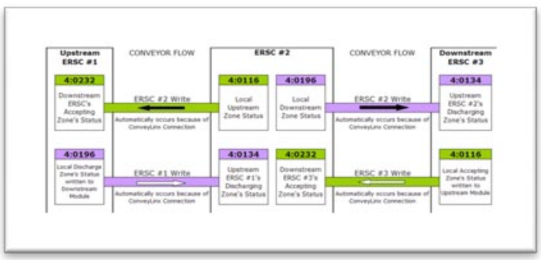
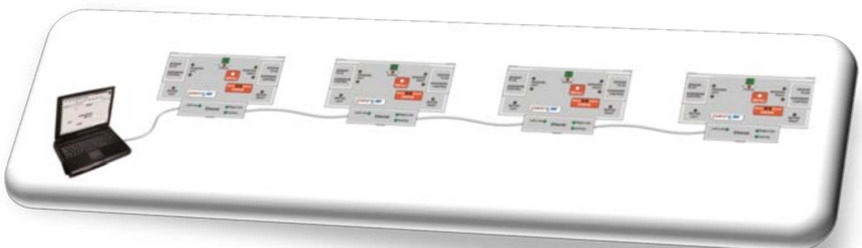
CONVEYLINX® Ethernet Modular Conveyor Controls

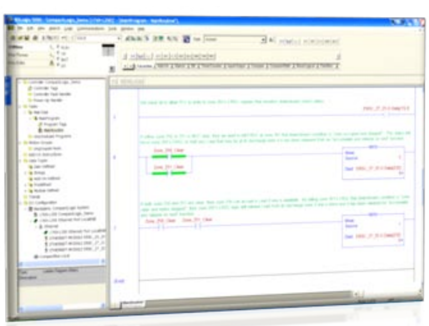
ERSC Developer's Guide

Version 4.0

June 2011







Insight Automation Publication *ERSC*-1500

Glossary of Terms

ConveyLinx	Conveyor controls architecture based upon modular distributed devices connected via Ethernet network.
ERSC	**E**thernet **R**oller **S**peed **C**ontrol module - Conveyor control module that is part of the *ConveyLinx* family. Each ***ERSC*** can accommodate up to 2 **MDR** conveyor **zones**
Ethernet I/P	Ethernet/IP (Ethernet Industrial Protocol) is a network communication standard capable of handling large amounts of data at speeds of 10 Mbps or 100 Mbps, and at up to 1500 bytes per packet. The specification uses an open protocol at the Application layer. It is especially popular for control applications. Ethernet/IP typically employs active star network technology. This type of network is easy to set up, operate, maintain, and expand. It allows mixing of 10 Mbps and 100 Mbps products, and is compatible with most Ethernet switches. Ethernet/IP is used with personal computers, mainframes, robots, input/output (I/O) devices and adapters, programmable logic controllers (PLCs), and other devices. The specification is supported by the Industrial Ethernet Association (IEA), ControlNet International (CI), and the Open DeviceNet Vendor Association (ODVA).
JST	This is the name of a particular connector manufacturer that produces a specific plug/socket arrangement for **MDR** connection to control cards. This name is accepted within the conveyor and **MDR** industry as a simple description of the particular socket style used on **ERSC** hardware.
Load	A separate (usually wrapped or boxed) object to be transported by the conveyor. The terms **tray, tote,** or **carton** may also be used interchangeably in this document.
MDR	**M**otorized **D**rive **R**oller or **M**otor **D**riven **R**oller - Brushless DC motor and gearbox assembly integrated into a single conveyor roller.
Modbus TCP	Application layer messaging protocol at Level 7 of the OSI Model that provides client/server communication between devices connected on different types of buses or networks. The Modbus messaging structure was developed by Modicon in 1979. Different versions of Modbus used today include Modbus RTU (based on serial communication like RS485 and RS232), Modbus ASCII and Modbus TCP, which is the Modbus RTU protocol embedded into TCIP packets. It's an open protocol, meaning the specification is available free of charge for download, and there are no licensing fees required for using Modbus or Modbus TCP/IP protocols. Modbus TCP/IP specification was developed in 1999 to combining a ubiquitous physical network (Ethernet) with a universal networking standard (TCP/IP) and a vendor-neutral data representation. Modbus TCP/IP used the Modbus instruction set and wraps TCP/IP around it.

Photo-sensor	A device, mounted near the end of the conveyor **zone** to sense the presence of a **load** on the zone
PLC	**P**rogrammable **L**ogic **C**ontroller – A wide variety of industrial computing devices that control automatic equipment
RJ-11 / RJ-12	**R**egistered **J**ack Style **11 / 12** – Standard connector / receptacle format utilizing 4 or 6 pin connections. The typical standard connection for telephones. RJ-11 utilizes 4 pins and RJ-12 utilizes 6 pins but both styles use the same physical size.
RJ-45	**R**egistered **J**ack Style **45** – Standard connector / receptacle format utilizing 8 pin connections. The typical standard for computer network cable connections
TCP/IP	**T**ransport **C**ontrol **P**rotocol / **I**nternet **P**rotocol - **IP** is the protocol which oversees the transmission of information packets from device to device on an Ethernet network. **TCP** makes sure the packets have arrived and that the message is complete. These two protocols are the basic language of the Internet and are often referred to together as **TCP/IP**.
Train Release	Conveyor control method for **zone** configured conveyor that dictates that when a **zone** is discharging, the upstream **zone's load** can move in unison with the discharging **load**.
Zone	A basic (linear or curved) cell of the conveyor consisting of a set of slave rollers driven by one or more **MDR's** and a single **photo-sensor**.
ZPA	**Z**ero **P**ressure **A**ccumulation – Term that describes the conveyor controls and mechanical scheme that will cause loads to queue on a conveyor in discrete **zones** such that loads do not touch each other

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, 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	October 2009	Initial Release
3.0	July 2010	Supersedes Version 1.0 – VALID FOR FIRMWARE 3.0 AND HIGHER - Added Ethernet I/P Functionality with examples
4.0	June 2011	Added functionality to Ethernet I/P Instances.

Table of Contents

Glossary of Terms....................3
  Symbol Conventions....................4
Important User Information....................5
Summary of Changes....................6
Table of Contents....................7
Preface....................9
  Who Should Use This Manual?....................9
  Purpose of This Manual....................9
  Not Included in This Manual....................9
Introduction to *ConveyLinx®*....................11
  ConveyLinx® Concept....................11
  ConveyLinx ERSC Operational Modes....................13
    ZPA Mode....................13
    PLC I/O Mode....................13
ConveyLinx Network Architecture....................15
  ERSC Modbus Register Structure....................15
  ERSC Ethernet I/P Register Structure....................16
External Controller with ZPA Mode....................17
  Registers for ERSC in ZPA Mode....................17
    Zone Control Description – ZPA Mode....................17
    ERSC Left and Right Status – ZPA Mode....................22
  ZPA Mode Register Data Flow....................24
    External Controller as *ConveyLinx* Bridge....................25
  Ethernet I/P Controller with ZPA Mode....................27
    Input Instance for ZPA Mode (Instance ID 5)....................28
    Output Instance for ZPA Mode (Instance ID 6)....................29
    Creating ZPA Mode Instance in RSLogix 5000....................31
    Programming Example....................33
  Ethernet I/P ZPA Mode with PLC Enable....................37
    Input Instance for ZPA Mode – PLC Enable (Instance ID 25)....................38
    Output Instance for ZPA Mode – PLC Enable (Instance ID 26)....................39
    Creating ZPA Mode – PLC Enable Instance on RSLogix 5000....................40
    Programming Example for PLC Enable Instance....................40
External Controller with PLC I/O Mode....................43
  Setting PLC I/O Mode in EasyRoll....................44
    Optional "Clear Connections" Choice....................45
  Registers for ERSC in PLC I/O Mode....................47

Sensor & Control Port Digital I/O – PLC I/O Mode 47
Left Motor Control & Status – PLC I/O Mode....48
Right Motor Control & Status – PLC I/O Mode 50
Adjacent Connected Module Data – PLC I/O Mode 52
Motor Ports as Digital I/O – PLC I/O Mode....54
Ethernet I/P Controller with PLC I/O Mode 56
Input Instance for PLC I/O Mode (Instance ID 7)....57
Output Instance for PLC I/O Mode (Instance ID 8)....58
Creating PLC I/O Mode Instance on RSLogix 5000 59
Ethernet I/P Guidelines....61
Ethernet I/P Logix5000 MSG Instruction 62
Notes: 65

Preface

Who Should Use This Manual?

This manual is intended for users who need to utilize a PLC or PC to connect to *ConveyLinx* Ethernet network to access module status and control conveyor operation.

You should have an intermediate to advanced level of understanding of PLC logic and network structures. Familiarity with Ethernet I/P and Modbus TCP protocols is also essential.

You should have a basic understanding of electrical circuitry and familiarity with relay logic, conveyor equipment, photoelectric sensors, etc. in order to follow example scenarios and sample programs included herein. If you do not, obtain the proper training before using this product.

For basic understanding of *ERSC* module hardware and simple application and installation guidelines, please refer to Insight Automation publication *ConveyLinx User's Guide* (publication *ERSC-1000*)

Purpose of This Manual

The purpose of this manual is to:

- Define *ERSC* Module's Modbus TCP data communication register architecture and their basic meanings and functions.
- Provide instructions on how to attach an instance of an *ERSC* module to an Ethernet I/P PLC
- Provide conveyor layout example along with sample PLC program code for simple zone control using Ethernet I/P

Not Included in This Manual

Because system applications vary; this manual assumes users and application engineers have properly sized their power distribution capacity per expected motor loading and expected operational duty cycle. Please refer to conveyor equipment and/or motor roller manufacturer's documentation for power supply sizing recommendations.

Introduction to *ConveyLinx®*

ConveyLinx® Concept

ConveyLinx control system as applied to conveyor control is a series of individual *ConveyLinx ERSC* modules interconnected via standard Ethernet cabling to form an integrated solution for MDR (Motorized Drive Roller) conveyor functionality. Each *ConveyLinx ERSC* module can accommodate up to 2 MDR's and 2 photo-sensors to provide control for up to 2 conveyor zones. Each *ERSC* also includes convenient connectivity ports for upstream and downstream Ethernet network cabling as well as connectivity ports for discrete I/O signals with non-networked controls for local interlock interface functions.

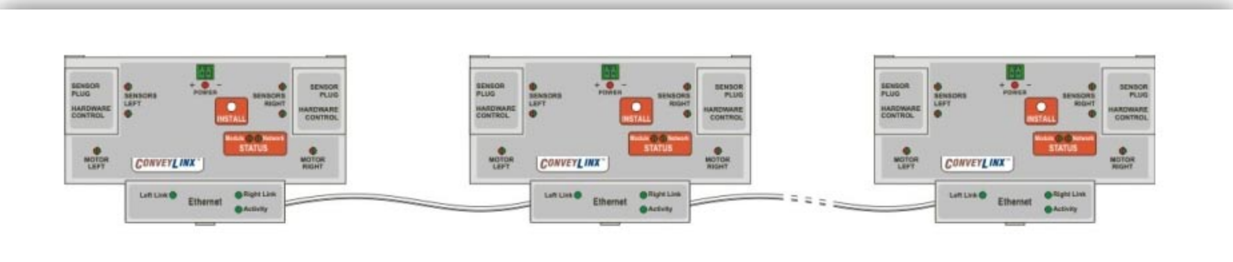


Figure 1 - *ConveyLinx®* Concept with *ERSC* Modules

ConveyLinx ERSC modules can be easily automatically configured to operate multiple zones of linear conveyor "right out of the box" with the push of a button without any special tools or PC software required. However, with the *ConveyLinx Easy Roll* software tool and a PC; each *ERSC* module's default configuration can be modified to customize functionality for specific applications.

Please refer to Insight Automation publication *ConveyLinx User's Guide* (publication *ERSC*-1000) for *ConveyLinx ERSC* module hardware details and basic function. The remaining sections of this document assumes reader if familiar with *ConveyLinx ERSC* module and *EasyRoll* software operation.

ConveyLinx ERSC Operational Modes

Each and every *ERSC* module on a conveyor system functions in ***only one*** of two possible operational modes:

- **Zero Pressure Accumulation (ZPA) Mode**
- **PLC I/O Mode**

These operational modes dictate which of the modules data registers are valid for use by a networked external controller.

ZPA Mode

ZPA Mode is the default mode of any *ERSC* that has been configured by the *Auto-Configuration Procedure*. In this mode, each *ERSC* has established logical connections to its neighboring *ERSC* modules in order to operate conveyor with Zero Pressure Accumulation (ZPA) functionality. No external controller is required for the conveyor to function and operation is as described in *ConveyLinx User's Guide* (publication *ERSC*-1000).

PLC I/O Mode

In PLC I/O Mode, the *ERSC* suspends all automatic ZPA functionality and its input, output and motor control functions are explicitly controlled by a networked external logic controller. The external controller reads from and writes to the *ERSC*'s internal data registers over the Ethernet network using Ethernet I/P or Modbus TCP protocol in order to initiate all *ERSC* functionality.

ERSC **module can only be placed in PLC I/O Mode with *EasyRoll* software tool and only after it has been through an *Auto-Configuration Procedure*.**

IMPORTANT CONCEPT TO REMEMBER: **A PLC or external controller can connect and interact with any *ERSC* regardless of whether it is in ZPA or PLC I/O Mode. PLC I/O Mode *requires* a PLC or external controller for operation. ZPA Mode *ERSC* can interface with a PLC or external controller to report local module status and provide local zone interaction.**

The following chart provides a Quick Reference for the operational modes:

ERSC Control Strategy Quick Reference		
Any Mode	**Read Status**	• **Available to any Modbus TCP or Ethernet I/P networked device regardless of Module Mode** • **Read Sensor & Control port inputs** • **Read MDR diagnostics**
ZPA Mode	**Default**	• **Out of the Box** • **No PLC required** • **Hard-wired connections available to control zone stop/release** • **Some electrical connections may require RJ-12 to terminal breakout module for signal compatibility** • **See User's Guide for connection details**
	Interface PLC	• **Ethernet I/P or Modbus TCP Networks supported** • **PLC can control over the network: zone accumulate/release, tracking data read/write, zone speed, monitor sensor & control port inputs** • **See page 17 for Modbus TCP details** • **See page 27 for Ethernet I/P details** • **See page 33 for Ethernet I/P programming example**
	Interface PLC + Port Outputs	• **All functions available with Default configuration** • **All functions available with Interface PLC** • **Using EasyRoll, allow PLC access to Control Port digital output signal** • **See** *Control Port Outputs* **on page 22 for description**
PLC I/O Mode	**Basic PLC**	• **Requires Mode change in EasyRoll** • **Suspends ALL ZPA functionality** • **PLC has complete access and control of Sensor Port Inputs and Control Port Inputs and Output** • **PLC has complete control of both MDR ports (speed, accel/decel, open/closed loop, braking method, etc.)** • **See page 44 for setting PLC I/O Mode** • **See page 47 for Modbus TCP register descriptions** • **See page 56 for Ethernet I/P instance descriptions**
	Full PLC	• **All functions with Basic PLC** • **Requires PLC to configure** • **One or both MDR ports can be changed to provide up to 3 high powered digital outputs (1A each) per MDR port for a total of 6 digital outputs available.** • **See page 54 for details on set-up**

***ConveyLinx* Network Architecture**

Each *ERSC* communicates to its adjacent modules and to any connected PC or PLC via Ethernet physical media. *ConveyLinx* modules recognize (2) TCP/IP based protocols: Modbus TCP and Ethernet I/P. Modbus TCP is the “native” protocol for communications between *ConveyLinx* modules and the *EasyRoll* PC software. When *ConveyLinx* modules are used even for basic ZPA control with no external connections to a PC or PLC, they utilize Modbus TCP for inter-module communication. Ethernet I/P is also recognized by *ConveyLinx* modules and any given *ERSC* can be attached to an Ethernet I/P capable PLC (Allen-Bradley ControlLogix or CompactLogix platforms) and be recognized as a “generic Ethernet I/P device”. Both protocols access the same internal data locations on a given *ERSC*.

The logical addressing of these internal data locations is identical for each *ERSC*. The on-board *ERSC* communication and control processes attach logical meanings to each data location and read and write data to specific locations to initiate and/or react to events. Certain data locations contain information as to how the *ERSC* is configured (MDR type, speed, direction, etc.) for its local controls. Other data locations are used for inter-module communications for conveyor operation. For example, when an upstream *ERSC* has a Load ready to discharge to its neighboring downstream *ERSC*, the upstream *ERSC* will write a specific value to a specific address in the downstream *ERSC*’s internal data memory. The downstream *ERSC*’s on board logic monitors this internal memory location and knows that a specific value means that an upstream Load is coming and to engage the proper control logic to convey the Load.

Because *ConveyLinx* utilizes an open architecture (Modbus TCP) for inter-module communications; with proper definition and expected usage of certain accessible memory locations, external control devices (PC’s and PLC’s) can easily interact with *ERSC* modules to monitor and control various points along the conveyor path.

ERSC Modbus Register Structure

Each *ERSC* utilizes Modbus register architecture for remote data access over Ethernet. Modbus TCP is a simple protocol for data exchange based upon a query/response mechanism. Each *ERSC*’s memory structure contains a fixed array of internal data locations that are constructed as Modbus *Holding Registers*. Each *ERSC* has a fixed reserve of *Holding Registers* with each capable of holding a 16-bit numerical integer value. Modbus TCP protocol provides for read/write access to any available *Holding Register*. The structure of these registers allows for individual *ERSC*’s to read from and write to specific register address locations to achieve inter-module communications. Certain registers are read from and written to by the *EasyRoll* software in order to monitor and/or change default configuration values such as MDR speed, direction, type, etc.

Modbus TCP addressing convention utilizes a "4:xxxx" notation. The "4:" in Modbus protocol designates that the address is a *Holding Register* and the xxxx is a numerical value representing the offset or index for a specific location. The "xxxx" values used in this document are to be interpreted as if they are for a Modbus PLC which means that the first register address is "4:0001" and that there is no "4:0000" register. Some PLC data structures and PC development environments utilize the "4:0000" designation and their indexes will be offset by 1. Please refer to your PLC or PC application documentation for the Modbus convention used on their platforms.

Any Modbus TCP capable PC or PLC can connect to any *ERSC* visible on its network and read and write data to its *Holding Registers* by utilizing *Read Holding Registers* and *Write Holding Registers* standard Modbus function codes. The *Read Holding Registers* and *Write Holding Registers* functions can read or write from 1 to up to 10 registers in a single transaction command.

For more information and open protocol specification, please visit www.modbus.org

ERSC Ethernet I/P Register Structure

When an *ERSC* is attached to an external Ethernet I/P controller (Logix 5000 based PLC), it is done so as a Generic Ethernet I/O device. Part of this procedure in the PLC is to instruct the Generic device as to which data configuration or instance of Ethernet I/P the Generic device is to use to report and respond to data to and from the PLC. The *ERSC* recognizes 2 specific instance types; one for each of the *ERSC* operational modes.

These instances essentially group the appropriate Modbus registers into contiguous Input and Output array images that fit into the Allen-Bradley Logix 5000 controller tags.

This manual assumes any reader interested in utilizing Ethernet I/P interface to ConveyLinx is experienced with Allen-Bradley Logix 5000 programming software and is familiar with User Defined Data Type structures and attaching Ethernet I/P Generic I/O device instances to a PLC program project.

External Controller with ZPA Mode

When an *ERSC* is in its default ZPA mode, an external networked PLC or PC controller can connect to the *ERSC* and perform the following:

- Instruct either or both the upstream and downstream zone to accumulate the next load that arrives
- Receive indication that a new load has arrived at either zone
- Receive indication that a load has departed from either zone
- Read tracking data associated with load at accumulated zone
- Update tracking data associated with load at accumulated zone
- Instruct accumulated zone to release load and accumulate on next load arrival
- Change the MDR speed for either zone
- Remove accumulation control and return zone to normal operation
- Read fault and error status of either zone or motor

Registers for ERSC in ZPA Mode

When an *ERSC* is in ZPA Mode, its primary task is to operate its local conveyor zones and respond to its immediate upstream and downstream conditions. External controller interaction with an *ERSC* in ZPA mode is intended to be for decision point monitoring and general status data gathering.

In general, when utilizing ZPA Mode registers; "upstream" and "downstream" registers are logically determined by conveyor flow after the system has been Auto-Configured and will not necessarily be associated with the *ERSC*'s physical "left" or "right" side's connections. For motor specific items, register's description will explicitly indicate left or right.

Zone Control Description – ZPA Mode

Upstream and Downstream zones work exactly the same for zone control, only the register address are different depending on which zone (or both) that need to be controlled.

To accumulate a zone, have the external controller write a 1 to the *Accumulate Enable* register and then monitor the *Arrival Count* value. When the *Arrival Count* value is changed, then there is a new load arrival that is stopped and accumulated in the zone. The external controls can then read the *Tracking Data* registers, process accordingly, and write new data to these registers if desired. To release the load from the zone, the external controller needs to change the value in the *Accumulate on Next* register. The *ERSC* recognizes the change of data in the *Accumulate on Next* register and will automatically release the load if the next downstream zone is ready to accept. If the next downstream zone is not ready to accept the load, the *ERSC* remembers that it needs to release the load and will automatically do so when downstream conditions are ready without any further action by the external controls. If desired, the external controls can monitor the *Departure Count* register to determine that a load has successfully left the accumulated zone.

If the external control needs to understand the ZPA status of the *ERSC*'s local zones; the *Zone Status* registers provide a numerical value that indicates the possible states of a ZPA controlled conveyor zone. Similarly if the external controls needs to understand the immediate upstream and/or downstream zones adjacent to the connected *ERSC*; this data is provided by the *Adjacent Upstream Status* and *Adjacent Downstream Status* registers.

An important requirement placed on the external controls is to maintain its own internal data values as to the previous values for *Arrival Count* and *Departure Count* to use in its internal logic to determine when the *ERSC* changes these values to indicate a new arrival or new departure

For *ERSC* modules that are auto-configured as <u>single zone</u>, regardless of whether the left or right side is physically used as the single zone; external controller <u>must</u> use the "<u>Upstream"</u> control registers to interface with the single zone.

Upstream Zone Control – ZPA Mode

Item Description	*4:xxxx Register*	*Item Usage*
Accumulate Enable	4:0104	1 = Sets zone to accumulate mode 0 = Release if accumulated and cancel accumulate
Accumulate on Next	4:0105	For "one-time" use: • Always set to 0 and use Accumulate Enable register *4:0104* only For "Release and Accumulate on Next" function: • Always set Accumulate/Release *4:0104* = 1 • Increment or decrement value in *4:0105*. Any change of data in *4:105* initiates release and accumulate on next
Arrival Count	4:0106	Integer Value • Increments by 1 each time a Load arrives • Value rolls over from 65,535 back to 0
Departure Count	4:107	Integer Value • Increments by 1 each time a Load departs • Value rolls over from 65,535 back to 0
Zone Status[1]	4:0116	Integer Value of Low Byte 0xXX01 = Zone sensor clear and motor stopped 0xXX02 = Zone sensor clear, motor running, discharging to downstream zone 0xXX03 = Zone sensor clear, motor running, accepting load from upstream zone 0xXX04 = Zone sensor blocked, motor running, discharging to downstream zone 0xXX05 = Zone sensor blocked and motor stopped
Tracking Data –Word #1	4:0119	16 Bit Value tracked with Load and passed from zone to zone
Tracking Data – Word #2	4:0120	16 Bit Value tracked with Load and passed from zone to zone
Adjacent Upstream Status[1]	4:0134	Integer Value of Low Byte 0xXX01 = Zone sensor clear and motor stopped 0xXX02 = Zone sensor clear, motor running, discharging to downstream zone 0xXX03 = Zone sensor clear, motor running, accepting load from upstream zone 0xXX04 = Zone sensor blocked, motor running, discharging to downstream zone 0xXX05 = Zone sensor blocked and motor stopped

Downstream Zone Control – ZPA Mode

Item Description	*4:xxxx Register*	*Item Usage*
Accumulate Enable	4:0184	1 = Sets zone to accumulate mode 0 = Release if accumulated and cancel accumulate
Accumulate on Next	4:0185	For "one-time" use: • Always set to 0 and use Accumulate/Release register *4:0184* only For "Release and Accumulate on Next" function: • Always set Accumulate/Release *4:0184* = 1 • Increment or decrement value in *4:0185*. Any change of data in *4:185* initiates release and accumulate on next
Arrival Count	4:0186	Integer Value • Increments by 1 each time a Load arrives • Value rolls over from 65,535 back to 0
Departure Count	4:0187	Integer Value • Increments by 1 each time a Load departs • Value rolls over from 65,535 back to 0
Zone Status[1]	4:0196	Integer Value of Low Byte 0xXX01 = Zone sensor clear and motor stopped 0xXX02 = Zone sensor clear, motor running, discharging to downstream zone 0xXX03 = Zone sensor clear, motor running, accepting load from upstream zone 0xXX04 = Zone sensor blocked, motor running, discharging to downstream zone 0xXX05 = Zone sensor blocked and motor stopped
Tracking Data –Word #1	4:0199	16 Bit Value tracked with Load and passed from zone to zone
Tracking Data – Word #2	4:0200	16 Bit Value tracked with Load and passed from zone to zone
Adjacent Downstream Status[1]	4:0232	Integer Value of Low Byte 0xXX01 = Zone sensor clear and motor stopped 0xXX02 = Zone sensor clear, motor running, discharging to downstream zone 0xXX03 = Zone sensor clear, motor running, accepting load from upstream zone 0xXX04 = Zone sensor blocked, motor running, discharging to downstream zone 0xXX05 = Zone sensor blocked and motor stopped

[1] Zone Status Values and External Visibility: The values 0xXX01 thru 0xXX05 are shown because these are the possible logical values used for inter-module communication. External networked devices (PLC or PC) utilizing either Modbus TCP or Ethernet I/P to monitor these registers may not actually see each of these values change in sequence as a load is conveyed from zone to zone, even though the inter-module communications and ZPA is functioning normally.

IMPORTANT NOTE: Status register values utilize both the HIGH BYTE and the LOW BYTE of the 16-Bit integer value. The HIGH BYTE is reserved for future use and MAY CONTAIN DATA. PLC/PC programmers MUST MASK THE HIGH BYTE from these registers before use in program logic.

For PLC/PC programming purposes, you can only depend on seeing values 0xXX01 and 0xXX05 in program logic for determining zone status. The values 0xXX02, 0xXX03, and 0xXX04 may not always be visible to PLC/PC from inter-module communication depending upon speed of the conveyor, length of the zone, and/or location of the zone sensors.

ERSC Left and Right Status – ZPA Mode

These registers are primarily status data only supplied to external controller for alarming and/or operator interface. The only register value that is configured to be written to is the *Clear Motor Error* register which is used to remotely reset the *ERSC* for certain specific motor errors that ordinarily requires a physical cycle or power on the *ERSC* to reset. The *Motor Error* bit shown (bit 3 in both 4:0088 and 4:0089) will be true if any of the associated bits 7 thru 15 are also true.

Item Description	*4:xxxx Register*	*Item Usage*
Module Status #1	4:0088	Bitwise Value - Read only bit 0 = Reserved bit 1 = Reserved bit 2 = Reserved bit 3 = Left Motor Error bit 4 = Ethernet Connections NOT OK bit 5 = Upstream Jam Error bit 6 = Left Sensor Error bit 7 = Low Voltage Error – Module Power Supply less than 18V bit 8 = Left Motor Over-heated – Calculated temperature over 120°C bit 9 = Left Motor Over-current – Over limit for selected MDR bit 10 = Left Motor Short Circuit bit 11 = Left Motor Not Connected bit 12 = Left Motor Overload – MDR slower than 10% of selected speed bit 13 = Left Motor Stalled bit 14 = Left Motor Hall Sensor Error bit 15 = Left Motor Not Used
Module Status #2	4:0089	Bitwise Value - Read only bit 0 = Reserved bit 1 = Reserved bit 2 = Reserved bit 3 = Right Motor Error bit 4 = Reserved bit 5 = Downstream Jam Error bit 6 = Right Sensor Error bit 7 = Low Voltage Error - Module Power Supply less than 18V bit 8 = Right Motor Over-heated - Calculated temperature over 120°C bit 9 = Right Motor Over-current - Over limit for selected MDR bit 10 = Right Motor Short Circuit bit 11 = Right Motor Not Connected bit 12 = Right Motor Overload – MDR slower than 10% of selected speed bit 13 = Right Motor Stalled bit 14 = Right Motor Hall Sensor Error bit 15 = Right Motor Not Used
Left Motor Speed Reference	4:0040	Value in % PWM Range: 0 to 1000 *Example: 400 = 40%*
Right Motor Speed Reference	4:0064	Value in % PWM Range: 0 to 1000 *Example: 400 = 40%*
Motor Fault Reset	4:0022	Logical 0 or 1 0 = Stop Reset 1 = Send Reset
Control Port Outputs[1]	4:0063	Bitwise Value – "1" Energizes Output bit 1 = Left Control Port bit 3 = Right Control Port

Motor Short Circuit and Motor Hall Sensor Error are classified as "fatal" errors that require either a cycle of power on the *ERSC* or an explicit Motor Fault Reset command from external controller.

External controller must continuously write "1" to the *Motor Fault Reset* register for at least 500 msec for reset to occur.

[1] By default, Register 4:0063 does not have access to control port outputs. Control port outputs are available at Register 4:0063 only if enabled for a particular module via *EasyRoll* software tool. To enable PLC control of Control Port output(s), use the Advanced Dialog screen as shown below in Figure 2.

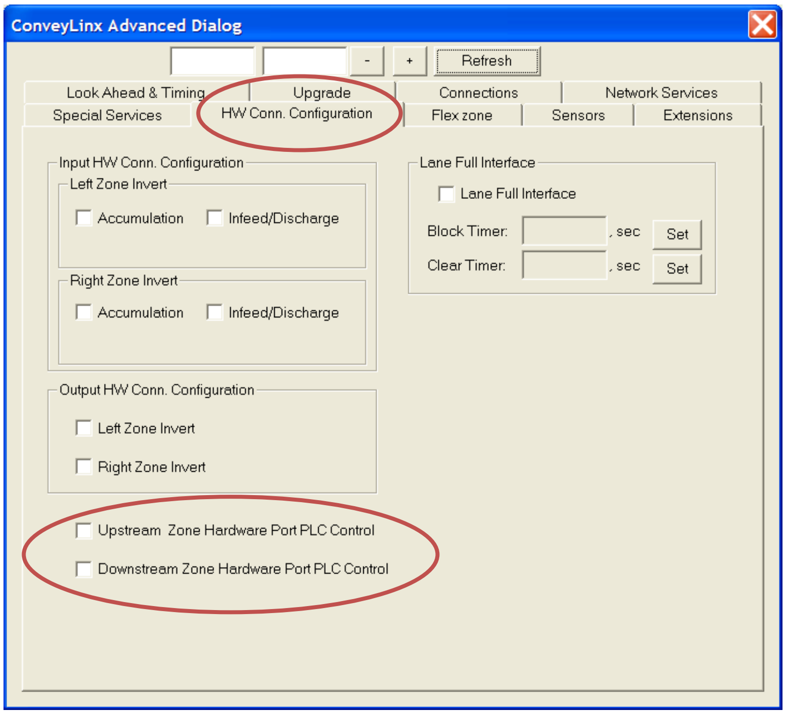


Figure 2 - *EasyRoll* Advanced Dialog showing PLC control of Control Port Selection

Please note that this enable/disable is based upon upstream or downstream configuration of the *ERSC* determined during Auto-Configuration and NOT by "left" or "right" Control Port designation.

ZPA Mode Register Data Flow

Because each *ERSC* has read/write access to its adjacent *ERSCs* and by virtue of the Auto-Configuration procedure that defines conveyor flow ***connections***; each *ERSC* knows the I.P. address of its adjacent upstream and downstream *ERSC*. In the direction of flow, each *ERSC* writes its zone's status values to its next upstream and next downstream *ERSC*. The following figure graphically illustrates a simple configuration of three *ERSC*'s and the logical ***connections*** made by each *ERSC* to communicate with its adjacent modules and the local register used.

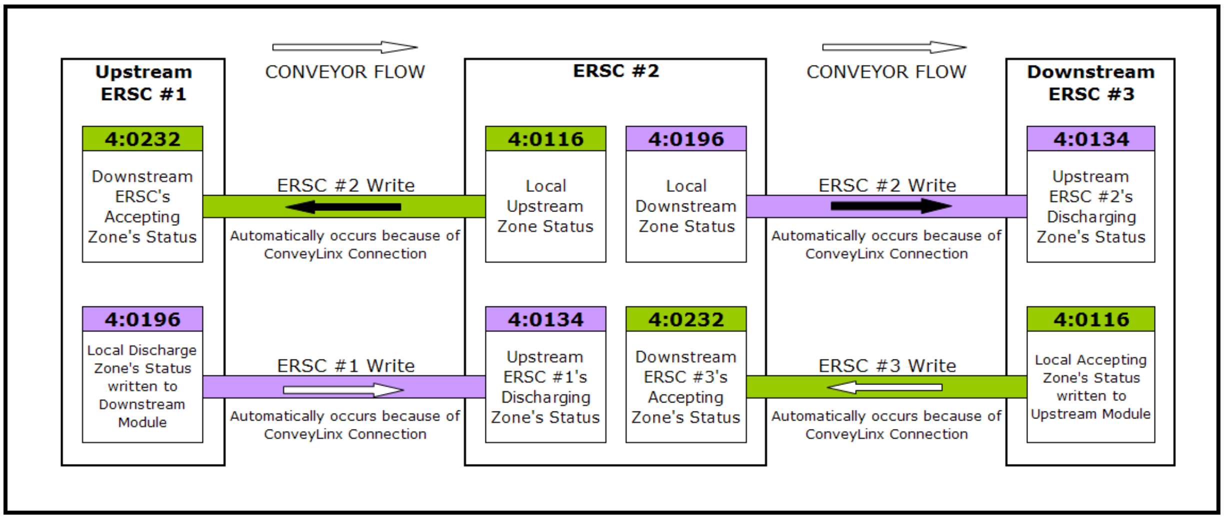


Figure 3 - *ERSC* Register Mapping for Normal Flow

Please note that the following examples in this section assume the High Byte of all zone status registers have been masked such that only the Low Byte values are used.

From the preceding charts, we can see that all of the *Holding Registers* in Figure 3 contain integer values indicating a zone status. In essence, the ZPA control process in each *ERSC* writes its zone's status to its adjacent neighbors and monitors its local registers that are written to by its adjacent *ERSC* neighbors. The ZPA process acts upon what is written to it and updates its local status accordingly. For example, the following sequence would be typical:

- *ERSC* #1 continually reads register 4:0232 to know the status of *ERSC* #2's local upstream zone
- If a load arrives at *ERSC* #1 discharge zone and the value in its register 4:0232 is equal to 4 or 5, then it will stop the load at its downstream discharge zone
- *ERSC* #1 continues to read register 4:0232. Once the value in its register 4:0232 has changed to a value of 1 or 2; the load in its discharge zone can be conveyed to *ERSC*

#2. To initiate this, *ERSC* #1 writes its status value of 4 (located in register 4:0196) to register 4:0134 on *ERSC* #2.

- When *ERSC* #2 control logic sees its register 4:0134 change to a value of 4, it knows that a load is being discharged onto its upstream zone and thus will energize its upstream zone MDR to run and accept the Load from *ERSC* #1.

There would be a similar sequence for *ERSC* #2 to convey a Load to *ERSC* #3.

External Controller as *ConveyLinx* Bridge

An external controller or PLC can also write valid zone status data to *ERSC*s that are adjacent to a PLC controlled non-*ConveyLinx* conveyor or machine. The PLC simply mimics the *ERSC* interface to control flow from upstream *ConveyLinx* controlled conveyor and to initiate Load transfer to downstream *ConveyLinx* controlled conveyor.

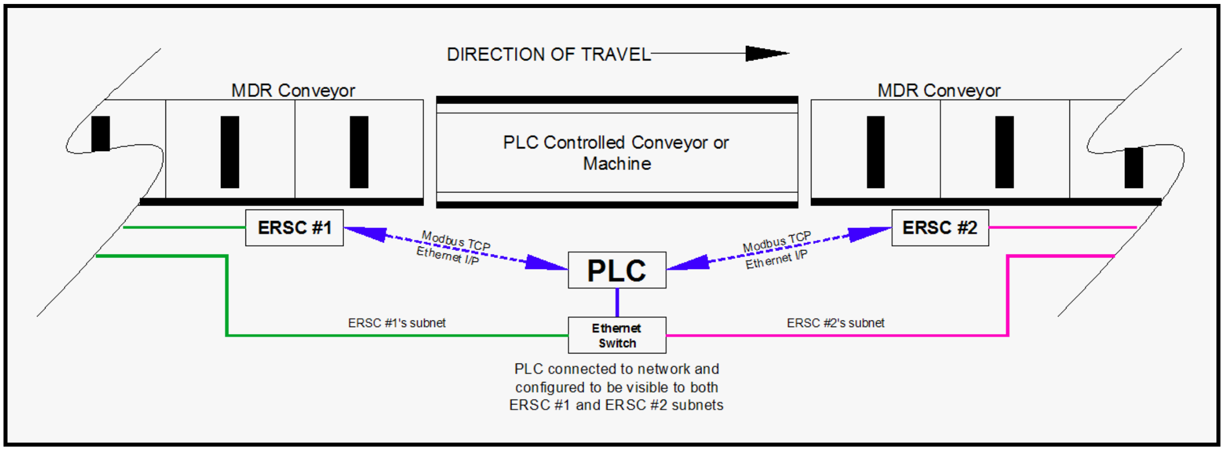


Figure 4 - PLC *ConveyLinx* Bridge

Figure 4 shows a simple example where a PLC would act as a bridge between two separate *ConveyLinx* subnets. In this case, *ERSC* #1 is at the end of its particular subnet, so when it was configured; it knows that it does not have a logical downstream connection. Similarly, *ERSC* #2 is at the beginning of its particular subnet, so when it was configured it knows that it does not have a logical upstream connection. In order for Loads to flow from *ERSC* #1 eventually to *ERSC* #2; the PLC must act as a bridge and "mimic" the writing of register data to *ERSC* #1 and *ERSC* #2 as if it was an *ERSC* module. This is illustrated in Figure 5.

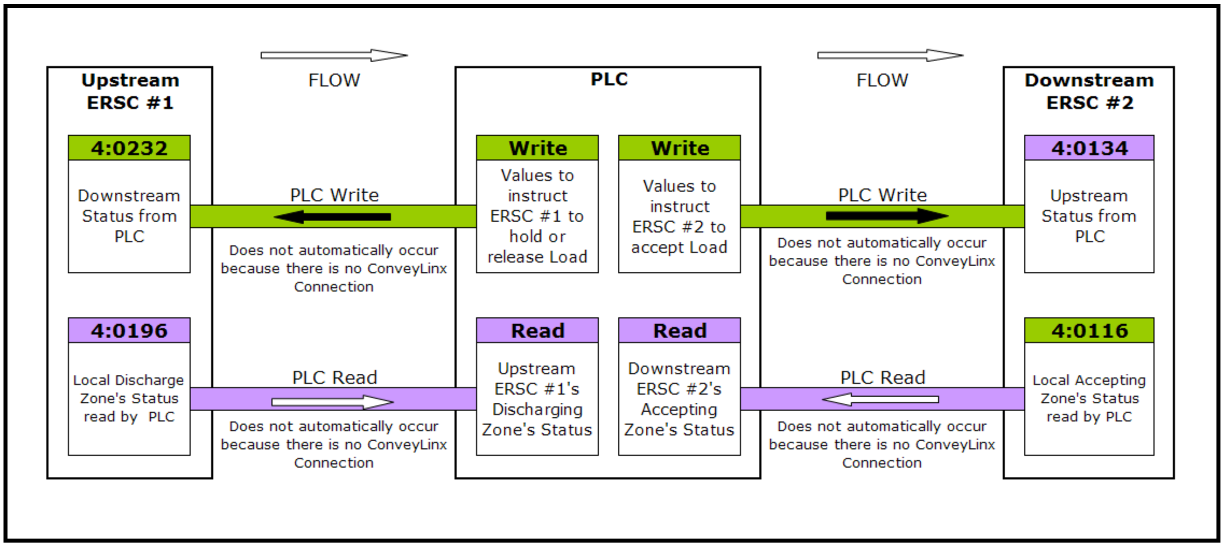


Figure 5 – Data flow chart for PLC *ConveyLinx* Bridge Example

For example, if the PLC wants to make sure that any Load stops at the discharge of *ERSC* #1, it will write a value of 5 into *ERSC* #1 register 4:0232. A value of 5 means "zone sensor blocked and motor stopped" and will be interpreted by *ERSC* #1 as a reason to stop and hold any Load that arrives at its discharge zone.

The PLC can monitor the status of *ERSC* #1 discharge zone by reading *ERSC* #1 register 4:0196. If the PLC reads a value of 3; it knows that a Load is being conveyed into *ERSC* #1's discharge zone. If this value is 5, the PLC knows a Load is present and stopped in *ERSC* #1's discharge zone. When the PLC is ready to accept the Load, it simply writes a value of 3 to *ERSC* #1 register 4:0232.

For the discharge from the PLC controlled conveyor or machine, the PLC can monitor the status of the upstream zone of *ERSC* #2 by reading *ERSC* #2 register 4:0116. For example, if a value of 5 is read, the PLC will know that the *ERSC* #2 accepting zone is occupied and is not ready to accept a Load. If the value read is a 1 or 2, then the PLC knows it can discharge a Load when ready. In order to discharge a Load, the PLC will need to write a value of 4 to *ERSC* #2 register 4:0134 to tell *ERSC* #2 to run its upstream accepting zone.

Ethernet I/P Controller with ZPA Mode

This section assumes reader is experienced with Allen-Bradley Logix 5000 programming software and is familiar with User Defined Data Type structures and attaching Ethernet I/P Generic I/O device instances to a PLC program project.

The instance created in the Logix 5000 tree for an Ethernet I/P connection to an *ERSC* in ZPA mode consists of an Input array of 20 integers and an Output array of 20 registers. The *ERSC* automatically maps the Modbus register associated with the specific data item into one of the array positions. The data values, bit values, and usage are identical as to the Modbus descriptions.

From this point forward, it is assumed the reader is familiar with Allen-Bradley Logix platform addressing notation:

[*ModuleName*]:O.Data[*Index*].*Bit*

[*ModuleName*]:I.Data[*Index*].*Bit*

Where:

- ***ModuleName* is the instance of the device when created**
- **"O.Data" indicates the output image of the device**
- **"I.Data" indicates the input image of the device**
- **"[*Index*].*Bit*" indicates the word and bit within the image. If the bit notation is absent the notation refers to the entire word data type**

Input Instance for ZPA Mode (Instance ID 5)

Item Description	*Logix Tag Address*	*Modbus Register Address*
Zone Status Upstream	[*ModuleName*]:I.Data[0]	4:0116
Zone Status Downstream	[*ModuleName*]:I.Data[1]	4:0196
Arrival Count Upstream	[*ModuleName*]:I.Data[2]	4:0106
Departure Count Upstream	[*ModuleName*]:I.Data[3]	4:0107
Arrival Count Downstream	[*ModuleName*]:I.Data[4]	4:0186
Departure Count Downstream	[*ModuleName*]:I.Data[5]	4:0187
Module Status #1	[*ModuleName*]:I.Data[6]	4:0088
Module Status #2	[*ModuleName*]:I.Data[7]	4:0089
Tracking High Word Upstream	[*ModuleName*]:I.Data[8]	4:0119
Tracking Low Word Upstream	[*ModuleName*]:I.Data[9]	4:0120
Tracking High Word Downstream	[*ModuleName*]:I.Data[10]	4:0199
Tracking Low Word Downstream	[*ModuleName*]:I.Data[11]	4:0200
Release Upstream	[*ModuleName*]:I.Data[12]	4:0105
Release Downstream	[*ModuleName*]:I.Data[13]	4:0185
Reserved	[*ModuleName*]:I.Data[14]	Reserved
Reserved	[*ModuleName*]:I.Data[15]	Reserved
Reserved	[*ModuleName*]:I.Data[16]	Reserved
Reserved	[*ModuleName*]:I.Data[17]	Reserved
Sensor & Control Port Inputs	[*ModuleName*]:I.Data[18]	4:0035
Reserved	[*ModuleName*]:I.Data[19]	Reserved

Output Instance for ZPA Mode (Instance ID 6)

Item Description	*Logix Tag Address*	*Modbus Register Address*
Tracking High Word Upstream[1]	[*ModuleName*]:O.Data[0]	4:0132
Tracking Low Word Upstream[1]	[*ModuleName*]:O.Data[1]	4:0133
Tracking High Word Downstream[1]	[*ModuleName*]:O.Data[2]	4:0212
Tracking Low Word Downstream[1]	[*ModuleName*]:O.Data[3]	4:0213
Accumulate Enable Upstream	[*ModuleName*]:O.Data[4]	4:0104
Accumulate Enable Downstream	[*ModuleName*]:O.Data[5]	4:0184
Speed Upstream[2]	[*ModuleName*]:O.Data[6]	4:0040
Speed Downstream[2]	[*ModuleName*]:O.Data[7]	4:0064
Release Upstream	[*ModuleName*]:O.Data[8]	4:0105
Release Downstream	[*ModuleName*]:O.Data[9]	4:0185
Upstream Module Status[3]	[*ModuleName*]:O.Data[10]	4:0134
Downstream Module Status[3]	[*ModuleName*]:O.Data[11]	4:0232
Reserved	[*ModuleName*]:O.Data[12]	Reserved
Reserved	[*ModuleName*]:O.Data[13]	Reserved
Reserved	[*ModuleName*]:O.Data[14]	Reserved
Reserved	[*ModuleName*]:O.Data[15]	Reserved
Clear Motor Error	[*ModuleName*]:O.Data[16]	4:0022
Control Port Outputs[4]	[*ModuleName*]:O.Data[17]	4:0063
Reserved	[*ModuleName*]:O.Data[18]	Reserved
Reserved	[*ModuleName*]:O.Data[19]	Reserved

[1] Tracking Registers Functionality in Ethernet I/P: Because the *ERSC* is connected as I/O, the PLC inherently is always trying to update the Output image on (at least) RPI intervals. In order to keep the PLC from inadvertently overwriting the "real" tracking data registers; the *ERSC*'s Instance ID 6 implementation utilizes the holding register locations shown and automatically updates the "real" tracking registers with this new data only upon release of the load from the zone. Included in this automatic functionality are two special reserved values that can be used for convenience:

- Set both tracking registers shown to 0: This will instruct the *ERSC* to not modify the existing "real" tracking data and allow it to continue downstream "as-is" when the load is released.
- Set both tracking registers shown to 0xFFFF: This will instruct the *ERSC* to clear the "real" tracking data and when the load is released, the "real" tracking data will be "0" in both registers.

[2] **Speed Control for ZPA Mode in Ethernet I/P:** Leaving these registers at "0" will instruct the *ERSC* to use its configured speed. Any non-zero value will instruct the *ERSC* to use this non zero value as the speed reference. The speed will stay at this reference until this register is changed to a new non zero value or set to "0". When this register is set to "0", the *ERSC* will use its configured speed as reference.

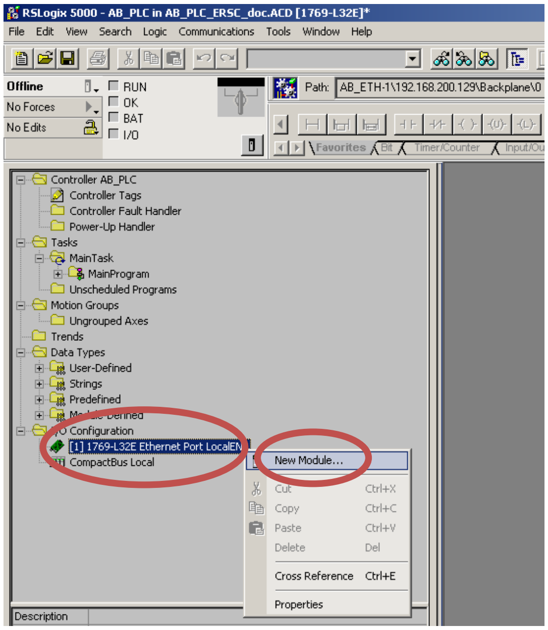
[3] **Affecting Adjacent *ERSC* Communication in Ethernet I/P:** Referring to Figure 3 on page 24, these registers are the ones populated by adjacent *ERSC*'s which indicate the adjacent zone's status. By placing non-zero values in these registers with the PLC, the *ERSC* is instructed to ignore the data written by these adjacent modules and act upon the data written by the PLC. When these registers are set to "0", the *ERSC* will act upon the data as written by the adjacent *ERSC*'s.

[4] **Using *ERSC*'s Control Port:** By default; the *ERSC*'s control port outputs provide interlocking functions between modules as described in the *ConveyLinx User's Guide.* Access by external controls to logically manipulate the on/off state of the control port output signals is not allowed. The *EasyRoll* software tool provides the ability to configure one or more *ERSC* modules so that the default interlocking functionality is suspended and external controls can have access to logically manipulate the on/off state of one or both of the control port outputs. Once this configuration option is enabled in *EasyRoll*, the PLC can access a given module's control port output via this Logix tag address.

Please note that this enable/disable in *EasyRoll* is based upon upstream or downstream configuration of the *ERSC* determined during Auto-Configuration and NOT by "left" or "right" Control Port designation.

Creating ZPA Mode Instance in RSLogix 5000

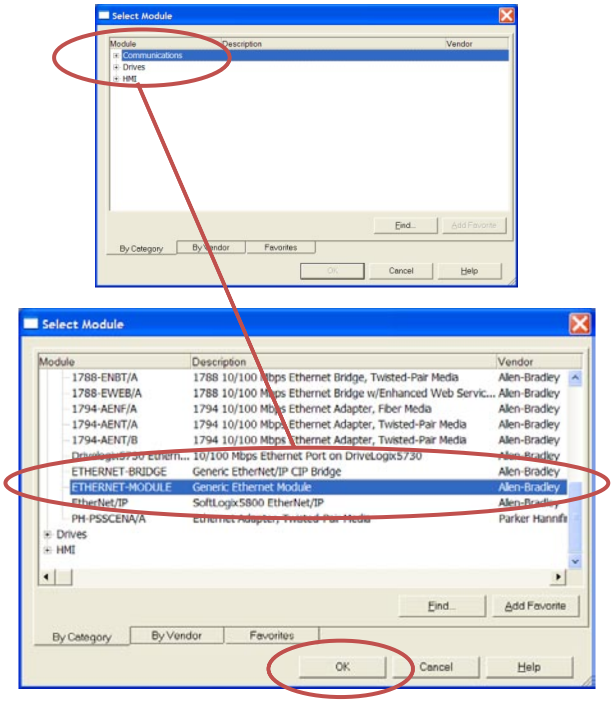
This section will provide the set-by-step procedure for creating an instance of an *ERSC* into the I/O configuration for an Allen-Bradley CompactLogix processor in RSLogix 5000 software.



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

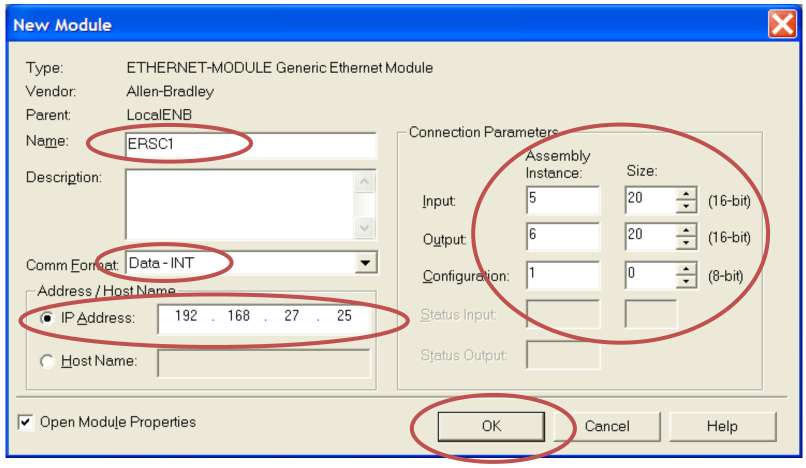
From the Select Module pop-up window, expand the Communications tree and select "Generic Ethernet Module" and click OK

Step #3

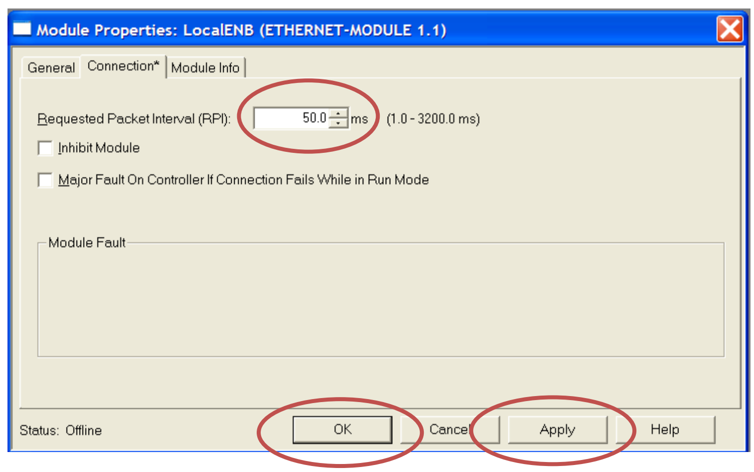
Fill in the Name field. This will be the *ModuleName* that will appear in your program Tag Database for any addressing.

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

Fill in the Connection Parameters as shown.



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



Step #4

Set RPI to a value no lower than 50ms

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

Programming Example

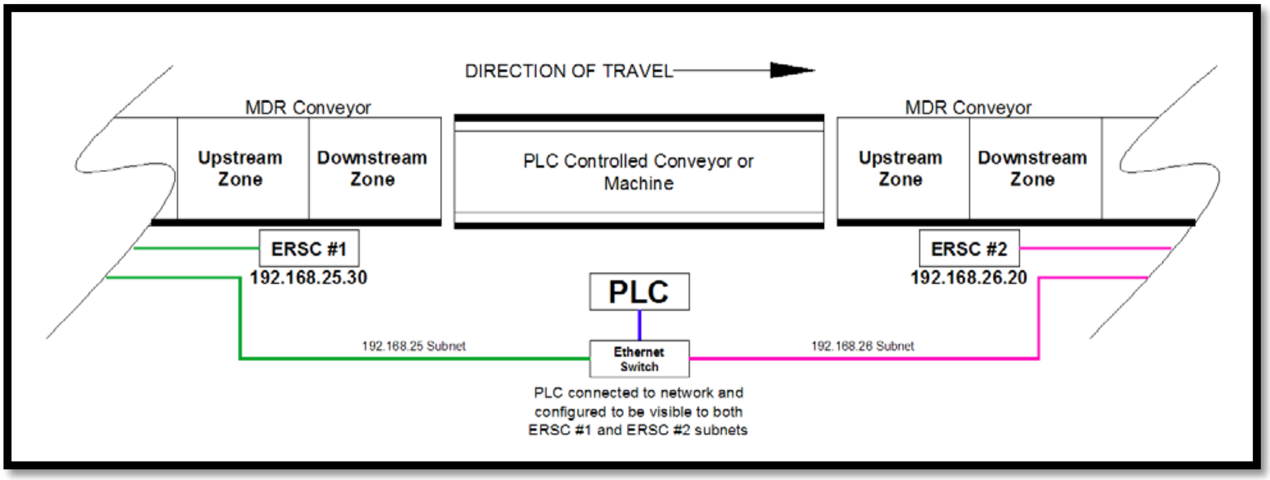


Figure 6 - Simple Interface Programming Example

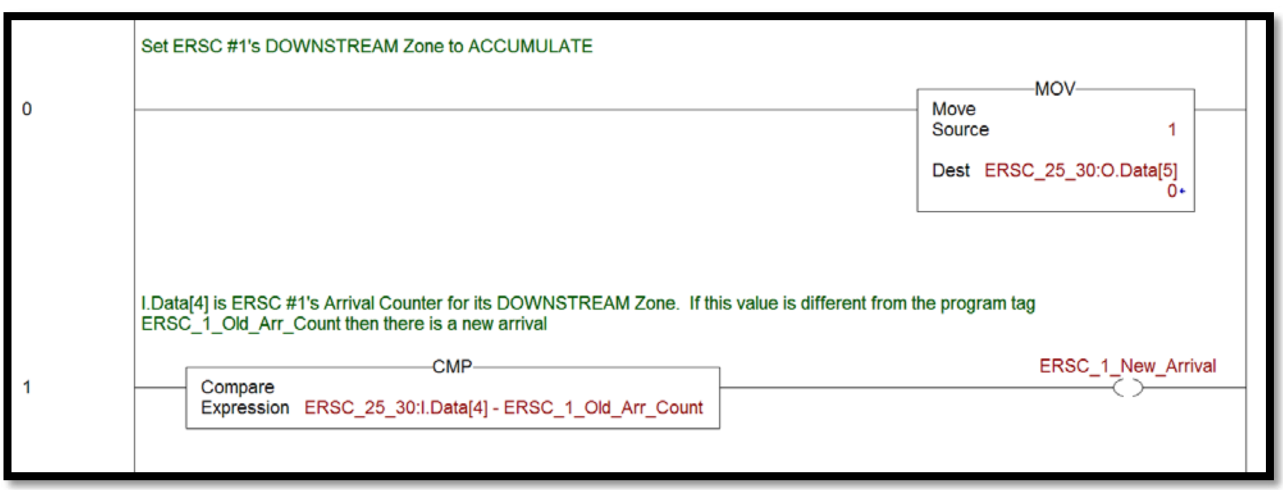
Figure 6 depicts a simple example for a sample program. *ERSC* #1 controls MDR conveyor that needs to stop and accumulate any load that arrives at its downstream zone and wait for the PLC to indicate that it is clear to enter the machine conveyor. *ERSC* #2 controls MDR conveyor that needs to have its upstream zone started to accept a load from the PLC controlled machine. The PLC also needs to know if *ERSC* #2's upstream zone is clear before attempting to discharge a load from the machine.

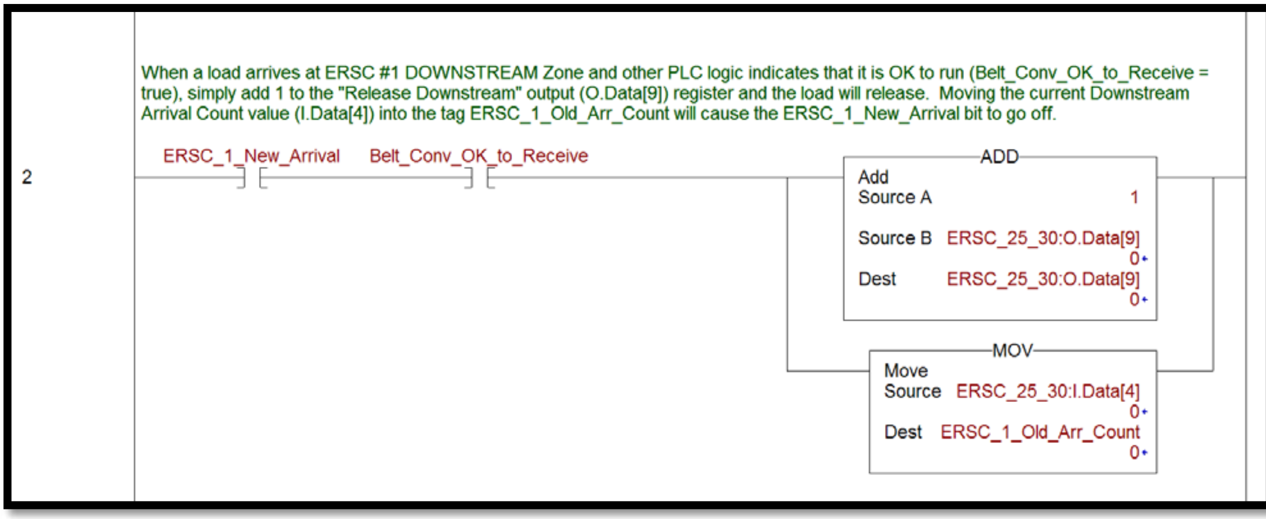
In a sample program we have created an instance each for *ERSC* #1 and #2 with the following identifiers:

ERSC #1's *[ModuleName]* = *ERSC*_25_30

ERSC #2's *[ModuleName]* = *ERSC*_26_20

Sample Logic for ERSC #1





Rung 0 logic places *ERSC* #1's downstream zone in accumulation mode. As long as this register has a value = 1, the zone will accumulate.

Rung 1 logic requires a user defined tag to retain the previous or "old" arrival count value. When the current arrival count value is different from the previous value, a new arrival is indicated.

Rung 2 logic examines if there is both a new arrival and the conditions are OK to continue to convey the load; the release downstream register is incremented by 1 to instruct the *ERSC* to "release and accumulate on next". The current arrival count is then moved into the user defined tag for previous arrival count so that they are equal. This will cause the CMP instruction in Rung 1 to be false and the coil *ERSC*_1_New_Arrival will be false. The CMP instruction will not be true again until a new load arrives and the *ERSC* increments its downstream arrival count register.

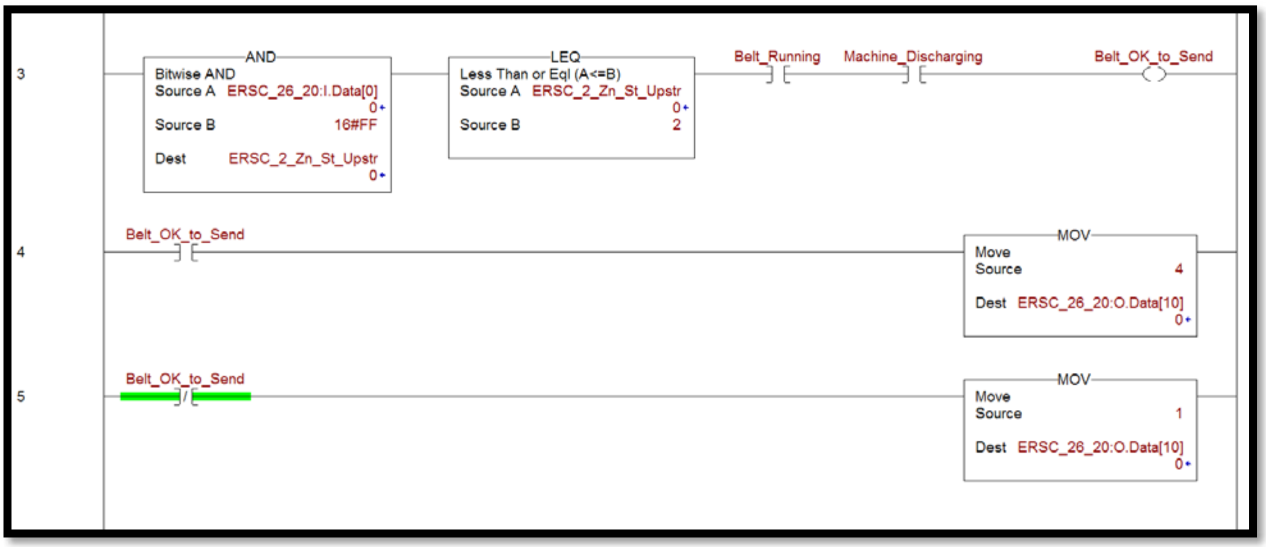
To make sure *ERSC* #1 Downstream zone accumulates upon power up, use *EasyRoll* configuration tool software to set the zone to "Accumulate". When set from *EasyRoll*, this will be retained in the flash memory of the *ERSC* so that the zone will initially accumulate if a load happens to be in the zone at the time of power up.

Be sure to click the "Accumulate" switch on the main *EasyRoll* screen to cause the zone to accumulate upon power up

When a zone is commanded to accumulate via Ethernet I/P with a PLC, the "Accumulate" switch icon on *EasyRoll* main screen will visibly indicate that the zone is accumulated. This visible icon will look the same as if it was clicked ON from *EasyRoll*.

Keep in mind that the PLC command to accumulate over Ethernet I/P <u>IS NOT retained upon power loss to the *ERSC*</u>. Only if the "Accumulate" switch is toggled "ON" in *EasyRoll* is the accumulate condition for the zone retained in flash memory for use upon power-up.

Sample Logic for ERSC #2



To control the upstream zone for *ERSC* #2, we simply manipulate the Upstream Status register O.Data[10] (4:0134) to mimic an upstream *ERSC* condition to tell *ERSC* #2 that a load is coming.

In rung 3, note first that the Low Byte of *ERSC* #2's local upstream zone status input data has been masked to an internal tag for logic use. The logic then examines this internal tag to make sure that it is either "clear and stopped" (I.Data[0] = 1) or "clear and running load to downstream" (I.Data[0] = 2). Either of these conditions should allow a new load to enter *ERSC* #2's upstream zone. Any other states of *ERSC* #2's upstream zone should indicate to the PLC to not send a new load to *ERSC* #2. Also in Rung 3 are user defined tags for belt running and machine discharging a load.

In Rung 4, if "Belt_OK_to_Send" is true, then a value of 4 is written to O.Data[10] (register 4:0134). This tells *ERSC* #2 to run its upstream zone because a load is coming from an upstream source.

In Rung 5, if "Belt_OK_to_Send" is false, then a value of 1 is written to O.Data[10] (register 4:0134). This tells *ERSC* #2 that upstream source is "clear and stopped", indicating that it does not have to run its upstream zone to accept a load. Note that *ERSC* #2's upstream zone will continue to function normally to convey loads from its upstream to downstream zone regardless of the value written to O.Data[10]. For example, if *ERSC* #2's upstream and downstream zones happen to be occupied, setting O.Data[10] = 4 will not cause the upstream zone to ignore its immediate status or downstream conditions and run on its own.

Examples shown are to illustrate the concepts without any attempt to utilize RSLogix built-in organization tools. Experienced RSLogix 5000 programmers can utilize tag aliases, data type structures, etc. to streamline the programming process particularly for more complex systems.

Ethernet I/P ZPA Mode with PLC Enable

For control system applications where the PLC needs to take specific action to recover from a loss of communications due to an *ERSC* module in ZPA mode that has had its power cycled off and on; there is an additional set of instances implemented that provides 2 new registers that allows the PLC to manipulate the function of the *ERSC* module for recovery.

When an *ERSC* has lost power and then is powered back up, due to perhaps a system E-stop that disconnects control power, some of the *ERSC*'s working register values are reset to 0. Among these are the arrival counters, departure counters, and the accumulate commands for each configured zone(s) on the *ERSC*. Upon power cycle to the *ERSC*, if the PLC can establish its full Ethernet I/P connection prior to the ZPA task becoming fully functional, the preceding PLC programming example would still function as expected. However, because PLC Ethernet I/P connection time is variable and not fixed; a robust control system design cannot count on the PLC establishing Ethernet I/P connection prior to the *ERSC*'s ZPA task commanding the module as if no PLC was connected. A consequence of this in the previous programming example is that if a load happens to be accumulated in a PLC controlled zone at the time of power loss, upon powering back up, the load can release without the PLC commanding it to do so. This release could be caused by the PLC logic detecting a change in arrival count and thus incrementing the release or it could be caused by the fact that the accumulate command is cleared in the *ERSC* due to power cycle and because the PLC has not established communications to set the accumulate command bit, the *ERSC* releases the zone because there is no command present to accumulate.

Ethernet I/P Input Instance 25 and Output Instance 26 are used for applications where the *ERSC* remains in a "hold" state until the PLC has established communications and writes the current input value for Module Reset Counter into the Module reset Counter output register to signal the *ERSC* to begin its ZPA functionality. Otherwise, the register mapping is identical to Input Instance 5 and Output Instance 6

Input Instance for ZPA Mode – PLC Enable (Instance ID 25)

Item Description	*Logix Tag Address*	*Modbus Register Address*
Zone Status Upstream	[*ModuleName*]:I.Data[0]	4:0116
Zone Status Downstream	[*ModuleName*]:I.Data[1]	4:0196
Arrival Count Upstream	[*ModuleName*]:I.Data[2]	4:0106
Departure Count Upstream	[*ModuleName*]:I.Data[3]	4:0107
Arrival Count Downstream	[*ModuleName*]:I.Data[4]	4:0186
Departure Count Downstream	[*ModuleName*]:I.Data[5]	4:0187
Module Status #1	[*ModuleName*]:I.Data[6]	4:0088
Module Status #2	[*ModuleName*]:I.Data[7]	4:0089
Tracking High Word Upstream	[*ModuleName*]:I.Data[8]	4:0119
Tracking Low Word Upstream	[*ModuleName*]:I.Data[9]	4:0120
Tracking High Word Downstream	[*ModuleName*]:I.Data[10]	4:0199
Tracking Low Word Downstream	[*ModuleName*]:I.Data[11]	4:0200
Release Upstream	[*ModuleName*]:I.Data[12]	4:0105
Release Downstream	[*ModuleName*]:I.Data[13]	4:0185
Reserved	[*ModuleName*]:I.Data[14]	Reserved
Reserved	[*ModuleName*]:I.Data[15]	Reserved
Reserved	[*ModuleName*]:I.Data[16]	Reserved
Reserved	[*ModuleName*]:I.Data[17]	Reserved
Sensor & Control Port Inputs	[*ModuleName*]:I.Data[18]	4:0035
Current Module Reset Counter	[*ModuleName*]:I.Data[19]	Reserved

Output Instance for ZPA Mode – PLC Enable (Instance ID 26)

Item Description	Logix Tag Address	Modbus Register Address
Tracking High Word Upstream[1]	[*ModuleName*]:O.Data[0]	4:0132
Tracking Low Word Upstream[1]	[*ModuleName*]:O.Data[1]	4:0133
Tracking High Word Downstream[1]	[*ModuleName*]:O.Data[2]	4:0212
Tracking Low Word Downstream[1]	[*ModuleName*]:O.Data[3]	4:0213
Accumulate Enable Upstream	[*ModuleName*]:O.Data[4]	4:0104
Accumulate Enable Downstream	[*ModuleName*]:O.Data[5]	4:0184
Speed Upstream[2]	[*ModuleName*]:O.Data[6]	4:0040
Speed Downstream[2]	[*ModuleName*]:O.Data[7]	4:0064
Release Upstream	[*ModuleName*]:O.Data[8]	4:0105
Release Downstream	[*ModuleName*]:O.Data[9]	4:0185
Upstream Module Status[3]	[*ModuleName*]:O.Data[10]	4:0134
Downstream Module Status[3]	[*ModuleName*]:O.Data[11]	4:0232
Reserved	[*ModuleName*]:O.Data[12]	Reserved
Reserved	[*ModuleName*]:O.Data[13]	Reserved
Reserved	[*ModuleName*]:O.Data[14]	Reserved
Reserved	[*ModuleName*]:O.Data[15]	Reserved
Clear Motor Error	[*ModuleName*]:O.Data[16]	4:0022
Control Port Outputs[4]	[*ModuleName*]:O.Data[17]	4:0063
Previous Module Reset Counter	[*ModuleName*]:O.Data[18]	Reserved
Reserved	[*ModuleName*]:O.Data[19]	Reserved

[1] **Tracking Registers Functionality in Ethernet I/P:** Same note as for Output Instance for ZPA Mode (Instance ID 6) on page 29.

[2] **Speed Control for ZPA Mode in Ethernet I/P:** Same note as for Output Instance for ZPA Mode (Instance ID 6) on page 29.

[3] **Affecting Adjacent *ERSC* Communication in Ethernet I/P:** Same note as for Output Instance for ZPA Mode (Instance ID 6) on page 29.

[4] **Using *ERSC*'s Control Port:** Same note as for Output Instance for ZPA Mode (Instance ID 6) on page 29.

Creating ZPA Mode – PLC Enable Instance on RSLogix 5000

Steps #1, #2, and #4 are identical to the instructions shown in section Creating ZPA Mode Instance in RSLogix 5000 on page 31.

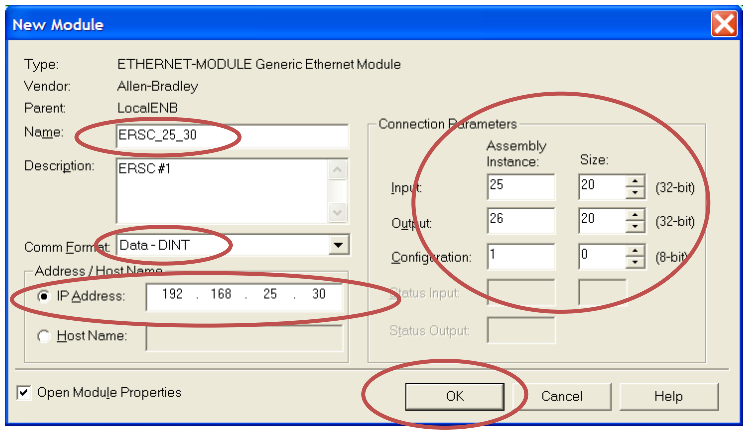
The only difference is in Step #3 as shown below. The Assembly Instance values change from 5 and 6 to 25 and 26. All other parameters remain the same:

Step #3

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, substituting "25" and "26" for "5" and "6".

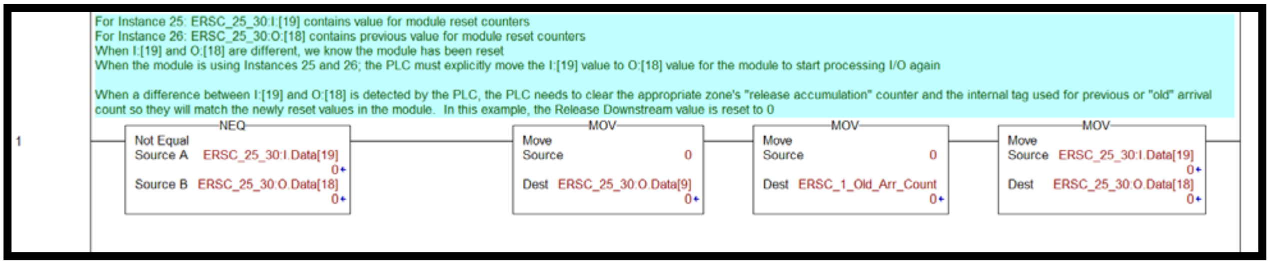


It is <u>very important</u> to select *Comm Format* data type to be INT or interface to *ERSC* will not operate correctly!

Programming Example for PLC Enable Instance

From the example shown in section *Programming Example* beginning on page 33, let's configure *ERSC* #1 to use Instances 25 and 26. The accumulate and release functions in the example program remain unchanged. The only addition will be to add the detection of the difference between the current and previous *ERSC* reset counter values in order for the PLC to reset its internal arrival counter and then enable the *ERSC* to operate.

The sample rung below shows the logic:



To make sure *ERSC* #1 Downstream zone accumulates upon power up, use *EasyRoll* configuration tool software to set the zone to "Accumulate". When set from *EasyRoll*, this will be retained in the flash memory of the *ERSC* so that the zone will initially accumulate if a load happens to be in the zone at the time of power up.

Be sure to click the "Accumulate" switch on the main *EasyRoll* screen to cause the zone to accumulate upon power up

When a zone is commanded to accumulate via Ethernet I/P with a PLC, the "Accumulate" switch icon on *EasyRoll* main screen will visibly indicate that the zone is accumulated. This visible icon will look the same as if it was clicked ON from *EasyRoll*.

Keep in mind that the PLC command to accumulate over Ethernet I/P <u>IS NOT retained upon power loss to the *ERSC*</u>. Only if the "Accumulate" switch is toggled "ON" in *EasyRoll* is the accumulate condition for the zone retained in flash memory for use upon power-up.

External Controller with PLC I/O Mode

When an *ERSC* is in PLC I/O mode, all automatic functions of detecting loads and running motors are suspended by the local *ERSC* on-board logic and the external controller must explicitly read inputs and write data output to cause motors to run. The following items are available for external controller when *ERSC* is in PLC I/O Mode:

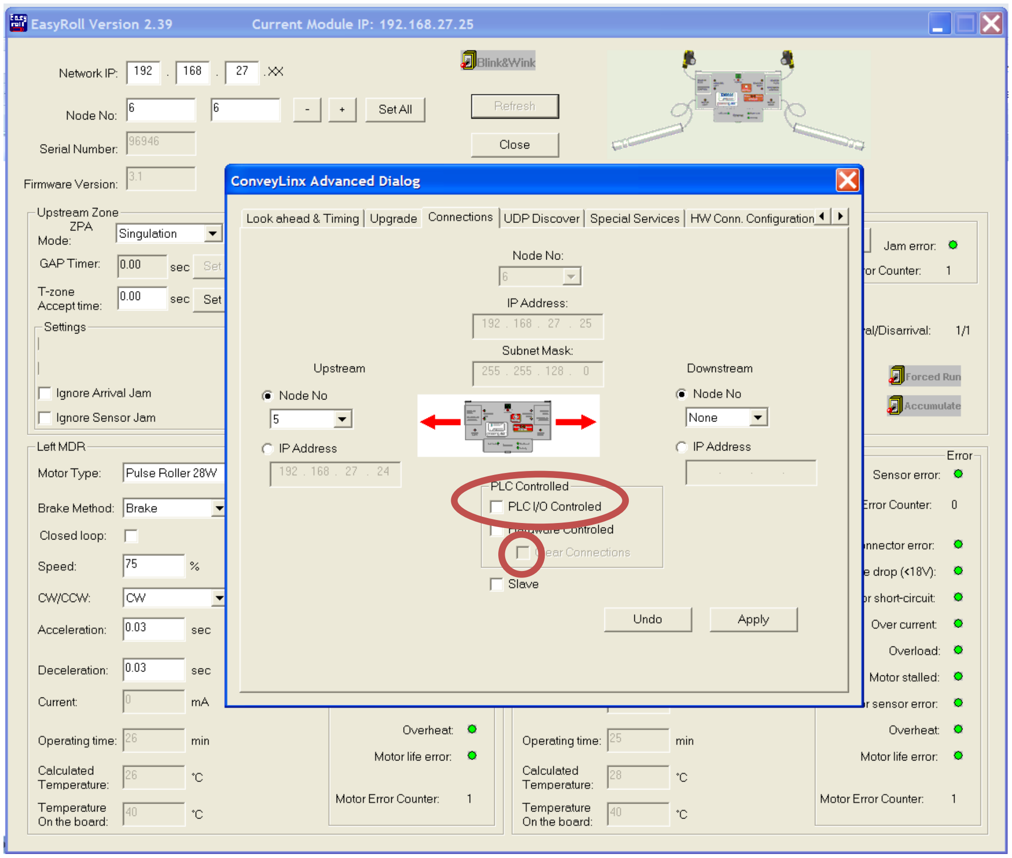
- Status of all available digital inputs on Sensor and Control ports (8 total inputs)
- Module voltage reading
- Left and Right motor status of frequency, current, and calculated temperature
- Left and Right motor diagnostic error status word
- Control of Control Port digital outputs
- Ability to independently run both Left and Right motors
- Ability to set speed, acceleration, deceleration, PI Mode, and Braking method for Left and Right motors
- Ability to configure one or both motor ports to digital output mode
- Ability to remotely clear fatal motor error condition
- Ability to instruct module to E-Stop motor outputs

Refer to *ConveyLinx User's Guide* (publication *ERSC*-1000) for connection details for Sensor and Control Port input and output signals.

Setting PLC I/O Mode in EasyRoll

Individual *ERSC*'s must be placed into PLC I/O Mode from the *EasyRoll* software tool. This is done by invoking the Advanced Dialog and using the Connections tab.

Configuring *ERSC* for PLC I/O Mode



From the main screen, first enter the correct *Subnet* into the "Network IP" boxes and the correct *Node* you want to connect. Invoke the *ConveyLinx Advanced Dialog* and select the *Connections* tab.

Note that the *Node* is being viewed is in the center and it is greyed out. Select the "PLC I/O Controlled" checkbox. With this checked the "Clear Connections" checkbox becomes enabled. **Check or Uncheck the "Clear Connections" checkbox depending upon your application.** Click "Apply" to initiate the change. The *ERSC* will restart and this may take several seconds to complete.

Optional "Clear Connections" Choice

The decision to "Clear Connections" is based upon the application. When a string of *ERSC* modules are Auto-Configured, each successive *ERSC* in the string establishes a logical upstream / downstream **connection** with its neighbor *ERSC*'s. These **connections** provide the basis for the logical flow of inter-module status data for ZPA functionality as described in section *ZPA Mode Register Data Flow* on page 24. However, if a single *ERSC* node within a string of ZPA configured nodes needs to be utilized in PLC I/O mode, these logical connections can remain in place and be used to PLC programmer's advantage.

By **NOT** clearing the **connections**, the *ERSC* in PLC I/O will maintain its inter-module data exchange as described in section *ZPA Mode Register Data Flow* on page 24. This could be advantageous for applications where you want to control a specialized conveyor section such as a right angle transfer or merge conveyor with an *ERSC* in PLC I/O mode. For example, if the PLC I/O mode *ERSC* is being either fed or feeds conveyors controlled by *ERSC*'s in standard ZPA mode, these *ERSC*'s will populate PLC I/O configured module's registers with their respective status data. Likewise, the PLC can manipulate the PLC I/O configured module's zone status registers and these registers will automatically be written to the adjacent *ERSC*'s by virtue of these already established **connections** without requiring the PLC to explicitly perform the messaging.

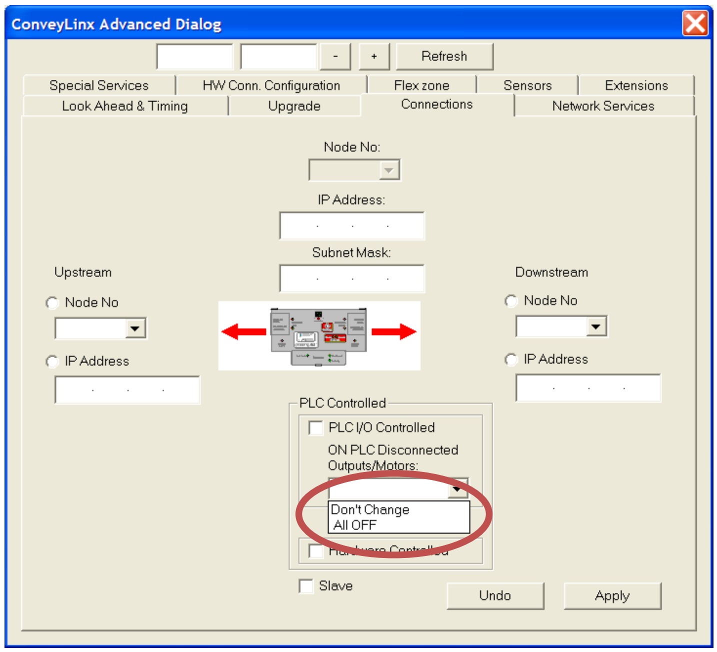
If you choose the option to "Clear Connections", this automatic data transfer of status is inhibited. This means that for an *ERSC* module in PLC I/O mode whose connections have been cleared; its status registers are not automatically written to its adjacent neighbors. Clearing the **connections** could be advantageous when utilizing several ERSC's in a row configured as PLC I/O where having this additional inter-module communication is not required and would only add to unnecessary communication bandwidth usage.

IMPORTANT NOTE: Once a given *ERSC* has been placed in PLC I/O mode, the ONLY way to return it to ZPA mode is to perform an Auto-Configuration procedure on the entire subnet in which the module resides. There is no "undo" or "reset" function for this action.

Configuring Action for Loss of Communication

When changing the mode of a given *ERSC* to PLC I/O mode in *EastRoll*, you are given the option to select the behavior of the *ERSC's* outputs upon loss of communications with the PLC.

Configuring Communication Loss Action for *ERSC* in PLC I/O Mode



Select "Don't Change" if you want module's logical outputs and MDR's to remain in the state they were in at the time of the communication loss

Select "All OFF" to instruct the *ERSC* to turn off all logical outputs and stop all MDR's at the time of communication loss.

Upon re-establishing communications with the PLC, the *ERSC* will automatically resume having its outputs and MDR's controlled by PLC command.

Registers for ERSC in PLC I/O Mode

When an *ERSC* is placed in PLC I/O mode; it suspends all of its internal ZPA logic control. Any sensors or motors connected to the *ERSC* require explicit interaction with an external controller. The external controller will have typical Ethernet-based remote I/O performance from an *ERSC* when in PLC I/O mode.

Sensor & Control Port Digital I/O – PLC I/O Mode

Item Description	4:xxxx Register	Item Usage
Outputs	4:0037	Bitwise Value – "1" Energizes Output bit 0 = Reserved bit 1 = Left Control Port bit 2 = Reserved bit 3 = Right Control Port
Inputs	4:0035	Bitwise Value - Read only bit 0 = Left Sensor Port - Pin 3 bit 1 = Left Hardware Port - Pin 3 bit 2 = Right Sensor Port - Pin 3 bit 3 = Right Hardware Port - Pin 3 bit 4 = Left Sensor Port - Pin 4 bit 5 = Left Hardware Port - Pin 4 bit 6 = Right Sensor Port - Pin 4 bit 7 = Right Hardware Port - Pin 4
Module Voltage	4:0024	Value in mV of Module Power Supply Range: 0 to 35000 *Example:23500 = 23.5 Volts*
Module E-Stop	4:0019	Integer Value 1 = E-Stop condition – reset all digital outputs to "0" and stop all motors 0 = Run Condition – digital outputs and motors under PLC logic control

Left Motor Control & Status – PLC I/O Mode

Item Description	*4:xxxx Register*	*Item Usage*	
Run / Reverse	4:0260	Bit 0: 1 = Run Command 0 = Stop Command Bit 8: 0 = Run in Configured Direction 1 = Run opposite of Configured Direction	
Speed Reference	4:0040	Value in % PWM Range: 0 to 1000 *Example: 400 = 40%* 0 = Run at Configured Speed	
Acceleration Ramp	4:0043	Time value in milliseconds Range: 0 to 1000 *Example: 900 = 0.9 sec* 0 = Use Configured Acceleration	
Deceleration Ramp	4:0044	Time value in milliseconds Range: 0 to 1000 *Example: 900 = 0.9 sec* 0 = Use Configured Deceleration	
Brake Method	4:0261	Integer Value 0 = Use Configured Brake Method 1 = Use Standard Brake Method 2 = Use Free Coast Brake Method 3 = Use Servo 1 Brake Method 4 = Use Servo 2 Brake Method	
Speed Control Method	4:0262	Integer Value 0 = Use Configured Speed Control Method 1 = Use Open Loop 2 = Use Closed Loop 3 = Use Servo 1 Brake Method 4 = Use Servo 2 Brake Method	
Motor Status	4:0058	Bitwise Value	
		bit 00 = Motor Status* bit 01 = Motor Status* bit 02 = Reserved bit 03 = Reserved bit 04 = Reserved bit 05 = Reserved bit 06 = Reserved bit 07 = Low Voltage	bit 08 = Overheated bit 09 = Over Current bit 10 = Short Circuit bit 11 = Motor Not Connected bit 12 = Overloaded bit 13 = Motor Stalled bit 14 = Hall Sensor Error bit 15 = Motor Not Used
Motor Current	4:0055	Integer Value in mA – Current that motor is currently drawing *For example: 1900 = 1.9 Amps*	
Motor Frequency	4:0056	Integer Value in Hz – Current PWM frequency that motor is currently running *For example: 300 = 300 Hz*	
Temperature	4:0057	High Byte / Low Byte Value of temperatures in °C High Byte = Calculated motor temperature Low Byte = Temperature reading from on-board sensor	
Motor Fault Reset	4:0022	Logical 0 or 1 0 = Stop Reset 1 = Send Reset	

*For Motor Status bit 0 and bit 1		
Bit 1	*Bit 0*	*Description*
0	0	Motor not running, standard or servo braking applied
0	1	Motor running in configured direction
1	0	Motor running opposite of configured direction
1	1	Motor not running and no braking applied (free to spin)

Right Motor Control & Status – PLC I/O Mode

Item Description	4:xxxx Register	Item Usage
Run / Reverse	4:0270	Bit 0 1 = Run Command 0 = Stop Command Bit 8 0 = Run in Configured Direction 1 = Run opposite of Configured Direction
Speed Reference	4:0064	Value in % PWM Range: 0 to 1000 *Example: 400 = 40%* 0 = Run at Configured Speed
Acceleration Ramp	4:0067	Time value in milliseconds Range: 0 to 1000 *Example: 900 = 0.9 sec* 0 = Use Configured Acceleration
Deceleration Ramp	4:0068	Time value in milliseconds Range: 0 to 1000 *Example: 900 = 0.9 sec* 0 = Use Configured Deceleration
Brake Method	4:0271	Integer Value 0 = Use Configured Brake Method 1 = Use Standard Brake Method 2 = Use Free Coast Brake Method 3 = Use Servo 1 Brake Method 4 = Use Servo 2 Brake Method
Speed Control Method	4:0272	Integer Value 0 = Use Configured Speed Control Method 1 = Use Open Loop 2 = Use Closed Loop 3 = Use Servo 1 Brake Method 4 = Use Servo 2 Brake Method
Motor Status	4:0082	Bitwise Value bit 00 = Motor Status* bit 01 = Motor Status* bit 02 = Reserved bit 03 = Reserved bit 04 = Reserved bit 05 = Reserved bit 06 = Reserved bit 07 = Low Voltage bit 08 = Overheated bit 09 = Over Current bit 10 = Short Circuit bit 11 = Motor Not Connected bit 12 = Overloaded bit 13 = Motor Stalled bit 14 = Hall Sensor Error bit 15 = Motor Not Used
Motor Current	4:0079	Integer Value in mA – Current that motor is currently drawing *For example: 1900 = 1.9 Amps*
Motor Frequency	4:0080	Integer Value in Hz – PWM frequency that *ERSC* is sending to motor *For example: 300 = 300 Hz*
Temperature	4:0081	High Byte / Low Byte Value of temperatures in °C High Byte = Calculated motor temperature Low Byte = Temperature reading from on-board sensor
Motor Fault Reset	4:0022	Logical 0 or 1 0 = Stop Reset 1 = Send Reset

**For Motor Status bit 0 and bit 1*		
Bit 1	*Bit 0*	*Description*
0	0	Motor not running, standard or servo braking applied
0	1	Motor running in configured direction
1	0	Motor running opposite of configured direction
1	1	Motor not running and no braking applied (free to spin)

Motor Short Circuit and Motor Hall Sensor Error are classified as “fatal” errors that require either a cycle of power on the *ERSC* or an explicit Motor Fault Reset command from external controller.

External controller must continuously write “1” to the *Motor Fault Reset* register for at least 500 msec for reset to occur.

***Motor Frequency* values are MDR brand and model dependent and are based upon number of motor poles and RPM. Please consult your particular MDR manufacturer’s documentation for expected values.**

Adjacent Connected Module Data – PLC I/O Mode

The following are some useful registers for applications where the ConveyLinx **connections** are maintained between an *ERSC* in PLC I/O mode and adjacent *ERSC*'s remaining in ZPA mode.

Item Description	*4:xxxx Register*	*Item Usage*
Tracking Word – High Upstream Connected *ERSC*[1]	4:0139	Most Significant 16 Bits of 32 Bit Tracking Value from Upstream module
Tracking Word – Low Upstream Connected *ERSC*[1]	4:0140	Least Significant 16 Bits of 32 Bit Tracking Value from Upstream module
Tracking Word – High to Downstream Connected *ERSC*[2]	4:0201	Most Significant 16 Bits of 32 Bit Tracking Value to Downstream module
Tracking Word – Low to Downstream Connected *ERSC*[2]	4:0202	Least Significant 16 Bits of 32 Bit Tracking Value to Downstream module
Sensor & Control Port Input Signal LED Settings[3]	4:0034	Bitwise Value bit 0 = Left Sensor Port - Pin 3 Red LED bit 1 = Left Control Port - Pin 3 Red LED bit 2 = Right Sensor Port - Pin 3 Red LED bit 3 = Right Control Port - Pin 3 Red LED bit 4 = Left Sensor Port - Pin 4 Green LED bit 5 = Left Control Port - Pin 4 Green LED bit 6 = Right Sensor Port - Pin 4 Green LED bit 7 = Right Control Port - Pin 4 Green LED

[1] **Upstream Tracking Registers:** When ConveyLinx connections are maintained to an *ERSC* in PLC I/O mode; the upstream *ERSC* in ZPA mode will still pass along its tracking data the same as it passes its status. When the upstream modules discharge zone releases its load, when its zone status changes from "Sensor blocked and Running" (value = 5) to any of its next possible states (value < 5), this upstream *ERSC* writes its tracking data to these registers in the downstream *ERSC* that is in PLC I/O mode. In this way tracking is automatically transferred from ZPA *ERSC*'s to PLC I/O *ERSC* that may be controlling a merge or transfer conveyor. Please note the PLC must manipulate this tracking data as required and explicitly write it to any downstream *ERSC*'s based upon conveyor application and function.

[2] **To Downstream Tracking Registers:** In a similar fashion to the Upstream Tracking Registers, the PLC can populate these tracking data registers with pertinent data. When the PLC manipulates the PLC I/O mode's *ERSC* register 4:0196 with a value of 5 (sensor blocked and motor running) indicating that a load is on its way; the automatic data transfer due to ConveyLinx **connections** will transfer the data in 4:0201 and 4:0202 to the downstream *ERSC*.

[3] **Sensor & Control Port Input Signal LED Settings:** For an *ERSC* in standard ZPA mode, the Auto-Configuration procedure sets values in this register to allow the *ERSC* to correctly display the Sensor and Control port Input circuit LEDs to facilitate diagnostics. This is done, for example, to make visual LED diagnostics the same for "zone blocked" regardless of the sensor type.

For example, suppose the zone photo sensors used are “Light Operate, Normally Open”. This means that the sensor’s output is energizing the *ERSC*’s sensor input pin 4 when the zone is clear. The *ERSC* Sensor port LED indicator for pin 4 (green) should illuminate when the zone is blocked; so the Auto-Configuration procedure sets a bit in register 4:0034 to correspond to the pin 4 signal on the appropriate sensor port. If the sensor is electrically opposite such that its output energizes pin 4 of the sensor port when the zone is blocked, then the bit corresponding to pin 4 for this sensor port is clear such that the sensor port’s LED illuminates green when pin 4 is energized.

When an *ERSC* is placed in PLC I/O mode; register 4:0034 is cleared of the values set during the Auto-Configure procedure. Register 4:0034 is made available for PLC I/O mode Sensor and Control port inputs to give the PLC programmer the same flexibility for configuring LED operation. By setting or clearing the corresponding bit for a given port’s pin 3 or 4 signal, the PLC programmer can determine which physical state (on or off) of the input will cause the signal’s corresponding LED to illuminate. A bit value of “1” will cause the LED to illuminate when the physical input is “OFF” and a bit value of “0” will cause the LED to illuminate when the physical input is “ON”.

Motor Ports as Digital I/O – PLC I/O Mode

Any given *ERSC*, once in PLC I/O Mode, can be set by an external controller to convert either or both motor ports to provide three (3) 24VDC digital outputs each. When a motor port is used as digital output, the *ERSC* provides status data relevant to digital output operation.

Left Motor Port as Digital Output

Item Description	4:xxxx Register	Item Usage
Control & Status	4:0060	Bitwise Value – "1" Energizes Output bit 0 = Motor Port Pin 3 bit 1 = Motor Port Pin 4 bit 2 = Motor Port Pin 5 bit 3 = Reserved bit 4 = Reserved bit 5 = Reserved bit 6 = Reserved bit 7 = Reserved bit 8 = Reserved bit 9 = Reserved bit 10 = Reserved bit 11 = Reserved bit 12 = Short Circuit Error on one or more outputs bit 13 = Reserved bit 14 = Over Current – More than 1A detected on one or more outputs bit 15 = Digital Output Enable 0 = Use port as Motor Control 1 = Use port as Digital Output

Right Motor Port as Digital Output

Item Description	4:xxxx Register	Item Usage
Control & Status	4:0084	Bitwise Value – "1" Energizes Output bit 0 = Motor Port Pin 3 bit 1 = Motor Port Pin 4 bit 2 = Motor Port Pin 5 bit 3 = Reserved bit 4 = Reserved bit 5 = Reserved bit 6 = Reserved bit 7 = Reserved bit 8 = Reserved bit 9 = Reserved bit 10 = Reserved bit 11 = Reserved bit 12 = Short Circuit Error on one or more outputs bit 13 = Reserved bit 14 = Over Current – More than 1A detected on one or more outputs bit 15 = Digital Output Enable 0 = Use port as Motor Control 1 = Use port as Digital Output

Motor Port as Digital Output Usage

External controller must first set bit 15 = 1 in the *Control & Status* register for the motor port (Left or Right) that is to be used as digital output. If bit 15 = 0, then *ERSC* ignores the bit 0 thru bit 2 commands and will not provide meaningful status on bits 12 and 14. Bit 0, bit 1, and bit 2 can be independently set and reset by the external controller and all 3 digital outputs can be energized simultaneously.

Short Circuit Error on bit 12 is classified as a "fatal" error that will require either a cycle of power on the *ERSC* or an explicit Motor Fault Reset command from external controller the same as if the port was being used as a motor port.

External controller must continuously write "1" to the *Motor Fault Reset* register for at least 500 msec for reset to occur.

Figure 7 shows the wiring diagram for a given motor port configured for digital output. Please note that the ERSC switches to GND to complete the circuit.

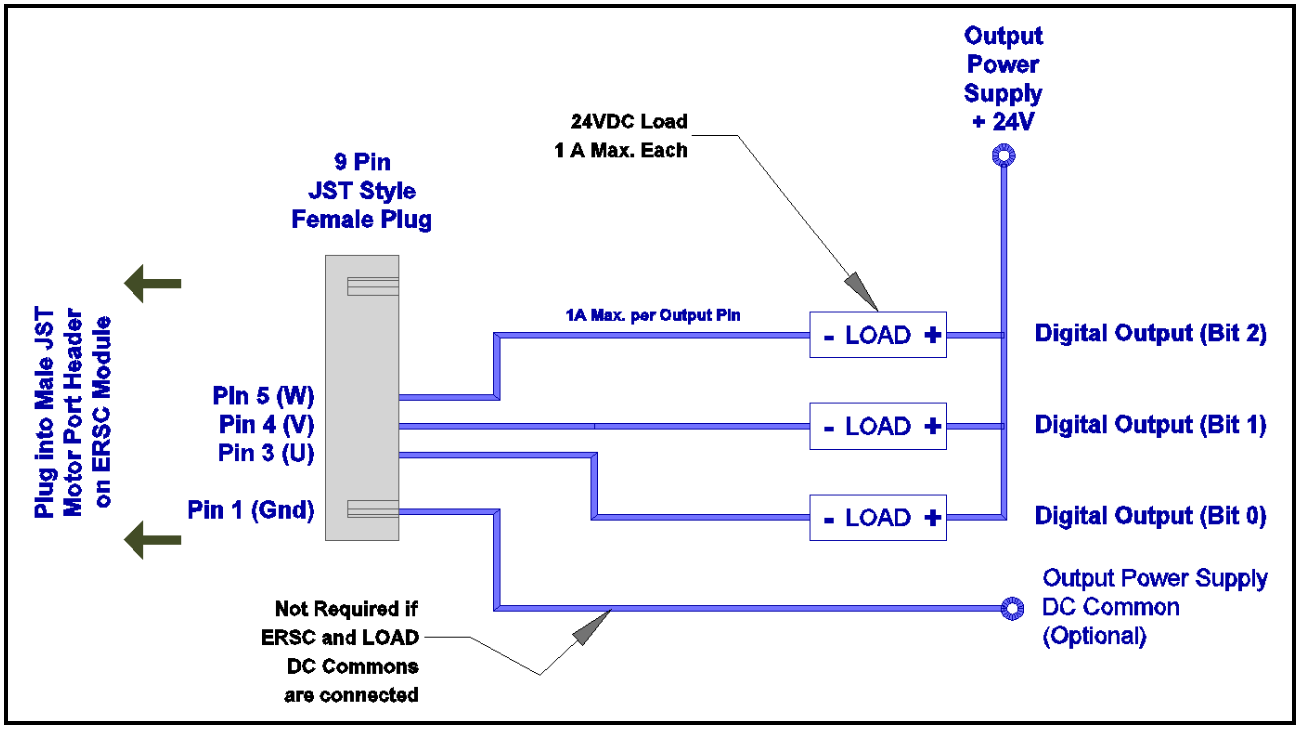


Figure 7 - Wiring Diagram for Motor Port Configured for Digital Output

Connection to Pin 1 (GND) as shown is only required if the Output Power Supply is at different DC common than the ERSC's power supply. If both power supplies have the same DC common, then this connection is not required.

Ethernet I/P Controller with PLC I/O Mode

This section assumes reader is experienced with Allen-Bradley Logix 5000 programming software and is familiar with User Defined Data Type structures and attaching Ethernet I/P Generic I/O device instances to a PLC program project.

The instance created in the Logix 5000 tree for an Ethernet I/P connection to an *ERSC* in PLC I/O mode consists of an Input array of 20 integers and an Output array of 22 registers. The *ERSC* automatically maps the Modbus register associated with the specific data item into one of the array positions. The data values, bit values, and usage are, unless otherwise noted, identical as to the Modbus descriptions.

From this point forward, it is assumed the reader is familiar with Allen-Bradley Logix platform addressing notation:

[*ModuleName*]:O.Data[*Index*].*Bit*

[*ModuleName*]:I.Data[*Index*].*Bit*

Where:

- ***ModuleName* is the instance of the device when created**
- **"O.Data" indicates the output image of the device**
- **"I.Data" would indicate input image of the device**
- **"[*Index*].*Bit*" indicates the word and bit within the image. If the bit notation is absent the notation refers to the entire word data type**

Input Instance for PLC I/O Mode (Instance ID 7)

Item Description	*Logix Tag Address*	*Modbus Register Address*
Reserved	[*ModuleName*]:I.Data[0]	Reserved
Sensor & Control Port Inputs	[*ModuleName*]:I.Data[1]	4:0035
Reserved	[*ModuleName*]:I.Data[2]	Reserved
Module Voltage	[*ModuleName*]:I.Data[3]	4:0024
Left Motor Current	[*ModuleName*]:I.Data[4]	4:0055
Left Motor Frequency	[*ModuleName*]:I.Data[5]	4:0056
Left Motor Temperature	[*ModuleName*]:I.Data[6]	4:0057
Left Motor Status	[*ModuleName*]:I.Data[7]	4:0058
Right Motor Current	[*ModuleName*]:I.Data[8]	4:0079
Right Motor Frequency	[*ModuleName*]:I.Data[9]	4:0080
Right Motor Temperature	[*ModuleName*]:I.Data[10]	4:0081
Right Motor Status	[*ModuleName*]:I.Data[11]	4:0082
Left Motor Port Digital I/O Status	[*ModuleName*]:I.Data[12]	4:0060
Right Motor Port Digital I/O Status	[*ModuleName*]:I.Data[13]	4:0084
Upstream Module Status[1]	[*ModuleName*]:I.Data[14]	4:0134
Downstream Module Status[1]	[*ModuleName*]:I.Data[15]	4:0232
Upstream Tracking Word Hi[1]	[*ModuleName*]:I.Data[16]	4:0139
Upstream tracking Word Lo[1]	[*ModuleName*]:I.Data[17]	4:0140
Reserved	[*ModuleName*]:I.Data[18]	Reserved
Reserved	[*ModuleName*]:I.Data[19]	Reserved

[1] **Status and Tracking Registers:** These registers only contain meaningful data if the ConveyLinx connections between upstream and/or downstream *ERSC*'s are preserved when placing the *ERSC* in question into PLC I/O mode from within *EasyRoll*. If connections are cleared in *EasyRoll*, these registers will not contain and pertinent data and will not be updated by adjacent *ERSC*'s.

For Sensor and Control Port inputs register I.Data[1], the *ERSC* transmits Ethernet I/P messaging to PLC immediately upon data state change and does not wait for the next RPI cycle.

Output Instance for PLC I/O Mode (Instance ID 8)

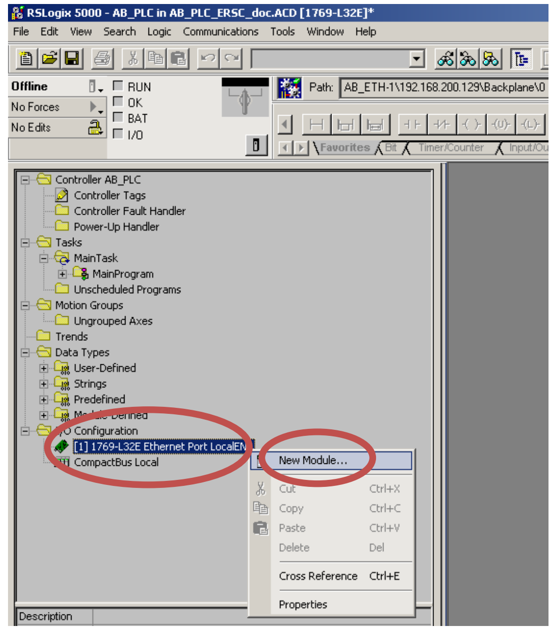
Item Description	Logix Tag Address	Modbus Register Address
Module E-Stop	[*ModuleName*]:O.Data[0]	4:0019
Left Motor Port Digital Control	[*ModuleName*]:O.Data[1]	4:0060
Right Motor Port Digital Control	[*ModuleName*]:O.Data[2]	4:0084
Sensor & Control Port Digital Output Control	[*ModuleName*]:O.Data[3]	4:0037
Left Motor Run / Reverse	[*ModuleName*]:O.Data[4]	4:0260
Left Motor Brake Method[1]	[*ModuleName*]:O.Data[5]	4:0261
Left Motor Speed Control Method[1]	[*ModuleName*]:O.Data[6]	4:0262
Right Motor Run / Reverse	[*ModuleName*]:O.Data[7]	4:0270
Right Motor Brake Method[1]	[*ModuleName*]:O.Data[8]	4:0271
Right Motor Speed Control Method[1]	[*ModuleName*]:O.Data[9]	4:0272
Left Motor Speed Reference[1]	[*ModuleName*]:O.Data[10]	4:0040
Right Motor Speed Reference[1]	[*ModuleName*]:O.Data[11]	4:0064
Left Motor Acceleration Ramp[1]	[*ModuleName*]:O.Data[12]	4:0043
Left Motor Deceleration Ramp[1]	[*ModuleName*]:O.Data[13]	4:0044
Right Motor Acceleration Ramp[1]	[*ModuleName*]:O.Data[14]	4:0067
Right Motor Deceleration Ramp[1]	[*ModuleName*]:O.Data[15]	4:0068
Clear Motor Error	[*ModuleName*]:O.Data[16]	4:0022
Status to Downstream Module	[*ModuleName*]:O.Data[17]	4:0196
Status to Upstream Module	[*ModuleName*]:O.Data[18]	4:0116
Sensor Polarity	[*ModuleName*]:O.Data[19]	4:0034
Tracking Word – High to Downstream Connected *ERSC*[2]	[*ModuleName*]:O.Data[20]	4:0201
Tracking Word – Low to Downstream Connected *ERSC*[2]	[*ModuleName*]:O.Data[21]	4:0202

[1] **Motor Control Registers in Ethernet I/P:** Leaving these registers at "0" will instruct the *ERSC* to use its configured motor control settings. Any non-zero value set by the PLC will instruct the *ERSC* to use this PLC-set non-zero value as the setting's value. Any changed setting will stay at the PLC-set values until changed by the PLC to a new non-zero value. If the PLC sets the value to "0", the *ERSC* will revert back to using its configured value.

[2] **Status and Tracking Registers:** These registers only write to adjacent *ERSC*'s if the ConveyLinx connections between upstream and/or downstream *ERSC*'s are preserved when placing the *ERSC* in question into PLC I/O mode from within *EasyRoll*.

Creating PLC I/O Mode Instance on RSLogix 5000

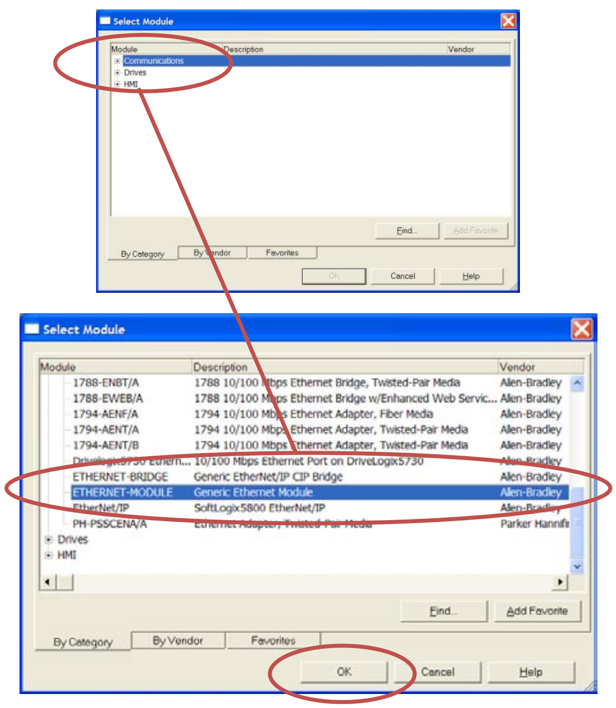
This section will provide the set-by-step procedure for creating an instance of an *ERSC* into the I/O configuration for an Allen-Bradley CompactLogix processor in RSLogix 5000 software.



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

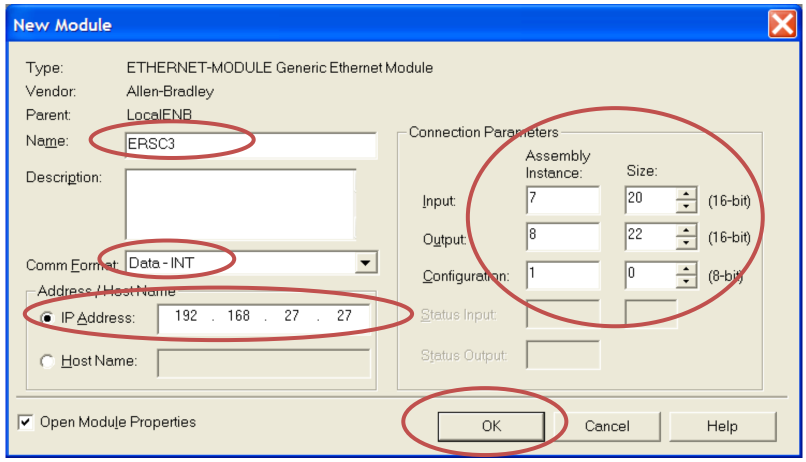
From the Select Module pop-up window, expand the Communications tree and select "Generic Ethernet Module" and click OK

<u>Step #3</u>

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. **<u>Please note that Input Instance 7 has size of 20 and Output Instance 8 has size of 22</u>**

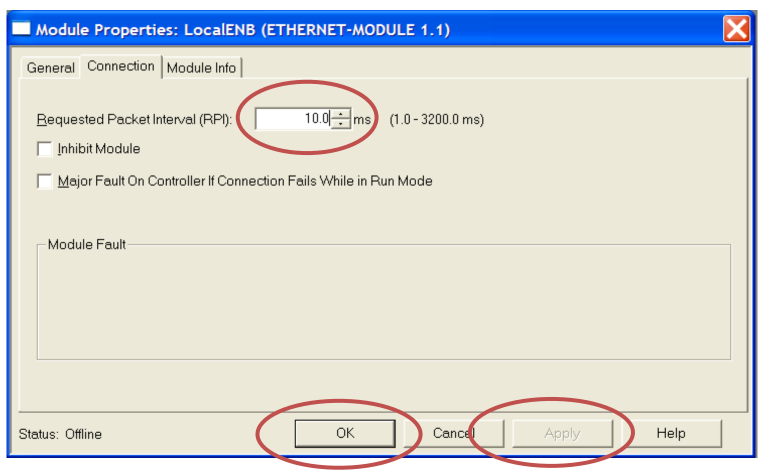


It is <u>very important</u> to select *Comm Format* data type to be INT or interface to *ERSC* will not operate correctly!

<u>Step #4</u>

Set RPI to a value no lower than 10ms

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



Ethernet I/P Guidelines

Each Allen-Bradley PLC has 2 metrics for limiting 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 *ERSC* 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 *ERSC* connections as generic I/O modules utilizing any combination of ZPA mode and PLC I/O Mode instances.

When an *ERSC* is attached as a "full-time generic I/O module" to the PLC, the connection is continually maintained and data exchanged at a minimum RPI value and if the PLC cannot communicate with the *ERSC* 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 *ERSC* it can physically reach over its Ethernet port without the *ERSC* being "full-time connected as a generic I/O module". This is accomplished with a Logix5000 MSG instruction.

Application Guideline:

Reserve Ethernet I/P TCP connections for *ERSC*'s in PLC I/O Mode and for key ZPA Mode *ERSC*s 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).

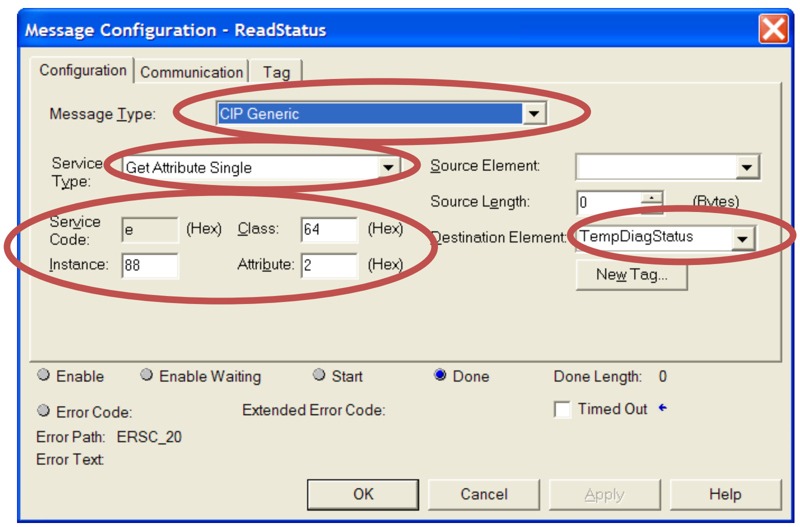
Ethernet I/P Logix5000 MSG Instruction

Any *ERSC* on the network will respond to an appropriately configured Logix5000 MSG instruction without the *ERSC* being attached as a Generic I/O instance to the PLC. The *ERSC* will allow a MSG instruction to read up to 30 contiguous Modbus registers in a single instruction. The *ERSC* will allow a MSG instruction to write 1 Modbus register in a single instruction.

Message Configuration for Reading Data from ERSC

Read MSG Setup

- Select "CIP Generic" as the Message Type
- Select "Get Attribute Single" and the Service Type
- Class is always set to 64
- Instance is the 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 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.

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.

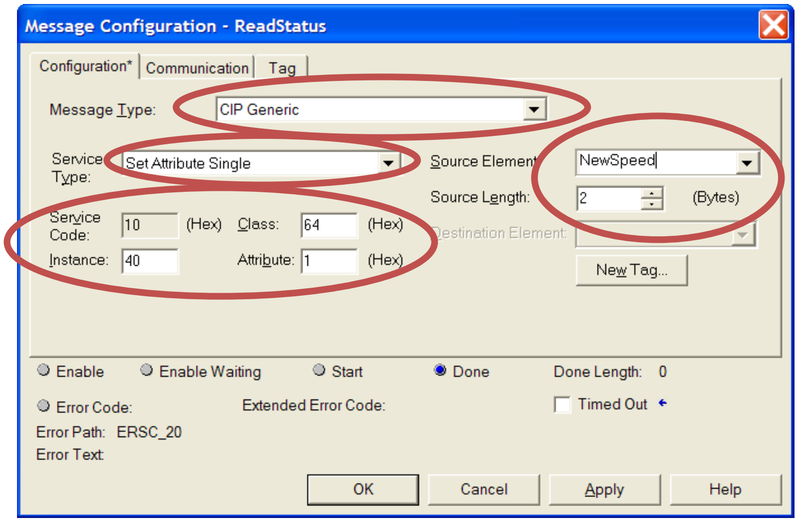
Although a read MSG instruction can be used on an *ERSC* in PLC I/O mode, it is assumed that any *ERSC* in PLC I/O will already be utilizing a permanent TCP connection and should not ever need to be accessed with a read MSG instruction.

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 Writing Data to ERSC

Write MSG Setup

- Select "CIP Generic" as the Message Type
- Select "Set Attribute Single" and 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 upstream zone (Modbus register 4:0040). The tag "NewSpeed" contains the value of speed reference for the *ERSC*'s upstream zone.

Source Element tag MUST be of data type INT or instruction will not produce expected results.

Programmer must be aware of data values residing in Source Element tags and apply the value ranges and usage as described or unexpected results will occur.

Notes:

Industrial Software Co.

45, Lokorska Str.
1225, Sofia, BULGARIA
Phone/Fax: (+359 2) 975 11 80/1/2/3/4
E-mail: indsoft@einet.bg
www.indsoft.bg

Insight Automation Inc.

2748 Circleport Drive
Erlanger, KY USA 41075
859-647-1111
www.insightautomation.cc

Publication ERSC-1500 Rev 4.0 – June 2011