_1:I1.Data[1]	0		Decimal	INT	
ERSC_1:I1.Data[2]	0		Decimal	INT	
ERSC_1:I1.Data[3]	0		Decimal	INT	
ERSC_1:I1.Data[4]	0		Decimal	INT	
ERSC_1:I1.Data[5]	0		Decimal	INT	
ERSC_1:I1.Data[6]	0		Decimal	INT	
ERSC_1:I1.Data[7]	0		Decimal	INT	
ERSC_1:I1.Data[8]	0		Decimal	INT	
ERSC_1:I1.Data[9]	0		Decimal	INT	
ERSC_1:I1.Data[10]	0		Decimal	INT	
ERSC_1:I1.Data[11]	0		Decimal	INT	
ERSC_1:I1.Data[12]	0		Decimal	INT	
ERSC_1:I1.Data[13]	0		Decimal	INT	
ERSC_1:I1.Data[14]	0		Decimal	INT	
ERSC_1:I1.Data[15]	0		Decimal	INT	
ERSC_1:I1.Data[16]	0		Decimal	INT	
ERSC_1:I1.Data[17]	0		Decimal	INT	
ERSC_1:I1.Data[18]	0		Decimal	INT	
ERSC_1:I1.Data[19]	0		Decimal	INT	
ERSC_1:I1.Data[20]	0		Decimal	INT	
ERSC_1:O1	{...}	{...}		_055C:ERSC_3015BAF1:O:0	

FIGURE 3 - ZPA MODE CONTROLLER TAGS FOR INPUTS

Scope: SampleAOI Show: Unused Enter Name Filter...

Name	Value	Force Mask	Style	Data Type	Description
ERSC_1:I1	{...}	{...}		_055C:ERSC_84EED4D3:I:0	
ERSC_1:I1.ConnectionFaulted	0		Decimal	BOOL	
ERSC_1:I1.Data	{...}	{...}	Decimal	INT[21]	
ERSC_1:O1	{...}	{...}		_055C:ERSC_3015BAF1:O:0	
ERSC_1:O1.Data	{...}	{...}	Decimal	INT[25]	
ERSC_1:O1.Data[0]	0		Decimal	INT	
ERSC_1:O1.Data[1]	0		Decimal	INT	
ERSC_1:O1.Data[2]	0		Decimal	INT	
ERSC_1:O1.Data[3]	0		Decimal	INT	
ERSC_1:O1.Data[4]	0		Decimal	INT	
ERSC_1:O1.Data[5]	0		Decimal	INT	
ERSC_1:O1.Data[6]	0		Decimal	INT	
ERSC_1:O1.Data[7]	0		Decimal	INT	
ERSC_1:O1.Data[8]	0		Decimal	INT	
ERSC_1:O1.Data[9]	0		Decimal	INT	
ERSC_1:O1.Data[10]	0		Decimal	INT	
ERSC_1:O1.Data[11]	0		Decimal	INT	
ERSC_1:O1.Data[12]	0		Decimal	INT	
ERSC_1:O1.Data[13]	0		Decimal	INT	
ERSC_1:O1.Data[14]	0		Decimal	INT	
ERSC_1:O1.Data[15]	0		Decimal	INT	
ERSC_1:O1.Data[16]	0		Decimal	INT	
ERSC_1:O1.Data[17]	0		Decimal	INT	
ERSC_1:O1.Data[18]	0		Decimal	INT	
ERSC_1:O1.Data[19]	0		Decimal	INT	
ERSC_1:O1.Data[20]	0		Decimal	INT	
ERSC_1:O1.Data[21]	0		Decimal	INT	
ERSC_1:O1.Data[22]	0		Decimal	INT	
ERSC_1:O1.Data[23]	0		Decimal	INT	
ERSC_1:O1.Data[24]	0		Decimal	INT	

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 *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 in the next section), you **must change the data type to INT** because these items are written expecting INT data type.

NOTE ② - RPI SETTINGS

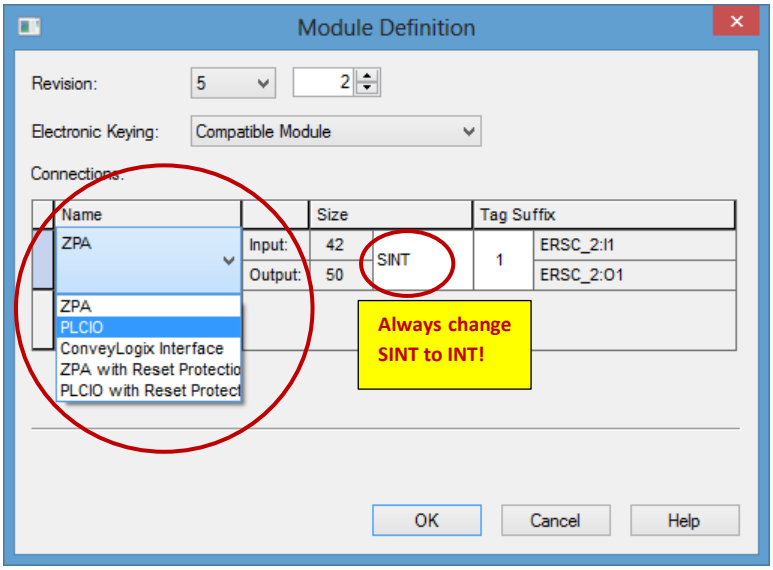
Please note that RPI settings do not affect the ERSC nearly as much as the PLC's Ethernet port's throughput. A combination of the quantity of ERSC connections along with small RPI values can create a bottleneck at the PLC's Ethernet port. A higher quantity of ERSC connections coupled with a small RPI for each can result in dropped connections to all devices (ERSC and other connected Ethernet devices). This is not an issue with the ERSC (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 ERSC, with the exception of changing the default connection type of ZPA in *Step 4* to the connection you need for the particular ERSC 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 ERSCs you are connecting to be set to the proper corresponding mode. If your connection is PLC I/O mode, the ERSC must be placed in PLC I/O mode using EasyRoll. Similarly for ConveyLogix Interface connection, the ERSC 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 ERSC 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

Pulseroller 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 ERSC module regardless of mode; you PLC program needs to directly access the register data array tags created when you created the ERSC instance. The AOIs attach to created ERSC'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 mode of the ERSC 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 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 ERSC firmware version and mode of operation:

ERSC Firmware	ERSC Mode	EDS File	AOI File
4.24 and Earlier	ZPA Mode	ConveyLinx_ZPA_Instance_1.eds	ERSC_ZPA_424.L5X
4.24 and Earlier	PLC I/O Mode	ConveyLinx_PLC_IO_Instance_1.eds	ERSC_PLCIO_424.L5X
4.25	ZPA Mode	ConveyLinx_V5_6.eds	ERSC_ZPA_425_5xx.L5X
4.25	PLC I/O Mode	ConveyLinx_V5_6.eds	ERSC_PLCIO_425_5xx.L5X
5.02 and Later	ZPA Mode	ConveyLinx_V5_6.eds	ERSC_ZPA_425_5xx.L5X
5.02 and Later	PLC I/O Mode	ConveyLinx_V5_6.eds	ERSC_PLCIO_425_5xx.L5X

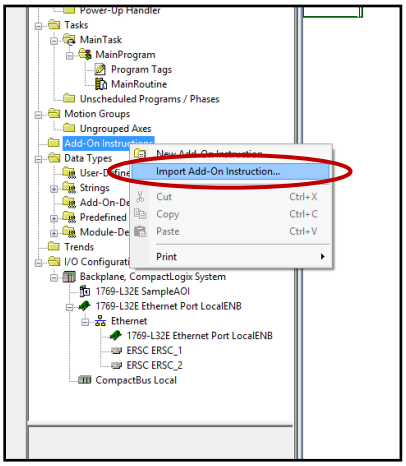
Please note that there may be updates since publication of this document. We recommend that you please go to pulseroller.com to download the latest versions of AOI files when available.

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

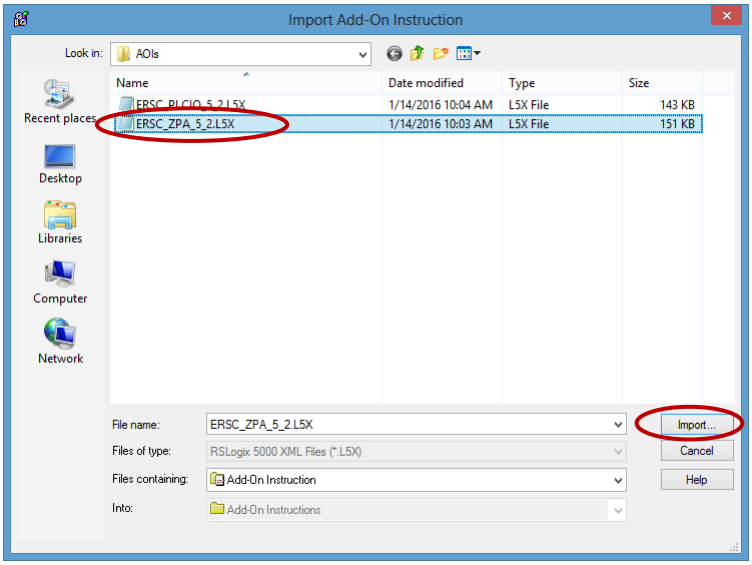
Right click on the *Add On Instruction* folder in the explorer tree. From the pop-up menu select Import *Add On Instruction…*



Step 2

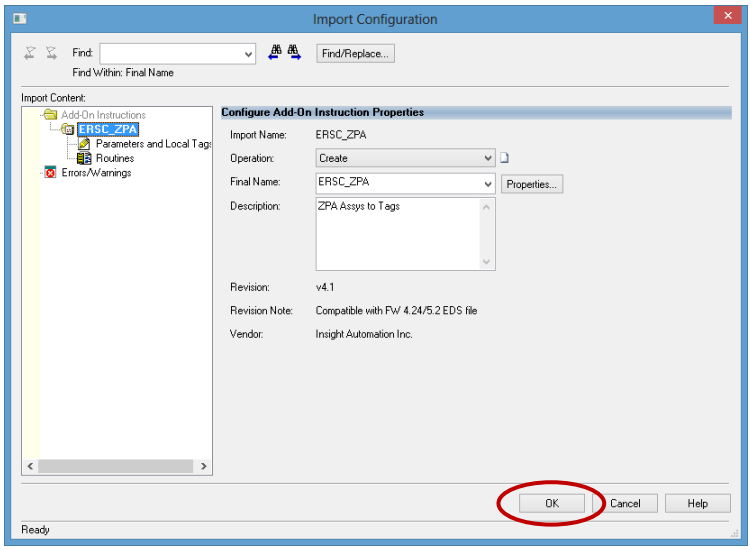
Navigate to the folder location where you downloaded your AOIs, select the file then click import. In this example we are importing the AOI for ZPA mode.

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.



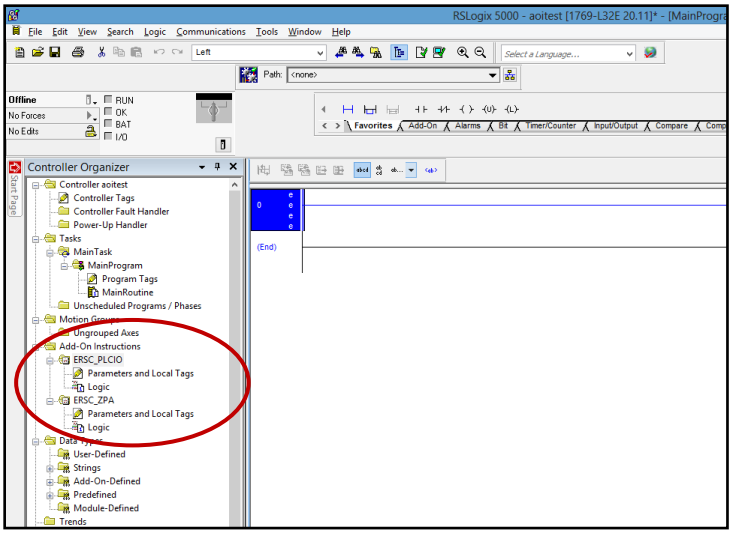
Step 3

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.



Simply repeat this process to import the other AOI for the ERSC in PLC I/O mode.

When you are done, these AOIs will appear in the explorer tree as shown.



EXAMPLE FOR ASSIGNING AOI TO ERSC MODULES IN YOUR PROJECT

For our example, we are going to add one ZPA mode ERSC and one PLC I/O mode ERSC to our current project. This was added following the steps outlined in section *Creating a ZPA Mode ERSC Module in the Ethernet Tree* beginning on page 21. For our example we are assuming this module has been configured with I.P. addresse 192.168.200.22. Also, the module connection has been changed to *ZPA with Reset Protection* connection as outlined in section *Creating Other Connection Types* on page 27.

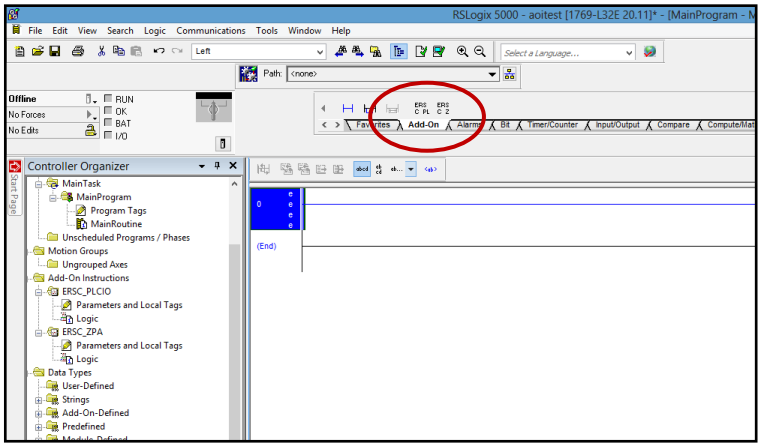
Pulseroller provided AOI's require connections to be "with Reset Protection". Please refer to *Creating Other Connection Types* on page 27 for details on creating these connections. Also refer to the *PLC Developer's Guide* for details on Reset Protection assemblies.

Assigning New Modules to AOI

Now that we have our ERSC's defined with their correct connection types and the AOI instructions imported into our RSLogix5000 project; the next step is to create an instance of the appropriate AOI for each physical ERSC.

Step 1

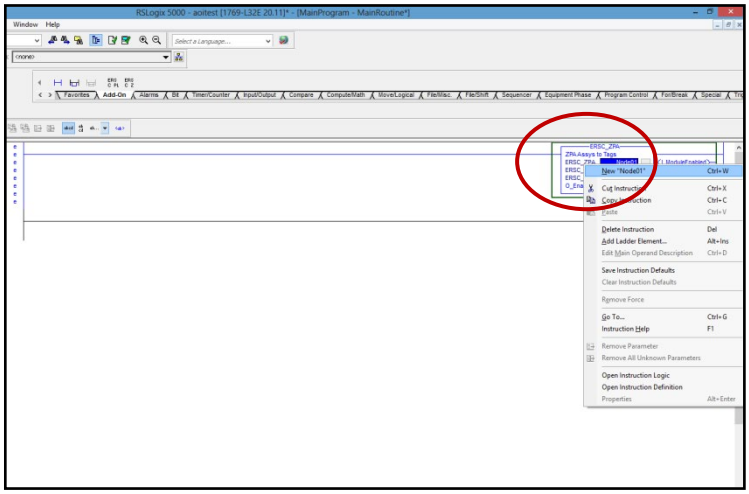
Locate the AOIs and place in your ladder diagram. For our example we are selecting the ERSC_ZPA instruction

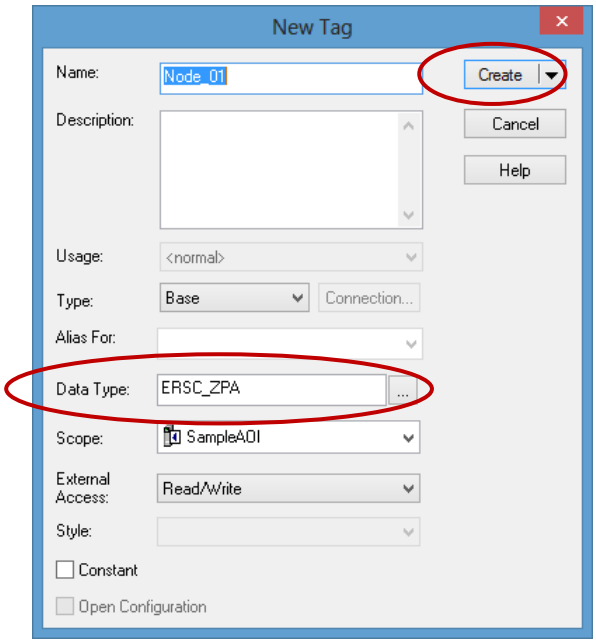


Step 2

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 entered "Node_01" and then created the new tag



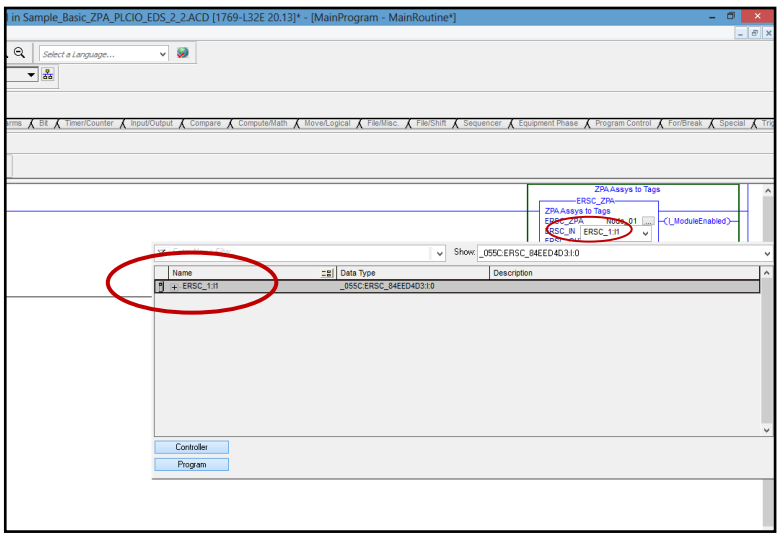


Step 3

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

Step 4

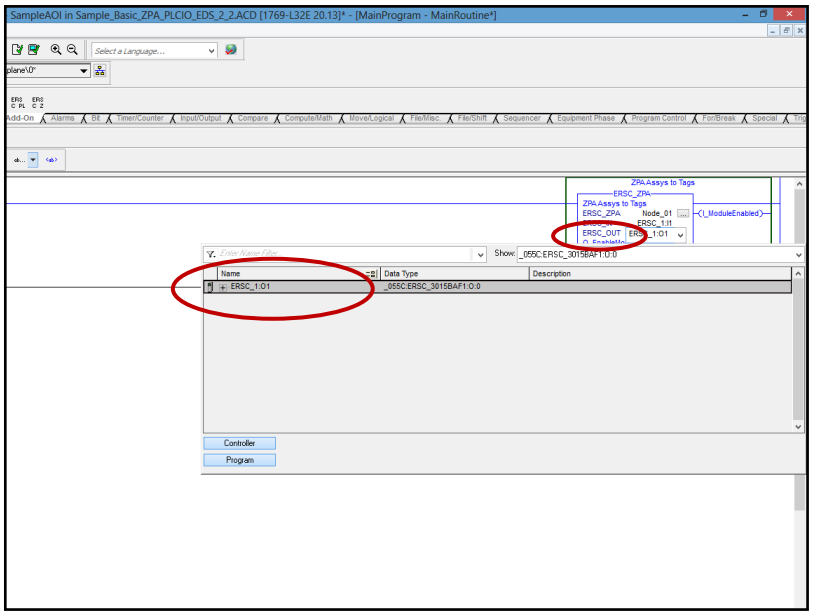
The AOI requires two other parameters; *ERSC_IN* for the data coming from the module and *ERSC_OUT* for data from the PLC to the module. These will be assigned to the I/O arrays created by the EDS file when we previously added the module. Here we will assign the *ERSC_IN* parameter by clicking the drop-down box arrow to automatically show all tags that match the data type for the *ERSC_IN* parameter. In this case, *ERSC_1* is the only ZPA module we created, so it is the only selection. Double click this and it will be assigned to the *ERSC_IN* parameter of our ZPA AOI.



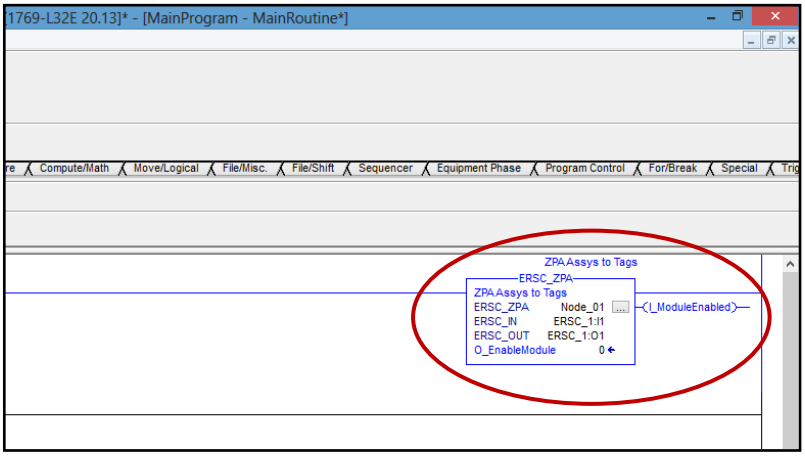
Step 5

Similarly to ***Step 4***, we need to select the physical module for the *ERSC_OUT* parameter. Clicking the drop down box arrow will show all physical modules that have the matching data type for the ERSC_OUT parameter.

Double click this and it will be assigned to the *Node_01* instance of our ZPA AOI's *ERSC_OUT* parameter.



At this point, the AOI has been set up to use in your logic program. All of the tags associated with using the *ERSC_1* module in ZPA mode are in the structured tag *Node_01*.



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

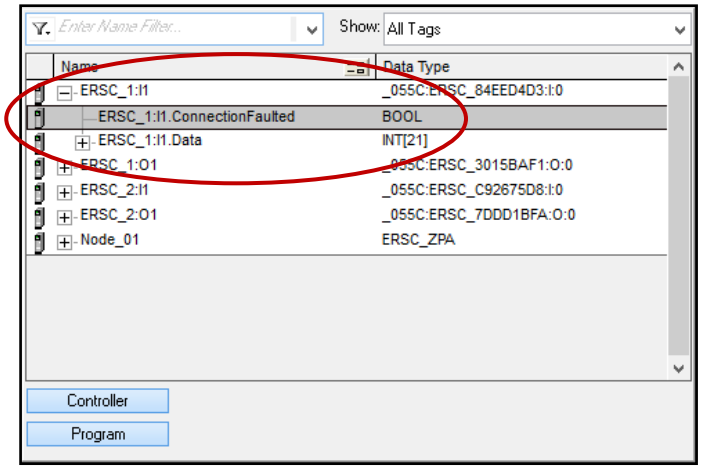
Enabling the Module for Operation

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 ERSC module to process output data coming from the PLC.

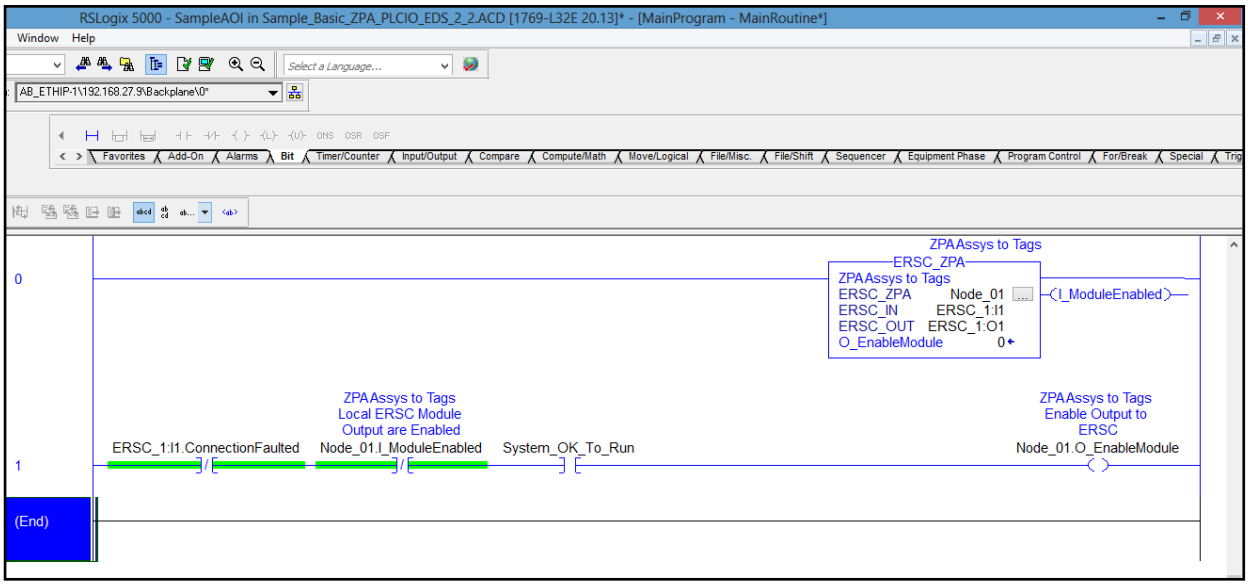
The Reset Protection topic is covered in publication ERSC-1500 PLC Developer's Guide

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 (ERSC_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 "ERSC_1:I" 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 start. 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 the ERSC_PLCIO AOI, so this same type of rung is needed to enable these.

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

Please note that even if the ERSC 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 ERSC-ZPA AOI along with the register reference from the PLC Developer's Guide 4.7 and later or ConveyLinx-Ai PLC Developer's Guide version 1.0 and later.

Tag Name	Data Type	Developer's Guide Register	Bit	Description
I_ArrivalUPZn	BOOL	4:0116	AOI Logic	Arrival at Local Upstream Zone – Only active if zone becomes occupied and it has been set to accumulate
I_ArrivalDNZn	BOOL	4:0196	AOI Logic	Arrival at Local Downstream Zone – Only active if zone becomes occupied and it has been set to accumulate
I_ControlPortPin3_Left	BOOL	4:0035	1	Left Control Port Pin 3 Energized
I_ControlPortPin3_Right	BOOL	4:0035	3	Right Control Port Pin 3 Energized
I_ControlPortPin4_Left	BOOL	4:0035	5	Left Control Port Pin 4 Energized
I_ControlPortPin4_Right	BOOL	4:0035	7	Right Control Port Pin 4 Energized
I_ConveyStopByLeftControlPort	BOOL	4:0020	8	ConveyStop Activated at Local Left Control Port
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 Stop Group
I_ConveyStopByRightControlPort	BOOL	4:0020	9	ConveyStop Activated at Local Right Control Port
I_GetForwardTracking	DINT	4:0201 (MSW) 4:0202 (LSW)	-	Current Forward Tracking Value at Induct to Local Upstream Zone
I_Heartbeat	BOOL	4:0035	15	Module Heartbeat
I_JamAtUPZn	BOOL	4:0088	5	Sensor Jam at Local Upstream Zone
I_JamAtDNZn	BOOL	4:0089	5	Sensor Jam at Local Downstream Zone
I_ModuleEnabled	BOOL	-	-	Local ERSC Module Output are Enabled
I_ModuleFault	BOOL	4:0088 4:0089	AOI Logic	Module Fault Active (Logical OR of bits 2,4,and 7 from 5-6 and bits 2 and 7 from 5-7
I_ModuleStatus	DINT	4:0088 (MSW) 4:0089 (LSW)		Modules Status Words 1 and 2
I_MtrError_Left	BOOL	4:0088	3	Left Motor Error is Active
I_MtrError_Right	BOOL	4:0089	3	Right Motor Error is Active
I_SensorPortPin3_Left	BOOL	4:0035	0	Left Sensor Port Pin 3 Energized
I_SensorPortPin3_Right	BOOL	4:0035	2	Right Sensor Port Pin 3 Energized
I_SensorPortPin4_Left	BOOL	4:0035	4	Left Sensor Port Pin 4 Energized
I_SensorPortPin4_Right	BOOL	4:0035	6	Right Sensor Port Pin 4 Energized
I_TrackingDNZn	DINT	4:0199 (MSW) 4:0200 (LSW)	-	Current Tracking Value for Arrival at Local Downstream Zone
I_TrackingUPZn	DINT	4:0119 (MSW) 4:0120 (LSW)	-	Current Tracking Value for Arrival at Local Upstream Zone
I_ZoneStatusDnZn	SINT	4:0196	Lo Byte	Zone Status Local Downstream Zone Forward Direction
I_ZoneStatusUpZn	SINT	4:0116	Lo Byte	Zone Status Local Uptream Zone Forward Direction
O_AccForArrivalDNZn	BOOL	4:0184	0	Set Local Downstream Zone to Accumulate
O_AccForArrivalUPZn	BOOL	4:0104	0	Set Local Upstream Zone to Accumulate
O_AccumAdjUpstreamToDNZn	BOOL	4:0184	8	Accumulate Adjacent Upstream to Local Downstream Zone
O_AccumAdjUpstreamToUPZn	BOOL	4:0104	8	Accumulate Adjacent Upstream to Local Upstream Zone
O_ClearJamDNZn	BOOL	4:0189	0	Clear Jam at Local Upstream Zone
O_ClearJamUPZn	BOOL	4:0109	0	Clear Jam at Local Downstream Zone
O_ClearMotorError	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_ControlPortOutputLeft	BOOL	4:0063	1	Set Left Control Port Output

Tag Name	Data Type	Developer's Guide Register	Bit	Description
O_ControlPortOutputRight	BOOL	4:0063	3	Set Right COntrol Port Output
O_ConveyMerge_DisableCenter	BOOL	4:0387	4	Set to disable center release
O_ConveyMerge_DisableLeft	BOOL	4:0387	5	Set to disable left release
O_ConveyMerge_DisableRight	BOOL	4:0387	6	Set to disable right release
O_ConveyMerge_EnablePLCCtrl	BOOL	4:0387	15	Set to enable PLC over-ride of configured ConveyMerge
O_ConveyMergePriority	SINT	4:0387	-	Numerical value to set merge priority
O_ConveyStopCommand	INT	4:0020	-	Set Local ConveyStop Command Word
O_DAModeCmdDNZn	SINT	4:0375	Lo Byte	Direction & Accumulation Mode Command Byte for Downstream Zone
O_DAModeCmdUPZn	SINT	4:0365	Lo Byte	Direction & Accumulation Mode Command Byte for Upstream Zone
O_DAModeValueDNZn	SINT	4:0375	Hi Byte	Direction & Accumulation Mode Data Byte for Downstream Zone
O_DAModeValueUPZn	SINT	4:0365	Hi Byte	Direction & Accumulation Mode Data Byte for Upstream Zone
O_EnableModule	BOOL	-	AOI Logic	Enable Output to ERSC
O_JogFwdDNZn	BOOL	4:0184	10	Jog Forward for Local Downstream Zone
O_JogFwdUpZn	BOOL	4:0104	10	Jog Forward for Local Upstream Zone
O_JogRevDNZn	BOOL	4:0184	11	Jog Reverse for Local Downstream Zone
O_JogRevUPZn	BOOL	4:0104	11	Jog Reverse for Local Upstream Zone
O_ReleaseDNZn	BOOL	4:0105	AOI Logic	Release and Accumulate on Next at Downstream Zone – Automatically increments release counter
O_ReleaseUPZn	BOOL	4:0185	AOI Logic	Release and Accumulate on Next at Upstream Zone – Automatically increments release counter
O_SpeedLeftMtr	INT	4:0040	-	Set Left Motor Speed Reference
O_SpeedRightMtr	INT	4:0064	-	Set Right Motor Speed Reference
O_StatusDownstreamDischarge	INT	4:0232	-	Set Downstream Discharge Zone Status Value
O_StatusUpstreamInduct	INT	4:0134	-	Set Upstream Induct Zone Status Value
O_TrackingDNZn	DINT	4:0212 (MSW) 4:0213 (LSW)	-	Set Tracking Value for Local Downstream Zone
O_TrackingInductFwd	DINT	4:0139 (MSW) 4:0140 (LSW)	-	Set Forward Induct Tracking Value
O_TrackingUPZn	DINT	4:0132 (MSW) 4:0133 (LSW)	-	Set Tracking Value for Local Upstream Zone
O_WakeUpDNZn	BOOL	4:0184	12	Wakeup Local Downstream Zone
O_WakeUpUPZn	BOOL	4:0104	12	Wakeup Local Upstream Zone

PLC I/O AOI Tag Descriptions

The following chart lists each tag made available in the ERSC-PLCIO AOI along with the register reference from the *PLC Developer's Guide 4.7* and later.

Tag Name	Data Type	Developer's Guide Register	Bit	Decription
I_ControlPortPin3_Left	BOOL	4:0019	1	Port Inputs
I_ControlPortPin3_Right	BOOL	4:0035	3	Port Inputs
I_ControlPortPin4_Left	BOOL	4:0035	5	Port Inputs
I_ControlPortPin4_Right	BOOL	4:0035	7	Port Inputs
I_ConveyStopByLeftControlPort	BOOL	4:0019	8	ConveyStop
I_ConveyStopByLostConnection	BOOL	4:0019	6	ConveyStop
I_ConveyStopByPLCDisconnect	BOOL	4:0019	7	ConveyStop
I_ConveyStopByPLCCmd	BOOL	4:0019	10	ConveyStop
I_ConveyStopByRemoteModule	BOOL	4:0019	5	ConveyStop
I_ConveyStopByRightControlPort	BOOL	4:0019	9	ConveyStop
I_DigitalMtrOverCurrent_Left	BOOL	4:0060	14	Left Motor Port as Digital
I_DigitalMtrOverCurrent_Right	BOOL	4:0084	14	Right Motor Port as Digital
I_DigitalMtrShortCkt_Left	BOOL	4:0060	12	Left Motor Port as Digital
I_DigitalMtrtShortCkt_Right	BOOL	4:0084	12	Right Motor Port as Digital
I_DownstreamModuleStatus	SINT	4:0232	Lo Byte	Module Status
I_Heartbeat	BOOL	4:0035	15	Port Inputs
I_ModuleEnabled	BOOL	-	-	Module Status
I_ModuleVoltage	REAL	4:0024	-	Module Status
I_MtrCurrent_Left	REAL	4:0055	-	Left Motor Status
I_MtrCurrent_Right	REAL	4:0079	-	Right Motor Status
I_MtrFreq_Left	INT	4:0056	-	Left Motor Status
I_MtrFreq_Right	INT	4:0080	-	Right Motor Status
I_MtrRunningCCW_Left	BOOL	4:0058	AOI Logic	Left Motor Status
I_MtrRunningCCW_Right	BOOL	4:0082	AOI Logic	Right Motor Status
I_MtrRunningCW_Left	BOOL	4:0058	AOI Logic	Left Motor Status
I_MtrRunningCW_Right	BOOL	4:0082	AOI Logic	Right Motor Status
I_MtrStatus_Left	INT	4:0058	-	Left Motor Status
I_MtrStatus_Right	INT	4:0082	-	Right Motor Status
I_SensorDetectLeftPort	BOOL	4:0036	1	Sensor Port Status
I_SensorDetectRightPort	BOOL	4:0036	0	Sensor Port Status
I_SensorPortPin3_Left	BOOL	4:0035	0	Port Inputs
I_SensorPortPin3_Right	BOOL	4:0035	2	Port Inputs
I_SensorPortPin4_Left	BOOL	4:0035	4	Port Inputs
I_SensorPortPin4_Right	BOOL	4:0035	6	Port Inputs
I_ServoCmdStatus_Left	BOOL	4:0011	2	Left Servo Function
I_ServoCmdStatus_Right	BOOL	4:0016	2	Right Servo Function
I_ServoLastCmdComplete_Left	BOOL	4:0011	0	Left Servo Function
I_ServoLastCmdComplete_Right	BOOL	4:0016	0	Right Servo Function
I_ServoPosition_Left	INT	4:0062		Left Servo Function
I_ServoPosition_Right	INT	4:0086		Right Servo Function
I_ServoResetStatus_Left	BOOL	4:0011	1	Left Servo Function
I_ServoResetStatus_Right	BOOL	4:0016	1	Right Servo Function
I_TemperatureCalculated_Left	SINT	4:0057	Hi Byte	Left Motor Status
I_TemperatureCalculated_Right	SINT	4:0081	Hi Byte	Right Motor Status
I_TemperatureOnBoard_Left	SINT	4:0057	Lo Byte	Left Motor Status
I_TemperatureOnBoard_Right	SINT	4:0081	Lo Byte	Right Motor Status
I_UpstreamModuleStatus	SINT	4:0134	Lo Byte	ZPA Status

Tag Name	Data Type	Developer's Guide Register	Bit	Decription
I_UpstreamTracking	DINT	4:0139 (MSW) 4:0140 (LSW)		ZPA Tracking
O_BrakeMethod_Left	SINT	4:0261	Lo Byte	Left Motor Control
O_BrakeMethod_Right	SINT	4:0271	Lo Byte	Right Motor Control
O_ClearMotorError	BOOL	4:0022		Motor Control
O_ControlPortOutput_Left	BOOL	4:0037	1	Left Motor Control
O_ControlPortOutput_Right	BOOL	4:0037	3	Right Motor Control
O_ControlPortPin3Mask_Left	BOOL	4:0034	1	Sensor/Control Port Configuration
O_ControlPortPin3Mask_Right	BOOL	4:0034	3	Sensor/Control Port Configuration
O_ControlPortPin4Mask_Left	BOOL	4:0034	5	Sensor/Control Port Configuration
O_ControlPortPin4Mask_Right	BOOL	4:0034	7	Sensor/Control Port Configuration
O_ConveyStopCommand	INT	4:0020		ConveyStop
O_DischargeTracking	DINT	4:0201 (MSW) 4:0202 (LSW)		ZPA Tracking
O_DownstreamStatus	SINT	4:0196	Lo Byte	ZPA Status
O_EnableModule	BOOL	-	-	Module Control
O_LeftMtrDigitalPin3	BOOL	4:0060	0	Left Motor Port Digital Control
O_LeftMtrDigitalPin4	BOOL	4:0060	1	Left Motor Port Digital Control
O_LeftMtrDigitalPin5	BOOL	4:0060	2	Left Motor Port Digital Control
O_MtrAccel_Left	INT	4:0043		Left Motor Control
O_MtrAccel_Right	INT	4:0067		Right Motor Control
O_MtrDecel_Left	INT	4:0044		Left Motor Control
O_MtrDecel_Right	INT	4:0068		Right Motor Control
O_RightMtrDigitalPin3	BOOL	4:0084	0	Right Motor Port Digital Control
O_RightMtrDigitalPin4	BOOL	4:0084	1	Right Motor Port Digital Control
O_RightMtrDigitalPin5	BOOL	4:0084	2	Right Motor Port Digital Control
O_RunMtrFwd_Left	BOOL	4:0260	1	Left Motor Control
O_RunMtrFwd_Right	BOOL	4:0270	1	Right Motor Control
O_RunMtrRev_Left	BOOL	4:0260	8	Left Motor Control
O_RunMtrRev_Right	BOOL	4:0270	8	Right Motor Control
O_SensorPortPin3Mask_Left	BOOL	4:0034	0	Sensor/Control Port Configuration
O_SensorPortPin3Mask_Right	BOOL	4:0034	2	Sensor/Control Port Configuration
O_SensorPortPin4Mask_Left	BOOL	4:0034	4	Sensor/Control Port Configuration
O_SensorPortPin4Mask_Right	BOOL	4:0034	6	Sensor/Control Port Configuration
O_ServoCmdPulses_Left	INT	4:0008		Left Servo Function
O_ServoCmdPulses_Right	INT	4:0013		Right Servo Function
O_ServoGoCmd_Left	BOOL	4:0009	1	Left Servo Function
O_ServoGoCmd_Right	BOOL	4:0014	1	Right Servo Function
O_ServoZero_Left	BOOL	4:0009	0	Left Servo Function
O_ServoZero_Right	BOOL	4:0014	0	Right Servo Function
O_LeftMtrDigital_Enable	BOOL	4:0060	15	Left Motor Port Digital Control
O_RightMtrDigital_Enable	BOOL	4:0084	15	Right Motor Port Digital Control
O_SpeedMethod_Left	SINT	4:0262	Lo Byte	Left Motor Control
O_SpeedMethod_Right	SINT	4:0272	Lo Byte	Right Motor Control
O_SpeedReference_Left	INT	4:0040		Left Motor Control
O_SpeedReference_Right	INT	4:0064		Right Motor Control
O_UpstreamStatus	SINT	4:0116	Lo Byte	ZPA Status
O_LeftMtrDigital_BrakePinEnable_NoD	BOOL	4:0060	7	Enable the Left Motor Port Brake pin digital output
O_LeftMtrDigital_BrakePin	BOOL	4:0060	6	Left Motor Brake Pin Digital Control
O_LeftMtrDigital_ClearOC	BOOL	4:0060	8	Left Motor Digital Clear Over-current Error
O_RightMtrDigital_BrakePinEnable_NoD	BOOL	4:0084	7	Enable the Right Motor Port Brake pin digital output

Tag Name	Data Type	Developer's Guide Register	Bit	Decription
O_RightMtrDigital_BrakePin	BOOL	4:0084	6	Right Motor Brake Pin Digital Control
O_RightMtrDigital_ClearOC	BOOL	4:0084	8	Right Motor Digital Clear Over-current Error

Using Logix5000 MSG Instruction

Access to ConveyLinx ERSC 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 ERSC 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-1500 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 ERSC module. The ERSC 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 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

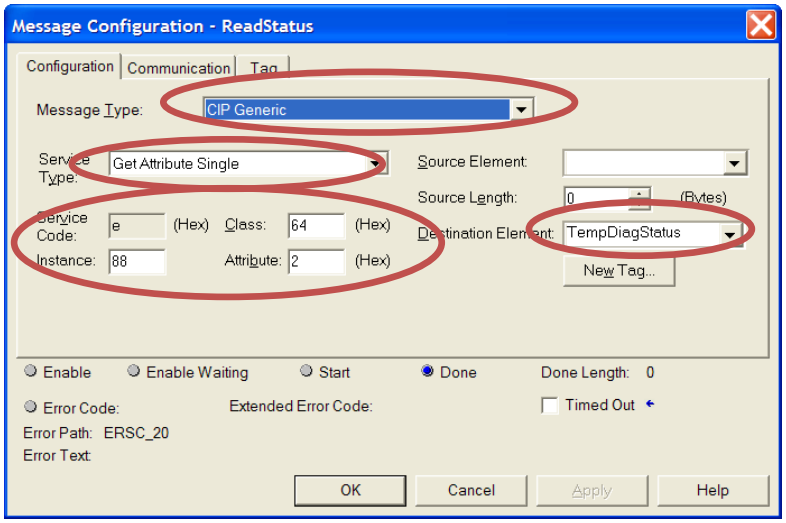
What this means is that there are certain restrictions on what you can do with MSG instructions with an ERSC. Here is a list:

- 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 ERSC 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 *ERSC*. In this example, "TempDiagStatus" is the user defined tag.



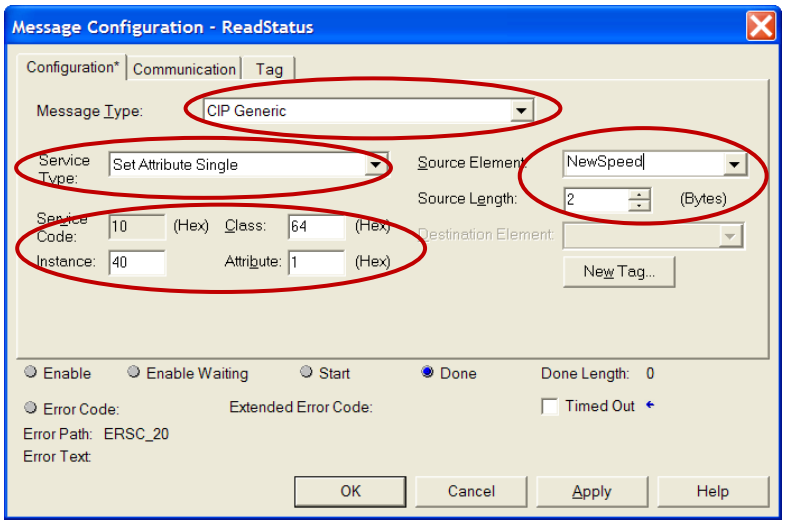
The acceptable values for "Attribute" are from 0x1 to 0x1E which is 1 to 30 contiguous registers. In the above example, the data being read is Module Status #1 and Module Status #2 registers (4:0088 and 4:0089). This same MSG instruction could be duplicated for each *ERSC* in ZPA mode in a given conveyor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

Message Configuration for Writing Data To ERSC 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 *ERSC*'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 "Destination 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 an *ERSC* in PLC I/O mode, it is assumed that any *ERSC* in PLC I/O will already be utilizing a permanent TCP connection and should not ever need to be accessed with a read MSG instruction.

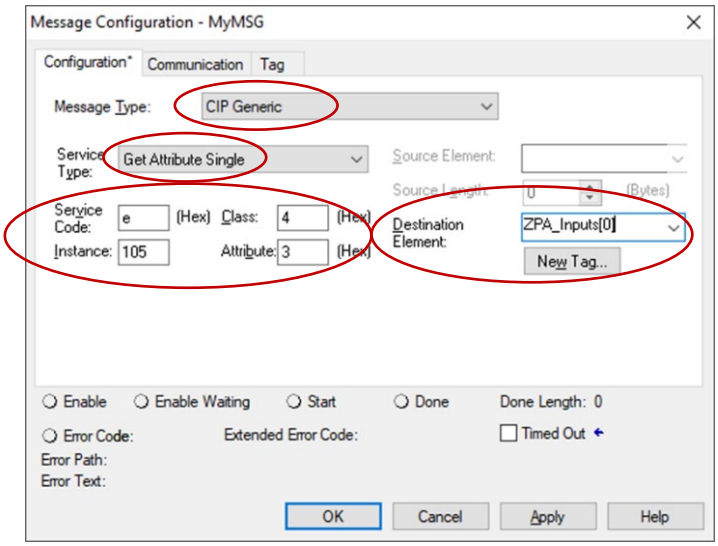
Refer to Allen-Bradley reference documentation for the particular PLC processor being used as to the proper usage and expected performance loading on the processor communication channels due to multiple MSG instructions executing simultaneously.

Message Configuration for Reading ERSC Assembled Registers

Please note this configuration is only valid for ERSC firmware versions 4.25 and higher and 5.02 and higher

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 *ERSC*. 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



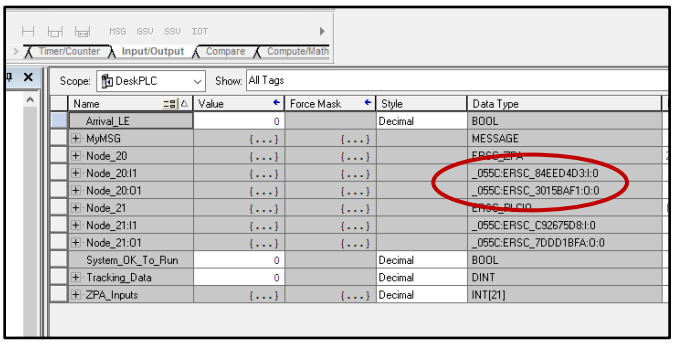
Please note that the data type of your *Destination Element* tag that you create must be an INT array equal to the size of the Assembly. Please refer to section *Parameters for Each Assembly* beginning on page 16 to determine the register array size required for each Assembly. Please consult Allen-Bradley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

APPENDIX A – EDS MODULE DATA TYPE CROSS REFERENCE

When you create an instance of a device from an EDS file in your RSLogix 5000 environment; the EDS file provides a Module Defined Data Type for the inputs and outputs of the device. This Module Defined Data Type's name is automatically generated by the EDS file creation's software and is based upon (among other things) a checksum of the items in the file. This often creates a somewhat cryptic alpha-numeric string for the name.

This example shows the EDS generated Module Data Type for both the inputs and outputs for a given instance of an ERSC Module. The "055C" indicates Insight Automation's ODVA Vendor ID. The remaining data is generated by the EDS file creation software based upon file content.



In situations where you may inherit an existing program and your RSLogix 5000 environment is missing the EDS file used for this program; you will need to determine which version of EDS file was used and then go find it on our Pulseroller.com web site. Similarly, you may also have the situation where you need to be able to match your AOI version to its correct EDS file. The following chart cross references the most common EDS files, AOI files, and Module Data Types:

EDS File	AOI	Module Data Type	Firmware
ConveyLinx_ZPA_Instance_1.eds	ERSC_ZPA_424.L5X	_055C:ERSC_ZPA_84EED4D3:I:0	4.24 and older
		_055C:ERSC_ZPA_FB496954:O:0	
ConveyLinx_PLC_IO_Instance_1.eds	ERSC_PLCIO_424.L5X	_055C:ERSC_PLCIO_C92675D8:I:0	4.24 and older
		_055C:ERSC_PLCIO_7DDD1BFA:O:0	
ConveyLinx_V5_4.eds	ERSC_ZPA_5_2.L5X	_055C:ERSC_84EED4D3:I:0	4.25 / 5.02 and newer
		_055C:ERSC_3015BAF1:O:0	
	ERSC_PLCIO_5_2.L5X	_055C:ERSC_C92675D8:I:0	4.25 / 5.02 and newer
		_055C:ERSC_7DDD1BFA:O:0	
ConveyLinx_V5_6.eds	ERSC_ZPA_425_5xx.L5X	_055C:ERSC_84EED4D3:I:0	4.25 / 5.02 and newer
		_055C:ERSC_3015BAF1:O:0	
	ERSC_PLCIO_425_5xx.L5X	_055C:ERSC_C92675D8:I:0	4.25 / 5.02 and newer +
		_055C:ERSC_7DDD1BFA:O:0	
ConveyLinx_ZPA_Instance.eds	ERSC_ZPA_424.L5X	_055C:ERSC_ZPA_84EED4D3:I:0	4.24 and older
		_055C:ERSC_ZPA_FB496954:O:0	
ConveyLinx_PLC_IO_Instance.eds	ERSC_PLCIO_424.L5X	_055C:ERSC_PLCIO_C92675D8:I:0	4.24 and older
		_055C:ERSC_PLCIO_7DDD1BFA:O:0	

For versions or Module Data Types not shown, please contact support@pulseroller.com

NOTES:

PULSE ROLLER

www.pulseroller.com

North & South America

sales@pulseroller.com

support@pulseroller.com

Global

Global_sales@pulseroller.com

Global_support@pulseroller.com