



P2M2HBVL MODUFLEX IO-LINK
ADD-ON INSTRUCTION FOR
ROCKWELL PLC WITH BALLUFF
BNI006A EIP IO-LINK MASTER
QUICK START GUIDE

PREFACE

This Quick Start Guide (QSG) is designed to help integrate Parker Hannifin's P2M IO-Link valve manifold into an Allen-Bradley (AB) PLC environment utilizing the Balluff BNI006A EIP IO-Link Master module. The QSG assumes that you are already using the Balluff AOI 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:

http://usa.balluff.com/AOI/distribution/AOI/AOI_BNI006A_50_31_040.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 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 IO-Link device.

The "AB_P2M2HBVL_ioLV1_10_w_BNI006A_PRM_Rx" AOI facilitates the call-up of the acyclic service data.

The "AB_P2M2HBVL_ioLV1_10_w_BNI006A_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 P2M manual here:

You can download this QSG as well as an example RSLogix5000 file here:

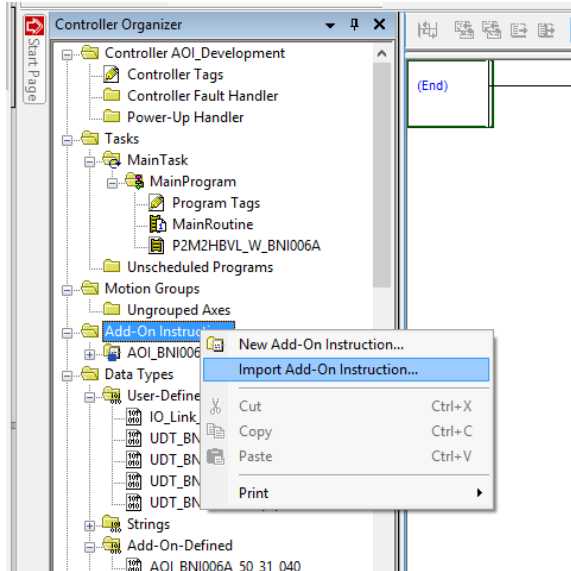
www.parker.com/PDN/io-link

Process Data Add-On Instruction

The “AB_P2M2HBVL_IOLv1_10_w_BNI006A_PD_Rx” AOI simplifies the usage of Parker P2M IO-Link devices with Allen-Bradley CompactLogix, ControlLogix and GuardLogix PLCs when connected, via Ethernet/IP, to a Balluff BNI006A IO-Link Master. Data is mapped to user-friendly control and diagnostic tags on the PLC side.

IMPORTING THE INSTRUCTION

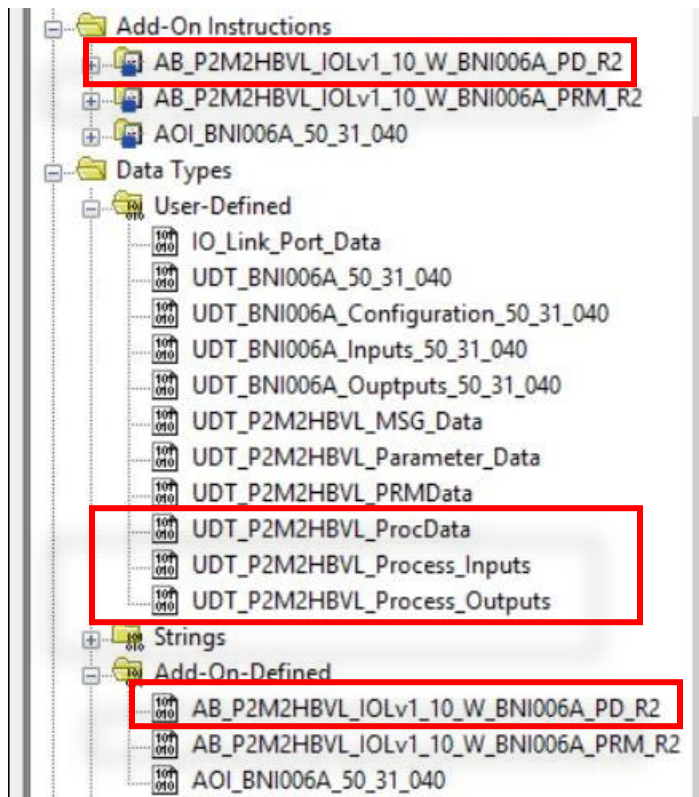
1. Right click Add-On Instruction in Controller Organizer and select “Import Add-On Instruction...”



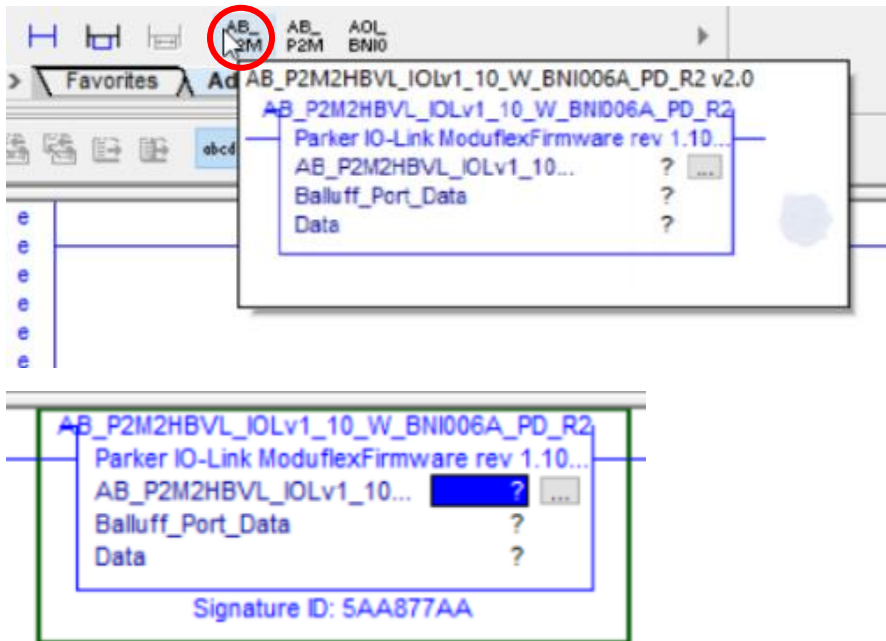
2. Select the “AB_P2M2HBVL_IOLv1_x_w_BNI006A_PD_Rx” where IOLv1_x_ is the firmware version of the P2M module and _Rx is the revision of AOI.

| | | | |
|--|-------------------|---------------------|-------|
| AB_P2M2HBVL_IOLv1_10_W_BNI006A_PD_R2.L5X | 5/25/2018 4:30 PM | Logix Designer X... | 57 KB |
|--|-------------------|---------------------|-------|

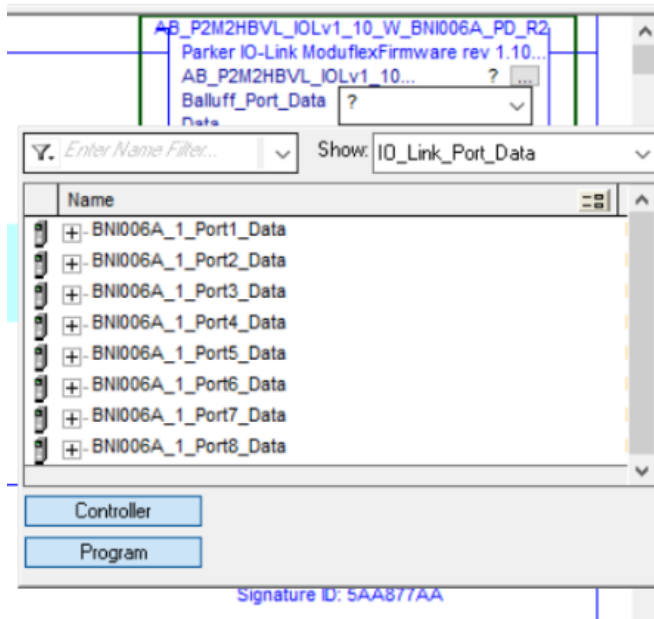
Choose OK on Import Configuration Window and you should then see the new AOI instance along with User-Defined and Add-On Defined data types created in the controller organizer.



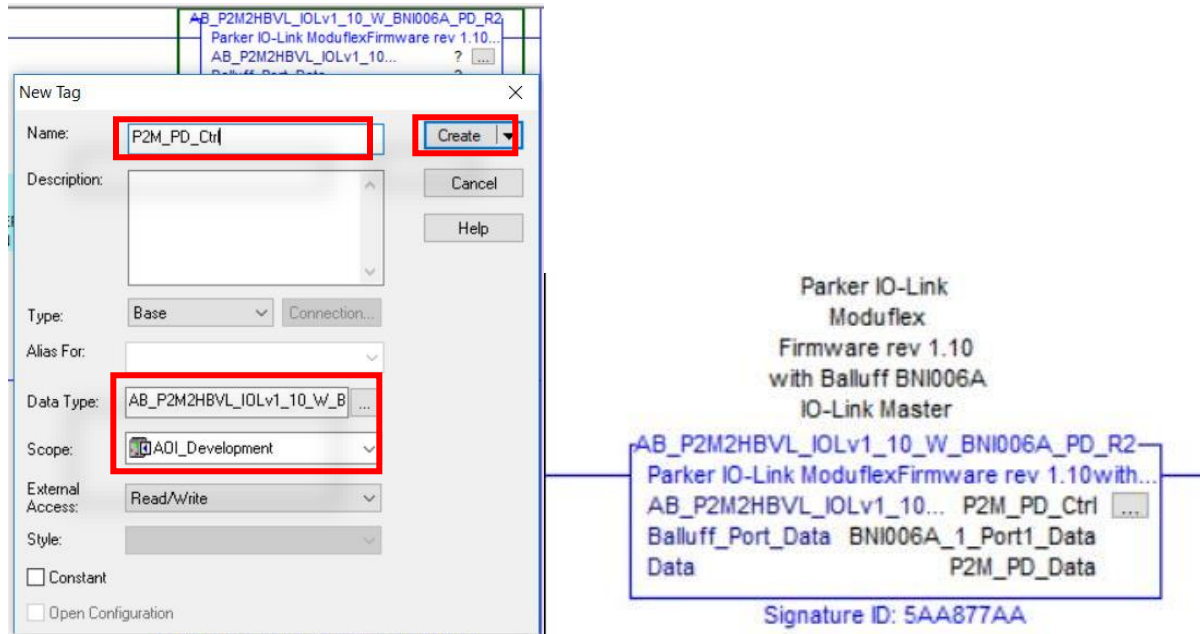
3. Add instance of instruction to an empty rung of ladder by clicking on the AB_P2M under the Add-On tab in the top toolbar. The instruction will drop onto the selected rung.



4. Point the Balluff_Port_Data field to the port that the P2M is connected to on the BNI006A. If you choose the incorrect port 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.



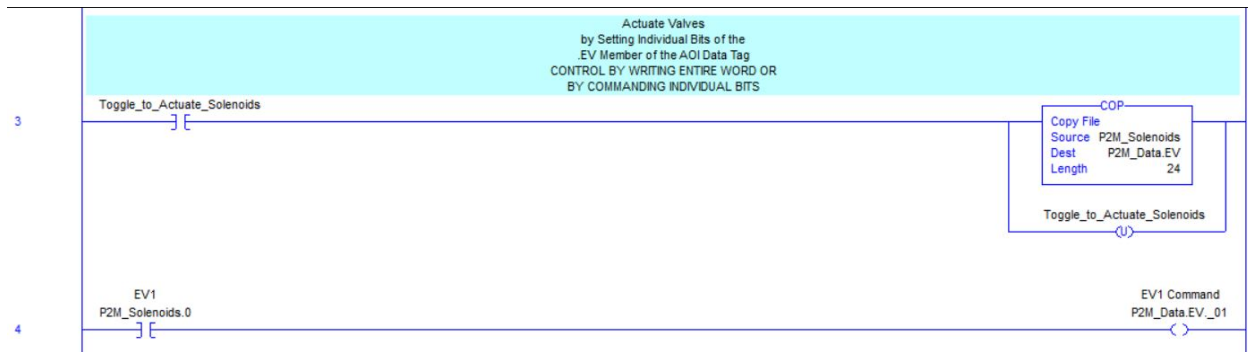
- Assign an instance name for the AOI and create other tags necessary for operation. Right click on the question marks and select “New Tag”. Note that the name must be unique for each tag and each instance of the P2M AOI. The scope and data type fields will auto-populate with the correct values, so these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



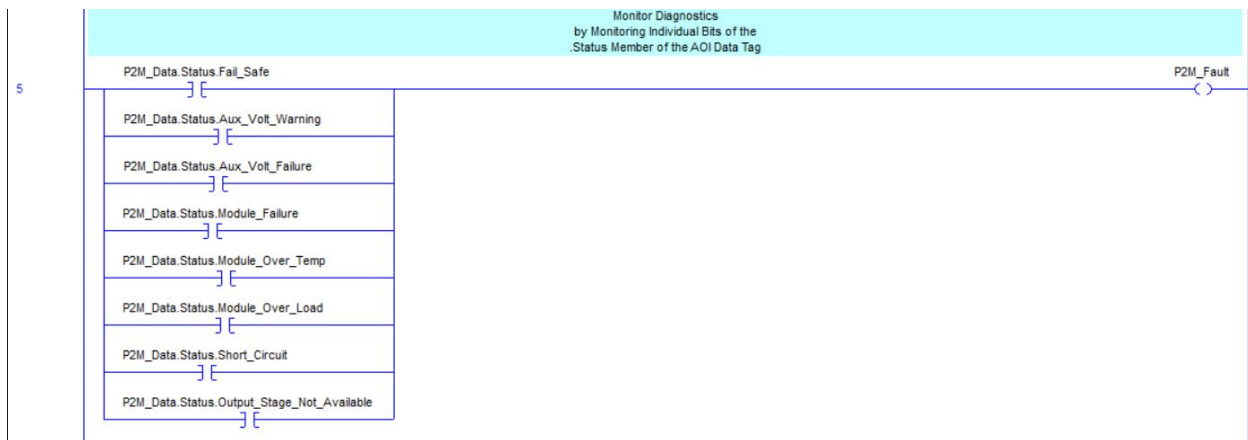
USING THE INSTRUCTION

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 data. Cyclic data includes input status and valve output control. This means that P2M_Data.Status.xxx and P2M_Data.EV.##_ are live tags with real data just by calling the AOI. See appendix for all data points available. See ladder logic examples below:

a. Toggling Solenoid Valves (Cyclic)



b. Monitoring Status Bits (Cyclic)

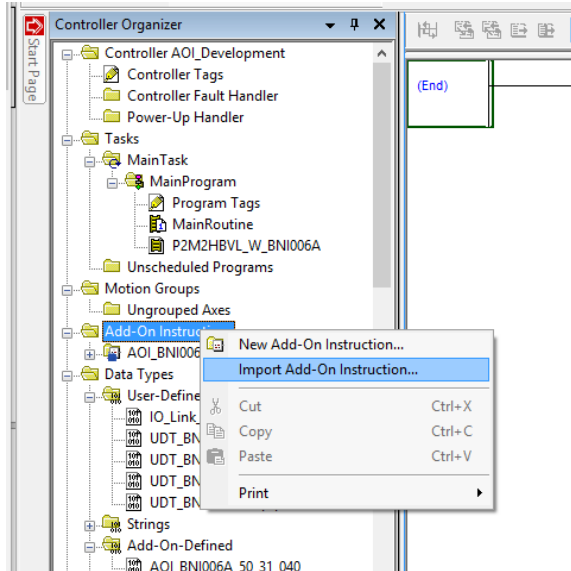


Parameter Data Add-On Instruction

The “AB_P2M2HBVL_IOLv1_10_w_BNI006A_PRM_Rx” AOI simplifies the usage of Parker P2M IO-Link devices with Allen-Bradley CompactLogix, ControlLogix and GuardLogix PLCs when connected, via Ethernet/IP, to a Balluff BNI006A IO-Link Master. The AOI facilitates the reading and writing of parameter data between the PLC and the Parker P2M IO-Link device.

IMPORTING THE INSTRUCTION

1. Right click Add-On Instruction in Controller Organizer and select “Import Add-On Instruction...”

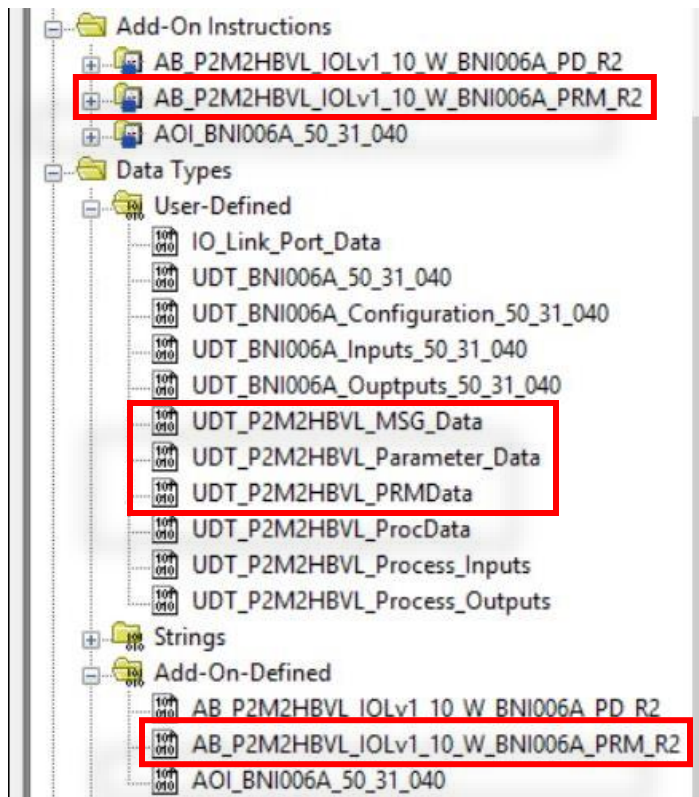


2. Select the “AB_P2M2HBVL_IOLv1_x_w_BNI006A_PRM_Rx” where IOLv1_x_ is the firmware version of the P2M module and _Rx is the revision of AOI.

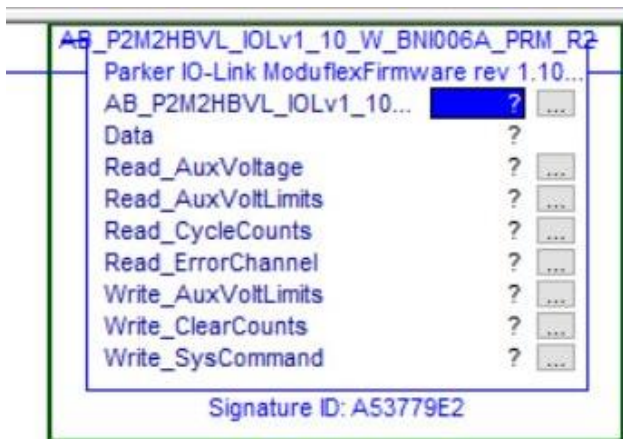
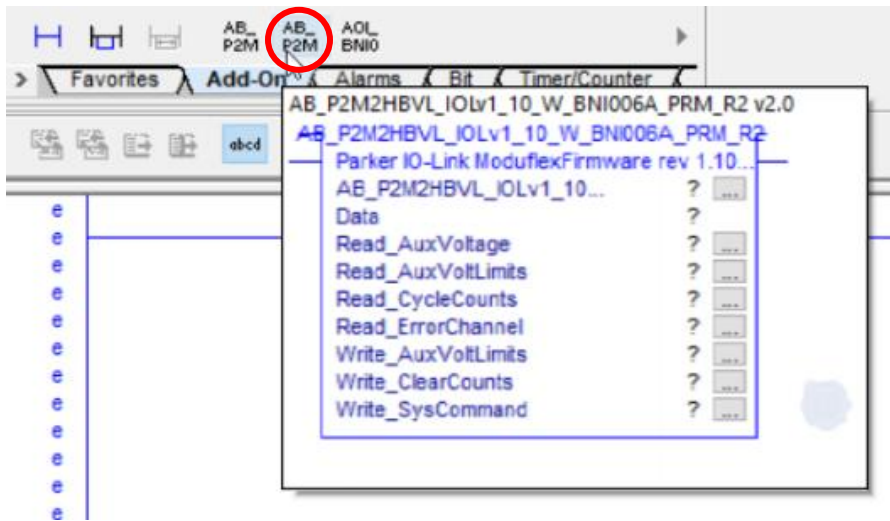
AB_P2M2HBVL_IOLv1_10_W_BNI006A_PRM_R2.L5X

5/25/2018 4:30 PM Logix Designer X...

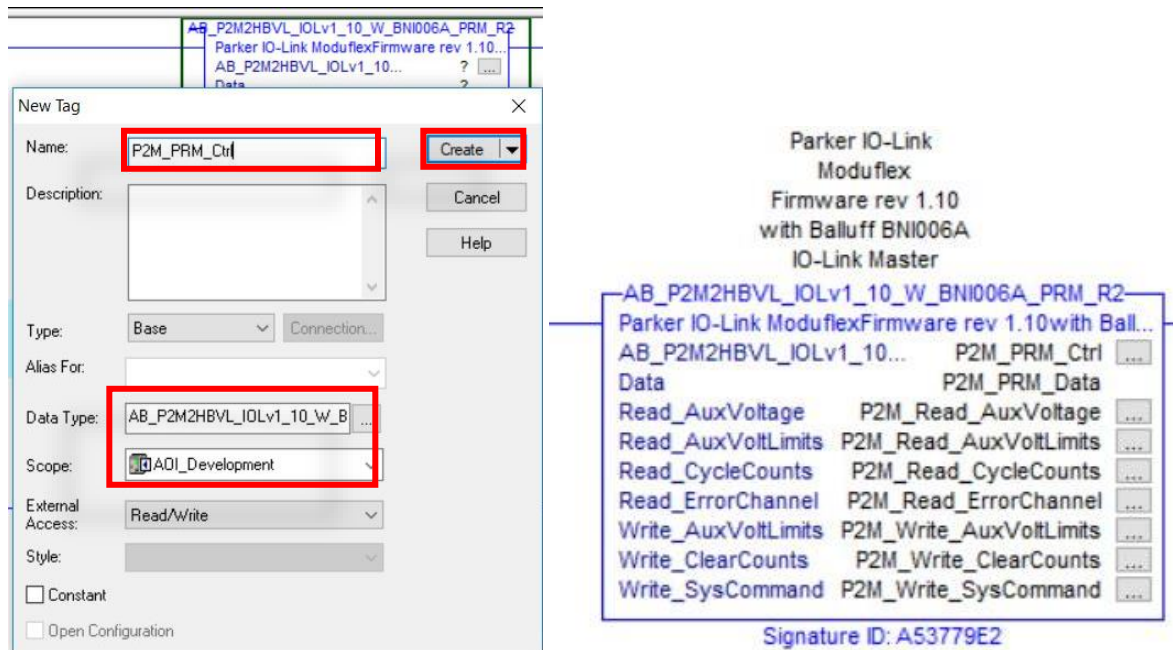
Choose OK on Import Configuration Window and you should then see the new AOI instance along with User-Defined and Add-On Defined data types created in the controller organizer.



3. Add instance of instruction to an empty rung of ladder by clicking on the AB_P2M under the Add-On tab in the top toolbar. The instruction will drop onto the selected rung.

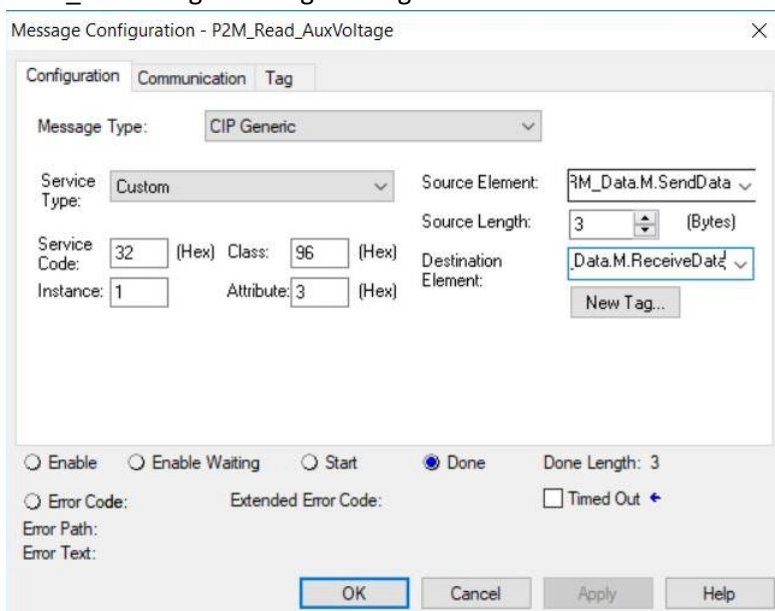


- Assign an instance name for the AOI and create other tags necessary for operation. To do this, right click on the question marks and select “New Tag”. Note that the name must be unique for each tag and each instance of the P2M AOI. The scope and data type fields will auto-populate with the correct 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.



Note: for the following configuration windows, the Instance field represents the port (1-8) on the Balluff IO-Link Master to which the P2M is connected. Attribute field shall be 3 for read operations and 2 for write operations. All other fields can be entered directly as shown below.

Read_AuxVoltage Message Configuration:



Read_AuxVoltLimits Message Configuration:

Message Configuration - P2M_Read_AuxVoltLimits

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Service Code: 32 (Hex) Class: 96 (Hex) Instance: 1 Attribute: 3 (Hex)

Source Element: RM_Data.M.SendData

Source Length: 3 (Bytes)

Destination Element: Data.M.ReceiveData

New Tag...

☐ Enable ☐ Enable Waiting ☐ Start ☒ Done Done Length: 5

☐ Error Code: Extended Error Code: ☐ Timed Out

Error Path:
Error Text:

OK Cancel Apply Help

Read_CycleCounts Message Configuration:

Message Configuration - P2M_Read_CycleCounts

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Service Code: 32 (Hex) Class: 96 (Hex) Instance: 1 Attribute: 3 (Hex)

Source Element: RM_Data.M.SendData

Source Length: 3 (Bytes)

Destination Element: Data.M.ReceiveData

New Tag...

☐ Enable ☐ Enable Waiting ☐ Start ☒ Done Done Length: 97

☐ Error Code: Extended Error Code: ☐ Timed Out

Error Path:
Error Text:

OK Cancel Apply Help

Read_ErrorChannel Message Configuration:

Message Configuration - P2M_Read_ErrorChannel

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Service Code: 32 (Hex) Class: 96 (Hex) Instance: 1 Attribute: 3 (Hex)

Source Element: P2M_Data.M.SendData

Source Length: 3 (Bytes)

Destination Element: Data.M.ReceiveData

New Tag...

☐ Enable
 ☐ Enable Waiting
 ☐ Start
 ☒ Done
 Done Length: 5

☐ Error Code:
 Extended Error Code:
 ☐ Timed Out

Error Path:

Error Text:

OK Cancel Apply Help

Write_AuxVoltLimits Message Configuration:

Message Configuration - P2M_Write_AuxVoltLimits

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Service Code: 32 (Hex) Class: 96 (Hex) Instance: 1 Attribute: 2 (Hex)

Source Element: P2M_Data.M.SendData[0]

Source Length: 7 (Bytes)

Destination Element: Data.M.SendData[7]

New Tag...

☐ Enable
 ☐ Enable Waiting
 ☐ Start
 ☒ Done
 Done Length: 1

☐ Error Code:
 Extended Error Code:
 ☐ Timed Out

Error Path:

Error Text:

OK Cancel Apply Help

Write_ClearCounts Message Configuration:

Message Configuration - P2M_Write_ClearCounts

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Service Code: 32 (Hex) Class: 96 (Hex) Instance: 1 Attribute: 2 (Hex)

Source Element: I_Data.M.SendData[0]

Source Length: 7 (Bytes)

Destination Element: I_Data.M.SendData[7]

New Tag...

☐ Enable
 ☐ Enable Waiting
 ☐ Start
 ☒ Done
 Done Length: 1

☐ Error Code:
 Extended Error Code:
 ☐ Timed Out

Error Path:

Error Text:

OK Cancel Apply Help

Write_SysCommand Message Configuration:

Message Configuration - P2M_Write_SysCommand

Configuration Communication Tag

Message Type: CIP Generic

Service Type: Custom

Service Code: 32 (Hex) Class: 96 (Hex) Instance: 1 Attribute: 2 (Hex)

Source Element: I_Data.M.SendData[0]

Source Length: 4 (Bytes)

Destination Element: I_Data.M.SendData[4]

New Tag...

☐ Enable
 ☐ Enable Waiting
 ☐ Start
 ☒ Done
 Done Length: 1

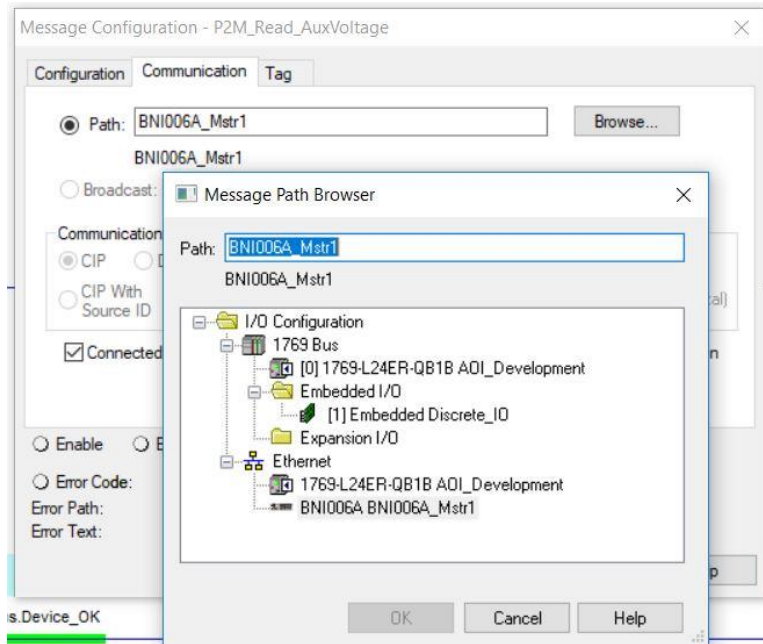
☐ Error Code:
 Extended Error Code:
 ☐ Timed Out

Error Path:

Error Text:

OK Cancel Apply Help

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



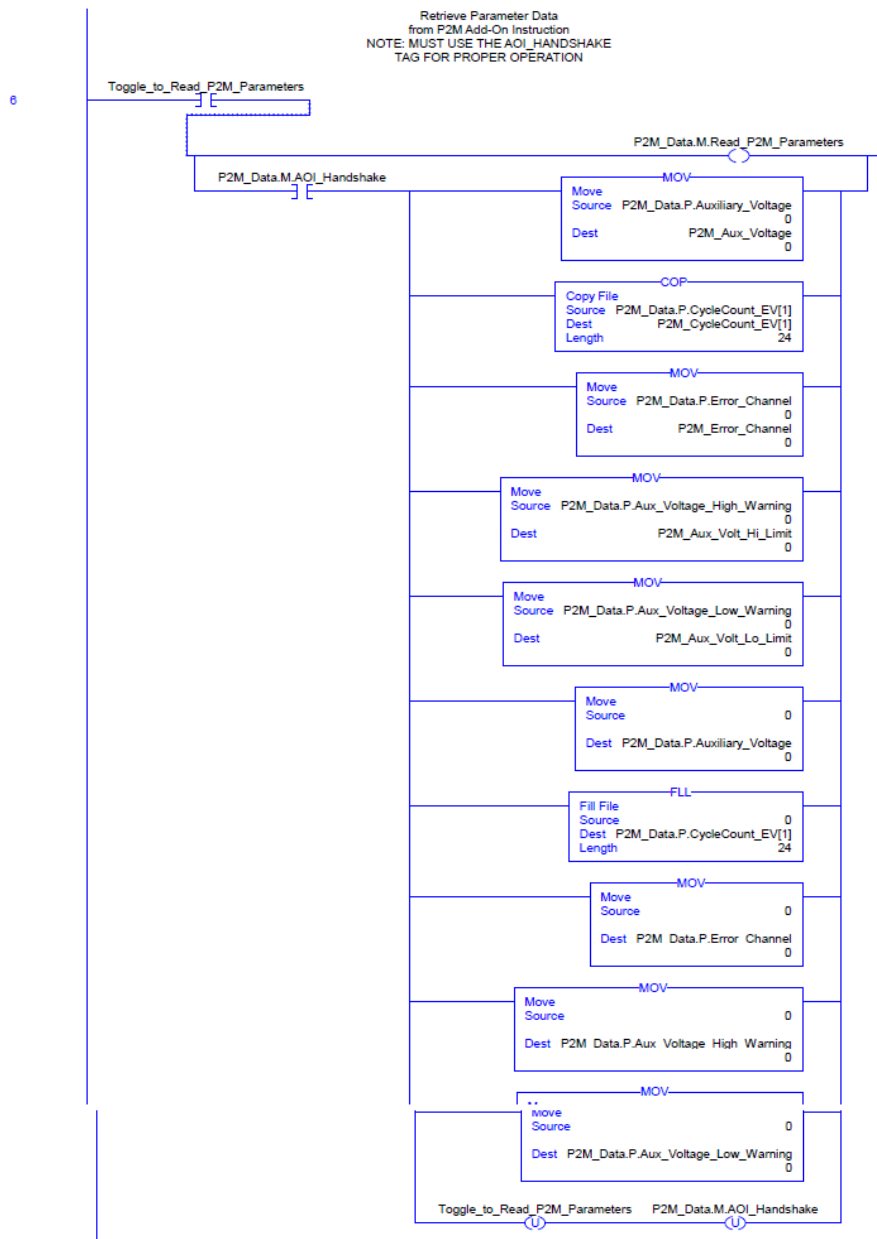
USING THE INSTRUCTION

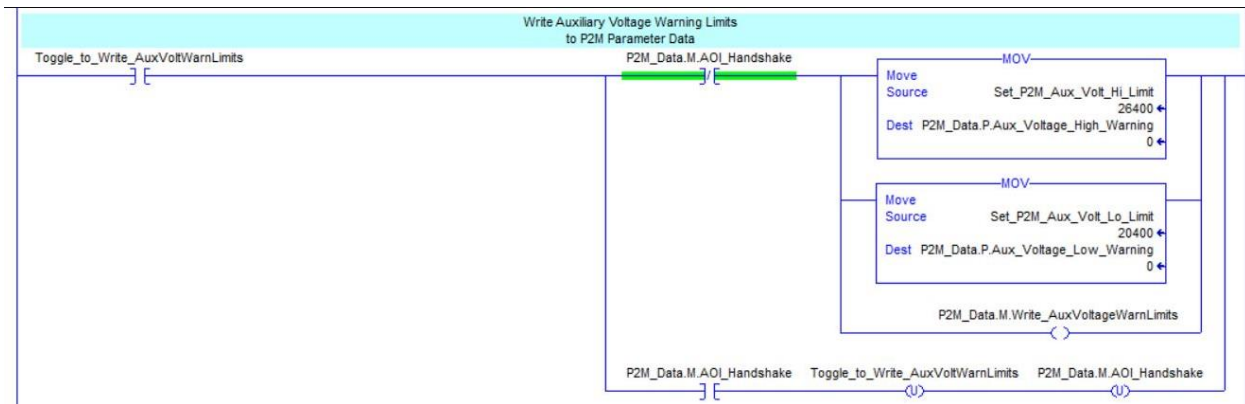
It is important to note the difference between cyclic and acyclic data. Process Data (cyclic) is updated without a request; whereas Parameter Data (acyclic) requires the program to toggle a bit to read or write data contained inside the slave device. See appendix for all data points available. See ladder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

a. Read Parameter Data (Acyclic)

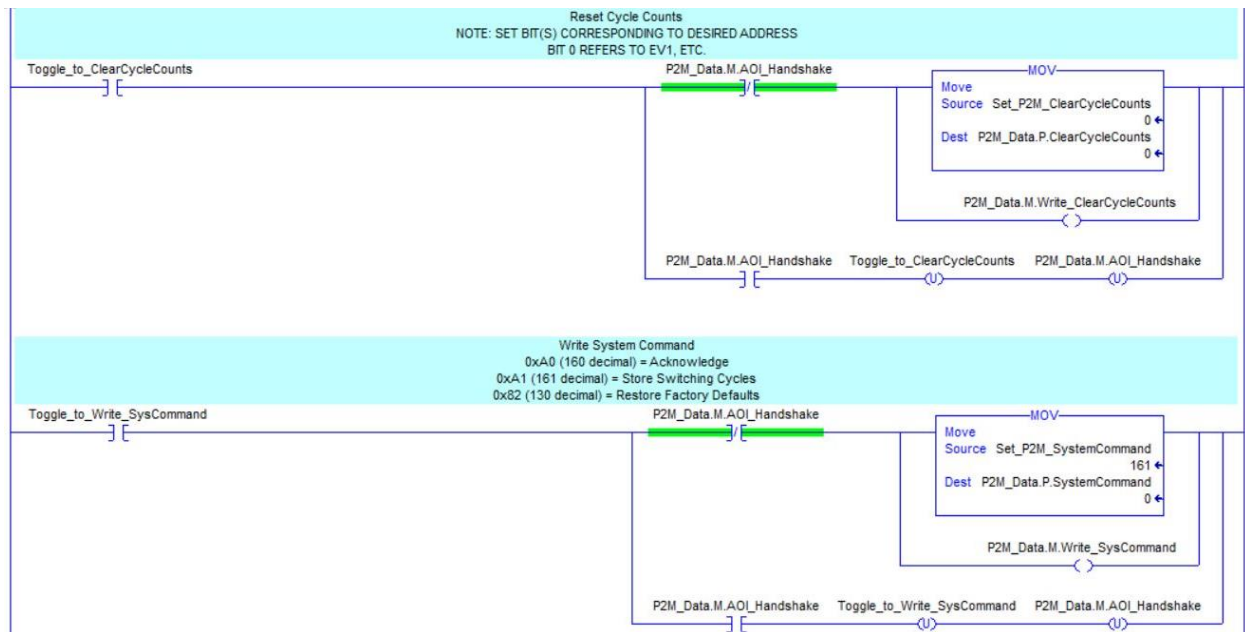
*****When initiating a read OR a write request you must reset the “P2M Data.M.AOI_Handshake bit” AFTER processing the data*****

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



b. 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.



APPENDIX

Process Data Structures

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

“AB_P2M2HBVL_IOLv1_10_w_BNI006A_PD_Rx”

Name:

Description:

Members: Data Type Size: 16 byte(s)

| | Name | Data Type | Style | Description | External Access |
|-------------------------------------|--------|------------------------------|-------|-------------|-----------------|
| <input checked="" type="checkbox"/> | Status | UDT_P2M2HBVL_Process_Inputs | | | Read/Write |
| <input checked="" type="checkbox"/> | EV | UDT_P2M2HBVL_Process_Outputs | | | Read/Write |

1 of 100

P2M2HBVL with BALLUFF BNI006A – AOI QSG

Name:

Description:

Members:
Data Type Size: 16 byte(s)

| Name | Data Type | Style | Description | External Access |
|---|------------------------------|---------|-------------|-----------------|
| <input type="checkbox"/> Status | UDT_P2M2HBVL_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 |
| <input checked="" type="checkbox"/> EV | UDT_P2M2HBVL_Process_Outputs | | | Read/Write |

Name:

Description:

Members: Data Type Size: 16 byte(s)

| Name | Data Type | Style | Description | External Access |
|----------|------------------------------|---------|--------------|-----------------|
| ⊕ Status | UDT_P2M2HBVL_Process_Inputs | | | Read/Write |
| ⊖ EV | UDT_P2M2HBVL_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

“AB_P2M2HBVL_IOLv1_10_w_BNI006A_PRM_Rx”

Name:

Description:

Members: Data Type Size: 328 byte(s)

| | Name | Data Type | Style | Description | External Access |
|--------------------------|------|-----------------------------|-------|-------------|-----------------|
| <input type="checkbox"/> | P | UDT_P2M2HBVL_Parameter_Data | | | Read/Write |
| <input type="checkbox"/> | M | UDT_P2M2HBVL_MSG_Data | | | Read/Write |

top of 010

Name:

Description:

Members: Data Type Size: 328 byte(s)

| | Name | Data Type | Style | Description | External Access |
|--------------------------|--------------------------|-----------------------------|---------|-------------|-----------------|
| <input type="checkbox"/> | P | UDT_P2M2HBVL_Parameter_Data | | | Read/Write |
| | Auxiliary_Voltage | INT | Decimal | | Read/Write |
| | Error_Channel | DINT | Decimal | | Read/Write |
| | Aux_Voltage_High_Warning | INT | Decimal | | Read/Write |
| | Aux_Voltage_Low_Warning | INT | Decimal | | Read/Write |
| | CycleCount_EV | DINT[25] | Decimal | | Read/Write |
| | ClearCycleCounts | DINT | Decimal | | Read/Write |
| | SystemCommand | INT | Decimal | | Read/Write |
| <input type="checkbox"/> | M | UDT_P2M2HBVL_MSG_Data | | | Read/Write |

top of 010

P2M2HBVL with BALLUFF BNI006A – AOI QSG

Name:

Description:

Members:
Data Type Size: 328 byte(s)

| | Name | Data Type | Style | Description | External Access |
|------------|----------------------------|-----------------------------|---------|-------------|-----------------|
| | P | UDT_P2M2HBVL_Parameter_Data | | | Read/Write |
| | M | UDT_P2M2HBVL_MSG_Data | | | Read/Write |
| | Read_P2M_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 |
| 100 010 | | | | | |