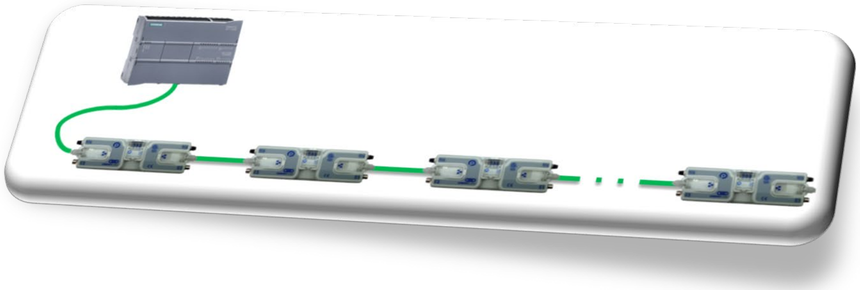
PULSEROLLER

CONVEYLINX

ConveyLinx-Ai2 PLC Developer's Guide

Version 1.1

September 2019



Publication *ERSC*-1506

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

GLOSSARY OF TERMS

ConveyLinx ***(Part No. ERSC)***	**E**thernet **R**oller **S**peed **C**ontrol module – An IP20 rated conveyor control module that is part of the *ConveyLinx* family. Each ConveyLinx ***ERSC*** can accommodate up to 2 **MDR** conveyor **zones.** ERSC models utilize MDR's with 3 individual Hall Effect sensor leads and utilize a JST style connector
ConveyLinx-Ai2 ***(Part No. ERSC-Ai2)***	**E**thernet **R**oller **S**peed **C**ontrol **Ai2** module – An IP54 rated conveyor control module that is part of the *ConveyLinx* family. Each ***ERSC-Ai2*** can accommodate up to 2 **MDR** conveyor **zones.** ERSC-Ai2 models utilize MDR's with a modulated Hall Effect sensor and utilize 4 conductor M8 style motor and sensor connectors.
ConveyStop	Optional functionality available for *ConveyLinx* systems to provide controlled stop and reset of MDR function. Requires software package to configure and enable.
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.
Carton	A separate (usually wrapped or boxed) object to be transported by the conveyor. The terms **tray, tote,** or **load** 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 **Carton** on the zone
PLC	**P**rogrammable **L**ogic **C**ontroller – A wide variety of industrial computing devices that control automatic equipment
Profinet I/O	PROFINET IO uses traditional Ethernet hardware and software to define a network that structures the task of exchanging data, alarms and diagnostics with Programmable Controllers and other automation controllers**.**
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 Carton** can move in unison with the discharging **Carton**.
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 Cartons to queue on a conveyor in discrete **zones** such that Cartons 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-Ai2 modules contain ESD (Electrostatic Discharge) sensitive parts and components. Static control precautions are required when installing, testing, servicing or replacing these modules. Component damage may result if ESD control procedures are not followed. If you are not familiar with static control procedures, reference any applicable ESD protection handbook. Basic guidelines are:

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

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

The illustrations, charts, sample programs and layout examples shown in this guide are intended solely for purposes of example. Since there are many variables and requirements associated with any particular installation, Insight Automation Inc. does not assume responsibility or liability (to include intellectual property liability) for actual use based on the examples shown in this publication. Reproduction of the contents of this manual, in whole or in part, without written permission of Insight Automation Inc. 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	March 2017	Initial Release
1.1	September 2019	Minor typographical revisions

Contact Information

PULSE ROLLER

North & South America

sales@pulseroller.com

support@pulseroller.com

www.pulseroller.com

Global

Global_sales@pulseroller.com

Global_support@pulseroller.com

TABLE OF CONTENTS

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 at least one of Modbus TCP, Ethernet I/P, or Profinet I/O 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 *ConveyLinx-Ai2* module hardware and simple application and installation guidelines, please refer to Insight Automation publication *ConveyLinx-Ai2 User's Guide* (publication *ERSC-1006*)

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-Ai2* modules interconnected via standard Ethernet cabling to form an integrated solution for MDR (Motorized Drive Roller) conveyor functionality. Each *ConveyLinx-Ai2* module can accommodate up to 2 MDR's and 2 photo-sensors to provide control for up to 2 conveyor zones. Each *module* also includes convenient connectivity ports for upstream and downstream Ethernet network cabling as well as connectivity for discrete I/O signals with non-networked controls for local interlock interface functions.

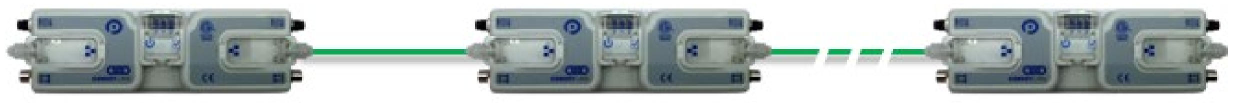


FIGURE 1 - *CONVEYLINX*® CONCEPT WITH CONVEYLINX-AI2 MODULES

Utilizing the complimentary *EasyRoll* software tool, modules can be easily configured to operate multiple zones of linear conveyor "right out of the box". Also, with the *Easy Roll* software tool and a PC; each module's default configuration can be modified to customize functionality for specific applications.

Please refer to Insight Automation publication ERSC-1006 *ConveyLinx-Ai User's Guide* for module hardware details and basic function. The remaining sections of this document assumes reader if familiar with *ConveyLinx-Ai2* module and *EasyRoll* software operation.

ConveyLinx-Ai2 Operational Modes

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

- **Zero Pressure Accumulation (ZPA) Mode**
- **PLC I/O Mode (with or without ConveyLogix program)**

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 mode of any *ConveyLinx-Ai2* that has been configured by the *Auto-Configuration Procedure* who had at least one sensor connected at the time of Auto-Configuration. In this mode, each module has established logical connections to its neighboring 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-Ai2 User's Guide* (publication *ERSC*-1006).

If a ConveyLinx-Ai2 module has no sensors connected during the Auto-Configuration Procedure, it will default to PLC I/O Mode

PLC I/O Mode

In PLC I/O Mode, the *ConveyLinx-Ai2* 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 *module*'s internal data registers over the Ethernet network using Ethernet I/P, Modbus TCP, or Profintet I/O protocol in order to initiate all *ConveyLinx-Ai2* functionality.

Modules can be placed in PLC I/O Mode by either using the *EasyRoll* software tool after they have been through an *Auto-Configuration Procedure* OR by having no sensor devices connected prior to running the *Auto-Configuration Procedure*.

IMPORTANT CONCEPT TO REMEMBER: **A PLC or external controller can connect and interact with any *ConveyLinx-Ai2* module 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 *module* 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:

ConveyLinx-Ai2 Control Strategy Quick Reference		
Any Mode	**Read Status**	• **Available to any Modbus TCP, Profinet I/O, or Ethernet I/P networked device regardless of Module Mode** • **Read Sensor & Control port inputs** • **Read MDR diagnostics**
ZPA Mode	**Default**	• **No PLC required** • **Hard-wired connections available to control zone stop/release** • **Some electrical connections will require additional M8 splitter devices to access all signals** • **See ConveyLinx-Ai2 User's Guide for connection details**
	Interface PLC	• **Ethernet I/P, Profinet I/O, 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 21 for I/O register details** • **See page 32 for Examples**
PLC I/O Mode	**Interface PLC**	• **Default Out of the Box Mode** • **Suspends ALL ZPA functionality** • **PLC has complete access and control of Sensor Port I/O** • **PLC has complete control of both MDR ports (speed, accel/decel, braking method, etc.)** • **See page 46 for setting PLC I/O Mode** • **See page 49 Input register descriptions** • **See page 53 for Output register descriptions** • **See page 59 for servo movement example** • **One or both MDR ports can be changed to provide up to 2 high powered digital outputs (1A each) per MDR port for a total of 4 digital outputs available. See page 67 for details**
	ConveyLogix	• **Requires user program download from *ConveyLogix* software** • **Does not require Interface PLC**
	ConveyLogix & Interface PLC	• **Supports connection to Interface PLC** • **See page 65 for details on ConveyLogix interface with PLC**

CONVEYLINX NETWORK ARCHITECTURE

Each *module* Communicates to its adjacent modules and to any connected PC or PLC via Ethernet physical media. *ConveyLinx* modules recognize (3) TCP/IP based protocols: Modbus TCP, Profinet I/O, 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 recognized by *ConveyLinx* modules and any given *module* can be attached to an Ethernet I/P capable PLC (Allen-Bradley ControlLogix or CompactLogix platforms) and be recognized as a "Generic Ethernet Module" or explicitly as an module by using Insight Automation provided EDS file. Profinet I/O is recognized by module modules and can be attached to any Profinet I/O capable PLC (applicable Siemens platforms). All protocols access the same internal data locations on a given *module*.

Each *ConveyLinx-Ai2* module's internal data structure is arranged and addressed as Modbus Holding registers. The on-board Communication and control processes attach logical meanings to each holding register and read and write data to specific registers to initiate and/or react to events. Certain registers contain information as to how the *module* is configured (MDR type, speed, direction, etc.) for its local controls. Other registers are used for inter-module communications for conveyor operation. For example, when an upstream *module* has a Carton ready to discharge to its neighboring downstream *module*, the upstream *module* will write a specific value to a specific address in the downstream *module*'s internal registers. The downstream *module*'s on board logic monitors these registers and knows that a specific value means that an upstream Carton is coming and to engage the proper control logic to convey the Carton.

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

UNDERSTANDING ASSEMBLIES

The holding registers described above we will define as ***Module Register Addresses*** and are used for the function of the module regardless of if any remote PLC or PC is connected. There are many *Module Register Addresses* that are used for the operation of the module that are not applicable, required, or should even be made available to a remote PLC. When connecting to a PLC, the module needs to gather certain specific *Module Register Addresses* located scattered throughout the entire listing of *Module Register Addresses* into a concise contiguous group or ***Assembly*** that the PLC can efficiently read from and write to. Figure 2 illustrates this concept.

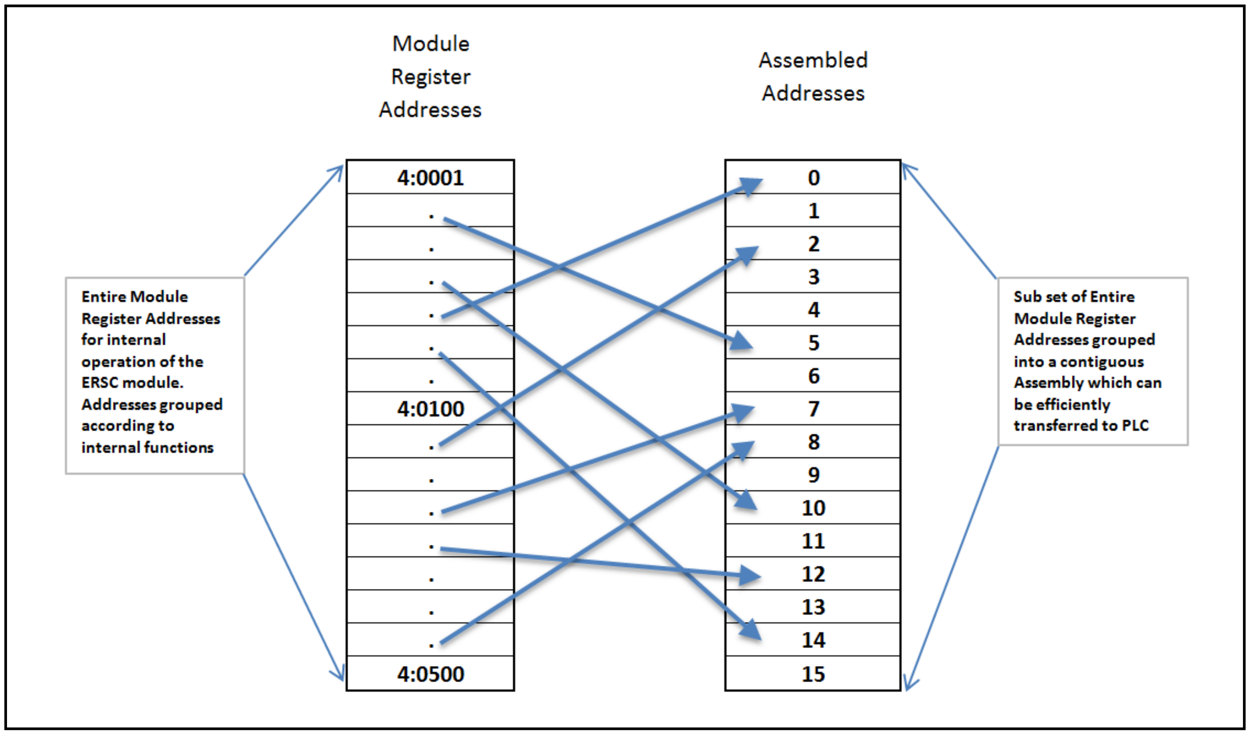

Module Register Addresses
Assembled Addresses
4:0001
4:0100
4:0500
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Entire Module Register Addresses for internal operation of the ERSC module. Addresses grouped according to internal functions
Sub set of Entire Module Register Addresses grouped into a contiguous Assembly which can be efficiently transferred to PLC

FIGURE 2 - PLC ASSEMBLY CONCEPT

So, no matter which of the three connection protocols you use (Modbus TCP, EIP, or Profinet I/O), for any given assembly, the internal data you are accessing with your PLC is the same.

Modbus Assembly Instance Structure

Each *ConveyLinx-Ai2* 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 *module*'s memory structure contains a fixed array of internal data locations that are constructed as Modbus *Holding Registers*. Each *module* 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 *module*'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 *ConveyLinx-Ai2* visible on its network and access Input and Output *Holding Register Assemblies*. The ERSC supports the following Modbus TCP Service Codes:

- Service Code 3 - Read Holding Register (up to 45 registers per instruction)
- Service Code 6 - Write Single Register
- Service Code 16 - Write Multiple Registers (up to 45 registers per instruction)
- Service Code 23 - Read/Write Multiple Registers (up to 45 registers per instruction)

When using Input and Output Assemblies with Modbus TCP PLC, it is important to always use the first address shown in the assembly group as the beginning register to read or write regardless of which register in the assembly is needed by the PLC. Trying to access Input or Output Assemblies starting with any register in the assembly other than the first register will cause the module to return an error.

For example, for ZPA Mode Assembly Inputs, if you only need to read register 4:1504; your I/O connection set-up in your PLC must use the starting address of 4:1500 and a sufficient length of registers to read (in this example at least 5 registers) in order to get to the desired register. If you set up your PLC to start reading at 4:1504, the module will return an error.

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

Please refer to *Appendix C – ODVA Compliant Registers for* **Ethernet I/P for controller requiring ODVA compliant assembly identification values.**

Ethernet I/P Assembly Instance Structure

When a *ConveyLinx-Ai2* module is attached to an external Ethernet I/P controller (Logix 5000 based PLC, for example), it is done so as a Generic Ethernet I/O device or by installing EDS file(s) provided by PulseRoller.com.

For this manual Ethernet I/P Instances are described using the Generic Ethernet I/O method of connection. Consult pulseroller.com for more information on obtaining EDS file(s) for incorporation into your RSLogix5000 development software.

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.

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 user-defined name of the device**
- **"O.Data" indicates data written by the PLC to the device**
- **"I.Data" indicates data read by the PLC from 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**

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

Please refer to Connecting ConveyLinx to Rockwell PLC's – Insight Automation publication ERSC-1520 for details on how to connect ConveyLinx to Rockwell Ethernet I/P capable PLCs.

PROFINET IO ASSEMBLY INSTANCE STRUCTURE

Prior to connecting any *modules* to a Profinet IO capable (Siemens, for example) PLC, a *GSDXML* configuration file needs to be installed in your PLC programming software (i.e. STEP 7 or TIA Portal). The *GSDXML* file contains, among other things, the definition of memory size required when connecting a *module* to a Siemens PLC with Profinet IO. When assigning an individual *module* to a PLC, you decide what memory address to use. The *GSDXML* file stipulates that each module requires 64 bytes of input data from the *module* to the PLC and 64 bytes of output data from the PLC to the *module*. When assigning an individual *module* to a PLC, you decide what physical PLC memory address to use as the beginning of these 64 byte blocks.

In all of the charts showing *module* data registers and their assigned function, for Profinet IO these are indicated as "Byte 0, Byte 1,...Byte 32", etc. These are offsets from the starting address in the Siemens PLC memory assigned by the Siemens programming software when the *module* was installed into the network. For example, if a *module's* inputs are installed beginning at %IB256, then from our charts, the data for "Byte 6 and Byte 7" would be at addresses %IB262 and %IB263 respectively.

Please refer to Connecting ConveyLinx to Profinet PLCs – Insight Automation publication ERSC-1525 for details on how to connect ConveyLinx to Profinet I/O capable PLCs.

PLC Controller with ZPA Mode

When a *ConveyLinx-Ai2* module is in ZPA mode, an external networked PLC or PC controller can connect to it and perform the following:

- Instruct either or both the upstream and downstream zone to accumulate the next Carton that arrives
- Receive indication that a new Carton has arrived at either zone
- Receive indication that a Carton has departed from either zone
- Read tracking data associated with Carton at accumulated zone
- Update tracking data associated with Carton at accumulated zone
- Instruct accumulated zone to release Carton and accumulate on next Carton 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
- Optionally take over control of Aux I/O on either or both Sensor Ports

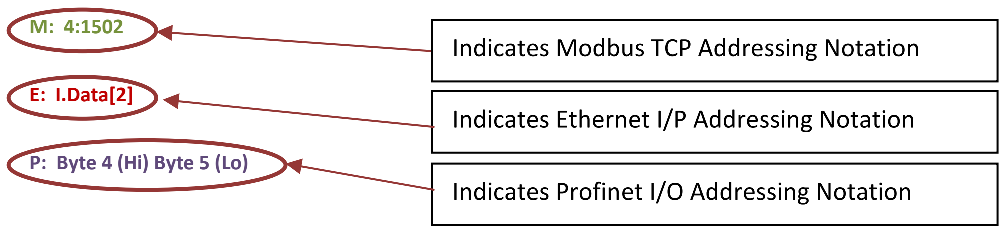
Notes About Assemblies for ConveyLinx-Ai2 in ZPA Mode

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

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 *module*'s physical "left" or "right" side's connections. For motor and port specific I/O items, register's description will explicitly indicate "left" or "right".

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

Register Chart Legend


M: 4:1502
E: I.Data[2]
P: Byte 4 (Hi) Byte 5 (Lo)
Indicates Modbus TCP Addressing Notation
Indicates Ethernet I/P Addressing Notation
Indicates Profinet I/O Addressing Notation

ZPA Mode Assembly Inputs for PLC

This chart shows the data sent to the PLC from the module when connected. All registers in the module are 16-bit integer data type.

Register Name / Module Address	*Assembled Address for PLC*	*Description*
Local Status Upstream Zone Forward Direction 4:0116	**M: 4:1500 (Low Byte)** **E: I.Data[0] (Low Byte)** **P: Byte 1**	Unsigned Value of Byte 0x01 = Zone sensor clear and motor stopped 0x02 = Zone sensor clear, motor running, accepting from upstream zone 0x04 = Zone sensor blocked, motor running, discharging to downstream zone 0x05 = Zone sensor blocked and motor stopped 0x06 = Busy (state during ConveyStop active mode) See *ZPA Mode Note* ① on page 24 for further details
Local Status Upstream Zone Reverse Direction 4:0116	**M: 4:1500 (High Byte)** **E: I.Data[0] (High Byte)** **P: Byte 0**	
Local Status Downstream Zone Forward Direction 4:0196	**M: 4:1501 (Low Byte)** **E: I.Data[1] (Low Byte)** **P: Byte 3**	
Local Status Downstream Zone Reverse Direction 4:0196	**M: 4:1501 (High Byte)** **E: I.Data[1] (High Byte)** **P: Byte 2**	
Arrival Count Local Upstream Zone 4:0106	**M: 4:1502** **E: I.Data[2]** **P: Byte 4 (Hi) Byte 5 (Lo)**	Unsigned Integer Value • Increments by 1 each time a Carton arrives in the zone • Value rolls over from 65,535 back to 0
Departure Count Local Upstream Zone 4:0107	**M: 4:1503** **E: I.Data[3]** **P: Byte 6 (Hi) Byte 7 (Lo)**	Unsigned Integer Value • Increments by 1 each time a Carton departs the zone • Value rolls over from 65,535 back to 0
Arrival Count Local Downstream Zone 4:0186	**M: 4:1504** **E: I.Data[4]** **P: Byte 8 (Hi) Byte 9 (Lo)**	Unsigned Integer Value • Increments by 1 each time a Carton arrives in the zone • Value rolls over from 65,535 back to 0
Departure Count Local Downstream Zone 4:0187	**M: 4:1505** **E: I.Data[5]** **P: Byte 10 (Hi) Byte 11 (Lo)**	Unsigned Integer Value • Increments by 1 each time a Carton departs the zone • Value rolls over from 65,535 back to 0
Module Status Word 1 4:0088	**M: 4:1506** **E: I.Data[6]** **P: Byte 12 (Hi) Byte 13 (Lo)**	Bitwise Value - Read only bit 0 = Module Reset Flag – 1 when module resets, 0 when PLC connected bit 1 = Reserved bit 2 = Over-Voltage – Module power supply has exceeded 30V bit 3 = Left Motor Error – bits 7 thru 15 indicate specific 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 105°C bit 9 = Left Motor at Max. Torque bit 10 = Left Motor Short Circuit bit 11 = Left Motor Not Connected bit 12 = Left Motor Overload – Motor has been stalled for more than 20 seconds bit 13 = Left Motor Stalled – Motor running slower than 10% of selected speed bit 14 = Left Motor Hall Sensor Error bit 15 = Left Motor Not Used

Register Name / Module Address	*Assembled Address for PLC*	*Description*
Module Status Word 2 **4:0089**	**M: 4:1507** **E: I.Data[7]** **P: Byte 14 (Hi) Byte 15 (Lo)**	Bitwise Value - Read only bit 0 = Reserved bit 1 = Reserved bit 2 = Over-Voltage – Motor power supply has exceeded 30V bit 3 = Right Motor Error bit 4 = Reserved bit 5 = Downstream Jam Error bit 6 = Right Sensor Error bit 7 = Low Voltage Error - Motor Power Supply less than 18V bit 8 = Right Motor Over-heated - Calculated temperature over 105°C bit 9 = Right Motor at Max. Torque bit 10 = Right Motor Short Circuit bit 11 = Right Motor Not Connected bit 12 = Right Motor Overload – Motor has been stalled for more than 20 seconds bit 13 = Right Motor Stalled – Motor running slower than 10% of selected speed bit 14 = Right Motor Hall Sensor Error bit 15 = Right Motor Not Used
Current Upstream Zone Tracking Word 1 **4:0119**	**M: 4:1508** **E: I.Data[8]** **P: Byte 16 (Hi) Byte 17 (Lo)**	When Carton is accumulated in UPSTREAM zone: Value = Tracking data word #1 (16-bit integer) for the Carton currently accumulated and stopped in the module's Upstream zone.
Current Upstream Zone Tracking Word 2 **4:0120**	**M: 4:1509** **E: I.Data[9]** **P: Byte 18 (Hi) Byte 19 (Lo)**	When Carton is accumulated in UPSTREAM zone: Value = Tracking data word #2 (16-bit integer) for the Carton currently accumulated and stopped in the module's Upstream zone.
Current Downstream Zone Tracking Word 1 **4:0199**	**M: 4:1510** **E: I.Data[10]** **P: Byte 20 (Hi) Byte 21 (Lo)**	When Carton is accumulated in DOWNSTREAM zone: Value = Tracking data word #1 (16-bit integer) for the Carton currently accumulated and stopped in the module's Downstream zone.
Current Downstream Zone Tracking Word 2 **4:0200**	**M: 4:1511** **E: I.Data[11]** **P: Byte 22 (Hi) Byte 23 (Lo)**	When Carton is accumulated in DOWNSTREAM zone: Value = Tracking data word #2 (16-bit integer) for the Carton currently accumulated and stopped in the module's Downstream zone.
Current Release Count for Upstream Zone **4:0105**	**M: 4:1512** **E: I.Data[12]** **P: Byte 24 (Hi) Byte 25 (Lo)**	Copy of the current value in the *Release Upstream* output register which can be used by PLC logic to confirm release count prior to writing new data to the *Release Upstream* output register
Current Release Count for Downstream Zone **4:0185**	**M: 4:1513** **E: I.Data[13]** **P: Byte 26 (Hi) Byte 27 (Lo)**	Copy of the current value in *Release Downstream* which can be used by PLC logic to confirm release count prior to writing new data to the output *Release Downstream* register
Get Tracking Forward Direction Word 1 **4:0201**	**M: 4:1514** **E: I.Data[14]** **P: Byte 28 (Hi) Byte 29 (Lo)**	When module is discharging to Non-ConveyLinx controlled conveyor: Value = Tracking data word #1 (16-bit integer) for the Carton that has just discharged from the local downstream zone when local conveyor is operating in default or "forward" direction.
Get Tracking Forward Direction Word 2 **4:0202**	**M: 4:1515** **E: I.Data[15]** **P: Byte 30 (Hi) Byte 31 (Lo)**	When module is discharging to Non-ConveyLinx controlled conveyor: Value = Tracking data word #2 (16-bit integer) for the Carton that has just discharged from the local downstream zone when local conveyor is operating in default or "forward" direction.
Get Tracking Reverse Direction Word 1 **4:0121**	**M: 4:1516** **E: I.Data[16]** **P: Byte 32 (Hi) Byte 33 (Lo)**	When module is discharging to Non-ConveyLinx controlled conveyor: Value = Tracking data word #1 (16-bit integer) for the load that has just discharged from the local downstream zone when local conveyor is operating in opposite of default or "reverse" direction.

Register Name / Module Address	*Assembled Address for PLC*	*Description*
Get Tracking Reverse Direction Word 2 4:0122	**M: 4:1517** **E: I.Data[17]** **P: Byte 34 (Hi) Byte 35 (Lo)**	When module is discharging to Non-ConveyLinx controlled conveyor: Value = Tracking data word #2 (16-bit integer) for the Carton that has just discharged from the local downstream zone when local conveyor is operating in opposite of default or "reverse" direction.
Sensor Port Inputs 4:0035	**M: 4:1518** **E: I.Data[18]** **P: Byte 36 (Hi) Byte 37 (Lo)**	Bitwise Value - Read only bit 00 = Left Sensor Port – Aux I/O (M8 Pin2) bit 02 = Right Sensor Port – Aux I/O (M8 Pin2) bit 04 = Left Sensor Port – Signal (M8 Pin4) bit 06 = Right Sensor Port - Signal (M8 Pin4) bit 15 = 2 sec on / 2 sec off heartbeat All other bits reserved See *ZPA Mode Note* ② on page 25
Reserved	**M: 4:1519** **E: I.Data[19]** **P: Byte 38 (Hi) Byte 38 (Lo)**	
ConveyStop Status 4:0019	**M: 4:1520** **E: I.Data[20]** **P: Byte 40 (Hi) Byte 41 (Lo)**	Bitwise Value - Read only bit 00 – bit04 = Reserved bit 05 = Stop active on another module in Stop Group bit 06 = Stop active due to lost communication connection bit 07 = Stop active due to lost PLC connection bit 08 = Reserved bit 09 = Reserved bit 10 = Stop active due to Stop Command from PLC bit 11- bit 15 = Reserved Refer to Insight Automation publication ERSC-1800 *ConveyStop User's Guide* for details on using *ConveyStop*.

ZPA Mode Note ①

The values 0xXX01 thru 0xXX06 are shown because these are the possible logical values used for inter-module communication. External networked devices (PLC or PC) monitoring these registers may; depending on their scan rate, not actually see each of these values change in sequence as a Carton 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 used for zone status for reversing conveyor applications and MAY CONTAIN DATA. PLC/PC programmers working with single direction conveyor applications MUST MASK THE HIGH BYTE or otherwise ignore the high byte in processing status data from these registers.

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 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.

ZPA Mode Note ②

The values for the signals on the Sensor port's bit 4 and bit 6 are determined by the module's initial Auto-Configuration results and the bit values are set to a 1 when sensor is blocked and 0 when sensor is clear regardless of the sensor type used. For example, each Sensor port Pin 4 signal is for the sensor's output. If the sensor is light energized, Normally Open (N.O.) then the electrical signal on Pin 4 is ON when the sensor is clear and OFF when the sensor is blocked. However, when reading bit 4 or bit 6 in this register, the bit will be a 1 when the sensor is blocked and 0 when the sensor is clear.

ZPA Mode Assembly Outputs for PLC

These registers are written by the PLC to the *module* when the module is in ZPA mode. All registers are 16-bit Integer data type.

Register Name / Module Address	***Assembled Address for PLC***	***Description***
Set Local Upstream Zone Tracking Word 1 **4:0132**	**M: 4:1600** **E: O.Data[0]** **P: Byte 0 (Hi) Byte 1 (Lo)**	Write value for 16-bit integer tracking data word #1 for the Carton accumulated in the Upstream Zone. See *ZPA Mode Note* ③ on page 28 for special reserved values
Set Local Upstream Zone Tracking Word 2 **4:0133**	**M: 4:1601** **E: O.Data[1]** **P: Byte 2 (Hi) Byte 3 (Lo)**	Write value for 16-bit integer tracking data word #2 for the Carton accumulated in the Upstream Zone. See *ZPA Mode Note* ③ on page 28 for special reserved values
Set Local Downstream Zone Tracking Word 1 **4:0212**	**M: 4:1602** **E: O.Data[2]** **P: Byte 4 (Hi) Byte 5 (Lo)**	Write value for 16-bit integer tracking data word #1 for the Carton accumulated in the Downstream Zone. See *ZPA Mode Note* ③ on page 28 for special reserved values
Set Local Downstream Zone Tracking Word 2 **4:0213**	**M: 4:1603** **E: O.Data[3]** **P: Byte 6 (Hi) Byte 7 (Lo)**	Write value for 16-bit integer tracking data word #2 for the Carton accumulated in the Downstream Zone. See *ZPA Mode Note* ③ on page 28 for special reserved values
Accumulation Control for Local Upstream Zone **4:0104**	**M: 4:1604** **E: O.Data[4]** **P: Byte 8 (Hi) Byte 9 (Lo)**	<u>Bitwise Values</u> bit 0 = Set/Clear Accumulation Mode for Local Zone bit 8 = Accumulate adjacent upstream zone bit 9 = Set Arrival Confirmation for adjacent downstream zone bit 10 = Jog zone in default direction bit 11 = Jog zone in opposite of default direction bit 12 = Wake up Local Zone bit 13 = Enable Maintenance Mode All other bits reserved See *ZPA Mode Note* ⑤ on page 29 for further details
Accumulation Control for Local Downstream Zone **4:0184**	**M: 4:1605** **E: O.Data[5]** **P: Byte 10 (Hi) Byte 11 (Lo)**	
Set Left MDR Speed **4:0040**	**M: 4:1606** **E: O.Data[6]** **P: Byte 12 (Hi) Byte 13 (Lo)**	Value in mm/sec for MDR or RPM x 10 for PGD Range: depends upon the Ai MDR or PGD connected *MDR Example: 400 = 0.40 m/s* *PGD Example: 400 = 40 RPM* 0 = Remain at last non zero value entered See *ZPA Mode Note* ④ on page 29 for further details
Set Right MDR Speed **4:0064**	**M: 4:1607** **E: O.Data[7]** **P: Byte 14 (Hi) Byte 15 (Lo)**	
Release and Accumulate on Next Arrival for Local Upstream Zone **4:0105**	**M: 4:1608** **E: O.Data[8]** **P: Byte 16 (Hi) Byte 17 (Lo)**	<u>When bit 0 of *Accumulation Control for Local Upstream Zone* is set:</u> Changing the value in this register will cause the Carton accumulated in this zone to release and the zone will be armed to automatically accumulate the next Carton that arrives
Release and Accumulate on Next Arrival for Local Downstream Zone **4:0185**	**M: 4:1609** **E: O.Data[9]** **P: Byte 18 (Hi) Byte 19 (Lo)**	<u>When bit 0 of *Accumulation Control for Local Downstream Zone* is set:</u> Changing the value in this register will cause the Carton accumulated in this zone to release and the zone will be armed to automatically accumulate the next Carton that arrives

Register Name / Module Address	***Assembled Address for PLC***	***Description***
Set Status for Upstream Induct 4:0134	**M: 4:1610** **E: O.Data[10]** **P: Byte 20 (Hi) Byte 21 (Lo)**	Only used when local module is accepting loads from Non-ConveyLinx controlled conveyor: Set value to 4 to cause the local upstream zone to run to accept the Carton being delivered by the non-ConveyLinx controlled conveyor. Set value to 1 to cause the local upstream zone to accept the tracking data written in *Set Induct Tracking Word 1 / Word 2*
Set Status for Downstream Discharge 4:0232	**M: 4:1611** **E: O.Data[11]** **P: Byte 22 (Hi) Byte 23 (Lo)**	Only used when local module is discharging loads to Non-ConveyLinx controlled conveyor: • Set value to 5 to cause the local downstream zone to accumulate and hold any Carton that arrives. • Set value to 1 to allow the local downstream zone to release the Carton
Set Induct Tracking Forward Direction Word 1 4:0139	**M: 4:1612** **E: O.Data[12]** **P: Byte 24 (Hi) Byte 25 (Lo)**	Only used when local module is accepting loads from Non-ConveyLinx controlled conveyor: Set value for 16-bit integer tracking word #1 for Carton leaving non-ConveyLinx controlled conveyor that is in transit to arrive on the local upstream zone when conveyor is running in default or “forward” direction.
Set Induct Tracking Forward Direction Word 2 4:0140	**M: 4:1613** **E: O.Data[13]** **P: Byte 26 (Hi) Byte 27 (Lo)**	Only used when local module is accepting loads from Non-ConveyLinx controlled conveyor: Set value for 16-bit integer tracking word #2 for Carton leaving non-ConveyLinx controlled conveyor that is in transit to arrive on the local upstream zone when conveyor is running in default or “forward” direction.
Set Induct Tracking Reverse Direction Word 1 4:0237	**M: 4:1614** **E: O.Data[14]** **P: Byte 28 (Hi) Byte 29 (Lo)**	Only used when local module is accepting loads from Non-ConveyLinx controlled conveyor: Set value for 16-bit integer tracking word #1 for Carton leaving non-ConveyLinx controlled conveyor that is in transit to arrive on the local upstream zone when conveyor is running in opposite of default or “reverse” direction.
Set Induct Tracking Forward Direction Word 2 4:0238	**M: 4:1615** **E: O.Data[15]** **P: Byte 30 (Hi) Byte 31 (Lo)**	Only used when local module is accepting loads from Non-ConveyLinx controlled conveyor: Set value for 16-bit integer tracking word #2 for Carton leaving non-ConveyLinx controlled conveyor that is in transit to arrive on the local upstream zone when conveyor is running in opposite of default or “reverse” direction.
Clear Motor Error 4:0022	**M: 4:1616** **E: O.Data[16]** **P: Byte 32 (Hi) Byte 33 (Lo)**	Logical 0 or 1 0 = Stop Reset 1 = Send Reset See *ZPA Mode Note* ⑥ on page 30 for further details
Set Aux I/O Outputs 4:0063	**M: 4:1617** **E: O.Data[17]** **P: Byte 34 (Hi) Byte 35 (Lo)**	Bitwise Value bit 8 = Left Aux I/O (Pin 2) Usage: 0 = Use as Input / 1 = Use as Output bit 9 = Right Aux I/O (Pin 2) Usage: 0 = Use as Input / 1 = Use as Output bit 12 = Left Aux I/O (Pin 2) as Output: 1 = ON, 0 = OFF (only if bit 8 is ON) bit 13 = Right Aux I/O (Pin 2) as Output: 1 = ON, 0 = OFF (only if bit 9 is ON) All other bits reserved See *ZPA Mode Note* ⑦ on page 30 for further details
Reserved	**M: 4:1618** **E: O.Data[18]** **P: Byte 36 (Hi) Byte 37 (Lo)**	

Register Name / Module Address	*Assembled Address for PLC*	*Description*
ConveyStop Command 4:0020	**M: 4:1619** **E: O.Data[19]** **P: Byte 38 (Hi) Byte 39 (Lo)**	Integer Value 0 = No Command 1 = Command local module's Stop Group to go to Stopped State 2 = Command local module's Stop Group to Clear Stopped State Refer to Insight Automation publication ERSC-1800 *ConveyStop User's Guide* for details on using *ConveyStop*.
Clear Sensor Jam Command for Local Upstream Zone 4:0109	**M: 4:1620** **E: O.Data[20]** **P: Byte 40 (Hi) Byte 41 (Lo)**	Use when PLC has detected a local Upstream jam in Module Status Word #1 Bit 5: PLC creates transition from 0 to 1 to send command to local upstream zone to clear the jam condition. See *ZPA Mode Note* ⑧ on page 31
Clear Sensor Jam Command for Local Downstream Zone 4:0189	**M: 4:1621** **E: O.Data[21]** **P: Byte 42 (Hi) Byte 43 (Lo)**	Use when PLC has detected a local Downstream jam in Module Status Word #2 Bit 5: PLC creates transition from 0 to 1 to send command to local downstream zone to clear the jam condition. See *ZPA Mode Note* ⑧ on page 31
Direction & Accumulation Mode Control for Local Upstream Zone 4:0365	**M: 4:1622** **E: O.Data[22]** **P: Byte 44 (Hi) Byte 45 (Lo)**	Used to change direction of flow or accumulation mode for a contiguous group of zones beginning with the local upstream / downstream zone Value for Low Byte of Register 0 = Normal Function 1 = Accumulate Zones 2 = Accumulate Zones 3 = Change Accumulation Release mode 4 = Return Release Mode to Configured Default 10 = Set Direction to Configured Default Direction (Forward) 11 = Set Direction to opposite of Configured Direction (Reverse) Value for High Byte of Register Number of ZONES beginning with the local Upstream / Downstream Zone for which the Low Byte value is applied – from 1 to 220. If ALL ZONES in the subnet need to be controlled then leave the High Byte = "0".
Direction & Accumulation Mode Control for Local Downstream Zone 4:0375	**M: 4:1623** **E: O.Data[23]** **P: Byte 46 (Hi) Byte 47 (Lo)**	
ConveyMerge Interface 4:0387	**M: 4:1624** **E: O.Data[24]** **P: Byte 48 (Hi) Byte 49 (Lo)**	Used to dynamically modify a *ConveyMerge* configuration already established with *EasyRoll*. Bitwise values: Bit 15: set to enable PLC modification Bit 4: set to disable release from Center line Bit 5: set to disable release from Left line Bit 6: set to disable release from Right line Bits 0 thru 3 are interpreted as a numerical value to change the release priority: Value: 0 = First come, first served release 1 = Center line has priority 2 = Left line has priority 3 = Right line has priority Please refer to publication ERSC-1002 *User's Guide Supplement for ConveyMerge* for details on configuring merging sections with *EasyRoll* software.

ZPA Mode Note ③

Because the *module* is connected as I/O, the PLC inherently is always trying to update the Output image on (at least) RPI intervals. In order to prevent the PLC from inadvertently overwriting the "real" tracking data registers; the

Assembly Output implementation utilizes the holding register locations shown and automatically updates the "real" tracking registers with this new data only upon release of the Carton 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 *module* to not modify the existing "real" tracking data and allow it to continue downstream "as-is" when the Carton is released.
- Set both tracking registers shown to 0xFFFF: This will instruct the *module* to clear the "real" tracking data and when the Carton is released, the "real" tracking data will be "0" in both registers.

Both word 1 and word 2 of the tracking data in question must have 0xFFFF written to it in order to signal the module to clear the tracking data. If only one word has 0xFFFF written to it and the other does not, the value of 0xFFFF will be the new tracking data for that word.

ZPA Mode Note ④

Leaving these registers at "0" will instruct the *module* to use its configured speed. Any non-zero value will instruct the *module* to use this non-zero value as the speed reference as long as the value is within the maximum and minimum limits as defined for the connected motor. If the value is higher than the maximum limit, the motor's speed will be set to the maximum allowed speed. Similarly, if the value provided is lower than the minimum limit, the motor's speed will be set to the minimum allowed speed. 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 *module* continues to use the last non zero value it was given. Please note that setting this value to "0" will not cause the motor to run at "0" speed.

ZPA Mode Note ⑤

Bit 8 – Accumulate Adjacent Upstream Zone

Setting this bit will cause the next upstream zone of the local module to accumulate. This next upstream zone can be either on the local module or the downstream zone of the adjacent upstream module.

Bit 9 – Set Arrival Confirmation for Downstream Zone

By default for ZPA operation, *ConveyLinx* requires a confirmation from the downstream zone when a Carton is discharged. Without this confirmation, the releasing zone will detect a jam condition. This bit is used in applications where the Carton is removed from the conveyor (either manually or say by a PLC controlled external mechanism such as a pusher or diverter) and the PLC needs to "confirm" the removal of the Carton in order to satisfy the ZPA confirmation logic.

Bits 10 & 11 – Jog Controls

These bits can be used by the PLC to jog the local zone for specialized applications when local movement of the Carton on a zone is required. An example would be once a Carton has arrived in the local zone, the PLC determines that the Carton needs to be repositioned or perhaps squared up against a PLC controlled pop-up stop.

Jog control bits DO over-ride ZPA logic control and should be used with caution! Improper usage of jog controls can produce unexpected results and/or damage to product and equipment.

Bit 12 – Wake up Local Zone

Setting this bit will cause the local zone to "wake up" and run to accept a carton the same as if its upstream module had written a status value of "4". This function would be useful for a merge onto a main line of ZPA conveyor.

Bit 13 – Enable Maintenance Mode

Setting this bit will place the local zone in maintenance mode. In this mode the motor will not run regardless of zone conditions. The zone upstream of this local zone will receive a "busy" status to inhibit release of any item into this local zone. While in this state, the Sensor and Motor LEDs will flash on and off in green color.

ZPA Mode Note ⑥

Errors deemed "fatal" for the module (motor short circuit and Hall Effect sensor fault) require either removal of power to reset or remote reset by PLC. Setting bit 0 of this register to 1 will initiate this remote error reset from the PLC to the local module. Setting this bit will reset a fatal error on either (or both) the left or right MDR.

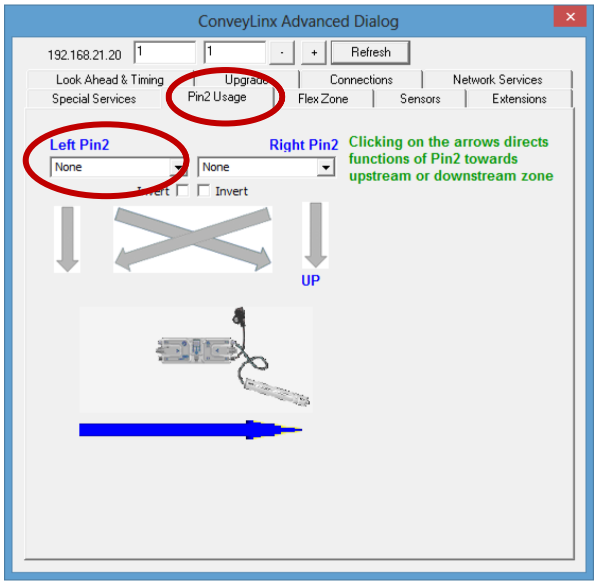
External controller must continuously set bit 0 = 1 in the *Clear Motor Error* register for at least 500 msec for the module to recognize the reset command.

ZPA Mode Note ⑦

In certain applications, it may be desired to have the PLC actuate a device along the conveyor (i.e. illuminate a light or energize a solenoid coil, etc.). With the *ConveyLinx* network this can be done without installing a separate PLC I/O system to do this.

By default in ZPA mode, the PLC does not have control of the local connected module's Aux I/O (Pin 2) signals. The function of the module's Aux I/O signal is set in *EasyRoll*. In order for the PLC to use an Aux I/O Pin2 as an output, the setting in EasyRoll for the particular Aux I/O Pin 2 must be set to "None" in order for the bit wise settings in this register to be recognized.

To use the Left or Right Pin 2 signals with a PLC when the module is in ZPA mode, you first have to select “None” from the drop-down box on the “Pin 2 Usage” tab in the Advance Dialog from EasyRoll. Once this has been set, then the bit-wise functionality in the *Set Aux I/O Outputs* register is enabled.



Please refer to the ConveyLinx-Ai2 User’s Guide for electrical connection details before using PLC control for Aux I/O Pin outputs.

ZPA Mode Note ⑧

Refer to *ConveyLinx Ai2 User’s Guide* (publication ERSC-1006) and/or *EasyRoll* software on screen help for description of various jam conditions. These registers are applicable only to a Sensor Jam condition. By default in ZPA mode, the module will attempt 3 times to automatically clear a sensor jam and if the sensor is still blocked, the module will stop the zone. To reset this condition the Carton must be manually removed after the 3rd attempt and the sensor cleared before the zone will return to normal operation.

These registers allow the PLC to remotely attempt another auto-clear cycle in an attempt to clear the jam condition. Please note that this function requires the PLC to make a transition from 0 to 1 in the register to initiate another auto-clear cycle. Holding the value to 1 will not cause the retry to continue indefinitely. Each attempt requires a new transition from 0 to 1.

ZPA Mode Examples

Example 1 – Basic Accumulate and Release with Tracking Data

Figure 3 shows a typical arrangement of an upstream or downstream zone on a module in ZPA mode that is not the most upstream or most downstream zone in a given network. This example will show how to cause a Carton to accumulate, how to detect a Carton has arrived, how to write tracking data, and finally how to release the Carton.

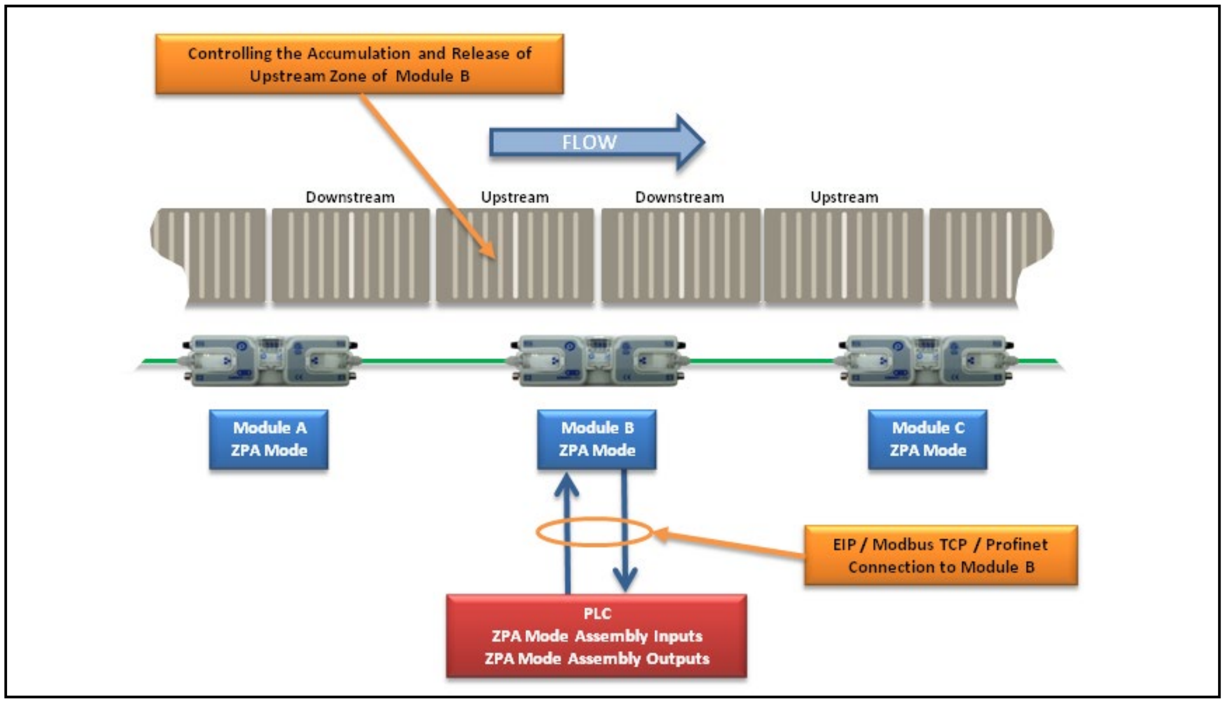

Controlling the Accumulation and Release of Upstream Zone of Module B
FLOW
Downstream
Upstream
Downstream
Upstream
Module A ZPA Mode
Module B ZPA Mode
Module C ZPA Mode
EIP / Modbus TCP / Profinet Connection to Module B
PLC ZPA Mode Assembly Inputs ZPA Mode Assembly Outputs

FIGURE 3 - BASIC ACCUMULATE / RELEASE CONTROL EXAMPLE

For this example, the PLC must establish ZPA Mode Assembly Input/Output connections to *Module B* as shown in Figure 3.

Upstream Zone Example

First, let's assume we want to accumulate any Carton that arrives on the upstream zone of *Module B*. With the PLC:

1. Set bit 0 in *Accumulation Control for Local Upstream Zone* register to instruct this zone to accumulate any Carton that arrives.
2. Monitor *Arrival Count Local Upstream Zone* and *Departure Count Local Upstream Zone* registers. On the leading edge when these two values become not equal, the PLC knows there is a new arrival. Note that as long as a Carton is physically occupying the upstream zone, these two values will not be equal.
3. Upon a successful arrival of a Carton in the upstream zone, then tracking data in *Current Upstream Zone Tracking Word 1* and *Current Upstream Zone Tracking Word 2* will be valid for the newly arrived Carton.
4. The PLC may then decide that this tracking data is to be updated. The PLC can then write new tracking data to registers *Set Local Upstream Zone Tracking Word 1* and *Set Local Upstream Zone Tracking Word 2.*

5. When the PLC is ready to release the Carton in the upstream zone, it should read the value in the *Current Release Count for Upstream Zone* register, add 1 to this value, and then write this new value to *Release and Accumulate on Next Arrival for Local Upstream Zone* register. When *Module B* sees this new value in this register, it will release the Carton in the upstream zone and automatically accumulate the next new Carton that arrives. Please note that if the downstream conditions from *Module B* are full when this new value is written, *Module B* will remember that it was instructed to release and will release the Carton when downstream conditions become clear without any further signal from the PLC.
6. The PLC can detect when the Carton has departed the sensor on *Module B* upstream zone by examining the values in *Arrival Count Local Upstream Zone* and *Departure Count Local Upstream Zone* registers. On the leading edge of when these two values are equal, the PLC will know that the Carton has departed the zone sensor in *Module B* upstream zone.

<u>If the PLC wants to cancel the accumulation control for *Module B* upstream zone</u>:

PLC can reset bit 0 in *Accumulation Control for Local Upstream Zone* register. This will signal *Module B* to release any Carton accumulated and not accumulate the next Carton that arrives at *Module B* upstream zone.

Please note that if any new tracking data has been written to *Set Local Upstream Zone Tracking Word 1* and/or *Set Local Upstream Zone Tracking Word 2* and accumulation control is then canceled by resetting bit 0, this data will NOT be assigned to the Carton when it is released. The <u>ONLY</u> way to pass tracking data to a Carton is by following Step 5 above.

Downstream Zone Example

Downstream Zone example is identical to Upstream Zone example shown above except the registers used are different. It is repeated below to explicitly show which registers to use.

First, let's assume we want to accumulate any Carton that arrives on the downstream zone of *Module B*. With the PLC:

1. Set bit 0 in *Accumulation Control for Local Downstream Zone* register to instruct this zone to accumulate any Carton that arrives.
2. Monitor *Arrival Count Local Downstream Zone* and *Departure Count Local Downstream Zone* registers. On the leading edge when these two values become not equal, the PLC knows there is a new arrival. Note that as long as a Carton is physically occupying the upstream zone, these two values will not be equal.
3. Upon a successful arrival of a Carton in the upstream zone, then tracking data in *Current Downstream Zone Tracking Word 1* and *Current Downstream Zone Tracking Word 2* will be valid for the newly arrived Carton.
4. The PLC may then decide that this tracking data is to be updated. The PLC can then write new tracking data to registers *Set Local Downstream Zone Tracking Word 1* and *Set Local Downstream Zone Tracking Word 2.*
5. When the PLC is ready to release the Carton in the upstream zone, it should read the value in the *Current Release Count for Downstream Zone* register, add 1 to this value, and then write this new value to *Release*

and Accumulate on Next Arrival for Local Downstream Zone register. When *Module B* sees this new value in this register, it will release the Carton in the downstream zone and automatically accumulate the next new Carton that arrives. Please note that if the downstream conditions from *Module B* are full when this new value is written, *Module B* will remember that it was instructed to release and will release the Carton when downstream conditions become clear without any further signal from the PLC.

6. The PLC can detect when the Carton has departed the sensor on *Module B* upstream zone by examining the values in *Arrival Count Local Downstream Zone* and *Departure Count Local Downstream Zone* registers. On the leading edge of when these two values are equal, the PLC will know that the Carton has departed the zone sensor in *Module B* downstream zone.

If the PLC wants to cancel the accumulation control for *Module B* downstream zone:

PLC can reset bit 0 in *Accumulation Control for Local Downstream Zone* register. This will signal *Module B* to release any Carton accumulated and not accumulate the next Carton that arrives at *Module B* downstream zone.

Please note that if any new tracking data has been written to *Set Local Downstream Zone Tracking Word 1* and/or *Set Local Downstream Zone Tracking Word 2* and accumulation control is then canceled; this data will NOT be assigned to the Carton when it is released. The ONLY way to pass tracking data to a Carton is by following Step 5 above.

Example 2 – Basic Bar Code Reader

This example shows how to easily set up the conveyor control to easily singulate cartons through a bar code scanning region. *Figure 4* shows a conveyor and *module* configuration for a simple barcode scanning example.

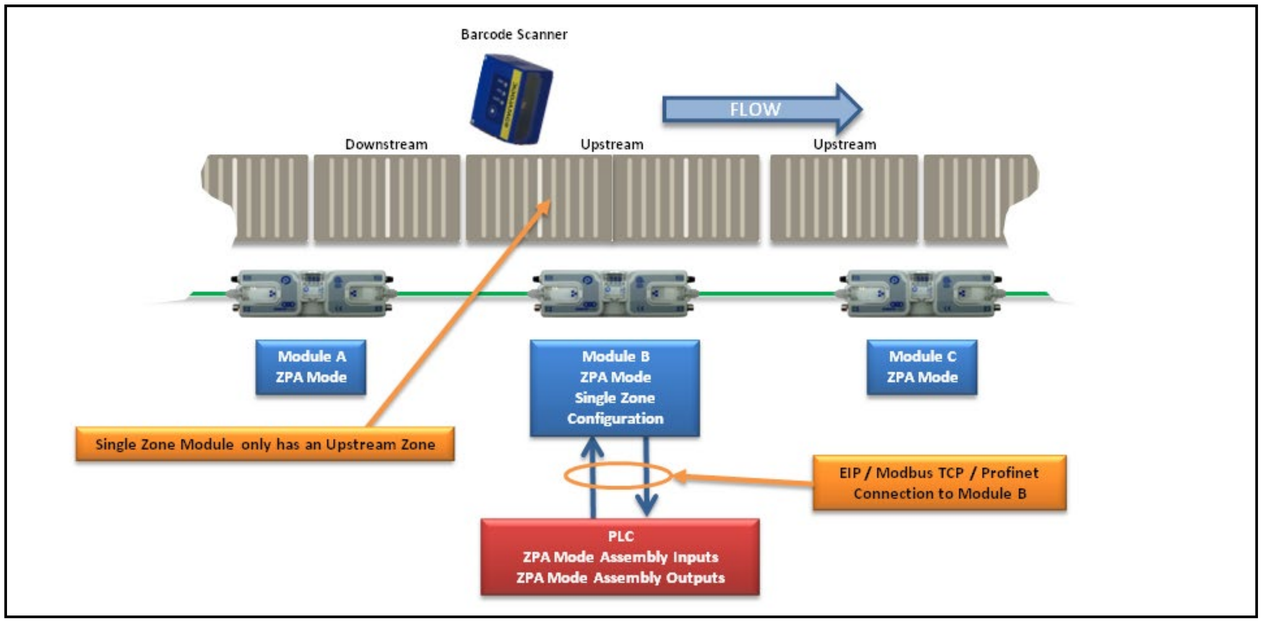


FIGURE 4 - SIMPLE BARCODE SCANNER APPLICATION

In this example, *Module B* must be logically configured as a single zone. This can be either a single long zone with one MDR and one sensor, or a single long zone with 2 MDRs and one sensor. Please refer to the *ConveyLinx User's Guide* for details on single zone operation.

Also, for this example to work properly; the discharging zone of *Module A* and the single zone of *Module B* must be in the default singluation release mode (not Train mode). *Module A* must also not have its Arrival Jam disabled from *EasyRoll* main screen. Please refer to the *ConveyLinx User's Guide* and/or *EasyRoll* software on screen help for details on these items.

Because of built-in ZPA functionality, when a Carton leaves *Module A* discharge zone, *Module A* waits until it receives confirmation from *Module B* that the Carton arrived. If a new Carton arrives at *Module A*, it will accumulate until this arrival confirmation occurs. Because *Module B* is a single long zone, the space for what would have been the upstream zone for *Module B* (if it was configured as two zones) will now essentially be left clear when a Carton is accumulated on *Module B*. This is the area where the bar code scanner is located. In this configuration, any new arrival at *Module B* will be assured to be the Carton associated with the last scan from the bar code reader. For this example, the PLC must establish a connection with *Module B* and then simply follow the zone control as described for basic accumulate and release.

EXAMPLE 3 – UPSTREAM ACCEPT INTERFACE

This example describes how to use a PLC to control the "wake-up" and passing of tracking data to the most upstream zone of a ConveyLinx controlled conveyor. *Figure 5* shows a typical example configuration.

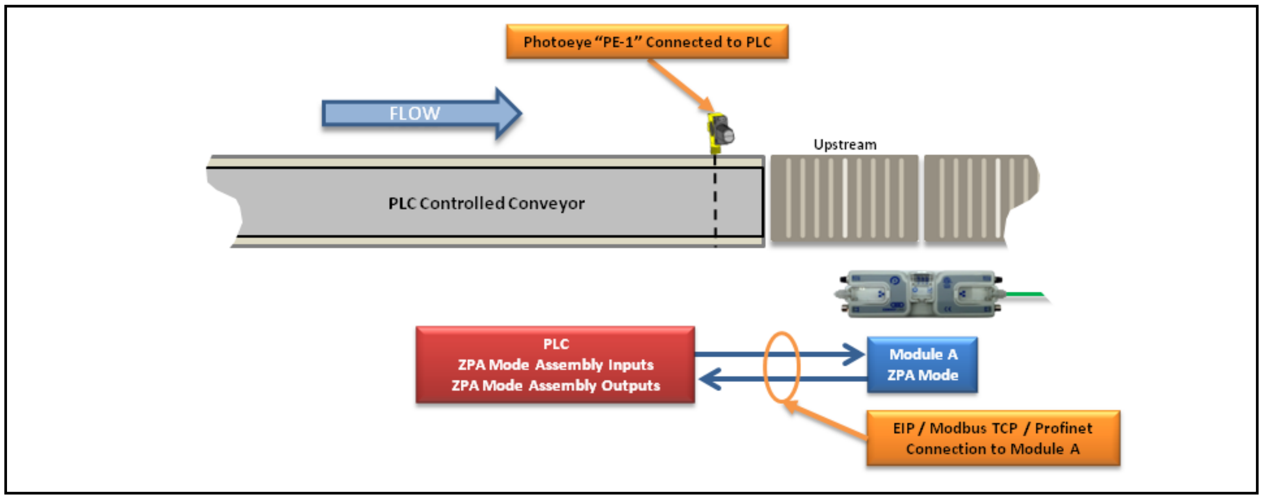


FIGURE 5 - UPSTREAM ACCEPT INTERLOCK EXAMPLE

For this example, the PLC must establish a connection with *Module A*. When PLC is ready to transfer a Carton from the PLC controlled conveyor to the Upstream Zone of the MDR Conveyor, the PLC logic must:

1. Write a "4" into *Set Status for Upstream Induct* register. This will cause the upstream zone of *Module A* to run to accept Carton.
2. When Carton clears *PE-1*, write tracking data to *Set Induct Tracking Forward Direction Word 1 / 2*.
3. When Carton clears *PE-1,* write a "1" to *Set Status for Upstream Induct* register. When *Module A* sees this register change to "1", it knows that the tracking data written to *Set Induct Tracking Forward Word 1 / 2* is valid and when the Carton reaches the sensor in its upstream zone, the tracking data will be correctly inducted.

Module A **must see the *Set Status for Upstream Induct* register change from 4 to 1 in order to recognize the tracking data written to *Set Induct Tracking Forward Word* registers. If using the clearing of *PE-1* to initiate the write of "1" the *Set Status for Upstream Induct* register in the PLC logic; the physical length of the Carton must be less that the distance between *PE-1* and *Module A's* upstream zone sensor because *Module A* inducts the tracking data upon the Carton arriving at its upstream sensor and if *Set Status for Upstream Induct* has not changed from 4 to 1, the tracking data will not be valid.**

If a value of "4" is in the *Set Status for Upstream Induct* register at the same time the Carton reaches the local upstream zone's sensor; the local upstream zone will assume the Carton is physically longer than one zone and will invoke it's on board Flex Zone logic and behave accordingly. Please refer to the *ConveyLinx-Ai2 User's Guide* for description of Flex Zone operation. To reliably transfer tracking data and insure single Carton induction to the local upstream zone, the PLC programmer must insure proper use of the *Set Status for Upstream Induct* register.

Example 4 – Downstream Discharge Interface

This example shows how to control the discharge of a Carton from the most downstream zone of a ConveyLinx controlled conveyor. This example also shows how to properly accept the tracking data from the ConveyLinx module. *Figure 6* shows a typical example.

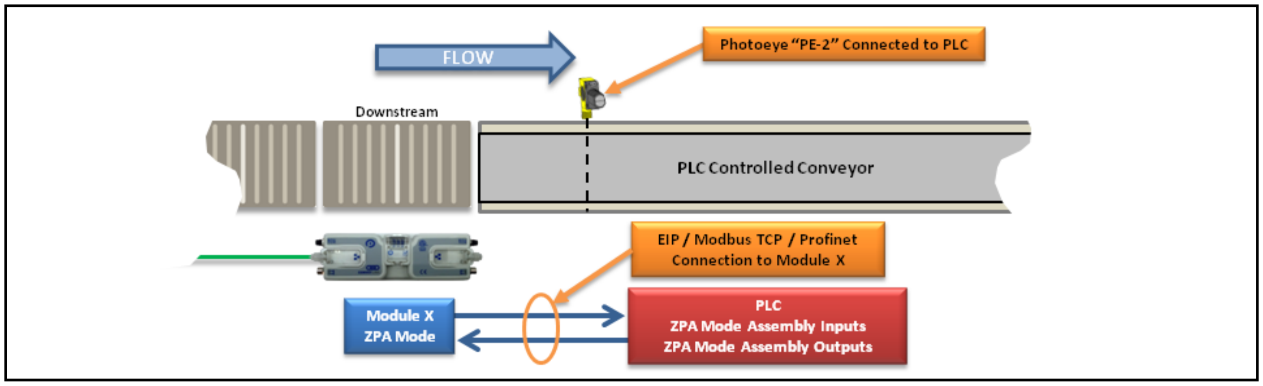

FLOW
Photoeye "PE-2" Connected to PLC
Downstream
PLC Controlled Conveyor
EIP / Modbus TCP / Profinet Connection to Module X
Module X ZPA Mode
PLC
ZPA Mode Assembly Inputs
ZPA Mode Assembly Outputs

FIGURE 6 - DOWNSTREAM DISCHARGE INTERFACE EXAMPLE

For this example, the PLC must establish a connection with *Module X*.

1. To hold any Carton that arrives at *Module X* downstream zone, the PLC writes a "5" to *Set Status for Downstream Discharge* register. This tells *Module X* that downstream conditions are "full and stopped" so it will not try to release any loads downstream.
2. When the PLC controlled conveyor is ready to accept a Carton from the MDR downstream zone, it writes a "1" to *Set Status for Downstream Discharge* register. This tells *Module X* that downstream conditions are clear and if it has a Carton on its downstream zone, it will run to release it downstream.
3. When the Carton clears the local sensor in *Module X* Downstream Zone, the tracking data for that Carton will be populated in *Get Tracking Forward Direction Word 1/2* registers.
4. When Carton arrives at *PE-2*, the PLC writes a "5" to *Set Status for Downstream Discharge* register. This tells *Module X* that the Carton successfully transferred. If this is not done, then *Module X* will produce an Arrival jam condition.

Optional Step:

If the application is such that the PLC would like to inhibit a new Carton from entering *Module X* downstream zone for whatever reason, there is a way to accumulate the next upstream zone to the *Module X* downstream zone. The PLC can set bit 8 in the *Accumulation Control for Local Downstream Zone* and this will cause the adjacent upstream zone to accumulate (whether this zone is on *Module X* or next adjacent upstream ConveyLinx Module). Please note that if *Module X* is configured as a single zone, then you must set bit 8 in the *Accumulation Control for Local Upstream Zone* register.

To return the accumulated upstream zone to normal operation; the PLC simply resets bit 8 in this register

Example 5 – Simple Divert Example

This example shows how to use a PLC to control a simple divert mechanism to divert a Carton from a ZPA zone and capture its tracking. This example also illustrates how to use the set downstream arrival function with the PLC to keep the discharging Module from generating a jam condition when the Carton is moved from the ZPA zone to the divert lane. *Figure 7* shows the layout for this example.

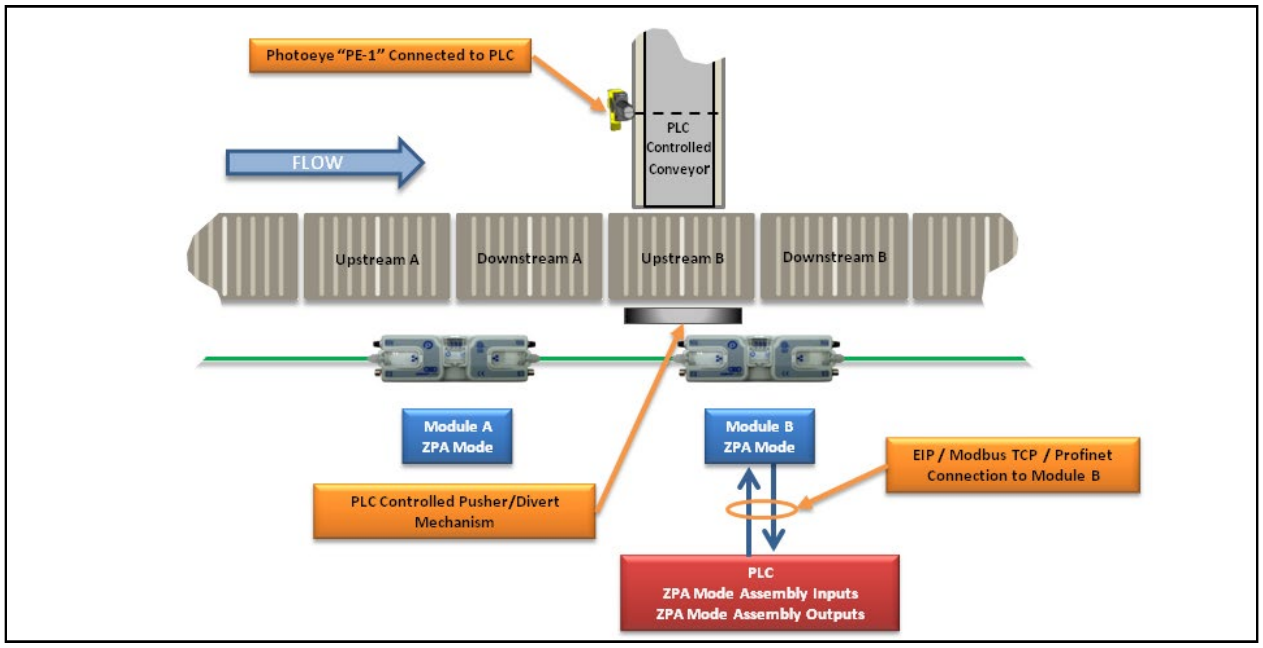

Photoeye "PE-1" Connected to PLC
FLOW
PLC Controlled Conveyor
Upstream A
Downstream A
Upstream B
Downstream B
Module A ZPA Mode
Module B ZPA Mode
EIP / Modbus TCP / Profinet Connection to Module B
PLC Controlled Pusher/Divert Mechanism
PLC
ZPA Mode Assembly Inputs
ZPA Mode Assembly Outputs

FIGURE 7 - SIMPLE DIVERT EXAMPLE

This example assumes that the tracking data arriving from each new Carton arrival at *Module B* Upstream Zone contains a value that will indicate to the PLC that a Carton does or does not need to divert. It is not required that the PLC use Carton tracking data be used to determine when to divert a Carton.

1. Set bit 0 in *Accumulation Control for Local Upstream Zone* register to instruct this zone to accumulate any Carton that arrives.
2. Monitor *Arrival Count Local Upstream Zone* and *Departure Count Local Upstream Zone* registers. On the leading edge when these two values become not equal, the PLC knows there is a new arrival. Note that as long as a Carton is physically occupying the upstream zone, these two values will not be equal.
3. Upon a successful arrival of a Carton in the upstream zone, then tracking data in *Current Upstream Zone Tracking Word 1* and *Current Upstream Zone Tracking Word 2*) will be valid for the newly arrived Carton.

<u>Assuming the PLC determines that the newly arrived Carton needs to divert</u>:

- Set bit 8 of *Accumulation Control for Local Upstream Zone* register. This will keep *Module A* from releasing a Carton to *Module B* while the divert operation is in progress.
- PLC will then initiate it's divert mechanism.

- When carton reaches PE-1, the PLC will then set bit 9 of the *Accumulation Control for Local Upstream Zone* register. This will tell *Module B* that the carton "successfully arrived at its downstream position".
- Once carton has cleared PE-1, then the PLC should clear bit 9 of the *Accumulation Control for Local Upstream Zone* register.
- PLC then can clear bit 8 of the *Accumulation Control for Local Upstream Zone* register to instruct *Module A* that it is OK to allow the next carton to enter *Module B*'s upstream zone.

Please note that in this example, the setting of bit 9 in the *Accumulation Control for Local Upstream Zone* register can simply be written in the PLC logic based upon block and clear of PE-1. Care must be taken so that *Module B's* Upstream zone's sensor is not blocked at the same time this bit 9 is set. If this zone's sensor is blocked at the same time bit 9 is set; *Module B* will assume that the carton is more than one zone in length and will automatically initiate Flex Zone operation which may produce unexpected results in carton flow. Please refer to User's Guide for complete description of Flex Zone function and how to disable if needed.

<u>If PLC determines that the Carton needs to go straight and not divert</u>:

The PLC can simply modify Carton's tracking data if desired and release the Carton as described in section *Example 1 – Basic Accumulate and Release with Tracking Data* on page 32.

EXAMPLE 6 – MERGE ONTO ZPA MAIN LINE

This example shows how to perform a simple merge with only a single *Module* connection to the PLC. Figure 8 depicts the conveyor layout and *Module* configuration for this example. In the example, *Module B* could be eliminated by simply making the shaded merging area be 2 extended zones on *Module C*. This example shows how easy it is to put pieces together with *EasyRoll* and using the *Extensions* functionality. Please refer to *ConveyLinx User's Guide* and *EasyRoll* software on screen directions for more details on creating extension zones.

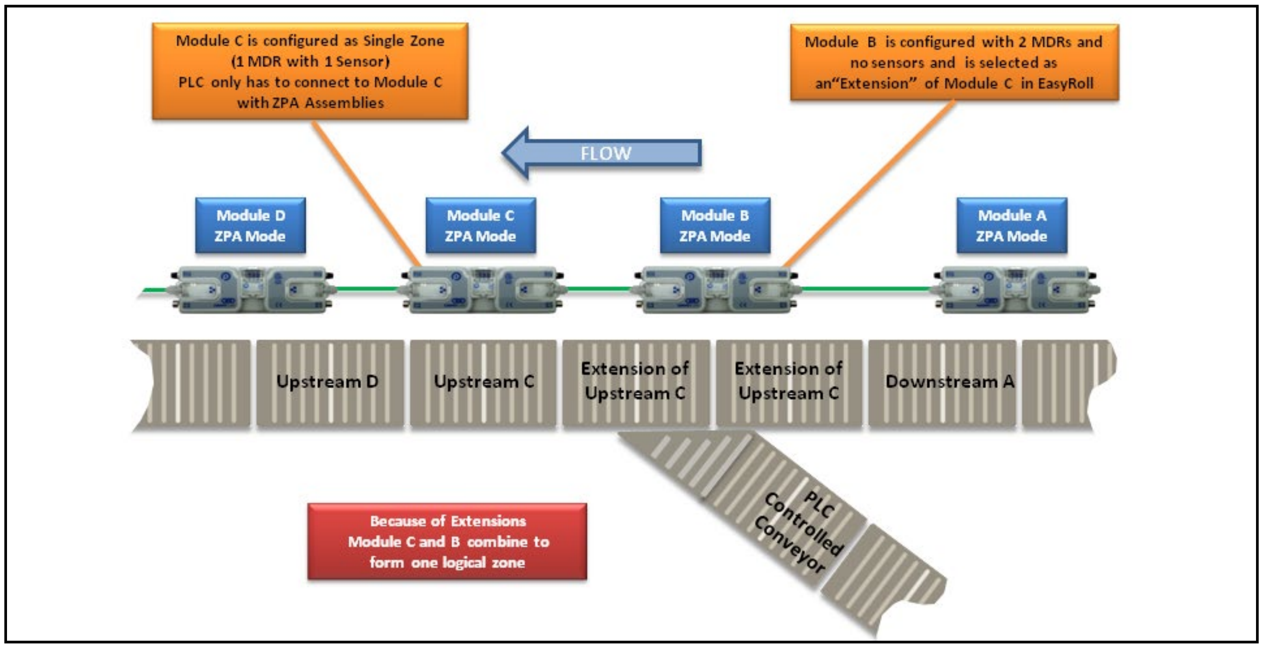

Module C is configured as Single Zone (1 MDR with 1 Sensor) PLC only has to connect to Module C with ZPA Assemblies
Module B is configured with 2 MDRs and no sensors and is selected as an"Extension" of Module C in EasyRoll
FLOW
Module D ZPA Mode
Module C ZPA Mode
Module B ZPA Mode
Module A ZPA Mode
Upstream D
Upstream C
Extension of Upstream C
Extension of Upstream C
Downstream A
PLC Controlled Conveyor
Because of Extensions Module C and B combine to form one logical zone

FIGURE 8 - MERGE ONTO ZPA MAIN LINE EXAMPLE

In this example the PLC only needs to connect to *Module C* and from this connection, the PLC can monitor conditions on *Module A* as well as "wake-up" *Module C / Module B* zone to accept a carton from the merging curve section.

1. With no PLC intervention, as long as cartons arrive into *Module A,* they will be conveyed to *Module C/B* and onto *Module D* in normal ZPA fashion.
2. To verify that *Module C* (and by extension *Module B*) is ready to accept a carton from the merge curve; the PLC needs to know that *Module C*'s zone is clear and stopped by examining *Local Status Upstream Zone* register. When this register equals 1, the zone is clear and stopped.
3. When the PLC needs to convey a carton from the merging curve, the PLC needs to set bit 8 in *Accumulation Control for Local Upstream Zone* register to accumulate the adjacent upstream zone, in this example setting bit 8 will cause *Module A* downstream zone to accumulate any carton that arrives.
4. When the PLC is ready to release a carton from the merging curve conveyor, the PLC sets bit 12 in the *Accumulation Control for Local Upstream Zone* register to wake up the *Module C* zone (along with *Module B*'s zones because *Module B* is an extension of *Module C*).
5. When the PLC sees the *Local Status Upstream Zone* register change to a value of 4 or 5, the PLC knows that the carton made it to *Module C*'s zone sensor and must then reset bit 12 in the *Accumulation Control for Local Upstream Zone* register. Either of these values will indicate a successful arrival at *Module C*.

6. Once bit 12 of the *Accumulation Control for Local Upstream Zone* register has been reset, the PLC can reset bit 8 of the same register to indicate to *Module A* that it is OK to continue normal ZPA function.

Please note that merging functionality is built-in and configurable with ConveyLinx-Ai2 modules. Please see publication ERSC-1020 Supplement to User's Guide for ConveyMerge from pulseroller.com for details.

Reduced Size ZPA Mode Assemblies

For some PLC controllers, the data size footprint required for a given assembly can be a limiting factor on how many devices can connect to a given controller. *ConveyLinx-Ai2* includes input/output assemblies with fewer registers that contain basic functions in applications where the full functionality of the Standard ZPA Mode Assemblies is not required. The Reduced Size Assemblies provide basic accumulation and release control along with module diagnostic data while removing the interfaces for reading/writing tracking data.

Reduced Size ZPA Mode Assembly Inputs

Please note that the functions provided by this assembly are already included and described in detail in section *ZPA Mode Assembly Inputs for PLC* on page 22.

ERSC Internal Address	*Register Name*	*Assembled EIP Address for PLC*	*Assembled Modbus Address for PLC*	*Assembled Profinet Address for PLC*
4:0105	Local Status Upstream Zone	**I.Data[0]**	**4:2900**	**Byte 0 (Hi) Byte 1 (Lo)**
4:0106	Local Status Downstream Zone	**I.Data[1]**	**4:2901**	**Byte 2 (Hi) Byte 3 (Lo)**
4:0107	Arrival Count Local Upstream Zone	**I.Data[2]**	**4:2902**	**Byte 4 (Hi) Byte 5 (Lo)**
4:0116	Departure Count Local Upstream Zone	**I.Data[3]**	**4:2903**	**Byte 6 (Hi) Byte 7 (Lo)**
4:0185	Arrival Count Local Downstream Zone	**I.Data[4]**	**4:2904**	**Byte 8 (Hi) Byte 9 (Lo)**
4:0186	Departure Count Local Downstream Zone	**I.Data[5]**	**4:2905**	**Byte 10 (Hi) Byte 11 (Lo)**
4:0187	Module Status Word 1	**I.Data[6]**	**4:2906**	**Byte 12 (Hi) Byte 13 (Lo)**
4:0196	Module Status Word 2	**I.Data[7]**	**4:2907**	**Byte 14 (Hi) Byte 15 (Lo)**
4:0035	Current Release Count for Upstream Zone	**I.Data[8]**	**4:2908**	**Byte 16 (Hi) Byte 17 (Lo)**
4:0088	Current Release Count for Downstream Zone	**I.Data[9]**	**4:2909**	**Byte 18 (Hi) Byte 19 (Lo)**
4:0089	Sensor & Control Port Inputs	**I.Data[10]**	**4:2910**	**Byte 20 (Hi) Byte 21 (Lo)**
4:0019	ConveyStop Status	**I.Data[11]**	**4:2911**	**Byte 22 (Hi) Byte 23 (Lo)**

Reduced Size ZPA Mode Assembly Outputs

Please note that the functions provided by this assembly are already included and described in detail in section *ZPA Mode Assembly Outputs for PLC* on page 26.

ERSC Internal Address	Register Name	EIP Address for PLC	Modbus Address for PLC	Profinet Address for PLC
4:0104	Accumulation Control for Local Upstream Zone	**O.Data[0]**	**4:3000**	**Byte 0 (Hi) Byte 1 (Lo)**
4:0184	Accumulation Control for Local Downstream Zone	**O.Data[1]**	**4:3001**	**Byte 2 (Hi) Byte 3 (Lo)**
4:0040	Set Left MDR Speed	**O.Data[2]**	**4:3002**	**Byte 4 (Hi) Byte 5 (Lo)**
4:0064	Set Right MDR Speed	**O.Data[3]**	**4:3003**	**Byte 6 (Hi) Byte 7 (Lo)**
4:0105	Release and Accumulate on Next Arrival for Local Upstream Zone	**O.Data[4]**	**4:3004**	**Byte 8 (Hi) Byte 9 (Lo)**
4:0185	Release and Accumulate on Next Arrival for Local Downstream Zone	**O.Data[5]**	**4:3005**	**Byte 10 (Hi) Byte 11 (Lo)**
4:0134	Set Status for Upstream Induct	**O.Data[6]**	**4:3006**	**Byte 12 (Hi) Byte 13 (Lo)**
4:0232	Set Status for Downstream Discharge	**O.Data[7]**	**4:3007**	**Byte 14 (Hi) Byte 15 (Lo)**
4:0022	Clear Motor Error	**O.Data[8]**	**4:3008**	**Byte 16 (Hi) Byte 17 (Lo)**
4:0063	Set Control Port Outputs	**O.Data[9]**	**4:3009**	**Byte 18 (Hi) Byte 19 (Lo)**
4:0020	ConveyStop Command	**O.Data[10]**	**4:3010**	**Byte 20 (Hi) Byte 21 (Lo)**
4:0109	Clear Sensor Jam Command for Local Upstream Zone	**O.Data[11]**	**4:3011**	**Byte 22 (Hi) Byte 23 (Lo)**
4:0189	Clear Sensor Jam Command for Local Downstream Zone	**O.Data[12]**	**4:3012**	**Byte 24 (Hi) Byte 25 (Lo)**
4:0365	Direction & Accumulation Mode Control for Local Upstream Zone	**O.Data[13]**	**4:3013**	**Byte 26(Hi) Byte 27 (Lo)**
4:0375	Direction & Accumulation Mode Control for Local Downstream Zone	**O.Data[14]**	**4:3014**	**Byte 28 (Hi) Byte 29 (Lo)**

PLC CONTROLLER WITH PLC I/O MODE

When a *ConveyLinx-Ai2* 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 *ConveyLinx-Ai2* 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 Aux I/O Pin 2 as digital outputs
- Ability to independently run both Left and Right motors
- Ability to set speed, acceleration, deceleration, 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.

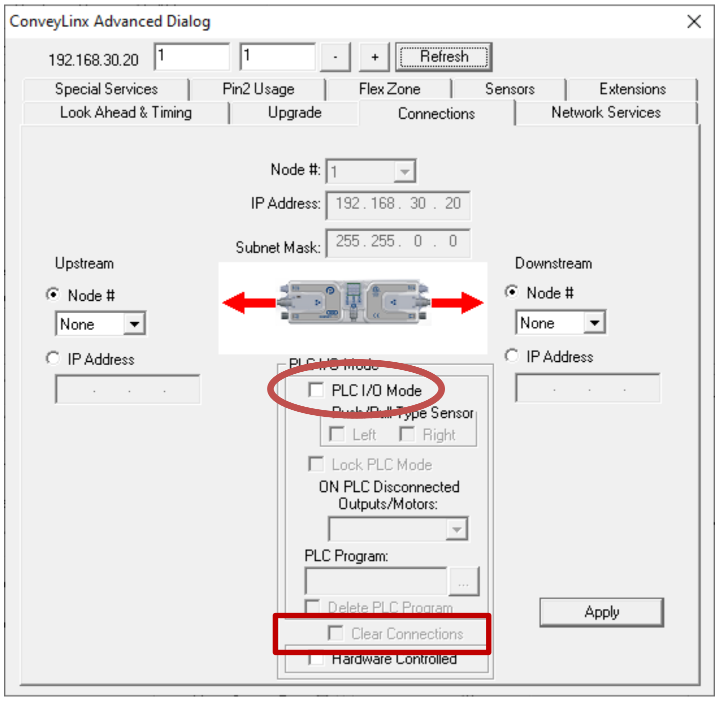
When a *ConveyLinx-Ai2* is placed in PLC I/O mode; it suspends all of its internal ZPA logic control. Any sensors or motors connected to the *ConveyLinx-Ai2* require explicit interaction with an external controller. The external controller will have typical Ethernet-based remote I/O performance from a *ConveyLinx-Ai2* when in PLC I/O mode.

Setting PLC I/O Mode in EasyRoll

Individual *ConveyLinx-Ai2*'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.

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. **<u>Check or Uncheck the "Clear Connections" checkbox depending upon your application.</u>** Click "Apply" to initiate the change. The *ConveyLinx-Ai2* will restart and this may take several seconds to complete.



192.168.30.20 1 1

Node #: 1

IP Address: 192 . 168 . 30 . 20

Subnet Mask: 255 . 255 . 0 . 0

Upstream: Node # None

Downstream: Node # None

Optional "Clear Connections" Choice

Refer to *Appendix B – ConveyLinx Connections* for further details on ConveyLinx Connections.

The decision to "Clear Connections" is based upon the application. When a string of *ConveyLinx-Ai2* modules are Auto-Configured, each successive *ConveyLinx-Ai2* in the string establishes a logical upstream / downstream **connection** with its neighbor *ConveyLinx-Ai2*'s. These **connections** provide the basis for the logical flow of inter-module status data for ZPA functionality. However, if a single *ConveyLinx-Ai2* 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 *ConveyLinx-Ai2* in PLC I/O will maintain its inter-module data exchange. 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 a *ConveyLinx-Ai2* in PLC I/O mode. For example, if the PLC I/O mode *ConveyLinx-Ai2* is being either fed or feeds conveyors controlled by *ConveyLinx-Ai2*'s in standard ZPA mode, these *ConveyLinx-Ai2*'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 *ConveyLinx-Ai2*'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 a *ConveyLinx-Ai2* 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 *ConveyLinx-Ai2*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 *ConveyLinx-Ai2* has been placed in PLC I/O mode, the ONLY way to return it to ZPA mode is to perform an Auto-Configuration procedure or Restore from a backup file with *EasyRoll*. There is no "undo" or "reset" function for this action.

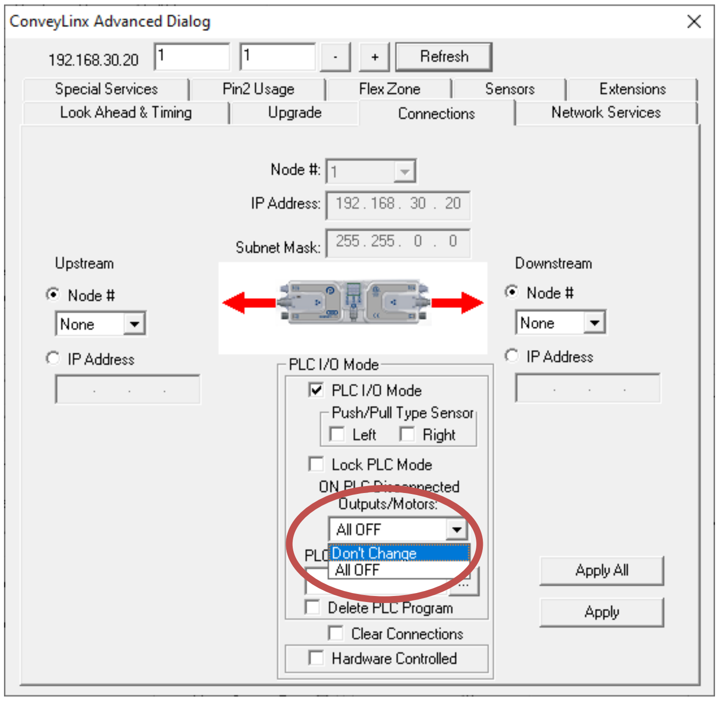
Configuring Action for Loss of Communication

When changing the mode of a given *ConveyLinx-Ai2* to PLC I/O mode in *EasyRoll*, you are given the option to select the behavior of the *ConveyLinx-Ai2's* outputs upon loss of communications with the PLC.

Select "Don't Change" from the drop-down box 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 *ConveyLinx-Ai2* 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 *ConveyLinx-Ai2* will automatically resume having its outputs and MDR's controlled by PLC command.



PLCI/O Mode Assembly Inputs

This chart shows the data sent by a *ConveyLinx-Ai2* in PLC I/O mode to the PLC when connected. All registers in the module are 16-bit integer data type.

Register Name / Module Address	*Address*	*Data Description*
ConveyStop Status 4:0019	**M: 4:1700** **E: I.Data[0]** **P: Byte 0 (Hi) Byte 1 (Lo)**	Bitwise Value - Read only bit 00 = Reserved bit 01 = Reserved bit 02 = Reserved bit 03 = Reserved bit 04 = Reserved bit 05 = Stop active on another module in Stop Group bit 06 = Stop active due to lost communication connection bit 07 = Stop active due to lost PLC connection bit 08 = Reserved bit 09 = Reserved bit 10 = Stop active due to Stop Command from PLC bit 11 = Reserved bit 12 = Reserved bit 13 = Reserved bit 14 = Reserved bit 15 = Reserved Refer to Insight Automation publication ERSC-1800 *ConveyStop User's Guide* for details on using *ConveyStop*.
Sensor Port Inputs 4:0035	**M: 4:1701** **E: I.Data[1]** **P: Byte 2 (Hi) Byte 3 (Lo)**	Bitwise Value - Read only bit 00 = Left Sensor Port – Auxiliary (M8 Pin2) bit 02 = Right Sensor Port - Auxiliary (M8 Pin2) bit 04 = Left Sensor Port – Signal (M8 Pin4) bit 06 = Right Sensor Port - Signal (M8 Pin4) bit 15 = 2 sec on / 2 sec off heartbeat All other bits reserved See *PLC I/O Mode Note* ④ on page 52
Sensor Detect 4:0036	**M: 4:1702** **E: I.Data[2]** **P: Byte 4 (Hi) Byte 5 (Lo)**	Bitwise Value - Read only bit 0 = Device is connected to Right Sensor Port bit 1 = Device is connected to Left Sensor Port
Motor Voltage 4:0024	**M: 4:1703** **E: I.Data[3]** **P: Byte 6 (Hi) Byte 7 (Lo)**	Value in mV of MDR Power Supply Range: 0 to 35000 *Example:23500 = 23.5 Volts*
Left Motor Current 4:0055	**M: 4:1704** **E: I.Data[4]** **P: Byte 8 (Hi) Byte 9 (Lo)**	Integer Value in mA – Current that motor is currently drawing *Example: 1900 = 1.9 Amps*
Left Motor Frequency 4:0056	**M: 4:1705** **E: I.Data[5]** **P: Byte 10 (Hi) Byte 11 (Lo)**	Integer Value in Hz – Current frequency that motor is currently running *Example: 300 = 300 Hz* See *PLC I/O Mode Note* ② on page 52

Register Name / Module Address	Address	Data Description	
Left Motor Temperature	M: 4:1706	High Byte / Low Byte Value of temperatures in °C	
4:0057	E: I.Data[6]	High Byte = Calculated motor temperature	
	P: Byte 12 (Hi) Byte 13 (Lo)	Low Byte = Temperature reading from on-board sensor	
Left Motor Status	M: 4:1707	Bitwise Value – See *PLC I/O Mode Note* ③ on page 52	
4:0058	E: I.Data[7]	bit 00 = Motor Status	bit 08 = Overheated
	P: Byte 14 (Hi) Byte 15 (Lo)	bit 01 = Motor Status	bit 09 = At Max. Torque
		bit 02 = Port in Digital Mode	bit 10 = Short Circuit
		bit 03 = Reserved	bit 11 = Motor Not Connected
		bit 04 = Reserved	bit 12 = Overloaded
		bit 05 = Board Overheat	bit 13 = Motor Stalled
		bit 06 = Over-Voltage	bit 14 = Hall Sensor Error
		bit 07 = Low Voltage	bit 15 = Motor Not Used
Right Motor Current	M: 4:1708	Integer Value in mA – Current that motor is currently drawing	
4:0079	E: I.Data[8]	*Example: 1900 = 1.9 Amps*	
	P: Byte 16 (Hi) Byte 17 (Lo)		
Right Motor Frequency	M: 4:1709	Integer Value in Hz – Current frequency that motor is currently running	
4:0080	E: I.Data[9]	*Example: 300 = 300 Hz*	
	P: Byte 18 (Hi) Byte 19 (Lo)	See *PLC I/O Mode Note* ② on page 52	
Right Motor Temperature	M: 4:1710	High Byte / Low Byte Value of temperatures in °C	
4:0081	E: I.Data[10]	High Byte = Calculated motor temperature	
	P: Byte 20 (Hi) Byte 21 (Lo)	Low Byte = Temperature reading from on-board sensor	
Right Motor Status	M: 4:1711	Bitwise Value - See *PLC I/O Mode Note* ③ on page 52	
4:0082	E: I.Data[11]	bit 00 = Motor Status	bit 08 = Overheated
	P: Byte 22 (Hi) Byte 23 (Lo)	bit 01 = Motor Status	bit 09 = At. Max. Torque
		bit 02 = Port in Digital Mode	bit 10 = Short Circuit
		bit 03 = Reserved	bit 11 = Motor Not Connected
		bit 04 = Reserved	bit 12 = Overloaded
		bit 05 = Reserved	bit 13 = Motor Stalled
		bit 06 = Over-Voltage	bit 14 = Hall Sensor Error
		bit 07 = Low Voltage	bit 15 = Motor Not Used
Left Motor Port Digital I/O Status	M: 4:1712	<u>Bitwise Value – Read Only</u>	
4:0060	E: I.Data[12]	bit 12 = Short Circuit Error on one or more outputs	
	P: Byte 24 (Hi) Byte 25 (Lo)	bit 14 = Over Current – More than 1A detected on one or more outputs	
Right Motor Port Digital I/O Status	M: 4:1713		
4:0084	E: I.Data[13]	See *Appendix A - Motor Port as Digital I/O* on page 67 for usage details	
	P: Byte 26 (Hi) Byte 27 (Lo)		
Upstream Module Status	M: 4:1714	Integer Value of Low Byte	
4:0134	E: I.Data[14]	0x01 = Sensor clear and motor stopped	
	P: Byte 28 (Hi) Byte 29 (Lo)	0x02 = Sensor clear, motor running, accepting from upstream	
Downstream Module Status	M: 4:1715	0x03 = Reserved	
4:0232	E: I.Data[15]	0x04 = Sensor blocked, motor running, discharging to downstream	
	P: Byte 30 (Hi) Byte 31 (Lo)	0x05 = Sensor blocked and motor stopped	
		0x06 = *ConveyLinx-Ai2* Busy	
		See *PLC I/O Mode Note* ① on page 51 for details	
Current Tracking Word 1 for Adjacent Upstream Module	M: 4:1716	Value = Tracking data word #1 (16-bit integer) for the Carton that has just discharged from the *ConveyLinx-Ai2* Adjacent to this local *ConveyLinx-Ai2* module.	
4:0139	E: I.Data[16]		
	P: Byte 32 (Hi) Byte 33 (Lo)	See *PLC I/O Mode Note* ① on page 51 for Details	

Register Name / Module Address	Address	Data Description
Current Tracking Word 2 for Adjacent Upstream Module 4:0140	**M: 4:1717 E: I.Data[17] P: Byte 34 (Hi) Byte 35 (Lo)**	Value = Tracking data word #2 (16-bit integer) for the Carton that has just discharged from the *ConveyLinx-Ai2* Adjacent to this local *ConveyLinx-Ai2* module. See *PLC I/O Mode Note* ① on page 51 for Details
Reserved	**M: 4:1718 E: I.Data[18] P: Byte 36 (Hi) Byte 37 (Lo)**	
Left Motor Servo Position 4:0062	**M: 4:1719 E: I.Data[19] P: Byte 38 (Hi) Byte 39 (Lo)**	Signed integer value that indicates the current position of the Left Motor in relation to its “0” position See section *Servo Motor Control Example* on page 59
Right Motor Servo Position 4:0086	**M: 4:1720 E: I.Data[20] P: Byte 40 (Hi) Byte 41 (Lo)**	Signed integer value that indicates the current position of the Right Motor in relation to its “0” position See section *Servo Motor Control Example* on page 59
Left Motor Servo Status 4:0011	**M: 4:1721 E: I.Data[21] P: Byte 42 (Hi) Byte 43 (Lo)**	Bit 0: Servo Command Status 1 = Last Servo Run Command Complete 0 = Servo Command in Process Bit 1: Servo Reset Status Echoes state of Left Motor Servo Command bit 0 Bit 2: Servo Command Status Echoes state of Left Motor Servo Command bit 1 See section *Servo Motor Control Example* on page 59
Right Motor Servo Status 4:0016	**M: 4:1722 E: I.Data[22] P: Byte 44 (Hi) Byte 45 (Lo)**	Bit 0: Servo Command Status 1 = Last Servo Run Command Complete 0 = Servo Command in Process Bit 1: Servo Reset Status Echoes state of Right Motor Servo Command bit 0 Bit 2: Servo Command Status Echoes state of Right Motor Servo Command bit 1 See section *Servo Motor Control Example* on page 59
Left Motor Real Speed 4:0507	**M: 4:1723 E: I.Data[23] P: Byte 46 (Hi) Byte 47 (Lo)**	Motor Speed and Set Speed Status Bit 0 thru Bit 13: Numerical value of speed. • For Motor Roller the value is in mm/s • For PGD the value is in RPM x 10 Bit 14: 1 = Set Speed is above motor’s maximum speed Bit 15: 1 = Set Speed is below motor’s minimum speed Refer to PLC I/O Mode Note ⑧ on page 57 for details
Right Motor Real Speed 4:0508	**M: 4:1724 E: I.Data[24] P: Byte 48 (Hi) Byte 49 (Lo)**	

PLC I/O Mode Note ①

These registers only contain meaningful data if the ConveyLinx connections between upstream and/or downstream *ConveyLinx-Ai2*s are preserved when placing the local *ConveyLinx-Ai2* into PLC I/O mode from within *EasyRoll*. If connections are cleared in *EasyRoll*, these registers will not contain any pertinent data and will not be updated by adjacent *ConveyLinx-Ai2s*. Refer to section *Optional “Clear Connections” Choice* on page 47 for further details.

PLC I/O Mode Note ②

Motor frequency is determined by a combination of motor RPM and the number of poles the motor contains and these will be different depending on the particular MDR used. Please consult MDR documentation to determine meaningful values for motor frequency

PLC I/O Mode Note ③

Bits 0 and 1 are used in combination to provide 4 possible states. The following chart defines the bit values for these states:

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 CCW Direction
1	0	Motor running in CW Direction
1	1	Motor not running and no braking applied (free to spin)

PLC I/O Mode Note ④

The electrical logic state of the signal on the Sensor/Control Port pins are bit-wise exclusive OR (XOR) with the bits set in the *Sensor & Control Port Input Signal Condition Mask* register to arrive at the bit values seen in the *Sensor & Control Port Inputs* register. This allows the PLC programmer to control whether an electrically energized condition results in a logical 1 or a logical 0 in the *Sensor & Control Port Inputs* register. This same relationship is also mirrored by the state of the input's corresponding LED indicator.

See *PLC I/O Mode Note* ⑦ on page 56 for further details.

PLC I/O Mode Note ⑤

The data in this register is only valid when a given motor port is placed into Digital I/O mode by the PLC. When a given motor port is being used as Digital I/O the only meaningful bits are bits 12 and 14 as shown. Other bits in this register may contain values and should be ignored as valid input data.

Refer to *Appendix A - Motor Port as Digital I/O* on page 75 for wiring details for using a motor port as digital I/O.

PLC I/O Mode Assembly Outputs

These registers are written by the PLC to the *Module* when the module is in PLC I/O mode. All registers are 16-bit Integer data type.

Register Name	PLC Address	Data Description
ConveyStop Command 4:0020	**M: 4:1800** **E: O.Data[0]** **P: Byte 0 (Hi) Byte 1 (Lo)**	Integer Value 0 = No Command 1 = Command local module's Stop Group to go to Stopped State 2 = Command local module's Stop Group to Clear Stopped State Refer to Insight Automation publication ERSC-1800 *ConveyStop User's Guide* for details on using *ConveyStop*.
Set Left Motor Port Digital Control 4:0060	**M: 4:1801** **E: O.Data[1]** **P: Byte 2 (Hi) Byte 3 (Lo)**	Bitwise Values bit 0: Reserved bit 1: Energize Motor Port Pin 4 bit 2: Energize Motor Port Pin 3 bit 3 thru bit 14 = Reserved bit 15 = Digital Output Enable 0 = Use port as Motor Control 1 = Use port as Digital Output See *Appendix A - Motor Port as Digital I/O* on page 75 for connection details.
Set Right Motor Port Digital Control 4:0084	**M: 4:1802** **E: O.Data[2]** **P: Byte 4 (Hi) Byte 5 (Lo)**	Bitwise Values bit 0: Energize Motor Port Pin 2 bit 1: Reserved bit 2: Energize Motor Port Pin 3 bit 3 thru bit 14 = Reserved bit 15 = Digital Output Enable 0 = Use port as Motor Control 1 = Use port as Digital Output See *Appendix A - Motor Port as Digital I/O* on page 75 for connection details
Sensor Port Digital Output Control 4:0037	**M: 4:1803** **E: O.Data[3]** **P: Byte 6 (Hi) Byte 7 (Lo)**	Bitwise Values bit 0: 1 = Left Pin 2 ON, 0 = Left Pin 2 OFF bit 1: 1 = Right Pin 2 ON, 0 = Right Pin 2 OFF bit 2 thru bit 4: Reserved bit 5: 1 = Enable Left Pin 2 as Output, 0 = Disable Left Pin 2 as Output bit 6: 1 = Enable Right Pin 2 as Output, 0 = Disable Right Pin 2 as Output bit 7 thru bit 15: Reserved
Left Motor Run / Reverse 4:0260	**M: 4:1804** **E: O.Data[4]** **P: Byte 8 (Hi) Byte 9 (Lo)**	Bit 0: 1 = Run Command 0 = Stop Command Bit 8: 0 = Run in Configured Direction 1 = Run opposite of Configured Direction
Left Motor Brake Method 4:0261	**M: 4:1805** **E: O.Data[5]** **P: Byte 10 (Hi) Byte 11 (Lo)**	Integer Value 0 = Use Configured Brake Method 1 = Use Standard Brake Method 2 = Use Free Coast Brake Method 3 = Use Servo Brake Method

Register Name	*PLC Address*	*Data Description*
Left Motor Slave Mode 4:0262	**M: 4:1806** **E: O.Data[6]** **P: Byte 12 (Hi) Byte 13 (Lo)**	Integer Value 0 = Ignore 1 = Slave Mode OFF – Left motor independently controlled 2 = Slave Mode ON – Left motor mirrors Right motor control 3 = Slave Mode ON – Left motor runs in opposite direction of Right motor Refer to *PLC I/O Mode Note* ⑨ on page 57 for details
Right Motor Run / Reverse 4:0270	**M: 4:1807** **E: O.Data[7]** **P: Byte 14 (Hi) Byte 15 (Lo)**	Bit 0: 1 = Run Command 0 = Stop Command Bit 8: 0 = Run in Configured Direction 1 = Run opposite of Configured Direction
Right Motor Brake Method 4:0271	**M: 4:1808** **E: O.Data[8]** **P: Byte 16 (Hi) Byte 17 (Lo)**	Integer Value 0 = Use Configured Brake Method 1 = Use Standard Brake Method 2 = Use Free Coast Brake Method 3 = Use Servo Brake Method
Right Motor Slave Mode 4:0272	**M: 4:1809** **E: O.Data[9]** **P: Byte 18 (Hi) Byte 19 (Lo)**	Integer Value 0 = Ignore 1 = Slave Mode OFF – Right motor independently controlled 2 = Slave Mode ON – Right motor mirrors Left motor control 3 = Slave Mode ON – Right motor runs in opposite direction of Left motor Refer to *PLC I/O Mode Note* ⑨ on page 57 for details
Left Motor Speed Reference 4:0040	**M: 4:1810** **E: O.Data[10]** **P: Byte 20 (Hi) Byte 21 (Lo)**	Integer value to set motor speed For MDR value is in mm/s For PGD value is in RPM x 10 Value = 0 Remain at last non zero value entered Refer to *PLC I/O Mode Note* ⑧ on page 57 for details
Right Motor Speed Reference 4:0064	**M: 4:1811** **E: O.Data[11]** **P: Byte 22 (Hi) Byte 22 (Lo)**	
Left Motor Acceleration Ramp 4:0043	**M: 4:1812** **E: O.Data[12]** **P: Byte 24 (Hi) Byte 25 (Lo)**	Integer Value For MDR value is in mm For PGD value is in motor pulses* Deceleration Range: 0 to 10000 (Both MDR and PGD) Acceleration Range: 30 to 10000 (Both MDR and PGD) Value = 0: Remain at last non-zero value entered * Please consult PGD documentation for gear ratio to equate motor pulses to shaft revolution
Left Motor Deceleration Ramp 4:0044	**M: 4:1813** **E: O.Data[13]** **P: Byte 26 (Hi) Byte 27 (Lo)**	
Right Motor Acceleration Ramp 4:0067	**M: 4:1814** **E: O.Data[14]** **P: Byte 28 (Hi) Byte 29 (Lo)**	
Right Motor Deceleration Ramp 4:0068	**M: 4:1815** **E: O.Data[15]** **P: Byte 30 (Hi) Byte 31 (Lo)**	
Clear Motor Error 4:0022	**M: 4:1816** **E: O.Data[16]** **P: Byte 32 (Hi) Byte 33 (Lo)**	Logical 0 or 1 0 = Stop Reset 1 = Send Reset Refer to *ZPA Mode Note* ⑥ on page 30 for details.

Register Name	PLC Address	Data Description
Set Status to Downstream Module 4:0196	M: 4:1817 E: O.Data[17] P: Byte 34 (Hi) Byte 35 (Lo)	Used to write *ConveyLinx-Ai2* ZPA Status data to downstream *ConveyLinx-Ai2*: 4 = Instruct Downstream *ConveyLinx-Ai2* to "wake-up" and run its most upstream zone 1 = Instructs Downstream *ConveyLinx-Ai2* that carton has exited local zone and to accept any tracking data written in *Set Discharge Tracking Word 1 / Word 2* registers when carton arrives. See *PLC I/O Mode Note* ⑥ on page *56*
Set Status to Upstream Module 4:0116	M: 4:1818 E: O.Data[18] P: Byte 36 (Hi) Byte 37 (Lo)	Used to write *ConveyLinx-Ai2* ZPA Status data to next upstream *ConveyLinx-Ai2*: 5 = Instructs Upstream *ConveyLinx-Ai2*'s discharge zone to accumulate and hold any carton that arrives at its discharge zone. 1 = Instructs Upstream *ConveyLinx-Ai2*'s discharge zone to release any carton that arrives at its discharge zone. See *PLC I/O Mode Note* ⑥ on page *56*
Sensor Port Input Signal Condition Mask 4:0034	M: 4:1819 E: O.Data[19] P: Byte 38 (Hi) Byte 39 (Lo)	Bitwise Value bit 00 = Left Sensor Port – Auxiliary (M8 Pin2) bit 02 = Right Sensor Port - Auxiliary (M8 Pin2) bit 04 = Left Sensor Port – Signal (M8 Pin4) bit 06 = Right Sensor Port - Signal (M8 Pin4) All other bits reserved See *PLC I/O Mode Note* ⑦ on page 56
Set Discharge Tracking Word 1 4:0201	M: 4:1820 E: O.Data[20] P: Byte 40 (Hi) Byte 41 (Lo)	Only used when local PLC I/O Mode *ConveyLinx-Ai2* needs to pass tracking data to a downstream connected *ConveyLinx-Ai2*. Used in conjunction with *Set Status to Downstream Module* register See *PLC I/O Mode Note* ⑥ on page *56*
Set Discharge Tracking Word 2 4:0202	M: 4:1821 E: O.Data[21] P: Byte 42 (Hi) Byte 42 (Lo)	
Reserved	M: 4:1822 E: O.Data[22] P: Byte 44 (Hi) Byte 44 (Lo)	
Left Motor Servo Command Distance 4:0008	M: 4:1823 E: O.Data[23] P: Byte 46 (Hi) Byte 47 (Lo)	Signed integer value in mm of the distance to move to on the next Left Motor Servo Run Command Valid values are from -32767 to +32767 See section *Servo Motor Control Example* on page 59
Left Motor Servo Command Word 4:0009	M: 4:1824 E: O.Data[24] P: Byte 48 (Hi) Byte 49 (Lo)	Bit 0: Reset Command 1 = Set Current Distance Count as "0" Bit 1: Servo Run Command 1 = Run in Motor from current count to the set mm count in Left Motor Servo Command Distance Register See section *Servo Motor Control Example* on page 59
Right Motor Servo Command Distance 4:0013	M: 4:1825 E: O.Data[25] P: Byte 50 (Hi) Byte 51 (Lo)	Signed integer value in mm of the distance to move to on the next Left Motor Servo Run Command Valid values are from -32767 to +32767 See section *Servo Motor Control Example* on page 59

Register Name	*PLC Address*	*Data Description*
Right Motor Servo Command Word 4:0014	**M: 4:1826** **E: O.Data[26]** **P: Byte 52 (Hi) Byte 52 (Lo)**	Bit 0: Reset Command 1 = Set Current Distance Count as "0" Bit 1: Servo Run Command 1 = Run in Motor from current count to the set mm count in Left Motor Servo Command Distance Register See section *Servo Motor Control Example* on page 59

PLC I/O Mode Note ⑥

For registers involving status and tracking to upstream or downstream *ConveyLinx-Ai2*'s in ZPA mode, the connections must be preserved when placing the *ConveyLinx-Ai2* in question into PLC I/O mode from within *EasyRoll*. Refer to section *Optional "Clear Connections" Choice* on page 47 for further details.

PLC I/O Mode Note ⑦

For a *ConveyLinx-Ai2* in standard ZPA mode, the Auto-Configuration procedure sets values in this register to allow the *ConveyLinx-Ai2* 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 *ConveyLinx-Ai2*'s sensor input pin 4 when the zone is clear. The *ConveyLinx-Ai2* Sensor port LED indicator for pin 4 (green) should illuminate when the zone is blocked; so, the Auto-Configuration procedure sets a bit in the *Sensor & Control Port Input Signal Condition Mask* register 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 a *ConveyLinx-Ai2* is placed in PLC I/O mode; the *Sensor & Control Port Input Signal Condition Mask* register is cleared of the values set during the Auto-Configure procedure. The *Sensor & Control Port Input Signal Condition Mask* register is made available for PLC I/O mode Sensor port inputs to give the PLC programmer the same flexibility for configuring which electrical state (on or off) of the input will cause a logical 1 to appear in the *Sensor & Control Port Inputs* register and illuminate the pin's corresponding LED. By setting or clearing the corresponding bit for a given port's sensor or aux I/O signal, the PLC programmer can determine which physical state (on or off) of the input signal will cause its corresponding pin's bit in the *Sensor & Control Port Inputs* register to be set and its corresponding LED to illuminate.

The following is an example that shows the bit patterns and signals for one of the *ConveyLinx-Ai2*'s inputs. The same pattern applies to all available *ConveyLinx-Ai2* inputs:

Right Sensor Port - Pin 4 Signal			
Electrical Signal	*Sensor Port Input Signal Condition Mask* Register bit 6	*Sensor Port Inputs* Register bit 6	LED State
OFF	0	0	Green = OFF
ON	0	1	Green = ON
OFF	1	1	Green = ON
ON	1	0	Green = OFF

Be careful when changing the *Sensor & Control Port Input Signal Condition Mask* in your Ethernet I/P PLC program. The input bit values in the *Sensor & Control Port Inputs* register can show unexpected or opposite values from expected until the PLC has updated the *Sensor & Control Port Input Signal Condition Mask* data. This update could take several program scans depending upon the Ethernet IP RPI settings. Take care to be sure that the *Sensor & Control Port Input Signal Condition Mask* data is written to the *ConveyLinx-Ai2* Before acting on any input values in the *Sensor & Control Port Inputs* register.

Please note that the Sensor/Control Port LEDs are tri-colored. In applications where both Pin 4 and Pin 2 are used on the same Sensor Port; please note that when the green and red LEDs are illuminated at the same time, the color will be amber.

PLC I/O Mode Note ⑧

The Ai motor technology encodes the specific motor data into a memory chip inside the motor. This data is read by the *ConveyLinx-Ai2* module. Included in this data are the mechanical characteristics for the motor including the minimum and maximum speed settings. You can enter values in the speed reference registers that are outside the allowable range for the connected motor. If you enter a value greater than the maximum; the speed will be set to the maximum. If you enter a value lower than the minimum, the speed will be set to the minimum.

For the Motor Real Speed input registers, you can know if the value entered in the Speed Reference register is outside the allowable range by monitoring bits 14 and 15 of these registers.

For example, let's say we have an MDR whose maximum speed is 1.34 m/s. At this speed, the value in the Real Speed input register will show a decimal value very close or equal to 1340 (which is the speed in mm/s). If we enter a value of 2.0 m/s for the speed reference (value of 2000 in Speed Reference register) and then run the motor, in the Real Speed input register for that motor you will see a decimal value very close or equal to 17724. Analyzing this value at the bit level, you will see that bit 14 is set indicating that the Speed Reference value is above the maximum value. If you strip off bit 14 from this value, the remaining decimal value (i.e. the binary value of bits 0 thru 13) will be very close or equal to 1340; which is the maximum speed allowed for the MDR. Similarly, if the Speed Reference value entered is below the minimum value, bit 15 will be set in the motor's Real Speed register. Stripping out bit 15, the binary value of bits 0 thru 13 will indicate a value very close or equal to the motor's minimum allowed speed.

PLC I/O Mode Note ⑨

When enabling Slave Mode for a given motor (Left or Right), the motor type, speed, acceleration, deceleration, and braking mode settings used for the opposite motor are applied to the Slave Mode motor. The Slave Mode motor then runs and stops in unison with the control of the opposite or Master motor. For example, if Slave Mode is ON for the Left motor; then the Right motor's settings will be copied to the Left motor. In your PLC logic, you only need to start/stop the Right motor and the Left motor will automatically follow the Right motor.

Please note that if you enable Slave Mode for both the Left and Right motors at the same time, then the ConveyLinx-Ai2 will default to the Left motor being in Slave Mode and the Right motor will provide the master control.

SERVO MOTOR CONTROL EXAMPLE

Servo Motor Control requires ConveyLinx-Ai2 firmware 4.14 or later

The Servo Motor Control function allows you to establish a zero or "home" rotational position of the MDR and then instruct the *ConveyLinx-Ai2* to rotate an assigned number of mm in either direction from this zero or "home" position. PLC I/O Mode Assemblies provide the data registers to allow external PLC control of this function for both the Left and Right MDR ports.

The Servo Motor Control function utilizes the existing motor speed, acceleration, and deceleration registers. The existing starting, stopping, and direction control registers are not used and these functions are incorporated into the Servo Control registers.

For our example, we want to perform the following cycle using PLC control of the Left MDR on and *ConveyLinx-Ai2*:

- Establish a zero or home position by external input to PLC (sensor or operator button)
- Rotate in the CCW direction for 7000 mm at a speed of 0.8 m/s with acceleration ramp of 100 mm and a deceleration ramp of 50 mm
- Rotate the CW direction for 9000 mm at a speed of 1 m/s with acceleration ramp of 50 mm and a deceleration ramp of 10 mm.
- Rotate CCW back to the zero or home position at a speed of 0.8 m/s with acceleration ramp of 50 mm and a deceleration ramp of 10 mm.
- Wait for a cycle dwell time of 4 seconds and then repeat the rotation cycles

Now let's define some logic names that correspond to actual PLC I/O Mode Assembly tag values to make our example easier to follow:

Example Tag	*Data Type*	*Assembly Register (/bit)*
ZERO	Boolean	Left Servo Command Word - bit 0
ZERO_ACK	Boolean	Left Motor Servo Status - bit 1
RUN	Boolean	Left Servo Command Word - bit 1
READY	Boolean	Left Motor Servo Status - bit 0
POSITION	Integer	Left Servo Command Distance (in mm)
SPEED	Integer	Left Motor Speed Reference
ACCEL	Integer	Left Motor Acceleration Ramp
DECEL	Integer	Left Motor Deceleration Ramp

Step #1: Upon external signal from sensor or button, set ZERO bit to establish zero or "home" position. When PLC sees that the ZERO_ACK bit is set, then the PLC resets the ZERO bit.

Step #2: To make the first rotation, we need write the speed, ramp values, and distance to rotate to the appropriate registers:

- Write 800 to *SPEED* (800 mm/s = 0.8 m/s)
- Write 100 to *ACCEL*
- Write 50 to *DECEL*
- Write 7000 to *POSITION*

Step #3: If READY is reset, then the PLC can set the RUN bit to begin the rotation. When the rotation is complete, the *ConveyLinx-Ai2* sets the READY bit. This will be the signal to the PLC to reset the RUN bit. Once the *ConveyLinx-Ai2* sees that the RUN bit has been reset, it will reset the READY bit.

Step #4: For the second rotation, we need to write the speed and ramp values to the appropriate registers:

- Write 1000 to *SPEED* (1000 mm/s = 1 m/s)
- Write 50 to *ACCEL*
- Write 10 to *DECEL*

Because we want to rotate in the opposite direction, we need to determine the new location based upon the zero or "home" position. In this case, we know we went 7000 "forward" and we want to move 9000 "backward". The position we want to end up is 7000-9000 = -2000.

- Write -2000 to *POSITION*

Step #5: Repeat *Step #3*.

Step #6: For the 3rd rotation, we keep the ramp values from the 2nd rotation, but we need to set the speed and the position to rotate. In this case we want to go to the zero or "home position.

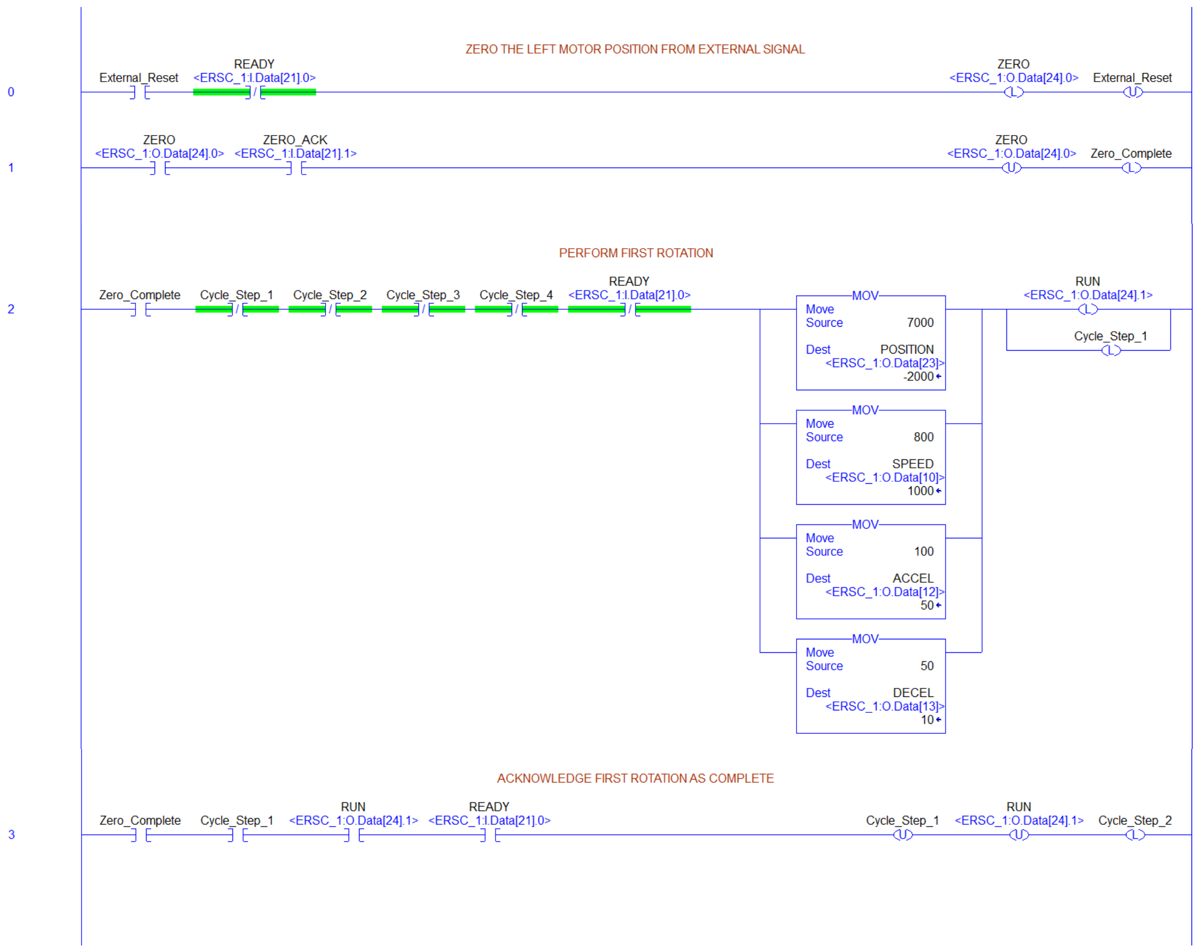
- Write 800 to SPEED
- Write 0 to *POSITION*

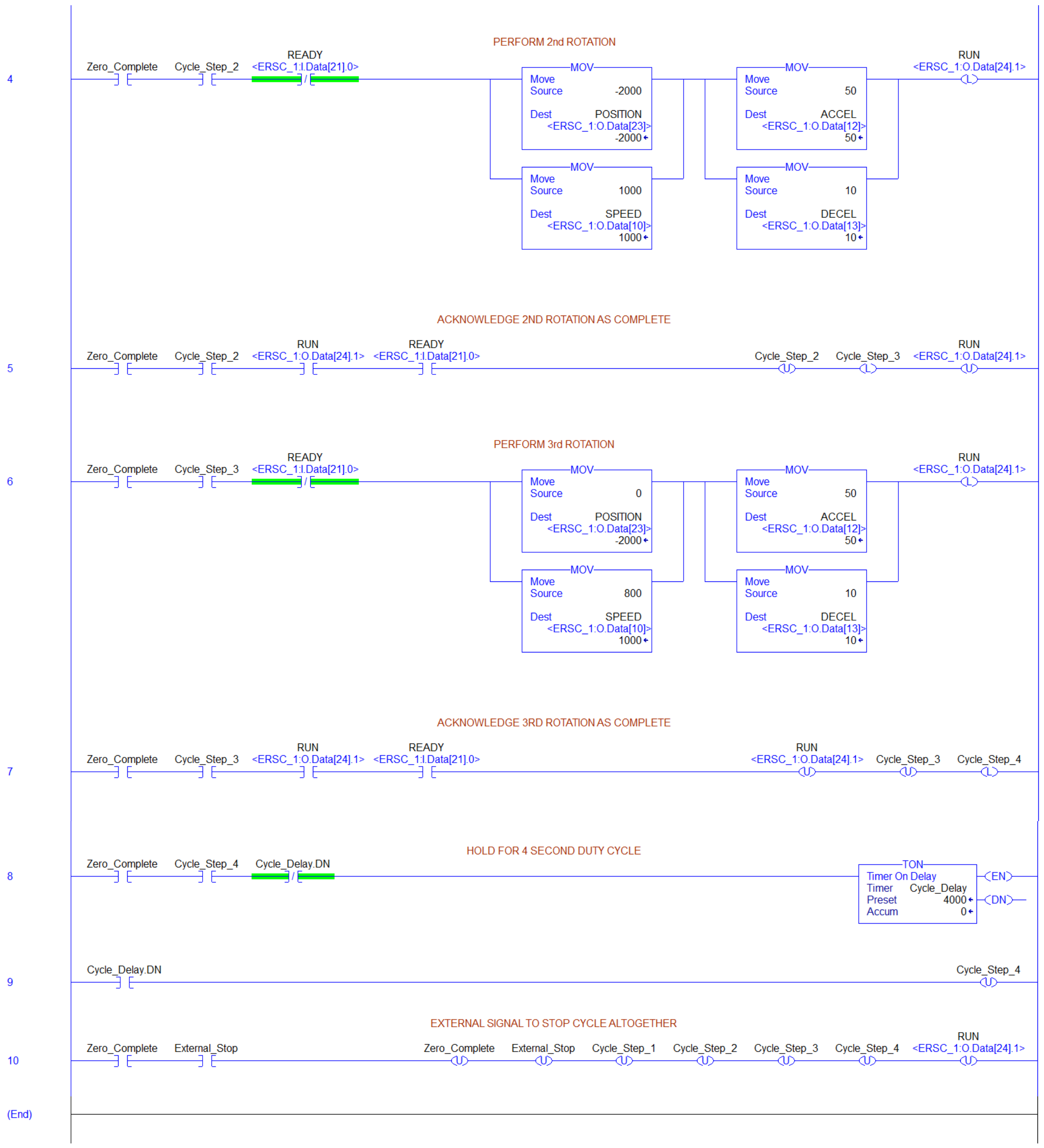
Step #7: Repeat *Step #3*

Rotation Note: Rotation from 0 to a positive value always causes rotation in CCW direction and rotation from 0 to a negative value always causes rotation in CW direction. Servo Motor Control ignores the CW/CCW direction setting in *EasyRoll* and/or any configured or default direction that may exist in the *ConveyLinx-Ai2*'s configuration.

Values shown in this example are for demonstration purposes only. Actual position, speeds, acceleration, and deceleration values are highly dependent upon the mechanical application.

PLC Ladder Diagram


ZERO THE LEFT MOTOR POSITION FROM EXTERNAL SIGNAL
0
External_Reset
READY <ERSC_1:I.Data[21].0>
ZERO <ERSC_1:O.Data[24].0>
External_Reset
1
ZERO <ERSC_1:O.Data[24].0>
ZERO_ACK <ERSC_1:I.Data[21].1>
ZERO <ERSC_1:O.Data[24].0>
Zero_Complete
PERFORM FIRST ROTATION
2
Zero_Complete
Cycle_Step_1
Cycle_Step_2
Cycle_Step_3
Cycle_Step_4
READY <ERSC_1:I.Data[21].0>
MOV Move Source 7000 Dest POSITION <ERSC_1:O.Data[23]> -2000
MOV Move Source 800 Dest SPEED <ERSC_1:O.Data[10]> 1000
MOV Move Source 100 Dest ACCEL <ERSC_1:O.Data[12]> 50
MOV Move Source 50 Dest DECEL <ERSC_1:O.Data[13]> 10
RUN <ERSC_1:O.Data[24].1>
Cycle_Step_1
ACKNOWLEDGE FIRST ROTATION AS COMPLETE
3
Zero_Complete
Cycle_Step_1
RUN <ERSC_1:O.Data[24].1>
READY <ERSC_1:I.Data[21].0>
Cycle_Step_1
RUN <ERSC_1:O.Data[24].1>
Cycle_Step_2


4
PERFORM 2nd ROTATION
Zero_Complete
Cycle_Step_2
READY <ERSC_1:I.Data[21].0>
MOV Move Source -2000 Dest POSITION <ERSC_1:O.Data[23]> -2000
MOV Move Source 1000 Dest SPEED <ERSC_1:O.Data[10]> 1000
MOV Move Source 50 Dest ACCEL <ERSC_1:O.Data[12]> 50
MOV Move Source 10 Dest DECEL <ERSC_1:O.Data[13]> 10
RUN <ERSC_1:O.Data[24].1> L
5
ACKNOWLEDGE 2ND ROTATION AS COMPLETE
Zero_Complete
Cycle_Step_2
RUN <ERSC_1:O.Data[24].1>
READY <ERSC_1:I.Data[21].0>
Cycle_Step_2 U
Cycle_Step_3 L
RUN <ERSC_1:O.Data[24].1> U
6
PERFORM 3rd ROTATION
Zero_Complete
Cycle_Step_3
READY <ERSC_1:I.Data[21].0>
MOV Move Source 0 Dest POSITION <ERSC_1:O.Data[23]> -2000
MOV Move Source 800 Dest SPEED <ERSC_1:O.Data[10]> 1000
MOV Move Source 50 Dest ACCEL <ERSC_1:O.Data[12]> 50
MOV Move Source 10 Dest DECEL <ERSC_1:O.Data[13]> 10
RUN <ERSC_1:O.Data[24].1> L
7
ACKNOWLEDGE 3RD ROTATION AS COMPLETE
Zero_Complete
Cycle_Step_3
RUN <ERSC_1:O.Data[24].1>
READY <ERSC_1:I.Data[21].0>
RUN <ERSC_1:O.Data[24].1> U
Cycle_Step_3 U
Cycle_Step_4 L
8
HOLD FOR 4 SECOND DUTY CYCLE
Zero_Complete
Cycle_Step_4
Cycle_Delay.DN
TON Timer On Delay Timer Cycle_Delay Preset 4000 Accum 0
EN
DN
9
Cycle_Delay.DN
Cycle_Step_4 U
10
EXTERNAL SIGNAL TO STOP CYCLE ALTOGETHER
Zero_Complete
External_Stop
Zero_Complete U
External_Stop U
Cycle_Step_1 U
Cycle_Step_2 U
Cycle_Step_3 U
Cycle_Step_4 U
RUN <ERSC_1:O.Data[24].1> U
(End)

Reduced Size PLC I/O Mode Assemblies

For some PLC controllers, the data size footprint required for a given assembly can be a limiting factor on how many devices can connect to a given controller. *ConveyLinx-Ai2* includes input/output assemblies with fewer registers that contain basic functions in applications where the full functionality of the PLC I/O Mode Assemblies is not required. The Reduced Size Assemblies provide basic I/O and motor control while removing the interfaces for servo moves, upstream/downstream data, tracking data, etc.

Reduced Size PLC I/O Mode Assembly Inputs

Please note that the functions provided by this assembly are selected from and described in detail in section *PLCI/O Mode Assembly Inputs* on page 49.

ERSC Internal Address	*Register Name*	*Assembled Address for PLC*
4:0035	Sensor & Control Port Inputs	**M: 4:2700** **E: I.Data[0]** **P: Byte 0 (Hi) Byte 1 (Lo)**
4:0036	Sensor Detect	**M: 4:2701** **E: I.Data[1]** **P: Byte 2 (Hi) Byte 3 (Lo)**
4:0057	Left Motor Temperature	**M: 4:2702** **E: I.Data[2]** **P: Byte 4 (Hi) Byte 5 (Lo)**
4:0058	Left Motor Status	**M: 4:2703** **E: I.Data[3]** **P: Byte 6 (Hi) Byte 7 (Lo)**
4:0081	Right Motor Temperature	**M: 4:2704** **E: I.Data[4]** **P: Byte 8 (Hi) Byte 9 (Lo)**
4:0082	Right Motor Status	**M: 4:2705** **E: I.Data[5]** **P: Byte 10 (Hi) Byte 11 (Lo)**
4:0060	Left Motor Port Digital I/O Status	**M: 4:2706** **E: I.Data[6]** **P: Byte 12 (Hi) Byte 13 (Lo)**
4:0084	Right Motor Port Digital I/O Status	**M: 4:2707** **E: I.Data[7]** **P: Byte 14 (Hi) Byte 15 (Lo)**
N/A	Reserved	**M: 4:2708** **E: I.Data[8]** **P: Byte 16 (Hi) Byte 17 (Lo)**

Reduced Size PLC I/O Mode Assembly Outputs

Please note that the functions provided by this assembly are selected from and described in detail in section *PLC I/O Mode Assembly Outputs* on page 53.

ERSC Internal Address	*Register Name*	*Assembled Address for PLC*
4:0060	Set Left Motor Port Digital Control	**M: 4:2800** **E: O.Data[0]** **P: Byte 0 (Hi) Byte 1 (Lo)**
4:0084	Set Right Motor Port Digital Control	**M: 4:2801** **E: O.Data[1]** **P: Byte 2 (Hi) Byte 3 (Lo)**
4:0037	Control Port Digital Output Control	**M: 4:2802** **E: O.Data[2]** **P: Byte 4 (Hi) Byte 5 (Lo)**
4:0260	Left Motor Run / Reverse	**M: 4:2803** **E: O.Data[3]** **P: Byte 6 (Hi) Byte 7 (Lo)**
4:0270	Right Motor Run / Reverse	**M: 4:2804** **E: O.Data[4]** **P: Byte 8 (Hi) Byte 9 (Lo)**
4:0040	Left Motor Speed Reference	**M: 4:2805** **E: O.Data[5]** **P: Byte 10 (Hi) Byte 11 (Lo)**
4:0064	Right Motor Speed Reference	**M: 4:2806** **E: O.Data[6]** **P: Byte 12 (Hi) Byte 13 (Lo)**
4:0022	Clear Motor Error	**M: 4:2807** **E: O.Data[7]** **P: Byte 14 (Hi) Byte 15 (Lo)**
N/A	Reserved	**M: 4:2808** **E: O.Data[8]** **P: Byte 16 (Hi) Byte 17 (Lo)**

ConveyLogix Interface Assemblies

ConveyLogix Version 2.02 and later is required for creating ConveyLogix programs that can make the data available for access by these assemblies.

ConveyLinx-Ai2s in PLC I/O mode allow for download of a user program generated with the *ConveyLogix* programming tool. There may be instances when you would like for an external PLC to be able to interface with a *ConveyLinx-Ai2* that is running a *ConveyLogix* program. Because a custom user generated program is running in the *ConveyLinx-Ai2*, the previously defined existing Assemblies for ZPA and PLO I/O modes are no longer valid and if used, could produce unexpected results.
When using *ConveyLogix* programming software version 2.2 and later, the environment provides predefined controlled input and output register tags that can be used by the programmer to interface with an external PLC. The Assemblies described in this section are those that the external PLC would connect and use to access the data in the *ConveyLinx-Ai2 ConveyLogix* program.
This data is divided into 16 Integer Input words and 16 Integer output words. It is left to the *ConveyLogix* programmer and the PLC programmer to utilize as much or as little of these data blocks as needed for their particular application.

ConveyLogix Assembly Inputs

ERSC ConveyLogix Controller Tag	*Assembled EIP PLC Tag*	*Assembled Modbus PLC Register*	*Assembled Profinet I/O*
ToPLCArray[0]	**E: I.Data[0]**	**M: 4:13100**	**P: Byte 0 (Hi) Byte 1 (Lo)**
ToPLCArray[1]	**E: I.Data[1]**	**M: 4:13101**	**P: Byte 2 (Hi) Byte 3 (Lo)**
ToPLCArray[2]	**E: I.Data[2]**	**M: 4:13102**	**P: Byte 4 (Hi) Byte 5 (Lo)**
ToPLCArray[3]	**E: I.Data[3]**	**M: 4:13103**	**P: Byte 6 (Hi) Byte 7 (Lo)**
ToPLCArray[4]	**E: I.Data[4]**	**M: 4:13104**	**P: Byte 8 (Hi) Byte 9 (Lo)**
ToPLCArray[5]	**E: I.Data[5]**	**M: 4:13105**	**P: Byte 10 (Hi) Byte 11 (Lo)**
ToPLCArray[6]	**E: I.Data[6]**	**M: 4:13106**	**P: Byte 12 (Hi) Byte 13 (Lo)**
ToPLCArray[7]	**E: I.Data[7]**	**M: 4:13107**	**P: Byte 14 (Hi) Byte 15 (Lo)**
ToPLCArray[8]	**E: I.Data[8]**	**M: 4:13108**	**P: Byte 16 (Hi) Byte 17 (Lo)**
ToPLCArray[9]	**E: I.Data[9]**	**M: 4:13109**	**P: Byte 18 (Hi) Byte 19 (Lo)**
ToPLCArray[10]	**E: I.Data[10]**	**M: 4:13110**	**P: Byte 20 (Hi) Byte 21 (Lo)**
ToPLCArray[11]	**E: I.Data[11]**	**M: 4:13111**	**P: Byte 22 (Hi) Byte 23 (Lo)**
ToPLCArray[12]	**E: I.Data[12]**	**M: 4:13112**	**P: Byte 24 (Hi) Byte 25 (Lo)**
ToPLCArray[13]	**E: I.Data[13]**	**M: 4:13113**	**P: Byte 26 (Hi) Byte 27 (Lo)**
ToPLCArray[14]	**E: I.Data[14]**	**M: 4:13114**	**P: Byte 28 (Hi) Byte 29 (Lo)**
ToPLCArray[15]	**E: I.Data[15]**	**M: 4:13115**	**P: Byte 30 (Hi) Byte 31 (Lo)**

ConveyLogix Assembly Outputs

ERSC ConveyLogix Controller Tag	*Assembled EIP PLC Tag*	*Assembled Modbus PLC Register*	*Assembled Profinet I/O*
FromPLCArray[0]	**E: O.Data[0]**	**M: 4:13200**	**P: Byte 0 (Hi) Byte 1 (Lo)**
FromPLCArray[1]	**E: O.Data[1]**	**M: 4:13201**	**P: Byte 2 (Hi) Byte 3 (Lo)**
FromPLCArray[2]	**E: O.Data[2]**	**M: 4:13202**	**P: Byte 4 (Hi) Byte 5 (Lo)**
FromPLCArray[3]	**E: O.Data[3]**	**M: 4:13203**	**P: Byte 6 (Hi) Byte 7 (Lo)**
FromPLCArray[4]	**E: O.Data[4]**	**M: 4:13204**	**P: Byte 8 (Hi) Byte 9 (Lo)**
FromPLCArray[5]	**E: O.Data[5]**	**M: 4:13205**	**P: Byte 10 (Hi) Byte 11 (Lo)**
FromPLCArray[6]	**E: O.Data[6]**	**M: 4:13206**	**P: Byte 12 (Hi) Byte 13 (Lo)**
FromPLCArray[7]	**E: O.Data[7]**	**M: 4:13207**	**P: Byte 14 (Hi) Byte 15 (Lo)**
FromPLCArray[8]	**E: O.Data[8]**	**M: 4:13208**	**P: Byte 16 (Hi) Byte 17 (Lo)**
FromPLCArray[9]	**E: O.Data[9]**	**M: 4:13209**	**P: Byte 18 (Hi) Byte 19 (Lo)**
FromPLCArray[10]	**E: O.Data[10]**	**M: 4:13210**	**P: Byte 20 (Hi) Byte 21 (Lo)**
FromPLCArray[11]	**E: O.Data[11]**	**M: 4:13211**	**P: Byte 22 (Hi) Byte 23 (Lo)**
FromPLCArray[12]	**E: O.Data[12]**	**M: 4:13212**	**P: Byte 24 (Hi) Byte 25 (Lo)**
FromPLCArray[13]	**E: O.Data[13]**	**M: 4:13213**	**P: Byte 26 (Hi) Byte 27 (Lo)**
FromPLCArray[14]	**E: O.Data[14]**	**M: 4:13214**	**P: Byte 28 (Hi) Byte 29 (Lo)**
FromPLCArray[15]	**E: O.Data[15]**	**M: 4:13215**	**P: Byte 30 (Hi) Byte 31 (Lo)**

ZPA and PLC I/O Mode Assemblies with Reset Protection

Please note that this functionality is only applicable for Ethernet I/P and Modbus TCP PLCs. Functionality is not applicable or used with Profinet I/O PLCs.

For control system applications where the Ethernet I/P PLC needs to take specific action to recover from a loss of communications due to an *ConveyLinx-Ai2* module 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 *ConveyLinx-Ai2* module for recovery.

When a *ConveyLinx-Ai2* in ZPA Mode has lost power and then is powered back up, due to perhaps a system E-stop that disconnects control power, some of the *ConveyLinx-Ai2*'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 *ConveyLinx-Ai2*. Upon power cycle to the *ConveyLinx-Ai2*, if the PLC can establish its full Ethernet I/P connection prior to the ZPA task becoming fully functional, the preceding PLC programming examples 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 *ConveyLinx-Ai2*'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 *ConveyLinx-Ai2* due to power cycle and because the PLC has not established communications to set the accumulate command bit, the *ConveyLinx-Ai2* releases the zone because there is no command present to accumulate.

For *ConveyLinx-Ai2*'s in PLC I/O mode, Reset Protection may not be as much a concern as for a module in ZPA mode, however reset protection assemblies are available for PLC I/O mode. Some items such as current servo position, etc. are reset upon restoration of power and thus the PLC programmer may want to detect this condition and act accordingly.

Reset Protection assemblies are used for applications where the *ConveyLinx-Ai2* remains in a “hold” state until the PLC has established communications. Otherwise, the register mappings for these assemblies are the same as their non-protected counterparts.

Please note that the Modbus TCP starting addresses for each assembly with reset protection is different from their non-protected counterparts.

ZPA Mode Assembly Inputs with Reset Protection

ERSC Internal Address	Register Name	Assembled Ethernet I/P Address for PLC	Assembled Modbus Address for PLC
4:0116	Local Status Upstream Zone	E: I.Data[0]	M: 4:3500
4:0196	Local Status Downstream Zone	E: I.Data[1]	M: 4:3501
4:0106	Arrival Count Local Upstream Zone	E: I.Data[2]	M: 4:3502
4:0107	Departure Count Local Upstream Zone	E: I.Data[3]	M: 4:3503
4:0186	Arrival Count Local Downstream Zone	E: I.Data[4]	M: 4:3504
4:0187	Departure Count Local Downstream Zone	E: I.Data[5]	M: 4:3505
4:0088	Module Status Word 1	E: I.Data[6]	M: 4:3506
4:0089	Module Status Word 2	E: I.Data[7]	M: 4:3507
4:0119	Current Upstream Zone Tracking Word 1	E: I.Data[8]	M: 4:3508
4:0120	Current Upstream Zone Tracking Word 2	E: I.Data[9]	M: 4:3509
4:0199	Current Downstream Zone Tracking Word 1	E: I.Data[10]	M: 4:3510
4:0200	Current Downstream Zone Tracking Word 2	E: I.Data[11]	M: 4:3511
4:0105	Current Release Count for Upstream Zone	E: I.Data[12]	M: 4:3512
4:0185	Current Release Count for Downstream Zone	E: I.Data[13]	M: 4:3513
4:0201	Get Tracking Forward Direction Word 1	E: I.Data[14]	M: 4:3514
4:0202	Get Tracking Forward Direction Word 2	E: I.Data[15]	M: 4:3515
4:0237	Get Tracking Reverse Direction Word 1	E: I.Data[16]	M: 4:3516
4:0238	Get Tracking Reverse Direction Word 2	E: I.Data[17]	M: 4:3517
4:0035	Sensor & Control Port Inputs	E: I.Data[18]	M: 4:3518
N/A	**Current Module Reset Count**	E: I.Data[19]	M: 4:3519
4:0019	ConveyStop Status	E: I.Data[20]	M: 4:3520

ZPA Mode Assembly Outputs with Reset Protection

ERSC Internal Address	Register Name	Assembled Ethernet I/P Address for PLC	Assembled Modbus Address for PLC
4:0132	Set Local Upstream Zone Tracking Word 1	E: O.Data[0]	M: 4:3600
4:0133	Set Local Upstream Zone Tracking Word 2	E: O.Data[1]	M: 4:3601
4:0212	Set Local Downstream Zone Tracking Word 1	E: O.Data[2]	M: 4:3602
4:0213	Set Local Downstream Zone Tracking Word 2	E: O.Data[3]	M: 4:3603
4:0104	Accumulation Control for Local Upstream Zone	E: O.Data[4]	M: 4:3604
4:0184	Accumulation Control for Local Downstream Zone	E: O.Data[5]	M: 4:3605
4:0040	Set Left MDR Speed	E: O.Data[6]	M: 4:3606
4:0064	Set Right MDR Speed	E: O.Data[7]	M: 4:3607
4:0105	Release and Accumulate on Next Arrival for Local Upstream Zone	E: O.Data[8]	M: 4:3608
4:0185	Release and Accumulate on Next Arrival for Local Downstream Zone	E: O.Data[9]	M: 4:3609
4:0134	Set Status for Upstream Induct	E: O.Data[10]	M: 4:3610
4:0232	Set Status for Downstream Discharge	E: O.Data[11]	M: 4:3611
4:0139	Set Induct Tracking Forward Direction Word 1	E: O.Data[12]	M: 4:3612
4:0140	Set Induct Tracking Forward Direction Word 2	E: O.Data[13]	M: 4:3613
4:0237	Set Induct Tracking Reverse Direction Word 1	E: O.Data[14]	M: 4:3614
4:0238	Set Induct Tracking Forward Direction Word 2	E: O.Data[15]	M: 4:3615
4:0022	Clear Motor Error	E: O.Data[16]	M: 4:3616
4:0063	Set Control Port Outputs	E: O.Data[17]	M: 4:3617
N/A	**Set Module Reset Counter**	**E: O.Data[18]**	**M: 4:3618**
4:0020	ConveyStop Command	E: O.Data[19]	M: 4:3619
4:0109	Clear Sensor Jam Command for Local Upstream Zone	E: O.Data[20]	M: 4:3620
4:0189	Clear Sensor Jam Command for Local Downstream Zone	E: O.Data[21]	M: 4:3621
4:0365	Direction & Accumulation Mode Control for Local Upstream Zone	E: O.Data[22]	M: 4:3622
4:0375	Direction & Accumulation Mode Control for Local Downstream Zone	E: O.Data[23]	M: 4:3623
4:0387	ConveyMerge Interface	E: O.Data[24]	M: 4:3624

PLC I/O Mode Assembly Inputs with Reset Protection

ERSC Internal Address	*Register Name*	*Assembled Ethernet I/P Address for PLC*	*Assembled Modbus Address for PLC*
4:0019	ConveyStop Status	E: I.Data[0]	M: 4:3700
4:0035	Sensor & Control Port Inputs	E: I.Data[1]	M: 4:3701
4:0036	Sensor Detect	E: I.Data[2]	M: 4:3702
4:0024	Module Voltage	E: I.Data[3]	M: 4:3703
4:0055	Left Motor Current	E: I.Data[4]	M: 4:3704
4:0056	Left Motor Frequency	E: I.Data[5]	M: 4:3705
4:0057	Left Motor Temperature	E: I.Data[6]	M: 4:3706
4:0058	Left Motor Status	E: I.Data[7]	M: 4:3707
4:0079	Right Motor Current	E: I.Data[8]	M: 4:3708
4:0080	Right Motor Frequency	E: I.Data[9]	M: 4:3709
4:0081	Right Motor Temperature	E: I.Data[10]	M: 4:3710
4:0082	Right Motor Status	E: I.Data[11]	M: 4:3711
4:0060	Left Motor Port Digital I/O Status	E: I.Data[12]	M: 4:3712
4:0084	Right Motor Port Digital I/O Status	E: I.Data[13]	M: 4:3713
4:0134	Upstream Module Status	E: I.Data[14]	M: 4:3714
4:0232	Downstream Module Status	E: I.Data[15]	M: 4:3715
4:0139	Current Tracking Word 1 for Adjacent Upstream Module	E: I.Data[16]	M: 4:3716
4:0140	Current Tracking Word 2 for Adjacent Upstream Module	E: I.Data[17]	M: 4:3717
N/A	**Current Module Reset Counter**	**E: I.Data[18]**	**M: 4:3718**
4:0062	Left Motor Servo Position	E: I.Data[19]	M: 4:3719
4:0086	Right Motor Servo Position	E: I.Data[20]	M: 4:3720
4:0011	Left Motor Servo Status	E: I.Data[21]	M: 4:3721
4:0016	Right Motor Servo Status	E: I.Data[22]	M: 4:3722

PLC I/O Mode Assembly Outputs with Reset Protection

ERSC Internal Address	*Register Name*	*Assembled Ethernet I/P Address for PLC*	*Assembled Modbus Address for PLC*
4:0020	ConveyStop Command	E: O.Data[0]	M: 4:3800
4:0060	Set Left Motor Port Digital Control	E: O.Data[1]	M: 4:3801
4:0084	Set Right Motor Port Digital Control	E: O.Data[2]	M: 4:3802
4:0037	Control Port Digital Output Control	E: O.Data[3]	M: 4:3803
4:0260	Left Motor Run / Reverse	E: O.Data[4]	M: 4:3804
4:0261	Left Motor Brake Method	E: O.Data[5]	M: 4:3805
4:0262	Left Motor Speed Control Method	E: O.Data[6]	M: 4:3806
4:0270	Right Motor Run / Reverse	E: O.Data[7]	M: 4:3807
4:0271	Right Motor Brake Method	E: O.Data[8]	M: 4:3808
4:0272	Right Motor Speed Control Method	E: O.Data[9]	M: 4:3809
4:0040	Left Motor Speed Reference	E: O.Data[10]	M: 4:3810
4:0064	Right Motor Speed Reference	E: O.Data[11]	M: 4:3811
4:0043	Left Motor Acceleration Ramp	E: O.Data[12]	M: 4:3812
4:0044	Left Motor Deceleration Ramp	E: O.Data[13]	M: 4:3813
4:0067	Right Motor Acceleration Ramp	E: O.Data[14]	M: 4:3814
4:0068	Right Motor Deceleration Ramp	E: O.Data[15]	M: 4:3815
4:0022	Clear Motor Error	E: O.Data[16]	M: 4:3816
4:0196	Set Status to Downstream Module	E: O.Data[17]	M: 4:3817
4:0116	Set Status to Upstream Module	E: O.Data[18]	M: 4:3818
4:0034	Sensor Port Input Signal Condition Mask	E: O.Data[19]	M: 4:3819
4:0201	Set Discharge Tracking Word 1	E: O.Data[20]	M: 4:3820
4:0202	Set Discharge Tracking Word 2	E: O.Data[21]	M: 4:3821
N/A	**Set Module Reset Count**	**E: O.Data[22]**	**M: 4:3822**
4:0008	Left Motor Servo Command Pulses	E: O.Data[23]	M: 4:3823
4:0009	Left Motor Servo Command Word	E: O.Data[24]	M: 4:3824
4:0013	Right Motor Servo Command Pulses	E: O.Data[25]	M: 4:3825
4:0014	Right Motor Servo Command Word	E: O.Data[26]	M: 4:3826

Reduced Size PLC I/O Mode Assembly Inputs with Reset Protection

ERSC Internal Address	Register Name	Assembled Ethernet I/P Address for PLC	Assembled Modbus Address for PLC
4:0035	Sensor & Control Port Inputs	E: I.Data[0]	M: 4:4700
4:0036	Sensor Detect	E: I.Data[1]	M: 4:4701
4:0057	Left Motor Temperature	E: I.Data[2]	M: 4:4702
4:0058	Left Motor Status	E: I.Data[3]	M: 4:4703
4:0081	Right Motor Temperature	E: I.Data[4]	M: 4:4704
4:0082	Right Motor Status	E: I.Data[5]	M: 4:4705
4:0060	Left Motor Port Digital I/O Status	E: I.Data[6]	M: 4:4706
4:0084	Right Motor Port Digital I/O Status	E: I.Data[7]	M: 4:4707
N/A	**Current Module Reset Count**	E: I.Data[8]	M: 4:4708

Reduced Size PLC I/O Mode Assembly Outputs with Reset Protection

ERSC Internal Address	Register Name	Assembled Ethernet I/P Address for PLC	Assembled Modbus Address for PLC
4:0060	Set Left Motor Port Digital Control	E: O.Data[0]	M: 4:4800
4:0084	Set Right Motor Port Digital Control	E: O.Data[1]	M: 4:4801
4:0037	Control Port Digital Output Control	E: O.Data[2]	M: 4:4802
4:0260	Left Motor Run / Reverse	E: O.Data[3]	M: 4:4803
4:0270	Right Motor Run / Reverse	E: O.Data[4]	M: 4:4804
4:0040	Left Motor Speed Reference	E: O.Data[5]	M: 4:4805
4:0064	Right Motor Speed Reference	E: O.Data[6]	M: 4:4806
4:0022	Clear Motor Error	E: O.Data[7]	M: 4:4807
N/A	**Set Module Reset Count**	**E: O.Data[8]**	**M: 4:4808**

Reset Protection Usage with the PLC

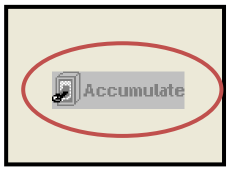
When the PLC determines that it has a valid connection to the *ConveyLinx-Ai2* in question and is ready for it to respond to output data being written by the PLC:

- Your PLC program needs to move the value in the *Current Module Reset Counter* register into the *Set Module Reset Counter* register.
- When the *ConveyLinx-Ai2* detects that the value in *Set Module Reset Counter* register is equal to the value in the *Current Module Reset Counter* register; the *ConveyLinx-Ai2* will respond to data being written by the PLC to the remaining output assembly registers.

Please note that when the value in *Current Module Reset Counter* register is not equal to the value in the *Set Module Reset Counter* register; the values in the Input registers will updated by the *ConveyLinx-Ai2* and will be valid. In this state, even though the PLC may be writing data to Output registers, the *ConveyLinx-Ai2* will ignore it.

FOR ZPA ZONES: To make sure a given 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 *ConveyLinx-Ai2* 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 ZPA zone is commanded to accumulate with a PLC and you happen to connect to that *ConveyLinx-Ai2* with *EasyRoll*; the “Accumulate” switch icon on the 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 a PLC command to accumulate IS NOT retained upon power loss to the *ConveyLinx-Ai2*. Only if the “Accumulate” switch is toggled “ON” with the *EasyRoll* is the accumulate condition for the zone retained in flash memory for use upon power-up.

Appendix A - Motor Port as Digital I/O

Motor port as digital I/O is only applicable for a ConveyLinx-Ai2 in PLC I/O Mode

Each motor port can provide 1 or 2 independently controlled digital outputs and each of these outputs can be energized simultaneously for a total of 4 outputs available per ConveyLinx-Ai2 module. Each individual output has a 0.75A load capacity.

External controller must first set bit 15 = 1 in the *Set Left/Right Motor Port Digital Control* register for the motor port (Left or Right) that is to be used as digital output. If bit 15 = 0, then the *ConveyLinx-Ai2* ignores the bit 0 thru bit 2 commands and will not provide meaningful status on bits 12 and 14 in the corresponding *Left/Right Motor Port Digital I/O Status* register for the port in question.

Please note that the connection Pin numbers for the Left and Right Motor Ports are NOT the same

Figure 9 shows the wiring diagram for the LEFT MOTOR PORT ONLY for digital output. Please note that the ConveyLinx-Ai2 switches to GND to complete the circuit.

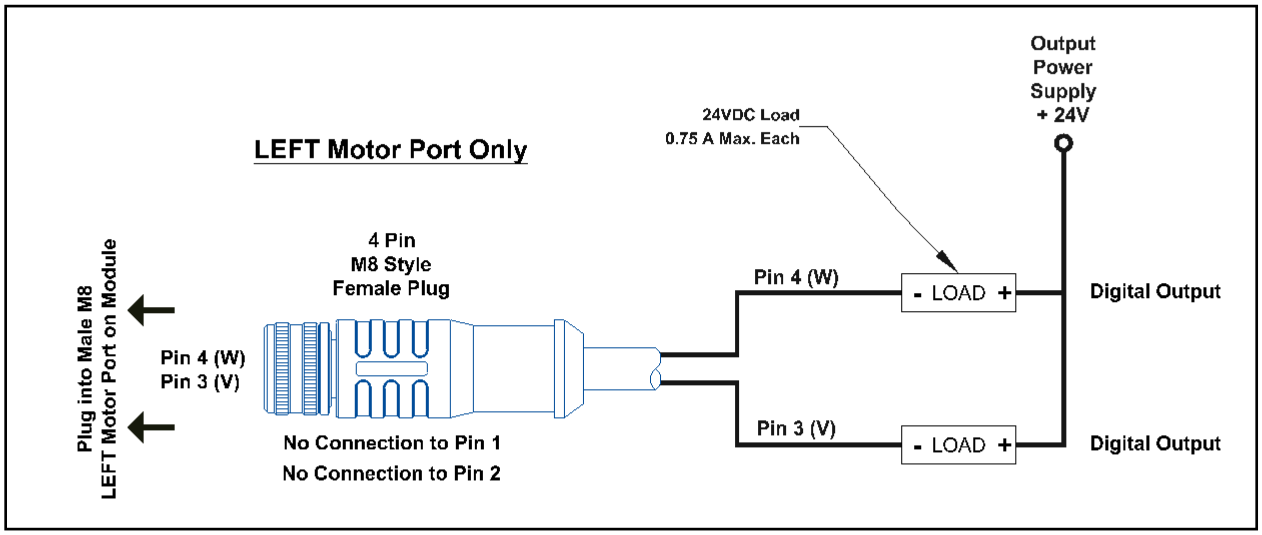


FIGURE 9 - WIRING DIAGRAM FOR LEFT MOTOR PORT CONFIGURED FOR DIGITAL OUTPUT

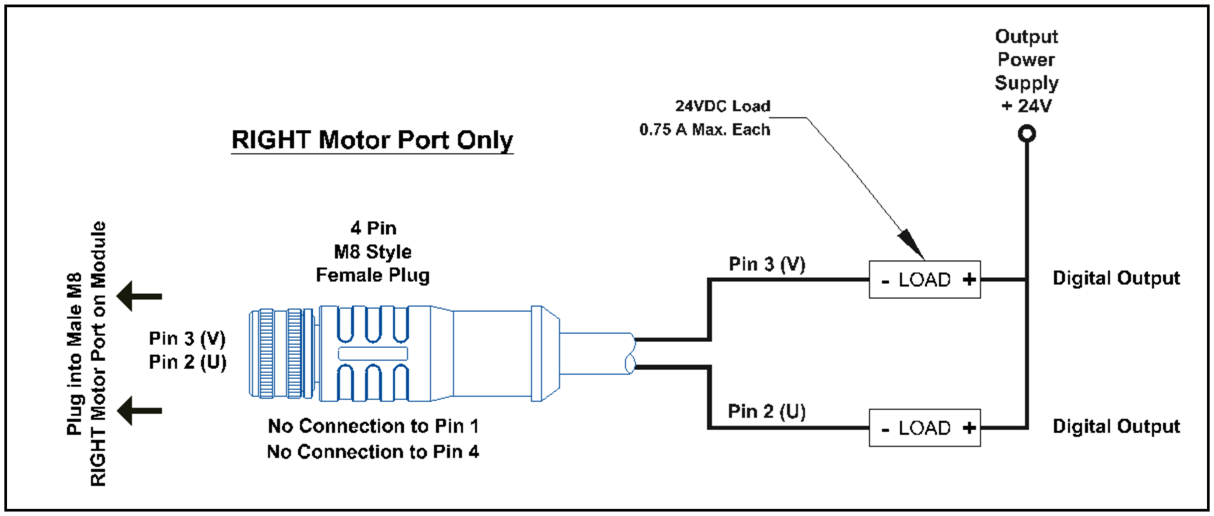


FIGURE 10 - WIRING DIAGRAM FOR RIGHT MOTOR PORT CONFIGURED FOR DIGITAL OUTPUT

Short Circuit Error on bit 12 is classified as a "fatal" error that will require either a cycle of power on the *ConveyLinx-Ai2* 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.

Appendix B – ConveyLinx Connections

ZPA Mode Register Data Flow

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

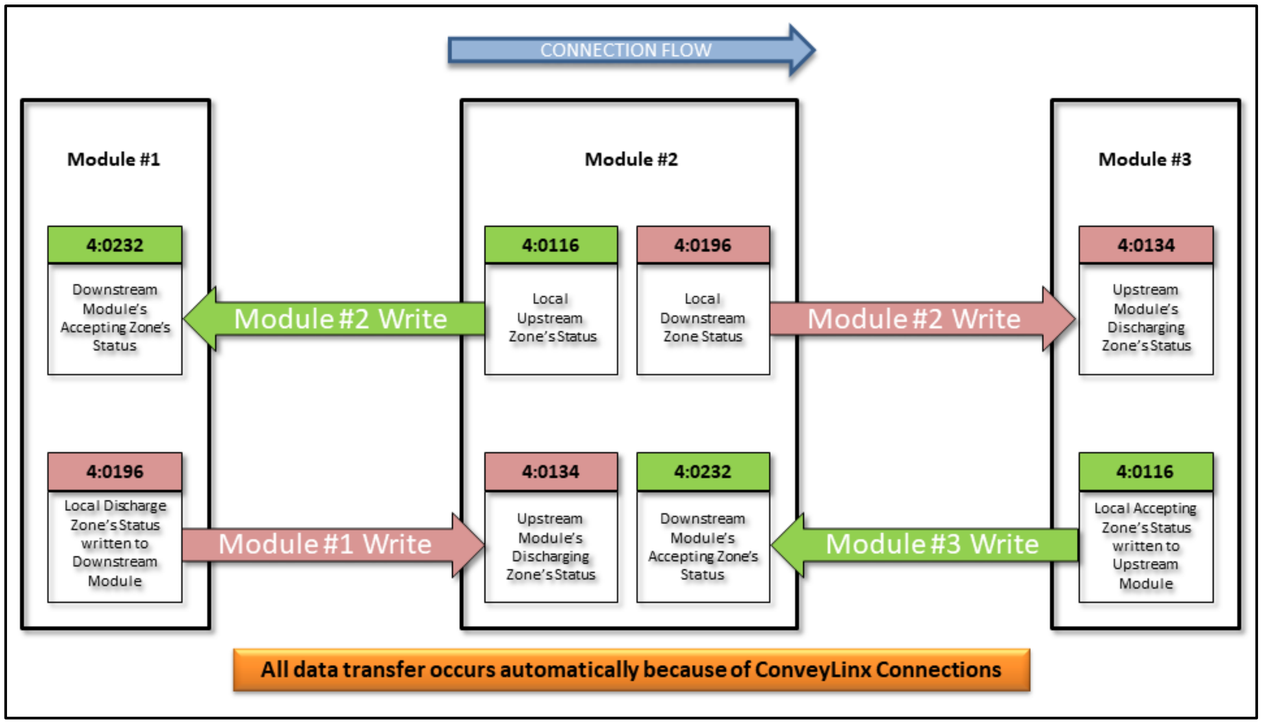

CONNECTION FLOW
Module #1
Module #2
Module #3
4:0232
Downstream Module's Accepting Zone's Status
Module #2 Write
4:0116
Local Upstream Zone's Status
4:0196
Local Downstream Zone Status
Module #2 Write
4:0134
Upstream Module's Discharging Zone's Status
4:0196
Local Discharge Zone's Status written to Downstream Module
Module #1 Write
4:0134
Upstream Module's Discharging Zone's Status
4:0232
Downstream Module's Accepting Zone's Status
Module #3 Write
4:0116
Local Accepting Zone's Status written to Upstream Module
All data transfer occurs automatically because of ConveyLinx Connections

FIGURE 11 - *MODULE* 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 11* contain integer values indicating a zone status. In essence, the ZPA control process in each *Module* writes its zone's status to its adjacent neighbors and monitors its local registers that are written to by its adjacent *Module* 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:

- *Module* #1 continually reads register 4:0232 to know the status of *Module* #2's local upstream zone

- If a Carton arrives at *Module* #1 discharge zone and the value in its register 4:0232 is equal to 4 or 5, then it will stop the Carton at its downstream discharge zone
- *Module* #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 Carton in its discharge zone can be conveyed to *Module* #2. To initiate this, *Module* #1 writes its status value of 4 (located in register 4:0196) to register 4:0134 on *Module* #2.
- When *Module* #2 control logic sees its register 4:0134 change to a value of 4, it knows that a Carton is being discharged onto its upstream zone and thus will energize its upstream zone MDR to run and accept the Carton from *Module* #1.

There would be a similar sequence for *Module* #2 to convey a Carton to *Module* #3.

External Controller as ConveyLinx Bridge

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

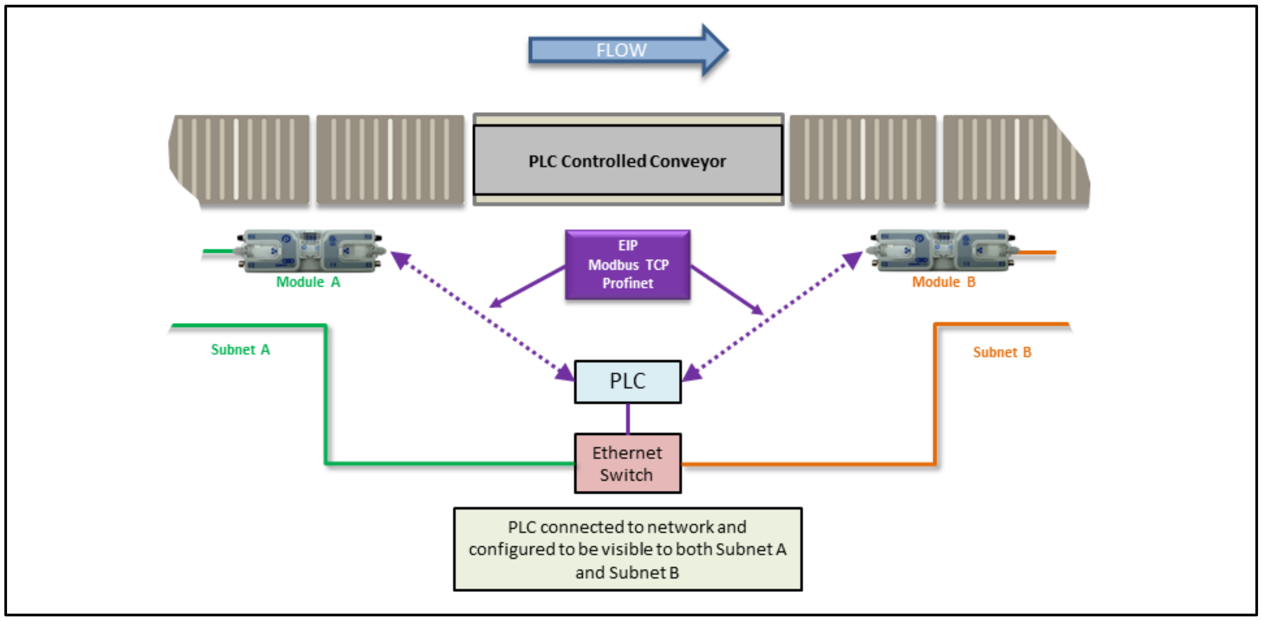


FIGURE 12 - PLC *CONVEYLINX* BRIDGE

Figure 12 shows a simple example where a PLC would act as a bridge between two separate *ConveyLinx* subnets. In this case, *Module* A 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, *Module* B 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 *Module A,* through the PLC controlled conveyor, and eventually to *Module B*; the PLC must act as a bridge and "mimic" the writing of register data to *Module A* and *Module B* as if *Module A* and *Module B* have configured ConveyLinx connections. This is illustrated in *Figure 13*.

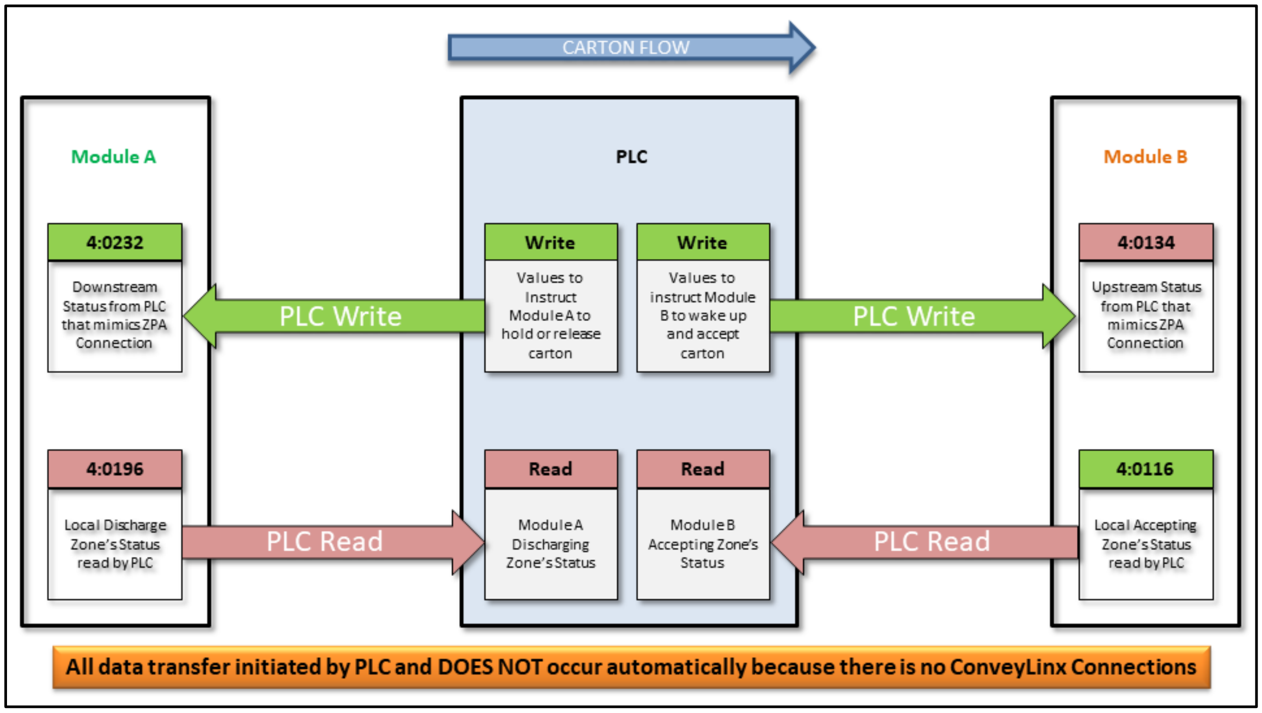

CARTON FLOW
Module A
PLC
Module B
4:0232
Downstream Status from PLC that mimics ZPA Connection
PLC Write
Write
Values to Instruct Module A to hold or release carton
Write
Values to instruct Module B to wake up and accept carton
PLC Write
4:0134
Upstream Status from PLC that mimics ZPA Connection
4:0196
Local Discharge Zone's Status read by PLC
PLC Read
Read
Module A Discharging Zone's Status
Read
Module B Accepting Zone's Status
PLC Read
4:0116
Local Accepting Zone's Status read by PLC
All data transfer initiated by PLC and DOES NOT occur automatically because there is no ConveyLinx Connections

FIGURE 13 – DATA FLOW CHART FOR PLC *CONVEYLINX* BRIDGE EXAMPLE

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

The PLC can monitor the status of *Module A* discharge zone by reading *Module A* register 4:0196. If this value is 5, the PLC knows a Carton is present and stopped in *Module A*'s discharge zone. When the PLC is ready to accept the Carton, it simply writes a value of 1 to *Module A* register 4:0232. To keep *Module A*'s discharge zone from generating an Arrival Jam, the PLC must write a value of 5 back to *Module A* register 4:0232 to indicate that the carton successfully arrived downstream.

For the discharge from the PLC controlled conveyor or machine, the PLC can monitor the status of the upstream zone of *Module B* by reading *Module B* register 4:0116. For example, if a value of 5 is read, the PLC will know that the *Module B* accepting zone is occupied and is not ready to accept a Carton. If the value read is a 1 or 2, then the PLC knows it can discharge a Carton when ready. In order to discharge a Carton, the PLC will need to write a value of 4 to *Module B* register 4:0134 to tell *Module B* to run its upstream accepting zone. When the PLC has detected (or by simple timeout) that the carton has left the PLC controlled conveyor, it must write a 1 to *Module B* register 4:0134 to indicate that the upstream carton cleared its sensor. If the PLC does not write a 1 to *Module B* register 4:0134 before the carton arrives at *Module B*'s accepting zone's sensor, *Module B* will believe the upstream ZPA discharging zones sensor is still blocked and thus trigger the Flex Zone feature.

NOTES:

North & South America

sales@pulseroller.com

support@pulseroller.com

www.pulseroller.com

Global

Global_sales@pulseroller.com

Global_support@pulseroller.com

Publication ERSC-1506 Rev 1.1 – September 2019