



AL1222

AL1122



P2M2HBVL OR P2HL IO-LINK
ADD-ON INSTRUCTION FOR ROCKWELL
PLC WITH IFM AL1122 & AL1222
ETHERNET/IP IO-LINK MASTER
QUICK START GUIDE

PREFACE

This Quick Start Guide (QSG) is designed to help integrate Parker Hannifin's P2M or P2H IO-Link valve manifold into an Allen-Bradley (AB) PLC environment utilizing the IFM AL1122 or AL1222 EIP IO-Link Master module. The QSG assumes that you are already using the IFM Generic_8PORT_IOL AOI for cyclic data for its master module and that it is communicating to the AB PLC via an Ethernet-IP network. You can find this AOI and instructions on how to implement it here:

[https://www.ifm.com/download/files/Startup%20Package_EIP_AL1x2x_Rev14/\\$file/Startup%20Package_EIP_AL1x2x_Rev14.zip](https://www.ifm.com/download/files/Startup%20Package_EIP_AL1x2x_Rev14/$file/Startup%20Package_EIP_AL1x2x_Rev14.zip)

The QSG is agnostic to IO Link Device Classification, such that it shall function the same whether you are controlling an A-Class or B-Class P2M / P2H Module. The guide will walk the user through obtaining the necessary files, importing/configuring the AOI, and initiating parameter reads and writes from/to the P2M / P2H IO-Link device.

The "P2M2HBVL_P2HL_AB_IFM_AL1x22_PRM_Rx" AOI facilitates the call-up of the acyclic service data.

The "P2M2HBVL_P2HL_AB_IFM_AL1x22_PD_Rx" AOI facilitates communication and handling of process data between PLC and the IO-Link slave device.

You can download resources such as the IODD configuration file, this QSG, a sample RSLogix5000 file and the full user manual here:

www.parker.com/PDN/io-link

Process Data Add-On Instruction Set Up

The “P2M2HBVL_P2HL_AB_IFM_AL1x22_PD_Rx” AOI simplifies the usage of Parker P2M and P2H IO-Link devices with Allen-Bradley CompactLogix, ControlLogix and GuardLogix PLCs when connected, via Ethernet/IP, to an IFM AL1x22 IO-Link Master. Data is mapped to user-friendly control and diagnostic tags on the PLC side.

The setup of the AOI below “Generic_8PORT_IOL” needs to be done first and the AOI can be sourced from the IFM website.

1. C.Port_Process_Data_Size should = AOI instruction Port_Process_Data_Size
2. Proper setup of port on IOL Master for IOL mode
3. Byte swap = off or 0
4. Data size set to integer in IFM module properties (image below)

**** Note Data size of 2 Bytes cannot be used.**

FIGURE 1 IFM AOI CONFIGURATION

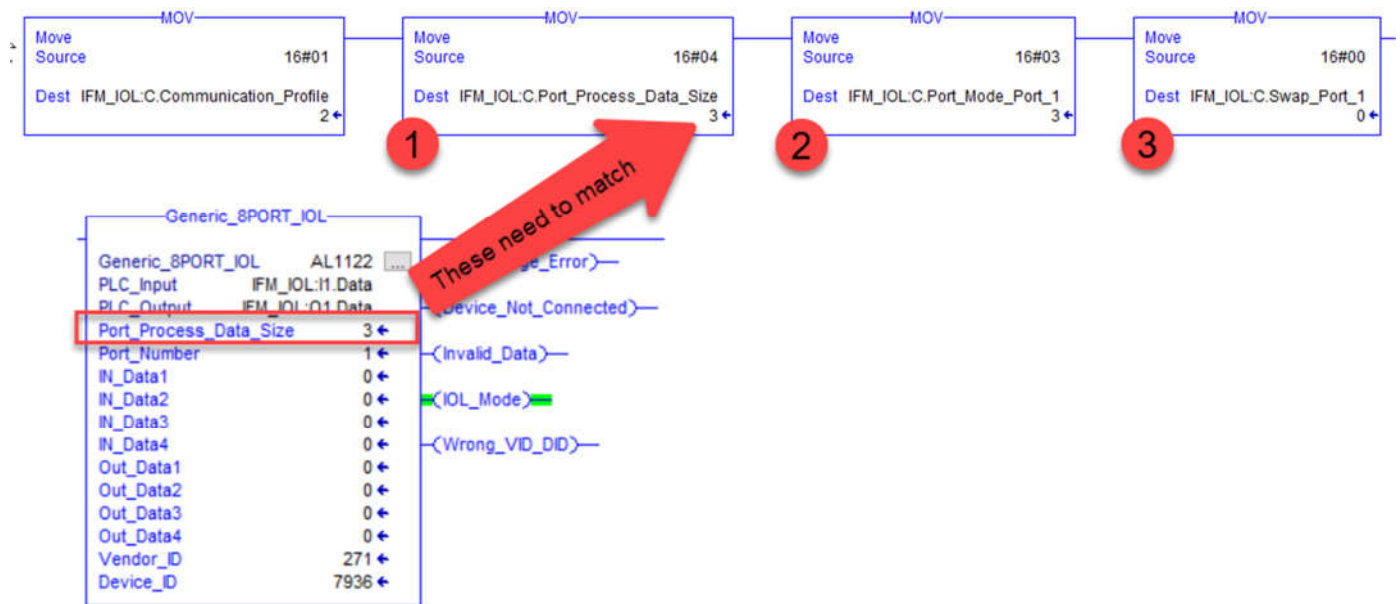


FIGURE 2 DATA TYPE SELECTION FOR IFM SHOULD BE SET TO INTEGER

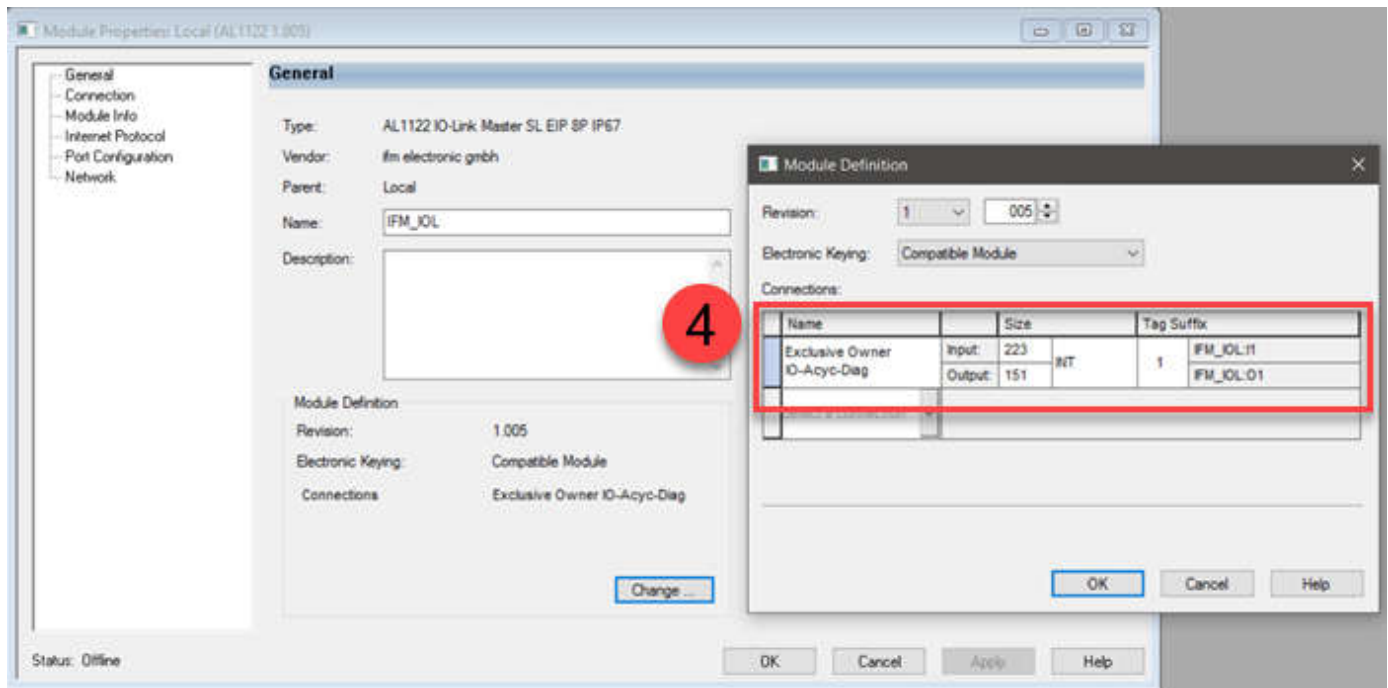
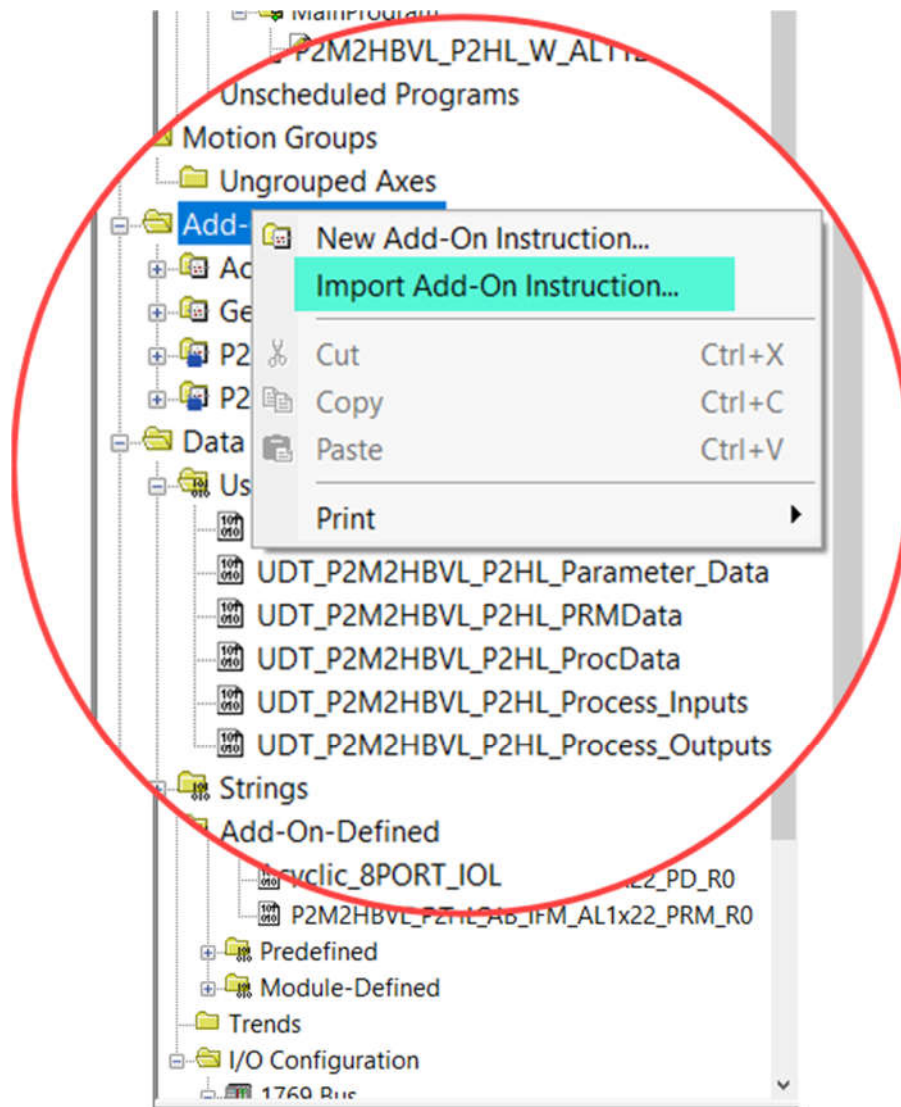




FIGURE 3 HOW TO IMPORT AOI

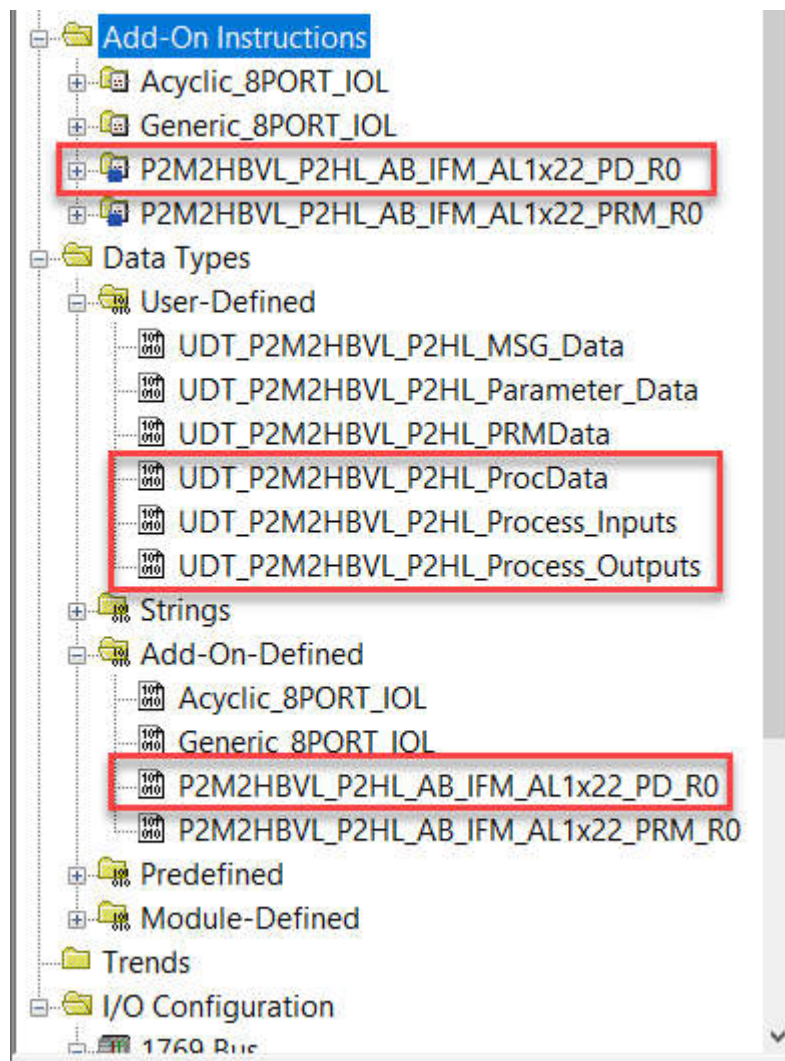


1. Right click Add-On Instruction in Controller Organizer and select “Import Add-On Instruction...”
2. Select the “P2M2HBVL_P2HL_AB_IFM_AL1x22_PD_Rx” where _Rx is the revision of AOI.

 P2M2HBVL_P2HL_AB_IFM_AL1x22_PD_R0.L5X		7/3/2019 4:12 PM	RSLogix 5000 XML ...
---	---	------------------	----------------------

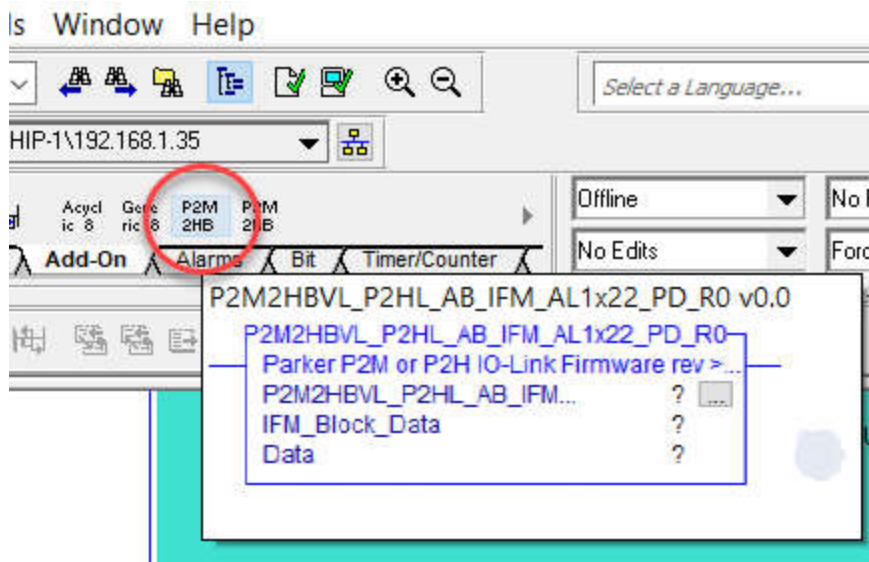
3. Choose OK on Import Configuration Window and you should then see the new AOI instance along with User-Defined and Add-On Defined data types created in the controller organizer.

FIGURE 4 LIST OF PROCESS DATA AOI'S AND DATA TYPES AFTER IMPORT



4. Add instance of AOI instruction to an empty rung of ladder by clicking on the P2M under the Add-On tab in the top toolbar. The instruction will drop onto the selected rung.

FIGURE 5 SELECT AOI TO ADD TO RUNG



5. Point the IFM_Block_Data field to the tag assigned in Generic_8PORT_IOL IFM AOI. If you choose the incorrect tag the AOI will NOT function and undesired results are likely. IO-Link is sensitive to the port which it has been assigned to communicate on.

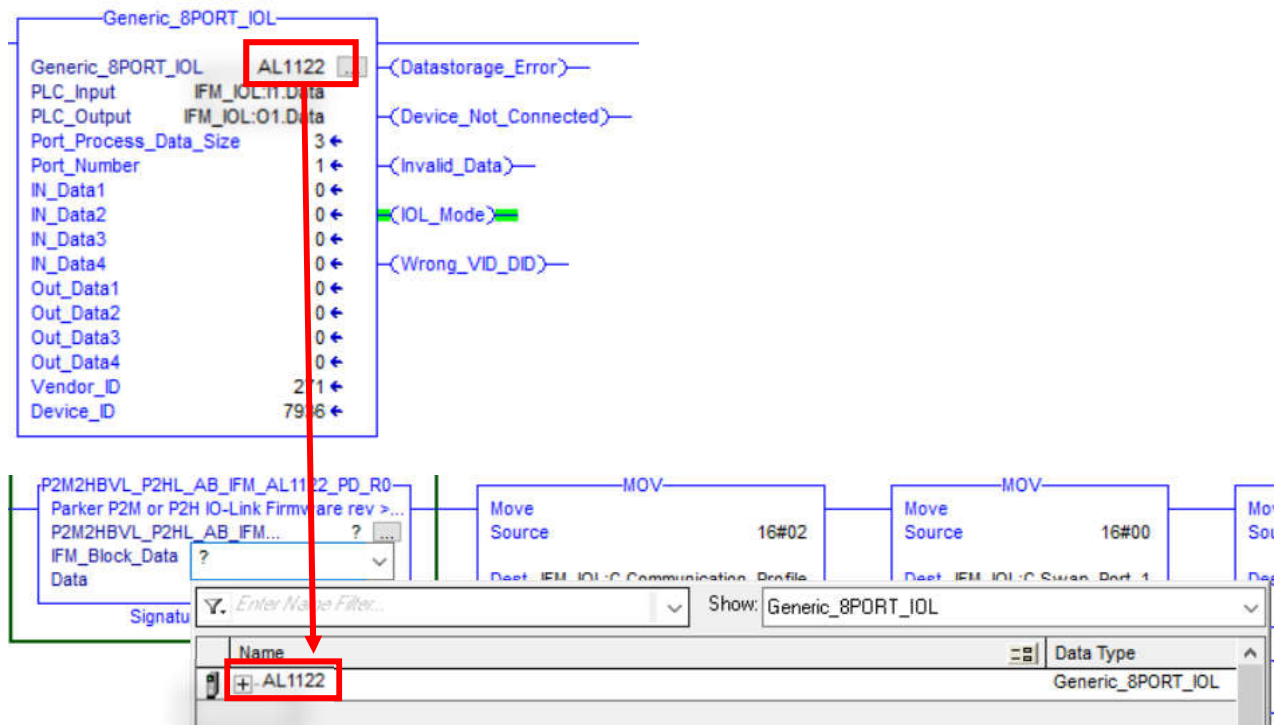
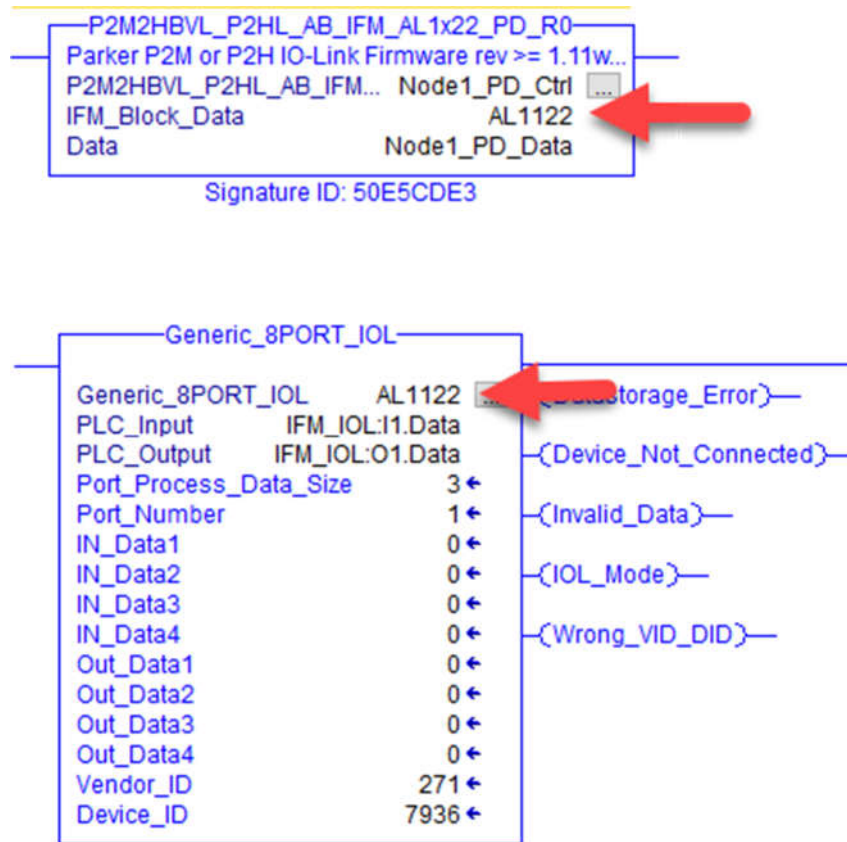


FIGURE 6 THESE TWO DATA TAGS SHOULD BE THE SAME



- Assign an instance name for the AOI and create other tags necessary for operation. Right click on the question marks and select "New Tag". Note that the name must be unique for each tag and each instance of the P2M / P2H AOI. The scope and data type fields will auto-populate with the correct values, so these should not need to be changed. All fields are required. See Appendix for structure breakdown of the "Data" variable.

FIGURE 7 CREATE TAG FOR AOI BLOCK

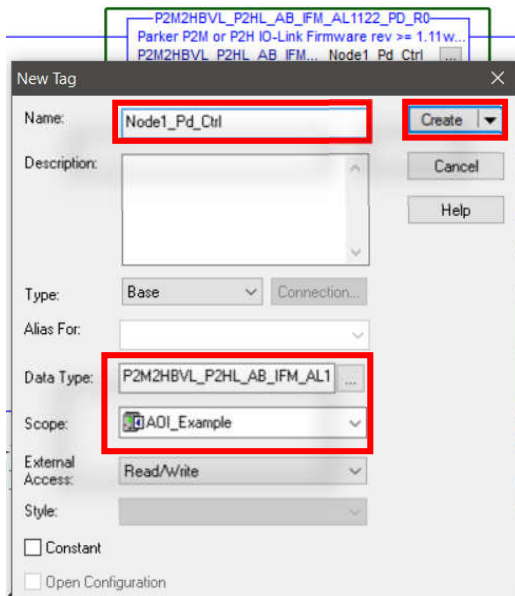
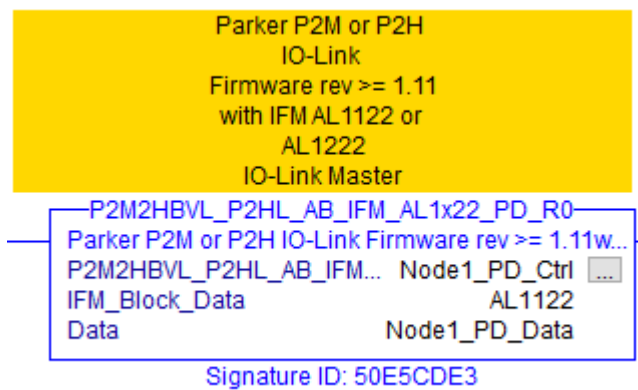


FIGURE 8 FULLY CONFIGURED AOI



USING THE INSTRUCTION

It is important to note the difference between cyclic and acyclic data. Process Data (cyclic) is updated without a request; whereas Parameter Data (acyclic) requires the program to toggle a bit to pull (or push) data. Cyclic data includes input status and valve output control. This means that Node1_PD_Data.Status.xxx and Node1_PD_Data.EV.##_ are live tags (containing real data) that exist simply because the AOI instruction was used. See appendix for all data points available. See ladder logic examples below:

FIGURE 9 EXAMPLE TOGGING SOLENOID VALVES (CYCLIC)

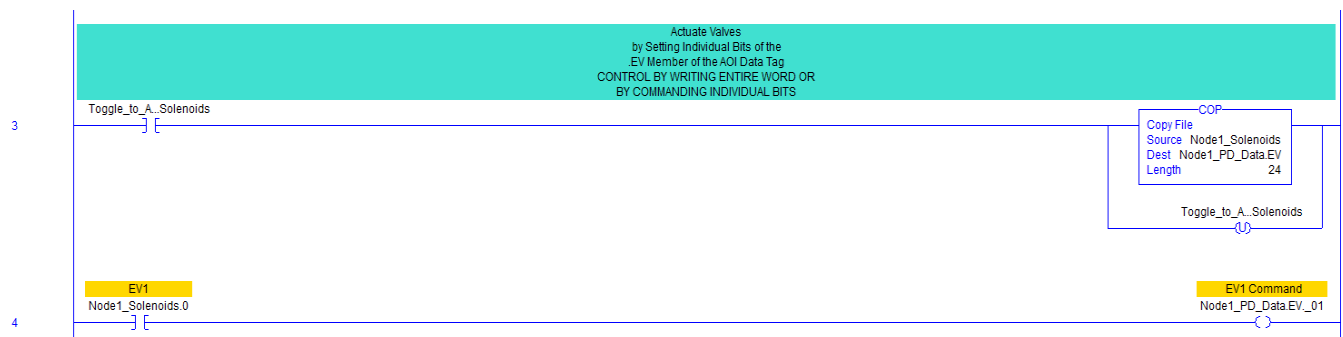
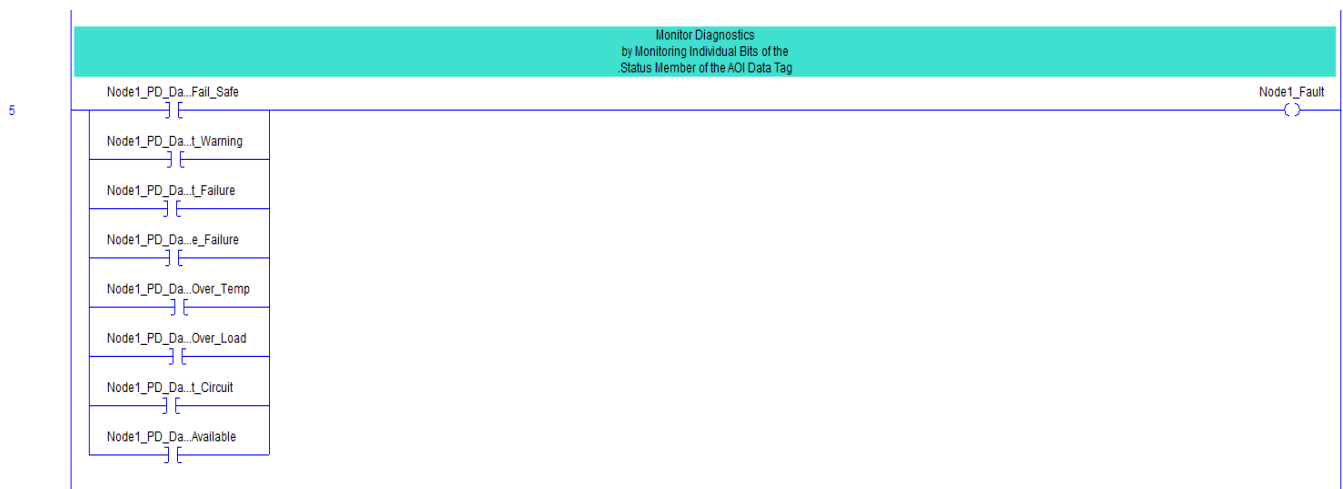


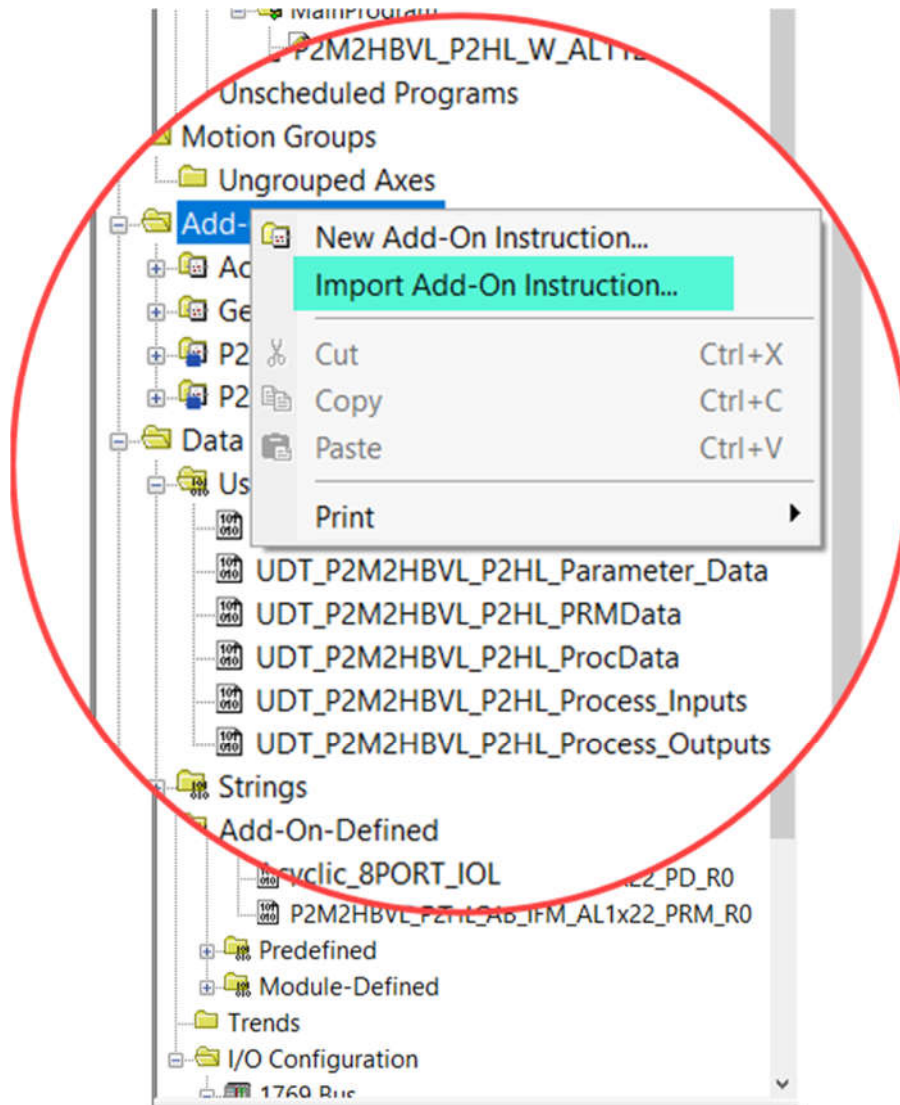
FIGURE 10 MONITORING STATUS BITS (CYCLIC)



Parameter Data Add-On Instruction Set Up

The “P2M2HBVL_P2HL_AB_IFM_AL1x22_PRM_Rx” AOI simplifies the usage of Parker P2M and P2H IO-Link devices with Allen-Bradley CompactLogix, ControlLogix and GuardLogix PLCs when connected, via Ethernet/IP, to an IFM AL1122 or 1222 IO-Link Master. The AOI facilitates the reading and writing of parameter data between the PLC and the Parker P2M or P2H IO-Link device.

FIGURE 11 HOW TO IMPORT AOI

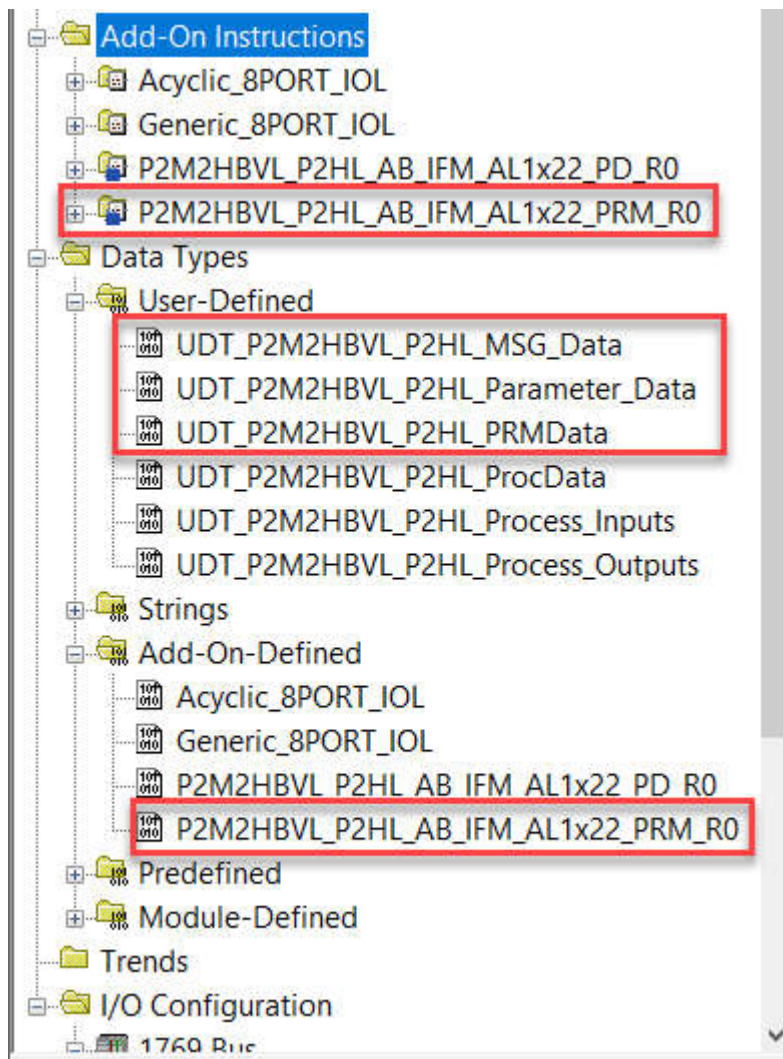


1. Right click Add-On Instruction in Controller Organizer and select “Import Add-On Instruction...”
2. Select the “P2M2HBVL_P2HL_AB_IFM_AL1x22_PRM_Rx” where _Rx is the revision of AOI.

P2M2HBVL_P2HL_AB_IFM_AL1x22_PRM_R0.L5X 7/3/2019 4:12 PM RSLogix 5000 XML ...

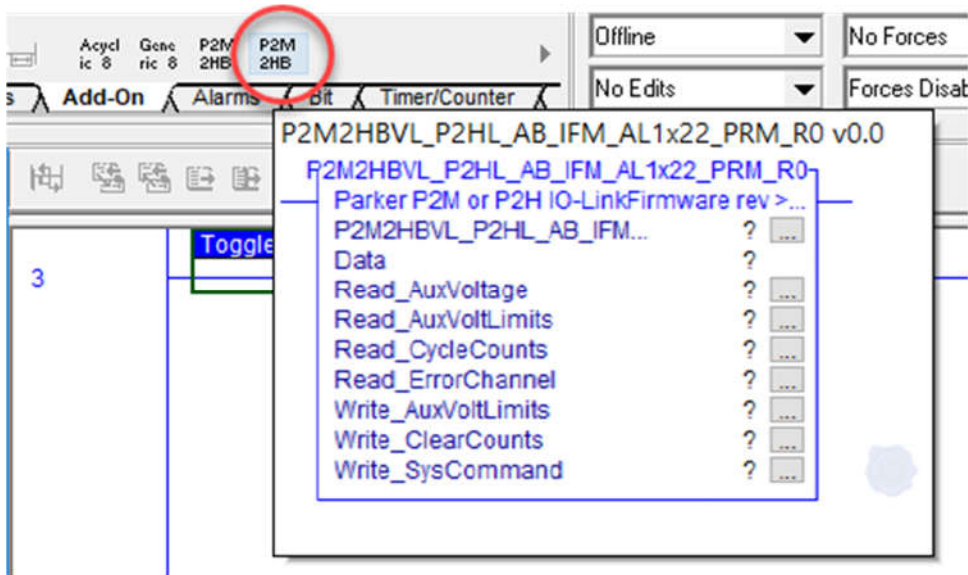
3. Choose OK on Import Configuration Window and you should then see the new AOI instance along with User-Defined and Add-On Defined data types created in the controller organizer.

FIGURE 12 LIST OF PROCESS DATA AOI'S AND DATA TYPES AFTER IMPORT



1. Add instance of instruction to an empty rung of ladder by clicking on the P2M under the Add-On tab in the top toolbar. The instruction will drop onto the selected rung.

FIGURE 13 SELECT AOI TO ADD TO RUNG



2. Assign an instance name for the AOI and create other tags necessary for operation. To do this, right click on the question marks and select "New Tag". Note that the name must be unique for each tag and each instance of the P2M / P2H AOI. The scope and data type fields will auto-populate with the correct values, so these should not need to be changed. Click the ellipsis button next to each message tag to provide configuration information. Configuration of all tags is required. See Appendix for structure breakdown of the "Data" variable.

FIGURE 14 CREATING AOI INSTANCE TAG

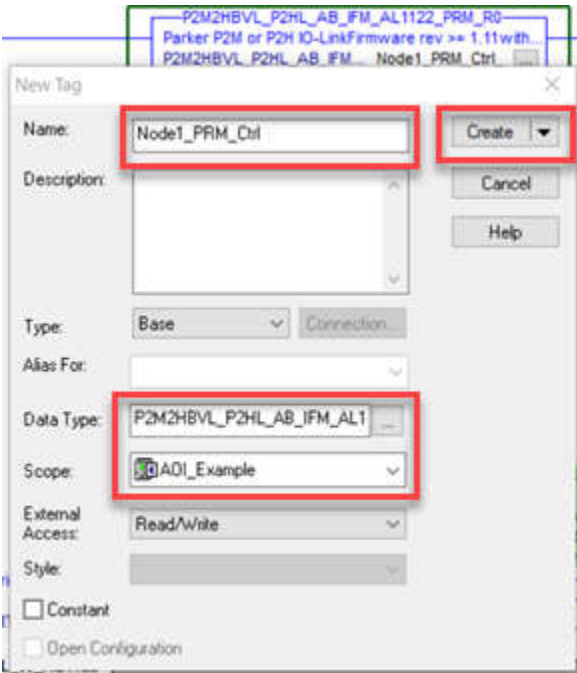
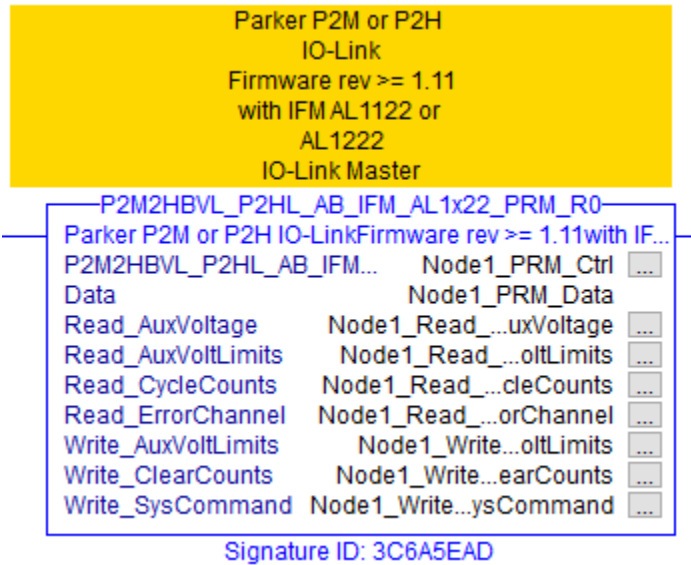


FIGURE 15 COMPLETED FUNCTION BLOCK EXAMPLE



Note: for the following configuration windows, the Service Code = 4b for read operations and 4c for write operations, the Attribute field represents the port (1-8) on the IFM IO-Link Master to which the P2M / P2H is connected. Instance field shall be 1 IO-Link Master Communication. All other fields can be entered directly as shown below.

FIGURE 16 READ_AUXVOLTAGE MESSAGE CONFIGURATION

Message Configuration - Node1_Read_AuxVoltage

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Service Code: 4b (Hex) Class: 80 (Hex) Instance: 1 Attribute: 1 (Hex)

Source Element: Node1_PRM_Data.M. Source Length: 3 (Bytes) Destination Element: Node1_PRM_Data.M.

Done Done Length: 2

OK Cancel Apply Help

FIGURE 17 READ_AUXVOLTLIMITS MESSAGE CONFIGURATION

Message Configuration - Node1_Read_AuxVoltLimits

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Service Code: 4b (Hex) Class: 80 (Hex) Instance: 1 Attribute: 1 (Hex)

Source Element: Node1_PRM_Data.M. Source Length: 3 (Bytes) Destination Element: Node1_PRM_Data.M.

Done Done Length: 4

OK Cancel Apply Help

FIGURE 18 READ_CYCLECOUNTS MESSAGE CONFIGURATION

Message Configuration - Node1_Read_CycleCounts

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Source Element: Node1_PRM_Data.M.

Source Length: 3 (Bytes)

Destination Element: Node1_PRM_Data.M.

New Tag...

Service Code: 4b (Hex) Class: 80 (Hex)

Instance: 1 Attribute: 1 (Hex)

Enable Enable Waiting Start Done Done Length: 96

Error Code: Extended Error Code: Timed Out

Error Path: Error Text:

OK Cancel Apply Help

FIGURE 19 READ_ERRORCHANNEL MESSAGE CONFIGURATION

Message Configuration - Node1_Read_ErrorChannel

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Source Element: Node1_PRM_Data.M.

Source Length: 3 (Bytes)

Destination Element: Node1_PRM_Data.M.

New Tag...

Service Code: 4b (Hex) Class: 80 (Hex)

Instance: 1 Attribute: 1 (Hex)

Enable Enable Waiting Start Done Done Length: 4

Error Code: Extended Error Code: Timed Out

Error Path: Error Text:

OK Cancel Apply Help

FIGURE 20 WRITE_AUXVOLTLIMITS MESSAGE CONFIGURATION

Message Configuration - Node1_Write_AuxVoltLimits

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Service Code: 4c (Hex) Class: 80 (Hex) Instance: 1 Attribute: 1 (Hex)

Source Element: Node1_PRM_Data.M.

Source Length: 7 (Bytes)

Destination Element: Node1_PRM_Data.M.

New Tag...

☐ Enable ☐ Enable Waiting ☐ Start ☒ Done Done Length: 0

☐ Error Code: Extended Error Code: ☐ Timed Out

Error Path:
Error Text:

OK Cancel Apply Help

FIGURE 21 WRITE_CLEARCOUNTS MESSAGE CONFIGURATION

Message Configuration - Node1_Write_ClearCounts

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Service Code: 4c (Hex) Class: 80 (Hex) Instance: 1 Attribute: 1 (Hex)

Source Element: Node1_PRM_Data.M.

Source Length: 7 (Bytes)

Destination Element: Node1_PRM_Data.M.

New Tag...

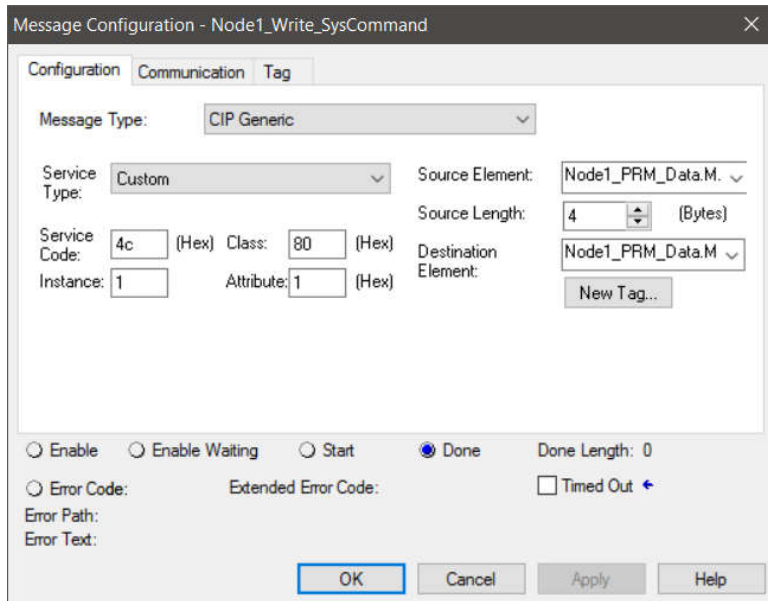
☐ Enable ☐ Enable Waiting ☐ Start ☒ Done Done Length: 0

☐ Error Code: Extended Error Code: ☐ Timed Out

Error Path:
Error Text:

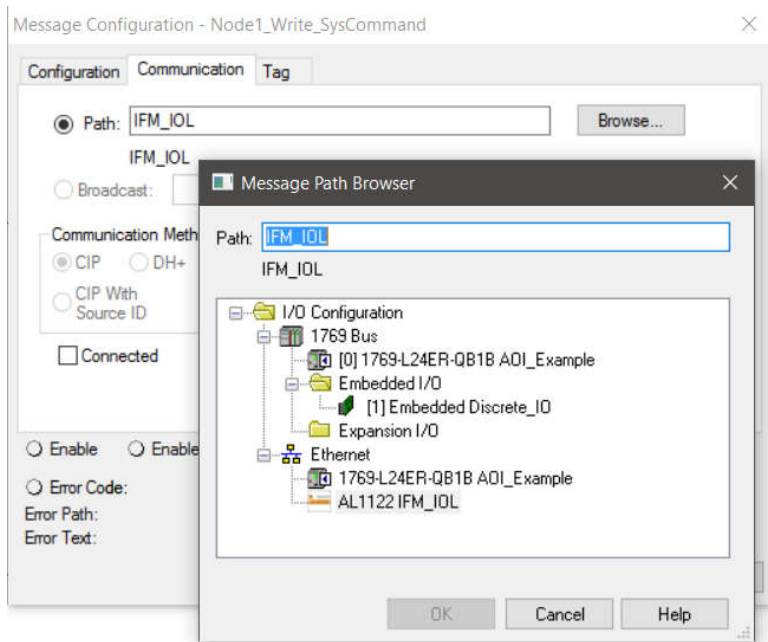
OK Cancel Apply Help

FIGURE 22 WRITE_SYSCommand MESSAGE CONFIGURATION



For each message tag, the device path must also be selected within the “Communication” tab. Select the “Browse” button and select the appropriate IFM IO-Link Master.

FIGURE 23 CONFIGURE PATH FOR MESSAGE BLOCK



USING THE INSTRUCTION

It is important to note the difference between cyclic and acyclic data. Process Data (cyclic) is updated without a request; whereas Parameter Data (acyclic) requires the program to toggle a bit to read or write data contained inside the slave device. See appendix for all data points available. See ladder logic examples below:

Read Parameter Data (Acyclic)

*****When initiating a read OR a write request you must reset the “Node1_PRM_Data.M.AOI_Handshake” bit AFTER processing the data*****

It is also important to not initiate multiple read or write requests at the same time. Write your logic such that only one of the request bits is turned on at a time and wait for the Handshake bit to go high before executing the next request.

FIGURE 24 HANDSHAKE RESET

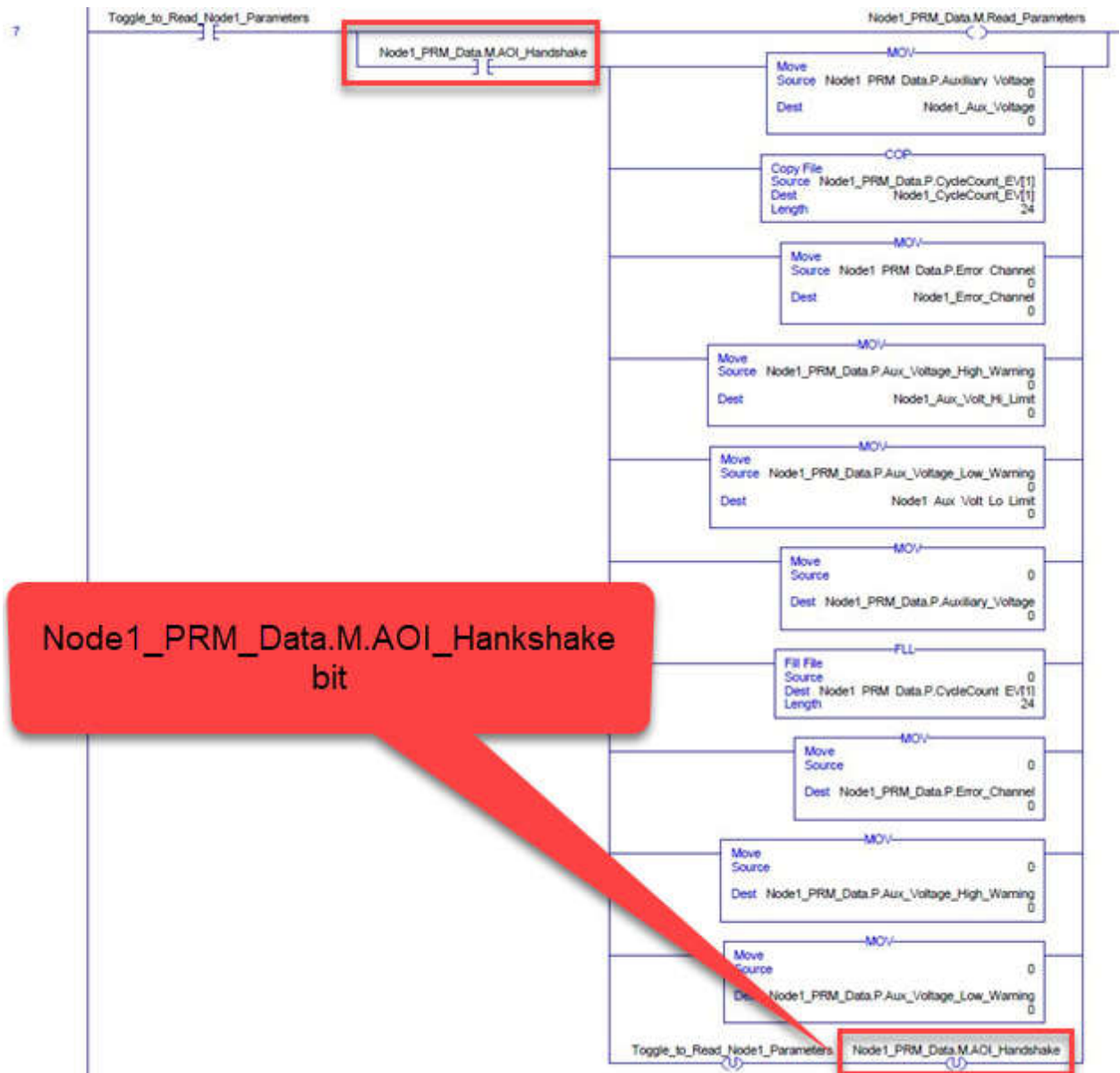
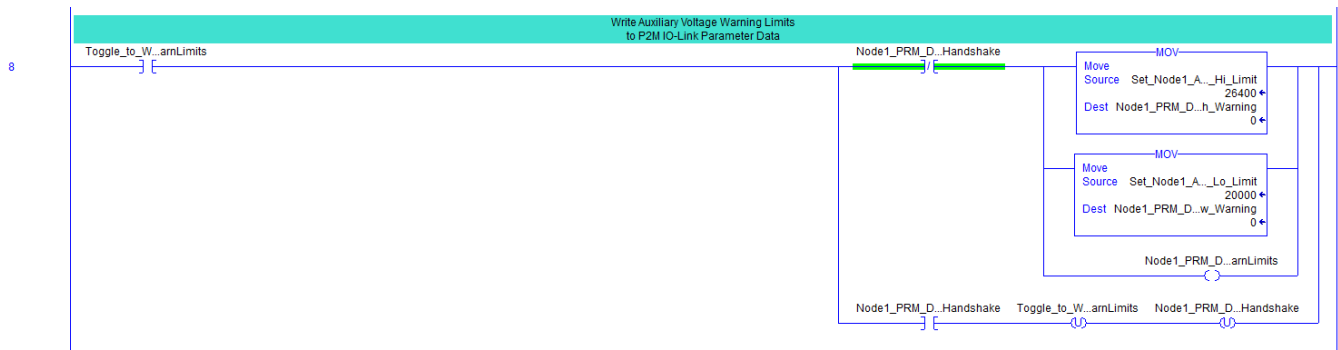
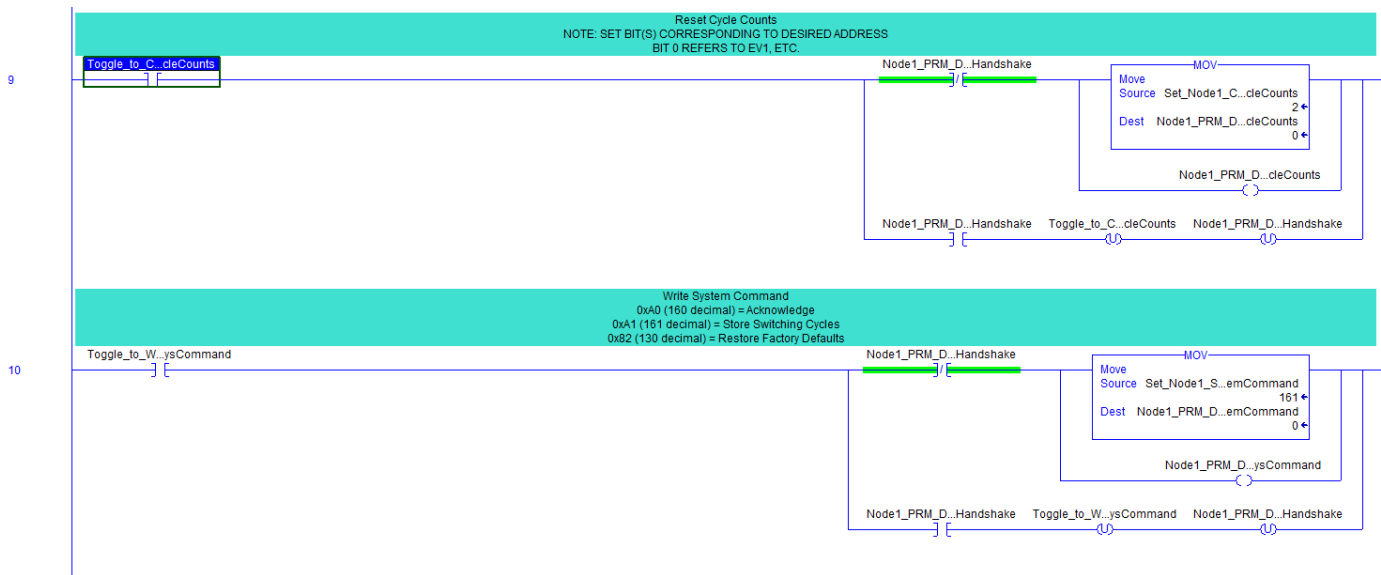


FIGURE 25 WRITE PARAMETER DATA (ACYCLIC)



Note: When sending new limit setpoints to the P2M module, the values will not be written unless the high limit is more than one volt greater than the low limit, and greater than zero.

FIGURE 26 RESET CYCLE COUNTS AND SYSTEM COMMAND EXAMPLE



APPENDIX

Process Data Structures

User Defined / Add-On Defined Data Structures utilized by AOI
“P2M2HBVL_P2HL_AB_IFM_AL1x22_PD_Rx”

FIGURE 27 PROCESS DATA STRUCTURES

Name:UDT_P2M2HBVL_P2HL_ProcData

Description:

Members:Data Type Size: 16 byte(s)

	Name	Data Type	Style	Description	External Access
<input checked="" type="checkbox"/>	Status	UDT_P2M2HBVL_P2HL_Process_Inputs			Read/Write
<input checked="" type="checkbox"/>	EV	UDT_P2M2HBVL_P2HL_Process_Outputs			Read/Write

FIGURE 28 PROCESS DATA STRUCTURE INPUTS

Name:

Description:

Members: Data Type Size: 16 byte(s)

Name	Data Type	Style	Description	External Access
<input checked="" type="checkbox"/> Status	UDT_P2M2HBVL_P2HL_Process_Inputs			Read/Write
<input type="checkbox"/> Fail_Safe	BOOL	Decimal		Read/Write
<input type="checkbox"/> Aux_Volt_Warning	BOOL	Decimal		Read/Write
<input type="checkbox"/> Aux_Volt_Failure	BOOL	Decimal		Read/Write
<input type="checkbox"/> Module_Failure	BOOL	Decimal		Read/Write
<input type="checkbox"/> Module_Over_Temp	BOOL	Decimal		Read/Write
<input type="checkbox"/> Module_Over_Load	BOOL	Decimal		Read/Write
<input type="checkbox"/> Short_Circuit	BOOL	Decimal		Read/Write
<input type="checkbox"/> Output_Stage_Not_Available	BOOL	Decimal		Read/Write
<input type="checkbox"/> Device_OK	BOOL	Decimal		Read/Write
<input type="checkbox"/> Mismatch_Fault	BOOL	Decimal		Read/Write
<input type="checkbox"/> Comm_Fault	BOOL	Decimal		Read/Write
<input type="checkbox"/> Validation_Failed	BOOL	Decimal		Read/Write
<input type="checkbox"/> Event_1_Error_Code	SINT	Decimal		Read/Write
<input type="checkbox"/> Event_1_Add_Code_1	SINT	Decimal		Read/Write
<input type="checkbox"/> Event_1_Add_Code_2	SINT	Decimal		Read/Write
<input type="checkbox"/> Event_2_Error_Code	SINT	Decimal		Read/Write
<input type="checkbox"/> Event_2_Add_Code_1	SINT	Decimal		Read/Write
<input type="checkbox"/> Event_2_Add_Code_2	SINT	Decimal		Read/Write
<input type="checkbox"/> Event_3_Error_Code	SINT	Decimal		Read/Write
<input type="checkbox"/> Event_3_Add_Code_1	SINT	Decimal		Read/Write
<input type="checkbox"/> Event_3_Add_Code_2	SINT	Decimal		Read/Write

FIGURE 29 PROCESS DATA STRUCTURE OUTPUTS

Name:

Description:

Members: Data Type Size: 16 byte(s)

Name	Data Type	Style	Description	External Access
EV	UDT_P2M2HBVL_P2HL_Process_Outputs			Read/Write
__01	BOOL	Decimal	EV1 Command	Read/Write
__02	BOOL	Decimal	EV2 Command	Read/Write
__03	BOOL	Decimal	EV3 Command	Read/Write
__04	BOOL	Decimal	EV4 Command	Read/Write
__05	BOOL	Decimal	EV5 Command	Read/Write
__06	BOOL	Decimal	EV6 Command	Read/Write
__07	BOOL	Decimal	EV7 Command	Read/Write
__08	BOOL	Decimal	EV8 Command	Read/Write
__09	BOOL	Decimal	EV9 Command	Read/Write
__10	BOOL	Decimal	EV10 Command	Read/Write
__11	BOOL	Decimal	EV11 Command	Read/Write
__12	BOOL	Decimal	EV12 Command	Read/Write
__13	BOOL	Decimal	EV13 Command	Read/Write
__14	BOOL	Decimal	EV14 Command	Read/Write
__15	BOOL	Decimal	EV15 Command	Read/Write
__16	BOOL	Decimal	EV16 Command	Read/Write
__17	BOOL	Decimal	EV17 Command	Read/Write
__18	BOOL	Decimal	EV18 Command	Read/Write
__19	BOOL	Decimal	EV19 Command	Read/Write
__20	BOOL	Decimal	EV20 Command	Read/Write
__21	BOOL	Decimal	EV21 Command	Read/Write
__22	BOOL	Decimal	EV22 Command	Read/Write
__23	BOOL	Decimal	EV23 Command	Read/Write
__24	BOOL	Decimal	EV24 Command	Read/Write

Parameter Data Structures

User Defined / Add-On Defined Data Structures utilized by AOI

“P2M2HBVL_P2HL_AB_IFM_AL1x22_PRM_Rx”

FIGURE 30 PARAMETER DATA STRUCTURES

Name:

Description:

Members: Data Type Size: 328 byte(s)

	Name	Data Type	Style	Description	External Access
<input checked="" type="checkbox"/>	P	UDT_P2M2HBVL_P2HL_Parameter_Data			Read/Write
<input checked="" type="checkbox"/>	M	UDT_P2M2HBVL_P2HL_MSG_Data			Read/Write

FIGURE 31 PARAMETER DATA STRUCTURE

Name:

Description:

Members: Data Type Size: 328 byte(s)

	Name	Data Type	Style	Description	External Access
<input checked="" type="checkbox"/>	P	UDT_P2M2HBVL_P2HL_Parameter_Data			Read/Write
<input type="checkbox"/>	Auxiliary_Voltage	INT	Decimal		Read/Write
<input type="checkbox"/>	Error_Channel	DINT	Decimal		Read/Write
<input type="checkbox"/>	Aux_Voltage_High_Warning	INT	Decimal		Read/Write
<input type="checkbox"/>	Aux_Voltage_Low_Warning	INT	Decimal		Read/Write
<input type="checkbox"/>	CycleCount_EV	DINT[25]	Decimal		Read/Write
<input type="checkbox"/>	ClearCycleCounts	DINT	Decimal		Read/Write
<input type="checkbox"/>	SystemCommand	INT	Decimal		Read/Write

FIGURE 32 PARAMETER DATA MESSAGE DATA

Name:

Description:

Members: Data Type Size: 328 byte(s)

Name	Data Type	Style	Description	External Access
P	UDT_P2M2HBVL_P2HL_Parameter_Data			Read/Write
M	UDT_P2M2HBVL_P2HL_MSG_Data			Read/Write
Read_Parameters	BOOL	Decimal		None
Write_AuxVoltageWarnLimits	BOOL	Decimal		None
Write_ClearCycleCounts	BOOL	Decimal		None
Write_SysCommand	BOOL	Decimal		None
AOI_Handshake	BOOL	Decimal		None
Write	BOOL	Decimal		None
Reset	BOOL	Decimal		None
Done	BOOL	Decimal		None
Error	BOOL	Decimal		None
status	INT	Decimal		None
ReceiveData	SINT[97]	Decimal		None
SendData	SINT[97]	Decimal		None
WriteData	DINT	Decimal		None